OPTIMAL PARAMETER SETTING OF SINGLE AND MULTI-TASK LASSO

A Thesis

Submitted to the Faculty

of

Purdue University

by

Huiting Su

In Partial Fulfillment of the

Requirements for the Degree

of

Master of Science

December 2018

Purdue University

West Lafayette, Indiana

THE PURDUE UNIVERSITY GRADUATE SCHOOL
STATEMENT OF THESIS APPROVAL

Dr. Hong Wan, Chair

School of Industrial Engineering

Dr. Cai Hua

School of Industrial Engineering

Dr. Steven Landry

School of Industrial Engineering

Approved by:

Dr. Abhijit Deshmukh

Head of Industrial Engineering

ACKNOWLEDGMENTS

I would like to thank Professor Hong Wan for her patience and guidance not only in research, but also in my study and career. I would also like to thank all my committee members for their help.

I thank my parents for their support, without which I would not have the opportunity to do research in US. My friend Yinzhen Li in the Math department also gave me a lot of help with my research and I really appreciate.

TABLE OF CONTENTS

Page

LIST OF TABLES

ABSTRACT

Su, Huiting MS, Purdue University, December 2018. Optimal Parameter Setting of Single and Multi-task LASSO. Major Professor: Hong Wan.

This thesis considers the problem of feature selection when the number of predictors is larger than the number of samples. The performance of supersaturated design (SSD) working with least absolute shrinkage and selection operator (LASSO) is studied in this setting. In order to achieve higher feature selection correctness, self-voting LASSO is implemented to select the tuning parameter while approximately optimize the probability of achieving Sign Correctness. Furthermore, we derive the probability of achieving Direction Correctness, and extend the self-voting LASSO to multi-task self-voting LASSO, which has a group screening effect for multiple tasks.

1. INTRODUCTION

1.1 Feature Selection

Modern regression analysis often involves huge number of features, which is also called variables. They are predictors that may have impact on the response. Ordinary least squares (OLS) method is often used to estimate coefficients for the variables.

A situation comes up more and more often, where the number of features p is larger than the number of observations n. For example, we want to classify healthy patients from cancer patients based on their gene expression. No more than 100 observations are available, while there can be 6000 to 60,000 features in the gene data [1]. In this situation, ordinary least squares (OLS) method cannot be used to fit the models as it requires at least p samples for p variables. Thus, it is necessary to reduce the number of features.

Feature selection is a way to screen the features and only keep the features that are most informative. It is particularly important when the true underlying model is sparse, and it can be used to remove redundant or irrelevant features for further study. The benefits of feature selection include: 1) Avoid over-fitting, and improve the prediction performance of the model; 2) Construct more interpretable model, and help understand the underlying mechanism; 3) Provide models that can save computational effort.

Typical domains using feature selection include literature mining, gene expression array analysis, and combinatorial chemistry. For example, feature selection is becoming a prerequisite for constructing models in bioinformatics due to the high dimensional nature of models in this field [2].

Popular feature selection methods include subset selection, dimension reduction, and regularization (also known as shrinkage). Subset selection is a group of methods

that fit a model on a subset of features, and select a subset of features which gives the best model. It includes best subset selection, stepwise selection, and backward stepwise selection (also known as backward elimination). The best-subset selection method simply fits a model on each subset of features and compares the error rate of all the models, which requires large amount of computational effort. Stepwise selection and backward stepwise selection are greedy search strategies that make changes to the chosen subset progressively [1]. However, the stepwise selection methods only improve the predicting performance under certain circumstances, for example, when few predictors have a strong relationship with the response. Dimension reduction methods like principal components regression and partial least squares transform the predictors and fit OLS model on the transformed variables. The principal components regression selects linear combinations that represent the predictors best while reducing the dimention, but this selection is not supervised, which means that the predicting performance is not guaranteed to be the best with the chosen linear combination. The partial least squares takes the response into consideration, but generally does not achieve much performance improvement in practice. Shrinkage methods add regularization term into the objective function, or equivalently add constraints to the model, so that some coefficients are shrunken towards zero. LASSO (least absolute shrinkage and selection operator), and Elastic Net are shrinkage methods.

1.2 LASSO

The LASSO (least absolute shrinkage and selection operator) proposed by Robert Tibshirani [3] is a widely used regularization technique to simultaneously conduct feature selection and regression.

For OLS, the goal is to minimize the residual sum of squares (RSS). LASSO is very similar to a technique called Ridge Regression which adds an l_2 penalty to the RSS, but LASSO uses l_1 penalty instead. By adding the constraint that the sum of the absolute value of the regression coefficients should be less than a specific value,

LASSO forces some certain coefficients to be shrunken to zero. For ridge regression, the coefficients of irrelevant variables can be very close to zero, but not exactly zero, which means that it keeps all the p variables in the model. On the contrary, LASSO can shrink some coefficients to zero, and thus perform variable selection. Its ability of performing variable selection lies on the form of the l_1 penalty.

There are many variants proposed in the literature, such as fused LASSO, elastic net, adaptive LASSO, and group LASSO. Fused LASSO is proposed in 2005 by Robert Tibshirani for problems in which features can be ordered according to some criterion, and it encourages not only the sparsity of coefficients, but also sparsity of their differences [4]. While the LASSO penalty does not take the correlation between variables into consideration, ridge regression has the property of shrinking strongly correlated variables towards each other. Elastic net penalty combines the LASSO penalty and ridge penalty by adding them together, and thus has the property of both methods [5]. Adaptive lasso proposed by Zou et al. in 2006 uses a weighted penalty to avoid falsely selecting noise predictors [6]. Group LASSO proposed by Yuan et al. selects specific features that are grouped together simultaneously for a task [7]. In our research, the original version of LASSO is considered, but it can also be extended to the variants.

1.3 Parameter Tuning for LASSO

The parameter λ has tremendous effect on the performance of LASSO, so a method to determine the value of λ is of significant importance. As LASSO perform feature selection and coefficient estimation simultaneously, the performance of LASSO can be measured by the prediction error or feature selection accuracy.

Cross Validation (CV) methods and information criteria is often used in model selection. The computation effort for information criteria is less, but the degrees of freedom (DF) of LASSO and error variance are required to calculate the information

criteria [8]. Thus, cross validation is a preferable method for model selection. However, these model selection methods focus only on minimizing the prediction error.

On the other hand, the self-voting method proposed in the work of Dadi and Wan [9] maximizes the feature selection accuracy of LASSO. This method determines the value of λ based on the probability that LASSO selects the correct features. In our research, the performance measure of LASSO is also feature selection accuracy, and the parameter λ is tuned to maximize it.

1.4 SSD Working with LASSO

Generally, data gathering and feature selection are considered separately. When investigating complex systems like large scale computer simulation models, the value or level of features can be varied according to the experimenter's choice. In order to figure out the features which have significant effects on the system performance measured by some specific performance metric, a screen experiment is often conducted on the system.

However, simulation is very computationally expensive. A simulation run for a system with moderate complexity may last a few days. The screening process using factorial design can be computationally unaffordable as it requires a large number of runs. Therefore, an effective experimental design should help experimenter to reduce simulation run and allocate computational effort to more informative combination of feature levels.

Booth and Cox [10] first introduced a cost-effective design called supersaturated design (SSD), in which the number of runs n in the design is smaller than the number of parameters p. Various techniques have been developed to construct SSD with optimal performance, and they are widely implemented in practice.

Dadi proposed a LASSO-based optimality criteria and partial gradient algorithms to construct supersaturated designs which approximately optimize the variable selec-

tion performance of LASSO. In our research, we use the algorithm proposed by Dadi to generate design for our screening process.

1.5 Multi-task Feature Selection

In the setting of multi-task learning, there are $l > 1$ responses which can also be called tasks. Our objective is to solve several regression problems for related tasks at once. The basic assumption in Multi-task feature selection is tasks are related. While learning independently using single-task feature selection methods for different tasks may achieve lower error, these tasks share common underlying sparsity, which is usual in practice. Learning one task can benefit from the information of other tasks. Thus learning the models simultaneously can give more interpretable model and can be useful in practice.

Obozinski et al. [11] extended the single-task LASSO to a block-regularization method called multi-task LASSO, which enforces the feature selection to be the same across tasks. Instead of penalizing the sum of l_1 norm, the multi-task LASSO penalizes the sum of l_2 norms of a block of coefficients for each feature. The coefficients are shrunken to zero simultaneously when a feature is not significant for any task, which is a grouping effect. The performance of multi-task LASSO depends on the level of sharing. Here we consider the scenario where all tasks share a subset of significant features.

1.6 Objective and Contribution

In the first part of this thesis, we study the performance of single-task LASSO working with SSD through a realistic case. In the second part, we derive the probability of multi-task LASSO achieving Direction Correctness, and demonstrate the self-voting procedure for multi-task LASSO selecting tuning parameter. Then experiments are conducted to compare the performance of LASSO when selecting tuning parameter with self-voting procedure or cross-validation.

2. PROBLEM STATEMENT

The problem we want to solve is select feature when the number of experiment runs is less than the number of factors.

2.1 Feature Selection

For single response, consider the linear model

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta}^0 + \boldsymbol{\epsilon} \tag{2.1}$$

where $\boldsymbol{y} = (y_1, y_2, ..., y_n)^T$ is the response vector which is n dimensional, $\boldsymbol{X}$ is an $n \times p$ design matrix, $\boldsymbol{\beta}^0$ is a p dimensional vector of the true coefficients of the predictors, and $\boldsymbol{\epsilon}$ is an n dimensional random noise vector in which all the elements are independent and identically normal distributed.

The Ordinary Least Square (OLS) estimate for the regression coefficients is

$$\hat{\boldsymbol{\beta}} = argmin_{\boldsymbol{\beta}}||\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}||^2 = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y} \tag{2.2}$$

When p is larger than n, $\boldsymbol{X}^T\boldsymbol{X}$ is singular, and not invertible. Regularization is introduced into the model to resolve this issue. With L1 penalization, the estimate becomes

$$\hat{\boldsymbol{\beta}}^\lambda = argmin_{\boldsymbol{\beta}}\frac{1}{2n}||\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}||_2^2 + \lambda||\boldsymbol{\beta}||_1 \tag{2.3}$$

where $||\boldsymbol{\beta}||_1 = \sum_j |\beta_j|$ is the l_1-norm penalty.

2.2 Multi-task Feature Selection

Under the setting of multi-task learning, there are l linear models we want to learn. The model becomes

$$\boldsymbol{Y} = \boldsymbol{X}\boldsymbol{B}^0 + \boldsymbol{E} \tag{2.4}$$

where $\boldsymbol{Y} = (\boldsymbol{y_1}, \boldsymbol{y_2}, ..., \boldsymbol{y_l})$ is a $n \times l$ matrix of responses of l tasks, $\boldsymbol{X}$ is an $n \times p$ design matrix as previous, $\boldsymbol{B}^0 = (\boldsymbol{\beta}_1^0, \boldsymbol{\beta}_2^0, ..., \boldsymbol{\beta}_l^0)$ is a $p \times l$ true coefficient matrix of the predictors for l tasks, and $\boldsymbol{E}$ is an $n \times l$ dimensional random noise vector in which all the elements are independent and identically normal distributed.

Using multi-task lasso penalty [11], the solution is

$$\hat{\boldsymbol{B}}^{\lambda} = min_{\boldsymbol{B}} \frac{1}{2n} ||\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{B}||_{Fro}^2 + \lambda ||\boldsymbol{B}||_{21} \tag{2.5}$$

where $||\boldsymbol{A}||_{Fro} = \sqrt{\sum_{ij} a_{ij}^2}$ is the Frobenius norm, and $||\boldsymbol{B}||_{21} = \sum_{j=1}^{p} ||\boldsymbol{B}_j||_2$ is the $l_{2,1}$ norm.

3. CASE STUDY

3.1 Introduction

The performance of SSD working with self-voting LASSO is studied in this chapter with a realistic case.

3.2 Case Description

A simplified manufacturing system case is studied in previous work implementing CSB (Controlled Sequential Bifurcation) method [12]. Furthermore, the results of CSB-X method and factorial design are reported in [13]. In the case, we want to figure out which machines to invest among 20 types of machines. In this section, partial gradient algorithm will be used to construct an RSS-Optimal SSD for experiment, and Self-Voting LASSO (least absolute shrinkage and selection operator) will be implemented to screen significant features. For detailed introduction of the system, please refer to [12].

The features which influence the system performance are the number of fast or slow machines at each station and the number of each kind of transporters. As shown in table 3.1, the current number of machines /transporters is taken as the "level - 1", and "level 0" and "level 1", which are determined according to the cost of each machine/transporter for experiment.

Performance measure of the system is the weighted mean cycle time of different products, which we want to minimize. It is obtained from discrete event simulation. There are 20 features in total, and simulation is time consuming for such a large system. Thus, supersaturated design experiment is needed and efficient in this case.

Table 3.1.
Feature Description and Levels (Unit Number)

Feature	Feature Description	Level -1	Level 0	Level 1
1	slow machines in OXIDIZE	92	93	94
2	fast machines in STEPPER	0	1	2
3	fast machines in COAT	0	2	4
4	slow machines in CLEAN	42	45	48
5	fast machines in TEST1	0	2	4
6	fast machines in TEST2	0	2	4
7	slow machines in STEPPER	30	32	34
8	slow machines in COAT	25	29	33
9	fast machines in CLEAN	0	2	4
10	slow machines in TEST1	21	25	29
11	slow machines in TEST2	21	25	29
12	slow machines in LOAD QUARTZ	5	13	21
13	slow machines in UNLOAD QUARTZ	5	13	21
14	fast machines in LOAD QUARTZ	0	5	10
15	fast machines in UNLOAD QUARTZ	0	5	10
16	AGVs	0	5	10
17	slow machines in DEVELOP	10	31	52
18	CONVEYORS	6	9	12
19	fast machines in OXIDIZE	0	1	2
20	fast machines in DEVELOP	0	13	26

3.3 Description of Experiments

To identify significant features among the 20 features, partial gradient method using RSS criteria is used to generate a 12×20 supersaturated design matrix X_c, with $RSS(k = 2) = 382.4067, ES^2 = 6.9053, D - Optimal(k = 2) = 0.97491, A - Optimal(k = 2) = 1.0587$.

$$X_c = \begin{bmatrix}
+ & - & + & + & + & - & + & - & - & - & - & + & + & + & - & - & - & - & + & - \\
- & - & - & - & - & + & + & + & - & + & + & - & - & + & + & - & - & + & + & - \\
+ & + & - & + & + & + & - & + & - & + & + & + & + & - & + & - & - & + & - & + \\
+ & + & - & - & - & + & + & + & + & - & - & + & + & + & + & + & - & - & + & + \\
+ & + & + & - & - & - & - & + & - & + & - & - & - & - & - & - & + & - & + & + \\
- & - & + & - & - & - & + & - & - & - & + & + & - & - & - & + & - & + & - & + \\
+ & - & - & + & + & - & + & - & + & + & + & - & - & + & + & + & + & - & - & + \\
+ & - & - & + & - & + & - & - & + & + & - & + & + & - & - & + & + & + & + & - \\
- & + & - & - & + & - & - & - & - & - & - & - & + & + & - & + & + & + & - & - \\
- & + & + & + & + & + & + & + & + & + & - & - & - & - & - & + & - & - & - & - \\
- & - & + & - & + & + & - & - & + & - & + & - & + & - & + & - & + & - & + & + \\
- & + & + & + & - & - & - & + & + & - & + & + & - & + & + & - & + & + & - & -
\end{bmatrix}.$$

The simulation model of the system is developed in Arena©, a discrete event simulation and automation software. For each replication, 365 days of operation are simulated with a 300-hour warm-up period to eliminate the influence of initial conditions. We code in Visual Basic for Applications (VBA) to control the simulation, and collect output data automatically.

Input of the simulation model is number of machines, according to the high or low level in the SSD matrix X_c. The performance measure calculated from the simulation is the long-run average cycle time (hours) weighted by percentage of 4 different products (A: 0.15, B: 0.35, C: 0.3, D: 0.2), taken the average of 10 replications. There are 100 simulations in total, each contains 10 replications.

Then we implement Self-voting LASSO studied in [9] to screen significant features for every simulation. When using LASSO to conduct feature selection, we need to tune the parameter λ. Self-voting LASSO on the other hand, can select λ automatically

while maximizing the correct selection probability of LASSO. For every simulation, there is a set of significant features selected. For the 100 runs of Self-voting LASSO, count the selected frequency of each feature, and estimate the probability $P_{sv}^{-1,0}$ of LASSO selecting each feature.

The experiment is conducted on a ThinkPad© X1 carbon laptop, with Intel(R) Core(TM) i5-5200U CPU@2.20 GHz, and 64-bit operating system.

3.4 Numerical Results

In order to make the experiment comparable to result of CSB in [12], and result of CSB-X and the factorial design in [13], two experiments are conducted:

- Experiment 1 take level -1 and level 0 as low level and high level, compared with CSB result;
- Experiment 2 take level -1 and level 1 as low level and high level, compared with CSB-X and factorial design result.

Result of the two experiments are reported in table 3.2, where $\hat{P}_{sv}^{-1,0}$ and $\hat{P}_{sv}^{-1,0}$ are the estimated probabilities of self-voting method selecting each feature in experiment 1 and experiment 2 respectively. $\hat{\beta}^{CSB}$ and $\hat{\beta}^{CSB-X}$ are the estimated regression coefficients of CSB and CSB-X respectively. Features selected by CSB, CSB-X and factorial design are shown in bold font. For $\hat{P}_{sv}^{-1,0}$ and $\hat{P}_{sv}^{-1,0}$, the features with chosen probability larger than 80% are shown in bold font. We can see that the self-voting result in experiment 2 is highly consistent with results of CSB-X and factorial design.

Note that the response is simply the weighted average cycle time of four different products. The actual goal is to reduce the cycle time of all the four products simultaneously. If changing the level of some feature gives rise to the cycle time of some products and reduces the cycle time of the others, then the effect on the weighted average cycle time can be counteracted. The reason why the weighted average cycle time works in this case is all features have same direction of effects on the cycle time

Table 3.2.
Self-Voting Lasso Result vs CSB, CSB-X and Factorial Design

Feature	$\hat{P}_{sv}^{-1,0}$	$\hat{\beta}^{CSB}$	$\hat{P}_{sv}^{-1,1}$	$\hat{\beta}^{CSB-X}$	95% Confidence Interval from the Factorial Design
1	0	0.08	0	-0.00527	(-0.0057, 0.0467)
2	0.01	-1.39	0.11	-1.269	**(-1.333, -1.205)**
3	**1**	**-2.2**	**0.8**	**-2.006**	**(-2.070, -1.943)**
4	0	-0.39	0.02	-0.162	(-0.206, -0.118))
5	0	**-3.03**	**0.89**	**-2.772**	**(-2.812, -2.732)**
6	0	**-3.27**	**0.8**	**-2.784**	**(-2.815, -2.753)**
7	0	-0.14	0	-0.239	(-0.283, -0.195)
8	0	-0.02	0	-0.295	(-0.344, -0.247)
9	**0.92**	-1.34	**1**	-1.404	**(-1.471 -1.338)**
10	0	0.19	0	-0.291	(-0.335, -0.247)
11	0	-0.14	0	-0.281	(-0.325, -0.238)
12	0.79	**-1.62**	0.04	-0.846	(-0.888, -0.804)
13	**0.84**	**-1.56**	**0.98**	-0.838	(-0.895, -0.781)
14	0	-0.83	0	**-1.676**	**(-1.709, -1.644)**
15	**1**	-0.74	**1**	**-1.638**	**(-1.674, -1.603)**
16	**1**	**-4.14**	**1**	**-3.169**	**(-3.211, -3.126)**
17	0	**-1.78**	0.05	-0.84	(-0.858, -0.822)
18	0	0.08	0	**-1**	(-1.053, -0.942)
19	**0.89**	-0.55	**1**	-0.569	(-0.607, -0.532)
20	0.29	**-1.77**	0.27	**-2.476**	**(-2.566, -2.385)**

of each product. Adding machines in the system would not prolong the cycle time of any product. Implement LASSO separately on the cycle time of four products in one simulation, and report the result in 3.3. For each feature, the coefficients for different products have the same sign (positive or negative).

Table 3.3.
LASSO Result for Different Products

Feature	A	B	C	D	Feature	A	B	C	D
1	0	0	0	0	11	0	0	0	0
2	0	0	0	0	12	0	0	0	0
3	0.66	0.79	0.32	0.67	13	-1.22	-0.16	0	0
4	0	0	0	0	14	0	0	0	0
5	0.00	0	0	0	15	-3.46	-1.64	-0.44	-0.89
6	0.00	0	0	0	16	-1.93	-0.66	0	-0.21
7	0	0	0	0	17	0	0	0	0
8	0	0	0	0	18	0	0	0	0
9	-1.60	-1.41	-0.63	-0.92	19	2.16	1.16	0.24	0.51
10	0	0	0	0	20	0	0	0	0

Conduct experiments on the scenario of varying the value of one feature reduce some responses and raise the others. Keep the same number of observations and number of features, and the same design. Generate 100 true model randomly with two significant features in each. The coefficients of the true features are generated following a uniform distribution, with different signs. The response is also the weighted average cycle time of four products.

Use LASSO to perform feature selection and regression on every model. Only for 59 model out of 100 models, the LASSO selected the correct significant features in the true model. On the contrary, for 95 out of 100 models in which same features have same effects on different products, the LASSO selected the correct significant features. A typical sample result is shown in table 3.4. For this model, the coefficients of the 7th feature for different products are all positive, so the feature is chosen and the estimated coefficient is close. However, for the 11th feature, the effects on different products are counteracted. Thus, this feature is not selected by LASSO, though it is significant.

Table 3.4.
Result of Scenario with Different Effect

Feature	β_A	β_B	β_C	β_D	$\hat{\beta}$	Feature	β_A	β_B	β_C	β_D	$\hat{\beta}$
1	0	0	0	0	0	11	-3.11	-5.54	6.45	3.47	0
2	0	0	0	0	0	12	0	0	0	0	0
3	0	0	0	0	0	13	0	0	0	0	0
4	0	0	0	0	0	14	0	0	0	0	0
5	0	0	0	0	0	15	0	0	0	0	0
6	0	0	0	0	0	16	0	0	0	0	0
7	4.21	4.95	6.90	6.12	4.42	17	0	0	0	0	0
8	0	0	0	0	0	18	0	0	0	0	0
9	0	0	0	0	0	19	0	0	0	0	0
10	0	0	0	0	0	20	0	0	0	0	0

In order to overcome the drawback in this scenario, a different way of adding penalty which can consider different effects on multiple responses is needed. As mentioned in the introduction, Obozinski et al. [11] proposed a method called multi-task LASSO, which select features considering across tasks (different responses). This method will be discussed in detail and further studied in the following chapter.

4. MULTI-TASK FEATURE SELECTION

4.1 Introduction

This chapter derives the probability of multi-task LASSO achieving Direction Correctness (PDC), and demonstrate the self-voting procedure to select tuning parameter using the PDC.

4.2 Correct Selection

The predictor is said to be a true variable if $||\boldsymbol{\beta}_j||_2 \neq 0$, and is said to be untrue if $||\boldsymbol{\beta}_j||_2 = 0$. Denote the set of indexes of true variables as $A = \{j : ||\boldsymbol{\beta}_j||_2 \neq 0\}$, and the set of indexes of untrue variables as $C = \{j : ||\boldsymbol{\beta}_j||_2 = 0\}$. The number of true variables is denoted as q.

There are two types of correctness that we consider: the estimated probability of achieving Sign Correctness (SC) and probability of achieving Direction Correctness (DC). Let $sgn(\cdot)$ be the function giving the sign of each element in a matrix or vector, and $D(\cdot)$ be the function giving the direction of each row vector in a matrix. For example, $D(\boldsymbol{\beta}_j) = \frac{\boldsymbol{\beta}_j}{||\boldsymbol{\beta}_j||_2}$.

If and only if the true variables are selected, and the signs of the selected variables are equal to the sign of true variables, sign correctness is achieved. In addition to sign correctness, if for each selected variable, the vector of estimated coefficients is parallel with the vector of true coefficients, direction correctness is achieved.

Note that when $l = 1$, $D(\boldsymbol{\beta}_j) = \frac{\boldsymbol{\beta}_j}{|\boldsymbol{\beta}_j|} = sgn(\boldsymbol{\beta}_j)$, which means Direction Correctness degenerates to the Sign Correctness of LASSO for single task.

4.3 Probability of Direction Correctness

Denote the rows of matrices with subscripts, and the columns with superscripts. For example, $\boldsymbol{X^1}$ is the first column of matrix $\boldsymbol{X}$, and $\boldsymbol{\beta_1}$ is the first row of matrix $\boldsymbol{B}$.

The multi-task loss function 2.5 is convex, so the solution that minimizes the loss function must satisfy the KKT condition. The gradient of loss function is $\forall j, k$

$$\frac{1}{2n}\frac{\partial \sum_{k=1}^{l}\sum_{i=1}^{n}[y_{ik}-(x_{i1}\beta_{1k}+x_{i2}\beta_{2k}+...+x_{ij}\beta_{jk}+...)]^2}{\partial \beta_{jk}}+\lambda\cdot\frac{\partial \sum_{j=1}^{p}\sqrt{\sum_{k=1}^{l}\beta_{jk}^2}}{\partial \beta_{jk}} \tag{4.1}$$

$$\frac{1}{n}(\boldsymbol{X}^j)^T(-\boldsymbol{Y}+\boldsymbol{X}\boldsymbol{B})+\lambda\boldsymbol{g} \tag{4.2}$$

Where $\boldsymbol{g}=\frac{\partial||\boldsymbol{\beta}_j||_2}{\partial \beta_{jk}}$. However, when $||\boldsymbol{\beta}_j||_2=0$, the loss function is not differentiable, and subgradient is used in this case.

$$\boldsymbol{g}_j=\frac{\boldsymbol{\beta}_j}{||\boldsymbol{\beta}_j||_2}, \qquad \forall j \text{ s.t. } ||\boldsymbol{\beta}_j||_2\neq 0 \tag{4.3}$$

$$\boldsymbol{g}_j : ||\boldsymbol{g}_j||_2\leq 1, \qquad \forall j \text{ s.t. } ||\boldsymbol{\beta}_j||_2= 0 \tag{4.4}$$

The multi-task LASSO estimate in equation 2.5 must satisfy the KKT condition as below [14].

$$\frac{1}{n}(\boldsymbol{X}^j)^T(-\boldsymbol{Y}+\boldsymbol{X}\hat{\boldsymbol{B}})+\lambda\cdot\frac{\hat{\boldsymbol{\beta}}_j}{||\hat{\boldsymbol{\beta}}_j||_2}=0, \qquad \forall j \text{ s.t. } ||\hat{\boldsymbol{\beta}}_j||_2\neq 0 \tag{4.5}$$

$$||-\frac{1}{n\lambda}(\boldsymbol{X}^j)^T(-\boldsymbol{Y}+\boldsymbol{X}\hat{\boldsymbol{B}})||_2\leq 1, \qquad \forall j \text{ s.t. } ||\hat{\boldsymbol{\beta}}_j||_2= 0 \tag{4.6}$$

The condition for correct selection is $\hat{A}=A$, and it is equivalent to

$$\hat{\boldsymbol{\beta}}_{\hat{C}}=\boldsymbol{\beta}_C=0 \tag{4.7}$$

Where $\boldsymbol{\beta}_C$ is a matrix containing the coefficients of untrue variables. Similarly, we denote the matrix containing the coefficients of true variables as $\boldsymbol{\beta}_A$, the dimension of which is $q\cdot l$.

Thus, $\boldsymbol{XB} = \boldsymbol{X}_A\boldsymbol{\beta}_A$. Then we have

$$\boldsymbol{Y} = \boldsymbol{X}_A\boldsymbol{\beta}_A + \boldsymbol{E} \tag{4.8}$$

Add the correct selection condition 4.7 and equation 4.8 into the KKT condition, then we have

$$\frac{1}{n}(\boldsymbol{X}^j)^T(-\boldsymbol{X}_A\boldsymbol{\beta}_A - \boldsymbol{E} + \boldsymbol{X}_A\hat{\boldsymbol{\beta}}_A) + \lambda \cdot \frac{\hat{\boldsymbol{\beta}}_j}{||\hat{\boldsymbol{\beta}}_j||_2} = 0, \qquad \forall j \in A \tag{4.9}$$

$$||(\boldsymbol{X}^j)^T(-\boldsymbol{X}_A\boldsymbol{\beta}_A - \boldsymbol{E} + \boldsymbol{X}_A\hat{\boldsymbol{\beta}}_A)||_2 \leq n\lambda, \qquad \forall j \in C \tag{4.10}$$

Condition 4.9 can be rewritten as

$$\frac{1}{n}\boldsymbol{X}_A^T(-\boldsymbol{X}_A\boldsymbol{\beta}_A - \boldsymbol{E} + \boldsymbol{X}_A\hat{\boldsymbol{\beta}}_A) + \lambda D(\hat{\boldsymbol{\beta}}_A) = 0 \tag{4.11}$$

where $D(\hat{\boldsymbol{\beta}}_A)$ is the directions of each vector $\hat{\boldsymbol{\beta}}_j, \forall j \in A$.

Add direction correctness condition $D(\hat{\boldsymbol{\beta}}_A) = D(\boldsymbol{\beta}_A)$ to 4.11 to ensure the direction correctness. The condition becomes

$$\hat{\boldsymbol{\beta}}_A - \boldsymbol{\beta}_A = (\boldsymbol{X}_A^T\boldsymbol{X}_A)^{-1}(\boldsymbol{X}_A^T\boldsymbol{E} - n\lambda D(\boldsymbol{\beta}_A)) \tag{4.12}$$

Replace $\hat{\boldsymbol{\beta}}_A - \boldsymbol{\beta}_A$ in 4.10, we have

$$||(\boldsymbol{X}^j)^T[\boldsymbol{X}_A(\boldsymbol{X}_A^T\boldsymbol{X}_A)^{-1}(\boldsymbol{X}_A^T\boldsymbol{E} - n\lambda D(\boldsymbol{\beta}_A)) - \boldsymbol{E}]||_2 \leq n\lambda, \quad \forall j \in C \tag{4.13}$$

Define $\boldsymbol{V} = (\boldsymbol{X}_A^T\boldsymbol{X}_A)^{-1}(\boldsymbol{X}_A^T\boldsymbol{E} - n\lambda D(\boldsymbol{\beta}_A))$. Then 4.12 is simplified as

$$\hat{\boldsymbol{\beta}}_A = \boldsymbol{\beta}_A + \boldsymbol{V} \tag{4.14}$$

In order to ensure $D(\hat{\boldsymbol{\beta}}_j) = D(\boldsymbol{\beta}_j)$, we need to make sure $D(\boldsymbol{V}_j) = D(\boldsymbol{\beta}_j), \forall j \in A$. When $\boldsymbol{V}_j$ and $\boldsymbol{\beta}_j$ are parallel, i.e.

$$\exists k_j \text{ s.t. } \boldsymbol{V}_j = k_j\boldsymbol{\beta}_j, \forall j \in A \tag{4.15}$$

the angle θ_j between them could be 0 or π. They need to satisfy

$$||\boldsymbol{V}_j||_2 < ||\boldsymbol{\beta}_j||_2, \forall j \text{ s.t. } \theta_j = \pi \tag{4.16}$$

To summarize, the conditions for multi-task lasso achieving direction correctness are

$$||(\boldsymbol{X}^j)^T[\boldsymbol{X}_A(\boldsymbol{X}_A^T\boldsymbol{X}_A)^{-1}(\boldsymbol{X}_A^T\boldsymbol{E} - n\lambda D(\boldsymbol{\beta}_A)) - \boldsymbol{E}]||_2 \leq n\lambda, \quad \forall j \in C \tag{4.17}$$

$$\exists k_j \text{ s.t. } \boldsymbol{V}_j = k_j\boldsymbol{\beta}_j, \quad \forall j \in A \tag{4.18}$$

$$||\boldsymbol{V}_j||_2 < ||\boldsymbol{\beta}_j||_2, \quad \forall j \text{ s.t. } \theta_j = \pi \tag{4.19}$$

As each element in $\boldsymbol{E}$ is normal distributed, the probability of these conditions hold can be estimated using Monte Carlo Method. As in simulation, it is almost impossible to get exactly parallel vectors, the second condition is relaxed to

$$|\frac{\boldsymbol{V}_j \cdot \boldsymbol{\beta}_j}{||\boldsymbol{V}_j||_2 \cdot ||\boldsymbol{\beta}_j||_2}| > \cos\theta_0 \tag{4.20}$$

$\cos\theta_0$ is set as a simulation parameter.

4.4 Self-Voting Multi-Task LASSO

Similar as the self-voting LASSO, we can also use the self-voting procedure for multi-task LASSO. The procedure is:

1. Sample several reasonable initial values λ^0. Use multi-task LASSO to get estimation $\hat{\beta}^{\lambda^0}$.

2. Regress only on estimated true variables and get the estimation $\hat{\beta}_{\hat{A}^{\lambda^0}}$.

3. Take $\hat{\beta}_{\hat{A}^{\lambda^0}}$ as the true coefficient and estimate the PDC using Monte Carlo Method. Approximate an "optimal" λ^*.

4. If λ^* is equal to λ^0, terminate the procedure and use $\hat{A}^{\lambda^0}$ as the final result.

5. NUMERICAL EXPERIMENTS

5.1 Introduction

This chapter contains numerical simulations conducted to study and compare the performance of multi-task LASSO and single-task LASSO when using self-voting or cross-validation to select tuning parameter. To have a reasonable comparison, the frequency of achieving Sign Correctness (SC) is chosen as the performance measure.

5.2 Description of Experiments

We study the performance of multi-task LASSO and compare with cross-validation in 7 different scenarios. Number of observations, number of features, number of tasks, and number of true features are varied in different scenarios.

For each scenario, we generate an optimal supersaturated design based on RSS_2 criterion at first. The SSDs used in all the scenarios are shown in the Appendix A. Then run 100 replications in which different models and responses are generated. In every replication, the true variables are randomly chosen, and their coefficients $\boldsymbol{\beta}_A$ are generated according to a uniform distribution $UNIF(3, 7)$. The random noise $\boldsymbol{E}$ is normal distributed as $N(0, 1)$. After the response is calculated, multi-task self-voting LASSO is implemented to screen the features. As the Monte Carlo estimation for PDC is computationally expensive, parallel computing is used to save time.

Given self-voting has local convergence, multiple starting points λ_0 is used to approximate the optimal λ. The λ^* which achieves highest PDC is chosen as the final result. L-BFGS-B method is used in the optimization process of λ.

To compare the performance, multi-task LASSO and single-task LASSO with cross-validation are also implemented to select feature. For single-task LASSO, it is

implemented to each task separately, and then the result is combined. Note that only when the sign of every element in the matrix $\hat{\boldsymbol{\beta}}$ is correct, the Sign Correctness is achieved. The estimated probability of achieving Sign Correctness is compared in the experiment.

The simulation model is developed in R, and the experiment is conducted on a ThinkPad© X1 carbon laptop, with Intel(R) Core(TM) i5-5200U CPU@2.20 GHz, and 64-bit operating system.

5.3 Numerical Result

5.3.1 Scenario 1

For scenario 1, we use a 12×16 supersaturated design X_{1216} generated with RSS_2 criterion, where $RSS(k=2) = 140.86, ES^2 = 5.2, D-Optimal(k=2) = 0.98, A-Optimal(k=2) = 1.04$. There are 3 different tasks i.e. responses, and 2 true variables.

$$X_{1216} = \begin{bmatrix} - & + & - & - & + & - & - & + & + & + & + & + & - & + & + & + \\ - & + & - & - & - & + & - & + & - & + & - & - & + & - & - & - \\ + & - & + & + & + & + & - & + & + & + & - & + & + & - & + & + \\ + & + & - & + & + & - & - & - & - & - & + & + & - & - & - & - \\ - & + & + & + & + & + & + & - & + & - & + & - & + & + & + & - \\ - & - & - & - & + & + & + & - & - & - & - & + & + & + & - & + \\ + & + & - & + & - & - & + & - & + & + & - & - & + & + & - & + \\ - & + & + & + & - & - & + & + & - & - & - & + & - & - & + & + \\ + & - & + & + & - & + & - & + & - & - & + & - & - & + & - & + \\ + & - & + & - & + & - & - & - & - & + & - & - & - & + & + & - \\ + & - & - & - & - & - & + & + & + & - & + & - & + & - & + & - \\ - & - & + & - & - & + & + & - & + & + & + & + & - & - & - & - \end{bmatrix}.$$

The result in one replication looks like in table 5.1. In this case, true coefficients are randomly generated. Multi-task self-voting LASSO achieves Sign Correctness while multi-task CV and single-task CV do not.

The frequency of the 3 methods achieving Sign Correctness is shown in table 5.2. The ratio of Multi-task Self-Voting LASSO achieving Sign Correctness is 0.94, which is much higher than Multi-task LASSO tuned with CV and Single-task LASSO with CV. Multi-task LASSO and Single-task LASSO do not show much difference when both tuned with cross-validation.

5.3.2 Scenario 2, 3 and 4

For scenario 2, 3 and 4 we have more features ($p = 22$) while the number of observations stays the same ($n = 12$). In scenario 2, the estimated PSC of single-task LASSO drops as the model size increases, while the estimated PSC is almost the same for the other two methods.

In scenario 3, number of observations and features are the same as they are in scenario 2, but there are more tasks to consider ($l = 5$). Achieving Sign Correctness is supposed to be harder in this scenario. The result in a typical replication looks like table 5.3, where the coefficients of feature 3 to feature 12 are not shown, as they are all zero.

The estimated probabilities of three methods achieving Sign Correctness in all three scenarios are reported in table 5.2. In a supposedly harder scenario like scenario 3, the performance of multi-task self-voting LASSO is still significantly better than the other two methods. Furthermore, as there are more tasks to consider at the same time, the performance of single-task LASSO drops considerably.

In Scenario 4, the number of tasks is increased to $l = 7$, and thus raises difficulty of achieving Sign Correctness. Multi-task self-voting LASSO achieved 100% correct selection, while the performance of multi-task LASSO with CV does not vary much. However, the estimated probability of sign correct selection is only 0.08 for single-task LASSO in this scenario.

Table 5.1.
Example of Result of Scenario 1

Feature	True Coefficients			Multi-task SV result			Multi-task CV result			Single-task CV result		
1	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0.05	-0.04	-0.04	0	0	0
3	0	0	0	0	0	0	-0.01	0.06	0.00	0	0	0
4	6.59	5.62	-4.42	4.41	4.36	-3.13	5.92	5.86	-4.21	5.21	4.63	-3.85
5	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	-0.03	0.01	0.06	0	0	0
8	0	0	0	0	0	0	0.01	-0.02	0.07	0	0	0
9	-3.83	3.36	4.36	-2.46	2.21	2.53	-3.88	3.44	3.78	-3.14	2.21	3.51
10	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0.04	0.13	0.57	0	0	0.55
16	0	0	0	0	0	0	0	0	0	0	0	0

Table 5.2.
PSC of Different Scenarios

Scenario	n	p	l	q	Multi-task SV	Multi-task CV	Single-task CV
1	12	16	3	2	0.94	0.62	0.66
2	12	22	3	2	0.97	0.58	0.48
3	12	22	5	2	0.99	0.55	0.14
4	12	22	7	2	1	0.53	0.08
5	12	22	5	4	0.51	0.05	0
6	16	26	3	2	0.96	0.45	0.4
7	16	30	5	2	0.97	0.54	0.16

5.3.3 Scenario 5

In the fourth scenario, we keep the other settings the same as scenario 3, and only change the number of true variables from 2 to 4. Though the SSD is designed for 2 true variables, sometimes we are unable to know the exact number of true variables. Thus, we want to know how bad the algorithms perform under this circumstance. The result reported in table 5.2 shows that, in scenario which is not so ideal, multi-task self-voting LASSO still performs better than the other two methods.

5.3.4 Scenario 6 and 7

The model size is again enlarged in scenario 6, and the estimated PSC of multi-task self-voting LASSO is almost constant, while the estimated PSC of the other two methods drop accordingly.

5.3.5 Conclusion

In conclusion, the multi-task self-voting LASSO outperform LASSO working with cross-validation in terms of feature selection accuracy. It achieves PSC no less than 95% in most of the scenarios we consider. While the estimated PSC of other methods decrease when the scale of the model increase, the performance of multi-task self-voting LASSO is robust.

Table 5.3.
Example of Result of Scenario 2

Feature	True Coefficients					Multi-task SV result				
1	-5.21	5.71	4.11	6.60	5.79	-3.20	3.31	2.71	4.23	3.83
2	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0
17	5.75	-4.37	-4.52	5.33	3.75	3.47	-2.80	-2.75	3.10	2.22
18	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0	0
Feature	Multi-task CV result					Single-task CV result				
1	-4.42	4.57	3.75	5.84	5.29	-3.57	4.29	3.05	4.24	5.11
2	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	-0.38	0	0	0
15	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0
17	5.07	-4.09	-4.02	4.53	3.25	4.39	-3.82	-3.43	2.84	2.85
18	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0	0

6. SUMMARY AND FUTURE WORK

In our research, we studied the optimal selection of tuning parameter for single-task and multi-task LASSO. We first studied the feature selection performance measured by probability of achieving Sign Correctness of LASSO, working with LASSO-optimal supersaturated design. Then we extended the probability of achieving Sign Correctness to probability of achieving Direction Correctness, and derived the condition of achieving Direction Correctness for multi-task LASSO. Using this probability proved, we developed an iterative procedure to select tuning parameter λ to maximize the feature selection performance. The performance of multi-task self-voting LASSO is studied in numerical experiments, and different scenarios were taken into consideration. It outperforms multi-task or single-task LASSO using cross-validation to select tuning parameter.

Our work can be extended in future research in several directions. First, the run time and performance of multi-task self-voting LASSO is influenced by the starting value of tuning parameter λ. In our research, starting value λ_0 is randomly generated according to uniform distribution in an empirical range. In future research, we expect to develop a method, which can be proved theoretically, to select a starting value of tuning parameter that is close to the optimal value. In this way, the feature selection accuracy can be even higher, and the number of iterations can be smaller.

Furthermore, the probability of achieving Direction Correctness can be extended for other methods in LASSO family with more complex penalty form. For example, group LASSO [7] and exclusive LASSO [15] also have grouping or block selection effect, and can benefit from analysis of correct selection.

REFERENCES

REFERENCES

[1] Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3(Mar):1157–1182, 2003.

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

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

[4] Robert Tibshirani, Michael Saunders, Saharon Rosset, Ji Zhu, and Keith Knight. Sparsity and smoothness via the fused lasso. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(1):91–108, 2005.

[5] 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.

[6] Hui Zou. The adaptive lasso and its oracle properties. *Journal of the American Statistical Association*, 101(476):1418–1429, 2006.

[7] 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.

[8] Lisa-Ann Kirkland, Frans Kanfer, and Sollie Millard. Lasso tuning parameter selection. In *Annual Proceedings of the South African Statistical Association Conference*, volume 2015, pages 49–56. South African Statistical Association (SASA), 2015.

[9] Dadi Xing. *Lasso-optimal supersaturated design and analysis for factor screening in simulation experiments*. PhD thesis, Purdue University, 2015.

[10] Kathleen HV Booth and David R Cox. Some systematic supersaturated designs. *Technometrics*, 4(4):489–495, 1962.

[11] Guillaume Obozinski, Ben Taskar, and Michael Jordan. Multi-task feature selection. *Statistics Department, UC Berkeley, Tech. Rep*, 2, 2006.

[12] Hong Wan, Bruce E Ankenman, and Barry L Nelson. Controlled sequential bifurcation: A new factor-screening method for discrete-event simulation. *Operations Research*, 54(4):743–755, 2006.

[13] Hong Wan, Bruce E Ankenman, and Barry L Nelson. Improving the efficiency and efficacy of controlled sequential bifurcation for simulation factor screening. *INFORMS Journal on Computing*, 22(3):482–492, 2010.

[14] Saharon Rosset and Ji Zhu. Piecewise linear regularized solution paths. *The Annals of Statistics*, pages 1012–1030, 2007.

[15] Yang Zhou, Rong Jin, and Steven Chu-Hong Hoi. Exclusive lasso for multi-task feature selection. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 988–995, 2010.

APPENDICES

APPENDIX A
DESIGN

The designs used in the experiments are all generated using partial gradient algorithm for optimal SSD proposed by Dadi [9]. All the designs are generated with RSS_2 criterion.

Table A.1.
Design: Optimal 12×16 SSD based on RSS_2

-	+	-	-	+	-	-	+	+	+	+	+	-	+	+	+
-	+	-	-	-	+	-	+	-	+	-	-	+	-	-	-
+	-	+	+	+	+	-	+	+	+	-	+	+	-	+	+
+	+	-	+	+	-	-	-	-	-	+	+	-	-	-	-
-	+	+	+	+	+	+	-	+	-	+	-	+	+	+	-
-	-	-	-	+	+	+	-	-	-	-	+	+	+	-	+
+	+	-	+	-	-	+	-	+	+	-	-	+	+	-	+
-	+	+	+	-	-	+	+	-	-	-	+	-	-	+	+
+	-	+	+	-	+	-	+	-	-	+	-	-	+	-	+
+	-	+	-	+	-	-	-	-	+	-	-	-	+	+	-
+	-	-	-	-	-	+	+	+	-	+	-	+	-	+	-
-	-	+	-	-	+	+	-	+	+	+	+	-	-	-	-

Table A.2.
Design: Optimal 12×22 SSD based on RSS_2

+	+	+	-	-	-	+	+	-	-	+	-	+	-	+	+	+	+	+	-	+	+
-	-	-	+	-	+	-	+	-	-	+	+	+	+	-	-	-	+	+	-	-	+
-	-	+	-	-	-	+	-	+	+	+	-	-	+	-	+	-	+	-	-	-	-
-	-	-	-	+	+	+	-	-	+	-	-	-	+	+	+	+	-	-	+	-	+
+	+	-	-	+	-	-	-	-	-	+	+	-	-	+	-	-	-	-	-	-	-
-	+	+	+	+	-	+	-	+	+	+	+	+	-	-	-	+	-	+	+	-	+
-	+	+	+	+	+	-	+	-	+	-	-	+	+	+	-	+	+	-	-	+	-
+	-	-	+	-	-	+	+	-	+	-	-	+	-	-	-	-	-	-	+	+	-
+	+	+	-	-	+	-	+	+	-	+	+	-	+	-	-	+	-	-	+	+	+
+	+	-	+	+	+	-	-	+	+	-	-	-	-	-	+	-	+	+	-	+	+
+	-	-	+	+	-	+	+	+	-	-	+	-	+	+	+	+	+	+	+	-	-
-	-	+	-	-	+	-	-	+	-	-	+	+	-	+	+	-	-	+	+	+	-

Table A.3.
Design: Optimal 16×26 SSD based on RSS_2

-	-	+	+	+	+	-	-	-	+	+	+	+	+	-	-	+	-	+	-	+	+	+	+	-	+
+	-	-	+	+	-	+	-	+	-	+	-	-	+	+	-	-	-	-	-	+	-	-	+	+	-
+	-	+	+	+	-	-	+	-	+	-	-	+	-	-	-	+	+	-	-	+	-	+	-	+	-
-	-	+	-	-	-	+	+	+	-	+	+	+	-	-	+	-	-	-	-	+	+	-	-	+	+
-	+	-	+	+	+	+	+	-	-	+	+	-	-	+	+	+	-	-	-	-	-	+	-	-	-
-	+	-	+	-	-	-	-	+	+	-	+	+	-	+	+	+	+	-	+	+	-	-	+	+	+
-	-	-	-	+	-	-	+	-	+	+	+	-	+	+	-	-	+	+	+	-	-	-	-	+	+
-	-	-	-	-	-	-	-	+	-	-	+	-	+	-	-	+	-	-	+	-	+	+	-	-	-
+	+	-	+	-	+	-	+	-	-	-	-	+	+	-	+	-	-	+	+	+	-	-	-	-	+
+	+	-	-	+	-	+	+	+	+	+	-	+	+	-	+	+	+	+	-	-	+	-	+	-	-
+	-	+	+	-	+	-	+	+	-	-	-	-	+	+	+	+	+	+	-	-	+	+	+	+	+
-	+	+	-	-	+	+	-	+	+	-	+	+	+	+	-	-	+	+	-	-	-	+	-	-	-
-	+	+	-	+	-	+	-	-	-	-	-	-	-	-	+	-	+	+	+	+	-	+	+	+	-
+	+	+	+	-	+	+	-	+	+	+	-	-	-	-	-	+	-	+	+	-	+	-	-	+	+
+	+	+	-	-	+	-	+	-	-	+	+	-	-	+	-	-	+	-	+	+	+	-	+	-	-
+	-	-	-	+	+	+	-	-	+	-	-	+	-	+	+	-	-	-	+	-	+	+	+	-	+

Table A.4.
Design: Optimal 16×30 SSD based on RSS_2

+	+	-	-	+	+	-	-	+	-	+	-	-	+	+	+	+	+	+	-	+	+	-	-	-	-	+	+	+	-
+	+	-	+	+	-	-	-	+	+	-	-	-	-	-	+	+	+	-	+	-	-	+	+	-	+	+	+	-	+
-	+	-	-	-	+	+	+	-	+	+	+	+	+	-	+	-	-	-	+	-	+	-	+	-	+	+	+	+	-
-	+	+	+	-	+	+	-	+	-	-	-	+	-	-	-	+	+	-	-	-	+	-	-	+	+	-	-	+	-
+	-	+	-	-	+	-	-	-	+	+	+	+	-	-	+	-	+	+	+	+	+	+	-	+	+	-	+	-	+
+	-	-	+	+	+	+	+	+	-	+	+	+	+	+	-	+	+	-	+	-	-	-	+	+	-	-	+	-	+
-	+	+	+	-	-	-	+	+	-	+	+	-	+	+	-	-	+	+	-	+	+	+	+	-	+	+	-	-	+
-	+	+	-	+	-	-	-	-	-	-	-	-	+	-	-	-	-	-	+	+	+	-	+	+	-	-	+	-	+
-	-	-	+	+	+	+	+	-	+	-	-	-	+	-	+	+	-	+	-	+	-	-	-	+	+	+	-	-	+
+	-	+	+	+	-	+	+	-	-	+	+	-	-	-	-	+	-	+	+	-	+	+	-	+	-	+	-	+	-
-	-	-	-	-	-	-	-	-	+	+	+	-	-	+	-	+	+	-	-	-	-	-	-	-	-	-	-	-	-
+	+	+	-	-	-	+	+	-	+	-	-	-	+	+	+	-	+	+	-	-	-	+	+	+	-	-	+	+	-
+	-	+	+	-	-	-	-	+	+	-	+	+	+	+	+	-	-	-	+	+	-	-	-	+	+	+	-	+	-
+	+	-	-	-	+	+	+	-	-	-	-	+	-	+	-	-	-	-	+	+	-	+	-	-	-	+	-	-	+
-	-	+	-	+	+	-	+	+	+	-	+	+	-	+	+	+	-	+	-	-	+	+	+	-	-	-	-	+	+
-	-	-	+	+	-	+	-	+	-	+	-	+	-	-	-	-	-	+	-	+	-	+	+	-	+	-	+	+	-