SUPERVISED SPARSE LEARNING WITH APPLICATIONS IN BIOINFORMATICS

by

KIN MING PUK

Presented to the Faculty of the Graduate School of
The University of Texas at Arlington in Partial Fulfillment
of the Requirements
for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT ARLINGTON

August 2018

ACKNOWLEDGEMENTS

To begin with, I would like to express my deepest appreciation to my committee chairperson, Dr. Shouyi Wang, for his financial support, Matlab expertise, patience and guidance over the course of my doctoral studies.

Thank you, Dr. Jay M. Rosenberger, for inspiring me to think in a mathematical and optimization perspective, helping me get calm in the weekly research meeting, thoroughly scrutinizing my manuscripts every time, and, most importantly, being my co-advisor.

Thank you, Dr. Victoria C.P. Chen, for giving me a good overview on statistical analysis, helping me identify my research advisors, co-coordinating my duties of teaching assistant, and your constructive comment on my research reasoning and presentation.

A thank-you is long overdue to Dr. Jung-Chih Chiao for initiating and leading the emotion recognition project. It is the EEG data collected from this emotion recognition project which makes this dissertation possible. Thank you, Dr. Chiao, for being a committee member for my dissertation as well.

My computational experiments were not able to run smoothly without Richard Zercher Jr., who taught me a lot of useful tricks on Linux configuration and maintenance.

Thank you, Julie Estill, for your chocolate and sharing – the good, the bad, and the ugly – with me at Woolf 420. I am sure you enjoy every moment of your retirement.

Last but not least, this work was funded in part by NSF Grant #1537504 and NSF Grant CMMI #1434401. Data collection of emotion recognition was funded by UTA Interdisciplinary Research Program.

I thank my mum, my dad, Tracy and Sausage for the rest.

August 21, 2018

ABSTRACT

SUPERVISED SPARSE LEARNING WITH APPLICATIONS IN BIOINFORMATICS

Kin Ming Puk, Ph.D.

The University of Texas at Arlington, 2018

Supervising Professor: Shouyi Wang

In machine learning and mathematical optimization, sparse learning is the use of mathematical norms such as L1-norm, group norm and L21-norm in order to seek a trade-off between the goodness-of-fit measure and sparsity of the result. Sparsity of result leads to a parsimonious learning model - in other words, only few features from the data matrix are required to build the learning model and for further interpretation. The motivations of employing sparse learning in bioinformatics are two-fold: firstly, a parsimonious learning model enhances the explanatory power; and secondly, a parsimonious model generally allows better prediction and generalizes better to new data.

This dissertation is a collection of recent advances of sparse learning in bioinformatics, and consists of 1) L21-regularized multi-target support vector regression (L21-MSVR), 2) the application of L21-MSVR in predicting optimal tibial soft-tissue insertion of the human knees, 3) hierarchical sparse group lasso (HSGL), which improves the hierarchical lasso by incorporating an extra group-norm regularization, and 4) the use of HSGL on an electroencephalography (EEG)-based emotion recognition

problem. The commonality between these articles is the use of mathematical norms, and improvement from existing optimization formulations in order to learn better and to allow a better interpretation of feature selection.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Machine learning [1] can be loosely defined as the use of computer systems to learn from data without being explicitly programmed in order to perform specific tasks. As an example, in figure 1.1, linear regression [2] attempts to model the linear trend of the blue points with the red line. This is the classical case of simple linear regression, where both the dependent and independent variables are one-dimensional. The equation of linear regression is given by:

$$\boldsymbol{y} = \boldsymbol{A}\mathbf{w} + \boldsymbol{\epsilon} \tag{1.1}$$

where the notations of the term are explained in table 1.1.

In order to solve (1.1), mathematical optimization is one possible way (it can also be solved via closed-form solution using normal equation, QR matrix decomposition and singular value decomposition [4]). The formulation of least-square optimization is as follows:

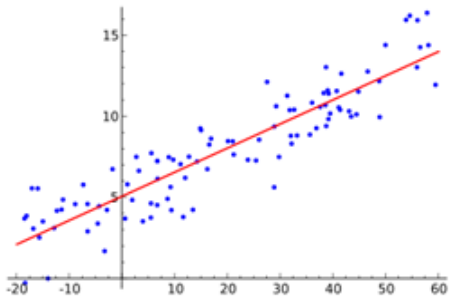


Figure 1.1: An illustration of linear regression [3].

Table 1.1: Notations used in chapter 1.

Symbol	Meaning
N	No. of observation
M	No. of features
$\boldsymbol{A} \in \mathbb{R}^{N \times M}$	Data matrix (Arranged as observation by feature)
$\boldsymbol{y} \in \mathbb{R}^{N \times 1}$	Single learning response
$\boldsymbol{Y} \in \mathbb{R}^{N \times P}$	Multi learning response
$\mathbf{w} \in \mathbb{R}^{M \times 1}$	Model vector
$\mathbf{W} \in \mathbb{R}^{M \times P}$	Model matrix
$\epsilon \in \mathbb{R}$	Residual

$$\min_{\mathbf{w}} \ ||\boldsymbol{A}\mathbf{w} - \boldsymbol{y}||_2 \tag{1.2}$$

However, the solution obtained from (1.2) is not necessarily unique, especially when the number of observations is less than that of features. In that case, an extra L_1-regularization on the model vector can be added, resulting in lasso (least absolute shrinkage and selection operator) [5]:

$$\min_{\mathbf{w}} \ ||\boldsymbol{A}\mathbf{w} - \boldsymbol{y}||_2 + \lambda||\mathbf{w}||_1 \tag{1.3}$$

Lasso is one of the classical examples in sparse learning, where the use of L_1-norm encourages the model vector $\mathbf{w}$ to be sparse. If λ is large, then the optimal $\mathbf{w}$ has only few non-zero elements (i.e. sparse) in the vector, and thus an appropriately tuned lasso model allows to select the few but yet important features that are the best predictors of the response [6].

If there is more than one response variable to predict simultaneously, then the formulation (1.2) can be generalized in matrix term, resulting in multi-response (or multi-target) regression [7], as follows:

$$\min_{\boldsymbol{W}} \ ||\boldsymbol{A}\mathbf{W} - \boldsymbol{Y}||_{2,2} + \lambda||\mathbf{W}||_{1,1} \tag{1.4}$$

In general, sparse (machine) learning [6] is a collection of learning methods in order to seek a trade-off between the goodness-of-fit measure (i.e. how well a learning model can describe a set of observations [8]) and sparsity of the model vector [6]. Mathematical norms such as L_1-norm ($||\mathbf{w}||_1 = \sum_i |\mathbf{w}_i|$), group norm ($||\mathbf{w}_k||_2 = \sqrt{\mathbf{w}_{k1}^2 + \cdots + \mathbf{w}_{kG_k}^2}$) and $L_{2,1}$-norm ($||\mathbf{W}||_{2,1} = \sum_{m=1}^{M} \sqrt{\sum_{p=1}^{P} \mathbf{w}_{mp}^2} = \sum_{m=1}^{M} ||\mathbf{w}_m||_2$) are popular choice in sparse learning. They are better known as sparsity-promotion norms [9] because:

- The presence of L_1-norm on vector $\mathbf{w}$ in a minimization optimization will force the sum of elements in $\mathbf{w}$ as small as possible, thus encouraging sparsity. For example, consider the following simple optimization example:

$$\begin{aligned} &\min_{\mathbf{w}} \ ||\mathbf{w}||_1 \\ &\text{s.t. } \mathbf{w} \geq \mathbf{0} \end{aligned} \tag{1.5}$$

  In this case (1.5), the optimum solution is $\mathbf{w} = \mathbf{0}$, with the objective value being zero. So the model vector $\mathbf{w}$ is very sparse, with all elements in the model vector being zero.

- In the case of group norm, the entire vector $\mathbf{w}$ will be divided into groups such as $\mathbf{w} = [\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_G]^T$, where there are G groups in this case. Group norm is basically to apply a L_2-norm on each of the group vector, resulting in $\sum_{g=1}^{G} ||\mathbf{w}_g||_2$. L_2-norm processes the "all-in-all-out" property [10] (where all other variables in the same group tend to be selected when only one variable is selected in that group). Consider the following numerical example,

$$\begin{aligned} \min_{\mathbf{w}} \ & ||\mathbf{w}||_2 \\ \text{s.t.} \ & \mathbf{w} = [w_1, w_2, w_3]^T \\ & w_1 + w_2 + w_3 > 1 \\ & w_1, w_2, w_3 > 0 \end{aligned} \tag{1.6}$$

It is assumed that the above selection problem requires at least one of the elements to be chosen in the above optimization problem (1.6), and that is why the constraint "$w_1 + w_2 + w_3 > 1$" is included. The optimum solution for (1.6) is $\mathbf{w} = [0.33, 0.33, 0.33]^T$, meaning that all three features are selected (all-in). If the "$w_1 + w_2 + w_3 > 1$" is removed from (1.6), then $\mathbf{w} = [0, 0, 0]^T$ (all-out). That explains the so-called "all-in-all-out" property.

Because of the "all-in-all-out" property of L_2-norm , either the variables of the whole group will be selected or ignored. Thus group norm encourages group sparsity on the model vector.

- $L_{2,1}$-norm is row-sparsity inducing (i.e. elements in a row are all zero). Similar to group norm, $L_{2,1}$-norm is to apply L_2-norm on each row vector of the model matrix. This explains why features on the entire row vector are selected or ignored, which is similar to group norm.

In view of the above, the motivation of this dissertation is the use of mathematical norms, and improvement from existing optimization formulations in order to learn better and to allow a better interpretation of feature selection. Chapter 2 proposes L21-regularized multi-target support vector regression (L21-MSVR). Starting from single-target support vector regression, it introduces how L21-MSVR is developed and the optimization algorithm used to solve it. Chapter 3 is the application of L21-MSVR in predicting optimal tibial soft-tissue insertion of the human knees. L21-MSVR is indeed developed in order to deal with the regression problem where there are

more features than observations and when there is a considerable number of response variables. Chapter 4 proposes hierarchical sparse group lasso (HSGL), which improves the hierarchical lasso by incorporating an extra group-norm regularization. Last but not least, chapter 5 is the use of HSGL on an EEG-based emotion recognition problem. Chapter 6 concludes this dissertation.

Bibliography

[1] Arthur L Samuel. Some studies in machine learning using the game of checkers. *IBM Journal of research and development*, 3(3):210–229, 1959.

[2] John Neter, Michael H Kutner, Christopher J Nachtsheim, and William Wasserman. *Applied linear statistical models*, volume 4. Irwin Chicago, 1996.

[3] Wikipedia, 2018.

[4] Gilbert Strang, Gilbert Strang, Gilbert Strang, and Gilbert Strang. *Introduction to linear algebra*, volume 3. Wellesley-Cambridge Press Wellesley, MA, 1993.

[5] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

[6] Laurent El Ghaoui, Guan-Cheng Li, Viet-An Duong, Vu Pham, Ashok N Srivastava, and Kanishka Bhaduri. Sparse machine learning methods for understanding large text corpora. In *CIDU*, pages 159–173, 2011.

[7] Hanen Borchani, Gherardo Varando, Concha Bielza, and Pedro Larrañaga. A survey on multi-output regression. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 5(5):216–233, 2015.

[8] GM Tallis. Goodness of fit. *Wiley Encyclopedia of Clinical Trials*, pages 1–10, 2007.

[9] Francis Bach, Rodolphe Jenatton, Julien Mairal, Guillaume Obozinski, et al. Optimization with sparsity-inducing penalties. *Foundations and Trends® in Machine Learning*, 4(1):1–106, 2012.

[10] Nengfeng Zhou and Ji Zhu. Group variable selection via a hierarchical lasso and its oracle property. *arXiv preprint arXiv:1006.2871*, 2010.

CHAPTER 2

L2,1-Regularized Multi-Response Support Vector Regression (L21-MSVR)

Abstract

A novel L2,1-norm based multi-response support vector regression method is proposed. The L2,1-norm based loss function is especially effective in selecting common features relevant to learning all the response variables at the same time. The unconstrained optimization problem can be efficiently solved by an iterative FISTA optimization algorithm. Simulated data is used to evaluate the robustness and effectiveness of the new proposed algorithm, and the result show promising performance when there are more features than observations, and when there are a considerable number of response variables.

Keywords: support vector regression, multi-response regression, L2,1-norm, FISTA

2.1 Introduction

The L2,1-regularized multi-response support vector regression was developed in order to better handle the prediction of optimal surgery location of human knees in the next chapter.

Predicting several responses simultaneously requires multi-target prediction models, also known as multi-response or multi-output learning. According to [1, 2], there are mainly two types of multi-response learning approaches:

- Problem transformation methods transform multi-target regression into other well-established, single-target learning models. One drawback of this type of

Table 2.1: Notations used in chapter 2.

Symbol	Meaning
N	No. of observation
M	No. of features
P	No. of response variables
$\boldsymbol{A} \in \mathbb{R}^{N \times M}$	Data matrix (Arranged as observation by feature)
$\boldsymbol{Y} \in \mathbb{R}^{N \times P}$	Learning response (or learning target)
$\mathbf{w} \in \mathbb{R}^{M \times 1}$	Model vector
$\mathbf{W} \in \mathbb{R}^{M \times P}$	Model matrix
$\epsilon \in \mathbb{R}$	Tuning parameter for L21-MSVR
$\boldsymbol{\xi}, \boldsymbol{\xi}^* \in \mathbb{R}^{N \times P}$	Slack variables for L21-MSVR

methods is that the relationships among the targets are often ignored so that the prediction performance might not be satisfactory. Examples include multi-target regressor stacking [3], which uses the prediction of the first response as part of the input data in predicting the next response variable, and multi-output support vector regression (SVR) [4], which considers the correlations between the targets using a vector virtualization method.

- Algorithm adaption methods learn by considering all dependencies and internal relationships between response variables. This type of methods is easier to interpret and produces better learning performance when response variables are correlated. Examples include statistical methods [5], multi-output SVR [6, 7], kernel methods [8], regression trees [9] and more. Interested readers can refer to [1] for a list of the state-of-the-art algorithms in this category.

2.2 Method

The proposed regression model[1] is an algorithm adaption method, which considers all the response variables at the same time during learning. Recall the formulation of L_2-regularized, L_2-loss support vector regression (SVR) model, where N, M and P are the number of observations, features and response variables respectively, as follows:

$$\begin{aligned} \min_{\boldsymbol{w}_p, \boldsymbol{\xi}, \boldsymbol{\xi}^*} \quad & \sum_{p=1}^{P} \left[||\boldsymbol{w}_p||_2 + C \sum_{n=1}^{N} (\xi_{pn})^2 + (\xi_{pn}^*)^2 \right] \\ \text{s.t.} \quad & y_{pn} - \boldsymbol{w}_p^T \boldsymbol{x_n} \leq \epsilon + \xi_{pn}, \forall p = 1, \ldots, P, n = 1, \ldots, N \\ & \boldsymbol{w}_p^T \boldsymbol{x_n} - y_{pn} \leq \epsilon + \xi_{pn}^*, \forall p = 1, \ldots, P, n = 1, \ldots, N \\ & \boldsymbol{\xi}, \boldsymbol{\xi^*} \geq \mathbf{0} \end{aligned} \tag{2.1}$$

$$f_p(\boldsymbol{x}) = \boldsymbol{w}_p^T \boldsymbol{x} + b, \ \ \forall p = 1, \ldots, P \tag{2.2}$$

The bias term b is eliminated for simplifying the presentation. It can be achieved by adding a column of one into the data matrix (or design matrix). Furthermore, the above can be generalized in matrix terms as follows, where $\mathbf{X} \in \mathbb{R}^{N \times M}$, $\mathbf{Y} \in \mathbb{R}^{N \times P}$, $\mathbf{W} \in \mathbb{R}^{M \times P}$, $\boldsymbol{\xi}, \boldsymbol{\xi}^*, \mathbf{1} \in \mathbb{R}^{N \times P}$ and $||\cdot||_{\mathrm{F}}$ is the Frobenius Norm:

$$\begin{aligned} \min_{\mathbf{W}, \boldsymbol{\xi}, \boldsymbol{\xi}^*} \quad & ||\mathbf{W}||_{\mathrm{F}} + C||\boldsymbol{\xi}||_{\mathrm{F}} + C||\boldsymbol{\xi}^*||_{\mathrm{F}} \\ \text{s.t.} \quad & \mathbf{Y} - \mathbf{XW} \leq \epsilon \mathbf{1} + \boldsymbol{\xi} \\ & \mathbf{XW} - \mathbf{Y} \leq \epsilon \mathbf{1} + \boldsymbol{\xi}^* \\ & \boldsymbol{\xi}, \boldsymbol{\xi}^* \geq 0 \end{aligned} \tag{2.3}$$

[1] In the model, a scalar is represented by a non-bold lower-case letter, a vector is represented by a bold lower-case letter and a matrix is represented by a bold upper-case letter.

$$f(\mathbf{X}) = \mathbf{XW} \tag{2.4}$$

The unconstrained version of formulation (2.3) is:

$$\begin{aligned} \min_{\mathbf{W}} \; J(\mathbf{W}) = ||\mathbf{W}||_{\mathrm{F}} + C||(\mathbf{Y} - \mathbf{XW} - \epsilon\mathbf{1})_+||_{\mathrm{F}} \\ + C||(\mathbf{XW} - \mathbf{Y} - \epsilon\mathbf{1})_+||_{\mathrm{F}} \end{aligned} \tag{2.5}$$

where the hinge loss operator is $(\cdot)_+ = \max(\cdot, 0)$. To further modify the formulation, consider the $L_{2,1}$-norm, which is defined as:

$$||\mathbf{W}||_{2,1} = \sum_{m=1}^{M} \sqrt{\sum_{p=1}^{P} w_{mp}^2} = \sum_{m=1}^{M} ||\mathbf{w}_m||_2 \tag{2.6}$$

where $\mathbf{w}_m$ denotes the m-th row vector of matrix $\mathbf{W}$. The initial formulation of $L_{2,1}$-regularized, $L_{2,2}$-loss multi-response support vector regression (L21-MSVR), is proposed:

$$\min_{\mathbf{W}} \; J_1(\mathbf{W}) = ||\mathbf{W}||_{2,1} + C||(\mathbf{Y} - \mathbf{XW} - \epsilon\mathbf{1})_+||_{\mathrm{F}} + C||(\mathbf{XW} - \mathbf{Y} - \epsilon\mathbf{1})_+||_{\mathrm{F}} \tag{2.7}$$

As a heuristics to further enhance computational efficiency, the final version of L21-MSVR is reformulated from (2.5) with the removal of hinge loss and equality loss constraints as follows [10]:

$$\begin{aligned} \min_{\mathbf{W},\boldsymbol{\xi},\boldsymbol{\xi}^*} \quad & ||\mathbf{W}||_{2,1} + C||\boldsymbol{\xi}||_{\mathrm{F}} + C||\boldsymbol{\xi}^*||_{\mathrm{F}} \\ \text{s.t.} \quad & \mathbf{Y} - \mathbf{XW} = \epsilon\mathbf{1} + \boldsymbol{\xi} \\ & \mathbf{XW} - \mathbf{Y} = \epsilon\mathbf{1} + \boldsymbol{\xi}^* \end{aligned} \tag{2.8}$$

$$\min_{\mathbf{W}} \; J_2(\mathbf{W}) = ||\mathbf{W}||_{2,1} + C||\mathbf{Y} - \mathbf{XW} - \epsilon\mathbf{1}||_{\mathrm{F}} + C||\mathbf{XW} - \mathbf{Y} - \epsilon\mathbf{1}||_{\mathrm{F}} \tag{2.9}$$

The reformulation of the slack variables $\boldsymbol{\xi}$ and $\boldsymbol{\xi}^*$ removes the hinge-loss operator so that the gradient of the Frobenius loss term can be computed directly instead of computing its subgradient. The derivative of formulation (2.15) is consequently as follows [11]:

$$\begin{aligned}\frac{\partial J_2(\mathbf{W})}{\partial \mathbf{W}} &= \frac{\partial}{\partial \mathbf{W}} \ ||\mathbf{W}||_{2,1} \\ &\quad + C\frac{\partial}{\partial \mathbf{W}}||\mathbf{Y} - \mathbf{X}\mathbf{W} - \epsilon\mathbf{1}||_{\mathrm{F}} \\ &\quad + C\frac{\partial}{\partial \mathbf{W}}||\mathbf{X}\mathbf{W} - \mathbf{Y} - \epsilon\mathbf{1}||_{\mathrm{F}} \\ &= \frac{\partial}{\partial \mathbf{W}} \ \mathrm{tr}(\mathbf{W}^T\mathbf{D}\mathbf{W}) + C(4\mathbf{X}^T\mathbf{X}\mathbf{W} - 4\mathbf{X}^T\mathbf{Y}) \\ &= 2\mathbf{D}\mathbf{W} + 4C\mathbf{X}^T\mathbf{X}\mathbf{W} - 4C\mathbf{X}^T\mathbf{Y}\end{aligned} \tag{2.10}$$

One may be tempted to develop analytical solution as in [10]. From derivative (2.10), it is found that $\mathbf{W} = (2\mathbf{D} + 4C\mathbf{X}^T\mathbf{X})^{-1}4C\mathbf{X}^T\mathbf{Y}$. However, as pointed out in [12], the matrix $\mathbf{D}$ is dependent on $\mathbf{W}$ and is unknown. Therefore, an iterative optimization algorithm will be more appropriate for computation.

2.2.0.1 Optimization Algorithm - FISTA

Formulation (2.7) can be easily solved with a commercial convex programming package such as CVX [13]. However, doing so may take more than a week in order to tune the parameters and get an optimal solution. In order to enhance computational efficiency, the FISTA (Fast ISTA) algorithm [14], which combines Nesterov's accelerated gradient descent with ISTA (iterative shrinkage-threshold algorithm), is chosen to efficiently solve formulation (2.7) because i) the formulation is in the form of standard, unconstrained quadratic programming; ii) a subgradient can be computed for the objective function consisting of $L_{2,2}$-norm and $L_{2,1}$-norm; iii) FISTA is easy to implement [15]; and iv) instead of having a sublinear convergence rate of $O(1/j)$ as in

gradient descent, FISTA enjoys a favorable convergence rate of $O(1/j^2)$, where j is the number of iterations.

A a summary of FISTA can be found in Algorithm (1). Its convergence is well analyzed and can be found in [15].

2.2.0.2 Implementation Specifics of the Algorithm

When $\mathbf{w}_m = 0$, then $d_{mm} = 0$ is a subgradient of $||\mathbf{W}||_{2,1}$, w.r.t. $\mathbf{w}_m$ [12]. However, if d_{mm} is set to 0 when $\mathbf{w}_m = 0$, the derived algorithm may not converge (i.e. $d_{mm} = \frac{1}{0} = \text{inf}$). Therefore, a small number ρ is used for regularization as in $d_{mm} = \frac{1}{2\sqrt{(\mathbf{w}_m)^T \mathbf{w}_m + \rho}}$.

There are a lot of choices for termination criteria such as the difference of objective values between successive iterations. For better computational efficiency, the ratio of the Frobenius Norms $\left(\frac{||\mathbf{W}^{(j+1)} - \mathbf{W}^{(j)}||_{\text{F}}}{||\mathbf{W}^{(j)}||_{\text{F}}}\right)$ is used at [12] in order to avoid the expensive computation of the objective value.

In practice, the starting step size γ and the parameter σ in backtracking ($\gamma = \sigma\gamma$) are set to 1 and 0.8 respectively [16].

2.3 Simulation Study

2.3.1 Data Generation

The simulation in [17, 18] was extended. A model that has both categorical and continuous prediction variables was considered.

To begin with, the number of observations and features in the simulated dataset is either 35 or 350. For the case of 35 features (or 350 features), ten (or one hundred (100)) independent standard normal variables, $Z_1, \ldots, Z_{10}$ (or $Z_1, \ldots, Z_{100}$) and W were generated. The covariates were then defined as $X_j = (Z_j + W)/\sqrt{2}$. Then the last

Algorithm 1 Solving multi-response support vector regression with $L_{2,1}$-norm regularization [12, 15]

Require: $\mathbf{X} \in \mathbb{R}^{N \times M}$ (data matrix), $\mathbf{Y} \in \mathbb{R}^{N \times P}$ (response matrix), $C, \epsilon, \alpha \in \mathbb{R}$ (learning parameters), $\rho \in \mathbb{R}$ (regularization parameter), $\sigma \in \mathbb{R}$ (backtracking parameter), tol $\in \mathbb{R}$ (termination parameter).

1: Initialize projection matrix $\mathbf{W}^{(0)} = \{w_{mp} = 0\}, \forall m, p$, $\mathbf{Z}^{(0)} = \mathbf{W}^{(0)}$, $t^{(0)} = 1$.

2: Set $j = 0$.

3: **while** termination criterion is not met (tol $> \frac{||\mathbf{W}^{(j+1)} - \mathbf{W}^{(j)}||_{\mathrm{F}}}{||\mathbf{W}^{(j)}||_{\mathrm{F}}}$) **do**

4: Initialize diagonal matrix $\mathbf{D}^{(j+1)}$, where the m-th diagonal element is $d_{mm}^{(j+1)} = \frac{1}{2||\mathbf{w}_m^{(j)}||_2}$ (or $d_{mm}^{(j+1)} = \frac{1}{2\sqrt{(\mathbf{w}_m^{(j)})^T \mathbf{w}_m^{(j)} + \rho}}$ if $||\mathbf{w}_m^{(j)}||_2 = 0$).

5: $t^{(j+1)} = \frac{1}{2} + \frac{1}{2}\sqrt{1 + 4(t^{(j)})^2}$.

6: $\mathbf{W}^{(j+1)} = \mathbf{Z}^{(j)}$ - $\gamma \frac{\partial J_2(\mathbf{Z}^{(j)})}{\partial \mathbf{Z}}$ according to (2.10).

7: $\mathbf{Z}^{(j+1)} = \mathbf{W}^{(j+1)} + \frac{t^{(j)} - 1}{t^{(j+1)}}(\mathbf{W}^{(j+1)} - \mathbf{W}^{(j)})$.

8: Use backtracking to find the appropriate step size γ.

9: Set $j = j + 1$.

10: **end while**

five (or fifty (50)) covariates $X_6, \ldots, X_{10}$ (or $X_{51}, \ldots, X_{100}$) were discretized to 0, 1, 2, and 3 by $\Phi^{-1}(1/4)$, $\Phi^{-1}(1/2)$, and $\Phi^{-1}(3/4)$. Each of $X_1, \ldots, X_5$ (or $X_1, \ldots, X_{50}$) was expanded through a fourth-order polynomial, and only the main effects of $X_6, \ldots, X_{10}$ (or $X_{51}, \ldots, X_{100}$) were considered. Therefore a total of five (or fifty (50)) continuous groups with four variables in each group and five (or fifty (50)) categorical groups with three variables in each group were simulated.

Each response is generated by randomly selecting and summing up 20%, 50% or 80% of the available features among the dataset together with the error term. Each feature is weighted with a normalized coefficient $(a_1, \ldots, a_N)$ as follows:

$$\begin{aligned} \boldsymbol{y} &= a_1\boldsymbol{x}_1 + a_2\boldsymbol{x}_2 + \ldots + a_N\boldsymbol{x}_N + \epsilon \\ 1 &= a_1 + \ldots + a_N \end{aligned} \tag{2.11}$$

The error term ϵ in the above follows a normal distribution N(0, σ^2), where σ^2 was set such that each of the signal to noise ratios, Var($\boldsymbol{X}^T\boldsymbol{W}$)/Var($\epsilon$)), was equal to 3. For example, assume the data matrix $\boldsymbol{X}$ consists of 10 features, i.e. $\boldsymbol{X} = [\boldsymbol{x}_1^T, \ldots, \boldsymbol{x}_{10}^T]^T$. Also assume the ratio of features used to construct each response is 20%, then 2 out of 10 features will be used to construct the response as follows: $\boldsymbol{y} = a_i\boldsymbol{x}_i + a_j\boldsymbol{x}_j + \epsilon$, where $i, j \in 1, \ldots, 10$ and ϵ is the error term.

In summary, there are 30 cases as follows:

1. Number of observations and features: i) 35 observations and 350 features, ii) 350 observations and 35 features
2. Number of responses: 2, 5, 10, 15, 20
3. Percentage of available features used to construct each response: 20%, 50%, 80%

In addition, there are two performance measures:

$$\begin{aligned} \text{aRRMSE} &= \frac{1}{P}\sum_{p=1}^{P}\sqrt{\frac{\sum_{n=1}^{N_{test}}(y_{pn} - \hat{y}_{pn})^2}{\sum_{n=1}^{N_{test}}(y_{pn} - \bar{y}_p)^2}}, \\ \text{Feat. Density} &= \frac{\text{No. of non-zero elements in feat. matrix}}{\text{Row size of feat. matrix} * \text{Column size of feat. matrix}} \end{aligned} \tag{2.12}$$

A double 5-fold cross validation was conducted. Parameters, if any, were tuned using the training dataset and five-fold cross validation to provide a realistic estimation of prediction errors and to prevent over-fitting. After running all possible combinations of parameters, those with the highest accuracy or lowest average relative root mean squared error (aRRMSE) was chosen. The above procedure was repeated 30 times.

2.3.2 Baseline Regressors

The following methods in the literature were used to compare with the proposed method. Notations in the following models have been changed for better consistency with the above.

- Generalized linear model via penalized maximum likelihood (GLMNET, [19, 20]): GLMNET is an extension of the single-target case where it is designed to model correlated responses with a multi-response Gaussian distribution. The absolute penalty on each single coefficient is replaced with a group-lasso penalty.

$$\begin{aligned} \min_{(\mathbf{W}_0,\mathbf{W})\in\mathbb{R}^{(M+1)\times P}} \quad & \frac{1}{2N}\sum_{n=1}^{N}||\mathbf{y}_n - \mathbf{W}_0^T - \ \mathbf{W}^T\mathbf{x}_n||_F^2 \\ & + \lambda\left[\ \frac{1}{2}(1-\alpha)||\mathbf{W}||_F^2 + \alpha\sum_{m=1}^{M}||\mathbf{w}_m||_2\ \right] \end{aligned} \tag{2.13}$$

- Multi-target regressor stacking using decision tree (MTRS-DT, [21]): MTRS-DT is inspired from "stacked generalization" which was designed for multi-label classification. There are two stages in the learning process. Firstly, P single-target models are learned as usual. MTRS-DT then predicts the target one-by-one with stacking: after predicting the first response variable, its output will be attached to the original testing dataset to build a new meta model, which is in turned used to predict the second target variable. The process continues until all response variables are predicted in this manner [1, 21]. The stacking idea is powerful and can be applied to any single-target regression model. However, the sequence of stacking affects the regression performance. In this experiment, the prediction of the first response variable will be used in the prediction of the next.
- Multi-target random forest (MTRF, [22, 23, 24, 25]): A random forest is a meta-learner that trains a number of decision trees, each of which uses a subset of

the original dataset and outputs the final prediction by averaging the prediction from each decision tree. Generally speaking, MTRF differs from its single-target counterpart in that during the splitting of a tree node, the performance measure considers all the output variables instead of considering a single output variable at a time.

- Single-response support vector regression (SSVR): It is simply the single-response L2-regularized, L2-loss support vector regression from Liblinear.

2.3.3 Result

The result are presented in tables 2.2, 2.3 and 2.4. In general, the strongest competitor of L21-MSVR is GLMNET, which performs generally well across different cases. L21-MSVR performs the best when there are more features than observations.

To begin with, L21-MSVR does not perform well in terms of aRRMSE when only 20% of features are used to form the responses, especially when there are more observations than features. Recall the property that L2,1-norm is to sum the L2-norm of each row vector in the model matrix. Considering the following hypothetic cases using only 20% of the available 10 features to form the responses:

$$\begin{aligned} \boldsymbol{y}_1 &= \boldsymbol{x}_1 + \boldsymbol{x}_2 \\ \boldsymbol{y}_2 &= \boldsymbol{x}_3 + \boldsymbol{x}_4 \end{aligned} \tag{2.14}$$

In the above case, the first response $\boldsymbol{y}_1$ requires the 1st and 2nd features, whereas the second response $\boldsymbol{y}_2$ requires the 3rd and 4th features. For L21-MSVR, because of the "all-in-all-out" features of L2-norm, even though the 3rd and 4th features are not responsible for learning the first response variable, they will be all selected for L21-MSVR. The same applies to the 1st and 2nd features in learning the second response variable for L21-MSVR. As a result, under cases where response variables

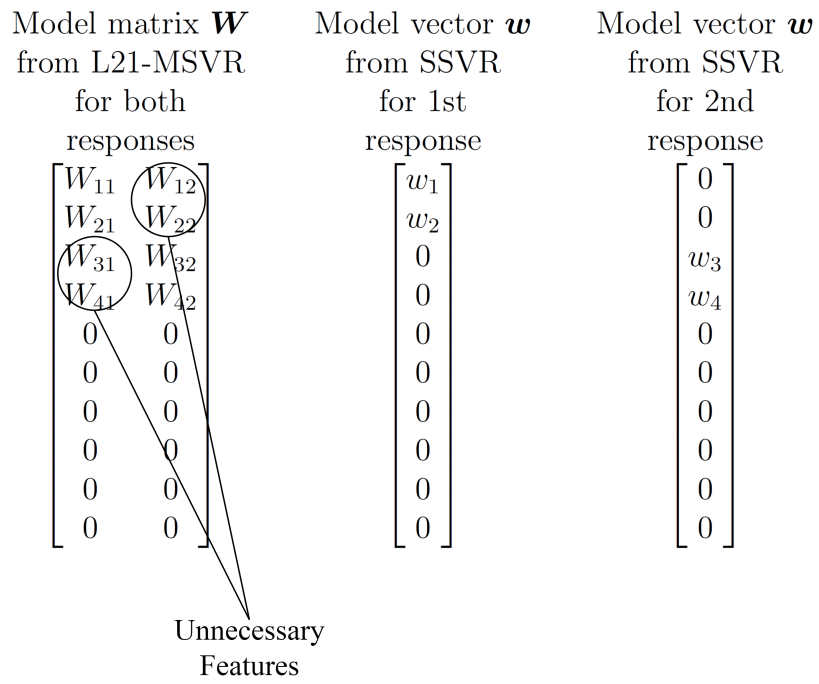


Figure 2.1: Model matrix and vectors as solved by L21-MSVR and SSVR in the hypothetical case. In the following, non-zero elements in the vector/matrix are represented by a variable such as w_1 and $W_1 1$. This figure is associated with the hypothetical case (2.14).

do not have a lot of correlation between each other such as this case of using 20% of available features in forming each response variable, L21-MSVR tends not to learn well and its model matrix $\boldsymbol{W}$ tends to have high density because "unnecessary features" tend to be selected, as illustrated in figure 2.1.

If the percentage of available features used to construct each response is increased to 50% and 80%, L21-MSVR performs well in the case of having more features than observations, as illustrated in Tables 2.3 and 2.4. Following the same rationale in the above, the number of "unnecessary features" to be included to learn the features gets relatively smaller when compared with the case of using 20%, as the same feature is probably more relevant to learning several responses at the same time.

Table 2.2: aRRMSE density of feature matrix (or vector), if any, of simulation experiments of L21-MSVR against other baseline regressors using 20% of the available features to generate each response. All of the numbers outside of parentheses are means over 30 repetitions, and the numbers in the parentheses are the corresponding standard errors.

			aRRMSE					Density of Feature Vector/Matrix				
No. of Obsns.	No. of Feat.	No. of Resp.	GLM-NET	L21-MSVR	MTRS-DT	MTRF	SSVR	GLM-NET	L21-MSVR	MTRS-DT	MTRF	SSVR
35	350	2	**0.675 (0.211)**	0.718 (0.235)	1.08 (0.146)	1.074 (0.173)	1.276 (0.212)	0.106 (0.01)	0.381 (0.162)	N/A	N/A	0.102 (0.092)
35	350	5	**0.633 (0.151)**	0.729 (0.192)	1.091 (0.126)	1.061 (0.165)	1.296 (0.133)	0.17 (0.015)	0.457 (0.135)	N/A	N/A	0.102 (0.102)
35	350	10	**0.71 (0.139)**	0.788 (0.217)	1.073 (0.078)	1.087 (0.132)	1.304 (0.145)	0.21 (0.017)	0.523 (0.11)	N/A	N/A	0.062 (0.065)
35	350	15	**0.681 (0.114)**	0.71 (0.158)	1.097 (0.084)	1.04 (0.14)	1.349 (0.15)	0.229 (0.025)	0.541 (0.148)	N/A	N/A	0.096 (0.079)
35	350	20	**0.699 (0.1)**	0.763 (0.16)	1.093 (0.092)	1.077 (0.111)	1.27 (0.107)	0.247 (0.026)	0.519 (0.151)	N/A	N/A	0.077 (0.061)
350	35	2	**0.072 (0.065)**	0.476 (0.4)	0.309 (0.066)	0.718 (0.263)	0.656 (0.256)	0.096 (0.02)	0.769 (0.11)	N/A	N/A	0.583 (0.238)
350	35	5	**0.074 (0.048)**	0.341 (0.324)	0.312 (0.064)	0.769 (0.233)	0.688 (0.138)	0.173 (0.029)	0.717 (0.149)	N/A	N/A	0.6 (0.241)
350	35	10	**0.077 (0.035)**	0.366 (0.268)	0.328 (0.073)	0.775 (0.177)	0.735 (0.11)	0.225 (0.042)	0.732 (0.137)	N/A	N/A	0.548 (0.19)
350	35	15	**0.097 (0.062)**	0.281 (0.271)	0.361 (0.117)	0.715 (0.207)	0.716 (0.086)	0.242 (0.033)	0.724 (0.138)	N/A	N/A	0.572 (0.224)
350	35	20	**0.087 (0.043)**	0.227 (0.254)	0.329 (0.076)	0.729 (0.212)	0.729 (0.102)	0.241 (0.028)	0.678 (0.138)	N/A	N/A	0.552 (0.248)

Table 2.3: aRRMSE density of feature matrix (or vector), if any, of simulation experiments of L21-MSVR against other baseline regressors using 50% of the available features to generate each response. All of the numbers outside of parentheses are means over 30 repetitions, and the numbers in the parentheses are the corresponding standard errors.

			aRRMSE					Density of Feature Vector/Matrix				
No. of Obsns.	No. of Feat.	No. of Resp.	GLM-NET	L21-MSVR	MTRS-DT	MTRF	SSVR	GLM-NET	L21-MSVR	MTRS-DT	MTRF	SSVR
35	350	2	0.965 (0.718)	**0.652 (0.215)**	1.085 (0.21)	1.197 (0.18)	1.323 (0.172)	0.108 (0.006)	0.436 (0.12)	N/A	N/A	0.042 (0.064)
35	350	5	0.745 (0.174)	**0.653 (0.188)**	1.084 (0.143)	1.131 (0.2)	1.322 (0.227)	0.158 (0.019)	0.447 (0.14)	N/A	N/A	0.062 (0.079)
35	350	10	0.753 (0.214)	**0.633 (0.167)**	1.107 (0.118)	1.188 (0.182)	1.324 (0.176)	0.191 (0.016)	0.465 (0.112)	N/A	N/A	0.047 (0.062)
35	350	15	0.735 (0.158)	**0.684 (0.208)**	1.132 (0.096)	1.222 (0.177)	1.224 (0.214)	0.202 (0.016)	0.537 (0.112)	N/A	N/A	0.04 (0.045)
35	350	20	0.684 (0.163)	**0.67 (0.157)**	1.105 (0.13)	1.176 (0.164)	1.284 (0.177)	0.207 (0.02)	0.49 (0.117)	N/A	N/A	0.042 (0.047)
350	35	2	**0.081 (0.043)**	0.249 (0.253)	0.432 (0.107)	0.829 (0.318)	0.642 (0.188)	0.211 (0.039)	0.695 (0.155)	N/A	N/A	0.439 (0.245)
350	35	5	**0.075 (0.047)**	0.267 (0.264)	0.446 (0.121)	0.724 (0.281)	0.694 (0.167)	0.251 (0.032)	0.684 (0.1)	N/A	N/A	0.524 (0.217)
350	35	10	**0.062 (0.024)**	0.292 (0.235)	0.416 (0.101)	0.728 (0.242)	0.711 (0.099)	0.267 (0.032)	0.741 (0.125)	N/A	N/A	0.534 (0.225)
350	35	15	**0.069 (0.033)**	0.236 (0.195)	0.415 (0.083)	0.765 (0.287)	0.716 (0.091)	0.254 (0.026)	0.663 (0.17)	N/A	N/A	0.434 (0.245)
350	35	20	**0.062 (0.023)**	0.346 (0.208)	0.426 (0.081)	0.684 (0.224)	0.69 (0.091)	0.264 (0.031)	0.766 (0.131)	N/A	N/A	0.561 (0.192)

Table 2.4: aRRMSE density of feature matrix (or vector), if any, of simulation experiments of L21-MSVR against other baseline regressors using 80% of the available features to generate each response. All of the numbers outside of parentheses are means over 30 repetitions, and the numbers in the parentheses are the corresponding standard errors.

			aRRMSE					Density of Feature Vector/Matrix				
No. of Obsns.	No. of Feat.	No. of Resp.	GLM-NET	L21-MSVR	MTRS-DT	MTRF	SSVR	GLM-NET	L21-MSVR	MTRS-DT	MTRF	SSVR
35	350	2	0.788 (0.171)	**0.591 (0.184)**	1.088 (0.197)	1.228 (0.238)	1.356 (0.169)	0.104 (0.006)	0.445 (0.134)	N/A	N/A	0.038 (0.046)
35	350	5	0.703 (0.173)	**0.51 (0.146)**	1.058 (0.168)	1.101 (0.275)	1.268 (0.24)	0.137 (0.013)	0.493 (0.118)	N/A	N/A	0.088 (0.093)
35	350	10	0.769 (0.16)	**0.607 (0.19)**	1.1 (0.153)	1.268 (0.258)	1.312 (0.299)	0.164 (0.02)	0.45 (0.106)	N/A	N/A	0.042 (0.062)
35	350	15	0.745 (0.183)	**0.644 (0.2)**	1.103 (0.149)	1.263 (0.29)	1.259 (0.195)	0.166 (0.025)	0.484 (0.11)	N/A	N/A	0.036 (0.052)
35	350	20	0.672 (0.151)	**0.605 (0.201)**	1.07 (0.162)	1.242 (0.241)	1.279 (0.254)	0.172 (0.025)	0.506 (0.109)	N/A	N/A	0.044 (0.058)
350	35	2	**0.06 (0.032)**	0.159 (0.088)	0.46 (0.124)	0.586 (0.296)	0.668 (0.174)	0.275 (0.038)	0.733 (0.131)	N/A	N/A	0.535 (0.23)
350	35	5	**0.07 (0.049)**	0.182 (0.13)	0.426 (0.083)	0.659 (0.323)	0.653 (0.119)	0.28 (0.019)	0.692 (0.151)	N/A	N/A	0.465 (0.254)
350	35	10	**0.047 (0.009)**	0.17 (0.183)	0.405 (0.073)	0.668 (0.302)	0.683 (0.151)	0.271 (0.033)	0.704 (0.131)	N/A	N/A	0.48 (0.226)
350	35	15	**0.05 (0.014)**	0.201 (0.138)	0.445 (0.085)	0.643 (0.278)	0.686 (0.141)	0.284 (0.017)	0.732 (0.14)	N/A	N/A	0.547 (0.197)
350	35	20	**0.049 (0.012)**	0.276 (0.165)	0.433 (0.099)	0.634 (0.257)	0.699 (0.149)	0.278 (0.019)	0.775 (0.086)	N/A	N/A	0.468 (0.255)

2.4 Concluding Remarks

This chapter presents a novel L2,1-regularized, L2,2-loss multi-response support vector regression, which handles the multi-response regression problem well when there are more features than observations. L21-MSVR learns especially well when the individual feature is relevant in learning most if not all of the response variables at the same time. This is probably not the usual case for most of the multi-response learning problems, where an individual feature is relevant to the learning of some (but not most) of the response variables only. However, as illustrated in the next chapter, L21-MSVR is particularly suitable for the prediction of tibial soft-tissue insertion of human knees, where the features are the surface points of the shape of a human knee so that each relevant feature is believed to be helpful in predicting the optimal insertion centroid of each and every of the tibial soft tissues simultaneously.

Future direction includes the incorporation of the extra L1,1-norm, in order to encourage the sparsity of each row in the model matrix, as follows:

$$\min_{\mathbf{W}} \; J_2(\mathbf{W}) = ||\mathbf{W}||_{2,1} + ||\mathbf{W}||_{1,1} + C||\mathbf{Y} - \mathbf{X}\mathbf{W} - \epsilon\mathbf{1}||_{\mathrm{F}} + C||\mathbf{X}\mathbf{W} - \mathbf{Y} - \epsilon\mathbf{1}||_{\mathrm{F}} \tag{2.15}$$

Bibliography

[1] Hanen Borchani, Gherardo Varando, Concha Bielza, and Pedro Larrañaga. A survey on multi-output regression. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 5(5):216–233, 2015.

[2] Min-Ling Zhang and Zhi-Hua Zhou. A review on multi-label learning algorithms. *IEEE transactions on knowledge and data engineering*, 26(8):1819–1837, 2014.

[3] Eleftherios Spyromitros-Xioufis, Grigorios Tsoumakas, William Groves, and Ioannis Vlahavas. Multi-label classification methods for multi-target regression. *arXiv preprint arXiv*, 1211, 2012.

[4] Wei Zhang, Xianhui Liu, Yi Ding, and Deming Shi. Multi-output ls-svr machine in extended feature space. In *Computational Intelligence for Measurement Systems and Applications (CIMSA), 2012 IEEE International Conference on*, pages 130–134. IEEE, 2012.

[5] Alan Julian Izenman. Reduced-rank regression for the multivariate linear model. *Journal of multivariate analysis*, 5(2):248–264, 1975.

[6] Feng Cai and Vladimir Cherkassky. Svm+ regression and multi-task learning. In *Neural Networks, 2009. IJCNN 2009. International Joint Conference on*, pages 418–424. IEEE, 2009.

[7] Matilde Sánchez-Fernández, Mario de Prado-Cumplido, Jerónimo Arenas-García, and Fernando Pérez-Cruz. Svm multiregression for nonlinear channel estimation in multiple-input multiple-output systems. *IEEE transactions on signal processing*, 52(8):2298–2307, 2004.

[8] Luca Baldassarre, Lorenzo Rosasco, Annalisa Barla, and Alessandro Verri. Multi-output learning via spectral filtering. *Machine learning*, 87(3):259–301, 2012.

[9] Glenn De'Ath. Multivariate regression trees: a new technique for modeling species–environment relationships. *Ecology*, 83(4):1105–1117, 2002.

[10] Johan AK Suykens and Joos Vandewalle. Least squares support vector machine classifiers. *Neural processing letters*, 9(3):293–300, 1999.

[11] K. B. Petersen and M. S. Pedersen. The matrix cookbook, nov 2012. Version 20121115.

[12] Xiao Cai, Feiping Nie, Heng Huang, and Chris Ding. Multi-class l2, 1-norm support vector machine. In *Data Mining (ICDM), 2011 IEEE 11th International Conference on*, pages 91–100. IEEE, 2011.

[13] Michael Grant and Stephen Boyd. CVX: Matlab software for disciplined convex programming, version 2.1. `http://cvxr.com/cvx`, March 2014.

[14] Sébastien Bubeck et al. Convex optimization: Algorithms and complexity. *Foundations and Trends® in Machine Learning*, 8(3-4):231–357, 2015.

[15] Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences*, 2(1):183–202, 2009.

[16] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[17] Nengfeng Zhou and Ji Zhu. Group variable selection via a hierarchical lasso and its oracle property. *arXiv preprint arXiv:1006.2871*, 2010.

[18] Sungwan Bang, Jongkyeong Kang, Myoungshic Jhun, and Eunkyung Kim. Hierarchically penalized support vector machine with grouped variables. *International Journal of Machine Learning and Cybernetics*, 8(4):1211–1221, 2017.

[19] Jerome Friedman, Trevor Hastie, and Rob Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of statistical software*, 33(1):1, 2010.

[20] Noah Simon, Jerome Friedman, and Trevor Hastie. A blockwise descent algorithm for group-penalized multiresponse and multinomial regression. *arXiv preprint arXiv:1311.6529*, 2013.

[21] Eleftherios Spyromitros-Xioufis, Grigorios Tsoumakas, William Groves, and Ioannis Vlahavas. Multi-target regression via input space expansion: treating targets as inputs. *Machine Learning*, 104(1):55–98, 2016.

[22] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[23] H Tim Kam. Random decision forest. In *Proc. of the 3rd Int'l Conf. on Document Analysis and Recognition, Montreal, Canada, August*, pages 14–18, 1995.

[24] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[25] Dragi Kocev, Celine Vens, Jan Struyf, and Sašo Džeroski. Ensembles of multi-objective decision trees. In *European conference on machine learning*, pages 624–631. Springer, 2007.

CHAPTER 3

An Efficient L2,1-Regularized and Structure-Supervised Multi-Response Prediction Model with Application to Tibial Soft Tissue Insertion Predictions

Abstract

The knee is the largest joint in the human body and is integral to movement. In the case of any tearing of the cruciate ligaments, it is essential to identify their original position for reconstruction and recovery. During surgical operation, it is difficult to locate ligaments intra-operatively and there is a high inter-person morphological variability between patients. Failing to do so will affect adjacent tissues, hindering the knee recovery. In a previous study, the feasibility of predicting optimal insertion sites of the cartilages and soft tissues under a reduced two-dimensional scenario was investigated. Such prediction problem is challenging because of the large number of features (508) and response variables (24), as opposed to the limited number of observations (20). Extended from the previous study, this research is one of the first few to precisely identify such positions under a three-dimensional setting. Because the tibia outline can be accurately and reliably measured from computed tomography (CT) imaging, features can be extracted for model building. In addition, a novel three-step structure-learning model using multi-response support vector regression and L2,1-norm regularization is employed so that the optimal insertion position of the cartilages and soft tissues can be predicted simultaneously. The proposed model is easy to implement and outperforms other state-of-the-art prediction models in the experiment.

Keywords: ligament, meniscal insertions, knee anatomic reconstruction, structured supervised learning, X-ray imaging, computed tomography, multi-response prediction, support vector regression, L2,1 regularization.

3.1 Introduction

The knee is a complex joint that provides basic human function and mobility, enabling ambulatory or sports activities such as walking, running, and jumping [1]. The structural components of a knee joint include cartilage, ligaments (connecting one bone to another), tendons (connecting muscle to bone) and bones (connected by ligaments, or by tendons to muscles). As a union of bones, the knee is where the thigh bone (femur) joins the shin bone (tibia) to form the tibiofemoral joint, and the kneecap (patella) joins the femur to form the patellofemoral joint. Articular cartilage covers the knee joint, helps absorb shock, and improves joint movement. Over-use and excessive stress or force can make the knee prone to a variety of injuries. Typical knee injuries include fractures, dislocations, anterior cruciate ligament (ACL) injuries, posterior cruciate ligament (PCL) injuries, collateral ligament injuries, meniscal tears and tendon tears [2, 3]. Injuries to the cruciate ligaments (ACL and PCL) are particularly devastating as they do not heal and would most often require surgical reconstruction. Among the two, the ACL is more commonly injured: an estimated 175,000 ACL reconstructions are performed annually, with a financial impact exceeding 2 billion dollars in the United States alone. On the other hand, PCL injuries occur less frequently and are thus believed to be under diagnosed; yet they affect about 3% of the general population and account for as many as 40% of patients with knee trauma seen in emergency rooms.

The anatomic reconstruction procedure involves creating bone tunnels and placing substituting grafts in the exact positions as the native ligaments. An ac-

Table 3.1: Notations used in chapter 3.

Symbol	Meaning
N	No. of observation
M	No. of features
P	No. of response variables
$\boldsymbol{A} \in \mathbb{R}^{N \times M}$	Data matrix (Arranged as observation by feature)
$\boldsymbol{Y} \in \mathbb{R}^{N \times P}$	Original learning response (or learning target)
$\hat{\boldsymbol{Y}} \in \mathbb{R}^{N \times P}$	Predicted learning response (or learning target)
$\mathbf{w} \in \mathbb{R}^{M \times 1}$	Model vector
$\mathbf{W} \in \mathbb{R}^{M \times P}$	Model matrix
$\epsilon \in \mathbb{R}$	Tuning parameter for L21-MSVR
$\boldsymbol{\xi}, \boldsymbol{\xi}^* \in \mathbb{R}^{N \times P}$	Slack variables for L21-MSVR

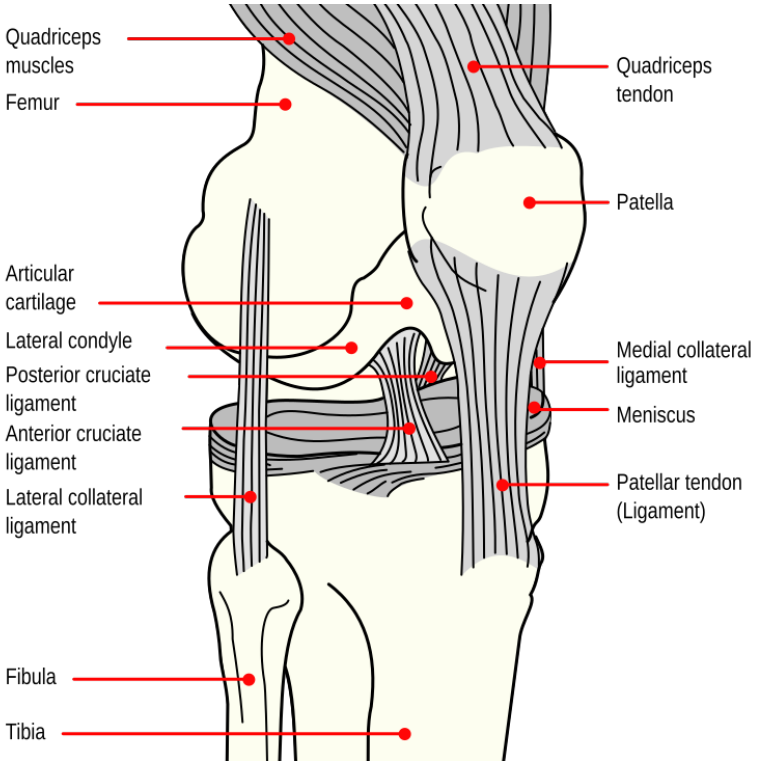


Figure 3.1: Right knee, seen from an angle between anteriorly and laterally [4]. The three main parts of the knee are the thigh bone (femur), the shin bone (tibia) and the kneecap (patella). The femur joins the tibia to form the main joint, whereas the patella meets the femur to make another joint [1].

curate replication of the natural anatomy in anatomic reconstruction is crucial to fully restore knee joint function and reduce impingement on or iatrogenic injury to adjacent structures [5]. However, current approaches to treating knee injuries may be inconsistent and ineffective in restoring knee function and preventing the development of osteoarthritis (OA). Analyses of long-term outcomes after ACL reconstruction have revealed that only 37% of patients were restored back to normal in terms of knee structure and function [6], and 90% of ACL reconstructed knees exhibited radiographic evidence of OA 3-10 years after injury [7]. A growing body of clinical evidence suggests that anatomic reconstruction can better restore joint function and deter the development of OA. However, a number of challenges are present in the practice of anatomic reconstruction of the cruciate ligaments.

Intraoperative identification of native cruciate ligament insertion sites, as a requisite for anatomical reconstruction, poses a tremendous challenge. Not all surgeons can maintain an acute awareness of the anatomy: about 85% of ACL reconstructions are done by surgeons who perform fewer than ten cases per year [8], and PCL reconstructions are even less frequently performed by most surgeons; for those who can, factors including the arthroscopic distortion, and disappearance of the ligament remnant (naturally or due to a notchplasty procedure) can still cause misidentification of the natural insertion or attachment sites. There is considerable variability of knee anatomy in terms of bone and soft tissue insertion morphology (position, size, and shape) [9]. Sample data from our preliminary study of tibial insertion site morphometry suggest that simplistic cross-referencing or generalization from one patient to another is likely to lead to nonanatomical tunnel drilling and iatrogenic injury to adjacent tissue structures such as meniscus roots. Although it may be difficult to gauge the incidence and impact of these iatrogenic injuries as complications of ACL or PCL surgeries, the importance of minimizing the risk of such injuries is readily recognized [10, 11].

The key to anatomic surgery of cruciate ligaments with minimized risk of iatrogenic injury is accurate, quantitative knowledge of the tissue morphology, documenting interperson variability and specificity versus uncertainty associated with alternative ways to predict morphometrics. Studies have investigated the quantification of the insertion sites of cruciate ligaments and other soft tissue components using statistical and quantitative approaches [12, 13, 14]. However, such quantitative analysis and measures cannot fully capture accurate spatial arrangement of soft tissues. The location and morphological measures cannot account for the interperson variability of cruciate ligaments and meniscus insertions, which are mostly characterized by qualitative measures [15, 16]. While in surgery, it is crucial to identify the native location of the cruciate ligaments to reconstruct the natural anatomy of the ligament structure. Therefore, one needs inference on the appropriate insertion sites of native cruciate ligaments. Due to the complex anatomy of the knee, identification of insertion sites of cruciate ligaments and meniscus roots in knee reconstruction surgery presents a great challenge intra-operatively or pre-operatively during surgical planning.

This study presents a new multi-response regression framework in order to achieve accurate prediction of the insertion sites for the six cruciate ligaments and two cartilage surfaces (collectively referred to as "cartilage and soft issues" in the following). The challenge of this study lies in the large number of features (508) and response variables (24) as opposed to the limited number of training samples (20). In particular, the proposed framework first digitalized outlines of the tibia from three-dimensional computed tomography (CT) images and aligned the outlines using generalized procrustes analysis (GPA) techniques. It then extracted patient-specific morphological features of the knee. Last but not least, the three-step structure-learning model identifies the insertion locations of all cartilages and soft tissues simultaneously.

In the first learning step, an $L_{2,1}$ sparsity-inducing regularization was incorporated into the multi-response support vector regression (SVR) to cope with the challenge of having more features and response variables than observations. Secondly, sparse learning model was constructed to learn the intrinsic spatial relationship between the cartilages and soft tissues. Finally, a structure-supervised prediction model that considers both the feature-response relationship from step 1 and the intrinsic spatial relationship from step 2 was used to simultaneously predict the optimal insertion centroids of the cartilages and soft tissues.

As opposed to ridge regression in [17], support vector regression is chosen to be the base algorithm. It generally performs better than ridge regression or lasso regression with careful parameter tuning at the expense of one more parameter (ϵ) to tune. Instead of training the projection vector $\mathbf{w}_m$ one-by-one in formulation (2.1), L21-MSVR trains the projection matrix $\mathbf{W}$ considering all responses. In addition, the $L_{2,1}$-norm encourages row sparsity, where the number of rows corresponds to the number of features. As a result, the $L_{2,1}$-norm encourages parsimony of features, which is itself a method of feature selection for features in the matrix representation [18, 19]. It is especially sparse when the number of features is a lot larger than that of the observations.

In summary, the contribution of this study includes the following: 1) a new method of feature extraction was developed to capture morphological features of the knee from aligned three-dimensional CT scan images; 2) a novel three-step structure-learning and structure-supervised prediction framework was developed. The proposed framework outperforms the state-of-the-art multi-response regression models.

3.2 Methods

3.2.1 Data Collection

Twenty tibia specimens (ten left and ten right unpaired knees; 11 from men and 9 from women; mean age at death: 61 $\pm$ 5 years) were used to acquire the morphometric data [20]. All epithelial, subcutaneous, and muscular tissues were removed from the specimens. High-resolution CT scans of the tibias were taken with slice spacing of 0.625 mm and 3-D bone models of the tibias were created in Mimics (Materialise Inc., Belgium). A Polaris Spectra optical tracking system (Northern Digital Inc., Ontario, Canada), with a manufacturer-reported accuracy of $\pm$0.25 mm, was used to digitize the outlines of the ACL, PCL, the medial cartilage (MCART), the lateral cartilage (LCART), the anterior and posterior medial meniscal roots (AMMR and PMMR), and the anterior-lateral and posterior-lateral meniscal roots (ALMR and PLMR). The digitization was performed by the same experimenter with repeatability, as assessed by intraclass correlation coefficients, ranging from 0.94 to 0.99. The digitized outlines were mapped onto the CT-based 3-D tibia models with a fiducial registration error smaller than 2% [13]. A closed spline was then fitted to each outline, resulting in 100 equidistant discrete points to represent the outline, as shown in figure 3.2.

A 3-D coordinate system was defined on each tibia based on its digitized and mapped cartilage outlines (see figure 3.2). First, the origin of the coordinate system was determined as the midpoint of the medial and lateral cartilage centroids. Principal component analysis (PCA) was then performed on the equidistant discrete points representing the cartilage outlines (200 points in total). The X-axis was the first principal component axis passing through the origin and pointing laterally. The Y-axis was orthogonal to the X-axis, passing through the origin and pointing anteriorly. To

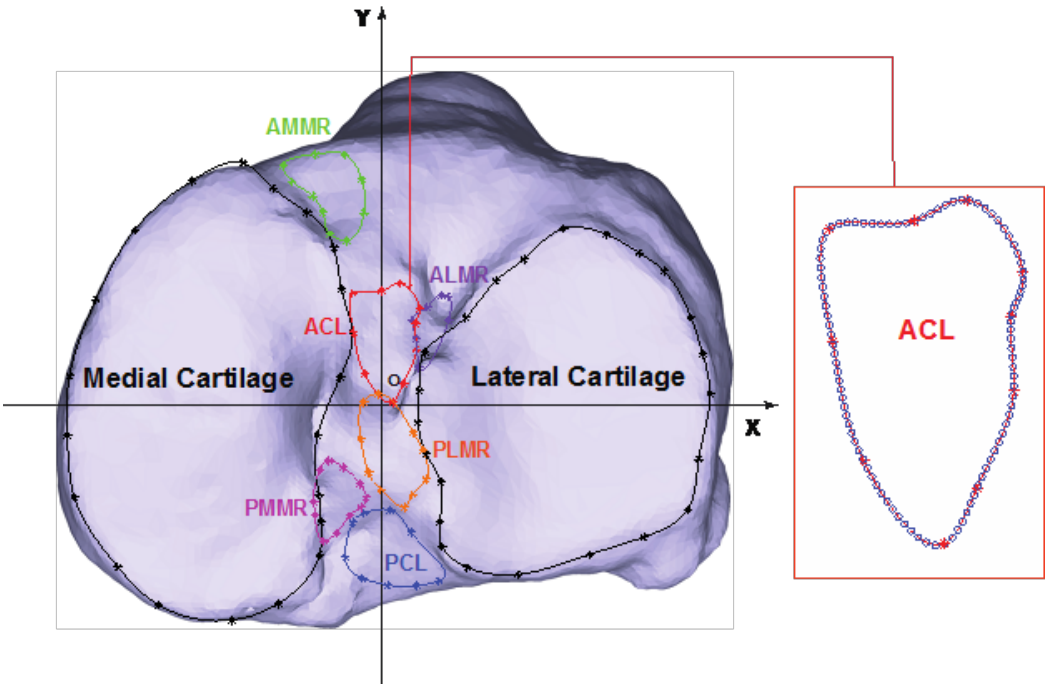


Figure 3.2: Digitized cartilage and insertion site outlines mapped onto the CT-based 3-D tibia model. The digitized points (asterisks) were spline-fitted, generating 100 equidistant points (circles on the close-up view of ACL insertion outline) on the fitted outlines to facilitate the subsequent analyses [13, 20].

make the Z-axis point proximally, the coordinate system was designed as a right-handed system for the right tibia and a left-handed system for the left tibia.

3.2.2 Image Alignment and Normalization Using Generalized Procrustes Analysis

Cartilage outlines for all 20 tibias were optimally aligned using GPA, which is an iterative process of applying procrustes superimposition to all possible pairs of configurations; a configuration here refers to a set of cartilage outline landmark coordinates in a predefined order. For each cartilage configuration pair, one configuration served as the base and the other as the target. Procrustes superimposition matches the target configuration onto the base, centering, rotating, and uniformly scaling the target configuration to minimize the shape difference. For multiple (20

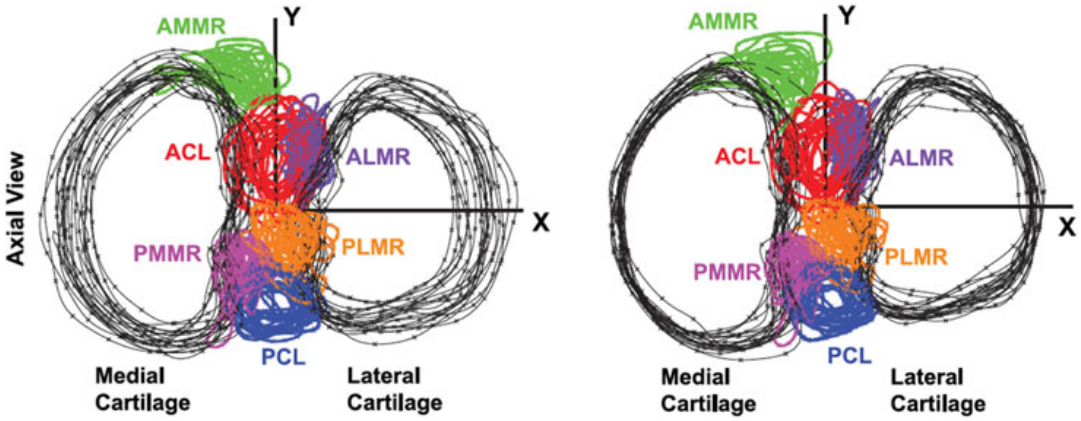


Figure 3.3: Outlines of tibial cartilage and six insertion sites from 20 subjects before (left plot) and after (right plot) cartilage-based GPA [13, 20].

in this study) configurations, GPA identified the reference or overall base configuration as the one with the smallest overall procrustes distance to all others (i.e., the 19 remaining tibial cartilage configurations). The 19 remaining configurations were then procrustes-superimposed onto this selected reference and their insertion sites transformed accordingly by the same translation, rotation, and scaling rules, without any shape distortion. Figure 3.3 shows the outlines of tibial cartilage and six insertion sites from 20 subjects before and after cartilage-based GPA.

3.2.3 Feature Engineering Using Spherical Coordinates

The outline of the tibia can be easily and reliably measured, making it a feature candidate to predict the locations of intangible soft tissues. Features are selected via a spherical coordinate system, which consists of radial distance (r), polar angle (θ) and azimuthal angle (ϕ). See figure 3.4 for an illustration.

A 3-D point with Cartesian coordinates (x, y, z) can be transformed into polar coordinates (r, θ, ϕ) following the rules: $r = \sqrt{x^2 + y^2 + z^2}$, $\theta = \text{arc cos}(z/r)$, and $\phi = \text{arc tan}(y/x)$. Thus the direction of a tibia point from the origin can be described by the two angles (θ, ϕ). The entire range of each angle ($[-\pi, \pi]$) is divided into N

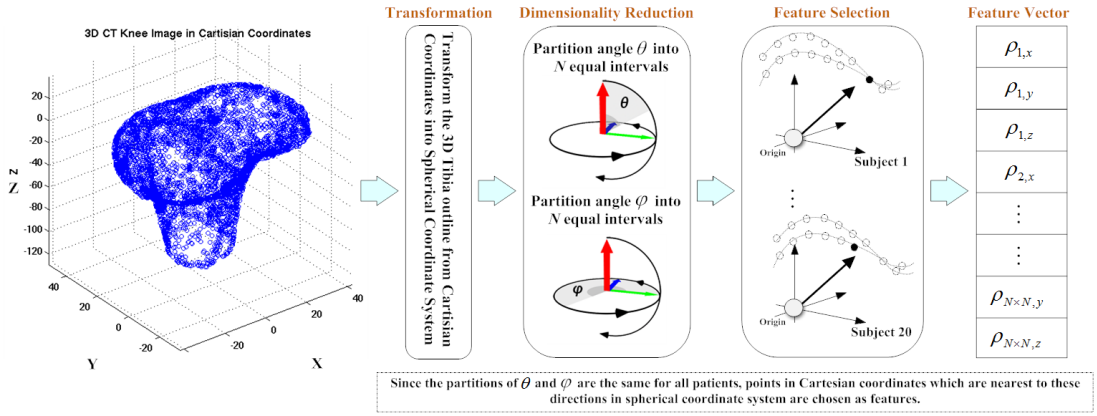


Figure 3.4: A visual illustration of feature selection. Points in the tibia feature space are converted to a spherical coordinate system. The entire range of each angle in the spherical coordinate is divided into $N = 13$ equal intervals, rendering 169 combinations for θ and ϕ. Outline points that are nearest to these directions are chosen as features in the form of Cartesian coordinates. Together with the side of the knee (left or right), there are $169 \times 3 + 1 = 508$ features for each observation.

equal intervals, resulting in a combination of $N \times N = N^2$ directions from the origin. Points along the outline tibia that are closest to these N^2 directions will be selected as features in the form of Cartesian coordinates. In this regard, N was chosen to be 13, and thus the number of selected features of each subject is $N^2 \times 3 = 507$.

In addition, raw CT image data consist of knees from left and right legs. Another useful feature is whether it is a left or right knee. Therefore the total number of features is $507 + 1 = 508$.

3.2.4 Response Variable

There is high variability in ligament and meniscus insertions. In a surgical procedure, the centroids of insertions are of particular importance to tunnel locations. Thus, this work focuses on the prediction of centroids of the ligament and meniscus insertions instead of the complete morphology of insertions.

For each subject $i \in 1, \ldots, 20$, the centroid of an insertion site j, (a_{ij}, b_{ij}, c_{ij}), is defined as:

$$\left(\frac{\max(x_{ij}) + \min(x_{ij})}{2}, \right. \\ \left. \frac{\max(y_{ij}) + \min(y_{ij})}{2}, \frac{\max(z_{ij}) + \min(z_{ij})}{2} \right). \tag{3.1}$$

3.2.5 Proposed Three-Step Spatial-Structure Supervised Learning and Prediction Model (L21-SSSL)

Extended from the $L_{2,1}$-Regularized, $L_{2,2}$-Loss Support Vector Regression Model (2.15) in the last chapter, the following L21-SSSL model is proposed, where $\mathbf{A} \in \mathbb{R}^{P \times P}$ is a matrix describing the linear relationship between each component of $\mathbf{Y}$:

Learning Step 1: Solve $L_{2,1}$-Regularized, $L_{2,2}$-Loss Support Vector Regression Model with formulation (2.15) to obtain the projection matrix $\mathbf{W}$.

Learning Step 2:

$$\begin{aligned} \min_{\mathbf{A}} \ & \alpha ||\mathbf{YA}||_1 + ||\mathbf{A}||_1 \\ \text{s.t.} \quad & \sum_{p=1}^{P} y_{np} a_{pk} = 0, \quad a_{kk} \neq 0, \\ & n \in \{1, \ldots, N\}, k \in \{1, \ldots, P\} \end{aligned} \tag{3.2}$$

Prediction Step:

$$\min_{\hat{\mathbf{Y}}} \quad ||\hat{\mathbf{Y}} - \mathbf{XW}||_1 + \alpha ||\hat{\mathbf{Y}}\mathbf{A}||_1 \tag{3.3}$$

As suggested in [17], the $\mathbf{A}$ matrix in (3.2) learns the spatial arrangement of the centroid of the soft tissues, by formulating each coordinate of the soft tissues as the linear representation of the remaining response variables. By forcing one of the elements along each column (or each row) of the $\mathbf{A}$ matrix to be non-zero, a linear representation is enforced.

In other words, the objective function of formulation (3.3) considers both the prediction error and the error of the spatial arrangement in the three-dimensional feature space, thus avoiding too much deviation of the centroid prediction from the original centroid position. In [17], formulation (2.7) (or formulation (2.15)) and formulation (3.2) are combined. However, these two formulations can be separated.

For the prediction step (3.3), the L_1-norm is used to minimize the prediction error of the objective function.

Both formulations (3.2) and (3.3) can be solved efficiently by any convex programming package such as CVX [21].

3.3 Evaluations

Experiments were executed on a workstation with an Intel i7-5960X CPU and 64 GB RAM with Ubuntu 16.04 operating system. Programming code was developed in Matlab R2016b.

3.3.1 Performance Measures

As presented in [22], evaluation measures for multi-response regression include average correlation coefficient, mean square error, average relative error, average root mean squared error (aRMSE) and average relative root mean squared error (aRRMSE). Among these measures, aRRMSE is considered as a more robust measure over others for multi-response prediction problems. It can be viewed as a normalized root mean squared error (RMSE) for each response variable as follows:

$$\text{aRRMSE} = \frac{1}{P}\sum_{p=1}^{P}\sqrt{\frac{\sum_{n=1}^{N_{test}}(y_{pn}-\hat{y}_{pn})^2}{\sum_{n=1}^{N_{test}}(y_{pn}-\bar{y}_p)^2}}, \tag{3.4}$$

where $\bar{y}_p$ is the average of responses for the p-th response variable, y_{pn} and $\hat{y}_{pn}$ are the actual and predicted responses of the p-th response variable and n-th observation respectively. During parameter tuning of the model, the performance using aRRMSE is considered so that learning will not be skewed towards variables of large error values and will not be affected by different scaling of the response variables. Finally, a leave-one-out cross validation was used for predicting the out-of-sample error. For each fold, one of the 20 observations is used as testing, and the remaining 19 observations are used for training.

3.3.2 Comparison with Baseline Methods

To illustrate the advantages of the proposed methods, four other multi-target prediction models are compared. All implementations are Matlab-based except for multi-target random forest, which is implemented via the scikit-learn package in Python. Parameters in each model, if any, are aRRMSE-tuned via a two-level grid search to ensure the learning performance is fair. The two-level grid search [23] consists of two parts - a coarse grid is applied to find a good region first, and then a finer grid search is applied on the identified area. Instead of running a full grid search, which is time consuming and computationally expensive, a two-level grid search is robust and at the same time reduces computational time.

The following methods in the literature were used to compare with the proposed method. Notations in the following models have been changed for better consistency with the above.

- Spatial Structure Supervised Learning (SSSL): SSSL was the first of its kind to learn an optimal insertion position of tibia soft tissue of knees in a two-dimensional setting and is the basis of this work [17]. The $\mathbf{A}$ matrix, which stores the spatial structure, is learnt during the first step and is carried over

to the prediction step. SSSL is an algorithm adaption method. Without the structural learning term, such method is reduced to the regular lasso model.

Learning step:

$$\begin{aligned} \min_{\mathbf{A},\mathbf{W}} \quad & \sum_{n=1}^{N} ||\mathbf{y}_n - \mathbf{W}^T\mathbf{x}_n||_2 + \alpha_1 \sum_{n=1}^{N} ||\mathbf{A}\mathbf{y}_n||_2 \\ & + \alpha_2 \sum_{i,j=1}^{P} |A_{ij}| + \alpha_3 \sum_{j=1}^{M}\sum_{p=1}^{P} |\mathrm{W}_{mp}| \\ \text{s.t.} \quad & [\mathbf{y}_1, \ldots, \mathbf{y}_P]\, \mathbf{a}_p = \mathbf{0}, \;\; a_{pp} \neq 0, p = 1, \ldots, P \end{aligned} \tag{3.5}$$

Prediction step:

$$\min_{\mathbf{y}} \quad ||\mathbf{y} - \mathbf{W}^T\mathbf{x}||_1 + \alpha_1||\mathbf{A}\mathbf{y}||_1 \tag{3.6}$$

- Generalized linear model via penalized maximum likelihood (GLMNET, [24, 25]): GLMNET is an extension of the single-target case where it is designed to model correlated responses with a multi-response Gaussian distribution. The absolute penalty on each single coefficient is replaced with a group-lasso penalty.

$$\begin{aligned} \min_{(\mathbf{W}_0,\mathbf{W})\in\mathbb{R}^{(M+1)\times P}} & \frac{1}{2N}\sum_{n=1}^{N} ||\mathbf{y}_n - \mathbf{W}_0^T - \mathbf{W}^T\mathbf{x}_n||_F^2 \\ & + \lambda \left[\frac{1}{2}(1-\alpha)||\mathbf{W}||_F^2 + \alpha \sum_{m=1}^{M} ||\mathbf{w}_m||_2 \right] \end{aligned} \tag{3.7}$$

- Multi-target regressor stacking using decision tree (MTRS-DT, [26]): MTRS-DT is inspired from "stacked generalization" which was designed for multi-label classification. There are two stages in the learning process. Firstly, P single-target models are learned as usual. MTRS-DT then predicts the target one-by-one with stacking: after predicting the first response variable, its output will be attached to the original testing dataset to build a new meta model, which is in turned used to predict the second target variable. The process continues until all response variables are predicted in this manner [22, 26]. The stacking idea

is powerful and can be applied to any single-target regression model. However, the sequence of stacking affects the regression performance. In this experiment, the prediction of the first response variable will be used in the prediction of the next.

- Multi-target random forest (MTRF, [27, 28, 29]): A random forest is a meta-learner that trains a number of decision trees, each of which uses a subset of the original dataset and outputs the final prediction by averaging the prediction from each decision tree. Generally speaking, MTRF differs from its single-target counterpart in that during the splitting of a tree node, the performance measure considers all the output variables instead of considering a single output variable at a time.

3.3.3 Discussion of Computational Result

Table 3.2 summarizes the performance of predicting each of eight insertion centroids as well as predicting all eight locations at the same time with the proposed models and baseline models using data of the left knee only, the right knee only and both knees. For each soft tissue centroid, the average relative root mean squared error (aRRMSE) over the 20 subjects is reported. In general, the proposed L21-SSSL performs better than other baseline models in terms of aRRMSE in five insertion areas (see ACL, ALMR, AMMR, MCART, and PLMR using data of both knees at Table 3.2). The proposed model is also the best method for predicting all centroids simultaneously (0.8524 for L21-SSSL).

The advantage of the proposed model, as well as GLMNET and SSSL, is that the more responses available in the dataset, the better the learning performance, provided that responses are related to each other, such as having spatial relationship in this

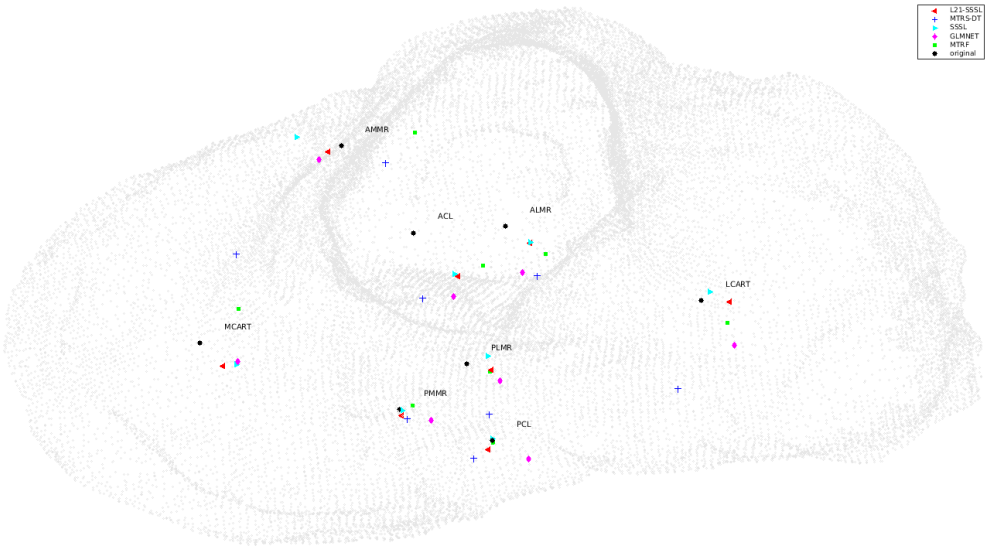


Figure 3.5: Outline of tibia and the centroids of soft tissues of a subject viewed at $(\theta, \phi) = (0, 90)$. Original centroid position of a soft issue is highlighted as black. The predicted centroids are as follows: red triangle for L21-SSSL, blue cross for MTRS-DT, cyan triangle for SSSL, pink diamond for GLMNET and green square for MTRF. The L21-SSSL predicted centroids are closer to the original centroids. This figure is generated with Matlab.

case. Because of the limited number of observations, a non-linear regression model, such as kernelized learning, may not be applicable because of possible over-fitting.

MTRS-DT and MTRF perform worse than the other three methods in this application. This is likely due to a lack of available observations. Decision trees and random forests are particularly suitable for datasets with many observations and high non-linearity in the decision boundary. Computational speed is also their strength.

Figure 3.5 shows the insertion centroids of the cartilages and soft tissues (in black) as well as the predicted positions using different algorithms - red triangle for L21-MSVR, blue cross for MTRS-DT, cyan triangle for SSSL, pink diamond for GLMNET and green square for MTRF. L21-MSVR is noticeably closer to the original position, followed by GLMNET and SSSL. Such experimental results confirm that

the proposed supervised learning can improve prediction of the eight soft tissues simultaneously.

3.4 Concluding Remarks

The motivation of this study is to develop a quantitative, automated framework in order to localize the optimal position of soft tissue insertion using patient-specific features from knee imaging data. An extensive quantitative analysis of the location and the interrelationship of soft tissue insertions on the tibial plateau has been performed, including digitalization of tibia outlines from 3-D CT images, imaging alignment using GPA, patient-specific morphological feature extraction from tibia outlines and multi-response support vector regression. The proposed methodology yielded the best prediction performance as compared with other baseline models for eight soft tissue structure locations.

A few directions are recommended for future exploration. A non-linear learning model might be considered, in addition to increasing the number of observations and other innovative ideas of feature extraction. Predicting the shape and size of the insertion sites, and predicting the performing angle of the surgical tools can be a promising direction to continue this work. This quantitative analysis framework could be the basis of a clinical application to assist surgeons to better identify soft tissue insertion sites. Such an application, preferably on a computer-aided surgery platform, would render a close replication of the native anatomy so that the risk of non-anatomic tunnel placement or iatrogenic injury to adjacent tissue structures can be minimized.

Bibliography

[1] Turner A Blackburn and Emily Craig. Knee anatomy: a brief review. *Physical therapy*, 60(12):1556–1560, 1980.

[2] Thomas J Gill, Samuel K Van de Velde, David W Wing, Luke S Oh, Ali Hosseini, and Guoan Li. Tibiofemoral and patellofemoral kinematics after reconstruction of an isolated posterior cruciate ligament injury: in vivo analysis during lunge. *The American journal of sports medicine*, 37(12):2377–2385, 2009.

[3] HOWARD B Tandeter and PESACH Shvartzman. Acute knee injuries: use of decision rules for selective radiograph ordering. *American family physician*, 60(9):2599–2608, 1999.

[4] Wikimedia Commons. File:knee diagram.svg — wikimedia commons, the free media repository, 2016. [Online; accessed 31-July-2017].

[5] Marc T Galloway, Edward S Grood, John N Mehalik, Martin Levy, Stephen C Saddler, and Frank R Noyes. Posterior cruciate ligament reconstruction: an in vitro study of femoral and tibial graft placement. *The American journal of sports medicine*, 24(4):437–445, 1996.

[6] David J Biau, Caroline Tournoux, Sandrine Katsahian, Peter Schranz, and Rémy Nizard. Acl reconstruction: a meta-analysis of functional scores. *Clinical orthopaedics and related research*, 458:180–187, 2007.

[7] Donald C Fithian, Elizabeth W Paxton, Mary Lou Stone, William F Luetzow, Rick P Csintalan, Daniel Phelan, and Dale M Daniel. Prospective trial of a treatment algorithm for the management of the anterior cruciate ligament-injured knee. *The American journal of sports medicine*, 33(3):335–346, 2005.

[8] William E Garrett Jr, Marc F Swiontkowski, James N Weinstein, John Callaghan, Randy N Rosier, Daniel J Berry, John Harrast, G Paul Derosa, et al. American board of orthopaedic surgery practice of the orthopaedic surgeon: part-ii, certification examination case mix. *JBJS*, 88(3):660–667, 2006.

[9] Mario Ferretti, Daniel Doca, Sheila M Ingham, Moises Cohen, and Freddie H Fu. Bony and soft tissue landmarks of the acl tibial insertion site: an anatomical study. *Knee Surgery, Sports Traumatology, Arthroscopy*, 20(1):62–68, 2012.

[10] Hemanth R Gadikota, Jae Ang Sim, Ali Hosseini, Thomas J Gill, and Guoan Li. The relationship between femoral tunnels created by the transtibial, anteromedial portal, and outside-in techniques and the anterior cruciate ligament footprint. *The American journal of sports medicine*, 40(4):882–888, 2012.

[11] James H Lubowitz. Anteromedial portal technique for the anterior cruciate ligament femoral socket: pitfalls and solutions. *Arthroscopy: The Journal of Arthroscopic & Related Surgery*, 25(1):95–101, 2009.

[12] Christopher D Harner, Goo Hyun Baek, Tracy M Vogrin, Gregory J Carlin, Shinji Kashiwaguchi, and Savio LY Woo. Quantitative analysis of human cruciate ligament insertions. *Arthroscopy: The Journal of Arthroscopic & Related Surgery*, 15(7):741–749, 1999.

[13] Kang Li, Madelyn O'Farrell, Daniel Martin, Sebastian Kopf, Christopher Harner, and Xudong Zhang. Mapping ligament insertion sites onto bone surfaces in knee by co-registration of ct and digitization data. *Journal of biomechanics*, 42(15):2624–2626, 2009.

[14] Rainer Siebold, Thomas Ellert, Stefan Metz, and Juergen Metz. Tibial insertions of the anteromedial and posterolateral bundles of the anterior cruciate

ligament: morphometry, arthroscopic landmarks, and orientation model for bone tunnel placement. *Arthroscopy: The Journal of Arthroscopic & Related Surgery*, 24(2):154–161, 2008.

[15] Andrew Edwards, Anthony MJ Bull, and Andrew A Amis. The attachments of the anteromedial and posterolateral fibre bundles of the anterior cruciate ligament. *Knee Surgery, Sports Traumatology, Arthroscopy*, 16(1):29–36, 2008.

[16] Goro Tajima, Masahiro Nozaki, Takanori Iriuchishima, Sheila JM Ingham, Wei Shen, Patrick Smolinski, and Freddie H Fu. Morphology of the tibial insertion of the posterior cruciate ligament. *JBJS*, 91(4):859–866, 2009.

[17] Cao Xiao, Shouyi Wang, Liying Zheng, Xudong Zhang, and Wanpracha Art Chaovalitwongse. A patient-specific model for predicting tibia soft tissue insertions from bony outlines using a spatial structure supervised learning framework. *IEEE Transactions on Human-Machine Systems*, 46(5):638–646, 2016.

[18] Xiao Cai, Feiping Nie, Heng Huang, and Chris Ding. Multi-class l2, 1-norm support vector machine. In *Data Mining (ICDM), 2011 IEEE 11th International Conference on*, pages 91–100. IEEE, 2011.

[19] Feiping Nie, Heng Huang, Xiao Cai, and Chris H Ding. Efficient and robust feature selection via joint l2, 1-norms minimization. In *Advances in neural information processing systems*, pages 1813–1821, 2010.

[20] Liying Zheng, Christopher D Harner, and Xudong Zhang. The morphometry of soft tissue insertions on the tibial plateau: data acquisition and statistical shape analysis. *PloS one*, 9(5):e96515, 2014.

[21] Michael Grant and Stephen Boyd. CVX: Matlab software for disciplined convex programming, version 2.1. `http://cvxr.com/cvx`, March 2014.

[22] Hanen Borchani, Gherardo Varando, Concha Bielza, and Pedro Larrañaga. A survey on multi-output regression. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 5(5):216–233, 2015.

[23] Chih-Wei Hsu, Chih-Chung Chang, Chih-Jen Lin, et al. A practical guide to support vector classification. 2003.

[24] Jerome Friedman, Trevor Hastie, and Rob Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of statistical software*, 33(1):1, 2010.

[25] Noah Simon, Jerome Friedman, and Trevor Hastie. A blockwise descent algorithm for group-penalized multiresponse and multinomial regression. *arXiv preprint arXiv:1311.6529*, 2013.

[26] Eleftherios Spyromitros-Xioufis, Grigorios Tsoumakas, William Groves, and Ioannis Vlahavas. Multi-target regression via input space expansion: treating targets as inputs. *Machine Learning*, 104(1):55–98, 2016.

[27] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[28] H Tim Kam. Random decision forest. In *Proc. of the 3rd Int'l Conf. on Document Analysis and Recognition, Montreal, Canada, August*, pages 14–18, 1995.

[29] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Table 3.2: The following table shows the prediction performance in average relative root mean squared error (aRRMSE) of the centroids of eight insertion sites using left-side, right-sided and both knees using leave-one-out cross validation. A lower aRRMSE indicates a better prediction performance. Bold entries correspond to the best prediction for the insertion area. Generally, learning performance is enhanced if the knee data of both sides is used and if multiple insertion sites are learned simultaneously.

	Left				
	SSSL	GLMNET	MTRS -DT	MTRF	L21- MSVR
ACL	1.0057	1.0206	1.1018	1.3590	**0.8977**
ALMR	1.0687	1.0655	1.1018	1.4793	**0.8905**
AMMR	0.8764	1.1000	1.0914	1.4421	**0.8638**
LCO	1.0336	1.0247	1.1203	1.7627	**0.8391**
MCO	1.2163	1.2023	1.0839	1.4627	**1.0215**
PCL	1.4237	1.3492	**1.0921**	1.6554	1.2344
PLMR	1.1319	**1.0168**	1.0947	1.5869	1.0350
PMMR	1.6138	1.3762	**1.1382**	1.5326	1.3821
All	1.1713	1.1444	1.1212	1.5351	**1.0205**

	Right				
	SSSL	GLMNET	MTRS -DT	MTRF	L21- MSVR
ACL	1.3455	1.3259	**1.1067**	1.3336	1.1679
ALMR	1.2636	1.2609	1.0909	1.1339	**1.0676**
AMMR	1.0767	1.0926	1.1340	1.3009	**0.9975**
LCO	1.1871	1.0845	1.1063	1.1748	**0.9617**
MCO	0.9374	0.7977	1.1026	1.1124	**0.7434**
PCL	1.3829	1.0841	**1.0734**	1.4416	1.1303
PLMR	1.2519	1.1376	1.0955	1.1747	**1.0474**
PMMR	1.2564	1.1213	1.0992	1.2166	**1.0343**
All	1.2127	1.1131	1.0891	1.2361	**1.0188**

	Both				
	SSSL	GLMNET	MTRS -DT	MTRF	L21- MSVR
ACL	1.0380	1.0055	1.0320	1.4772	**0.9458**
ALMR	0.9673	0.8844	1.2293	1.4026	**0.8574**
AMMR	0.7801	0.7255	1.2682	1.3257	**0.7157**
LCO	0.7200	**0.7087**	1.1775	1.2702	0.7090
MCO	0.6961	0.6839	1.2931	1.2407	**0.6783**
PCL	1.1923	1.2238	1.4040	1.3068	**1.0918**
PLMR	0.9900	0.9115	1.2621	1.4934	**0.8868**
PMMR	0.9073	**0.9038**	1.3180	1.3834	0.9341
All	0.9114	0.8809	1.2480	1.3625	**0.8524**

CHAPTER 4

Selection of Hierarchical Features via Sparse Group Regularization

Abstract

This research presents a framework of hierarchical sparse group lasso (HSGL), which selects features arranged in a hierarchical manner with sparse group regularization. HSGL is proposed to better address cases in which features can be naturally grouped, especially in bioinformatics and medical imaging. HSGL generalizes from sparse group lasso (SGL) and hierarchical lasso (HL) and is particularly successful in identifying a handful but yet important features from a large group of irrelevant features. In addition, HSGL is easy to interpret and implement, converges quickly and has better learning performance than other baseline learners when there are more features than observations.

Keywords: Feature selection, group selection, lasso, regularization, supervised learning, hierarchical group structure learning, sparse group lasso, hierarchical lasso, classification, regression.

4.1 Introduction

Feature selection is a popular research area in bioinformatics [1, 2, 3]. Problems such as gene expression, microarray analysis and medical imaging rely heavily on feature selection techniques in order to identify important features for interpretation. In the domain of machine learning, feature selection can be under supervised learning

or unsupervised learning [4]. Within the scope of supervised learning, there are three categories: i) wrapper methods, ii) filter methods and iii) embedded methods.

Wrapper methods use a predictive model to score feature subsets. Each new subset is used to train a model, which is tested on a validation set. Counting the number of mistakes made on that validation set (the error rate of the model) gives the score for that subset. A subset of features with the best performance is selected. The advantages of wrapper methods are that performance scores are easy to compute, and they identify optimal subsets to build learning models without specifying the number of features required beforehand. However, they are relatively slower than the other two methods. Examples include sequential forward selection (SFS) and sequential backward selection (SBS). Filter methods use proxy measures instead of learning error rates to score feature subsets. These measures are chosen to be fast to compute, while still capturing the usefulness of feature sets. Performance scores are assigned to features, and features with the highest scores are chosen. Filter methods are fast and intuitive, but the number of features needs to be specified. Examples include minimum redundance maximum relevance (mRMR) [5], Pearson's correlation, linear discriminant analysis (LDA) and ANOVA. Embedded methods combine the qualities of filter and wrapper methods. They are implemented by algorithms that have their own built-in feature selection methods. Examples include lasso [6] and support vector machines [7]. New effective supervised feature selection enhances the learning performance and eliminates irrelevant features for better interpretability of the model [3]. In this paper, a new framework for feature selection - hierarchical sparse group lasso (HSGL) - is proposed in order to better select features in a hierarchical structure. HSGL is the embedded method of supervised feature selection. Similar to lasso, HSGL can accommodate both (binary) classification and regression problems, depending on a decision function.

Table 4.1: Notations used in chapter 4.

General

Symbol	Meaning
N	No. of observation
M	No. of features
$\boldsymbol{A} \in \mathbb{R}^{N \times M}$	Data matrix (Arranged as observation by feature)
$\boldsymbol{y} \in \mathbb{R}^{N \times 1}$	Learning target (label or response)
$\boldsymbol{x}, \boldsymbol{d}, \boldsymbol{\beta} \in \mathbb{R}^{M \times 1}$	Feature vector (or model vector)
$\lambda, \lambda_{11}, \lambda_{12}, \lambda_2 \in \mathbb{R}$	Regularization parameters

For Group Lasso with L_2-norm (L2-GL) and L_∞-norm (Linf-GL), and Sparse Group Lasso (SGL)

Symbol	Meaning
$\beta_g \in \mathbb{R}, g = 1, \ldots, G$	Feature vector $\boldsymbol{\beta}$ at the gth group

For Hierarchical Lasso (HL) and Hierarchical Sparse Group Lasso (HSGL)

Symbol	Meaning
$\beta_{kg} \in \mathbb{R}, k = 1, \ldots, K,\ g = 1, \ldots, G_k$	Feature vector $\boldsymbol{\beta}$ at the kth top group and gth bottom group
$d_k \in \mathbb{R}$	Feature coefficient of the top layer $\boldsymbol{d}$ at the kth top group
$x_{kg} \in \mathbb{R}$	Feature coefficient of the bottom layer $\boldsymbol{x}$ at the kth top group and gth bottom group
$M_{kg} \in \mathbb{R},\ M = \sum_{k=1}^{K} \sum_{g=1}^{G_k} M_k g$	No. of features at the gth bottom group within the kth top group

Lasso (least absolute shrinkage and selection operator) [6] was originally developed as a regression analysis method. It minimizes the usual sum of squared errors, with a bound on the sum of the absolute values of the coefficients. A L_1-norm penalty in the formulation is an effective way to alleviate the problem of over-fitting. In addition, lasso involves solving an easy convex optimizaiton problem. The formulation of the lasso optimization problem with least-square loss is as follows, where $\boldsymbol{A} \in \mathbb{R}^{N \times M}$, $\boldsymbol{y} \in \mathbb{R}^{N \times 1}$, $\boldsymbol{\beta} \in \mathbb{R}^{M \times 1}$ and $\lambda \in \mathbb{R}$:

$$\min_{\boldsymbol{\beta}} \ ||\boldsymbol{y} - \boldsymbol{A}\boldsymbol{\beta}||_2 + \lambda ||\boldsymbol{\beta}||_1 \tag{4.1}$$

Lasso was later generalized to many variants such as elastic nets [8] and group lasso [9]. Group lasso consists of predefined groups of covariates regularized by an L_2-norm, where $||\boldsymbol{\beta}_k||_2 = \sqrt{\beta_{k1}^2 + \cdots + \beta_{kG_k}^2}$ and $p_g \in \mathbb{R}$:

$$\min_{\boldsymbol{\beta}} \ ||\boldsymbol{y} - \boldsymbol{A}\boldsymbol{\beta}||_2 + \lambda \sum_{g=1}^{G} \sqrt{p_g} ||\boldsymbol{\beta}_g||_2 \tag{4.2}$$

The group lasso with L_2-norm penalty was extended to the one with L_∞-norm penalty [10], where $||\boldsymbol{\beta}_k||_\infty = \max(|\beta_{k1}|, |\beta_{k2}|, \ldots, |\beta_{kG_k}|)$:

$$\min_{\boldsymbol{\beta}} \ ||\boldsymbol{y} - \boldsymbol{A}\boldsymbol{\beta}||_2 + \lambda \sum_{g=1}^{G} \sqrt{p_g} ||\boldsymbol{\beta}_g||_\infty \tag{4.3}$$

Both the L_2-norm and L_∞-norm of $\boldsymbol{\beta}_g$ become zero when $\boldsymbol{\beta}_g = \mathbf{0}$; hence when λ is appropriately tuned, the penalty term can effectively remove unimportant groups. However, the limitation of using the L_2-norm and L_∞-norm penalties is that, when one variable in a group is selected, all other variables in the same group tend to be selected, better known as an "all-in-all-out" property [11]. Once a component of $\boldsymbol{\beta}_g$ is non-zero, the value of the two norm functions are no longer zero.

To select variables inside a group (instead of choosing all in a group as in group lasso), sparse group lasso (SGL) [9, 12] has an additional L_1-norm penalty:

$$\min_{\boldsymbol{\beta}} \ ||\boldsymbol{y} - \boldsymbol{A}\boldsymbol{\beta}||_2 + \lambda_1 ||\boldsymbol{\beta}||_1 + \lambda_2 \sum_{g=1}^{G} ||\boldsymbol{\beta}_g||_2 \tag{4.4}$$

SGL enforces sparse effects both on a group and within-group level of features. There have been successful implementations using accelerated gradient descent [13] and blockwise coordinate descent [9].

Close variants of HSGL include hierarchical lasso (HL) [11], hierarchically penalized support vector machine (H-SVM) [14] and tree structured group lasso (TSGL) [13, 15]. Zhou and Zhu [11] proposed a two-layer hierarchical lasso (HL) with least-square loss as follows, where there are K groups in the top layer, and under the kth top group, there are G_k sub-groups in the bottom layer:

$$\min_{\boldsymbol{d}\geq 0,\boldsymbol{x}} \sum_{n=1}^{N} ||y_n - \sum_{k=1}^{K} d_k \sum_{g=1}^{G_k} \boldsymbol{A}_{n,kg}^T \boldsymbol{x}_{kg}||^2 + \lambda_1 \sum_{k=1}^{K} d_k + \lambda_2 ||\boldsymbol{x}||_1 \tag{4.5}$$

H-SVM replaces the least-square loss in HL with hinge loss $(\bullet)_+$:

$$\min_{\boldsymbol{d}\geq 0,\boldsymbol{x}} \sum_{n=1}^{N} [1 - y_i(x_0 + \sum_{k=1}^{K} d_k \boldsymbol{A}_{n,kg}^T \boldsymbol{x}_{kg})]_+ + \lambda_1 \sum_{k=1}^{K} d_k + \lambda_2 ||\boldsymbol{x}||_1 \tag{4.6}$$

For TSGL, the hierarchical structure of the features is represented as a tree with leaf nodes as features and internal nodes as clusters of features. The formulation of TSGL is as follows, where $\boldsymbol{\beta} \in \mathbb{R}^{N\times 1}$ forms a certain tree structure and w_j^i is a weight at jth node of depth i (see figure 4.1 for illustration):

$$\min_{\boldsymbol{\beta}} \sum_{n=1}^{N} ||y_n - \boldsymbol{A}_n^T \boldsymbol{\beta}||^2 + \lambda \sum_{i=0}^{I} \sum_{j=1}^{J_i} w_j^i ||\boldsymbol{\beta}|| \tag{4.7}$$

TSGL allows variation for the number of features in each group and at each layer, whereas l-layer HL, HSVM and HSGL always have l group assignments for each feature.

In short, TSGL, H-SVM, HL and HSGL all take hierarchical features as input. TSGL outputs a single feature vector that can be presented with the same hierarchical tree structure as the input, whereas HL, H-SVM and HSGL output l feature vectors that correspond to the n-layer hierarchical structure. Figure 4.1 visually illustrates their differences.

The contribution of this work includes (1) the improvement of hierarchical lasso by incorporating sparse group regularization to achieve better feature selection and learning performance, (2) mathematical analysis to analyze the mathematical properties of HSGL, which is similar to that in [11, 14] for HL and H-SVM, and (3) an illustration of the appropriate usage of HSGL in selecting and interpreting features.

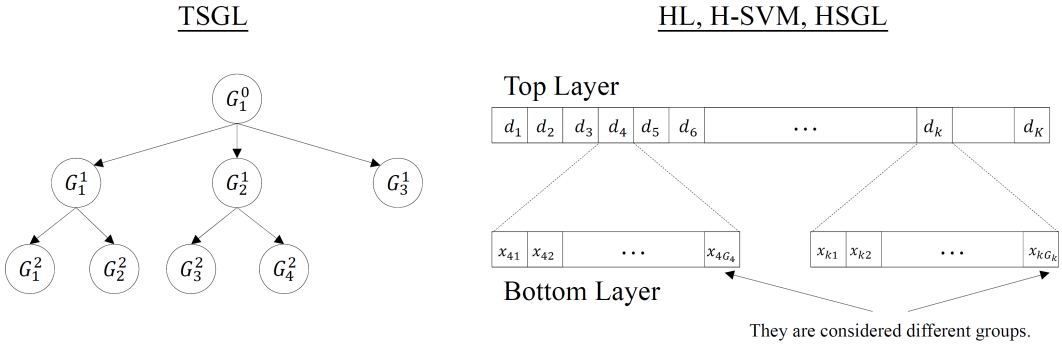


Figure 4.1: Comparing 1) Tree Structured Group Lasso (TSGL) on the left and 2) Two-Layer Hierarchical Lasso (HL), Hierarchically Penalized Support Vector Machine (H-SVM) and Hierarchical Sparse Group Lasso (HSGL) on the right. For TSGL, features are arranged into a tree-like hierarchy (G_j^i refers to the jth node at depth i), where features are represented as leaf nodes and groups of features are represented as internal nodes. In this example [13], there are eight features. The root node is $G_1^0 = \{1, 2, 3, 4, 5, 6, 7, 8\}$, depth 1 consists of $G_1^1 = \{1, 2\}, G_2^1 = \{3, 4, 5, 6\}, G_3^1 = \{7, 8\}$ and leaf (depth 2) consists of $G_1^2 = \{1\}, G_2^2 = \{2\}, G_3^2 = \{3, 4\}, G_4^2 = \{5, 6\}$. On the other hand, for l-layer HL, H-SVM and HSGL, each feature has l group assignments, and there will be l feature vectors obtained in the end. All l feature vectors are multiplied together in an element-wise manner to determine the final feature vector $\boldsymbol{\beta}$. In this example, there are two layers (i.e. $l = 2$).

4.2 Hierarchical Sparse Group Lasso

Hierarchical lasso and hierarchically penalized support vector machine [11] were extended for effectively removing unimportant variables at both group level and within-group level. To begin with, the feature vector β_{kg} is parameterized as

$$\boldsymbol{\beta}_{kg} = d_k \boldsymbol{x}_{kg}, \;\; k = 1, \ldots, K; \;\; g = 1, \ldots, G_k \tag{4.8}$$

where $d_k \geq 0$ for identifiability. In (4.8), $\boldsymbol{d}$ and $\boldsymbol{x}$ are the feature vectors at the top and bottom layers respectively. This decomposition indicates that feature vectors at the top layer d_k and bottom layer $\boldsymbol{x}_{kg}$ are all under the same top kth layer. The coefficient d_k thus affects all of its bottom sub-groups. Consider a two-layer hierarchical sparse group lasso (HSGL) with K groups in the top layer, and under the kth top group,

there are G_k sub-groups in the bottom layer, where $\boldsymbol{A} \in \mathbb{R}^{N\times M}$, $\boldsymbol{y} \in \mathbb{R}^{N\times 1}$, $\boldsymbol{d} \in \mathbb{R}^{K\times 1}$, $\boldsymbol{x} \in \mathbb{R}^{M\times 1}$, and $\lambda_{11}, \lambda_{12}, \lambda_2 \in \mathbb{R}$. The formulation of the HSGL optimization problem [1] is given by:

$$\begin{aligned}\min_{\boldsymbol{d}\geq 0, \boldsymbol{x}} \; & \sum_{n=1}^{N} ||y_n - \sum_{k=1}^{K} d_k \sum_{g=1}^{G_k} \boldsymbol{A}_{n,kg}^T \boldsymbol{x}_{kg}||^2 \\ & + \lambda_{11} \sum_{k=1}^{K} d_k + \lambda_{12}||\boldsymbol{x}||_1 + \lambda_2 \sum_{k=1}^{K} \sum_{g=1}^{G_k} d_k^2 ||\boldsymbol{x}_{kg}||_2\end{aligned} \tag{4.9}$$

Decision functions for prediction are:

$$\begin{aligned} f(\boldsymbol{A}_{\text{test}}) &= \boldsymbol{A}_{\text{test}}^T \boldsymbol{\beta} \qquad \text{(for regression)} \\ f(\boldsymbol{A}_{\text{test}}) &= \text{sign}(\boldsymbol{A}_{\text{test}}^T \boldsymbol{\beta}) \quad \text{(for classification)} \end{aligned} \tag{4.10}$$

In (4.9), $\boldsymbol{A}_{n,kg}$ is the nth observation at the kth top layer under the gth bottom layer. Parameters $\lambda_{11} \geq 0$, $\lambda_{12} \geq 0$ and $\lambda_2 \geq 0$ are tuned. λ_{11} controls the estimates at the upper group level, which can effectively remove unimportant groups; when d_k is zero, all β_{kg} in the kth group are equal to zero. λ_{12} controls the estimates at the variable-specific level: when d_k is not equal to zero, some of the $\boldsymbol{x}_{kg}$ hence some of the β_{kg}, $g = 1, \ldots, G_k$, still have the possibility of being zero. Similar to λ_{11}, λ_2 controls the estimates at the lower group level and can remove unimportant groups effectively.

As a side note, two-layer HSGL with least-squares loss (4.9) can be extended to three or more layers by similar parametrization of $\beta_{zkg} = c_z d_{zk} x_{zkg}$ where $z = 1, \ldots, Z$, $k = 1, \ldots, K_z$, and $g = 1, \ldots, G_{K_z}$ as follows:

[1] In the model, a scalar is represented by a non-bold lower-case letter, a vector is represented by a bold lower-case letter and a matrix is represented by a bold upper-case letter. Symbols in the formulations are changed for consistency.

$$\min_{\boldsymbol{c}\geq 0,\boldsymbol{d}\geq 0,\boldsymbol{x}} \sum_{n=1}^{N} ||y_n - \sum_{z=1}^{Z} c_z \sum_{k=1}^{K_z} d_{zk} \sum_{g=1}^{G_{K_z}} \boldsymbol{A}_{n,zkg}^T \boldsymbol{x}_{zkg}||^2 + \lambda_{11} \sum_{z=1}^{Z} c_z + \lambda_{12} \sum_{k=1}^{K_z} d_{zk} + \lambda_{13}||\boldsymbol{x}||_1 + \lambda_2 \sum_{z=1}^{Z} \sum_{k=1}^{K_z} \sum_{g=1}^{G_{K_z}} c_z^2 d_{zk}^2 ||\boldsymbol{x}_{zkg}||_2 \tag{4.11}$$

Two-layer HSGL is the focus of this chapter though.

Zhou and Zhu [11] claim that HL can effectively remove all unimportant groups and some of the unimportant variables within important groups, but it cannot effectively remove all unimportant variables within important groups. HSGL aims at addressing this issue by including the extra group-norm penalization at both the upper and bottom group layers to further improve its selection ability. Indeed, empirical experiments show that it is sometimes inappropriate to enforce regularization at both levels. Therefore, depending on the dataset, either the top or bottom level can be regularized instead of both as in (4.12) and (4.13) respectively:

HSGL with top-level penalization:

$$\min_{\boldsymbol{d}\geq 0,\boldsymbol{x}} \sum_{n=1}^{N} ||y_n - \sum_{k=1}^{K} d_k \sum_{g=1}^{G_k} \boldsymbol{A}_{n,kg}^T \boldsymbol{x}_{kg}||^2 + \lambda_{11} \sum_{k=1}^{K} d_k + \lambda_{12}||\boldsymbol{x}||_1 + \lambda_2 \sum_{k=1}^{K} d_k^2 \tag{4.12}$$

HSGL with bottom-level penalization:

$$\min_{\boldsymbol{d}\geq 0,\boldsymbol{x}} \sum_{n=1}^{N} ||y_n - \sum_{k=1}^{K} d_k \sum_{g=1}^{G_k} \boldsymbol{A}_{n,kg}^T \boldsymbol{x}_{kg}||^2 + \lambda_{11} \sum_{k=1}^{K} d_k + \lambda_{12}||\boldsymbol{x}||_1 + \lambda_2 \sum_{k=1}^{K} \sum_{g=1}^{G_k} ||\boldsymbol{x}_{kg}||_2 \tag{4.13}$$

Indeed, HSGL with top-level penalization does not have any group-norm penalization. Instead, it has the extra term d_k^2 and formulation (4.12) is similar to elastic net-regularized lasso. Interested readers are encouraged to implement formulations (4.12) and (4.13) in addition to the original formulation (4.9).

To reduce the challenge of parameter tuning, two of the three tuning parameters λ_{11} and λ_{12} can be combined into two as $\lambda_1 = \lambda_{11}\lambda_{12}$. It can be shown that formulation (4.9) is equivalent to

$$\min_{\boldsymbol{d}\geq 0,\boldsymbol{x}} \sum_{n=1}^{N} ||y_n - \sum_{k=1}^{K} d_k \sum_{g=1}^{G_k} \boldsymbol{A}_{n,kg}^T \boldsymbol{x}_{kg}||^2 + \sum_{k=1}^{K} d_k + \lambda_1 ||\boldsymbol{x}||_1 + \lambda_2 \sum_{k=1}^{K} \sum_{g=1}^{G_k} d_k^2 ||\boldsymbol{x}_{kg}||_2 \tag{4.14}$$

Lemma 4.2.1 *$(\hat{\boldsymbol{d}}^*, \hat{\boldsymbol{x}}^*)$ is a local minimizer of (4.9) if and only if $(\hat{\boldsymbol{d}}^\star = \lambda_{11}\hat{\boldsymbol{d}}^*, \hat{\boldsymbol{x}}^\star = \hat{\boldsymbol{x}}^*/\lambda_{11})$ is a local minimizer of (4.14) such that $\hat{d}_k^* \hat{\boldsymbol{x}}_{kg}^* = \hat{d}_k^\star \hat{\boldsymbol{x}}_{kg}^\star$ [14, 16].*

The proof of Lemma 4.2.1, along with those of Lemmas 4.2.2 and 4.2.3 below, can be found in the Appendix. Lemma 4.2.1 indicates that the final fitted models from (4.9) and (4.14) are equivalent.

Lemma 4.2.2 *Suppose $(\hat{\boldsymbol{d}}, \hat{\boldsymbol{x}})$ is a local minimizer of (4.14). Let $\hat{\boldsymbol{\beta}}$ be the estimate of hierarchical sparse group lasso related to $(\hat{\boldsymbol{d}}, \hat{\boldsymbol{x}})$, i.e. $\hat{\boldsymbol{\beta}}_{kg} = \hat{d}_k \hat{\boldsymbol{x}}_{kg}$. If $\hat{d}_k = 0$, then $\hat{\boldsymbol{x}}_{(k)} = 0$, where $\hat{\boldsymbol{x}}_{(k)} = [\hat{\boldsymbol{x}}_{k1}^T, \ldots, \hat{\boldsymbol{x}}_{kG_k}^T]^T$ and $\hat{\boldsymbol{x}} = [\hat{\boldsymbol{x}}_{(1)}^T, \ldots, \hat{\boldsymbol{x}}_{(K)}^T]^T$; if $\hat{d}_k \neq 0$, then $||\hat{\boldsymbol{\beta}}_{(k)}||_1 \neq 0$ and $\hat{d}_k = \sqrt{\lambda_1 ||\hat{\boldsymbol{\beta}}_{(k)}||_1}$, $\hat{\boldsymbol{x}}_{(k)} = \frac{\hat{\boldsymbol{\beta}}_{(k)}}{\sqrt{\lambda_1 ||\hat{\boldsymbol{\beta}}_{(k)}||_1}}$, where $\hat{\boldsymbol{\beta}}_{(k)} = [\hat{\boldsymbol{\beta}}_{k1}^T, \ldots, \hat{\boldsymbol{\beta}}_{kG_k}^T]^T$ and $\hat{\boldsymbol{\beta}} = [\hat{\boldsymbol{\beta}}_{(1)}^T, \ldots, \hat{\boldsymbol{\beta}}_{(K)}^T]^T$.*

Lemma 4.2.3 *If $(\hat{\boldsymbol{d}}, \hat{\boldsymbol{x}})$ is a local minimizer of (4.14), then $\hat{\boldsymbol{\beta}}$, where $\hat{\boldsymbol{\beta}}_{kg} = \hat{d}_k \hat{\boldsymbol{x}}_{kg}$, is a local minimizer of*

$$\min_{\boldsymbol{\beta}} \sum_{n=1}^{N} ||y_n - A_n^T \boldsymbol{\beta}||^2 + 2\sqrt{\lambda_1} \sum_{k=1}^{K} \sqrt{\sum_{g=1}^{G_k} ||\boldsymbol{\beta}_{kg}||_1} + \lambda_2 \sum_{k=1}^{K} \sum_{g=1}^{G_k} ||\boldsymbol{\beta}_{kg}||_2 \tag{4.15}$$

Moreover, there exists a local minimizer $(\hat{\boldsymbol{d}}, \hat{\boldsymbol{x}})$ of (4.14) such that (1) if $||\hat{\boldsymbol{\beta}}_{(k)}||_1 = 0$ then $\hat{d}_k = 0$ and $\hat{\boldsymbol{x}}_{(k)} = 0$, and (2) if $||\hat{\boldsymbol{\beta}}_{(k)}||_1 \neq 0$ then $\hat{d}_k = \sqrt{\lambda_1 ||\hat{\boldsymbol{\beta}}_{(k)}||_1}$ and $\hat{\boldsymbol{x}}_k = \frac{\hat{\boldsymbol{\beta}}_{(k)}}{\sqrt{\lambda_1 ||\hat{\boldsymbol{\beta}}_{(k)}||_1}}$.

The penalty function in (4.15) employs the L_1-norm penalty $||\boldsymbol{\beta}_{(k)}||_1 = \sum_{g=1}^{G_k} ||\boldsymbol{\beta}_{kg}||_1$ under each square root. The reformulated HSGL is indeed similar to SGL. Therefore, HSGL not only effectively selects important groups but also removes irrelevant features within a group. Furthermore, the presence of group-norm regularization on each group

k will further strengthen its selection ability, which is an improvement over HL and H-SVM.

4.3 Algorithm

HSGL involves solving a non-linear biconvex optimization problem [16]. Recall that a function $f : X \times Y \to \mathbb{R}$ is called biconvex, if $f(x, y)$ is convex in y for fixed $x \in X$, and $f(x, y)$ is convex in x for fixed $y \in Y$. It is proposed to locally optimize HSGL with alternate convex search (ACS) [11, 16], as summarized in algorithm (2). In essence, the two variables to be solved are optimized in turn until convergence criteria for the variable $\boldsymbol{\beta}_{kg}^{(i)} = d_k^{(i)} \boldsymbol{x}_{kg}^{(i)}$

$$\frac{||\boldsymbol{\beta}||_2 - ||\boldsymbol{\beta}_{\text{old}}||_2}{||\boldsymbol{\beta}_{\text{old}}||_2} < \epsilon \tag{4.16}$$

is met. The solution of the iterative algorithm is guaranteed to converge by Lemma 4.3.1 below.

Lemma 4.3.1 *Let $Q(\lambda_1, \lambda_2, \boldsymbol{d}, \boldsymbol{x})$ denote the value of the objective value of formulation (4.9), and let the two optimization problems in algorithm (2) be solvable. Then the sequence $\{Q(\lambda_1, \lambda_2, \boldsymbol{d_i}, \boldsymbol{x_i})\}_{i \in \mathbb{N}}$ converges to a limit value $a \in \mathbb{R}$ [14, 16].*

It is interesting to note that for HSGL, when $\boldsymbol{d}^{(i)}$ is fixed and $\tilde{\boldsymbol{A}}_{kg} = d_k \boldsymbol{A}_{kg}$, formulation (4.17) is essentially a sparse group lasso optimization problem:

$$\boldsymbol{x}^{(i)} \in \arg\min_{\boldsymbol{x}} \sum_{n=1}^{N} ||y_n - \sum_{k=1}^{K} \sum_{g=1}^{G_k} \tilde{\boldsymbol{A}}_{n,kg}^T \boldsymbol{x}_{kg}||^2 + \lambda_1 ||\boldsymbol{x}||_1 + \lambda_2 \sum_{k=1}^{K} \sum_{g=1}^{G_k} (d_k^{(i)})^2 ||\boldsymbol{x}_{kg}||_2 \tag{4.17}$$

Similarly, when $\boldsymbol{x}^{(i)}$ is fixed and $\hat{\boldsymbol{A}}_{kg} = \boldsymbol{A}_{kg} \boldsymbol{x}_{kg}$, formulation (4.18) is a quadratic programming problem:

Algorithm 2 Solving hierarchical sparse group lasso with alternate convex search

Require: $\mathbf{A} \in \mathbb{R}^{N \times M}$ (data matrix), $\mathbf{y} \in \mathbb{R}^{N \times 1}$ (label), $\lambda_1, \lambda_2 \in \mathbb{R}$ (regularization parameter), index_{layer1}, $\text{index}_{layer2} \in \mathbb{R}^M$ (group assignment of features for layers 1 and 2), $\epsilon \in \mathbb{R}$ (termination parameter).

1: Initialize $\boldsymbol{d}^{(0)} = \mathbf{1}$, $\boldsymbol{x}^{(0)} = \mathbf{1}$, $\beta^{(0)} = \mathbf{1}$.

2: Center $\boldsymbol{y}$ (for regression only). Center and normalize $\boldsymbol{A}_{kg}$.

3: Set $j = 0$.

4: **while** termination criterion (16) is not met **do**

5: Fix $\boldsymbol{d}^{(i)}$, set $\tilde{\boldsymbol{A}}_{kg} = d_k^{(i)} \boldsymbol{A}_{kg}$ and solve for formulation (17).

6: Fix $\boldsymbol{x}^{(i)}$, set $\hat{\boldsymbol{A}}_{(k)} = \sum_{g=1}^{G_k} \boldsymbol{A}_{kg}^T \boldsymbol{x}_{kg}^{(i)}$ and solve for formulation (18).

7: Update $\boldsymbol{\beta}_{kg}^{(i)} = d_k^{(i)} \boldsymbol{x}_{kg}^{(i)}$.

8: Set $i = i + 1$ and go to line 4.

9: **end while**

$$\boldsymbol{d}^{(i)} \in \underset{\boldsymbol{d} \geq 0}{\arg\min} \sum_{n=1}^{N} ||y_n - \sum_{k=1}^{K} d_k \sum_{g=1}^{G_k} \hat{\boldsymbol{A}}_{n,kg}^T||^2 + \sum_{k=1}^{K} d_k + \lambda_2 \sum_{k=1}^{K} \sum_{g=1}^{G_k} d_k^2 ||\boldsymbol{x}_{kg}^{(i)}||_2 \quad (4.18)$$

HSGL with top-level penalization (4.12) or bottom-level penalization (4.13) can be solved similarly with ACS as in algorithm (2).

4.4 Simulation Study

The simulation in [11, 14] was extended. A model that has both categorical and continuous prediction variables was considered. There are two cases – (1) more observations than features, and (2) more features than observations, and under each case, there will be three sub-cases: (1) "All-in-all-out", (2) "Not all-in-all-out" and (3) "Hybrid".

For the case of more (training) observations (400) than features (54), seventeen independent standard normal variables, $Z_1, \ldots, Z_{16}$ and W were generated. The covariates were then defined as $X_j = (Z_j + W)/\sqrt{2}$. Then the last eight covariates $X_9, \ldots, X_{16}$ were discretized to 0, 1, 2, and 3 by $\Phi^{-1}(1/4)$, $\Phi^{-1}(1/2)$, and $\Phi^{-1}(3/4)$. Each of $X_1, \ldots, X_8$ was expanded through a fourth-order polynomial, and only the main effects of $X_9, \ldots, X_{16}$ were considered. Therefore a total of eight continuous groups with four variables in each group and eight categorical groups with three variables in each group were simulated.

The error term ϵ in the above simulations follows a normal distribution N(0, σ^2), where σ^2 was set such that each of the signal to noise ratios, $\mathrm{Var}(\boldsymbol{X}^T\beta)/\mathrm{Var}(\epsilon)$), was equal to 3. For group lasso with L_2-norm (L2-GL), group lasso with L_∞-norm (Linf-GL) and sparse group lasso (SGL), there are two kinds of group assignments:

1. Variable: For the eight continuous groups, polynomials X_i, X_i^2, X_i^3 and X_i^4 are in the same group, whereas for the eight categorical groups, indicators $\mathbb{I}(X_j = 0)$, $\mathbb{I}(X_j = 1)$, and $\mathbb{I}(X_j = 2)$ are in the same group. There will be 16 such groups in this category.
2. Order and discretization: For the eight continuous groups, there are four groups – polynomials X_i, X_i^2, X_i^3, and X_i^4 which correspond to the each polynomial order i, whereas for the eight categorical groups, there are three groups – indicators $\mathbb{I}(X_j = 0)$, $\mathbb{I}(X_j = 1)$ and $\mathbb{I}(X_j = 2)$ which correspond to the three discretization level out of the total four. There will be seven such groups in this category.

For hierarchical lasso (HL) and hierarchical sparse group lasso (HSGL), the top and bottom group assignments are "variable" and "order and discretization".

As for the case of more features (350) than (training) observations (100), artificial data is generated similarly. There are 50 groups of continuous variables with each

group having four variables corresponding to the polynomial order between one and four, and another 50 groups of categorical groups with three variables in each group.

For both cases, there are three types of labels to be considered.

Case 1. "All-in-all-out"

$$\begin{aligned} Y_{\text{regression}} =& [X_3 + 0.5X_3^2 + 0.1X_3^3 + 0.1X_3^4] \\ &+ [X_6 - 0.5X_6^2 + 0.15X_6^3 + 0.1X_6^4] \\ &+ [\mathbb{I}(X_9 = 0) + \mathbb{I}(X_9 = 1) + \mathbb{I}(X_9 = 2)] + \epsilon \\ &\text{(More observations than features)} \\ Y_{\text{regression}} =& [X_3 + 0.5X_3^2 + 0.1X_3^3 + 0.1X_3^4] \\ &+ [X_6 - 0.5X_6^2 + 0.15X_6^3 + 0.1X_6^4] \\ &+ [\mathbb{I}(X_{51} = 0) + \mathbb{I}(X_{51} = 1) + \mathbb{I}(X_{51} = 2)] + \epsilon \\ &\text{(More features than observations)} \end{aligned} \tag{4.19}$$

Case 2. "Not all-in-all-out"

$$\begin{aligned} Y_{\text{regression}} =& [X_3 + X_3^2] + [2X_6 - 1.5X_6^2] \\ &+ [\mathbb{I}(X_9 = 0) + 2\ \mathbb{I}(X_9 = 1)] + \epsilon \end{aligned} \tag{4.20}$$

Case 3. "Hybrid"

$$
\begin{aligned}
Y_{\text{regression}} =& [X_3 + 0.5X_3^2 + 0.1X_3^3 + 0.1X_3^4] \\
&+ [X_6 - 0.15X_6^3] \\
&+ [\mathbb{I}(X_9 = 0) + \mathbb{I}(X_9 = 1) + \mathbb{I}(X_9 = 2)] \\
&+ [\mathbb{I}(X_{12} = 0) - 0.5\ \ \mathbb{I}(X_{12} = 2)] + \epsilon \\
&\text{(More observations than features)} \\
Y_{\text{regression}} =& [X_3 + 0.5X_3^2 + 0.1X_3^3 + 0.1X_3^4] \\
&+ [X_6 - 0.15X_6^3] \\
&+ [\mathbb{I}(X_{51} = 0) + \mathbb{I}(X_{51} = 1) + \mathbb{I}(X_{51} = 2)] \\
&+ [\mathbb{I}(X_{53} = 0) - 0.5\ \ \mathbb{I}(X_{53} = 2)] + \epsilon \\
&\text{(More features than observations)}
\end{aligned}
\tag{4.21}
$$

For all three sub-cases,

$$
Y_{\text{classification}} = \begin{cases} +1, & \text{if } Y_{\text{regression}} \geq \text{median}(Y) \\ -1, & \text{otherwise.} \end{cases}
\tag{4.22}
$$

In addition, there are five performance measures:

$$
\begin{aligned}
\text{Accuracy} &= \frac{\text{No. of correctly classified observations}}{\text{Total number of observations}} \\
&\quad \text{(for classification)} \\
\text{MSE} &= \sqrt{\frac{1}{N}\sum_{n=1}^{N}(y_n - \hat{y}_n)^2} \\
&\quad \text{(for regression)} \\
\text{Zero Var.} &: \text{Percentage of correctly removed} \\
&\quad \text{unimportant variables} \\
\text{Non-zero Var.} &: \text{Percentage of correctly identified} \\
&\quad \text{important variables} \\
\text{Feat. Density} &= \frac{\text{No. of non-zero elements in feat. vector}}{\text{Length of feature vector}}
\end{aligned}
\tag{4.23}
$$

10,000 testing observations were generated. Parameters, if any, were tuned using the training dataset and five-fold cross validation to provide a realistic estimation of prediction errors and to prevent over-fitting. After running all possible combinations of parameters, those with the highest accuracy or lowest mean squared error (MSE) were first chosen. Then, among all the chosen combinations, the one with lowest density of feature vector were chosen to train the model. It is observed that there can be multiple combinations of parameters that can achieve the best learning performance, and that using classification accuracy or MSE alone may not be robust for parameter tuning. The above procedure was repeated 200 times for both the classification and regression experiments. The result are found in tables 4.2 and 4.3 for more observations than features, and in tables 4.4 and 4.5 for more features than observations.

In general, HSGL performed well across all cases, having the ability to remove the unwanted features and to select the relevant features. It is followed by HL, which has a weaker ability of removing unwanted variables as noted in [11], and SGL.

For more observations than features, all the learners (classifiers and regressors) performed comparatively in terms of learning performance (MSE for regression, accuracy for classification). A possible explanation is that there are enough observations for learning generalization. On the other hand, the density of the feature vector, the percentage of correctly removed unimportant variables (Zero Var.), and the percentage of correctly identified important variables (Non-zero Var.) vary a lot across regressors and classifiers. In the case of regression (table 4.2), HSGL has the highest average of "Zero Var." and "Non-zero Var." and the lowest feature density. In the case of classification (table 4.3), SGL produced the most parsimonious model and selected the fewest number of features for satisfactory learning performance. However, this simulated experiment shows empirically that it is not always good to select the fewest number of features as possible, but rather to select all the relevant features. HSGL has the capability of doing so and performed better than HL and other baseline learners, as indicated by its high "Zero Var." and "Non-Zero Var." across different test sub-cases.

For more features than observations, in the case of regression (table 4.4), HSGL has the lowest MSE and feature density and the highest average of "Zero Var." and "Non-Zero Var.", followed by SGL (Variable Grouping), L2-GL (Variable Grouping) and Linf-GL (Variable Grouping). HL performed worse than HSGL in terms of MSE, probably because HL does not have group-norm regularization and is not good at removing lots of unimportant variables when there are more features than observations. As for classification (table 4.5), HSGL had the highest accuracy and the lowest feature density in general. No classifiers obtained 50% of "Non-Zero Var." while maintaining low feature density (e.g. 10% or less), implying that classification with more features than observations is a difficult learning problem.

Table 4.2: Simulation experiments of more observations (400) than features (54) with comparison of several regression-based variable-selection methods, including the hierarchical sparse group lasso (HSGL), hierarchical lasso (HL), sparse group lasso (SGL), L_2-norm group lasso, L_∞-norm group lasso, lasso and ordinary least square (OLS). "MSE" is the mean squared error on the test set. "Zero Var." is the percentage of correctly removed unimportant variables. "Non-zero Var." is the percentage of correctly identified important variables. Density is the percentage of non-zero elements in the feature vector. All of the numbers outside of parentheses are means over 200 repetitions, and the numbers in the parentheses are the corresponding standard errors.

Case 1: All-in-all-out

	HSGL	HL	SGL (Variable Grouping)	SGL (Order/Dis. Grouping)	L2-GL (Variable Grouping)	L2-GL (Order/Dis. Grouping)	Linf-GL (Variable Grouping)	Linf-GL (Order/Dis. Grouping)	Lasso (L1-reg)	OLS
MSE	0.2893 (0.0316)	0.287 (0.0235)	0.2778 (0.0201)	0.2877 (0.0234)	0.2769 (0.0189)	0.2844 (0.0201)	0.2828 (0.0199)	0.3069 (0.0247)	0.2917 (0.0232)	0.3253 (0.0286)
Zero Var.	0.9574 (0.0847)	0.6267 (0.2567)	0.3721 (0.2099)	0.5738 (0.3005)	0.4212 (0.2492)	0.1067 (0.1606)	0.2001 (0.2092)	0.0016 (0.0079)	0.8328 (0.0702)	0.0007 (0.0057)
Non-Zero Var.	0.8395 (0.17)	0.8541 (0.0994)	0.9645 (0.0819)	0.8655 (0.1086)	0.9923 (0.0497)	0.9955 (0.0373)	0.9973 (0.0272)	0.9991 (0.0129)	0.7955 (0.105)	1 (0)
Density	0.1992 (0.081)	0.4692 (0.2188)	0.6945 (0.1787)	0.514 (0.2567)	0.6605 (0.2018)	0.9138 (0.1326)	0.8377 (0.1696)	0.9985 (0.0066)	0.2909 (0.0617)	0.9985 (0.0066)

Case 2: Not All-in-all-out

	HSGL	HL	SGL (Variable Grouping)	SGL (Order/Dis. Grouping)	L2-GL (Variable Grouping)	L2-GL (Order/Dis. Grouping)	Linf-GL (Variable Grouping)	Linf-GL (Order/Dis. Grouping)	Lasso (L1-reg)	OLS
MSE	0.2679 (0.0142)	0.2685 (0.0093)	0.2696 (0.0088)	0.2691 (0.0096)	0.2723 (0.0085)	0.2796 (0.0089)	0.2782 (0.0118)	0.2914 (0.0103)	0.2701 (0.0093)	0.3166 (0.0174)
Zero Var.	0.9656 (0.0686)	0.7498 (0.1333)	0.5603 (0.2439)	0.7064 (0.2067)	0.4016 (0.2306)	0.1131 (0.1511)	0.1107 (0.1354)	0.0023 (0.0126)	0.8369 (0.065)	0.0007 (0.0057)
Non-Zero Var.	0.9282 (0.13)	0.7268 (0.1158)	0.8395 (0.164)	0.7505 (0.1273)	0.9936 (0.046)	0.9973 (0.0287)	0.9936 (0.046)	0.9982 (0.0181)	0.7059 (0.1116)	1 (0)
Density	0.2105 (0.0562)	0.3454 (0.1125)	0.5195 (0.2186)	0.3846 (0.1779)	0.6773 (0.185)	0.909 (0.1212)	0.9084 (0.1113)	0.9978 (0.012)	0.2711 (0.0534)	0.9985 (0.0066)

Case 3: Hybrid

	HSGL	HL	SGL (Variable Grouping)	SGL (Order/Dis. Grouping)	L2-GL (Variable Grouping)	L2-GL (Order/Dis. Grouping)	Linf-GL (Variable Grouping)	Linf-GL (Order/Dis. Grouping)	Lasso (L1-reg)	OLS
MSE	0.2852 (0.0439)	0.2843 (0.0173)	0.279 (0.0158)	0.2841 (0.0174)	0.2779 (0.0168)	0.2847 (0.0166)	0.2853 (0.0166)	0.3026 (0.0179)	0.2901 (0.0205)	0.3218 (0.0241)
Zero Var.	0.8893 (0.0867)	0.5562 (0.2705)	0.3281 (0.2052)	0.4899 (0.3025)	0.3135 (0.193)	0.069 (0.1105)	0.0934 (0.1423)	0.0016 (0.0079)	0.8 (0.0584)	0.0009 (0.0065)
Non-Zero Var.	0.8532 (0.1943)	0.8082 (0.1256)	0.9523 (0.0998)	0.8355 (0.1371)	0.9864 (0.068)	0.9936 (0.046)	0.9936 (0.046)	0.9991 (0.0129)	0.74 (0.1003)	1 (0)
Density	0.2545 (0.0858)	0.5185 (0.2346)	0.7279 (0.1765)	0.5767 (0.2635)	0.7466 (0.1569)	0.9443 (0.0913)	0.9237 (0.1179)	0.9985 (0.0066)	0.307 (0.0515)	0.9985 (0.0066)

Table 4.3: Simulation experiments of more observations (400) than features (54) with comparison of several classification-based variable-selection methods, including the hierarchical sparse group lasso (HSGL), hierarchical lasso (HL), sparse group lasso (SGL), L_2-norm group lasso, L_∞-norm group lasso, lasso and ordinary least square (OLS). Accuracy is the number of correctly classified testing sample divided by the total number of testing samples. "Zero Var." is the percentage of correctly removed unimportant variables. "Non-zero Var." is the percentage of correctly identified important variables. Density is the percentage of non-zero elements in the feature vector. All of the numbers outside of parentheses are means over 200 repetitions, and the numbers in the parentheses are the corresponding standard errors.

Case 1: All-in-all-out

	HSGL	HL	SGL (Variable Grouping)	SGL (Order/Dis. Grouping)	L2-GL (Variable Grouping)	L2-GL (Order/Dis. Grouping)	Linf-GL (Variable Grouping)	Linf-GL (Order/Dis. Grouping)	Lasso (L1-reg)	OLS
Accuracy	0.7754 (0.032)	0.7619 (0.0342)	0.7602 (0.0284)	0.7523 (0.025)	0.765 (0.0275)	0.7636 (0.0273)	0.7643 (0.0276)	0.7441 (0.0245)	0.7699 (0.0313)	0.7524 (0.0289)
Zero Var.	0.9287 (0.1333)	0.496 (0.3538)	0.9717 (0.0548)	0.915 (0.098)	0.1071 (0.2594)	0.0574 (0.132)	0.0729 (0.1546)	0.0113 (0.0384)	0.797 (0.1359)	0 (0)
Non-Zero Var.	0.8364 (0.1888)	0.7009 (0.285)	0.4686 (0.1894)	0.2986 (0.1634)	0.9841 (0.0737)	0.9923 (0.0435)	0.9955 (0.0285)	0.9818 (0.0531)	0.5332 (0.1898)	0.9959 (0.0246)
Density	0.222 (0.1155)	0.5404 (0.3368)	0.1142 (0.0644)	0.1267 (0.1075)	0.885 (0.2388)	0.9506 (0.1153)	0.9661 (0.1042)	0.984 (0.0467)	0.2704 (0.1409)	0.999 (0.0054)

Case 2: Not All-in-all-out

	HSGL	HL	SGL (Variable Grouping)	SGL (Order/Dis. Grouping)	L2-GL (Variable Grouping)	L2-GL (Order/Dis. Grouping)	Linf-GL (Variable Grouping)	Linf-GL (Order/Dis. Grouping)	Lasso (L1-reg)	OLS
Accuracy	0.7794 (0.06)	0.7531 (0.0334)	0.7699 (0.0264)	0.76 (0.028)	0.7667 (0.0225)	0.7655 (0.0227)	0.7663 (0.0221)	0.744 (0.0291)	0.7731 (0.0255)	0.7546 (0.0221)
Zero Var.	0.9452 (0.109)	0.6572 (0.3689)	0.9584 (0.0573)	0.8905 (0.0913)	0.1071 (0.2594)	0.0574 (0.132)	0.0729 (0.1546)	0.0113 (0.0384)	0.813 (0.1271)	0 (0)
Non-Zero Var.	0.8823 (0.1931)	0.56 (0.3242)	0.6314 (0.1879)	0.4341 (0.1534)	0.9841 (0.0737)	0.9923 (0.0435)	0.9955 (0.0285)	0.9818 (0.0531)	0.6009 (0.1458)	0.9959 (0.0246)
Density	0.218 (0.0996)	0.3811 (0.3534)	0.1583 (0.0639)	0.1745 (0.0951)	0.9131 (0.2083)	0.9547 (0.1018)	0.9432 (0.1226)	0.9869 (0.0451)	0.2711 (0.1236)	0.999 (0.0054)

Case 3: Hybrid

	HSGL	HL	SGL (Variable Grouping)	SGL (Order/Dis. Grouping)	L2-GL (Variable Grouping)	L2-GL (Order/Dis. Grouping)	Linf-GL (Variable Grouping)	Linf-GL (Order/Dis. Grouping)	Lasso (L1-reg)	OLS
Accuracy	0.7889 (0.0359)	0.775 (0.0414)	0.7701 (0.0345)	0.7644 (0.0363)	0.774 (0.0333)	0.7731 (0.0325)	0.7744 (0.0331)	0.755 (0.0356)	0.7802 (0.0356)	0.7652 (0.0327)
Zero Var.	0.8731 (0.1104)	0.4771 (0.3106)	0.8808 (0.0769)	0.7812 (0.1155)	0.1073 (0.2593)	0.0577 (0.132)	0.0731 (0.1545)	0.0115 (0.0385)	0.7341 (0.1089)	0.0002 (0.0033)
Non-Zero Var.	0.86 (0.1854)	0.7359 (0.2397)	0.5864 (0.1869)	0.4955 (0.1609)	0.9836 (0.0739)	0.9918 (0.0439)	0.995 (0.0291)	0.9814 (0.0534)	0.6105 (0.1448)	0.9955 (0.0254)
Density	0.2686 (0.1005)	0.5577 (0.2861)	0.2096 (0.0842)	0.2743 (0.1176)	0.8336 (0.2777)	0.9478 (0.1137)	0.9292 (0.1283)	0.9904 (0.0342)	0.3339 (0.1073)	0.999 (0.0054)

Table 4.4: Simulation experiments of more features (350) than observations (100) with comparison of several regression-based variable-selection methods, including the hierarchical sparse group lasso (HSGL), hierarchical lasso (HL), sparse group lasso (SGL), L_2-norm group lasso, L_∞-norm group lasso, lasso and ordinary least square (OLS). "MSE" is the mean squared error on the test set. "Zero Var." is the percentage of correctly removed unimportant variables. "Non-zero Var." is the percentage of correctly identified important variables. Density is the percentage of non-zero elements in the feature vector. All of the numbers outside of parentheses are means over 200 repetitions, and the numbers in the parentheses are the corresponding standard errors.

Case 1: All-in-all-out

	HSGL	HL	SGL (Variable Grouping)	SGL (Order/Dis. Grouping)	L2-GL (Variable Grouping)	L2-GL (Order/Dis. Grouping)	Linf-GL (Variable Grouping)	Linf-GL (Order/Dis. Grouping)	Lasso (L1-reg)	OLS
MSE	0.3718 (0.0638)	0.5918 (0.1104)	0.3928 (0.0687)	0.4423 (0.0838)	0.3925 (0.0525)	0.5202 (0.1134)	0.3972 (0.0569)	1.0995 (0.2296)	0.4343 (0.0728)	1.3112 (0.1413)
Zero Var.	0.9938 (0.0134)	0.7735 (0.0259)	0.855 (0.1109)	0.9117 (0.1368)	0.819 (0.1278)	0.3185 (0.1966)	0.7838 (0.1363)	0.008 (0.0245)	0.9531 (0.0387)	0.0037 (0.0052)
Non-Zero Var.	0.6123 (0.1983)	0.5805 (0.1471)	0.7182 (0.2704)	0.4818 (0.1688)	0.7941 (0.2203)	0.9682 (0.0963)	0.7932 (0.2068)	0.9936 (0.0335)	0.4268 (0.1553)	0.9991 (0.0129)
Density	0.0252 (0.0156)	0.2376 (0.0254)	0.163 (0.1112)	0.1007 (0.1352)	0.2002 (0.1256)	0.6905 (0.1918)	0.2344 (0.1342)	0.9921 (0.0247)	0.0588 (0.0395)	0.9963 (0.0051)

Case 2: Not All-in-all-out

	HSGL	HL	SGL (Variable Grouping)	SGL (Order/Dis. Grouping)	L2-GL (Variable Grouping)	L2-GL (Order/Dis. Grouping)	Linf-GL (Variable Grouping)	Linf-GL (Order/Dis. Grouping)	Lasso (L1-reg)	OLS
MSE	0.3506 (0.0607)	0.5344 (0.0934)	0.3762 (0.0516)	0.4106 (0.0761)	0.3812 (0.0454)	0.5315 (0.0896)	0.4201 (0.0514)	1.1276 (0.2088)	0.3926 (0.0541)	1.2702 (0.1308)
Zero Var.	0.9937 (0.0136)	0.7756 (0.0262)	0.8226 (0.1105)	0.9096 (0.1285)	0.7978 (0.1123)	0.2654 (0.1961)	0.781 (0.1302)	0.0108 (0.0311)	0.9339 (0.0399)	0.0037 (0.0052)
Non-Zero Var.	0.6645 (0.2128)	0.5605 (0.1408)	0.8468 (0.2101)	0.535 (0.1708)	0.8768 (0.175)	0.9895 (0.0517)	0.8632 (0.1852)	0.99 (0.0416)	0.4995 (0.1397)	0.9991 (0.0129)
Density	0.027 (0.0158)	0.2349 (0.026)	0.1985 (0.108)	0.1044 (0.1278)	0.2234 (0.1094)	0.7426 (0.1904)	0.2392 (0.1269)	0.9892 (0.0314)	0.0797 (0.0403)	0.9963 (0.0051)

Case 3: Hybrid

	HSGL	HL	SGL (Variable Grouping)	SGL (Order/Dis. Grouping)	L2-GL (Variable Grouping)	L2-GL (Order/Dis. Grouping)	Linf-GL (Variable Grouping)	Linf-GL (Order/Dis. Grouping)	Lasso (L1-reg)	OLS
MSE	0.4303 (0.0874)	0.6243 (0.124)	0.4534 (0.0768)	0.5021 (0.0939)	0.4434 (0.0743)	0.6139 (0.1049)	0.4913 (0.101)	1.2139 (0.157)	0.4894 (0.0779)	1.3934 (0.0913)
Zero Var.	0.9819 (0.0243)	0.7781 (0.0452)	0.8265 (0.1229)	0.8999 (0.1414)	0.7847 (0.1217)	0.2349 (0.1712)	0.7413 (0.1419)	0.0087 (0.0259)	0.9304 (0.0442)	0.0037 (0.0052)
Non-Zero Var.	0.5668 (0.2124)	0.5386 (0.1358)	0.68 (0.2646)	0.4386 (0.1631)	0.7923 (0.2182)	0.9977 (0.0231)	0.7755 (0.2325)	0.9927 (0.0357)	0.4068 (0.1412)	0.9991 (0.0129)
Density	0.0353 (0.0271)	0.2318 (0.0456)	0.1894 (0.123)	0.1107 (0.1405)	0.2334 (0.1206)	0.7724 (0.1658)	0.2749 (0.1408)	0.9913 (0.0261)	0.0802 (0.0449)	0.9963 (0.0051)

Table 4.5: Simulation experiments of more features (350) than observations (100) with comparison of several classification-based variable-selection methods, including the hierarchical sparse group lasso (HSGL), hierarchical lasso (HL), sparse group lasso (SGL), L_2-norm group lasso, L_∞-norm group lasso, lasso and ordinary least square (OLS). Accuracy is the number of correctly classified testing sample divided by the total number of testing samples. "Zero Var." is the percentage of correctly removed unimportant variables. "Non-zero Var." is the percentage of correctly identified important variables. Density is the percentage of non-zero elements in the feature vector. All of the numbers outside of parentheses are means over 200 repetitions, and the numbers in the parentheses are the corresponding standard errors.

Case 1: All-in-all-out

	HSGL	HL	SGL (Variable Grouping)	SGL (Order/Dis. Grouping)	L2-GL (Variable Grouping)	L2-GL (Order/Dis. Grouping)	Linf-GL (Variable Grouping)	Linf-GL (Order/Dis. Grouping)	Lasso (L1-reg)	OLS
Accuracy	0.7172 (0.0415)	0.6388 (0.0534)	0.7178 (0.0357)	0.7089 (0.0433)	0.5578 (0.2223)	0.6074 (0.0328)	0.616 (0.041)	0.5707 (0.0338)	0.6937 (0.0473)	0.6109 (0.0336)
Zero Var.	0.7801 (0.0266)	0.5815 (0.0878)	0.7738 (0.0349)	0.7427 (0.0798)	0.4723 (0.2686)	0.0239 (0.0641)	0.2105 (0.1503)	0.0046 (0.0142)	0.7514 (0.0532)	0.0029 (0.0038)
Non-Zero Var.	0.3627 (0.1818)	0.3464 (0.1419)	0.2855 (0.1653)	0.1814 (0.1155)	0.6023 (0.3658)	0.99 (0.0761)	0.8527 (0.1826)	0.9959 (0.0263)	0.1941 (0.1111)	0.9986 (0.0143)
Density	0.0305 (0.0347)	0.2703 (0.109)	0.035 (0.0446)	0.0686 (0.1002)	0.4126 (0.3369)	0.9707 (0.0787)	0.7393 (0.1849)	0.9943 (0.0181)	0.0586 (0.0669)	0.9964 (0.0048)

Case 2: Not All-in-all-out

	HSGL	HL	SGL (Variable Grouping)	SGL (Order/Dis. Grouping)	L2-GL (Variable Grouping)	L2-GL (Order/Dis. Grouping)	Linf-GL (Variable Grouping)	Linf-GL (Order/Dis. Grouping)	Lasso (L1-reg)	OLS
Accuracy	0.7232 (0.0416)	0.6354 (0.0499)	0.7089 (0.0446)	0.6891 (0.0508)	0.5906 (0.1937)	0.6097 (0.0337)	0.6135 (0.0371)	0.5682 (0.031)	0.6834 (0.0513)	0.6098 (0.0342)
Zero Var.	0.7795 (0.0221)	0.5854 (0.0897)	0.7634 (0.0462)	0.7333 (0.0819)	0.5156 (0.273)	0.0309 (0.0714)	0.1922 (0.1359)	0.0057 (0.0183)	0.7401 (0.0565)	0.0029 (0.0038)
Non-Zero Var.	0.3586 (0.1936)	0.3759 (0.1375)	0.3323 (0.1891)	0.1977 (0.1445)	0.5741 (0.3517)	0.99 (0.0761)	0.8836 (0.1708)	0.9941 (0.0317)	0.2259 (0.1317)	0.9986 (0.0143)
Density	0.0311 (0.0304)	0.2668 (0.1112)	0.0495 (0.0593)	0.0807 (0.1034)	0.359 (0.3413)	0.9621 (0.0874)	0.7628 (0.1667)	0.9929 (0.0233)	0.0736 (0.0721)	0.9964 (0.0048)

Case 3: Hybrid

	HSGL	HL	SGL (Variable Grouping)	SGL (Order/Dis. Grouping)	L2-GL (Variable Grouping)	L2-GL (Order/Dis. Grouping)	Linf-GL (Variable Grouping)	Linf-GL (Order/Dis. Grouping)	Lasso (L1-reg)	OLS
Accuracy	0.7071 (0.0456)	0.6336 (0.0523)	0.6943 (0.0466)	0.6752 (0.054)	0.6063 (0.163)	0.6099 (0.0349)	0.6184 (0.0447)	0.5615 (0.0229)	0.6711 (0.0549)	0.6032 (0.029)
Zero Var.	0.7715 (0.0336)	0.5806 (0.0877)	0.7508 (0.0509)	0.7353 (0.0753)	0.5127 (0.2712)	0.0337 (0.0811)	0.2201 (0.1729)	0.004 (0.012)	0.7257 (0.0603)	0.0029 (0.0038)
Non-Zero Var.	0.3459 (0.1922)	0.3605 (0.1439)	0.2932 (0.1771)	0.1995 (0.1426)	0.5423 (0.3449)	0.985 (0.0933)	0.8436 (0.1949)	0.9968 (0.023)	0.2155 (0.1186)	0.9986 (0.0143)
Density	0.0403 (0.0442)	0.2719 (0.1092)	0.0632 (0.0642)	0.0783 (0.0955)	0.3613 (0.3391)	0.9586 (0.0995)	0.7273 (0.2132)	0.995 (0.0152)	0.0906 (0.0758)	0.9964 (0.0048)

4.5 Real-Data Analysis

As the correct features related to the learning target are not known beforehand, the goal of real-data analysis is therefore not to compare the density of the feature vector $\boldsymbol{x}$ but to show that HSGL has comparable learning performance when compared with other baseline learners.

In this section, the Breast Cancer Wisconsin (Diagnostic) Dataset (WDBC) [17, 18] is used because (1) it can be easily obtained at the Machine Learning Repository, the University of California at Irvine, (2) it was preprocessed and (3) features can be presented in a hierarchical structure in order to run HSGL. Features were determined from a digitized image of a fine needle aspirate of a breast mass, which describes the characteristics of the cell nuclei present in the image. For each cell nucleus, ten features - radius, texture, perimeter, area, smoothness, compactness, concavity, number of concave points, symmetry and fractal dimension - were generated. To summarize, there are 569 observations and 30 features in this dataset.

For group lasso with L_2-norm (L2-GL), group lasso with L_∞-norm (Linf-GL) and sparse group lasso (SGL), there are two kinds of group assignments:

1. Cell Nucleus: There are three such groups in this category corresponding to the three cell nucleus in this case.
2. Feature Type: There are ten such groups in this category as described.

For hierarchical lasso (HL) and hierarchical sparse group lasso (HSGL), the top and bottom group assignments are cell nucleus-feature type or feature type-cell nucleus.

A double five-fold cross validation was used for predicting the out-of-sample error. Data is divided to into five random subsets. One of the subsets is used as testing, and the remaining four subsets are used for training, and such procedure is repeated for five times. Parameters, if any, were tuned using the training dataset using another

five-fold cross validation to provide a realistic estimation of prediction errors and to prevent over-fitting. After running all possible combinations of parameters, those with the highest accuracy were first chosen. Then, among all the chosen combinations, the one with lowest density of feature vector was chosen to train the model. The above procedure was repeated for 100 times. The result can be found in table 4.6.

In general, the accuracy across different classifiers remains similar, but the density of feature vectors varies substantially. HSGL excels in both accuracy and feature density, followed again by HL and SGL. The feature density for SGL using assignment of feature type can be as low as 0.24. However, as illustrated in the previous simulation experiment, that may not account for all but just some of the relevant features regarding this learning problem. On the other hand, since there can be useful variables in each group, L2-GL and Linf-GL are not able to remove unimportant variables inside the important group, resulting in a high feature density. Surprisingly, lasso as a simple method performs well.

This experiment illustrates the importance of choosing the appropriate top and bottom group assignments for HSGL and HL. The difference of accuracy of HSGL using 1) cell nucleus-feature type and 2) feature type-cell nucleus is 0.03 (0.95 vs. 0.92). Therefore, it is important to try different hierarchical group assignments in order to find an optimal hierarchy of arranging the features for a particular problem.

Table 4.6: Comparison of several classifiers on WDBC dataset, including the hierarchical sparse group lasso (HSGL), hierarchical lasso (HL), sparse group lasso (SGL), L_2-norm group lasso, L_∞-norm group lasso, lasso and ordinary least square (OLS). "Acc." is the number of correctly classified testing sample divided by the total number of testing samples. "Den." is the percentage of non-zero elements in the feature vector. All of the numbers outside of parentheses are means over 100 repetitions, and the numbers in the parentheses are the corresponding standard errors.

	HSGL (Cell Nucleus -Feat. Type)	HSGL (Feat. Type- Cell Nucleus)	HL (Cell Nucleus -Feat. Type)	HL (Feat. Type- Cell Nucleus)	SGL (Cell Nucleus)	SGL (Feat. Type)	L2-GL (Cell Nucleus)	L2-GL (Feat. Type)	Linf-GL (Cell Nucleus)	Linf-GL (Feat. Type)	Lasso (L1-reg)	OLS
Acc.	0.9512 (0.0046)	0.9237 (0.0098)	0.9572 (0.0019)	0.9525 (0.0032)	0.9470 (0.0021)	0.948 (0.0011)	0.9489 (0.0021)	0.9509 (0.002)	0.9488 (0.0021)	0.9543 (0.0031)	0.9575 (0.0007)	0.938 (0.0018)
Den.	0.4967 (0.0396)	0.1787 (0.013)	0.5487 (0.04)	0.7426 (0.0606)	0.444 (0.0217)	0.2407 (0.0097)	1 (0)	0.9738 (0.0352)	1 (0)	0.9682 (0.0215)	0.3032 (0.0123)	1 (0)

Table 4.7: Average density of feature groups selected by hierarchical sparse group lasso (HSGL), hierarchical lasso (HL) and sparse group lasso (SGL) in the experiment with the WDBC dataset. For HSGL and HL, the hierarchical structure is cell nucleus (top) - feature type (bottom), and for SGL, the group assignment can either be cell nucleus or feature type. Density is calculated using the mean of the density of each group assignment from the five-fold experiment over 100 repetitions. All of the numbers outside of parentheses are means over 100 repetitions, and the numbers in the parentheses are the corresponding standard errors. This table is to be interpreted together with table 4.8.

	Group: Cell Nucleus			Group: Feature Type									
	Cell Nucleus 1	Cell Nucleus 2	Cell Nucleus 3	Radius	Texture	Perimeter	Area	Smoothness	Compactness	Concavity	No. of Concave Points	Symmetry	Fractal Dimension
HSGL Den.	0.652 (0.477)	0.33 (0.471)	1 (0)	0.459 (0.229)	0.44 (0.262)	0.387 (0.179)	0.475 (0.222)	0.501 (0.284)	0.597 (0.401)	0.515 (0.42)	0.575 (0.245)	0.463 (0.217)	0.555 (0.235)
HL Den.	0.028 (0.165)	0.618 (0.486)	1 (0)	0.601 (0.194)	0.601 (0.194)	0.601 (0.194)	0.601 (0.194)	0.601 (0.194)	0.601 (0.194)	0.601 (0.194)	0.601 (0.194)	0.601 (0.194)	0.601 (0.194)
SGL Den.	0.435 (0.404)	0 (0)	0.897 (0.036)	1 (0)	0.667 (0)	0.074 (0.262)	0 (0)	0 (0)	0 (0)	0 (0)	0.667 (0)	0 (0)	0 (0)

Table 4.8: The accuracy of OLS using features from a specific group assignment with 5-fold cross-validation for the WDBC dataset. All of the numbers outside of parentheses are means over 100 repetitions of a five-fold cross-validated experiment, and the numbers in the parentheses are the corresponding standard errors. This table is to be interpreted together with table 4.7.

	Group: Cell Nucleus			Group: Feature Type									
	Cell Nucleus 1	Cell Nucleus 2	Cell Nucleus 3	Radius	Texture	Perimeter	Area	Smoothness	Compactness	Concavity	No. of Concave Points	Symmetry	Fractal Dimension
OLS Acc.	0.916 (0.001)	0.804 (0.001)	0.957 (0.001)	0.881 (0.001)	0.744 (0.001)	0.887 (0.001)	0.853 (0.001)	0.75 (0.002)	0.833 (0.001)	0.883 (0.001)	0.918 (0.001)	0.723 (0.002)	0.739 (0.002)

Finally, table 4.7 shows the average density of feature groups selected by HSGL, HL and SGL in the five-fold cross-validated experiment with 100 repetitions. For the group assignment of cell nucleus, cell nucleus 3 always has high feature density (HSGL: 1, HL: 1, SGL: 0.897) regardless the choice of classifier, showing that it is an important group for further interpretation. Cell nucleus 1 and 2 were not always selected and have high standard deviation of feature density. In the case of SGL, features from cell nucleus 2 were not selected at all for all 100 repetitions. On the other hand, for group assignment of feature type, four feature groups - radius (OLS acc.: 0.881), texture (OLS acc.: 0.744), perimeter (OLS acc.: 0.887) and number of concave points (OLS acc.: 0.918) - were selected by all three classifiers, whereas the six feature groups - area (OLS acc.: 0.853), smoothness (OLS acc.: 0.75), compactness (OLS acc.: 0.833), concavity (OLS acc.: 0.883), symmetry (OLS acc.: 0.723) and fractal dimension (OLS acc.: 0.739) - were not selected by SGL. This does not mean these six groups are unimportant to this learning problem. Indeed, according to the simulation study in the above, SGL may not be able to select features (or feature groups) that are relevant to the learning target. Those six feature groups may be relevant to the learning target and are subject to careful confirmation and interpretation by experts with prior knowledge in the subject area.

4.6 Concluding Remark

In this research, hierarchical sparse group lasso (HSGL) is proposed to improve hierarchical lasso, which cannot effectively remove all unimportant variables within important groups. Lemma 4.2.3 shows that HSGL is similar to sparse group lasso and can address the issue of HL in light of the results in both simulation and real-data experiments. Moreover, HSGL can be used as both a learner (for classification or regression) and a feature selector.

In order to apply HSGL on real datasets, users need to be aware of several issues. To begin with, the group-norm penalization at both top and bottom levels might be too strong and can result in inferior learning performance. Interested readers are encouraged to (1) try different hierarchical group assignment and (2) try penalization at top level (4.12), bottom level (4.13) or both (4.14) in order to seek the best hierarchical structure of features for a particular problem. Another issue is the interpretation of the selected features. Interested readers are encouraged to run both SGL and HSGL (or SGL, HL and HSGL if computational resources are allowed). From the classification results of simulated experiments, well-tuned SGL is likely to select fewer features than HSGL. Feature groups selected by both SGL and HSGL can be given more attention, while feature groups selected by HSGL but not SGL may be considered as well, but they require careful evaluation on their relevance to the learning problem.

Future directions include the change of loss function such as logistic loss for multi-class classification or hinge loss resulting in hierarchical sparse group support vector machine. HSGL can also be kernalized for non-linear decision boundaries. Finally, the mathematical analysis on two-layer HSGL can be extended to n-layer case.

Bibliography

[1] Yvan Saeys, Iñaki Inza, and Pedro Larrañaga. A review of feature selection techniques in bioinformatics. *bioinformatics*, 23(19):2507–2517, 2007.

[2] Benson Mwangi, Tian Siva Tian, and Jair C Soares. A review of feature reduction techniques in neuroimaging. *Neuroinformatics*, 12(2):229–244, 2014.

[3] UK Devika, Sheeba Babu, and Jubilant J Kizhakkethottam. Review on feature selection methods in high dimensional domains. In *Soft-Computing and Networks Security (ICSNS), 2015 International Conference on*, pages 1–3. IEEE, 2015.

[4] Jennifer G Dy and Carla E Brodley. Feature selection for unsupervised learning. *Journal of machine learning research*, 5(Aug):845–889, 2004.

[5] Hanchuan Peng, Fuhui Long, and Chris Ding. Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on pattern analysis and machine intelligence*, 27(8):1226–1238, 2005.

[6] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

[7] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[8] Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320, 2005.

[9] Ming Yuan and Yi Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(1):49–67, 2006.

[10] Peng Zhao, Guilherme Rocha, and Bin Yu. Grouped and hierarchical model selection through composite absolute penalties. *Department of Statistics, UC Berkeley, Tech. Rep*, 703, 2006.

[11] Nengfeng Zhou and Ji Zhu. Group variable selection via a hierarchical lasso and its oracle property. *arXiv preprint arXiv:1006.2871*, 2010.

[12] Noah Simon, Jerome Friedman, Trevor Hastie, and Robert Tibshirani. A sparse-group lasso. *Journal of Computational and Graphical Statistics*, 22(2):231–245, 2013.

[13] Jun Liu, Shuiwang Ji, Jieping Ye, et al. Slep: Sparse learning with efficient projections. *Arizona State University*, 6(491):7, 2009.

[14] Sungwan Bang, Jongkyeong Kang, Myoungshic Jhun, and Eunkyung Kim. Hierarchically penalized support vector machine with grouped variables. *International Journal of Machine Learning and Cybernetics*, 8(4):1211–1221, 2017.

[15] Seyoung Kim and Eric P Xing. Tree-guided group lasso for multi-task regression with structured sparsity. 2010.

[16] Jochen Gorski, Frank Pfeuffer, and Kathrin Klamroth. Biconvex sets and optimization with biconvex functions: a survey and extensions. *Mathematical Methods of Operations Research*, 66(3):373–407, 2007.

[17] W Nick Street, William H Wolberg, and Olvi L Mangasarian. Nuclear feature extraction for breast tumor diagnosis. In *Biomedical Image Processing and Biomedical Visualization*, volume 1905, pages 861–871. International Society for Optics and Photonics, 1993.

[18] Olvi L Mangasarian, W Nick Street, and William H Wolberg. Breast cancer diagnosis and prognosis via linear programming. *Operations Research*, 43(4):570–577, 1995.

CHAPTER 5

Emotion Recognition and Analysis Using Hierarchical Sparse Group Lasso

Abstract

This paper presents the use of hierarchical sparse group lasso (HSGL) on an EEG-based emotion recognition problem. It is well known that features in EEG-based studies can often be arranged in a hierarchical structure. In this regard, HSGL exploits such property in order to further investigate the emotion recognition problem on hand. Similar to SGL, HSGL does feature selection and classification at the same time. Besides, HSGL is easy to interpret, is computationally efficient, and has comparable classification accuracy when compared with other baseline classifiers. The experimental result and its associated interpretation can be helpful for researchers in identifying discriminating features in order to better design and build a light-weighted emotion recognition system.

Keywords: emotion recognition, multi-modal emotion processing, supervised learning, feature selection, electroencephalogram, EEG, group structure learning.

5.1 Introduction

In the context of machine learning, emotion recognition is a classification problem under supervised learning. A typical classification workflow includes data collection, data preprocessing, feature extraction, feature selection (if any), and classification. In other words, innovations of any emotion recognition system take place at one or

more of these steps. There has been many attempts of building emotion recognition systems before. Interested readers are referred to recent reviews [1, 2, 3, 4].

A good emotion recognition system requires generalization ability, computational efficiency and preferably interpretability. In order to do so, a light-weighted classifier using a handful of mathematical and time-series features is desired. There are several hurdles for achieving these three goals.

In most of the surveyed literature on non-deep learning approaches, the number of features is probably more than the number of observations in the data matrix. While it is easy to extract features from data, it is difficult to obtain observations/samples from more subjects, among other reasons (e.g. availability of funding, approval of ethics committee). A good emotion recognition system under laboratory settings should handle this situation well, unless data can be collected from a lot of experiment subjects or data from different experiments can be unified, which is unlikely. Nevertheless, there has been successful implementations of emotion recognition system which generalizes well under laboratory settings. In this regard, typical classifiers include linear discriminant analysis [5], support vector machine [6, 7, 8], k-nearest neighbor [9], and hidden markov model [10, 11]. Tree-based and neural-network classifiers are not usually used because they easily over-fit the training data under such situation.

Another hurdle is the ability of choosing only the essential features for model building. Typical non-neural network approach in machine learning after feature extraction would be to apply one of three methods in feature selection - wrapper, filter and embedded methods - before applying a classifier model. Wrapper and filter methods may not use classification accuracy as the selection criteria. Hence, satisfactory learning performance is not guaranteed. Embedded methods allow the use of classification accuracy (or other appropriate performance measures such as F1-score) in parameter tuning. It may lead to over-fitting because the set of parameters with

the best performance score will be chosen. However, such problem can be mitigated, if not solved completely, via a wider range of parameters during tuning and a higher fold of cross validation. The name of embedded method originates from the fact that such method is an embedded learner (either classifier or regressor). Sparse group lasso [12], hierarchical lasso [13] as well as hierarchical sparse group lasso applied in this work, all fall under the category of embedded method.

Last but not least, without computational efficiency, parameters cannot be tuned within acceptable amount of time. Computational efficiency of a classifier depends on the choice of optimization algorithm, algorithmic complexity, implementation quality and programming environment of the project, among other factors. Although CVX [14] is not a good package for large-data processing, it is especially good for rapid prototyping for formulations of linear programming (LP), quadratic programming (QP) and semi-definite programming (SDP). IBM ILOG CPLEX and Gurobi are commercial-grade optimization solvers which are well recognized in both industrial and academic communities. Sparse Learning with Efficient Projects (SLEP) toolbox [15] and Inria SPAMS (SPArse Modeling Software) [16] implements many sparse group learning methods very well such as sparse group lasso and tree structure group lasso. As in chapter 4, HSGL is implemented with IBM ILOG CPLEX and SLEP Toolbox, and is computational efficient.

On the other hand, sparse learning refers to the use of L_1-norm ($||\boldsymbol{\beta}||_1 = \sum_i |\beta_i|$) on the learning model ($\boldsymbol{\beta}$), whereas group learning [17] refers to the use of group norm ($||\boldsymbol{\beta}_k||_2 = \sqrt{\beta_{k1}^2 + \cdots + \beta_{kG_k}^2}$, where k refers to the kth group in the group structure) on the learning model.

Lasso (least absolute shrinkage and selection operator) [18] was originally developed as a regression analysis method. It minimizes the usual sum of squared errors, with a bound on the sum of the absolute values of the coefficients. Its decision function

can be amended so that classification can be solved via lasso. The regularization component in the formulation of lasso is an effective way to alleviate the problem of over-fitting. Lasso is closely related to statistical models such as generalized linear model and soft-thresholding.

Lasso is later generalized to a lot of its variants such as elastic net [19] and group lasso [20]. Group lasso consists of predefined groups of covariates regularized by L_2-norm, i.e.

$$||\boldsymbol{\beta}||_2 = \sqrt{\beta_1^2 + \cdots + \beta_n^2} \tag{5.1}$$

According to [13], appropriately tuning the coefficient of the L_2-norm can effectively set the whole coefficient vector $\boldsymbol{\beta}$ to be zero because of the singularity of $||\boldsymbol{\beta}||_2$ at $\boldsymbol{\beta} = \mathbf{0}$. Therefore, the features in a particular group can be either selected or excluded completely. To select variables inside a group (instead of choosing all in a group as in group lasso), sparse group lasso [12] is proposed by having one term regularized by the L_1-norm, i.e.,

$$||\boldsymbol{\beta}||_1 = |\beta_1| + \cdots + |\beta_n| \tag{5.2}$$

SGL enforces sparse effects both on a group and within-group level of features. There have been many successful implementation using accelerated gradient descent [15] and blockwise coordinate descent [20]. The formulation of SGL is as follows:

$$\min_{\boldsymbol{\beta}} \ ||\boldsymbol{y} - \boldsymbol{A}\boldsymbol{\beta}||_2 + \lambda_1||\boldsymbol{\beta}||_1 + \lambda_2 \sum_{g=1}^{G} ||\boldsymbol{\beta}_g||_2 \tag{5.3}$$

Close variants of HSGL include hierarchical lasso (HL) [13], hierarchically penalized support vector machine (H-SVM) [21] and tree structured group lasso (TSGL) [15, 22]. Zhou and Zhu [13] proposed a two-layer hierarchical lasso (HL) with

least-square loss as follows, where there are K groups in the top layer, and under the kth top group, there are G_k sub-groups in the bottom layer:

$$\min_{\boldsymbol{d}\geq 0,\boldsymbol{x}} \sum_{n=1}^{N} ||y_n - \sum_{k=1}^{K} d_k \sum_{g=1}^{G_k} \boldsymbol{A}_{n,kg}^T \boldsymbol{x}_{kg}||^2 + \lambda_1 \sum_{k=1}^{K} d_k + \lambda_2 ||\boldsymbol{x}||_1 \tag{5.4}$$

H-SVM replaces the least-square loss in HL with hinge loss $(\bullet)_+$:

$$\min_{\boldsymbol{d}\geq 0,\boldsymbol{x}} \sum_{n=1}^{N} [1 - y_i(x_0 + \sum_{k=1}^{K} d_k \boldsymbol{A}_{n,kg}^T \boldsymbol{x}_{kg})]_+ + \lambda_1 \sum_{k=1}^{K} d_k + \lambda_2 ||\boldsymbol{x}||_1 \tag{5.5}$$

For TSGL, the hierarchical structure of the features is represented as a tree with leaf nodes as features and internal nodes as clusters of features. The formulation of TSGL is as follows, where $\boldsymbol{\beta} \in \mathbb{R}^{N\times 1}$ forms a certain tree structure and w_j^i is a weight at jth node of depth i (see figure 4.1 for illustration):

$$\min_{\boldsymbol{\beta}} \sum_{n=1}^{N} ||y_n - \boldsymbol{A}_n^T \boldsymbol{\beta}||^2 + \lambda \sum_{i=0}^{I} \sum_{j=1}^{J_i} w_j^i ||\boldsymbol{\beta}|| \tag{5.6}$$

TSGL allows variation for the number of features in each group and at each layer, whereas l-layer HL, HSVM and HSGL always have l group assignments for each feature.

In short, TSGL, H-SVM, HL and HSGL all take hierarchical features as input. TSGL outputs a single feature vector that can be presented with the same hierarchical tree structure as the input, whereas HL, H-SVM and HSGL output l feature vectors that correspond to the n-layer hierarchical structure. Figure 4.1 in chapter 4 visually illustrates their differences.

There are three discrete states of emotion in the EEG dataset - neutral, anger and happiness. Figure 5.1 shows where the three states are located in a two-dimensional valence-arousal model. Please note that emotion can be categorized as discrete states [23], continuous states [24, 25] or a mix of both (i.e. placing discrete emotion states on

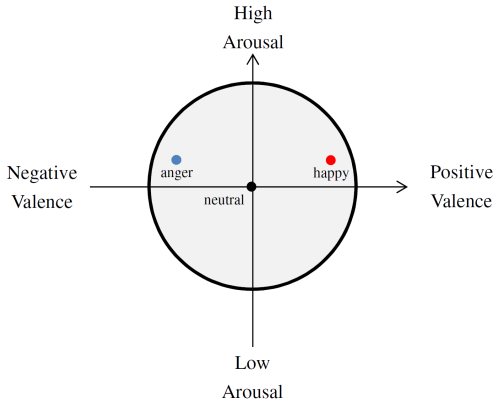


Figure 5.1: An illustration of the hybrid discrete-dimensional model of affect. This work focuses on the states of neutral, happiness and anger. They are considered far away along the valence dimension of the two-dimensional model of affect.

the multi-dimensional affective space [26, 27, 28]), and there is no unilateral agreement on the definition of emotion [29]. Only discrete states of neutral, happiness and anger are focused because 1) they are easy to define and induce in experiments and 2) they are considered far away in the hybrid discrete-dimensional model of affect.

Last but not least, contribution of this work includes i) the evaluation of discriminating features on emotion recognition with SGL, HL and HSGL, and ii) a stringent approach of cross validation - a double 5-fold cross validation was run for only one time in most of the surveyed literature. As assigning observations to different folds can introduce randomness, and it is possible that the accuracy obtained in one cross validation would be different from another. In this work, 30 times of a double 5-fold cross validation was run in order to get the mean and standard deviation of the accuracy and density of the feature vector for evaluation purposes.

5.2 Methods

5.2.1 Data Collection

Participants ($n = 41$) were recruited using university-approved fliers and given compensation for participation in the research project. Two participants were excluded from the experiment because of too thick of hair, which is a limitation of the electroencephalogram (EEG) device. All participants were 18 years or older and signed an informed consent form prior to participating in the experiment. Each participant completed a demographic form, a mental health form, a happiness assessment form, and an anger assessment form. After completing the forms, participants were sized in regard to the appropriate EEG cap size. An EEG cap was placed on the participant's head and electro-gel was applied to the hair to improve signal conductance from the scalp to the EEG electrodes.

Baseline measures of heart rate, blood pressure, and oximetry were recorded. These measures were also monitored throughout the experiment to detect physiological changes in response to different emotional stimuli. The initial twenty participants ($n = 20$) watched 3 brief video clips which are 2 minutes long. The movie length was chosen under the recommendation in [30, 31] that videos from 1 to 10 minutes can effectively elicit a single emotion. After that, they completed a brief self-report form of their emotional reactions following each clip. Blood pressure was also measured immediately following the conclusion of each video. There was a seven minute rest following the administration of each video clip. During this period, simple math problems were presented to the participant on a computer screen. One video clip depicted a screen saver which was thought to induce a neutral emotion, one video clip depicted a high school student being physically bullied at school which was thought to induce anger, and one video clip depicted nature scenery or baby which was thought

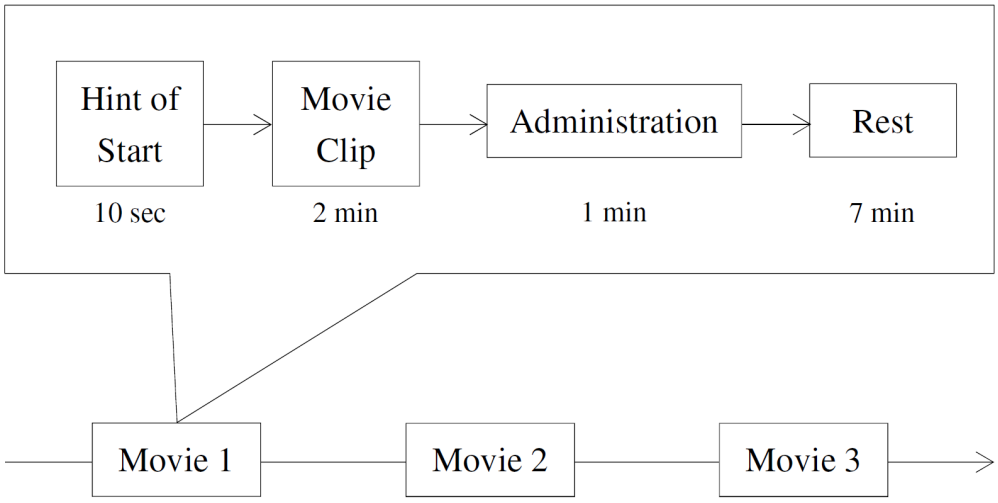


Figure 5.2: The procedure of the experiment for each showing of movie clips consists of hint of start, showing of the movie clip, administration such as filling in questionnaire and measuring blood pressure and rest. Sequence of movies (neutral, anger and happy) are randomized for each experiment subject. It takes around 30 minutes for the entire procedure.

to induce happiness. The video presentation order of the three video clips were randomized for each participant. After watching the series of videos, each participant was de-capped and given shampoo and towel to wash the gel from their hair.

5.2.2 Data Preprocessing

EEG data of 32 electrodes (figure 5.3) are imported into Matlab with software package "EEGlab" [32]. EEG signals will then be re-referenced at channels M1 and M2 since these two channels are least influenced by cognitive processing, resampled from 1000 Hz to 256 Hz for reducing data size, and bandpass filtered at 1-50 Hz for removing unnecessary signal noise.

Artifact is then removed from EEG signal with EEGlab plugin - ADJUST (An Automatic EEG artifact Detector based on the Joint Use of Spatial and Temporal features) [33]. Artifact features including eye blinks, (vertical and horizontal) eye

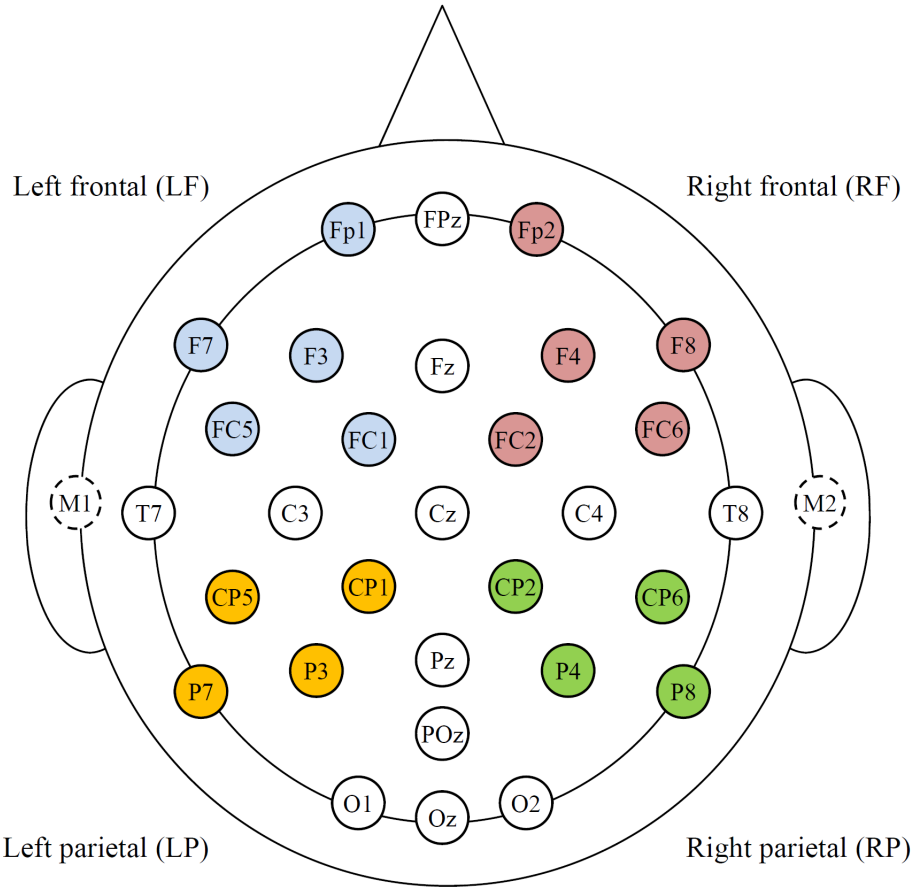


Figure 5.3: Position of the 32 EEG electrodes. Electrodes of four areas - left frontal (LF), right frontal (RF), left parietal (LP) and right parietal (RP) - are focused and highlighted in color. M1 and M2 are not available for analysis after re-referencing.

movements and generic discontinuities are accounted for. The four stages are that i) epoched EEG signal is first decomposed into different independent components using independent component analysis (ICA); ii) artifact features for each component are computed; iii) the value of each artifact feature for each component is to be checked against threshold value computed by Expectation-Maximization [34] in order to determine whether that component is an artifact; and iv) EEG signal is reconstructed using independent components which are not rejected.

Last but not least, the entire EEG signal is first partitioned into three epochs corresponding to the time the movie was shown to the participant. EEG signal of each movie will be further extracted from each 10-second window including 0-10 seconds, 10-20 seconds until 110-120 seconds. There are 24 epochs for each emotion state.

5.2.3 Feature Extraction

Seven groups of mathematical features are extracted from each trial of subjects as in [35, 36, 37]. These features are extracted for each of the 30 channels[1]. They are then concatenated as a feature vector. This procedure is applied to data of all three kinds of movies all time epochs and all participants. In the following, $X = \{x_1, x_2, ..., x_m\}$ denote a single-channel signal with m time points. The seven groups are:

1) Statistical features: Mean, variance, skewness, kurtosis are computed. More specifically, mean is the averaged signal amplitude, variance measures the signal variability to the mean, skewness quantifies the extent to which the distribution leans to one side of the mean, and kurtosis measures the 'peakedness' of the distribution.

2) Curve length [38, 36]: It is the sum of distances between any two pair of consecutive points. Intuition behind this feature is that curve length increase with the signal magnitude, frequency and amplitude variation. It is mathematically calculated as follows:

$$\frac{1}{m-1}\sum_{i=1}^{m-1}|x_{i+1} - x_i| \tag{5.7}$$

3) Number of Peaks [36]: It measures the overall frequency of a signal. It is mathematically calculated as follows:

$$\frac{1}{2}\sum_{i=1}^{m-2}\max[0, \text{sign}(x_{i+2} - x_{i+1}) - \text{sign}(x_{i+1} - x_i)] \tag{5.8}$$

[1]M1 and M2 are not available after re-referencing.

4) Average nonlinear energy [39]: It is sensitive to spectral changes and is thus useful for capture spectral information of a signal [40]. It is calculated as:

$$\frac{1}{m-2}\sum_{i=2}^{m-1} x_i^2 - x_{i-1}x_{i+1} \tag{5.9}$$

5) Number of zero crossing: It is the number of points where the sign of the EEG signal changes from positive to negative (or vice versa).

6) Spectral edge frequency: It measures the frequency below which a certain percentage of total power of the EEG time-series signal [41]. In this project, percentage values of 50%, 90% and 95% are considered.

7) Relative band power: Band power of EEG signals [42] in commonly used frequency bands of brain signal including theta (2.5 - 7 Hz), alpha (7.5 - 12 Hz), beta (12.5 -29 Hz), and gamma bands (30 - 50 Hz) is computed. Relative band power is computed as the ratio of the band power of the individual band over the sum of band power of all four bands.

Table 5.1 summarizes the features extracted in this work. For each channel and each time epoch, there are 15 features in total. Therefore, the total number of features is 10,800 (15 features, 30 channels, 24 time epochs).

Figure 5.4 shows the hierarchy of the features. The possible group assignment of features for HSGL can be time epoch, channel and feature group. Besides, the 30 channels can be divided into five groups as in figure 5.3.

Table 5.1: Summary of 7 groups of time-series features in chapter 5.

No	Group Name	Explanation	Count
1	Statistical	mean, variance, skewness, kurtosis	4
2	Curve length	-	1
3	No. of peaks	-	1
4	Average nonlinear energy	-	1
5	No. of zero crossing	-	1
6	Spectral edge frequency	50%, 95%, 99%	3
7	Relative band power	theta/all, alpha/all, beta/all, gamma/all	4

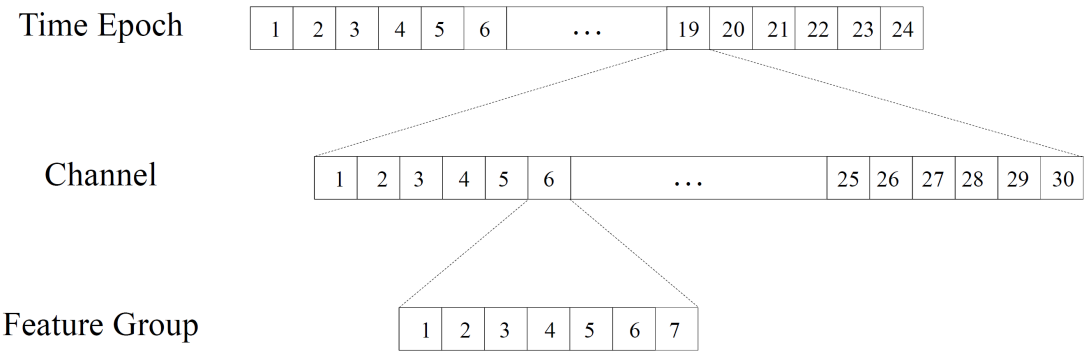


Figure 5.4: An visual illustration of the hierarchy of features in this EEG studies. The possible group assignment of features for HSGL can be time epoch, channel and feature group. Besides, the 30 channels can be divided into five groups as in figure 5.3.

5.2.4 Hierarchical Sparse Group Lasso

Hierarchical sparse group lasso[2] is based on lasso, SGL and HL. The formulation of a two-layer hierarchical sparse group lasso (HSGL) is as follows:

$$\begin{aligned} &\min_{\boldsymbol{d}\geq 0,\boldsymbol{x}} \; f(\boldsymbol{d},\boldsymbol{x}) \\ =&\min_{\boldsymbol{d}\geq 0,\boldsymbol{x}} \; \sum_{n=1}^{N} ||y_n - \sum_{k=1}^{K} d_k \sum_{g=1}^{G_k} \boldsymbol{A}_{n,kg}^T \boldsymbol{x}_{kg}||^2 + \lambda_1 \sum_{k=1}^{K} d_k + \lambda_2 ||\boldsymbol{x}||_1 + \lambda_3 \sum_{k=1}^{K} \sum_{g=1}^{G_k} ||\boldsymbol{x}_{kg}||_2 \end{aligned} \tag{5.10}$$

[2]In the model, a scalar is represented by a non-bold lower-case letter, a vector is represented by a bold lower-case letter and a matrix is represented by a bold upper-case letter. Symbols in the formulations are changed for better consistency.

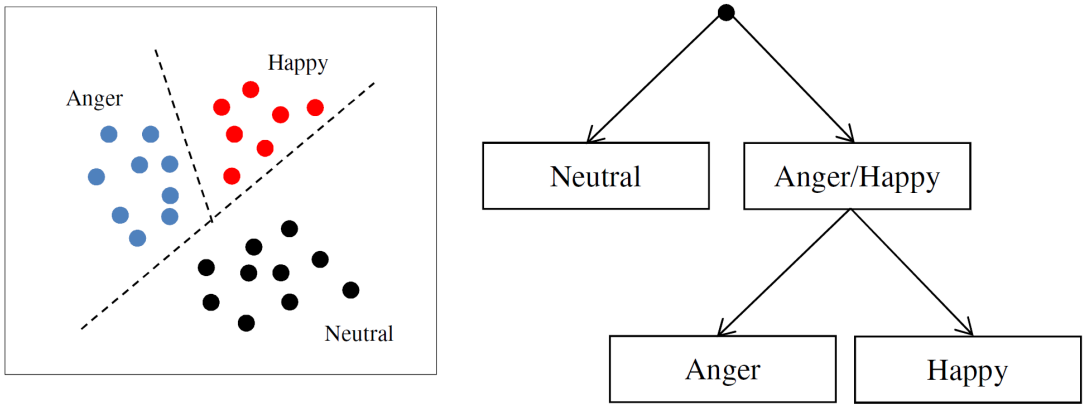


Figure 5.5: Adaptive training of multi-class HSGL classification with hierarchical splitting [45].

5.2.4.1 Optimization

HSGL is a non-linear biconvex optimization problem [43]. Recall that a function $f : X \times Y \leftarrow \mathbb{R}$ is called biconvex, if $f(x, y)$ is convex in y for fixed $x \in X$, and $f(x, y)$ is convex in x for fixed $y \in Y$. It is proposed to solve HSGL with alternative convex search [43, 13]. In essence, the two variables to be solved are optimized in turn until convergence criteria is met. For detail, please refer to section 4.3 in chapter 4.

5.2.4.2 Extension to Multi-Class Classification

Developed as a regression method, HSGL can be used as classification for labels with two classes. To cope with the multi-class classification, two classifications can be conducted [44, 45]: firstly, the classification of i) neutral vs. ii) happy and anger can be performed, i.e. happy and anger states are considered as the same class; after that, predicted labels that are happy and anger will be chosen for another classification between i) happy and ii) anger states. An visual illustration can be found in figure 5.5.

5.2.5 Personalized Feature Standardization

An observation obtained from preliminary experiments which is not reported in this paper is that classification accuracy tends to decrease with the number of experiment subjects available in the data. It indicates that there is high inter-person variability for the features extracted. To tackle this issue, personalization of feature measures in [36, 46] is adopted as follows:

$$\begin{aligned} \text{whisker}_{lower} &= \max(\text{minimum value}, \text{LQ} - 1.5 * \text{IQR}) \\ \text{whisker}_{upper} &= \min(\text{maximum value}, \text{UQ} + 1.5 * \text{IQR}) \\ \text{feat}_{new} &= \frac{\text{feat} - \text{whisker}_{lower}}{\text{whisker}_{upper} - \text{whisker}_{lower}} \end{aligned} \tag{5.11}$$

where LQ means lower quartile, UQ means upper quartile and IQR means interquartile range. Please note that such personalization is applied for each feature individually. For example, if your feature matrix is arranged as observation by feature, the personalization in equations (5.11) are applied column-wise for each subject.

The basic idea of this personalization is that the upper and lower whiskers define a personalized interval which contains most of the feature values. Outliers that do not fall within the whisker-interval are mapped to either 0 or 1. The implementation is easy but the enhancement on learning is surprisingly tremendous.

5.2.6 Classification

30 times of a double five-fold cross validation (CV) were run for predicting the out-of-sample error. For each run of CV, data is divided to into five random subsets. One of the subsets is used as testing, and the remaining four subsets are used for training, and such procedure is repeated for five times for each CV. Previous literature often runs CV for one time only. However, this work ran the double 5-fold CV for 30

times in order to evaluate the impact of randomly splitting data into different folds on classification accuracy and feature density.

Parameters, if any, will be tuned using the training dataset using another five-fold cross validation to provide a realistic estimation of prediction errors and to prevent over-fitting, explaining the term "double 5-fold CV". After running all possible combinations of parameters, those with the highest accuracy will be taken out. Then, among all the taken combinations, the one with lowest density of feature vector will be chosen to train the model. It is observed that there can be multiple combinations of parameters which can achieve the best accuracy. It follows that using classification accuracy alone may not be robust enough for parameter tuning.

5.3 Evaluations

Experiments were executed on a workstation with an Intel i7-5960X CPU and 64 GB RAM with Ubuntu 17.04 operating system.

5.3.1 Performance Measures

There are two performance measures in this regard - i) classification accuracy which is the ratio between the number of correctly classified number of emotion states and total number of emotion states, and ii) density of feature vector $\boldsymbol{x}$ (for SGL and SVM) or $\boldsymbol{\beta} = \boldsymbol{d} \odot \boldsymbol{x}$ (for HSGL) which is the ratio between the number of non-zero elements of the feature vector and that of the length. Model is deemed to be good if accuracy is high. As for density of feature vector, as there is no ground truth for the selection result, a lower density of feature vector does not mean the selection result is more superior. However, it is preferable to have a low density of feature density such as 2% vs. 20%.

5.3.2 Comparison with Baseline Methods

To illustrate the advantages of the proposed methods, four other baseline prediction models are compared. All implementations are Matlab-based. Notations in the following models have been changed for better consistency with the above.

- Support Vector Machine (SVM): It is simply the L_2-regularized, L_2-loss support vector machine implemented by Liblinear.

$$\min_{\boldsymbol{x}} \ \frac{1}{2}||\boldsymbol{x}||_2 + C\sum_{n=1}^{N}||\max(0, 1 - y_i \boldsymbol{a}_i^T \boldsymbol{x})||_2 \tag{5.12}$$

- Decision Tree (DT): Random forests are an ensemble learning method for classification, regression and other tasks, that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Random decision forests correct for decision trees' habit of over-fitting to their training set.
- Adaptive Boosting (Adaboost): AdaBoost, short for "Adaptive Boosting", is a machine learning meta-algorithm which is used in conjunction with many other types of learning algorithms to improve their performance. The output of the other learning algorithms ('weak learners') is combined into a weighted sum that represents the final output of the boosted classifier. AdaBoost is adaptive in the sense that subsequent weak learners are tweaked in favor of those instances misclassified by previous classifiers.
- Sparse Group Lasso (SGL): It has been introduced in earlier section. The formulation can be found in (5.3). SGL is implemented with the SLEP Toolbox [15].

5.3.3 Discussion of Computational Result

5.3.3.1 Overall

Table 5.2 lists the mean and standard deviation of accuracy and density of feature vector of running 5-fold cross-validated classification for 30 times, if any, for HSGL, SGL, adaboost, decision tree (DT) and L_2-regularized L_2-loss support vector machine (SVM) using all 10,800 available features. In other words, each number in the table is the mean and standard deviation of the 150 models obtained from each of the 5-fold cross validated classifications across the 30 cases.

Classification in the binary cases of "neutral vs. anger" and "neutral vs. happy" using HSGL (around 95%) performs competitively with SGL (around 93%) and other baseline classifiers (80%-96%). However, for the binary case "anger vs. happy" and the tertiary case ("neutral vs. anger vs. happy"), SGL performs better than HSGL in terms of accuracy. In addition, the choice of feature hierarchy affects significantly on learning performance of HSGL (66%-78% for "anger vs. happy").

Especially for the tertiary case, since SGL (82%) and HSGL (66%-78%) are not native multi-class classifiers, they do not perform well as compared with SVM (85%). There are certainly better multi-class classifiers for real-life emotion recognizer. SGL and HSGL are, nevertheless, good analysis tools on the identifying the discriminating features.

The density of feature vector of SGL is generally lower than that of HSGL across different cases.

Table 5.2: 30 times of double 5-fold cross-validated classification accuracy and density of feature vector, if any, for HSGL, SGL, adaboost, decision tree (DT) and L_2-regularized L_2-loss support vector machine (SVM) using all available 10,800 features. Group assignments are channel, time epoch and feature group and are only available for SGL. All of the numbers outside of parentheses are means over 30 repetitions, and the numbers in the parentheses are the corresponding standard errors.

Neutral vs. Anger

	HSGL			SGL			SVM	Adaboost	DT
	Epoch Time - Channel	Feat Gp - Epoch Time	Epoch Time - Feat Gp	Channel	Epoch Time	Feat Gp			
Accuracy	95.65% (1.56%)	95.89% (1.41%)	94.59% (1.74%)	92.93% (1.65%)	93.5% (0.98%)	93.01% (1.24%)	96.15% (1.78%)	93.63% (1.74%)	80.31% (3.6%)
Density	3.12% (0.6%)	19.7% (3.63%)	3.26% (0.46%)	2.23% (0.85%)	2.19% (0.56%)	5.3% (1.36%)	100% (0%)	N/A	N/A

Neutral vs. Happy

	HSGL			SGL			SVM	Adaboost	DT
	Epoch Time - Channel	Feat Gp - Epoch Time	Epoch Time - Feat Gp	Channel	Epoch Time	Feat Gp			
Accuracy	94.39% (1.71%)	93.21% (1.59%)	96.18% (0.89%)	85.37% (1.69%)	95.61% (1.14%)	92.03% (1.14%)	90.45% (2.28%)	93.2% (1.47%)	88.43% (4.48%)
Density	2.82% (0.52%)	22.2% (3.58%)	2.96% (0.63%)	0.62% (0.35%)	0.87% (0.18%)	3.35% (1.09%)	100% (0%)	N/A	N/A

Anger vs. Happy

	HSGL			SGL			SVM	Adaboost	DT
	Epoch Time - Channel	Feat Gp - Epoch Time	Epoch Time - Feat Gp	Channel	Epoch Time	Feat Gp			
Accuracy	66.06% (9.06%)	78.05% (2.24%)	66.54% (4.46%)	84.92% (1.71%)	86.99% (1.9%)	84.39% (2.94%)	86.54% (4.77%)	80.07% (3.84%)	59.05% (4.74%)
Density	2.51% (0.69%)	37.88% (6.35%)	2.95% (0.67%)	3.81% (0.64%)	5.35% (1.07%)	4.97% (2.12%)	100% (0%)	N/A	N/A

Neutral vs. Anger vs. Happy

	HSGL			SGL			SVM	Adaboost	DT
	Epoch Time - Channel	Feat Gp - Epoch Time	Epoch Time - Feat Gp	Channel	Epoch Time	Feat Gp			
Accuracy	74.47% (1.78%)	78.35% (2.86%)	78.78% (2.95%)	82.71% (1.54%)	82.49% (1.62%)	82.41% (1.74%)	85.71% (1.32%)	80.27% (3.14%)	53.81% (3.1%)

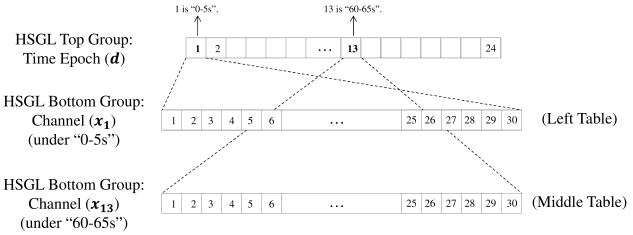
5.3.3.2 Hierarchical Structure: Time Epoch - Channel

Table 5.3 lists the average group density of feature vector with HSGL using group assignment of "Time Epoch - Channel" versus those of SGL using group assignment of "Channel" from all 10,800 available features. Each number in the table is the average density of feature groups at a particular time epoch from each of the 5 folds across the 30 classification runs.

For the selection result of the top hierarchy, only 1 time epoch out of 24 was selected with HSGL - features of "0-5s" were required for classification of "neutral vs. anger" and "neutral vs. happy", and those of "60-65s" were required for "anger vs. happy". For the selection result of SGL (the right table of Table 5.4), features of 2, 1, and 9 time epochs were required for "neutral vs. anger", "neutral vs. happy" and "anger vs. happy". HSGL clearly has the ability of using features from fewer feature groups while maintaining comparable classification accuracy.

For the selection result of the bottom hierarchy, HSGL selects more features across different channels as opposed to SGL. SGL and HSGL do not show a consistent choice of important channels for the bottom hierarchy.

Table 5.3: Result of feature selection of HSGL using group assignment of "Time Epoch - Channel" versus SGL using group assignment of "Time Epoch". Each number is the average density of feature groups at a particular time epoch from each of 5 folds across the 30 classification runs. For the top hierarchy of HSGL, the feature group of "0-5s" were selected for "neutral vs. anger" and "neutral vs. happy" (left table), and the feature group of "60-65s" was selected for "anger vs. happy" (middle table). The right table shows the result of feature selection of SGL using group assignment of channel. Last but not least, LF means left frontal, LP means parietal, RF means right frontal and RP means right parietal.



		HSGL - "0-5s"		
		Neutral vs. Anger	Neutral vs. Happy	Anger vs. Happy
1. LF	F3	26.67%	46.67%	0
1. LF	F7	53.33%	93.33%	0
1. LF	FC1	40.00%	53.33%	0
1. LF	FC5	66.67%	80.00%	0
1. LF	Fp1	13.33%	6.67%	0
2. LP	CP1	53.33%	60.00%	0
2. LP	CP5	33.33%	40.00%	0
2. LP	P3	60.00%	46.67%	0
2. LP	P7	66.67%	53.33%	0
3. RF	F4	13.33%	60.00%	0
3. RF	F8	40.00%	60.00%	0
3. RF	FC2	26.67%	40.00%	0
3. RF	FC6	33.33%	66.67%	0
3. RF	Fp2	40.00%	33.33%	0
4. RP	CP2	53.33%	60.00%	0
4. RP	CP6	40.00%	66.67%	0
4. RP	P4	20.00%	33.33%	0
4. RP	P8	26.67%	40.00%	0
5. Others	C3	13.33%	46.67%	0
5. Others	C4	40.00%	53.33%	0
5. Others	Cz	53.33%	13.33%	0
5. Others	Fpz	33.33%	53.33%	0
5. Others	Fz	26.67%	40.00%	0
5. Others	O1	20.00%	13.33%	0
5. Others	O2	20.00%	26.67%	0
5. Others	Oz	13.33%	26.67%	0
5. Others	POz	13.33%	20.00%	0
5. Others	Pz	46.67%	26.67%	0
5. Others	T7	40.00%	53.33%	0
5. Others	T8	60.00%	86.67%	0

		HSGL - "60-65s"		
		Neutral vs. Anger	Neutral vs. Happy	Anger vs. Happy
1. LF	F3	0	0	66.67%
1. LF	F7	0	0	66.67%
1. LF	FC1	0	0	60.00%
1. LF	FC5	0	0	66.67%
1. LF	Fp1	0	0	80.00%
2. LP	CP1	0	0	40.00%
2. LP	CP5	0	0	40.00%
2. LP	P3	0	0	33.33%
2. LP	P7	0	0	80.00%
3. RF	F4	0	0	46.67%
3. RF	F8	0	0	93.33%
3. RF	FC2	0	0	60.00%
3. RF	FC6	0	0	33.33%
3. RF	Fp2	0	0	80.00%
4. RP	CP2	0	0	46.67%
4. RP	CP6	0	0	33.33%
4. RP	P4	0	0	46.67%
4. RP	P8	0	0	13.33%
5. Others	C3	0	0	40.00%
5. Others	C4	0	0	13.33%
5. Others	Cz	0	0	53.33%
5. Others	Fpz	0	0	53.33%
5. Others	Fz	0	0	20.00%
5. Others	O1	0	0	86.67%
5. Others	O2	0	0	80.00%
5. Others	Oz	0	0	66.67%
5. Others	POz	0	0	60.00%
5. Others	Pz	0	0	6.67%
5. Others	T7	0	0	86.67%
5. Others	T8	0	0	53.33%

		SGL		
		Neutral vs. Anger	Neutral vs. Happy	Anger vs. Happy
1. LF	F3	0	0	0
1. LF	F7	0	0	44.17%
1. LF	FC1	0	0	0
1. LF	FC5	0	0	0
1. LF	Fp1	36.39%	0	60.28%
2. LP	CP1	0	0	0
2. LP	CP5	0	35.83%	0
2. LP	P3	5.28%	10.56%	0.28%
2. LP	P7	28.33%	0	0
3. RF	F4	0	0	0
3. RF	F8	21.94%	0	65.83%
3. RF	FC2	0	0	0
3. RF	FC6	0	0	0
3. RF	Fp2	0	0	0
4. RP	CP2	0	0	0
4. RP	CP6	0	0	0
4. RP	P4	0	0	0
4. RP	P8	24.17%	31.11%	0
5. Others	C3	37.78%	0	0
5. Others	C4	0	0	0
5. Others	Cz	0	0	0
5. Others	Fpz	18.33%	0	43.33%
5. Others	Fz	0	0	0
5. Others	O1	30.28%	31.11%	0
5. Others	O2	26.11%	34.72%	38.06%
5. Others	Oz	27.50%	0	37.50%
5. Others	POz	0	0	0.56%
5. Others	Pz	0	0	0
5. Others	T7	0	0	0
5. Others	T8	14.44%	9.44%	0

5.3.3.3 Hierarchical Structure: Feature Group - Time Epoch

Table 5.4 lists the average group density of feature vector with HSGL using group assignment of "Feat Gp - Time Epoch" versus those of SGL using group assignment of "Time Epoch" from all 10,800 available features. Each number in the table is the average density of feature groups at a particular time epoch from each of the 5 folds across the 30 classification runs.

The left table shows the result of feature selection of bottom hierarchy - time epoch - under statistical features. The right table shows the result of feature selection of SGL using group assignment of time epoch.

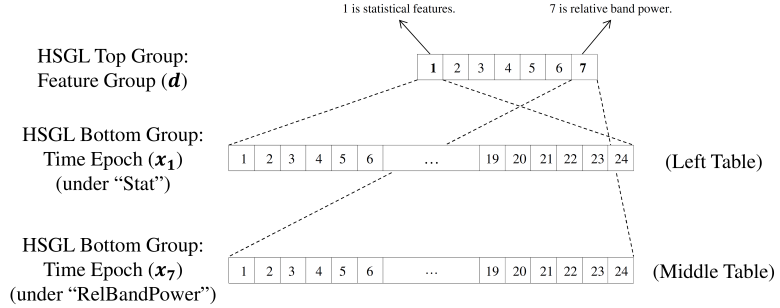
For the selection result of the top hierarchy of HSGL, two groups of features - statistical features and relative band power - were selected. For SGL (the right table of table 5.5), features were selected from all 7 feature groups for "anger vs. happy". Again, HSGL is able to select features from fewer feature groups for "anger vs. happy". In addition, feature groups including curve length (CL), number of peaks (Peaks), average non-linear energy (ANE), number of zero crossings (ZC) and spectral edge frequency are not selected by HSGL at all, meaning that they are not important in discriminating emotion.

For the bottom hierarchy of HSGL, under the top feature group ($\boldsymbol{d}$) of statistical features for HSGL, almost features from all time epochs for three binary classification cases were selected, as illustrated in the left of Table 5.4. It further explains the importance of statistical features in emotion discrimination. For "neutral vs. anger" and "neutral vs. happy", features of earlier time epochs (0-20s) were more chosen, whereas for "anger vs. happy", features of later times epochs (60-120s) were more chosen. Such observation is important in designing an real-life emotion recognizer - for example, a device just needs EEG signals of the first 20 seconds to detect the

deviation of the subject's emotion from the neutral state, whereas a device needs focus on EEG signals of 60 seconds or later in order to differentiate happy from anger.

Most importantly, comparing the result of HSGL with that of SGL, it seems that SGL requires fewer features than HSGL in emotion recognition (left and middle tables as opposed to the right table in table 5.4). However, the total number of statistical features is 4 (mean, std, skewnewss, kurtosis) $\times 24$ time epochs $\times 30$ channels $= 2880$, whereas the total number of features used by SGL is 10800. From HSGL, classification using statistical features alone would be sufficient, at least in the cases of "neutral vs. happy (around 95% accuracy)" and "neutral vs. anger (around 94% accuracy)", but it would require more features across the time epochs, as opposed to the situation of SGL that it requires only features at a few time epochs.

Table 5.4: Result of feature selection of HSGL using group assignment of "Feat Gp - Time Epoch" versus SGL using group assignment of "Time Epoch". Each number is the average density of feature groups at a particular time epoch from each of 5 folds across the 30 classification runs. For the top hierarchy of HSGL, the feature group of statistical features (Stat) was selected. The left table shows the result of feature selection of bottom hierarchy - time epoch - under statistical features. The right table shows the result of feature selection of SGL using group assignment of time epoch.



	HSGL - "Stat"		
	Neutral vs. Anger	Neutral vs. Happy	Anger vs. Happy
0s - 5s	88.33%	98.33%	85.00%
5s - 10s	94.17%	95.83%	92.50%
10s - 15s	89.17%	96.67%	95.83%
15s - 20s	89.17%	98.33%	93.33%
20s - 25s	85.83%	95.00%	95.00%
25s - 30s	84.17%	93.33%	90.00%
30s - 35s	87.50%	94.17%	93.33%
35s - 40s	79.17%	94.17%	95.00%
40s - 45s	83.33%	87.50%	90.00%
45s - 50s	79.17%	95.83%	92.50%
50s - 55s	71.67%	87.50%	92.50%
55s - 60s	80.00%	93.33%	91.67%
60s - 65s	86.67%	89.17%	95.00%
65s - 70s	79.17%	93.33%	90.00%
70s - 75s	85.00%	96.67%	94.17%
75s - 80s	84.17%	87.50%	95.00%
80s - 85s	85.00%	91.67%	85.83%
85s - 90s	65.00%	93.33%	87.50%
90s - 95s	75.83%	91.67%	96.67%
95s - 100s	81.67%	95.83%	92.50%
100s - 105s	77.50%	90.83%	94.17%
105s - 110s	80.00%	93.33%	89.17%
110s - 115s	85.83%	88.33%	95.83%
115s - 120s	80.83%	95.83%	95.00%

	HSGL - "RelBandPower"		
	Neutral vs. Anger	Neutral vs. Happy	Anger vs. Happy
0s - 5s	0	0	85.00%
5s - 10s	0	0	78.33%
10s - 15s	0	0	91.67%
15s - 20s	0	0	81.67%
20s - 25s	0	0	85.83%
25s - 30s	0	0	79.17%
30s - 35s	0	0	83.33%
35s - 40s	0	0	84.17%
40s - 45s	0	0	77.50%
45s - 50s	0	0	75.00%
50s - 55s	0	0	76.67%
55s - 60s	0	0	77.50%
60s - 65s	0	0	90.83%
65s - 70s	0	0	92.50%
70s - 75s	0	0	74.17%
75s - 80s	0	0	88.33%
80s - 85s	0	0	86.67%
85s - 90s	0	0	84.17%
90s - 95s	0	0	87.50%
95s - 100s	0	0	80.83%
100s - 105s	0	0	80.00%
105s - 110s	0	0	82.50%
110s - 115s	0	0	82.50%
115s - 120s	0	0	70.83%

	SGL		
	Neutral vs. Anger	Neutral vs. Happy	Anger vs. Happy
0s - 5s	28.00%	58.67%	0
5s - 10s	13.33%	0	12.00%
10s - 15s	0	0	0
15s - 20s	0	0	0
20s - 25s	0	0	25.11%
25s - 30s	0	0	0
30s - 35s	0	0	0
35s - 40s	0	0	22.44%
40s - 45s	0	0	0
45s - 50s	0	0	0
50s - 55s	0	0	0
55s - 60s	0	0	0
60s - 65s	0	0	33.33%
65s - 70s	0	0	0
70s - 75s	0	0	0
75s - 80s	0	0	26.44%
80s - 85s	0	0	0
85s - 90s	0	0	0
90s - 95s	0	0	29.78%
95s - 100s	0	0	20.89%
100s - 105s	0	0	36.00%
105s - 110s	0	0	0
110s - 115s	0	0	32.89%
115s - 120s	0	0	0

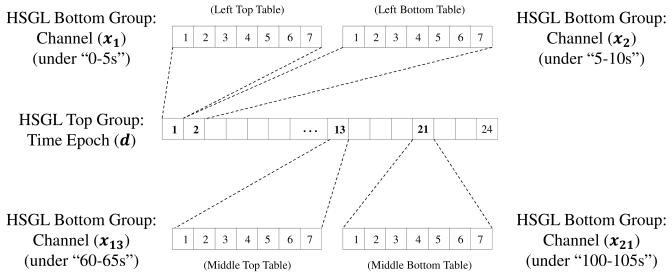
5.3.3.4 Hierarchical Structure: Epoch Time - Feature Group

Table 5.5 lists the average group density of feature vector with HSGL using group assignment of "Time Epoch - Feature Group" versus those of SGL using group assignment of "Time Epoch" from all 10,800 available features. Each number in the table is the average density of feature groups at a particular time epoch from each of the 5 folds across the 30 classification runs.

The left two tables show the result of feature selection of bottom hierarchy - feature group - under "0-5s" and "5-10s". The middle two tables show the result of feature selection of bottom hierarchy - feature group - under "60-65s" and "100-105s". The right table shows the result of feature selection of SGL using group assignment of time epoch.

The selection result of the top hierarchy for HSGL in this case is similar to that using "Epoch Time - Channel". As for the bottom hierarchy, features from all time epochs were selected - such an observation is similar to the above in that HSGL requires a certain portion of features in order to conduct classification, and that is why it selects most of the features under the bottom hierarchy.

Table 5.5: Result of feature selection of HSGL using group assignment of "Time Epoch - Feature Group" versus SGL using group assignment of "Time Epoch". Each number is the average density of feature groups at a particular time epoch from each of 5 folds across the 30 classification runs. For the top hierarchy of HSGL, the feature groups of "0-5s", "5-10s", "60-65s" and "100-105s" were selected. The left two tables show the result of feature selection of bottom hierarchy - feature group - under "0-5s" and "5-10s". The middle two tables show the result of feature selection of bottom hierarchy - feature group - under "60-65s" and "100-105s". The right table shows the result of feature selection of SGL using group assignment of time epoch.



	HSGL - "0-5s"		
Feat Gp	Neutral vs. Anger	Neutral vs. Happy	Anger vs. Happy
Stats	97.50%	98.33%	0
CL	96.67%	96.67%	0
Peaks	100.00%	96.67%	0
ANE	100.00%	96.67%	0
ZC	93.33%	100.00%	0
SEF	93.33%	96.67%	0
RelBandPower	95.83%	96.67%	0

	HSGL - "60-65s"		
Feat Gp	Neutral vs. Anger	Neutral vs. Happy	Anger vs. Happy
Stats	0	0	100.00%
CL	0	0	96.67%
Peaks	0	0	100.00%
ANE	0	0	100.00%
ZC	0	0	100.00%
SEF	0	0	98.89%
RelBandPower	0	0	98.33%

	SGL		
Feat Gp	Neutral vs. Anger	Neutral vs. Happy	Anger vs. Happy
Stats	57.05%	52.26%	15.94%
CL	0	0	6.25%
Peaks	0	0	10.00%
ANE	0	0	6.53%
ZC	0	0	11.25%
SEF	0	0	14.31%
RelBandPower	0	0	15.59%

	HSGL - "5-10s"		
Feat Gp	Neutral vs. Anger	Neutral vs. Happy	Anger vs. Happy
Stats	55.00%	0	0
CL	6.67%	0	0
Peaks	23.33%	0	0
ANE	6.67%	0	0
ZC	93.33%	0	0
SEF	8.89%	0	0
RelBandPower	19.17%	0	0

	HSGL - "100-105s"		
Feat Gp	Neutral vs. Anger	Neutral vs. Happy	Anger vs. Happy
Stats	0	0	83.33%
CL	0	0	80.00%
Peaks	0	0	83.33%
ANE	0	0	86.67%
ZC	0	0	93.33%
SEF	0	0	82.22%
RelBandPower	0	0	78.33%

5.4 Concluding Remarks

This paper attempts to address the interesting problem in EEG studies that features often follow a hierarchical structure. On the whole, SVM performs generally well in terms of classification accuracy and SGL yields the lowest density of feature vector. HSGL offers an refreshing view of hierarchical feature selection, and such result is consistent with that using SGL while having a lower group density of feature vectors. As presented in the computational result, HSGL is able to select a few feature groups at the top hierarchy, and these identified feature groups can be analyzed in details.

In order to learn better, there are still more directions for further exploration. Obviously the use of an optimization algorithm to which can attain global optimum in order to solve HSGL is essential. Other EEG and time-series features can be used in the study as well. Physiological signals such as heart rate and blood pressure should be incorporated in the analysis. This quantitative analysis framework could be the basis of a real-life applicable tool to assist social workers to help monitor emotion states and to do anger management of an individual.

Bibliography

[1] S Jerritta, M Murugappan, R Nagarajan, and Khairunizam Wan. Physiological signals based human emotion recognition: a review. In *Signal Processing and its Applications (CSPA), 2011 IEEE 7th International Colloquium on*, pages 410–415. IEEE, 2011.

[2] Youngmoo E Kim, Erik M Schmidt, Raymond Migneco, Brandon G Morton, Patrick Richardson, Jeffrey Scott, Jacquelin A Speck, and Douglas Turnbull. Music emotion recognition: A state of the art review. In *Proc. ISMIR*, pages

255–266, 2010.

[3] Shashidhar G Koolagudi and K Sreenivasa Rao. Emotion recognition from speech: a review. *International journal of speech technology*, 15(2):99–117, 2012.

[4] Yi-Hsuan Yang and Homer H Chen. Machine recognition of music emotion: A review. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 3(3):40, 2012.

[5] Jonghwa Kim and Elisabeth André. Emotion recognition based on physiological changes in music listening. *IEEE transactions on pattern analysis and machine intelligence*, 30(12):2067–2083, 2008.

[6] Philipp Michel and Rana El Kaliouby. Real time facial expression recognition in video using support vector machines. In *Proceedings of the 5th international conference on Multimodal interfaces*, pages 258–264. ACM, 2003.

[7] Yixiong Pan, Peipei Shen, and Liping Shen. Speech emotion recognition using support vector machine. *International Journal of Smart Home*, 6(2):101–108, 2012.

[8] Björn Schuller, Gerhard Rigoll, and Manfred Lang. Speech emotion recognition combining acoustic features and linguistic information in a hybrid support vector machine-belief network architecture. In *Acoustics, Speech, and Signal Processing, 2004. Proceedings.(ICASSP'04). IEEE International Conference on*, volume 1, pages I–577. IEEE, 2004.

[9] Haiyan Xu and Konstantinos N Plataniotis. Affect recognition using eeg signal. In *Multimedia Signal Processing (MMSP), 2012 IEEE 14th International Workshop on*, pages 299–304. IEEE, 2012.

[10] Björn Schuller, Gerhard Rigoll, and Manfred Lang. Hidden markov model-based speech emotion recognition. In *Multimedia and Expo, 2003. ICME'03. Proceedings. 2003 International Conference on*, volume 1, pages I–401. IEEE, 2003.

[11] Jen-Chun Lin, Chung-Hsien Wu, and Wen-Li Wei. Error weighted semi-coupled hidden markov model for audio-visual emotion recognition. *IEEE Transactions on Multimedia*, 14(1):142–156, 2012.

[12] Noah Simon, Jerome Friedman, Trevor Hastie, and Robert Tibshirani. A sparse-group lasso. *Journal of Computational and Graphical Statistics*, 22(2):231–245, 2013.

[13] Nengfeng Zhou and Ji Zhu. Group variable selection via a hierarchical lasso and its oracle property. *arXiv preprint arXiv:1006.2871*, 2010.

[14] Michael Grant, Stephen Boyd, and Yinyu Ye. Cvx: Matlab software for disciplined convex programming, 2008.

[15] Jun Liu, Shuiwang Ji, Jieping Ye, et al. Slep: Sparse learning with efficient projections. *Arizona State University*, 6(491):7, 2009.

[16] Julien Mairal, Francis Bach, Jean Ponce, et al. Sparse modeling for image and vision processing. *Foundations and Trends® in Computer Graphics and Vision*, 8(2-3):85–283, 2014.

[17] Jian Huang, Patrick Breheny, and Shuangge Ma. A selective review of group selection in high-dimensional models. *Statistical science: a review journal of the Institute of Mathematical Statistics*, 27(4), 2012.

[18] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

[19] Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320, 2005.

[20] Ming Yuan and Yi Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(1):49–67, 2006.

[21] Sungwan Bang, Jongkyeong Kang, Myoungshic Jhun, and Eunkyung Kim. Hierarchically penalized support vector machine with grouped variables. *International Journal of Machine Learning and Cybernetics*, 8(4):1211–1221, 2017.

[22] Seyoung Kim and Eric P Xing. Tree-guided group lasso for multi-task regression with structured sparsity. 2010.

[23] Silvan Tomkins. *Affect imagery consciousness: Volume I: The positive affects*. Springer publishing company, 1962.

[24] JA Ressel. A circumplex model of affect. *J. Personality and Social Psychology*, 39:1161–78, 1980.

[25] Hugo Lövheim. A new three-dimensional model for emotions and monoamine neurotransmitters. *Medical hypotheses*, 78(2):341–348, 2012.

[26] Israel C Christie and Bruce H Friedman. Autonomic specificity of discrete emotion and dimensions of affective space: a multivariate approach. *International journal of psychophysiology*, 51(2):143–153, 2004.

[27] Chad L Stephens, Israel C Christie, and Bruce H Friedman. Autonomic specificity of basic emotions: Evidence from pattern classification and cluster analysis. *Biological psychology*, 84(3):463–473, 2010.

[28] Haiyan Xu. *Towards Automated Recognition of Human Emotions using EEG*. PhD thesis, 2012.

[29] Paul R Kleinginna and Anne M Kleinginna. A categorized list of emotion definitions, with suggestions for a consensual definition. *Motivation and emotion*, 5(4):345–379, 1981.

[30] Rebecca D Ray. Emotion elicitation using films. *Handbook of emotion elicitation and assessment*, pages 9–28, 2007.

[31] Alexandre Schaefer, Frédéric Nils, Xavier Sanchez, and Pierre Philippot. Assessing the effectiveness of a large database of emotion-eliciting films: A new tool for emotion researchers. *Cognition and Emotion*, 24(7):1153–1172, 2010.

[32] Arnaud Delorme and Scott Makeig. Eeglab: an open source toolbox for analysis of single-trial eeg dynamics including independent component analysis. *Journal of neuroscience methods*, 134(1):9–21, 2004.

[33] Andrea Mognon, Jorge Jovicich, Lorenzo Bruzzone, and Marco Buiatti. Adjust: An automatic eeg artifact detector based on the joint use of spatial and temporal features. *Psychophysiology*, 48(2):229–240, 2011.

[34] Lorenzo Bruzzone and Diego F Prieto. Automatic analysis of the difference image for unsupervised change detection. *IEEE Transactions on Geoscience and Remote sensing*, 38(3):1171–1182, 2000.

[35] Behshad Hosseinifard, Mohammad Hassan Moradi, and Reza Rostami. Classifying depression patients and normal subjects using machine learning techniques and nonlinear features from eeg signal. *Computer methods and programs in biomedicine*, 109(3):339–345, 2013.

[36] Shouyi Wang, Jacek Gwizdka, and W Art Chaovalitwongse. Using wireless eeg signals to assess memory workload in the n-back task. *IEEE Transactions on Human-Machine Systems*, 46(3):424–435, 2016.

[37] Qi Yuan, Weidong Zhou, Shufang Li, and Dongmei Cai. Epileptic eeg classification based on extreme learning machine and nonlinear features. *Epilepsy research*, 96(1):29–38, 2011.

[38] Dale E Olsen, Ronald P Lesser, John C Harris, W Robert S Webber, and John A Cristion. Automatic detection of seizures using electroencephalographic signals, May 17 1994. US Patent 5,311,876.

[39] James F Kaiser. On a simple algorithm to calculate the'energy'of a signal. In *Acoustics, Speech, and Signal Processing, 1990. ICASSP-90., 1990 International Conference on*, pages 381–384. IEEE, 1990.

[40] Rajeev Agarwal and Jean Gotman. Adaptive segmentation of electroencephalographic data using a nonlinear energy operator. In *Circuits and Systems, 1999. ISCAS'99. Proceedings of the 1999 IEEE International Symposium on*, volume 4, pages 199–202. IEEE, 1999.

[41] JC Drummond, CA Brann, DE Perkins, and DE Wolfe. A comparison of median frequency, spectral edge frequency, a frequency band power ratio, total power, and dominance shift in the determination of depth of anesthesia. *Acta Anaesthesiologica Scandinavica*, 35(8):693–699, 1991.

[42] David Grimes, Desney S Tan, Scott E Hudson, Pradeep Shenoy, and Rajesh PN Rao. Feasibility and pragmatics of classifying working memory load with an electroencephalograph. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 835–844. ACM, 2008.

[43] Jochen Gorski, Frank Pfeuffer, and Kathrin Klamroth. Biconvex sets and optimization with biconvex functions: a survey and extensions. *Mathematical Methods of Operations Research*, 66(3):373–407, 2007.

[44] Yuan-Pin Lin, Chi-Hong Wang, Tien-Lin Wu, Shyh-Kang Jeng, and Jyh-Horng Chen. Eeg-based emotion recognition in music listening: A comparison of schemes for multiclass support vector machine. In *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*, pages 489–492. IEEE, 2009.

[45] Song Liu, Haoran Yi, L-T Chia, and Deepu Rajan. Adaptive hierarchical multiclass svm classifier for texture-based image classification. In *Multimedia and Expo, 2005. ICME 2005. IEEE International Conference on*, pages 4–pp. IEEE, 2005.

[46] Georg Buscher, Andreas Dengel, Ralf Biedert, and Ludger V Elst. Attentive documents: Eye tracking as implicit feedback for information retrieval and beyond. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, 1(2):9, 2012.

CHAPTER 6

Overall Conclusion

In this dissertation, various sparsity-inducing mathematical norms including L1-norm, group norm and L2,1-norm were used to improve existing mathematical formulations in machine learning, and the improved learning methods were applied on two problems in bioinformatics - i) predicting optimal tibial soft-tissue insertion of the human knees and ii) emotion recognition and analysis.

To begin with, the novel method of L2,1-regularized multi-response support vector regression (L21-MSVR) in chapter 2 learns well when there are more features than observations and there is a considerable number of response variables. The L2,1-norm is able to select features which learn most if not all of the response variables at the same time. This is particularly suitable for the prediction of optimal tibial soft-tissue insertion of the human knees in chapter 3, where the centroids of each and every of the eight soft tissues and cartilages to be predicted is especially dependent on the shape of the knee. The computational result shows that the proposed three-step spatial-structure supervised learning and prediction model (L21-SSSL) is able to slightly outperform the learning performance in terms of aRRMSE when compared with baseline regressors including GLMNET, which is the strongest competitor in this work.

On the other hand, features in bioinformatics learning can often be arranged in a hierarchical manner. Generalized from sparse group lasso (SGL) and hierarchical lasso (HL), hierarchical sparse group lasso (HSGL) proposed in chapter 4 is particularly successful in identifying a handful but yet important features from a large group of

irrelevant features. Computational results show that HSGL is able to yield comparable learning performance in terms of accuracy (for classification) or mean squared error (for regression) and, at the same time, select less features arranged in a hierarchical manner when compared with HL in both simulated and real-life datasets. The novel HSGL was then applied on an electroencephalography-based emotion recognition problem as in chapter 5. The result of feature selection with SGL and HSGL was analyzed and compared in order to seek the relevant groups of features for group assignment "time epoch", "channel" and "feature group". Such analysis would be useful for researchers and practitioner in designing a light-weighted emotion recognizer.

APPENDIX A

Mathematical Proofs for HSGL

The following proofs are very similar to those in [1, 2, 3]

A.1 Proof of Lemma 4.2.1

Let $Q^*(\lambda_{11}, \lambda_{12}, \lambda_2, \boldsymbol{d}, \boldsymbol{x})$ be the criterion to be minimized in formulation (4.9), and let $Q^\star(\lambda_1, \lambda_2, \boldsymbol{d}, \boldsymbol{x})$ be the criterion to be minimized in formulation (4.14). The goal is to prove that $(\hat{\boldsymbol{d}}^\star = \lambda_{11}\hat{\boldsymbol{d}}^*, \hat{\boldsymbol{x}}^\star = \hat{\boldsymbol{x}}^*/\lambda_{11})$ is a local minimizer of $Q^\star(\lambda_1, \lambda_2, \boldsymbol{d}, \boldsymbol{x})$, and vice versa.

Let $(\hat{\boldsymbol{d}}^*, \hat{\boldsymbol{x}}^*)$ be a local minimizer of $Q^*(\lambda_{11}, \lambda_{12}, \lambda_2, \boldsymbol{d}, \boldsymbol{x})$. It follows that

$$
\begin{aligned}
& Q^*(\lambda_{11}, \lambda_{12}, \lambda_2, \boldsymbol{d}, \boldsymbol{x}) \\
=& \sum_{n=1}^{N} ||y_n - \sum_{k=1}^{K} d_k \sum_{g=1}^{G_k} \boldsymbol{A}_{n,kg}^T \boldsymbol{x}_{kg}||^2 \\
& + \lambda_{11} \sum_{k=1}^{K} d_k + \lambda_{12}||\boldsymbol{x}||_1 + \lambda_2 \sum_{k=1}^{K} d_k^2 \sum_{g=1}^{G_k} ||\boldsymbol{x}_{kg}||_2 \\
=& \sum_{n=1}^{N} ||y_n - \sum_{k=1}^{K} (\lambda_{11} d_k) \sum_{g=1}^{G_k} \boldsymbol{A}_{n,kg}^T \frac{\boldsymbol{x}_{kg}}{\lambda_{11}}||^2 + \sum_{k=1}^{K} (\lambda_{11} d_k) \\
& + \lambda_{11}\lambda_{12} \left\| \frac{\boldsymbol{x}}{\lambda_{11}} \right\|_1 + \lambda_2 \sum_{k=1}^{K} (\lambda_{11} d_k)^2 \sum_{g=1}^{G_k} \left\| \frac{\boldsymbol{x}_{kg}}{\lambda_{11}} \right\|_2 \\
=& \sum_{n=1}^{N} ||y_n - \sum_{k=1}^{K} (\lambda_{11} d_k) \sum_{g=1}^{G_k} \boldsymbol{A}_{n,kg}^T \frac{\boldsymbol{x}_{kg}}{\lambda_{11}}||^2 + \sum_{k=1}^{K} (\lambda_{11} d_k) \\
& + \lambda_1 \left\| \frac{\boldsymbol{x}}{\lambda_{11}} \right\|_1 + \lambda_2 \sum_{k=1}^{K} (\lambda_{11} d_k)^2 \sum_{g=1}^{G_k} \left\| \frac{\boldsymbol{x}_{kg}}{\lambda_{11}} \right\|_2 \\
& = Q^\star(\lambda_1, \lambda_2, \lambda_{11}\boldsymbol{d}, \frac{\boldsymbol{x}}{\lambda_{11}}).
\end{aligned}
\tag{A.1}
$$

Since $(\hat{\boldsymbol{d}}^*, \hat{\boldsymbol{x}}^*)$ is a local minimizer of $Q^*(\lambda_{11}, \lambda_{12}, \lambda_2, \boldsymbol{d}, \boldsymbol{x})$, there exists $\delta > 0$ such that if $(\boldsymbol{d}', \boldsymbol{x}')$ satisfy $||\boldsymbol{d}' - \hat{\boldsymbol{d}}^*||_1 + ||\boldsymbol{x}' - \hat{\boldsymbol{x}}^*||_1 < \delta$, then $Q^*(\lambda_{11}, \lambda_{12}, \lambda_2, \hat{\boldsymbol{d}}^*, \hat{\boldsymbol{x}}^*) \leq Q^*(\lambda_{11}, \lambda_{12}, \lambda_2, \boldsymbol{d}', \boldsymbol{x}')$.

Choose δ' such that $\frac{\delta'}{\min(\lambda_{11}, \frac{1}{\lambda_{11}})} \leq \delta$. Consider any $(\boldsymbol{d}'', \boldsymbol{x}'')$ within a neighborhood satisfying $||\boldsymbol{d}'' - \hat{\boldsymbol{d}}^\star||_1 + ||\boldsymbol{x}'' - \hat{\boldsymbol{x}}^\star||_1 < \delta'$. Thus,

$$\begin{aligned}
& ||\frac{\boldsymbol{d}''}{\lambda_{11}} - \hat{\boldsymbol{d}}^*||_1 + ||\lambda_{11}\boldsymbol{x}'' - \hat{\boldsymbol{x}}^*||_1 \\
\leq & \frac{\lambda_{11}||\frac{\boldsymbol{d}''}{\lambda_{11}} - \hat{\boldsymbol{d}}^*||_1 + \frac{1}{\lambda_{11}}||\lambda_{11}\boldsymbol{x}'' - \hat{\boldsymbol{x}}^*||_1}{\min(\lambda_{11}, \frac{1}{\lambda_{11}})} \\
= & \frac{||\boldsymbol{d}'' - \lambda_{11}\hat{\boldsymbol{d}}^*||_1 + ||\boldsymbol{x}'' - \frac{\hat{\boldsymbol{x}}^*}{\lambda_{11}}||_1}{\min(\lambda_{11}, \frac{1}{\lambda_{11}})} \\
= & \frac{||\boldsymbol{d}'' - \hat{\boldsymbol{d}}^\star||_1 + ||\boldsymbol{x}'' - \hat{\boldsymbol{x}}^\star||_1}{\min(\lambda_{11}, \frac{1}{\lambda_{11}})} \\
< & \frac{\delta'}{\min(\lambda_{11}, \frac{1}{\lambda_{11}})} \\
\leq & \delta.
\end{aligned} \tag{A.2}$$

Consequently, $(\frac{\boldsymbol{d}''}{\lambda_{11}}, \lambda_{11}\boldsymbol{x}'')$ is in a neighborhood of $(\hat{\boldsymbol{d}}^*, \hat{\boldsymbol{x}}^*)$.

Hence,

$$\begin{aligned}
& Q^\star(\lambda_1, \lambda_2, \hat{\boldsymbol{d}}^\star, \hat{\boldsymbol{x}}^\star) \\
= & Q^*(\lambda_{11}, \lambda_{12}, \lambda_2, \hat{\boldsymbol{d}}^*, \hat{\boldsymbol{x}}^*) && \text{By (A.1)} \\
\leq & Q^*(\lambda_{11}, \lambda_{12}, \lambda_2, \frac{\hat{\boldsymbol{d}}''}{\lambda_{11}}, \lambda_{11}\hat{\boldsymbol{x}}'') && \text{By (A.2)} \\
= & Q^\star(\lambda_1, \lambda_2, \frac{\lambda_{11}\hat{\boldsymbol{d}}''}{\lambda_{11}}, \lambda_{11}\frac{\hat{\boldsymbol{x}}''}{\lambda_{11}}) && \text{By (A.1) again} \\
= & Q^\star(\lambda_1, \lambda_2, \hat{\boldsymbol{d}}'', \hat{\boldsymbol{x}}'')
\end{aligned} \tag{A.3}$$

Therefore, $(\hat{\boldsymbol{d}}^\star = \lambda_{11}\hat{\boldsymbol{d}}^*, \hat{\boldsymbol{x}}^\star = \hat{\boldsymbol{x}}^*/\lambda_{11})$ is a local minimizer of $Q^\star(\lambda_1, \lambda_2, \boldsymbol{d}, \boldsymbol{x})$.

It can be similarly proved that, for any minimizer $(\hat{\boldsymbol{d}}^\star, \hat{\boldsymbol{x}}^\star)$ of $Q^\star(\lambda_1, \lambda_2, \boldsymbol{d}, \boldsymbol{x})$, there is a corresponding local minimizer $(\hat{\boldsymbol{d}}^*, \hat{\boldsymbol{x}}^*)$ of $Q^*(\lambda_{11}, \lambda_{12}, \lambda_2, \boldsymbol{d}, \boldsymbol{x})$ such that $\hat{d}^*_k\hat{x}^*_{kg} = \hat{d}^\star_k\hat{x}^\star_{kg}$.

A.2 Proof of Lemma 4.2.2

Suppose $(\hat{\boldsymbol{d}}, \hat{\boldsymbol{x}})$ is a local minimizer of (4.14). Let $\hat{\boldsymbol{\beta}}$ satisfy $\hat{\boldsymbol{\beta}}_{kg} = \hat{d}_k \hat{\boldsymbol{x}}_{kg}$.

To begin with, $\hat{d}_k = 0$ if $\hat{\boldsymbol{x}}_{(k)} = 0$, where $\hat{\boldsymbol{x}}_{(k)} = [\hat{\boldsymbol{x}}_{k1}^T, \hat{\boldsymbol{x}}_{k2}^T, \ldots, \hat{\boldsymbol{x}}_{kG_k}^T]^T$ and $\hat{\boldsymbol{x}} = [\hat{\boldsymbol{x}}_{(1)}^T, \ldots, \hat{\boldsymbol{x}}_{(K)}^T]^T$ because, in Algorithm (2), the first (least-square loss) and third (group-norm penalization) terms of formulation (4.18) will become constant, and the second term ($\sum_{k=1}^K \hat{d}_k$) will achieve minimum when $\hat{d}_k = 0$. It can be similarly proved that, if $\hat{d}_k = 0$, then $\hat{\boldsymbol{x}}_{(k)} = 0$, according to formulation (4.17).

Hence, it remains to prove that if $\hat{d}_k \neq 0$, then $\hat{d}_k = \sqrt{\lambda_1 ||\hat{\boldsymbol{\beta}}_{(k)}||_1}$ and $\hat{\boldsymbol{x}}_{(k)} = \hat{\boldsymbol{\beta}}_{(k)}/\hat{d}_k$. Let $\boldsymbol{\beta}$ be fixed at $\hat{\boldsymbol{\beta}}$. Recall that there are four terms in formulation (4.14) - the loss term $l(\boldsymbol{d}, \boldsymbol{x})$, the term $\boldsymbol{d}$ ($\sum_{k=1}^K d_k$), the term $\boldsymbol{x}$ ($\lambda_1 ||\boldsymbol{x}||_1$) and the group-norm term ($\lambda_2 \sum_{k=1}^K d_k^2 \sum_{g=1}^{G_k} ||\boldsymbol{x}_{kg}||_2 = \lambda_2 ||\boldsymbol{\beta}||_2$). Thus, in order to minimize $Q^\star(\lambda_1, \lambda_2, \boldsymbol{d}, \boldsymbol{x})$, only the terms $\boldsymbol{d}$ ($\sum_{k=1}^K d_k$) and $\boldsymbol{x}$ ($\lambda_1 ||\boldsymbol{x}||_1$) need to be considered because $\boldsymbol{\beta}$ is fixed. For some k with $||\hat{\boldsymbol{\beta}}_{(k)}||_1 \neq 0$, what remains to be optimized in formulation (4.14) is $d_k + \lambda_1 ||\boldsymbol{\beta}_{(k)}||_1 / d_k$, which is minimized at $\hat{d}_k = \sqrt{\lambda_1 ||\hat{\boldsymbol{\beta}}_{(k)}||_1}$ by differentiation with respect to d_k. Substituting this back to $\hat{\boldsymbol{\beta}}_{(k)} = \hat{d}_k \hat{\boldsymbol{x}}_{(k)}$, $\hat{\boldsymbol{x}}_{(k)} = \frac{\hat{\boldsymbol{\beta}}_{(k)}}{\sqrt{\lambda_1 ||\hat{\boldsymbol{\beta}}_{(k)}||_1}}$ follows immediately.

A.3 Proof of Lemma 4.2.3

Suppose $(\hat{\boldsymbol{d}}, \hat{\boldsymbol{x}})$ is a local minimizer of $Q^\star(\lambda_1, \lambda_2, \boldsymbol{d}, \boldsymbol{x})$, the criterion in formulation (4.14) and let $Q(\lambda_1, \lambda_2, \boldsymbol{\beta})$ be the corresponding criterion in formulation (4.15). It is first shown that $\hat{\boldsymbol{\beta}}$, where $\hat{\boldsymbol{\beta}}_{kg} = \hat{d}_k \hat{\boldsymbol{x}}_{kg}$, is a local minimizer of $Q(\lambda_1, \lambda_2, \boldsymbol{\beta})$, i.e. there exists a δ' such that if $||\triangle\boldsymbol{\beta}||_1 < \delta'$, then $Q(\lambda_1, \lambda_2, \hat{\boldsymbol{\beta}}) \leq Q(\lambda_1, \lambda_2, \hat{\boldsymbol{\beta}} + \triangle\boldsymbol{\beta})$.

Denote $\triangle\boldsymbol{\beta} = [\triangle\boldsymbol{\beta}_{(1)}^T, \ldots, \triangle\boldsymbol{\beta}_{(K)}^T]^T$, $\triangle\boldsymbol{\beta}^{(1)} = [\triangle\boldsymbol{\beta}_{(1)}^{(1)T}, \ldots, \triangle\boldsymbol{\beta}_{(K)}^{(1)T}]^T$, $\triangle\boldsymbol{\beta}^{(2)} = [\triangle\boldsymbol{\beta}_{(1)}^{(2)T}, \ldots, \triangle\boldsymbol{\beta}_{(K)}^{(2)T}]^T$, and $\triangle\boldsymbol{\beta} = \triangle\boldsymbol{\beta}^{(1)} + \triangle\boldsymbol{\beta}^{(2)}$ such that, $\forall k = 1, \ldots, K$:

$$\triangle\boldsymbol{\beta}_{(k)} = \begin{cases} \triangle\boldsymbol{\beta}_{(k)}^{(1)} & \text{if } ||\hat{\boldsymbol{\beta}}_{(k)}||_1 \neq 0 \text{ (i.e. } \triangle\boldsymbol{\beta}_{(k)}^{(2)} = 0) \\ \triangle\boldsymbol{\beta}_{(k)}^{(2)} & \text{if } ||\hat{\boldsymbol{\beta}}_{(k)}||_1 = 0 \text{ (i.e. } \triangle\boldsymbol{\beta}_{(k)}^{(1)} = 0) \end{cases}$$

It follows that $||\triangle\boldsymbol{\beta}||_1 = ||\triangle\boldsymbol{\beta}^{(1)}||_1 + ||\triangle\boldsymbol{\beta}^{(2)}||_1$.

It remains to show that $Q(\lambda_1, \lambda_2, \hat{\boldsymbol{\beta}}) \leq Q(\lambda_1, \lambda_2, \hat{\boldsymbol{\beta}} + \triangle\boldsymbol{\beta}^{(1)})$ if δ' is small enough. By Lemma (4.2.2), $\hat{d}_{(k)} = \sqrt{\lambda_1 ||\hat{\boldsymbol{\beta}}_{(k)}||_1}$, $\hat{\boldsymbol{x}}_{(k)} = \frac{\hat{\boldsymbol{\beta}}_{(k)}}{\sqrt{\lambda_1 ||\hat{\boldsymbol{\beta}}_{(k)}||_1}}$ if $|\hat{d}_k|_1 \neq 0$, and $\hat{\boldsymbol{x}}_{(k)} = \mathbf{0}$ if $|\hat{d}_k|_1 = 0$. Furthermore, let $\hat{d}_k{}' = \sqrt{\lambda_1 ||\hat{\boldsymbol{\beta}}_{(k)} + \triangle\boldsymbol{\beta}_{(k)}^{(1)}||_1}$, $\hat{\boldsymbol{x}}_{(k)}{}' = \frac{\hat{\boldsymbol{\beta}}_{(k)} + \triangle\boldsymbol{\beta}_{(k)}^{(1)}}{\sqrt{\lambda_1 ||\hat{\boldsymbol{\beta}}_{(k)} + \triangle\boldsymbol{\beta}_{(k)}^{(1)}||_1}}$ if $|\hat{d}_k|_1 \neq 0$. Let $\hat{d}_k{}' = 0$, $\hat{\boldsymbol{x}}_{(k)}{}' = \mathbf{0}$ if $|\hat{d}_k{}'|_1 = 0$. It follows that $Q^\star(\lambda_1, \lambda_2, \hat{\boldsymbol{d}}', \hat{\boldsymbol{x}}') = Q(\lambda_1, \lambda_2, \hat{\boldsymbol{\beta}} + \triangle\boldsymbol{\beta}^{(1)})$ and $Q^\star(\lambda_1, \lambda_2, \hat{\boldsymbol{d}}, \hat{\boldsymbol{x}}) = Q(\lambda_1, \lambda_2, \hat{\boldsymbol{\beta}})$.

Hence it is only required to show that $Q^\star(\lambda_1, \lambda_2, \hat{\boldsymbol{d}}, \hat{\boldsymbol{x}}) \leq Q^\star(\lambda_1, \lambda_2, \hat{\boldsymbol{d}}', \hat{\boldsymbol{x}}')$.

Since $(\hat{\boldsymbol{d}}, \hat{\boldsymbol{x}})$ is a local minimizer of $Q^\star(\lambda_1, \lambda_2, \boldsymbol{d}, \boldsymbol{x})$, there exists a δ such that $Q^\star(\lambda_1, \lambda_2, \hat{\boldsymbol{d}}, \hat{\boldsymbol{x}}) \leq Q^\star(\lambda_1, \lambda_2, \boldsymbol{d}', \boldsymbol{x}')$ for any $(\boldsymbol{d}', \boldsymbol{x}')$ satisfying $||\boldsymbol{d}' - \hat{\boldsymbol{d}}||_1 + ||\boldsymbol{x}' - \hat{\boldsymbol{x}}||_1 < \delta$.

For $e = \min_k\{||\hat{\boldsymbol{\beta}}_{(k)}||_1 : ||\hat{\boldsymbol{\beta}}_{(k)}||_1 \neq 0\}, \forall k = 1, \ldots, K$ and $\delta' < e/2$,

$$\begin{aligned} & \left|\hat{d}_k{}' - \hat{d}_k\right| \\ = & \left|\sqrt{\lambda_1 ||\hat{\boldsymbol{\beta}}_{(k)} + \triangle\boldsymbol{\beta}_{(k)}^{(1)}||_1} - \sqrt{\lambda_1 ||\hat{\boldsymbol{\beta}}_{(k)}||_1}\right| \\ \leq & \left|\sqrt{\lambda_1 ||\hat{\boldsymbol{\beta}}_{(k)}||_1 + \lambda_1 ||\triangle\boldsymbol{\beta}_{(k)}^{(1)}||_1} - \sqrt{\lambda_1 ||\hat{\boldsymbol{\beta}}_{(k)}||_1}\right| \\ & (\text{By } \sqrt{||\boldsymbol{a} + \boldsymbol{b}||_1} \leq \sqrt{||\boldsymbol{a}||_1 + ||\boldsymbol{b}||_1}) \\ \leq & \frac{\sqrt{\lambda_1} ||\triangle\boldsymbol{\beta}_{(k)}^{(1)}||_1}{2\sqrt{||\hat{\boldsymbol{\beta}}_{(k)}||_1}} \\ & (\text{By } \sqrt{||\boldsymbol{a}||_1 + ||\boldsymbol{b}||_1} - \sqrt{||\boldsymbol{a}||_1} \leq \frac{||\boldsymbol{b}||_1}{2\sqrt{||\boldsymbol{a}||_1}}) \\ \leq & \frac{\sqrt{\lambda_1} ||\triangle\boldsymbol{\beta}_{(k)}^{(1)}||_1}{2\sqrt{e}} \end{aligned} \tag{A.4}$$

Furthermore, for $||\triangle\boldsymbol{\beta}^{(1)}||_1 < \delta'$ (and thus $||\triangle\boldsymbol{\beta}^{(1)}||_1 < e/2$),

$$
\begin{aligned}
& ||\hat{\boldsymbol{x}}_{(k)}{}' - \hat{\boldsymbol{x}}_{(k)}||_1 \\
= & \left\| \frac{\hat{\boldsymbol{\beta}}_{(k)} + \triangle\boldsymbol{\beta}^{(1)}_{(k)}}{\hat{d}_k{}'} - \frac{\hat{\boldsymbol{\beta}}_{(k)}}{\hat{d}_k} \right\|_1 \\
= & \left\| \frac{\triangle\boldsymbol{\beta}^{(1)}_{(k)}}{\hat{d}_k{}'} - \frac{\hat{\boldsymbol{\beta}}_{(k)}(\hat{d}_k{}' - \hat{d}_k)}{\hat{d}_k{}'\hat{d}_k} \right\|_1 \\
\leq & \left\| \frac{\triangle\boldsymbol{\beta}^{(1)}_{(k)}}{\hat{d}_k{}'} \right\|_1 + \left\| \frac{\hat{\boldsymbol{\beta}}_{(k)}(\hat{d}_k{}' - \hat{d}_k)}{\hat{d}_k{}'\hat{d}_k} \right\|_1 \\
= & \left\| \frac{\triangle\boldsymbol{\beta}^{(1)}_{(k)}}{\sqrt{\lambda_1 \left\| \hat{\boldsymbol{\beta}}_{(k)} + \triangle\boldsymbol{\beta}^{(1)}_{(k)} \right\|_1}} \right\|_1 \\
& + \left\| \frac{\hat{\boldsymbol{\beta}}_{(k)} \left(\sqrt{\left\| \hat{\boldsymbol{\beta}}_{(k)} + \triangle\boldsymbol{\beta}^{(1)}_{(k)} \right\|_1} - \sqrt{\left\| \hat{\boldsymbol{\beta}}_{(k)} \right\|_1} \right)}{\sqrt{\lambda_1 \left\| \hat{\boldsymbol{\beta}}_{(k)} \right\|_1 \left\| \hat{\boldsymbol{\beta}}_{(k)} + \triangle\boldsymbol{\beta}^{(1)}_{(k)} \right\|_1}} \right\|_1 \\
\leq & \frac{\left\| \triangle\boldsymbol{\beta}^{(1)}_{(k)} \right\|_1}{\sqrt{\lambda_1 \left\| \hat{\boldsymbol{\beta}}_{(k)} + \triangle\boldsymbol{\beta}^{(1)}_{(k)} \right\|_1}} \\
& + \frac{\left\| \hat{\boldsymbol{\beta}}_{(k)} \right\|_1}{\sqrt{\lambda_1 \left\| \hat{\boldsymbol{\beta}}_{(k)} + \triangle\boldsymbol{\beta}^{(1)}_{(k)} \right\|_1}} \frac{\left\| \triangle\boldsymbol{\beta}^{(1)}_{(k)} \right\|_1}{2\sqrt{\left\| \hat{\boldsymbol{\beta}}_{(k)} \right\|}} \\
\leq & \frac{1}{\sqrt{\lambda_1}} \left(\frac{\left\| \triangle\boldsymbol{\beta}^{(1)}_{(k)} \right\|_1}{\sqrt{\left\| \hat{\boldsymbol{\beta}}_{(k)} + \triangle\boldsymbol{\beta}^{(1)}_{(k)} \right\|_1}} + \frac{\left\| \triangle\boldsymbol{\beta}^{(1)}_{(k)} \right\|_1}{2\sqrt{\left\| \hat{\boldsymbol{\beta}}_{(k)} + \triangle\boldsymbol{\beta}^{(1)}_{(k)} \right\|_1}} \right) \\
\leq & \frac{3\left\| \triangle\boldsymbol{\beta}^{(1)}_{(k)} \right\|_1}{2\sqrt{\lambda_1}} \frac{1}{\sqrt{\left\| \hat{\boldsymbol{\beta}}_{(k)} \right\|_1 - \left\| \triangle\boldsymbol{\beta}^{(1)}_{(k)} \right\|_1}} \\
\leq & \frac{3\left\| \triangle\boldsymbol{\beta}^{(1)}_{(k)} \right\|_1}{2\sqrt{\lambda_1}} \frac{1}{\sqrt{e - e/2}} = \frac{3\left\| \triangle\boldsymbol{\beta}^{(1)}_{(k)} \right\|_1}{\sqrt{2\lambda_1 e}}
\end{aligned}
\tag{A.5}
$$

Therefore, there exists a small enough δ' satisfying $\delta' < e/2$ such that $||\hat{\boldsymbol{d}}' - \hat{\boldsymbol{d}}||_1 + ||\hat{\boldsymbol{x}}' - \hat{\boldsymbol{x}}||_1 < \delta$. Hence $Q^\star(\lambda_1, \lambda_2, \hat{\boldsymbol{d}}, \hat{\boldsymbol{x}}) \leq Q^\star(\lambda_1, \lambda_2, \hat{\boldsymbol{d}}', \hat{\boldsymbol{x}}')$ due to local minimality and $Q(\lambda_1, \lambda_2, \hat{\boldsymbol{\beta}}) \leq Q(\lambda_1, \lambda_2, \hat{\boldsymbol{\beta}} + \triangle\boldsymbol{\beta}^{(1)})$.

The following proves that $Q(\lambda_1, \lambda_2, \hat{\boldsymbol{\beta}} + \triangle\boldsymbol{\beta}^{(1)}) \leq Q(\lambda_1, \lambda_2, \hat{\boldsymbol{\beta}} + \triangle\boldsymbol{\beta}^{(1)} + \triangle\boldsymbol{\beta}^{(2)})$. The Lipschitz continuity of the least-square loss function $l(\boldsymbol{\beta}) = ||\boldsymbol{y} - \boldsymbol{A}\boldsymbol{\beta}||_2$ implies that

$$\begin{aligned} &\left| l\left(\hat{\boldsymbol{\beta}} + \triangle\boldsymbol{\beta}^{(1)} + \triangle\boldsymbol{\beta}^{(2)}\right)\right| - \left| l\left(\hat{\boldsymbol{\beta}} + \triangle\boldsymbol{\beta}^{(1)}\right)\right| \\ \leq\ & L_1{}' \left\|\triangle\boldsymbol{\beta}^{(2)}\right\|_1 \end{aligned} \tag{A.6}$$

for some $L_1{}' > 0$. Moreover, a real number L_1 can be chosen such that

$$\begin{aligned} &\left| l\left(\hat{\boldsymbol{\beta}} + \triangle\boldsymbol{\beta}^{(1)} + \triangle\boldsymbol{\beta}^{(2)}\right)\right| - \left| l\left(\hat{\boldsymbol{\beta}} + \triangle\boldsymbol{\beta}^{(1)}\right)\right| \\ =\ & L_1 \left\|\triangle\boldsymbol{\beta}^{(2)}\right\|_1 \end{aligned} \tag{A.7}$$

For the group-norm term ($\lambda_2 \sum_{k=1}^{K} d_k^2 \sum_{g=1}^{G_k} ||\boldsymbol{x}_{kg}||_2 = \lambda_2 ||\boldsymbol{\beta}||_2$), there exists a number $\epsilon > 0$ such that

$$\lambda_2 \left\|\hat{\boldsymbol{\beta}} + \triangle\boldsymbol{\beta}^{(1)} + \triangle\boldsymbol{\beta}^{(2)}\right\|_2 - \lambda_2 \left\|\hat{\boldsymbol{\beta}} + \triangle\boldsymbol{\beta}^{(1)}\right\|_2 = \lambda_2\epsilon > 0 \tag{A.8}$$

Hence, there exists the following numbers $L_1, L_2 \in \mathbb{R}$ such that

$$\begin{aligned} &Q(\lambda_1, \lambda_2, \hat{\boldsymbol{\beta}} + \triangle\boldsymbol{\beta}^{(1)} + \triangle\boldsymbol{\beta}^{(2)}) - Q(\lambda_1, \lambda_2, \hat{\boldsymbol{\beta}} + \triangle\boldsymbol{\beta}^{(1)}) \\ =\ & L_1 \left\|\triangle\boldsymbol{\beta}^{(2)}\right\|_1 + 2\sqrt{\lambda_1} \sum_{k=1}^{K} \sqrt{\left\|\boldsymbol{\beta}_{(k)}^{(2)}\right\|_1} + \lambda_2\epsilon \end{aligned} \tag{A.9}$$

Since $\left\|\triangle\boldsymbol{\beta}^{(2)}\right\|_1 < \delta'$, for a small enough δ', the second and third terms dominate the first term in (A.9) if λ_1 is large enough and λ_2 is small enough. It follows that $Q(\lambda_1, \lambda_2, \hat{\boldsymbol{\beta}} + \triangle\boldsymbol{\beta}^{(1)}) \leq Q(\lambda_1, \lambda_2, \hat{\boldsymbol{\beta}} + \triangle\boldsymbol{\beta}^{(1)} + \triangle\boldsymbol{\beta}^{(2)})$.

Overall, there exists a small enough $\delta' > 0$ such that, if $||\triangle\boldsymbol{\beta}||_1 < \delta'$, then $Q(\lambda_1, \lambda_2, \hat{\boldsymbol{\beta}}) \leq Q(\lambda_1, \lambda_2, \hat{\boldsymbol{\beta}} + \triangle\boldsymbol{\beta})$, which implies that $\hat{\boldsymbol{\beta}}$ is a local minimizer of $Q(\lambda_1, \lambda_2, \boldsymbol{\beta})$.

Similarly, it can be proved that if $\hat{\boldsymbol{\beta}}$ is a local minimizer of $Q(\lambda_1, \lambda_2, \boldsymbol{\beta})$, and if it is assumed that $\hat{d}_k = \sqrt{\lambda_1 \left\|\hat{\boldsymbol{\beta}}_{(k)}\right\|_1}$, $\hat{\boldsymbol{x}}_{(k)} = \frac{\hat{\boldsymbol{\beta}}_{(k)}}{\sqrt{\lambda_1 \left\|\hat{\boldsymbol{\beta}}_{(k)}\right\|_1}}$ for $\left\|\hat{\boldsymbol{\beta}}_{(k)}\right\|_1 \neq 0$ and $\hat{d}_k = 0$, $\hat{\boldsymbol{x}}_{(k)} = \mathbf{0}$ for $\left\|\hat{\boldsymbol{\beta}}_{(k)}\right\|_1 = 0$, then $(\hat{\boldsymbol{d}}, \hat{\boldsymbol{x}})$ is a local minimizer of $Q^\star(\lambda_1, \lambda_2, \boldsymbol{d}, \boldsymbol{x})$.

A.4 Proof of Lemma 4.3.1

To begin with, $Q^\star(\lambda_1, \lambda_2, \boldsymbol{d}, \boldsymbol{x})$ is bounded below due to the positivity of each term in the criterion. Denote $\hat{\boldsymbol{x}}^{(t)}$ and $\hat{\boldsymbol{d}}^{(t)}$ the minimizers of the criterion in problems (4.17) and (4.18) at the t-th iteration respectively. Note that the minimization problems in Steps 5 and 6 of algorithm (2) are equivalent to minimizing $Q^\star(\lambda_1, \lambda_2, \hat{\boldsymbol{d}}^{(t-1)}, \boldsymbol{x})$ with respect to $\boldsymbol{x}$ and $Q^\star(\lambda_1, \lambda_2, \boldsymbol{d}, \hat{\boldsymbol{x}}^{(t)})$ with respect to $\boldsymbol{d}$. Since $\hat{\boldsymbol{x}}^{(t)}$ is the minimizer of $Q^\star(\lambda_1, \lambda_2, \hat{\boldsymbol{d}}^{(t-1)}, \boldsymbol{x})$, thus $Q^\star(\lambda_1, \lambda_2, \hat{\boldsymbol{d}}^{(t-1)}, \hat{\boldsymbol{x}}^{(t)}) \leq Q^\star(\lambda_1, \lambda_2, \hat{\boldsymbol{d}}^{(t-1)}, \hat{\boldsymbol{x}}^{(t-1)})$. Similarly, since $\hat{\boldsymbol{d}}^{(t)}$ is the minimizer of $Q^\star(\lambda_1, \lambda_2, \boldsymbol{d}, \hat{\boldsymbol{x}}^{(t)})$, thus $Q^\star(\lambda_1, \lambda_2, \hat{\boldsymbol{d}}^{(t)}, \hat{\boldsymbol{x}}^{(t)}) \leq Q^\star(\lambda_1, \lambda_2, \hat{\boldsymbol{d}}^{(t-1)}, \hat{\boldsymbol{x}}^{(t)})$.

Combining these two together, it follows that,

$$\begin{aligned} Q(\lambda_1, \lambda_2, \hat{\boldsymbol{d}}^{(t)}, \hat{\boldsymbol{x}}^{(t)}) &\leq Q^\star(\lambda_1, \lambda_2, \hat{\boldsymbol{d}}^{(t-1)}, \hat{\boldsymbol{x}}^{(t)}) \\ &\leq Q^\star(\lambda_1, \lambda_2, \hat{\boldsymbol{d}}^{(t-1)}, \hat{\boldsymbol{x}}^{(t-1)}). \end{aligned} \tag{A.10}$$

which implies that $Q^\star(\lambda_1, \lambda_2, \boldsymbol{d}, \boldsymbol{x})$ decreases for each iteration.

Bibliography

[1] Nengfeng Zhou and Ji Zhu. Group variable selection via a hierarchical lasso and its oracle property. *arXiv preprint arXiv:1006.2871*, 2010.

[2] Sungwan Bang, Jongkyeong Kang, Myoungshic Jhun, and Eunkyung Kim. Hierarchically penalized support vector machine with grouped variables. *International Journal of Machine Learning and Cybernetics*, 8(4):1211–1221, 2017.

[3] SIJIAN Wang, B Nan, N Zhu, and J Zhu. Hierarchically penalized cox regression with grouped variables. *Biometrika*, 96(2):307–322, 2009.

BIOGRAPHICAL STATEMENT

Kin Ming (Frank) Puk received the Bachelor's degree in Information Engineering from the Chinese University of Hong Kong and the Master's degree in Operations Research from the University of Southern California. His research interests include medical imaging, data mining and optimization.