# Outlier Detection for Mixed Model with Application to RNA-Seq Data

by

Tzu-Ying Liu

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Biostatistics)
in The University of Michigan
2018

Tzu-Ying Liu

ltzuying@umich.edu

ORCID iD: 0000-0001-7533-4134

## ACKNOWLEDGEMENTS

I would like to thank my program advisor, Professor Hui Jiang and my committee members, Professor Srijan Sen of the Department of Psychiatry and Molecular and Behavioral Neuroscience Institute, Professor Ji Zhu of the Department of Statistics, Professor Peter Song and Professor Jack Kalbfleisch of Department of Biostatistics, for their exceedingly insightful comments that improved my research.

I also would like to express my gratitude to the two anonymous reviewers and the associate editor of The Journal of Computational and Graphical Statistics for their suggestions on my first project, especially for the image restoration application and the extension to high-dimension settings. I am very grateful to our collaborators, Dr. Marcin Cieslik, and Professor Arul Chinnaiyan; they provided me with helpful suggestions and encouragement. Finally, I would like to thank the funding investigator, Professor Alex Tsodikov. The research was supported by NIH Prostate SPORE grant 5P50CA186786.

# TABLE OF CONTENTS

**ACKNOWLEDGEMENTS** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . **i**

**LIST OF FIGURES** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . **iv**

**LIST OF TABLES** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . **vii**

**ABSTRACT** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . **viii**

**CHAPTER**

**I. Introduction** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1

**II. Minimizing Sum of Truncated Convex Functions** . . . . . . . . . . . . . . . . 4

2.1 Introduction . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 4
2.2 Applications . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 6
2.2.1 Outlier detection in linear models . . . . . . . . . . . . . . . . . . 6
2.2.2 Convex shape placement . . . . . . . . . . . . . . . . . . . . . . 9
2.2.3 Signal and image restoration . . . . . . . . . . . . . . . . . . . . 10
2.3 Methods . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 11
2.3.1 Notations . . . . . . . . . . . . . . . . . . . . . . . . . . . . 12
2.3.2 The general algorithm . . . . . . . . . . . . . . . . . . . . . . 13
2.3.3 Implementation in low-dimensional settings . . . . . . . . . . . . 14
2.3.4 Extension to high-dimensional settings . . . . . . . . . . . . . . . 16
2.3.5 Time complexity analysis . . . . . . . . . . . . . . . . . . . . . 17
2.4 Experiments . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 19
2.4.1 Outlier detection in simple linear regression . . . . . . . . . . . . 19
2.4.2 Sum of truncated quadratic functions . . . . . . . . . . . . . . . . 20
2.4.3 Convex shape placement . . . . . . . . . . . . . . . . . . . . . . 23
2.4.4 Signal and image restoration . . . . . . . . . . . . . . . . . . . . 23
2.5 Discussion . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 27

**III. Integrating Poly(A) Capture and Exome Capture RNA-Seq Data** . . . . 29

3.1 Introduction . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 29
3.2 The Data . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 33
3.3 Evidence of differences between the two types of measurements . . . . . . . . 35
3.4 Converting capture sequencing measurements to Poly(A) measurements . . . 37
3.4.1 Notation . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 39
3.4.2 Comparing prediction by genewise simple regression and mixed effect model . . . . . . . . . . . . . . . . . . . . . . . . . . . 39
3.5 Results . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 40
3.6 Discussion . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 43

3.7 Appendix for Chapter 3 . . . . . . . . . . . . . . . . . . . . . . . . . . 45

**IV. Outlier Detection for Mixed Model** . . . . . . . . . . . . . . . . . . . . 47

4.1 Extending mixed effect model for detecting individual outliers and outlying random effects . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 47
4.2 Estimation Scheme . . . . . . . . . . . . . . . . . . . . . . . . . . . . 49
4.3 Estimate the outliers and predict the random effects by Penalized Maximum Likelihood Estimation . . . . . . . . . . . . . . . . . . . . . . . . . . 51
4.3.1 Separation of the objective function by incorporating the predictors of the random effects . . . . . . . . . . . . . . . . . . . . . . . 53
4.3.2 Transform the objective function into a sum of truncated quadratic functions . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 55
4.3.3 Summary of estimating $\Delta$ and $\delta$ . . . . . . . . . . . . . . . . . . 59
4.4 Determine $\lambda_O$ and $\lambda_G$ . . . . . . . . . . . . . . . . . . . . . . . . 60
4.4.1 Variance of the conditional residuals and predicted random effects 61
4.4.2 Determine the cutoffs for conditional residuals and predicted random effects . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 62
4.5 Dilation factor . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 65
4.6 Simulation Study . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 67
4.7 Application to RNA-Seq data . . . . . . . . . . . . . . . . . . . . . . 70
4.8 Discussion . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 73

**V. Discussion** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 75

**APPENDIX** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . **77**
A1 Appendix for Chapter II . . . . . . . . . . . . . . . . . . . . . . . . . 77
A1.1 Application on detecting differential gene expression with $\ell_0$-penalized models . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 77
A1.2 Algorithms described in Section 2.3 . . . . . . . . . . . . . . . . . . 79
A1.3 The $\Theta$-IPOD algorithm for robust linear regression . . . . . . . . . 81
A1.4 The difference of convex (DC) functions algorithm . . . . . . . . . 81
A1.5 The iterative marginal optimization (IMO) algorithm for signal and image restoration . . . . . . . . . . . . . . . . . . . . . . . . . . . . 82
A1.6 Proofs . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 83
A1.7 Supplementary figures and tables . . . . . . . . . . . . . . . . . . . 87

**BIBLIOGRAPHY** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . **90**

# LIST OF FIGURES

**Figure**

2.1 The sum of two truncated quadratic functions $f_1 + f_2$ (in black), where $f_1(x) = \min\{4x^2 + 1, 3\}$ (in blue) and $f_2(x) = \min\{2(x-1)^2 + 2, 4\}$ (in red). . . . . . . . 6

2.2 The corresponding $C_i$'s of three convex functions $f_1, f_2, f_3$ define on $R^2$, where $C_i = \{\mathbf{x} : f_i(\mathbf{x}) \leq 0\}$. The boundaries of $\{C_i\}_{i=1}^3$ partition $R^2$ into eight disjoint pieces $\{A_j\}_{j=1}^8$. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 13

2.3 Comparison of different methods for outlier detection in simple linear regression. The figures show the mean percents of masking (top) and swamping (bottom) for different leverages of outliers: $L = 0$ (left) and $L = 20$ (right) and different percents of outliers (O%) for all the methods using 100 simulated replicates. The standard errors of the means are shown as error bars. When estimating the regression coefficients and identifying the outliers, we assume that the $\sigma^2$ is known and is 1. In other words, we estimate only $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ in (2.2). When implementing MM, LTS and GY estimators, the fact that $\sigma^2$ is known is not exploited because we use packaged functions for these methods, which assume $\sigma^2$ to be unknown and have no arguments for specifying $\sigma^2$. . . . . . . . . . . . . . . . . . . . . . . . . . . . 21

2.4 Contour plots of randomly generated sum of truncated quadratic functions in $R^2$. Global minima are marked with the plus sign. . . . . . . . . . . . . . . . . . . . . 23

2.5 Comparison of different algorithms for minimizing the sum of 50 randomly generated truncated quardratic funstions in 2-D. The figure shows the mean success rates (in percents) for all the methods using 100 simulated replicates for different complexities of the functions ($C$). The standard errors of the means are shown as error bars. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 24

2.6 Simulated random signal (left) and restored signal (right) are shown in solid lines. The underlying true signal are shown in dashed lines. . . . . . . . . . . . . . . . . 25

2.7 Restoration of synthetic and real images. For each row, from left to right: original image, image with Gaussian noise added, image restored using Gaussian smoothing with a $5 \times 5$ kernel and image restored using proposed algorithm. . . . . . . . . 27

3.1 mRNA isolation, adopted from [21]. In the process of RNA-seq library preparation, RNAs are first isolated from tissue samples using commercial kits. Then mRNAs are isolated from total RNAs by annealing to oligo-dT beads. rRNAs and tRNAs are washed away before mRNAs are released from the beads. . . . . . . . . . . . 31

3.2 RNA-seq workflow, adopted from [27]. The purified mRNAs are first fragmented into smaller pieces and reverse-transcribed to complementary DNAs (cDNAs). After formation of one strand of cDNA, the mRNA strand is removed and replaced by another strand of cDNA to generate a double-stranded cDNAs. Each end of the double-stranded DNA is then repaired, adenylated and ligated by adaptor before being enriched by polymerase chain reaction (PCR). Once a library has passed the quality control, it can be sent to various sequencing platforms and generate read counts data. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 32

3.3 Distribution of the RINs of 363 poly(A) libraries. Of the 363 CaptureSeq libraries, 357 had exactly the same RINs as their poly(A) counterparts. The other 6 CaptureSeq libraries had a difference from the poly(A) measurements as -2.7, -1.5, -1.1, -0.6, 0.1, and 0.7. The RINs range from 2.5 from 10 with a mean of 8.6 and a median of 9.2. The 1st and the 3rd quartiles are 8.1 and 9.6. . . . . . . . . . . . 34
3.4 Cluster analysis on the paired CaptureSeq and Poly(A) log2(CPM) measurements from 25 prostate cancer patients. Each row represent a gene and each column represent either a library of CaptureSeq ( columns with purple tags on the top) or Poly(A) (columns with orange tags on the top) measurements. The genes are the top 50 most varied genes across the 50 libraries. We used the Euclidean distance and the complete method of Hierarchical clustering on the genes. . . . . . . . . . 36
3.5 Box plots of the number of significant genes in each experiment comparing two groups of randomly drawn subjects . . . . . . . . . . . . . . . . . . . . . . . . 38
3.6 Prediction by the mixed model and genewise fixed effect models on libraries with RIN $\geq 7$: correlation of predicted and actual poly(A) log2-CPMs across all genes and all samples, based on 30 replicates for each sample size. . . . . . . . . . . . . 41
3.7 Prediction by the mixed model and genewise fixed effect models on libraries with RIN $\geq 7$: RMSE of predicted and actual poly(A) log2-CPMs across all genes and all samples, based on 30 replicates for each sample size. . . . . . . . . . . . . . . . 41
3.8 Cluster analysis on the paired predicted Poly(A) measurements based on capture sequencing measurements and true Poly(A) log2(CPM) measurements from 25 prostate cancer patients. Each row represents a gene, and each column represent either a predicted Poly(A) library ( columns with purple tags on the top) or true Poly(A) (columns with orange tags on the top) measurements. The genes are the top 50 most varied genes across the 50 libraries. We used the Euclidean distance and the complete method of Hierarchical clustering on the genes. The clustering is now by subject, which is suggested by the neighboring of same subject identification numbers. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 42
3.9 Distribution of the gene-wise ordinary least squares (OLS) estimates by having the paired the Exome Capture RNA-Seq measurements regressed on the Poly(A) Capture RNA-Seq measurements. The arrows indicate identified causes for some of the gene-specific random effects to be outliers of the assumed bivariate normal distribution. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 44
3.10 Examples of genes whose OLS estimates lie in the right lower quadrant of in Figure 3.9. The smaller clusters of observations shared the same sample identifiers across multiple genes, which suggests the existence of batch effects. . . . . . . . . 44
3.11 The distribution of and the correlations between the total length of exons in a gene, the proportion of the bases in the exons that are either "G" or "C", and the proportion of the length that is covered by the first generation probes of CaptureSeq designed by our collaborative biotech company, the intercepts, the slopes and the residual standard error of the genewise simple regressions by regressing Poly(A) log2(CPM) measurements on CaptureSeq log2(CPM) measurements. . . . . . . 45
3.12 Distributions of the Exome Capture RNA-Seq and the Poly(A) Capture RNA-Seq measurements. The first row shows the distribution of mean log2 count per million (CPM) for each gene. The second row displays the inverse relationship between the variance of log2CPM and the mean raw count. The third row illustrates the libraries sizes for all samples, which are measured by both Exome Capture and Poly(A) Capture methods. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 46
4.1 Estimation scheme of the proposed method. . . . . . . . . . . . . . . . . . . . . . 51

4.2 Illustration of outlier detection for a mixed effect by the proposed method in real RNA-Seq data. There are paired Poly(A) and Capture RNA-Seq measurements for 18,000 genes from 100 subjects. We use a mixed model to the predict Poly(A) measurements based on a single predictor, the Capture RNA-Seq measurements with gene-specific random intercepts and slopes. Because the outlying gene-specific effects will be estimated as zeros in the proposed method, we use estimated coefficients from the simple regression in to demonstrate the relative positions of the detected outlying random effects (red) and normal random effects (black). . . . . 70

4.3 The scatter plot of Capture and Poly(A) RNA-Seq measurements from one of the undetected outlying genes whose OLS estimates lies in the right upper quadrant in Figure 4.2. The gene-specific OLS estimates the standard error to be 1.1 while the common standard error assumed by the proposed method across all genes is 0.46. There are also unequal variances among observations within a gene. . . . . . . . . 71

4.4 The scatter plot of Capture and Poly(A) RNA-Seq measurements from one of the undetected outlying genes whose OLS estimates lies in the left upper quadrant in Figure 4.2. The gene-specific OLS estimates the standard error to be 0.72 while the common standard error assumed by the proposed method across all genes is 0.46. There are also unequal variances among observations within a gene. . . . . . 72

4.5 The scatter plot of Capture and Poly(A) RNA-Seq measurements from one of the detected genes whose OLS estimates are not outlying. The gene-specific OLS estimates the standard error to be 1.9 while the common standard error assumed by the proposed method across all genes is 0.46. There are also unequal variances among observations within a gene. . . . . . . . . . . . . . . . . . . . . . . . . . . 72

A1 Placement of different convex shapes to cover the maximum number of points uniformly sampled from the unit square. . . . . . . . . . . . . . . . . . . . . . . . . . 87

A2 Restoration of images. For each row, from left to right: original image, image with Gaussian noise added, image restored using Gaussian smoothing with a $5 \times 5$ kernel and image restored using proposed algorithm. . . . . . . . . . . . . . . . . . . . . 87

# LIST OF TABLES

<u>Table</u>

2.1 Comparison of different algorithms for signal restoration. The table shows the mean success rates (in percents), relative losses, root mean square errors (RMSE), as well as running times (in seconds) for all the methods using 100 simulated replicates. The standard errors of the means are given in parentheses. . . . . . . . . . . . . 26

3.1 Origins of cancer tissues of the 372 patients. . . . . . . . . . . . . . . . . . . . . 33

3.2 **Nonzero coefficients of the penalized regression for classifying samples from the two technologies.** We tuned the Elastic Net regression model using a sequence of $\alpha$ (from 0 to 1, with a step of 0.1), which determined the proportion of quadratic and L1 norm in the penalty term, and 10-fold cross-validation to choose $\lambda$. We trained the model on 272 pairs of measurements and then tested the model on the other 100 pairs. The misclassification rate on the test set is 0.005. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 37

3.3 **Differential expression experiments of randomly drawn subjects.** Number of genes found to be significantly differentially expressed for more than 50 times in a total of 1000 experiments. . . . . . . . . . . . . . . . . . . . . . . . . . . . 37

4.1 Summary of the estimates with corresponding outlyingness . . . . . . . . . . . . 60

4.2 Outlier detection for small data sets composed of 20 genes, each with10 paired measurements. The results are based on the medians of the 100 replicates. For the proposed method, the ideal masking (false negative) rate is 0 and the swamping (false positive) rate should be close to the user-defined threshold $\alpha$, which is 5% in the simulation setting. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 68

4.3 Outlier detection for data sets composed of 2000 genes, each with10 paired measurements. The results are based on the medians of the 100 replicates. For the proposed method, the ideal masking (false negative) rate is 0 and the swamping (false positive) rate should be close to the user-defined threshold $\alpha$, which is 5% in the simulation setting. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 69

4.4 Parameter estimation by the naive method and the proposed method based on data sets composed of 20 genes, each with 10 paired measurements. The results are based on the medians of the 100 replicates. . . . . . . . . . . . . . . . . . . . . 69

4.5 Parameter estimation by the naive method and the proposed method based data sets composed of 2000 genes, each with 10 paired measurements. The results are based on the medians of the 100 replicates. . . . . . . . . . . . . . . . . . . . . . . 69

A1 Comparison of different methods for outlier detection in simple linear regression. The table shows the leverages of outliers (L), percents of outliers (O%) and mean percents of masking and swamping for all the methods using 100 simulated replicates. The standard errors of the means are given in parentheses. . . . . . . . . 88

A2 Stopping criteria of the simulation studies in Sections 2.4.2 and 2.4.4 . . . . . . . . 88

A3 Comparison of different algorithms for global optimization of the sum of 50 randomly generated truncated quadratic functions in 2-D. The table shows the complexities of the functions ($C$) as well as mean success rates (in percents) and running times (in seconds) for all the methods using 100 simulated replicates. The standard errors of the means are given in parentheses. . . . . . . . . . . . . . . . . . . . . 89

# ABSTRACT

Extracting messenger RNA (mRNA) molecules using oligo-dT probes targeting on the Poly(A) tail is common in RNA-sequencing (RNA-seq) experiments. This approach, however, is limited when the specimen is profoundly degraded or formalin-fixed such that either the majority of mRNAs have lost their Poly(A) tails or the oligo-dT probes do not anneal with the formalin-altered adenines. For this problem, a new protocol called capture RNA sequencing was developed using probes for target sequences, which gives unbiased estimates of RNA abundance even when the specimens are degraded. However, despite the effectiveness of capture sequencing, mRNA purification by the traditional Poly(A) protocol still underlies most reference libraries. A bridging mechanism that makes the two types of measurements comparable is needed for data integration and efficient use of information.

In the first project, we developed an optimization algorithm that was later applied to outlier detection in a linear mixed model for data integration. In particular, we minimized the sum of truncated convex functions, which is often encountered in models with $L_0$ penalty. The solution is exact in one-dimensional and two-dimensional spaces. For higher-dimensional problems, we applied the algorithm in a coordinate descent fashion. Although the global optimality is compromised, this approach generates local solutions with much higher efficiency.

In the second project, we investigated the differences between Poly(A) libraries and capture sequencing libraries. We showed that without conversion, directly merging the two types of measurements lead to biases in subsequent analyses. A practical solution was to use a linear mixed model to predict one type of measurements based on the other. The predicted values based on this approach have high correlations, low errors and high efficiency compared with those based on the fixed model. Moreover, the procedure eliminates false positive findings and biases introduced by the technology differences between the two measurements.

In the third project, we noted outlying observations and outlying random effects when fitting the mixed model. As they lead to the discovery of dysfunctional probes and batch effects, we developed an algorithm that screened for the outliers and provided a robust estimation. Specifically, we modified the mean-shift model with variable selection using $L_0$ penalties, which was first introduced by Gannaz (2007), McCann and Welsch (2007) and She and Owen (2012). By incorporating the optimization method proposed in the first project, the algorithm became scalable and yielded exact solutions for low-dimensional problems. In particular, under the assumption of normality, there existed analytic expressions for the penalty parameters. In simulation studies, we showed that the proposed algorithm attained reliable outlier detection, delivered robust estimation and achieved efficient computation.

CHAPTER I

# Introduction

The first project was motivated by a problem of minimizing a sum of truncated quadratic functions, which is often encountered in regression models with $L_0$ penalties. In the process solving the problem, we found that the algorithm can be applied to a more general problem of minimizing the sum of truncated convex functions. The merit of this general algorithm is that the solution is exact. However, the drawback is that the algorithm is hard to implement for problems with higher dimensions and when the functions are not quadratic. In this project, we demonstrate the accuracy of the algorithm in solving low dimensional sum of truncated quadratic functions. For higher dimensional problems, we resort to coordinate descent algorithm and iteratively minimize one-dimensional problems one at a time. Despite the loss of global optimality, we found this approach very efficient comparing to other global optimization algorithms.

The second project was motivated by a collaborative project where we need to integrate two types of RNA-seq measurements. The first type of measurements, which is called Poly(A) capture, are based on mRNAs purified by oligo-dT probes targeting on the Poly(A) tails. This purification methods, however, is limited when the specimen is profoundly degraded or formalin-fixed. A newer type of measurements, called capture RNA sequencing, is developed to overcome these difficulties. Despite the effectiveness of this newer protocol, most reference libraries in major databases are still based on the Poly(A) RNA-Seq measurements, and it is costly to reproduce the reference libraries for Capture RNA-Seq measurements. In this project, we examine whether there are differences between these two types of measurements. In particular, we demonstrated that there are potential biases and false positive findings in the analyses if we combine the two types of data without conversion. To bridge these two types of measurement, we needed to build a model that converts measurements for more than 18,000 genes with a minimal number of samples. It turned out that heeding the hierarchical structure in the data by using the mixed model rewards excellent efficiency in parameter estimation and high accuracy in prediction of gene-specific coefficients.

The third project evolved as we found outliers of the mixed model in the second project. There are two levels of outliers: outlying gene-specific effects that do not conform to the assumed multivariate normal distribution and outlying observations with residuals that do not fit the univariate normal distribution. Further investigation revealed that these outlying effects were caused by various technical issues, such as dysfunctional probes, batch effects or genes associated with histone mRNA and mitochondrial mRNA. As identifying the underlying technical issue help improve the technology and accuracy of the measurements, we set off to develop a robust mixed model that detect these outliers.

We applied the idea of the mean-shift variable by [17] and [33] to the mixed model, creating a mean-shift variable for each gene-specific effect and also a mean-shift variable for each observation. As the model is over-parameterized, we adopt the sparse estimation using $L_0$ penalty proposed by [45]. The primary challenge is that

the objective function is not convex and hard to optimize. Under the assumption of independent observations, we were able to transform the objective function into a sum of truncated function and apply the optimization algorithm that we proposed in the first project. Also, the objective function is separable by groups, and thus the optimization can run in parallel. We compare the performance of the proposed method with a naive approach of combining robust regression and robust estimation for multivariate normal distribution in simulation studies. The results showed the proposed method effectively detected the outliers and delivered estimates close to the real values.

# CHAPTER II

# Minimizing Sum of Truncated Convex Functions

In this chapter, we study a class of problems where the sum of truncated convex functions is minimized. In statistical applications, they are commonly encountered when $\ell_0$-penalized models are fitted and usually lead to NP-Hard non-convex optimization problems. We propose a general algorithm for the global minimizer in low-dimensional settings. We also extend the algorithm to high-dimensional problems, where an approximate solution can be found efficiently. We compare our proposed algorithm with other existing algorithms in simulation studies and show its utility in edge-preserving image restoration on real data. This chapter is based on the following publication: "Minimizing sum of truncated convex functions and its applications", published in the Journal of Computational and Graphical Statistics.

## 2.1 Introduction

Regularization methods in statistical modeling have gain popularity in many fields, including variable selection, outlier detection, and signal processing. Recent studies [46, 45] have shown that models with non-convex penalties possess superior performance compared with those with convex penalties. While the latter in general can be obtained with ease by virtue of many well-developed methods for convex optimization [5], there are limited options in terms of global solutions for non-convex optimization, which are more and more commonly encountered in modern statistics and engineering. Current approaches often rely on convex relaxation [6], local solutions by iterative algorithms [16] or trading time for global optimality with stochastic search [53].

In this paper, we study a special class of non-convex optimization problems, for which the objective function can be written as a sum of truncated convex functions. That is,

$$\mathbf{x} = \arg\min_{\mathbf{x}} \sum_{i=1}^{n} \min\{f_i(\mathbf{x}), \lambda_i\}, \tag{2.1}$$

where $f_i : R^d \rightarrow R, i = 1, \ldots, n$, are convex functions and the truncated levels $\lambda_i \in R, i = 1, \ldots, n$, are constants. Due to the truncation of $f_i(\cdot)$ at $\lambda_i$, the objective function is often non-convex. See Figure 2.1 for an example.

While in general, such problems are NP-Hard (see Section 2.3 for formal results), we show that for some $f_i(\cdot)$ there is a polynomial-time algorithm for the global minimizer in low-dimensional settings. The idea is simple: When the objective function is piecewise convex (e.g., see Figure 2.1), we can partition the domain so that the objective function becomes convex when restricted to each piece. This way, we can find the global minimizer by enumerating all the pieces, minimizing the objective function on each piece, and taking the minimum among all local minima.

The rest of the paper is organized as follows. In Section 2.2, we demonstrate the utility of our algorithm in several applications where the objective function can be transformed into a sum of truncated convex functions. In Section 2.3, we lay out the general algorithm for the global solution and its implementation in low-dimensional settings. As we will see in the complexity analysis, the running time grows exponentially with the number of dimensions. We, therefore, make a compromised but

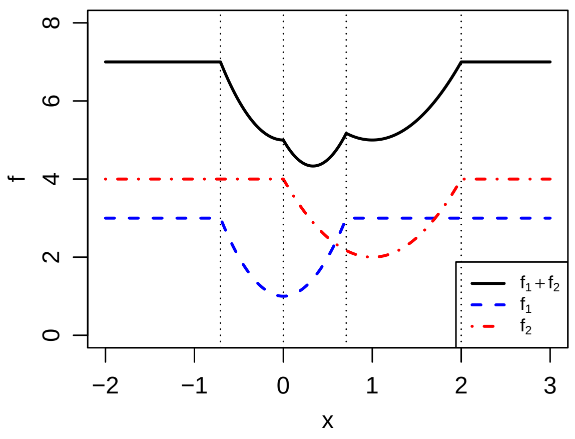

Figure 2.1: The sum of two truncated quadratic functions $f_1 + f_2$ (in black), where $f_1(x) = \min\{4x^2+1, 3\}$ (in blue) and $f_2(x) = \min\{2(x-1)^2+2, 4\}$ (in red).

efficient extension of the algorithm in high-dimensional settings. In Section 2.4, we compare our proposed algorithm with existing methods in simulation studies and apply our proposed algorithm to real-life image restoration problems. Discussions are given in Section 2.5.

## 2.2 Applications

### 2.2.1 Outlier detection in linear models

The task of outlier detection in linear regression can be formulated as a problem of variable selection. As in [17] and [33], given $n$ observations and $p$ covariates, we can add $n$ additional parameters $\{\gamma_i\}_{i=1}^n$ denoting the amount by which the observations are outlying. That is,

$$y_i = \mathbf{x}_i^T \boldsymbol{\beta} + \gamma_i + \epsilon_i, \qquad i = 1, \ldots, n, \tag{2.2}$$

where $y_i \in R, \mathbf{x}_i \in R^p, i = 1, \ldots, n$, are the observations, $\boldsymbol{\beta} \in R^p, \gamma_i \in R, i = 1, \ldots, n$, are the parameters of interest, and $\{\epsilon_i\}_{i=1}^n$ are i.i.d. $N(0, \sigma^2)$. Since there are $n+p$ parameters but only $n$ observations, the model is non-identifiable. [17] used an $\ell_1$ penalty in the objective function to force sparsity in $\gamma$ such that $y_i$ is considered an outlier if $\gamma_i \neq 0$ and an observation conforming to the assumed distribution if $\gamma_i = 0$. [33] treated (2.2) as a variable selection problem and applied the Least Angle Regression. Similar idea for outlier detection has also been used for robust Lasso regression [36, 26], Poisson regression [22], logistic regression [47], clustering [49, 18], as well as a large class of regression and classification problems intoduced in [29].

[45] took into consideration the issues of masking and swamping when there are multiple outliers in the data. By definition, masking refers to the situation when a true outlier is not detected because of other outliers. Swamping, on the other hand, refers to the situation when an observation conforming to the assumed distribution is considered outlying under the influence of true outliers. They pointed out that using the $\ell_0$ penalty instead of the $\ell_1$ penalty in the objective function could resolve both issues. Assuming $\sigma$ is known, adding an $\ell_0$ penalty to the negative log-likelihood function for model (2.2), the objective function becomes

$$f(\boldsymbol{\beta}, \boldsymbol{\gamma}) = \sum_{i=1}^{n} (y_i - \mathbf{x}_i^T \boldsymbol{\beta} - \gamma_i)^2 + \lambda \sum_{i=1}^{n} 1(\gamma_i \neq 0), \tag{2.3}$$

where $\lambda$ is a tuning parameter and $1(\cdot)$ is the indicator function. It can be shown that this problem can be solved by minimizing a sum of truncated quadratic functions.

*Proposition* 2.1. Minimizing (2.3) in $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ jointly is equivalent to minimizing the following sum of truncated quadratic functions in $\boldsymbol{\beta}$

$$g(\boldsymbol{\beta}) = \sum_{i=1}^{n} \min\{(y_i - \mathbf{x}_i^T \boldsymbol{\beta})^2, \lambda\}.$$

This result is consistent with the proposition by [45] that the estimate $\hat{\boldsymbol{\beta}}$ from minimizing (2.3) is an $M$-estimate associated with the skipped-mean loss. Since the objective function is non-convex, [45] proposed an iterative hard thresholding algorithm named Θ-IPOD (iterative procedure for outlier detection) to minimize it.

Similar to other iterative procedures, Θ-IPOD only guarantees local solutions. A simulation study comparing our proposed algorithm with Θ-IPOD and several other robust linear regression algorithms are presented in Section 2.4.1. We implement the Θ-IPOD algorithm in R (see Supplementary Algorithm 5 for details).

Furthermore, Proposition 2.1 can be extended to the class of generalized linear models (GLMs). Suppose that $Y_i \in R, i = 1, \ldots, n$, follow a distribution in the exponential family,

$$f(Y_i = y_i|\theta_i, \phi) = \exp\left\{\frac{y_i\theta_i - b(\theta_i)}{a(\phi)} + c(y_i, \phi)\right\},$$

where $\theta_i$ is the canonical parameter and $\phi$ is the dispersion parameter (assumed known here). For a GLM with canonical link function $g$, $\theta_i = g(\mu_i) = \mathbf{x}_i^T\boldsymbol{\beta} + \gamma_i$, the $\ell_0$-penalized negative log-likelihood function is

$$f(\boldsymbol{\beta}, \boldsymbol{\gamma}) = \sum_{i=1}^{n}\{b(\mathbf{x}_i^T\boldsymbol{\beta} + \gamma_i) - (\mathbf{x}_i^T\boldsymbol{\beta} + \gamma_i)y_i\} + \lambda\sum_{i=1}^{n}1(\gamma_i \neq 0). \tag{2.4}$$

It can be shown that minimizing (2.4) is equivalent to minimizing a sum of truncated convex functions.

*Proposition* 2.2. Minimizing (2.4) in $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ jointly is equivalent to minimizing the following function in $\boldsymbol{\beta}$

$$g(\boldsymbol{\beta}) = \sum_{i=1}^{n}\min\{b(\mathbf{x}_i^T\boldsymbol{\beta}) - (\mathbf{x}_i^T\boldsymbol{\beta})y_i, \lambda_i^*\},$$

where $\lambda_i^* = b(g(y_i)) - g(y_i)y_i + \lambda, i = 1, \ldots, n$, are constants. Since $b$ is convex [1], the above is a sum of truncated convex function.

**Example 2.3.** Suppose that $\{Y_i\}_{i=1}^n$ follow Poisson distributions with mean $\{\mu_i\}_{i=1}^n$, respectively, and that $g(\mu_i) = \log\mu_i = \mathbf{x}_i^T\boldsymbol{\beta} + \gamma_i$, where $\gamma_i = 0$ if $y_i$ conforms to the assumed distribution and $\gamma_i \neq 0$ if $y_i$ is an outlier. The $\ell_0$-penalized negative log-likelihood function is

$$f(\boldsymbol{\beta}, \boldsymbol{\gamma}) = \sum_{i=1}^{n} \left\{ e^{\mathbf{x}_i^T \boldsymbol{\beta} + \gamma_i} - (\mathbf{x}_i^T \boldsymbol{\beta} + \gamma_i) y_i \right\} + \lambda \sum_{i=1}^{n} 1(\gamma_i \neq 0). \tag{2.5}$$

According to Proposition 2.2, minimizing (2.5) is equivalent to minimizing the following function

$$g(\boldsymbol{\beta}) = \sum_{i=1}^{n} \min\{e^{\mathbf{x}_i^T \boldsymbol{\beta}} - (\mathbf{x}_i^T \boldsymbol{\beta}) y_i, \lambda_i^*\}, \text{ where } \lambda_i^* = \lambda - y_i \log y_i + y_i,$$

which is a sum of truncated convex functions.

### 2.2.2 Convex shape placement

Given a convex shape $S \subset R^d$, and $n$ points $\mathbf{p}_i \in R^d, i = 1, \ldots, n$, each associated with weight $w_i > 0$, the problem of finding a translation of $S$ such that the total weight of the points contained in $S$ is maximized has applications in the placement of facilities or resources such as radio stations, power plants or satellites [34]. For some simple shapes (e.g., circles or polygons) in low-dimensional settings, this problem has been well studied [7, 3].

We show that this problem can be solved by minimizing a sum of truncated convex functions. Without loss of generality, let $S_0 \subset R^d$ denote the region covered by $S$ when it is placed at the origin. Here the location of $S$ can be defined as the location of its centroid. For each point $\mathbf{p}_i$, let $S_i \subset R^d$ be the set of locations for placing $S$ such that it covers $p_i$. It is easy to see that $S_i = \{\mathbf{x} : \mathbf{p}_i - \mathbf{x} \in S_0\} = \{\mathbf{p}_i - \mathbf{y} : \mathbf{y} \in S_0\}$, and that the shape of $S_i$ is simply a mirror image of $S_0$ and therefore it is also convex. Furthermore, define convex function $f_i : R^d \to R$ as

$$f_i(\mathbf{x}) = \begin{cases} -w_i & \text{if } \mathbf{x} \in S_i, \\ \infty & \text{otherwise.} \end{cases}$$

Then the optimal placement of $S$ can be found by minimizing the sum of truncated convex functions $\sum_{i=1}^{n} \min\{f_i(\mathbf{x}), \lambda_i\}$ as in (2.1) where $\lambda_i = 0, i = 1, \ldots, n$.

Some examples of this application are given in Section 2.4.3.

### 2.2.3 Signal and image restoration

Signal restoration aims to recover the original signal from observations corrupted by noise. Suppose that the observed data $\mathbf{y}$ are generated from the original data $\mathbf{x}$ following the model [41]:

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \boldsymbol{\epsilon}$$

where $\mathbf{H}$ is a matrix performing some linear transformation on the data (e.g., smoothing) and $\boldsymbol{\epsilon}$ is the vector the measurement errors, often modeled as additive white Gaussian noise (AWGN). The goal is to estimate (a.k.a. restore or reconstruct) $\mathbf{x}$ from observed $\mathbf{y}$ and a known $\mathbf{H}$. When both $\mathbf{x}$ and $\mathbf{y}$ are (vectorized) images, the problem is called image restoration.

During this restoration process, one often wants to preserve the edges in the original signal, if there were any. One popular approach is to minimize the following regularized objective function (a.k.a. energy function [38]):

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} L(\mathbf{H}\mathbf{x} - \mathbf{y}) + \alpha p(\mathbf{x})$$

where $L(\mathbf{H}\mathbf{x} - \mathbf{y})$ is the loss function, usually taken as the negative log-likelihood function (e.g., $||\mathbf{H}\mathbf{x} - \mathbf{y}||^2$ in case of Gaussian noise), $p(\mathbf{x})$ is a penalty function to introduce the prior that one wishes to enforce on the original data $\mathbf{x}$, and $\alpha$ is a tuning parameter. Many penalty functions have been studied in the literature. While convex penalty functions are generally easier to optimize, non-convex penalty functions can lead to better restoration quality [39]. In particular, the truncated quadratic penalty has been found to be quite effective [37, 41]. For instance, to promote both sharp edges and smooth regions in the estimated $\hat{\mathbf{x}}$, a truncated quadratic penalty on the differences between neighboring data points can be used:

$$p(\mathbf{x}) = \sum_{i,j \in I, i \in D(j)} \min\{(x_i - x_j)^2, \lambda\},$$

where $I$ is the index set of all the data points (or pixels), and $i \in D(j)$ means that data points (or pixels) $i$ and $j$ are neighbors of each other. Together with this penalty function, the energy function $L(\mathbf{Hx} - \mathbf{y}) + \alpha p(\mathbf{x})$ with the loss function for Gaussian noise is in the form of a sum of truncated quadratic functions, where the loss function $L(\mathbf{Hx}-\mathbf{y}) = ||\mathbf{Hx}-\mathbf{y}||^2$ can be regarded as a sum of quadratic functions truncated at infinity. A simulation study comparing our proposed algorithm with other algorithms for signal restoration and an application of our proposed algorithm to image restoration on real data are presented in Section 2.4.4.

## 2.3 Methods

First, the general problem of minimizing a sum of truncated convex functions is in the class of NP-Hard. This can be shown by reducing the 3-satisfiability (3-SAT) problem [10, 25], an NP-complete problem, to the problem of minimizing a sum of truncated convex functions.

*Proposition* 2.4. The 3-SAT problem can be reduced to the problem of minimizing a sum of truncated convex functions.

Consequently, a universal algorithm for solving the general problem of minimizing a sum of truncated convex functions with polynomial running time is unlikely to exist [35]. However, when partitioning the search space such that the objective function is convex when restricted on each region and enumerating all the regions is feasible, a polynomial time algorithm does exist (note that here we consider observations as the input and hold dimensionality of the search space constant). Next, We show that it is, in fact, the case for some commonly used convex functions in low-dimensional settings.

### 2.3.1 Notations

Given $n$ convex functions $f_i : R^d \to R, i = 1, \ldots, n$, and constants $\lambda_i \in R$, $i = 1, \ldots, n$, we want to find $\mathbf{x} \in R^d$ such that the following sum is minimized at $\mathbf{x}$

$$f(\mathbf{x}) = \sum_{i=1}^{n} \min\{f_i(\mathbf{x}), \lambda_i\}. \tag{2.6}$$

Without loss of generality, we further assume $\lambda_i = 0$ for all $i$, since minimizing (2.6) is equivalent to minimizing

$$g(\mathbf{x}) = \sum_{i=1}^{n} \min\{g_i(\mathbf{x}), 0\} + \sum_{i=1}^{n} \lambda_i.$$

where $g_i : R^d \to R$ is defined as $g_i(\mathbf{x}) = f_i(\mathbf{x}) - \lambda_i$, which is also convex. Furthermore, we define $C_i \subset R^d$ as the convex region on which $f_i$ is less than or equal to zero,

$$C_i := \{\mathbf{x} : f_i(\mathbf{x}) \leq 0\},$$

and we define $\partial C_i := \{\mathbf{x} : f_i(\mathbf{x}) = 0\}$, the boundary of $C_i$, as the truncation boundary of $f_i$. Then, $\{\partial C_i\}_{i=1}^{n}$, the truncation boundaries of all the $f_i$'s, partition the domain $R^d$ into disjoint pieces $A_1, \ldots, A_m$ such that

$$A_j \cap A_k = \emptyset, \quad \forall j \neq k \quad \text{and} \quad \cup_{j=1}^{m} A_j = R^d,$$

where $A_j$ is defined as

$$A_j = (\bigcap_{k \in I_j} C_k) \cap (\bigcap_{l \notin I_j} C_l^c), \quad I_j \subset \{1, \ldots, n\}, \quad j = 1, \ldots, m,$$

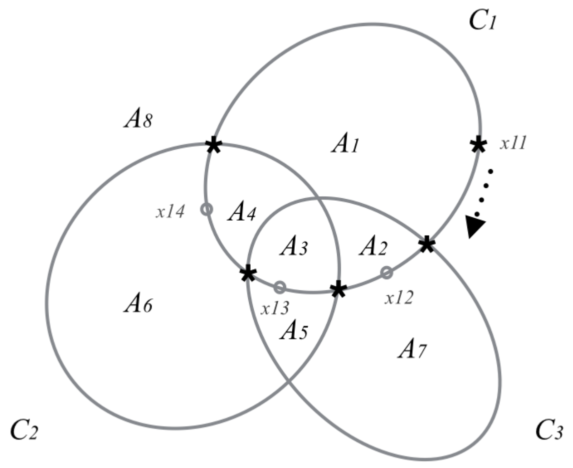

Figure 2.2: The corresponding $C_i$'s of three convex functions $f_1, f_2, f_3$ define on $R^2$, where $C_i = \{\mathbf{x} : f_i(\mathbf{x}) \leq 0\}$. The boundaries of $\{C_i\}_{i=1}^3$ partition $R^2$ into eight disjoint pieces $\{A_j\}_{j=1}^8$.

where $I_j$ is the index set for a subset of $\{f_1, \ldots, f_n\}$ such that given any $\mathbf{x} \in A_j$, $f_k(\mathbf{x}) \leq 0$ for all $k \in I_j$ and $f_k(\mathbf{x}) > 0$ for all $k \notin I_j$. An example of partitioning $R^2$ into disjoint pieces $A_1, \ldots, A_m$ is shown in Figure 2.2. The algorithms to find and traverse through all $A_j$'s while constructing the corresponding $I_j$'s will be described in Sections 2.3.2 and 2.3.3.

### 2.3.2 The general algorithm

Our goal is to find the local minimum on each region $A_j$ in the partition and take the minimum of all local minima as the global solution. That is,

$$\min_{\mathbf{x}} \sum_{i=1}^{n} \min\{f_i(\mathbf{x}), 0\} = \min_{j} \min_{\mathbf{x} \in A_j} \sum_{k \in I_j} f_k(\mathbf{x}).$$

To minimize $f(\mathbf{x})$ when restricted to $A_j$, we need to find the index set $I_j$, and minimize $\sum_{k \in I_j} f_k(\mathbf{x})$ subject to $\mathbf{x} \in A_j$, which leads to a series of constrained optimization problems. Although the objective function $\sum_{k \in I_j} f_k(\mathbf{x})$ is a sum of convex functions and therefore is also convex, the domain $A_j$ can be a non-convex set. For instance, except for $A_3$, all other $A_j$'s in Figure 2.2 are non-convex sets. Solving such constrained optimization problems can be very challenging. Fortunately, the following proposition shows that it is safe to ignore the constraint $\mathbf{x} \in A_j$ when minimizing $\sum_{k \in I_j} f_k(\mathbf{x})$, and consequently, we only need to solve a series of unconstrained convex optimization problems, which is much easier.

*Proposition* 2.5. Using the notations defined in Section 2.3.1, we have

$$\min_{\mathbf{x}} \sum_{i=1}^{n} \min\{f_i(\mathbf{x}), 0\} = \min_{j} \min_{\mathbf{x}} \sum_{k \in I_j} f_k(\mathbf{x})$$

Based on Proposition 2.5, a general framework for minimizing (2.6) is to enumerate all the regions $\{A_j\}_{j=1}^{m}$ and solve a unconstrained convex optimization problem for each region. See Supplementary Algorithm 1 for details.

### 2.3.3 Implementation in low-dimensional settings

The implementation of the general algorithm described above depends on both the class of functions $\{f_i\}_{i=1}^{n}$ and the dimension $d$. When $d = 1$, each $C_i$ is an interval on the real line and the boundary of $C_i$, $\partial C_i$, is composed of the two end-points of $C_i$, which are the locations where $f_i$ crosses zero. Without loss of generality, assuming that the $2n$ end-points of $\{C_i\}_{i=1}^{n}$ are all distinct, we can then order them sequentially along the real line which partitions $R$ into $m = 2n + 1$ fragments $\{A_j\}_{j=1}^{m}$. We can then go through them one by one sequentially and in the same time keep track of functions entering and leaving the set of untruncated functions on each fragment $A_j$. The detailed procedure for finding the global minimizer of $f(x)$ in 1-D is described in Supplementary Algorithm 2.

When $d = 2$, each $C_i$ is a convex region on $R^2$, and its boundary $\partial C_i$ is a curve. One way to enumerate all the $A_j$'s is to travel along each $\partial C_i$, and record the intersection points of $\partial C_i$ and $\partial C_k$ for $k \neq i$. We then use these intersection points to keep track of functions entering and leaving the set of untruncated functions on each $A_j$. The detailed procedure for finding the global minimizer of $f(\mathbf{x})$ in 2-D is described in Supplementary Algorithm 3.

Using the notations in Section 2.3.1 and the example in Figure 2.2 as an illustration, we start from an arbitrary point $\mathbf{x}_{11}$ on $\partial C_1$. On one side we have the region $A_1$, on which there is only one untruncated function ($I_1 = \{1\}$). On the other side we have $A_8$, on which every function is truncated ($I_8 = \emptyset$). Traveling clockwise, we come across $\partial C_3$. At this point, we add $f_3$, which gives the sets of untruncated functions on $A_2$ ($I_2 = \{1, 3\}$) and $A_7$ ($I_7 = \{3\}$). Similarly, we obtain $I_3 = \{1, 2, 3\}$ and $I_5 = \{2, 3\}$ when we come across $\partial C_2$. When we come acoss $\partial C_3$ for the second time, we remove $f_3$ from the set of untruncated function and obtain $I_4 = \{1, 2\}$ and $I_6 = \{2\}$. By repeating the process for all $C_i$'s, we enumerate the set of untruncated functions on all $A_j$'s.

What remains to be supplied in the 1-D algorithm are methods to find the end-points of any given $C_i$, and to minimize the sum of a subset of untruncated functions. Similarly, for the 2-D algorithm, we need ways to find the intersection points of any given $\partial C_i$ and $\partial C_k$, and to minimize the sum of a subset of untruncated functions. The implementation of these steps depends on the class of functions that we are dealing with. For some function classes, solutions for these steps are either straightforward or already well-studied. For instance, for quadratic functions, finding the end-points (in 1-D) or finding the intersections (in 2-D) requires solving quadratic equations, for which closed-form solutions exist. Minimizing the sum of a subset of quadratic functions can also be solved in closed-form. For convex shape placement problem described in Section 2.2.2, published algorithms exist for these steps for commonly encountered convex shapes such as circles or convex polygons [12]. For more general convex functions (e.g., those described in Section 2.2.1 for GLMs), iterative algorithms (e.g., gradient descent or the Newton-Raphson method for differential convex functions; bisection for non-differentiable univariate convex functions ) can be used for these steps.

### 2.3.4 Extension to high-dimensional settings

In three or higher dimensions, our algorithm can be implemented by following the same idea of tracking all the intersection points as in the 2-D case. Essentially, each boundary $\partial C_i$ is a $d-1$ dimensional surface and enumerating all the $A_j$'s can be achieved by traversing through all the pieces on each $\partial C_i$ that are formed by its intersections with all other $\partial C_k$'s, which is, in turn, a $d-1$ dimensional problem. For instance, when $d = 3$, we need to find all the intersection curves of $\partial C_i$ and $\partial C_k$ (both of which are surfaces) for $i \neq k$, and traverse along each intersection curve while keep tracking all other surfaces $\partial C_j, j \neq i \neq k$, it crosses. Apparently, this algorithm becomes increasingly complicated and inefficient for larger $d$, which renders it impractical.

Here, we propose a compromised but efficient extension of our proposed algorithm to high-dimensional settings. The price we pay is to give up the global minimizer, which is a sensible choice as Proposition (2.4) has shown that the general problem is NP-Hard. In particular, we propose to solve for an approximate solution using a cyclic coordinate descent algorithm, where we optimize one parameter a time while keeping all other parameters fixed, and cycle through all the parameters until converge. When restricting to only one parameter, the objective function is simply a sum of truncated convex functions in 1D. Therefore, we can use our 1-D algorithm to solve this subproblem in each iteration. This single-coordinate update algorithm, however, only guarantees the convergence of the sequence of objective function values evaluated. It does not guarantee to converge to a local minimizer. That is, $f(\mathbf{x}_i)$ converges to $\inf f(\mathbf{x}_i)$ but we need more conditions to guarantee the convergence of

$\mathbf{x}_i$. See Supplementary Algorithm 4 for details. We will evaluate the performance of this algorithm using both simulated and real data experiments in Section 2.4.4.

#### 2.3.5 Time complexity analysis

For time complexity analysis of our proposed algorithms, in low-dimensional settings, we can regard the dimension $d$ as a constant. That is, any univariate function of $d$ can be considered as $O(1)$.

For the 1-D algorithm, finding the $2n$ end-points takes $O(nS)$ time, where $S$ is the time for finding the two endpoints of a given function. Ordering the $2n$ end-points takes $O(n \log n)$ time. Traversing through all the end-points takes $O(nT)$ time, where $T$ is the time for minimizing the sum of a subset of untruncated functions. Similarly, for the 2-D algorithm, finding all the intersection points takes $O(n^2 S)$ time, where $S$ is the time for finding all the intersection points of any two given functions. Sorting all the intersection points along all the boundaries $\{\partial C_i\}_{i=1}^n$ takes $O(n^2 K \log(nK))$ time, where $K$ is the maximum number of intersection points any two boundaries $\partial C_i$ and $\partial C_j$ can have. Traversing through all the intersection points takes $O(n^2 KT)$ time.

First, we show that $K = O(1)$ for a large class of truncated convex functions. That is, given any two truncated convex functions in the class, the maximum number of intersection points their boundaries can have is bounded by a constant.

**Definition 2.6.** For any positive integer $k \in Z^+$, a class of curves $\mathcal{C}$ in $R^2$ is said to be $k$-intersecting if and only if for any two distinct curves in $\mathcal{C}$, the number of their intersection points is at most $k$.

**Definition 2.7.** A class of truncated functions in $R^2$ is said to be $k$-intersecting if and only if the set of their truncation boundaries is $k$-intersecting.

**Example 2.8.** The class of truncated quadratic functions in $R^2$ with positive definite Hessian matrices is $k$-intersecting with $k = 4$. This is easy to see given the facts that the truncation boundary of a quadratic function in $R^2$ with positive definite Hessian matrix is an ellipse, and two distinct ellipses can have at most four intersection points.

In fact, according to Bézout's theorem, the number of intersection points of two distinct plane algebraic curves is at most equal to the product of the degrees of the corresponding polynomials. Therefore, a class $\mathcal{F}$ of truncated bivariate polynomials is $k^2$-intersecting if for any function $f \in \mathcal{F}$ its untruncated version is a polynomial of degree at most $k$.

While $S$ and $T$ depend on the class of functions that we are dealing with, for some function classes, we have $S = O(1)$ and $T = O(1)$. That is, they both take constant time.

**Example 2.9.** For quadratic functions with positive definite Hessian matrices, $T = O(1)$. This is easy to see given the following three facts:

1. Given $n$ quadratic functions $f_i = \frac{1}{2}\mathbf{x}^T\mathbf{A}_i\mathbf{x} + \mathbf{b}_i^T\mathbf{x} + c_i, i = 1, \ldots, n$, their sum is $\sum_i f_i(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\mathbf{A}\mathbf{x} + \mathbf{b}^T\mathbf{x} + c$, where $\mathbf{A} = \sum_i \mathbf{A}_i, \mathbf{b} = \sum_i \mathbf{b}_i$, and $c = \sum_i c_i$, which is also a quadratic function.

2. To update the sum of quadratic functions when adding a new function to the sum or removing an existing function from the sum, we only need to update $\mathbf{A}, \mathbf{b}$ and $c$, which takes $O(1)$ time (it is in fact $O(d^2)$ time but can be simplified as $O(1)$ time since we consider $d$ as a constant in low-dimensional settings).

3. The minimizer of any quadratic function $\frac{1}{2}\mathbf{x}^T\mathbf{A}\mathbf{x}+\mathbf{b}^T\mathbf{x}+c$ with positive definite Hessian matrix is $-\mathbf{A}^{-1}\mathbf{b}$, which takes $O(1)$ time to compute (it is in fact $O(d^3)$

time but can be simplified as $O(1)$ time since we consider $d$ as a constant in low-dimensional settings).

Furthermore, $S = O(1)$, since all the intersection points (up to four of them) of any two given ellipses can be found using closed-form formulas [43].

Putting Examples 2.8 and 2.9 together, we know that the running time of the 1-D algorithm for sum of truncated quadratic functions with positive definite Hessian matrix is $O(n \log n)$, and the running time of the 2-D Algorithm for sum of truncated quadratic functions with positive definite Hessian matrix is $O(n^2 \log n)$. The time complexity analysis for other class of functions can be conducted similarly.

In high-dimensional settings, however, the running time of the general algorithm will be at least $O(n^d \log n)$, where $d$ is the dimension. In another word, the running time grows exponentially as the dimension increases, which is typical for NP-Hard problems. It is easy to see that the running time of the cyclic coordinate descent algorithm is $O(kdn \log n)$, where $k$ is the number iterations to converge, and $O(dn \log n)$ is the time for each round of $d$ one-dimensional updates.

## 2.4 Experiments

### 2.4.1 Outlier detection in simple linear regression

We simulate data for outlier detection in simple linear regression as described in Section 2.2.1 and compare the performance of our proposed method with the Θ-IPOD algorithm [45] and three other robust estimation methods: MM-estimator [51], least trimmed squares (LTS) [44] and [19] one-step procedure (denoted as GY). Our goal is to estimate the regression coefficients and identify the outliers with $\sigma$ assumed to be known and set to be 1. In other words, we estimate only $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ in (2.2). Given $n$ observations and $k$ outliers, let $\mathbf{X} = [\mathbf{1}_n, (\mathbf{x}_1, \ldots, \mathbf{x}_n)^T]$, $\boldsymbol{\beta} = (\beta_0, \beta_1)^T = (1, 2)^T$, and $L$ be a parameter controlling the leverage of the outliers. When $L > 0$, $x_i$ is drawn from $uniform(L, L+1)$ for $i = 1, \ldots, k$, and from $uniform(-15, 15)$ for $i = k+1, \ldots, n$. $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_n)^T$ represents deviations from the means, and each $\gamma_i$ is drawn from $exponential(0.1)+3$ for $i = 1, \ldots, k$, and $\gamma_i = 0$ for $i = k+1, \ldots, n$. Based on a popular choice for $\sqrt{\lambda}$ as $2.5\hat{\sigma}$ [45, 48, 32], we set $\sqrt{\lambda}$ as 2.5.

We simulate 100 independent data sets, each with 100 observations (i.e., $n = 100$). The results are shown in Figure 2.3 and Supplementary Table A1. The performance of each method is evaluated by the masking probability and the swamping probability under two scenarios: (i) No $L$ applied (denotes as $L = 0$), that is, $x_i$ is drawn from $uniform(-15, 15)$ for $i = 1, \ldots, n$, and (ii) $L = 20$. Masking probability, as in [45], is defined as the proportion of undetected true outliers among all outliers. Swamping probability, on the other hand, is the fraction of normal observations recognized as outliers. When implementing MM, LTS and GY estimators, the fact that $\sigma^2$ is known is not exploited because we use packaged functions for these methods, which assume $\sigma^2$ to be unknown and have no arguments for specifying $\sigma^2$.

### 2.4.2 Sum of truncated quadratic functions

We simulate sum of truncated quadratic functions with positive definite Hessian matrix in $R^2$ and compare the performance of the proposed algorithm with several other competing algorithms including a global search algorithm (the DIRECT algorithm) [24] and a branch-and-bound global optimization algorithm (StoGO) [31] both implemented in R package nloptr, a generalized simulating annealing algorithm (SA) implemented in R package GenSA [50], a particle swarm optimization algorithm (PSO) implemented in R package hydroPSO [52], as well as the difference of convex functions (DC) algorithm [2] which has been used to solve problems with truncated convex functions [46, 8]. We implement the DC algorithm in R (see Supplementary

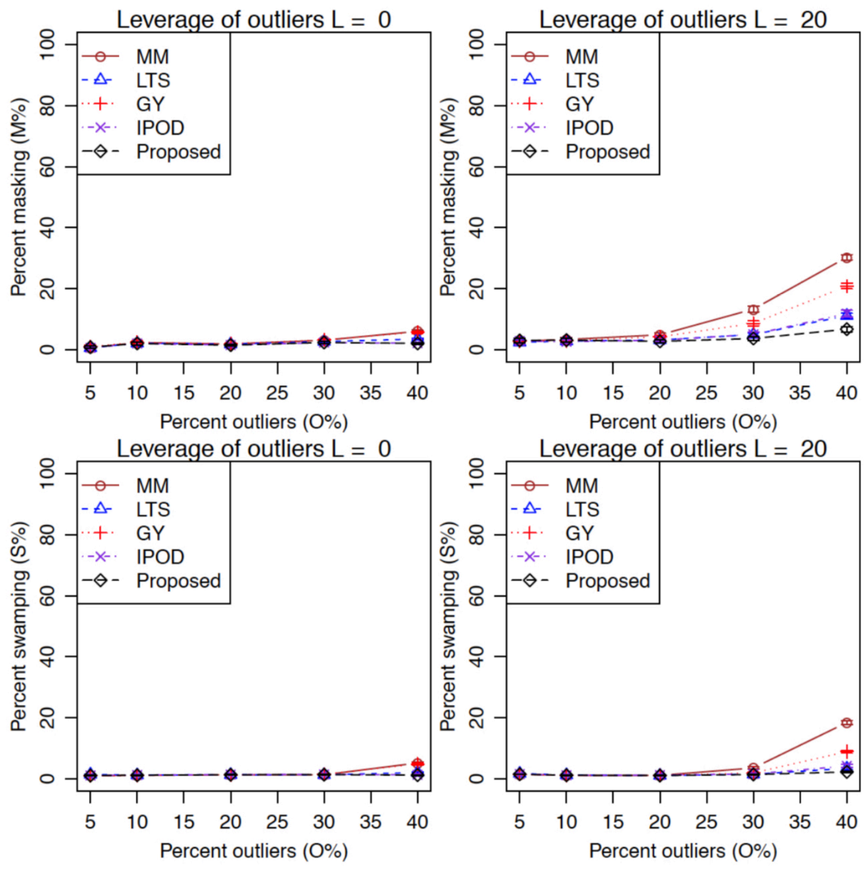

Figure 2.3: Comparison of different methods for outlier detection in simple linear regression. The figures show the mean percents of masking (top) and swamping (bottom) for different leverages of outliers: $L = 0$ (left) and $L = 20$ (right) and different percents of outliers (O%) for all the methods using 100 simulated replicates. The standard errors of the means are shown as error bars. When estimating the regression coefficients and identifying the outliers, we assume that the $\sigma^2$ is known and is 1. In other words, we estimate only $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ in (2.2). When implementing MM, LTS and GY estimators, the fact that $\sigma^2$ is known is not exploited because we use packaged functions for these methods, which assume $\sigma^2$ to be unknown and have no arguments for specifying $\sigma^2$.

Section A1.4 for details).

Following [20], we compare the performance of all the algorithms in terms of their effectiveness in finding the global minimum. We measure effectiveness by the success rate, where a success for a given algorithm in a given run is defined as having the estimated minimum no greater than any other algorithms by $10^{-5}$. This tolerance value is allowed to accommodate numerical precision issues. We set a maximum number of $10^4$ function evaluations, a maximum number of $10^4$ iterations and a convergence tolerance level of $10^{-8}$ for all competing algorithms whenever possible. See Supplementary Table A2 for details.

We randomly generate truncated quadratic functions in $R^2$ with varying degrees of complexity. Specifically, given a quadratic function with a positive definite Hessian matrix in $R^2$ truncated at zero, the truncation boundary is an ellipse. Let $a$ and $b$ be the lengths of the two axes of the ellipse, $u$ and $v$ be the x and y coordinates of the center of the ellipse, $\theta$ be the angle between the long axis of the ellipse and the x-axis, and $-z$ be the lowest value of the function. For simplicity, we use a single tuning parameter $C$ to control the complexity of the objective function. The larger the $C$, the more local minima the objective function will have. Examples of objective functions with different values of $C$ are given in Figure 2.4. In particular, we randomly sample $\theta$ from $uniform(0, \pi)$, $a$ from $uniform(0.01, 0.5)/C$, $b$ from $uniform(0.01, 0.5)$, $u$ and $v$ from $uniform(0, 1)$ and $z$ from $uniform(-10, -1)$. We simulate three scenarios where $C$ is 1, 5, and 10, respectively, and we compute the coefficients of the corresponding quadratic functions based on the above six parameters. For each value of $C$, we simulate 100 independent data sets each with 50 random quadratic functions (i.e., $n = 50$) truncated at $\lambda = 0$.

The performance of the proposed algorithm and other competing algorithms are

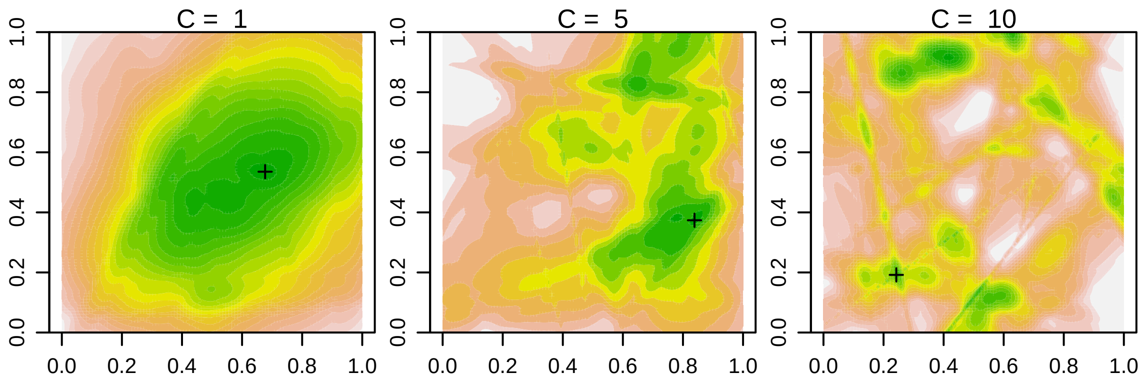

Figure 2.4: Contour plots of randomly generated sum of truncated quadratic functions in $R^2$. Global minima are marked with the plus sign.

shown in Figure 2.5 and Supplementary Table A3. We can see that our proposed algorithm has a success rate of 100% regardless of the value of $C$, as it guarantees to find the global minimizer. For all other competing algorithms, their success rates decline when $C$ increases.

### 2.4.3 Convex shape placement

Following Section 2.2.2, we randomly sample 30 points (i.e., $n = 30$) uniformly from the $[0, 1] \times [0, 1]$ unit square, and use our proposed algorithm to find a location to place $S$ such that it covers the maximum number of points. To demonstrate the generality of our proposed algorithm, we consider three shapes here: circle, square and hexagon. The results are shown in Supplementary Figure A1.

### 2.4.4 Signal and image restoration

Following Section 2.2.3, we simulate 1-D signal with additive Gaussian noise, and compare the performance of the proposed algorithm with several other algorithms including DIRECT, StoGO, SA, PSO (See Section 2.4.2 for more details of these algorithms) and a recently published iterative marginal optimization (IMO) algorithm [41], which was specifically designed for signal and image restoration. We implement the IMO algorithm in R (see Supplementary Section A1.5 for details).

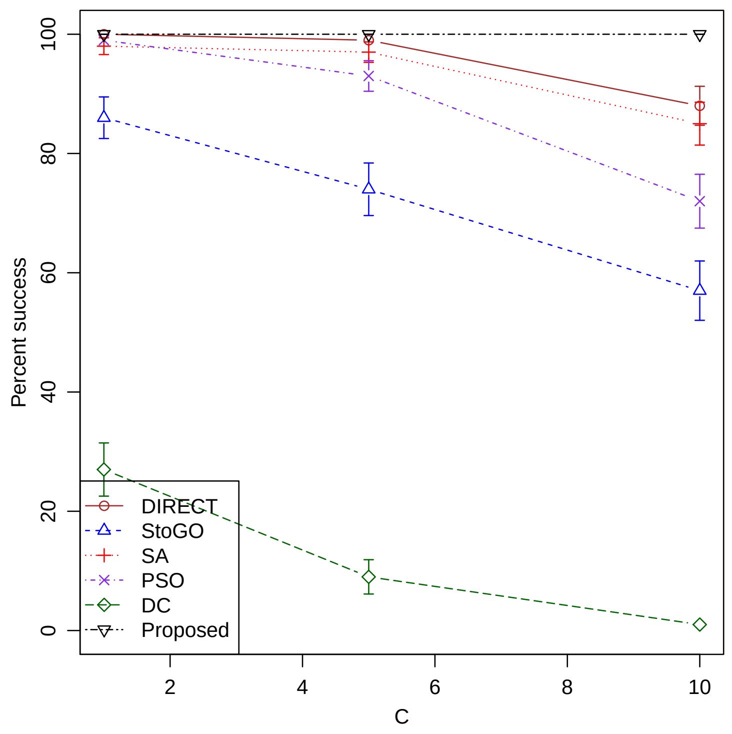

Figure 2.5: Comparison of different algorithms for minimizing the sum of 50 randomly generated truncated quardratic funstions in 2-D. The figure shows the mean success rates (in percents) for all the methods using 100 simulated replicates for different complexities of the functions ($C$). The standard errors of the means are shown as error bars.

The DC algorithm turns out to be numerically equivalent to the IMO algorithm, but much slower. Therefore, we did not include the DC algorithm in the comparison and simply named the IMO algorithm as IMO/DC.

The data are simulated by adding random Gaussian noise sampled i.i.d. from $N(0,1)$ to an underlying true signal. Each data set contains 100 data points equally spaced on the interval $[0,1]$. The true signal is design to be piece-wise smooth with different pieces being constant, linear, quadratic or sine waves (see Figure 2.6). All the algorithms are used to restore the signal by minimizing the following objective function,

$$\hat{\mathbf{y}} = \arg\min_{\hat{\mathbf{y}}} \sum_{i=1}^{d} (\hat{y}_i - y_i)^2 + w \sum_{i=1}^{d-1} \min\{(\hat{y}_i - \hat{y}_{i+1})^2, \lambda\},$$

where $d = 100, y_i$ and $\hat{y}_i, i = 1, \ldots, d$, are the observed and restored values at data

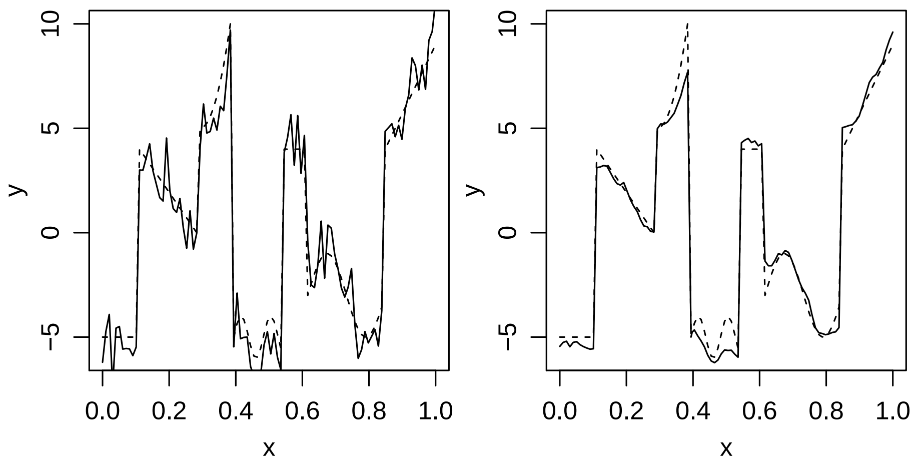

Figure 2.6: Simulated random signal (left) and restored signal (right) are shown in solid lines. The underlying true signal are shown in dashed lines.

point $i$, respectively. That is, we are solving the sum of 199 truncated quadratic functions (99 of them are truncated at $\lambda$, and the remaining 100 of them are truncated at infinity) in a 100-dimensional parameter space. The tuning parameters are empirically set as $w = 4$ and $\lambda = 9$, respectively.

We measure the performance of these algorithms using four different metrics:

1. Success rate, which is defined in Section 2.4.2. Note a success here only means that a given algorithm has found the best solution among all algorithms, which may or may not be the global minimizer.

2. Relative loss, which is defined as $|f(\hat{\mathbf{y}}) - f(\mathbf{y}^*)|/|f(\mathbf{y}^*)|$, where $\hat{\mathbf{y}}$ and $\mathbf{y}^*$ are the solution found by a given algorithm and the best solution found by all algorithms, respectively.

3. Root mean square error (RMSE), which is defined as $\sqrt{d^{-1}\sum_{i=1}^{d}(\hat{\mathbf{y}}_i - \tilde{\mathbf{y}}_i)^2}$, where $\hat{\mathbf{y}}$ and $\tilde{\mathbf{y}}$ are the solution found by a given algorithm and the underlying true signal, respectively.

4. Running time, measured in seconds.

Table 2.1: Comparison of different algorithms for signal restoration. The table shows the mean success rates (in percents), relative losses, root mean square errors (RMSE), as well as running times (in seconds) for all the methods using 100 simulated replicates. The standard errors of the means are given in parentheses.

| | DIRECT | StoGO | SA | PSO | IMO/DC | Proposed |
|---|---|---|---|---|---|---|
| Success rate | 0.0 (0.0) | 8.0 (2.7) | 52.0 (5.0) | 0.0 (0.0) | 28.0 (4.5) | 84.0 (3.7) |
| Relative loss | 0.08 (0.00) | 0.05 (0.02) | 0.04 (0.01) | 0.17 (0.01) | 0.10 (0.01) | 0.01 (0.00) |
| RMSE | 0.66 (0.01) | 0.56 (0.01) | 0.59 (0.01) | 0.63 (0.01) | 0.56 (0.01) | 0.55 (0.01) |
| Time | 0.40 (0.00) | 61.29 (0.27) | 0.31 (0.00) | 12.39 (0.13) | 1.50 (0.07) | 0.04 (0.00) |

The performance of the proposed algorithm and other competing algorithms are summarized in Table 2.1. In general, the proposed algorithm outperforms all other methods in terms of success rate, relative loss, and RMSE. It is also significantly faster than all other algorithms.

Finally, we apply the proposed algorithm for image restoration. Both synthetic and real images are used for this experiment (see Figure 2.7 and Supplementary Figure A2). All images are resized to $256 \times 256$, converted to gray scale and normalized to have pixel intensity levels in $[0, 1]$. Independent Gaussian noise sampled from $N(\mu = 0, \sigma^2 = 0.01)$ is added to each pixel, and the proposed algorithm is used to restore the original image via minimizing the following objective function,

$$\hat{\mathbf{z}} = \arg\min_{\hat{\mathbf{z}}} \sum_{i \in I} (\hat{z}_i - z_i)^2 + w \sum_{i,j \in I, i \in D(j)} \min\{(\hat{z}_i - \hat{z}_j)^2, \lambda\},$$

where $z_i$ and $\hat{z}_i, i \in I$, are the observed and restored intensity values at pixel $i$, respectively, $i \in D(j)$ means that pixels $i$ and $j$ are neighbors of each other, and the tuning parameters are empirically set as $w = 2$ and $\lambda = 0.02$, respectively. From Figure 2.7 and Supplementary Figure A2, we see that compared with Gaussian smoothing, the proposed algorithm can restore the smoothness in the image while maintaining the sharp edges. Even though this problem has a dimension of $d = 256 \times 256 = 65,536$ and the number of truncated quadratic functions is $n = 256 \times 256 + 2 \times 256 \times 255 = 196,096$, it only takes about 10 seconds for our algorithm to

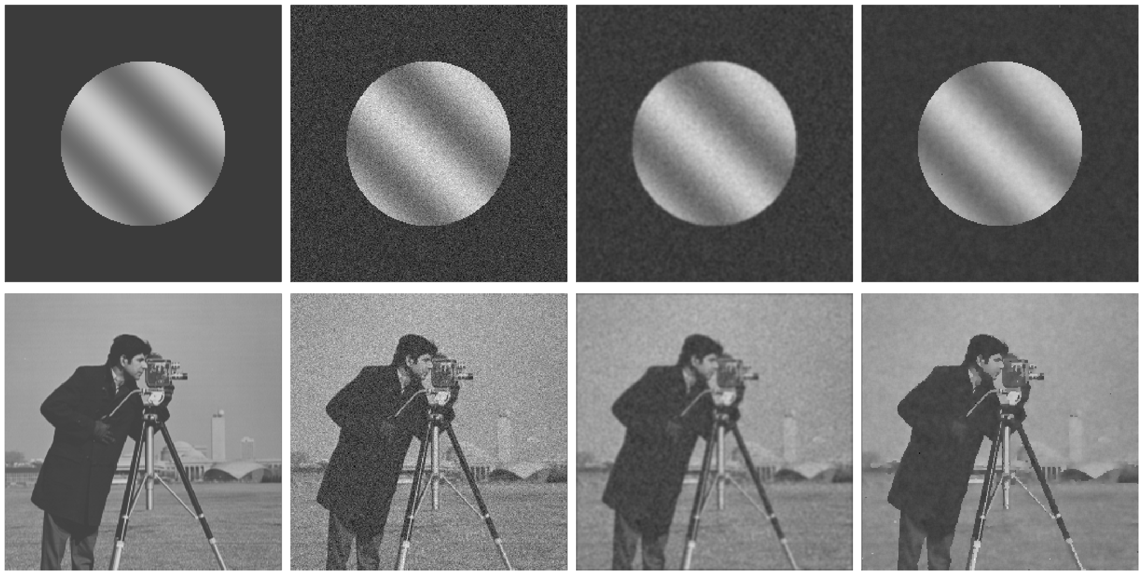

Figure 2.7: Restoration of synthetic and real images. For each row, from left to right: original image, image with Gaussian noise added, image restored using Gaussian smoothing with a $5 \times 5$ kernel and image restored using proposed algorithm.

converge.

## 2.5 Discussion

We know that summing convex functions together still gives us a convex function. Although simply truncating the function at a given level does not seem to add much complexity to a convex function, the sum of truncated convex functions is not necessarily convex, which makes this class very powerful and flexible in modeling various kinds of problems, as illustrated in several examples given in Section 2.2. Figure 2.4 further demonstrates the diverse landscape that can be achieved by a sum of truncated quadratic functions. This flexibility is supported by Proposition 2.4, which implies that any problem in the class of NP can be reduced to the minimization of a sum of truncated convex functions. A potential future work is to approximate a given non-convex function by a sum of truncated quadratic functions and then use our proposed algorithm to minimize it.

In the cyclic coordinate descent algorithm, instead of performing a univariate update in each round, we can also perform a bivariate update in each round using the 2-D algorithm (i.e., using a block coordinate descent algorithm), which may help increase the chance of finding the global minimizer, at the cost of more intensive computation.

Besides the applications described in this paper, minimizing a sum of truncated convex functions also has many other applications, such as detecting differential gene expression [23] (See Supplementary Section A1.1) and personalized dose finding [8]. This paper demonstrates that the proposed algorithm can be quite efficient when the truncation boundaries of the class of convex functions are simple shapes such as an ellipse and convex polygon, which cover the cases of truncated quadratic functions and truncated $\ell_1$ penalty (TLP [46]). Although these functions are seemingly limited, their applications are abundant, and we have shown only a few selected examples in this paper. In our future work, we will investigate the application of our proposed algorithm to other classes of convex functions.

R programs for reproducing the results in this paper are available at `http://www-personal.umich.edu/~jianghui/stcf/` [42]. We used Rcpp package that substantially decreased the run time of our algorithm [15, 14, 4]. We used the algorithm created by David Eberly (2015) to find intersections of ellipses in our 2D algorithm [13].

# CHAPTER III

# Integrating Poly(A) Capture and Exome Capture RNA-Seq Data

This chapter is motivated by a project for which we need to predict one type of RNA-seq measurements from another so that data from different types of measurements can be combined into one analysis. An initial attempt is to use gene-wise regressions. However, as there are more than 18,000 genes, this approach leads to a large number of parameters. Also, it would take at least 30 subjects to build a simple regression while 5 to 10 subjects is the desired sample size. In this chapter, we explain the differences between the two types of RNA-seq measurements and demonstrate potential biases when the differences are not eliminated before combining the data. Then we show that by using a linear mixed model, we not only reduce the technical differences between the two types of measurements but also improve the efficiency of the prediction. Finally, we described the discovery of outliers when fitting the data to a mixed model, which leads to the third project: outlier detection for the linear mixed model.

## 3.1 Introduction

Measuring the amount of messenger RNA (mRNA) molecules provides proxies for gene expression levels. A common way to extract mRNA molecules is to use oligo-dT probes targeting on the Poly(A) tails, which distinguishes mRNAs from other types of RNA molecules (Figure 3.1). This approach, however, is limited when the specimen is profoundly degraded or formalin-fixed. When the samples are degraded, mRNA molecules could lose the Poly(A) tails. On the other hand, when the specimens are formalin fixed, the adenines of the Poly(A) tails are altered such that the oligo-dT probes no longer anneal well.

A new protocol called capture RNA sequencing was developed to overcome these difficulties: instead of oligo-dT molecules, the probes are made of short sequences designed for targeted genes. [9] showed that RNAseq measurements (Figure 3.2)based on this new protocol leads to more accurate measurements when the specimens are degraded.

Despite the effectiveness of capture sequencing for degraded or formalin-fixed samples, most RNA-seq reference libraries in major databases are based on samples process by the traditional Poly(A) protocol. However, we will show that there are differences between RNA-seq measurements from these two types protocols and directly combining these two types of libraries in analyses could introduce technical biases. Therefore, for studies requiring comparison with reference libraries, it would be costly to re-build these reference libraries based on capture sequencing protocol.

In this chapter, we demonstrate the differences between poly(A) libraries and CaptureSeq libraries. Then we show that the Poly(A) measurements can be predicted from the CaptureSeq measurements efficiently using the linear mixed model. With the prediction model, we can eliminate technology biases and combine the capture sequencing measurements from degraded or formalin-fixed samples and Poly(A) based measurements from most reference libraries for efficient data use.

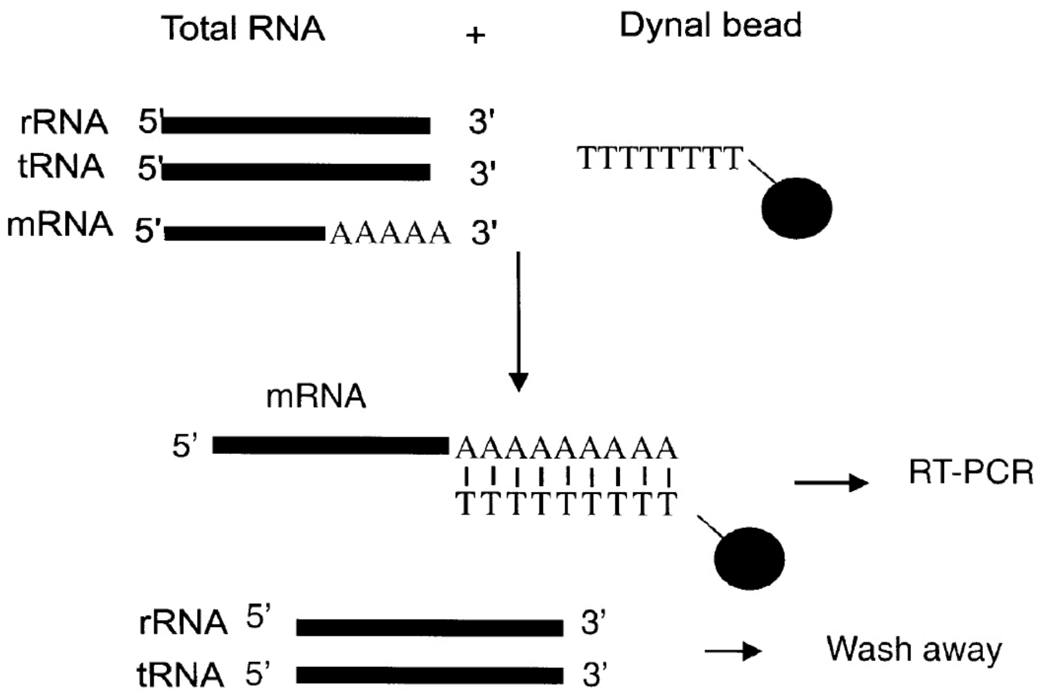

Figure 3.1: mRNA isolation, adopted from [21]. In the process of RNA-seq library preparation, RNAs are first isolated from tissue samples using commercial kits. Then mRNAs are isolated from total RNAs by annealing to oligo-dT beads. rRNAs and tRNAs are washed away before mRNAs are released from the beads.

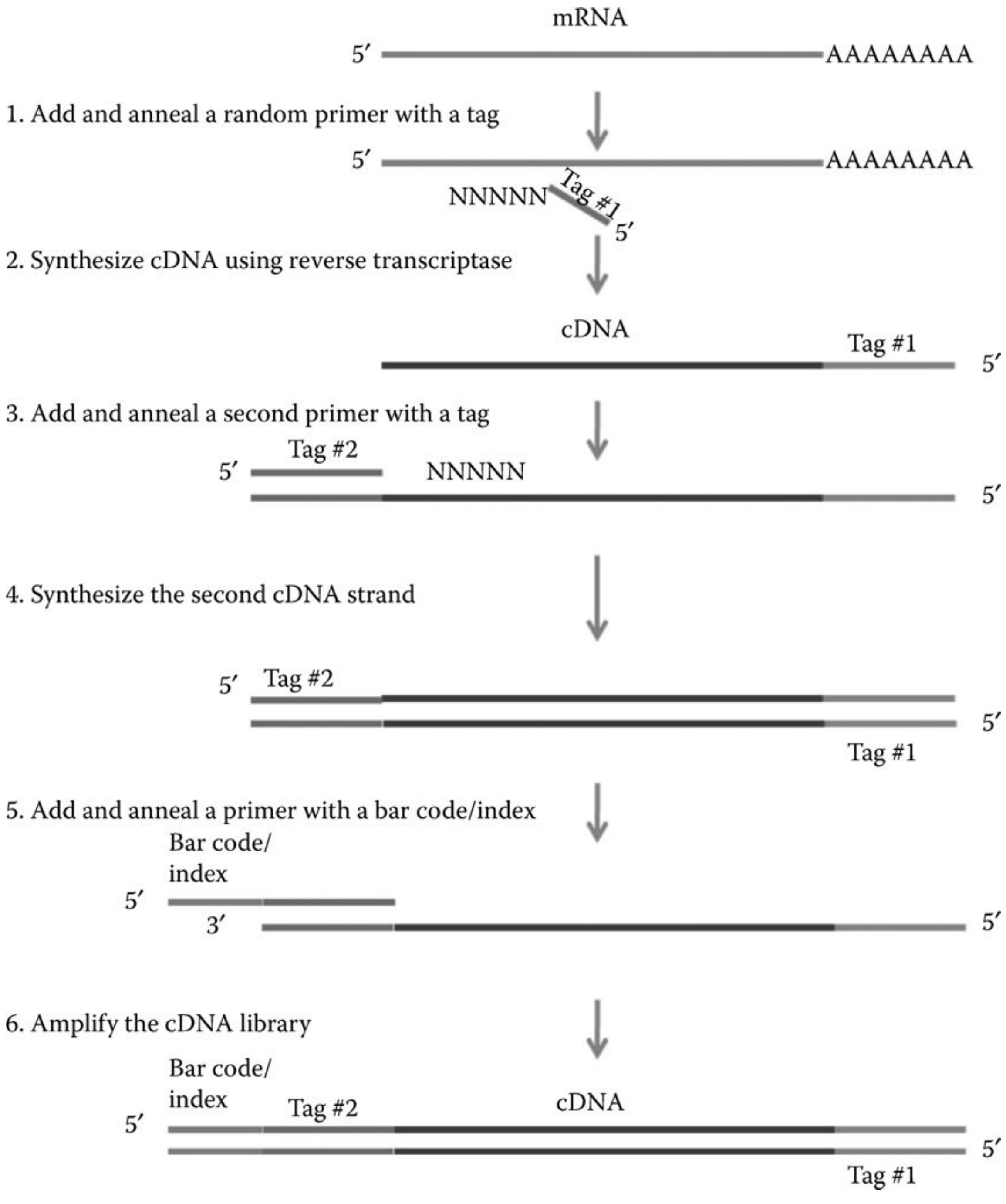
mRNA
5′
AAAAAAAA
1. Add and anneal a random primer with a tag
NNNNN
Tag #1
2. Synthesize cDNA using reverse transcriptase
cDNA
3. Add and anneal a second primer with a tag
Tag #2
4. Synthesize the second cDNA strand
5. Add and anneal a primer with a bar code/index
Bar code/ index
3′
6. Amplify the cDNA library

Figure 3.2: RNA-seq workflow, adopted from [27]. The purified mRNAs are first fragmented into smaller pieces and reverse-transcribed to complementary DNAs (cDNAs). After formation of one strand of cDNA, the mRNA strand is removed and replaced by another strand of cDNA to generate a double-stranded cDNAs. Each end of the double-stranded DNA is then repaired, adenylated and ligated by adaptor before being enriched by polymerase chain reaction (PCR). Once a library has passed the quality control, it can be sent to various sequencing platforms and generate read counts data.

Table 3.1: Origins of cancer tissues of the 372 patients.

| Tissue origins and Number of patients | | | | | |
|---|---|---|---|---|---|
| Breast | 67 | Prostate | 64 | Sarcomatoid | 42 |
| Unknown | 24 | Skin | 20 | Gall Bladder | 19 |
| Lung | 18 | Bladder | 14 | Esophagus | 14 |
| Ovary | 11 | Pancrease | 10 | Colon | 9 |
| Oral | 9 | Other | 8 | Stomach | 7 |
| Parotid Gland | 7 | Adrenal Gland | 7 | Brain | 6 |
| Kidney | 5 | Liver | 5 | Testis | 3 |
| Thymus | 2 | Thyroid Gland | 1 | | |

## 3.2 The Data

We received paired capture sequencing and Poly(A) RNA-seq read counts data from 372 cancer samples. There are a total of 23 types of cancers: prostate cancer and breast cancer account for 17 percents and 18 percents of the patients (Table 3.1). Of the 372 samples, 366 of them were frozen; two of them were refrigerated, and 4 were fresh. For each sample, a capture sequencing library and a poly(A) library were prepared. As the sample tissues were obtained by core needle biopsy, there were various degrees of tissue degradation. , and the RNA integrity numbers (RINs) of all 372 pairs of libraries were recorded except for 9 libraries. Figure 3.3 shows the distribution of the RINs. In general, a RIN $\geq 7$ is considered sufficient and it is often preferred to have a RIN $\geq 8$ [11].

For 18236 genes out of the 18955 genes, we were able to obtain information on gene lengths, lengths of sequences overlapping with those of probes, GC contents by matching Ensembl gene id in the Genome Reference Consortium Human genome build 38 (GRCh38) (Appendix, Figure 3.11). For subsequent analyses, we normalize the read data by counts per million (CPM) and take the base two logarithms.

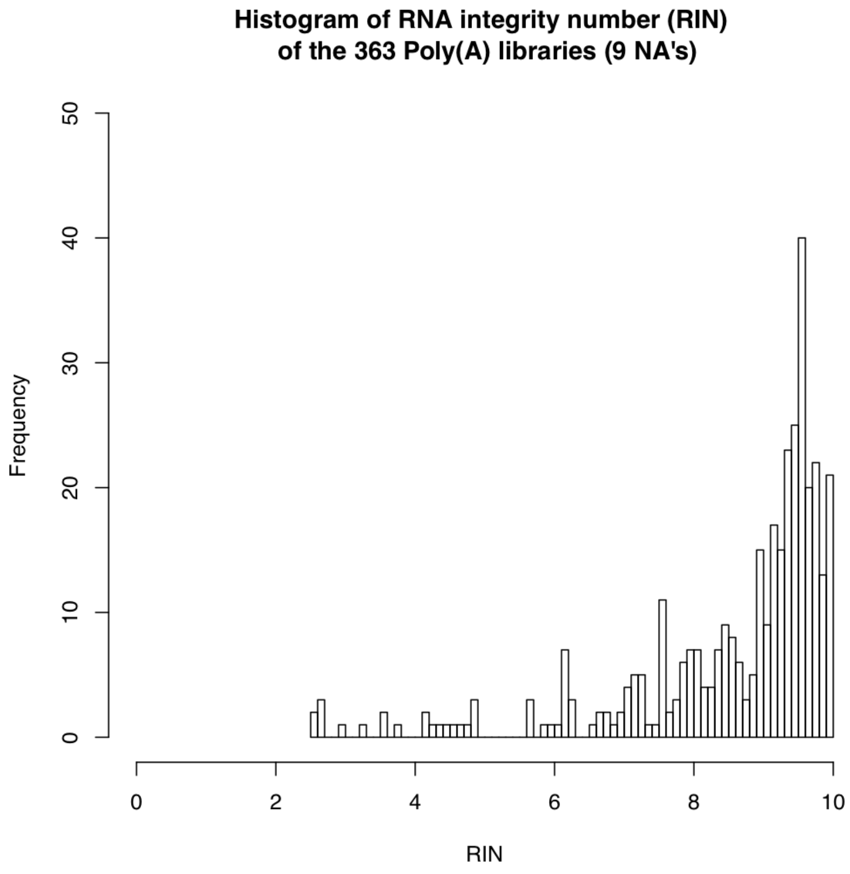

Figure 3.3: Distribution of the RINs of 363 poly(A) libraries. Of the 363 CaptureSeq libraries, 357 had exactly the same RINs as their poly(A) counterparts. The other 6 CaptureSeq libraries had a difference from the poly(A) measurements as -2.7, -1.5, -1.1, -0.6, 0.1, and 0.7. The RINs range from 2.5 from 10 with a mean of 8.6 and a median of 9.2. The 1st and the 3rd quartiles are 8.1 and 9.6.

## 3.3 Evidence of differences between the two types of measurements

To investigate the differences in measurements between the poly(A) and capture sequencing libraries, we performed cluster analyses on the poly(A) and capture sequencing log2CPM for patients with prostate cancers based on the top 50 most varied genes across the 50 libraries. The result, as shown in Figure 3.4, suggested that the mixed measurements clustered more by protocols than by individuals.

To see if there are specific genes that cause the clustering by different protocols, we used Elastic Net logistic regression to regress the two measurement types on the log2(CPM) measurements of all genes and found four genes with non-zero coefficients (Table 3.2). The classification error rate on the testing was 0.005. Gene ontology (GO) analysis for the four genes using David 6.7 found one keyword: "UBl conjugation" with a p-value of 0.03 after Benjamini-Hochberg adjustment (FDR). We also performed differential expression analysis on the two types of measurements using DESeq2 and found 14608 significant genes among a total of 18236 genes with adjusted p-value $< 0.05$, which suggests diffuse differences between these two types of measurements.

Finally, to confirm that without correction, the differences in measurements based on these two types of protocol would introduce biases and false positive findings, we randomly draw 25 patients' poly(A) $\log_2$CPM measurements out of 50 prostate cancer patients and compare the gene expression of these 25 patients with the remaining 25 patients using the function "limma trend". We perform such experiment for 1000 times and record: 1. the number of significant genes in each experiment (summarized in Figure 3.5), and 2. the number of genes found to be significantly differentially expressed in more than 50 experiments (summarized in Table 3.3) out of the 1000 experiments. We repeat the same comparison between 25 randomly drawn Poly(A)

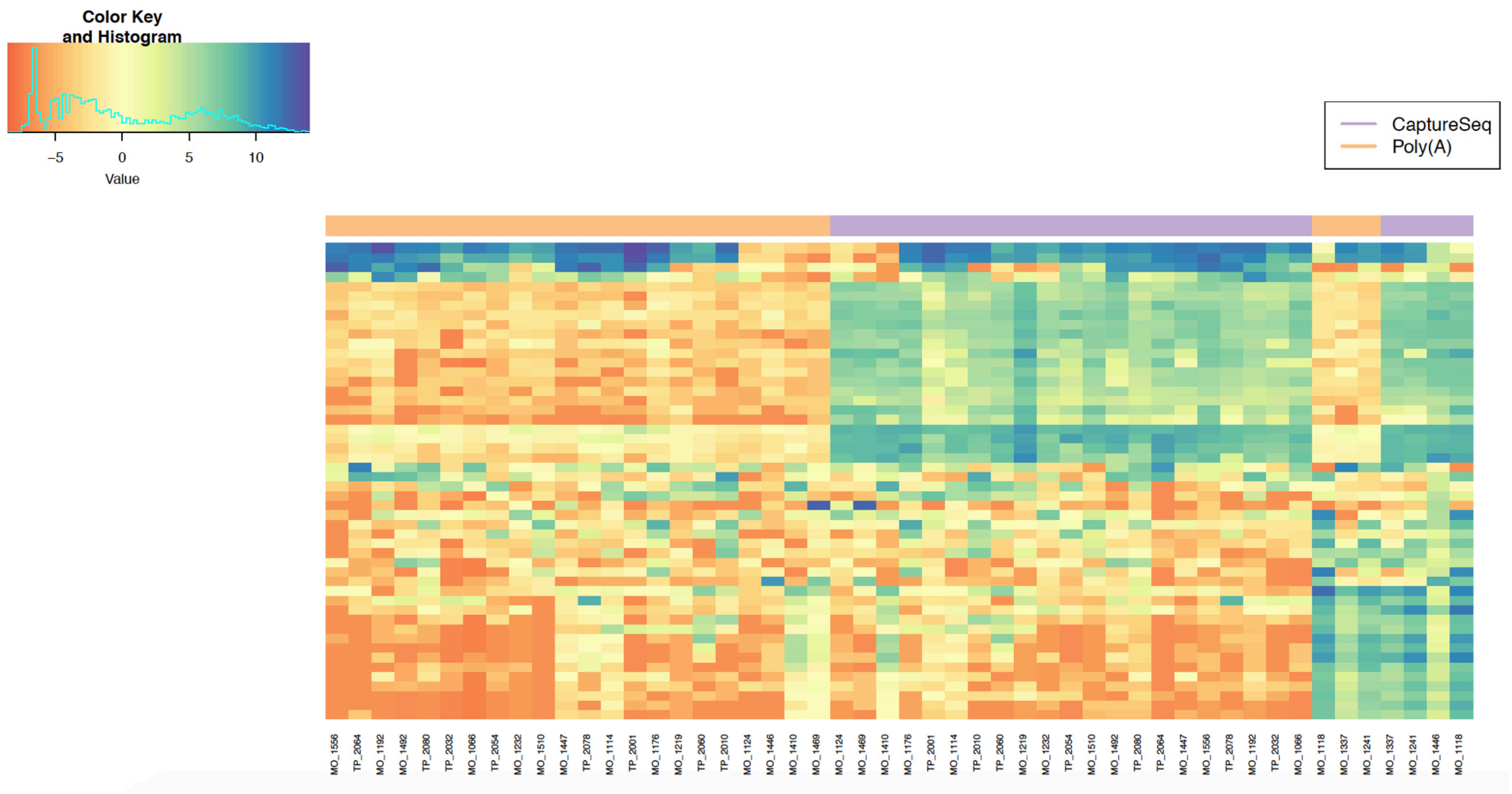

Figure 3.4: Cluster analysis on the paired CaptureSeq and Poly(A) log2(CPM) measurements from 25 prostate cancer patients. Each row represent a gene and each column represent either a library of CaptureSeq ( columns with purple tags on the top) or Poly(A) (columns with orange tags on the top) measurements. The genes are the top 50 most varied genes across the 50 libraries. We used the Euclidean distance and the complete method of Hierarchical clustering on the genes.

Table 3.2: **Nonzero coefficients of the penalized regression for classifying samples from the two technologies.** We tuned the Elastic Net regression model using a sequence of $\alpha$ (from 0 to 1, with a step of 0.1), which determined the proportion of quadratic and L1 norm in the penalty term, and 10-fold cross-validation to choose $\lambda$. We trained the model on 272 pairs of measurements and then tested the model on the other 100 pairs. The misclassification rate on the test set is 0.005.

| Coefficient / Gene Name | Coefficient Value |
|---|---|
| Intercept | 0.17948 |
| ENSG00000120948 | 0.00017 |
| ENSG00000115760 | -0.00091 |
| ENSG00000165119 | 0.00048 |
| ENSG00000197714 | -0.00042 |

Table 3.3: **Differential expression experiments of randomly drawn subjects.** Number of genes found to be significantly differentially expressed for more than 50 times in a total of 1000 experiments.

| Poly(A) v.s. Poly(A) | Poly(A) v.s. capture sequencing | Poly(A) v.s. predicted Poly(A) |
|---|---|---|
| 0 | 830 | 0 |

measurements and their paired capture sequencing measurements. The results suggest that without correction, we can have on average more than 600 falsely positive genes in each experiment and that there are about 830 genes tend to be erroneously recognized.

## 3.4 Converting capture sequencing measurements to Poly(A) measurements

As there are various differences between the two types of measurements, using these two types of measures as if they were based on the same purification protocol could introduce biases. One solution is to convert one type of measurement to the other before combining these two types of measures into the analyses. There are several factors assumed to influence the conversion, including the degree of RNA degradation, gene length, GC content, and the length of overlapping sequence between the gene and the probe. However, when these factors were incorporated, the prediction accuracy was not better. This is possibly caused by measurement errors in

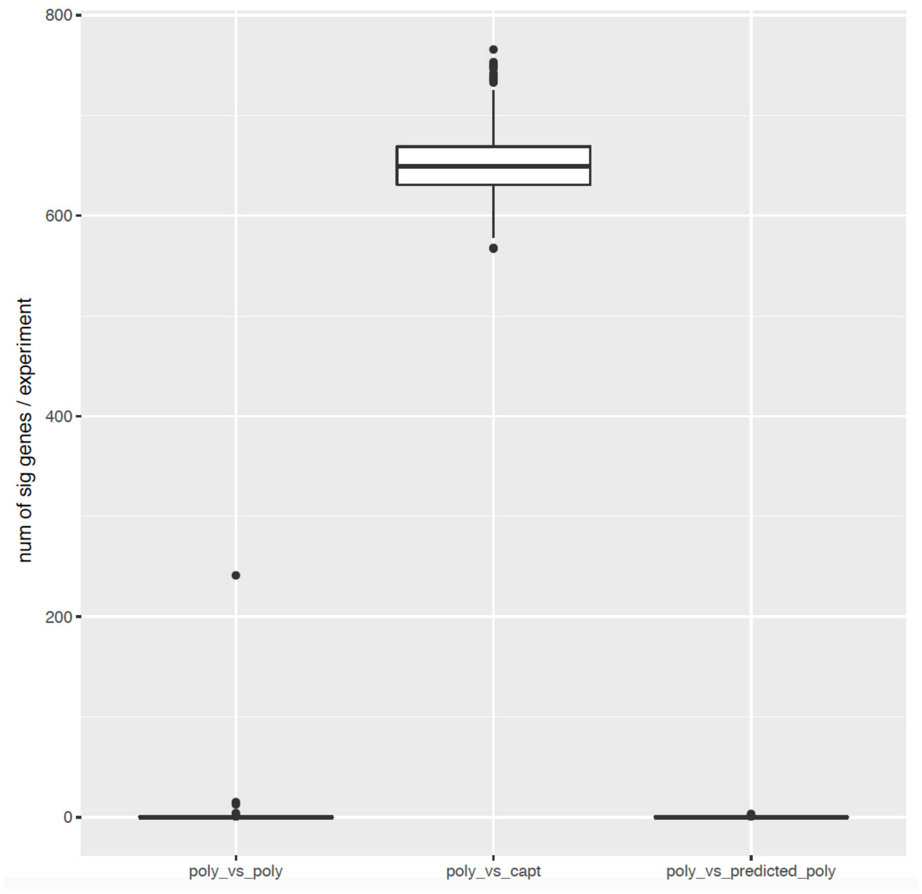

Figure 3.5: Box plots of the number of significant genes in each experiment comparing two groups of randomly drawn subjects

the covariates. For example, RINs with values below 7 do not correlate well with true RNA degradation levels. In addition, in practice, researchers only perform analyses on Poly(A) RNA-Seq samples when the RINs are greater than 7 for quality control. Finally, the simplest model with only capture sequencing measurements as covariates provides a high correlation between the actual and predicted values. Therefore, we use only samples with RIN greater than 7 and single covariate to capture sequencing measurements in the subsequent discussion.

### 3.4.1 Notation

We denote the two types of measurements as the following:

$Y_{gi}$ : poly(A) log2(CPM)$_{gi}$

$X_{gi}$ : capture sequencing log2(CPM)$_{gi}$

$g$: index for genes, $g$=1,...,m

$i$: index of subjects, $i$=1,...,$n_g$

### 3.4.2 Comparing prediction by genewise simple regression and mixed effect model

With only the capture sequencing measurements as the covariate, the gene-wise simple regression model that convert the capture sequencing measurements to poly(A) measurements is:

$$Y_{gi} = \beta_{0g} + \beta_{1g} X_{gi} + \epsilon_{gi} \tag{3.1}$$

where $\epsilon_{ig}$ is the measurement error $\sim N(0, \sigma^2)$. Adopting this approach, we will have $2m + 1$ parameters to be estimated. On the other hand, we could also assume a common distribution of the gene-wise coefficients using the mixed model:

$$Y_{gi} = (\beta_0 + b_{0g}) + (\beta_1 + b_{1g})X_{gi} + \epsilon_{gi} \tag{3.2}$$

where $\beta_0, \beta_1$ are the fixed effects; $b_{0g}$ and $b_{1g}$ are the gene-specific random effects with $(b_{0g}, b_{1g})^T$ following the bivariate normal distribution $(0, \Sigma)$ and $\epsilon_{ig}$ is the measurement error following $N(0, \sigma^2)$. In this case, the number of parameters is reduced from $2m + 1$ to 6 .

## 3.5 Results

We compare the correlation, and the root mean squared errors (RMSEs) between true Poly(A) measurements and predicted Poly(A) measurements using both the simple regression and the mixed model. We also plot the correlation and RMSEs against different sample sizes to reflect the level of efficiency. The results are summarized in Figure 3.6 and Figure 3.7, which show that the linear mixed model maintains high accuracy in the predicted values even when the sample size per gene is small, while ordinary least squares requires at least 10 subjects per gene to achieve similar accuracy and stabilize after there are more than 30 subjects per gene.

As it is not clear whether high correlations and low RMSEs translate to low biases in actual analyses, we further perform clustering on the true and predicted Poly(A) measurements. The result is shown in Figure 3.8. As opposed to Figure 3.4, which group by protocol, now the grouping of the columns is by subject, suggesting the elimination of differences caused by different purification protocols.

Lastly, we examine whether the correction using the linear mixed model eliminates false positive findings in differential expression analysis. Again, we randomly draw Poly(A) measurements from 25 out 50 prostate cancer patients and compare these measurements with their predicated Poly(A) measurements. The results are

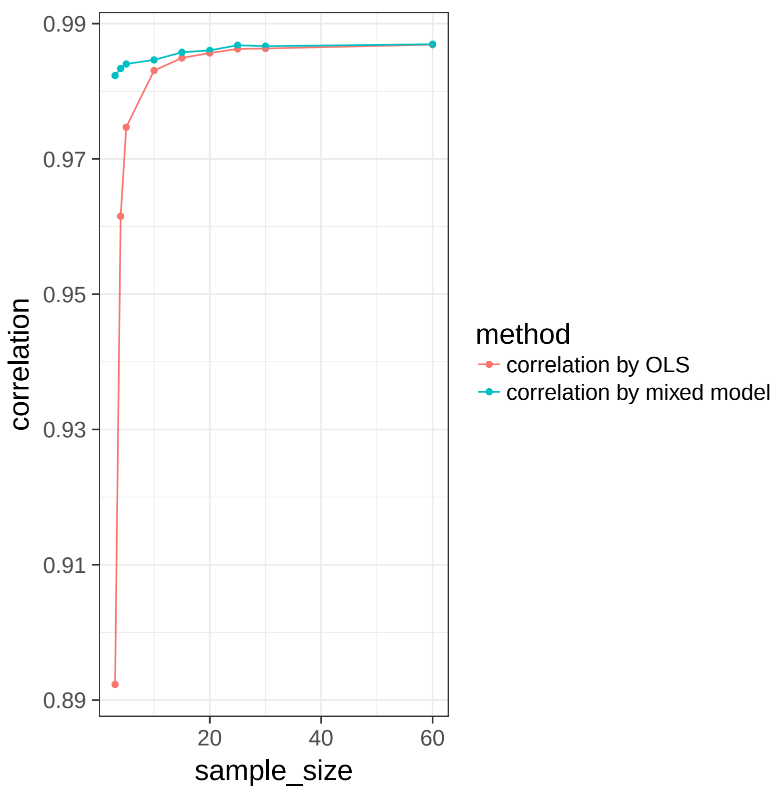

Figure 3.6: Prediction by the mixed model and genewise fixed effect models on libraries with RIN ≥ 7: correlation of predicted and actual poly(A) log2-CPMs across all genes and all samples, based on 30 replicates for each sample size.

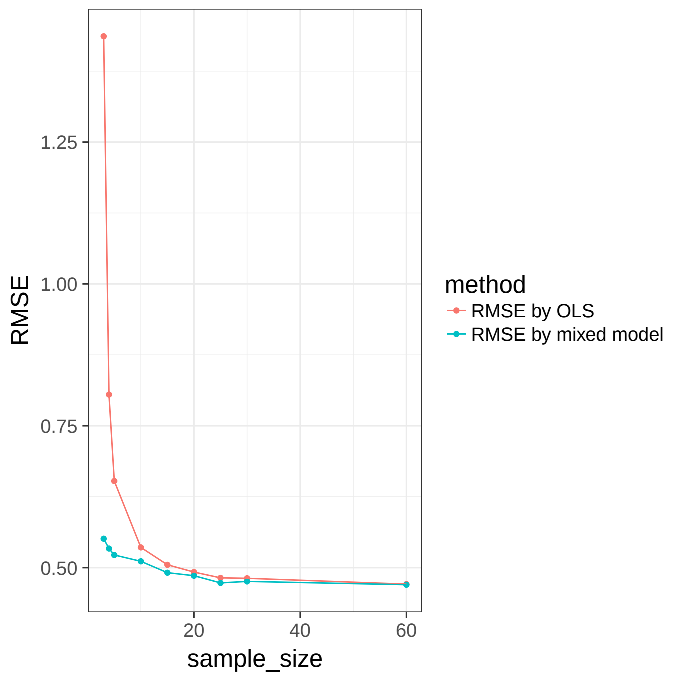

Figure 3.7: Prediction by the mixed model and genewise fixed effect models on libraries with RIN ≥ 7: RMSE of predicted and actual poly(A) log2-CPMs across all genes and all samples, based on 30 replicates for each sample size.

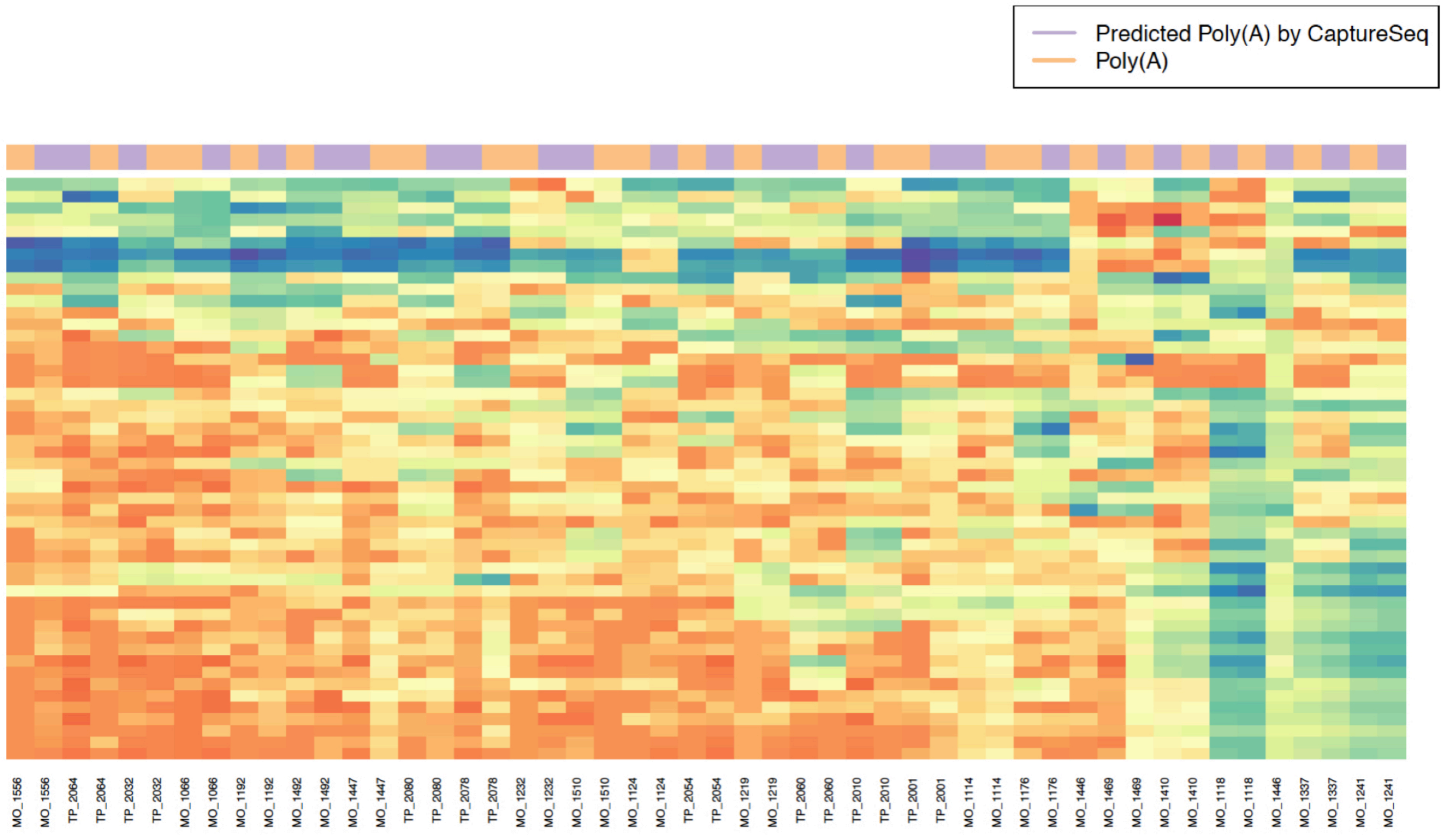

Figure 3.8: Cluster analysis on the paired predicted Poly(A) measurements based on capture sequencing measurements and true Poly(A) log2(CPM) measurements from 25 prostate cancer patients. Each row represents a gene, and each column represent either a predicted Poly(A) library ( columns with purple tags on the top) or true Poly(A) (columns with orange tags on the top) measurements. The genes are the top 50 most varied genes across the 50 libraries. We used the Euclidean distance and the complete method of Hierarchical clustering on the genes. The clustering is now by subject, which is suggested by the neighboring of same subject identification numbers.

also summarized in Figure 3.5 and Table 3.3 and numbers of false positive findings is reduced to those based on comparing similar patients with the same types of measurements.

## 3.6 Discussion

In this project, we first demonstrate that there are systemic differences in RNA-seq measurements based on different purification protocols and in order to avoid false positive finding and biases in analyses, it is necessary to make a conversion before combining these types of measurements. Secondly, although the transformed RNA-seq measurement $\log_2 CPM$ has its variance inversely proportional to the mean expression level [28], which is also shown in our data (Appendix, Figure 3.12), we found that the linear mixed model is effective and efficient in making the conversion and eliminate the technical differences between the two types of measurements.

However, when examining the coefficients of the gene-wise regressions, we found that quite a few of them do not conform to the assumed normal distribution, which is demonstrated in Figure 3.9. Besides, when examining the observations for a gene whose gene-wise coefficients are outlying, we also noted outliers in individual regression as shown in Figure 3.10. These outliers, either on the coefficient level or observation level, were later found to be caused by various technical issues such as dysfunction probes, batch effects, and erroneously including histone mRNA and mitochondrial mRNA. For the accuracy of the analyses, these outliers should be identified and the underlying causes should be corrected if possible. Therefore, in the next project, we develop an outlier detection algorithm that gives a robust estimation for the linear mixed model.

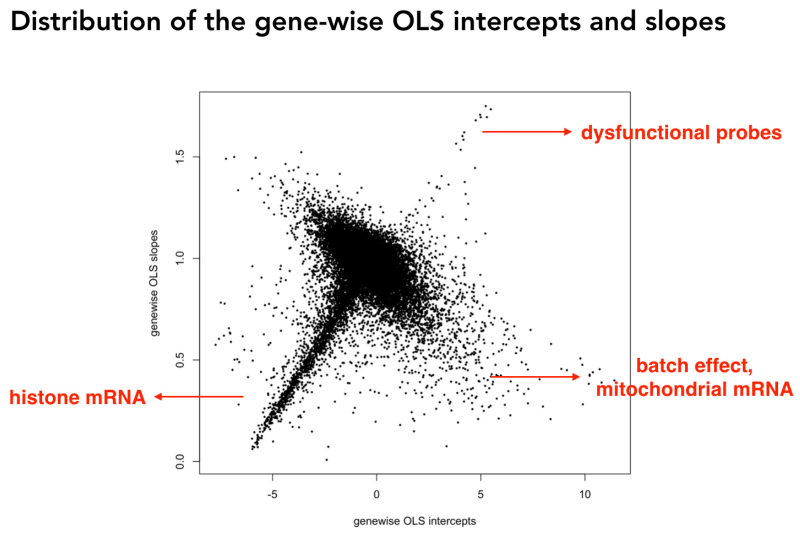

Figure 3.9: Distribution of the gene-wise ordinary least squares (OLS) estimates by having the paired the Exome Capture RNA-Seq measurements regressed on the Poly(A) Capture RNA-Seq measurements. The arrows indicate identified causes for some of the gene-specific random effects to be outliers of the assumed bivariate normal distribution.

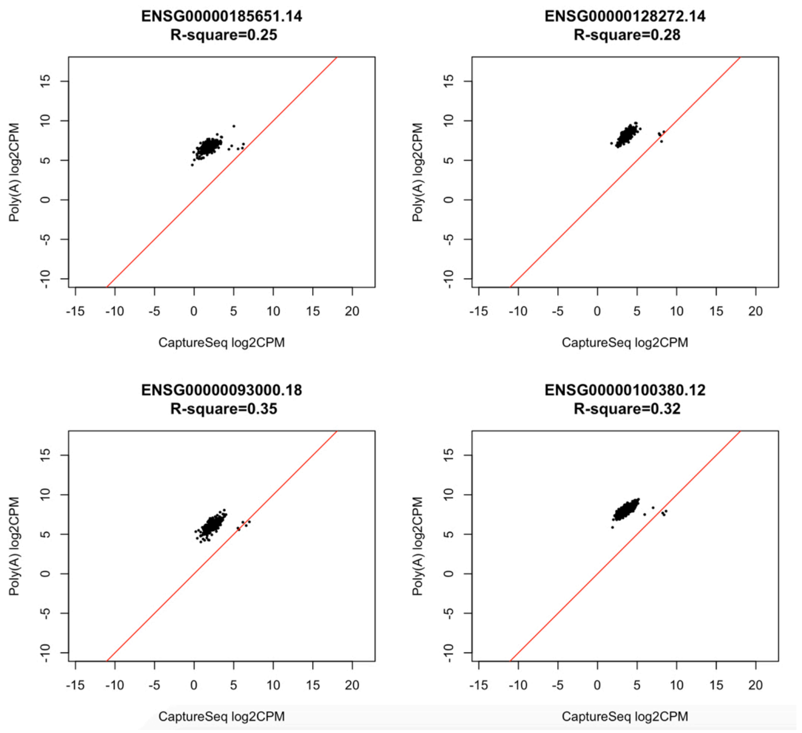

Figure 3.10: Examples of genes whose OLS estimates lie in the right lower quadrant of in Figure 3.9. The smaller clusters of observations shared the same sample identifiers across multiple genes, which suggests the existence of batch effects.

## 3.7 Appendix for Chapter 3

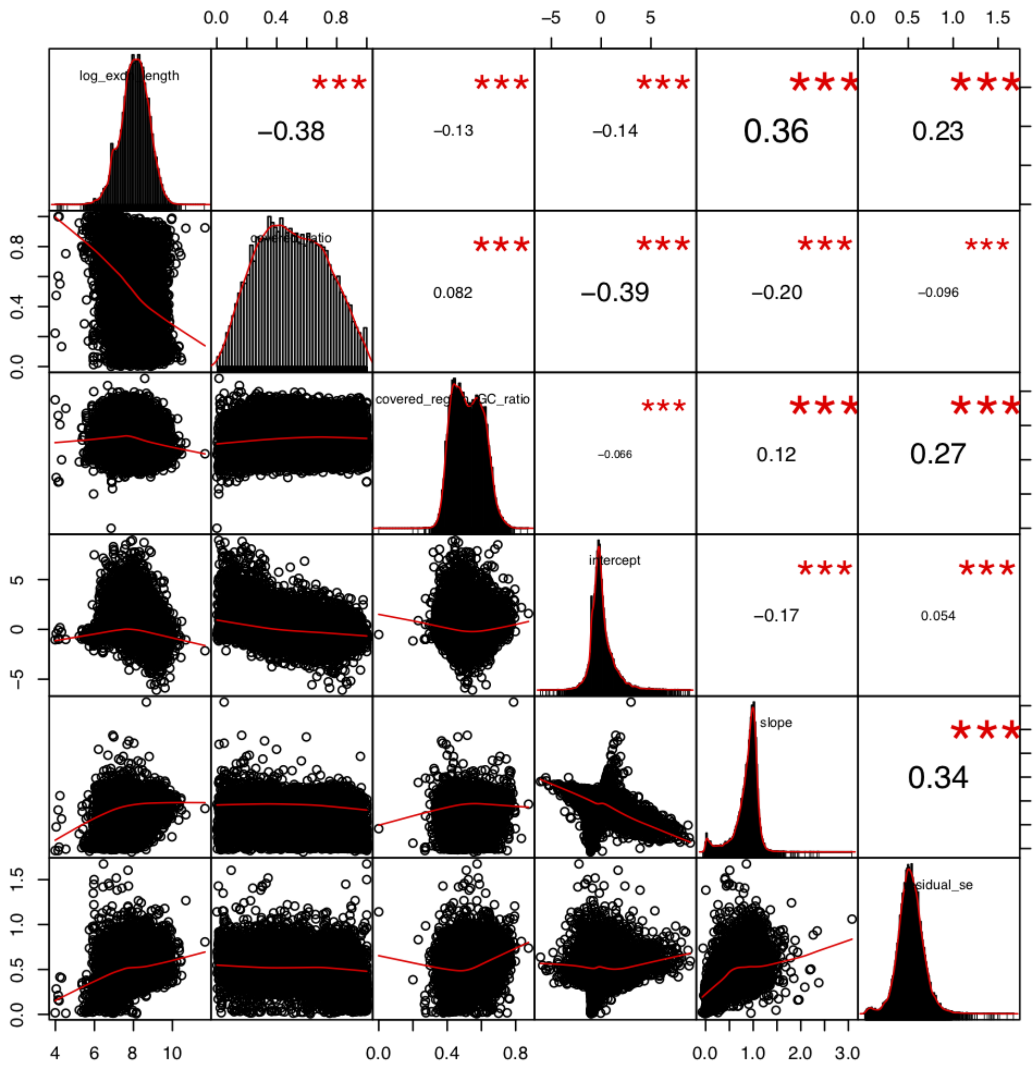

Figure 3.11: The distribution of and the correlations between the total length of exons in a gene, the proportion of the bases in the exons that are either "G" or "C", and the proportion of the length that is covered by the first generation probes of CaptureSeq designed by our collaborative biotech company, the intercepts, the slopes and the residual standard error of the genewise simple regressions by regressing Poly(A) log2(CPM) measurements on CaptureSeq log2(CPM) measurements.

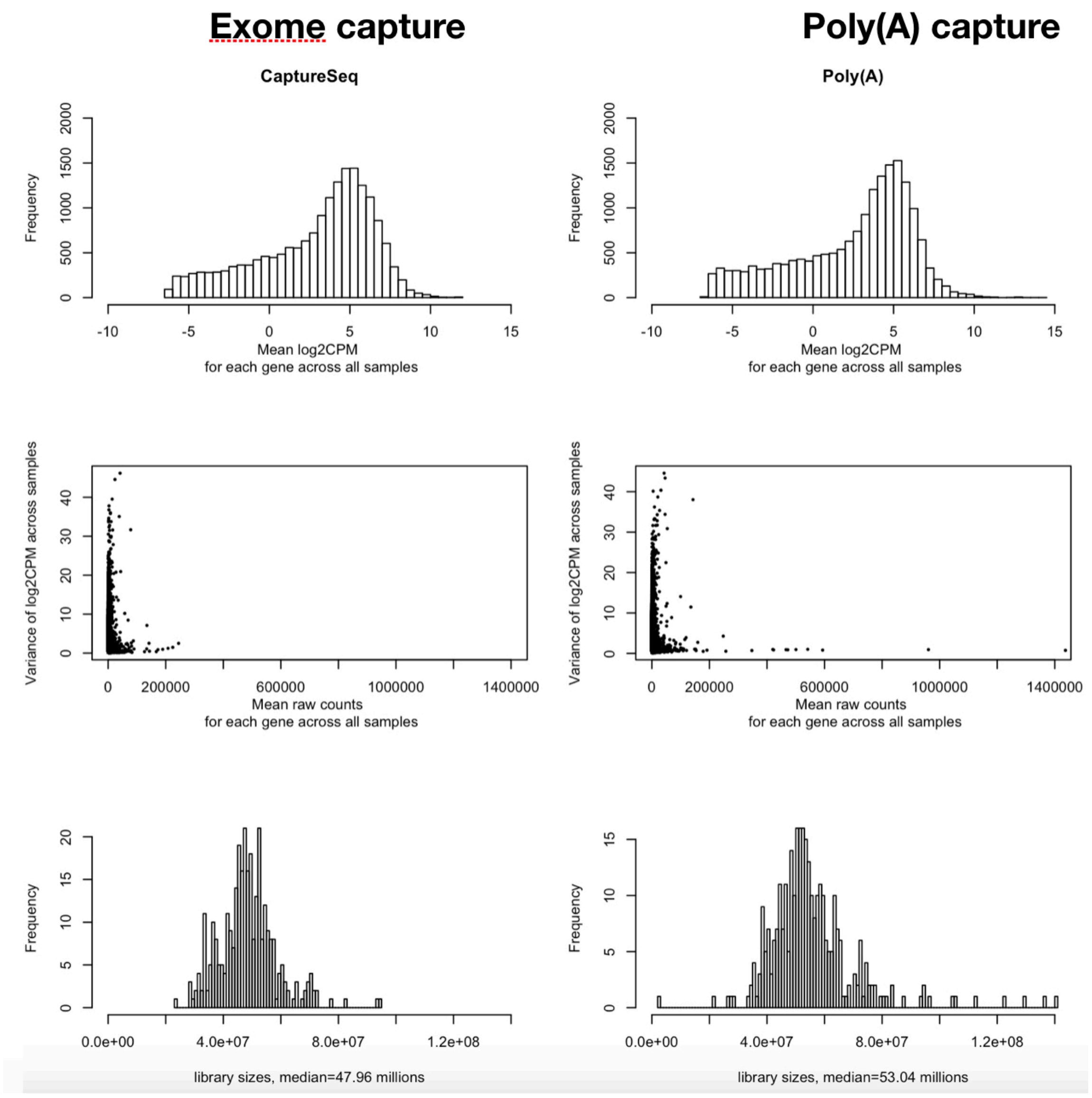

Figure 3.12: Distributions of the Exome Capture RNA-Seq and the Poly(A) Capture RNA-Seq measurements. The first row shows the distribution of mean log2 count per million (CPM) for each gene. The second row displays the inverse relationship between the variance of log2CPM and the mean raw count. The third row illustrates the libraries sizes for all samples, which are measured by both Exome Capture and Poly(A) Capture methods.

**CHAPTER IV**

# Outlier Detection for Mixed Model

## 4.1 Extending mixed effect model for detecting individual outliers and outlying random effects

She and Owen (2011) transformed the problem of outlier detection to variable selection in linear regression by using the mean shift model by Gannaz (2006) and McCann and Welsch (2007), which introduces a variable representing the amount deviating from the predicted mean for each observation, and selecting the outliers by using $L_0$ penalty on nonzero variables. Their approach showed reduced masking and swamping probabilities compared to other robust regression methods, including the MM-estimator, Gervini-Yohai's fully efficient one-step procedure (2002), the compound estimator S1S and least trimmed squares (LTS) (Rousseeuw and Leroy 1987). In the first project, we showed that improved optimization would improve the performance of the model and further reduce the masking and swamping rate of the analysis. In this project, we combine the merits of She and Owen's approach and our proposed optimization to solve the outlier detection problem in a linear mixed model.

There are two levels of outliers in a linear mixed model: outliers of individual observation and outliers of the random effects. Let $g$ denote the index for groups, g=1,...,m and $i$ denote the index for observations in each group, i=1,...,$n_g$. For each observation $y_{gi}$, we create a variable $\delta_{gi}$ to represent the amount of deviation from the predicted mean. In other words, if $y_{gi}$ conforms to the distribution specified by the model, then $\delta_{gi} = 0$; otherwise, $\delta_{gi} \neq 0$. Also, for each $(q \times 1)$ vector of random effects $\boldsymbol{b_g}$, we create a $(q \times 1)$ vector $\boldsymbol{\Delta_g}$ to represent the amount of deviation from the predicted random effect. That is, we will have $\boldsymbol{\Delta_g} \neq \boldsymbol{0}$ if $\boldsymbol{b_g}$ is an outlier for distribution of the random effects and $\boldsymbol{\Delta_g} = \boldsymbol{0}$ if $\boldsymbol{b_g}$ is not outlying. This set up then gives us a mean shift model in the linear mixed effects context:

$$\boldsymbol{y_g} = \boldsymbol{X_g \beta} + \boldsymbol{Z_g(b_g + \Delta_g)} + \boldsymbol{\delta_g} + \boldsymbol{\epsilon_g}, \quad g = 1, ..., m \tag{4.1}$$

$$\text{For each observation } y_{gi}, \quad \begin{cases} \delta_{gi} = 0, & \text{if } y_{gi} \text{ is a conformer} \\ \delta_{gi} \neq 0, & \text{if } y_{gi} \text{ is an outlier.} \end{cases} \tag{4.2}$$

$$\text{For each random effect } b_g + \Delta_g, \quad \begin{cases} \Delta_g = 0, & \text{if } b_g + \Delta_g \text{ is a conformer} \\ \Delta_g \neq 0, & \text{if } b_g + \Delta_g \text{ is an outlier.} \end{cases} \tag{4.3}$$

We follow the notation in Nobre and Singer (2007): $\boldsymbol{y_g}$ is a $(n_g \times 1)$ vector of responses, $\boldsymbol{\beta}$ is a $(p \times 1)$ vector of fixed effects, $\boldsymbol{X_g}$ and $\boldsymbol{Z_g}$ are $(n_g \times p)$ and $(n_g \times q)$ covariate matrices for the fixed effects and the random effects, respectively. $\boldsymbol{\delta_g} = (\delta_{g1}, ...\delta_{gn_g})^T$. $\boldsymbol{\epsilon_g}$ is a $(n_g \times 1)$ vector of measurement errors. In this paper, we assume that $\boldsymbol{b_1, ..., b_m}$ are independent and identically distributed following $N_q(0, \sigma^2 G)$ and $\boldsymbol{\epsilon_g}$ follows $N_{n_g}(0, \sigma^2 R_g)$. $\boldsymbol{b_g}$ and $\boldsymbol{\epsilon_g}$ are independent. $\boldsymbol{G}$ and $\boldsymbol{R_g}$ are positive definite matrices with dimensions $(q \times q)$ and $n_g \times n_g$, respectively.

To facilitate derivation, we re-write (4.1) as the following:

$$\boldsymbol{y} = \boldsymbol{X\beta} + \boldsymbol{Z}(\boldsymbol{b} + \boldsymbol{\Delta}) + \boldsymbol{\delta} + \boldsymbol{\epsilon} \tag{4.4}$$

Let $n = \sum_{g=1}^{m} n_g$, then in the above expression, $\boldsymbol{y}, \boldsymbol{\delta}$ and $\boldsymbol{\epsilon}$ are $(n \times 1)$ vectors with $\boldsymbol{y} = (\boldsymbol{y_1^T}, ..., \boldsymbol{y_m^T})^{\boldsymbol{T}}, \boldsymbol{\delta} = (\boldsymbol{\delta_1^T}, ..., \boldsymbol{\delta_m^T})^{\boldsymbol{T}}$, and $\boldsymbol{\epsilon} = (\boldsymbol{\epsilon_1^T}, ..., \boldsymbol{\epsilon_m^T})^{\boldsymbol{T}}$. $\boldsymbol{X}$ is a $(n \times p)$ matrix with $\boldsymbol{X} = (\boldsymbol{X_1^T}, ..., \boldsymbol{X_m^T})^{\boldsymbol{T}}$. $\boldsymbol{b}$ and $\boldsymbol{\Delta}$ are $(qm \times 1)$ vectors with $\boldsymbol{b} = (\boldsymbol{b_1^T}, ..., \boldsymbol{b_m^T})^{\boldsymbol{T}}$ and $\boldsymbol{\Delta} = (\boldsymbol{\Delta_1^T}, ..., \boldsymbol{\Delta_m^T})^{\boldsymbol{T}}$. Finally, $\boldsymbol{Z}$ is a $(n \times qm)$ matrix, which is the direct sum of all the $\boldsymbol{Z_g}$'s, denoted as $\boldsymbol{Z} = \bigoplus_{g=1}^{m} \boldsymbol{Z_g}$. In other words, $\boldsymbol{Z}$ is a block diagonal matrix with the blocks as the $\boldsymbol{Z_g}$'s.

## 4.2 Estimation Scheme

In the above model, the parameters are the fixed effect $\boldsymbol{\beta}$, deviations of the random effects from the specified distributions $\boldsymbol{\Delta}$, deviations of individual observations, $\boldsymbol{\delta}$, the covariance matrix for the distribution of the random effects $\sigma^2\boldsymbol{G}$ , and measurement error variation $\sigma^2\boldsymbol{R_g}, g = 1, ..., m$. As there are $N$ elements in $\boldsymbol{\delta}$ and $qm$ elements in $\boldsymbol{\Delta}$, the model is over-parameterized. We use a Penalized Maximum Likelihood Estimation to force sparsity of the nonzero outlier variables. As precise identification of outliers requires precise specification of the variances and vice versa, the algorithm uses group-wise robust regressions to form initial estimates of the variances. Because of a large number of parameters in $b, \Delta$, and $\delta$, we set the estimation into two parts: one part for $b, \Delta$ and $\delta$, where we separate the parameters by groups and estimate them by a parallel algorithm. The other part is for parameters shared by all groups of observations, which are the fixed effects $\boldsymbol{\beta}$, the random effect variances $\sigma^2\boldsymbol{G}$ and error variances $\sigma^2\boldsymbol{R_g}$. We iterate between these two parts until convergence, which is similar to the block-coordinate descent method. The following steps outline the scheme for model estimation:

1. Set a thresholding probability $\alpha$ for outlier detection such that when an observation has a probability less than $\alpha$ of coming from the estimated distribution, it will be identified as an outlier. Therefore, $\alpha$ will also be the false positive (swamping) rate of outlier detection.

2. Use group-wise robust regressions to form initial estimates of the random effects $\boldsymbol{b_g}$ and $\sigma^2\boldsymbol{R_g}, g = 1, ..., m$. In particular, we use the R function "lmRob" from the package "robust". The function generates initial regression estimates through S-estimation and then produces the final estimates by MM estimates. The error standard deviation common for all groups is estimated by the median of the residuals from group-wise robust regressions.

3. Based on the initial estimates of the random effects, we construct initial estimates of the covariance matrix $\sigma^2\boldsymbol{G}$ and the center of the random effects $\boldsymbol{\beta}$ with a robust multivariate method. In particular, we use the R function "covRob" from the package "robust" to estimate the location and scale of the multivariate distribution of the random effects.

4. Estimate $\boldsymbol{\Delta}$ and $\boldsymbol{\delta}$ by using Penalized Maximum Likelihood. None-zero $\boldsymbol{\Delta}$ and $\boldsymbol{\delta}$ indicate the outlying random effects and observations.

5. Estimate the fixed effects $\boldsymbol{\beta}$, the random effect variances $\sigma^2\boldsymbol{G}$ and error variances $\sigma^2\boldsymbol{R_g}$ using the 'lme4: Linear Mixed-Effects Models using "Eigen" and S4' package based on the non-outlying data. As these filtered observations follow truncated normal distribution, the variances obtained from 'lme4' have to be multiplied by dilation factors to yield the true variances. The dilation factor will be derived later.

6. Repeat step 4 and 5 until the estimates of $\boldsymbol{\beta}, \sigma, \boldsymbol{G}$, and $\boldsymbol{R_g}$ converge.

The estimation scheme is illustrated in Figure 4.1. For simplicity, we assume that

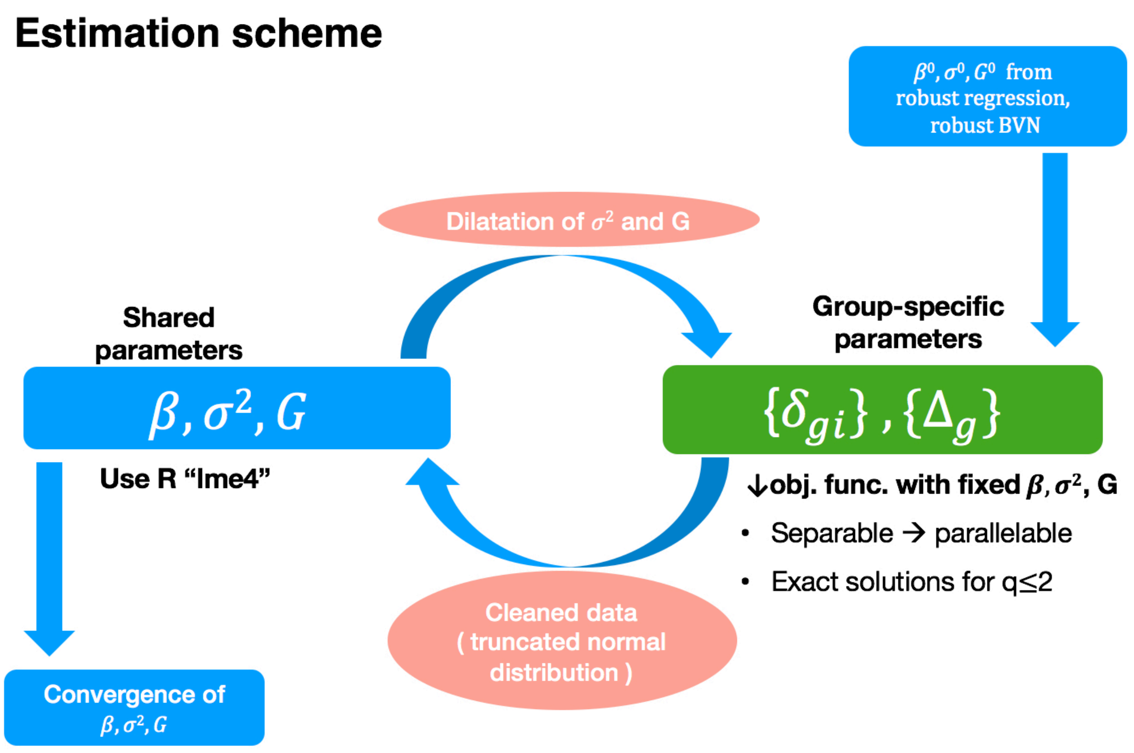
Estimation scheme
$\beta^0, \sigma^0, G^0$ from robust regression, robust BVN
Dilatation of $\sigma^2$ and G
Shared parameters
$\beta, \sigma^2, G$
Use R "lme4"
Group-specific parameters
$\{\delta_{gi}\}, \{\Delta_g\}$
↓obj. func. with fixed $\beta, \sigma^2$, G
• Separable → parallelable
• Exact solutions for q≤2
Cleaned data ( truncated normal distribution )
Convergence of $\beta, \sigma^2, G$

Figure 4.1: Estimation scheme of the proposed method.

the observations within a group (gene) are independent and have equal variances $(\boldsymbol{R_g} = \boldsymbol{I_{ng}})$ in subsequent sections.

### 4.3 Estimate the outliers and predict the random effects by Penalized Maximum Likelihood Estimation

Based on the model in (4.4), we have

$$\begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{b} \end{bmatrix} = \begin{bmatrix} \boldsymbol{X\beta} + \boldsymbol{Z\Delta} + \boldsymbol{\delta} \\ 0 \end{bmatrix} + \begin{bmatrix} \boldsymbol{Z} & \boldsymbol{I} \\ \boldsymbol{I} & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{b} \\ \boldsymbol{\epsilon} \end{bmatrix}$$

and

$$\begin{bmatrix} \boldsymbol{b} \\ \boldsymbol{\epsilon} \end{bmatrix} \sim N_{qm+n}( \begin{bmatrix} 0_{qm} \\ 0_n \end{bmatrix}, \begin{bmatrix} \sigma^2 \boldsymbol{D} & 0_{qm\times n} \\ 0_{n\times qm} & \sigma^2 \boldsymbol{\Sigma_\epsilon}, \end{bmatrix})$$

In the above expression $D = I_m \otimes G$ and $\Sigma_\epsilon = \oplus_{g=1}^m R_g$, where $\otimes$ is the Kronecker product of matrices. In other words, $D$ is a block diagonal matrix with each block being $G$.

The joint distribution of $\boldsymbol{y}$ and $\boldsymbol{b}$ therefore is

$$\begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{b} \end{bmatrix} \sim N_{n+qm}( \begin{bmatrix} \boldsymbol{X\beta} + \boldsymbol{Z\Delta} + \boldsymbol{\delta} \\ \mathbf{0}_{qm} \end{bmatrix}, \sigma^2 \begin{bmatrix} \boldsymbol{ZDZ}^T + \boldsymbol{\Sigma_\epsilon} & \boldsymbol{ZD} \\ \boldsymbol{DZ}^T & \boldsymbol{D} \end{bmatrix}) \tag{4.5}$$

If we write $(\boldsymbol{ZDZ^T} + \boldsymbol{\Sigma_\epsilon})$ as $\boldsymbol{\Sigma_y}$, we have the -2 log-likelihood of the observations as:

$$-2\log f(\boldsymbol{y}) = \log|2\pi\sigma^2\boldsymbol{\Sigma_y}| + (\boldsymbol{y} - \boldsymbol{X\beta} - \boldsymbol{Z\Delta} - \boldsymbol{\delta})^T\boldsymbol{\Sigma_y}^{-1}(\boldsymbol{y} - \boldsymbol{X\beta} - \boldsymbol{Z\Delta} - \boldsymbol{\delta})/\sigma^2 \tag{4.6}$$

Because the model is overparameterized, we penalize the nonzero elements of $\boldsymbol{\Delta}$ and $\boldsymbol{\delta}$ to make the estimation feasible. The reason that we use $L_0$ penalties instead of the $L_1$ penalties is because the former gives lower masking and swamping rates compared with the latter [45]. The penalized log-likelihood function then becomes:

$$\begin{aligned} -2\log f(\boldsymbol{y}) = & \log|2\pi\sigma^2\boldsymbol{\Sigma_y}| + (\boldsymbol{y} - \boldsymbol{X\beta} - \boldsymbol{Z\Delta} - \boldsymbol{\delta})^T\boldsymbol{\Sigma_y}^{-1}(\boldsymbol{y} - \boldsymbol{X\beta} - \boldsymbol{Z\Delta} - \boldsymbol{\delta})/\sigma^2 + \\ & \lambda_O\|\boldsymbol{\delta}\|_0 + \lambda_G \sum_g \mathbb{1}_{\boldsymbol{\Delta_g} \neq \mathbf{0}} \end{aligned} \tag{4.7}$$

The primary challenge to minimize (4.7) is that it is non-convex. However, we will show that if the objective function can be separated by the groups defined in the mixed model and be transformed to a sum of truncated quadratic functions, then we can apply the optimization method proposed in the second chapter. In the next section, we explain how we separated and transformed the objective function by incorporating the random effect predictors into the penalized likelihood function.

### 4.3.1 Separation of the objective function by incorporating the predictors of the random effects

According to (4.5) and the conditional distribution of multivariate normal vectors, the best unbiased linear predictors (BLUPs) for the random effects, $E(\boldsymbol{b} \mid \boldsymbol{y})$, (denoted as $\tilde{\boldsymbol{b}}$), is:

$$\tilde{\boldsymbol{b}} \equiv E(\boldsymbol{b} \mid \boldsymbol{y}) = \boldsymbol{D}\boldsymbol{Z}^T\Sigma_{\boldsymbol{y}}^{-1}(\boldsymbol{y} - \boldsymbol{X\beta} - \boldsymbol{Z\Delta} - \boldsymbol{\delta}) \tag{4.8}$$

Note that

$$\begin{aligned}\boldsymbol{Z}\tilde{\boldsymbol{b}} &= \boldsymbol{ZDZ}^T\Sigma_{\boldsymbol{y}}^{-1}(\boldsymbol{y} - \boldsymbol{X\beta} - \boldsymbol{Z\Delta} - \boldsymbol{\delta})\\ &= (\boldsymbol{\Sigma}_{\boldsymbol{y}} - \Sigma_{\boldsymbol{\epsilon}})\Sigma_{\boldsymbol{y}}^{-1}(\boldsymbol{y} - \boldsymbol{X\beta} - \boldsymbol{Z\Delta} - \boldsymbol{\delta})\end{aligned}$$

We thus have

$$\boldsymbol{y} - \boldsymbol{X\beta} - \boldsymbol{Z}(\tilde{\boldsymbol{b}} + \boldsymbol{\Delta}) - \boldsymbol{\delta} = \Sigma_{\epsilon}\Sigma_{\boldsymbol{y}}^{-1}(\boldsymbol{y} - \boldsymbol{X\beta} - \boldsymbol{Z\Delta} - \boldsymbol{\delta})$$

Therefore,

$$\begin{aligned}&[\boldsymbol{y} - \boldsymbol{X\beta} - \boldsymbol{Z}(\tilde{\boldsymbol{b}} + \boldsymbol{\Delta}) - \boldsymbol{\delta}]^T\Sigma_{\epsilon}^{-1}[\boldsymbol{y} - \boldsymbol{X\beta} - \boldsymbol{Z}(\tilde{\boldsymbol{b}} + \boldsymbol{\Delta}) - \boldsymbol{\delta}]\\ &= [\boldsymbol{y} - \boldsymbol{X\beta} - \boldsymbol{Z}(\tilde{\boldsymbol{b}} + \boldsymbol{\Delta}) - \boldsymbol{\delta}]^T\Sigma_{\boldsymbol{y}}^{-1}(\boldsymbol{y} - \boldsymbol{X\beta} - \boldsymbol{Z\Delta} - \boldsymbol{\delta})\\ &= (\boldsymbol{y} - \boldsymbol{X\beta} - \boldsymbol{Z\Delta} - \boldsymbol{\delta})^T\Sigma_{\boldsymbol{y}}^{-1}(\boldsymbol{y} - \boldsymbol{X\beta} - \boldsymbol{Z\Delta} - \boldsymbol{\delta}) - (\boldsymbol{Z}\tilde{\boldsymbol{b}})^T\Sigma_{\boldsymbol{y}}^{-1}(\boldsymbol{y} - \boldsymbol{X\beta} - \boldsymbol{Z\Delta} - \boldsymbol{\delta})\\ &= (\boldsymbol{y} - \boldsymbol{X\beta} - \boldsymbol{Z\Delta} - \boldsymbol{\delta})^T\Sigma_{\boldsymbol{y}}^{-1}(\boldsymbol{y} - \boldsymbol{X\beta} - \boldsymbol{Z\Delta} - \boldsymbol{\delta}) - \tilde{\boldsymbol{b}}^T\boldsymbol{D}^{-1}\tilde{\boldsymbol{b}}\end{aligned}$$

The penalized likelihood in (4.7) therefore can be expressed in terms of the parameters and the predictor of the random effects.

(4.9)

$$\begin{aligned}-2\log f(\boldsymbol{y}) =& \log|2\pi\sigma^2\boldsymbol{\Sigma}_{\boldsymbol{y}}| + [\boldsymbol{y} - \boldsymbol{X\beta} - \boldsymbol{Z}(\tilde{\boldsymbol{b}} + \boldsymbol{\Delta}) - \boldsymbol{\delta}]^T\Sigma_{\epsilon}^{-1}[\boldsymbol{y} - \boldsymbol{X\beta} - \boldsymbol{Z}(\tilde{\boldsymbol{b}} + \boldsymbol{\Delta}) - \boldsymbol{\delta}]/\sigma^2 +\\ &\tilde{\boldsymbol{b}}^T\boldsymbol{D}^{-1}\tilde{\boldsymbol{b}}/\sigma^2 + \lambda_O\|\boldsymbol{\delta}\|_0 + \lambda_G\sum_g \mathbb{1}_{\boldsymbol{\Delta}_g \neq \boldsymbol{0}}\end{aligned}$$

Because $\Delta, \delta$ and $\tilde{b}$ (which is a function of $\Delta$ and $\delta$) are collection of gene-specific parameters ($\Delta_g, \delta_g$ and $\tilde{b}_g$), we separate the parameters into two parts: separable parameters $\Delta, \delta$ and $\tilde{b}$ and other parameters shared by all groups, which are the fixed effect $\boldsymbol{\beta}, \boldsymbol{D}, \boldsymbol{\Sigma_\epsilon}$ and $\sigma$. Then we iteratively estimate these two groups of parameters in a coordinate descent fashion. That is, we estimate $\boldsymbol{\Delta}, \boldsymbol{\delta}$ and the random effect predictor $\tilde{\boldsymbol{b}}$ while holding $\boldsymbol{\beta}, \boldsymbol{D}, \boldsymbol{\Sigma_\epsilon}$ and $\sigma$ as constants and then vice versa. The benefit of doing so is that we can estimate the group-specific parameters in parallel and reduce computation time linearly.

Therefore when we estimate $\Delta, \delta$ and $\tilde{b}$, the first term in (4.9) can be dropped. Further, as $\boldsymbol{D}$ and $\boldsymbol{\Sigma_\epsilon}$ are block diagonal matrices, the objective function can be separated into independent groups:

(4.10)

$$\begin{aligned} f(\tilde{\boldsymbol{b}}, \boldsymbol{\delta}, \boldsymbol{\Delta}) = &\sum_g \Big\{ [\boldsymbol{y_g} - \boldsymbol{X_g\beta} - \boldsymbol{Z_g}(\tilde{\boldsymbol{b}}_{\boldsymbol{g}} + \boldsymbol{\Delta_g}) - \boldsymbol{\delta_g}]^T \boldsymbol{R}_{\boldsymbol{g}}^{-1} [\boldsymbol{y_g} - \boldsymbol{X_g\beta} - \boldsymbol{Z_g}(\tilde{\boldsymbol{b}}_{\boldsymbol{g}} + \boldsymbol{\Delta_g}) - \boldsymbol{\delta_g}]/\sigma^2 + \\ &\lambda_O \|\boldsymbol{\delta_g}\|_0 + \tilde{\boldsymbol{b}}_{\boldsymbol{g}}^T \boldsymbol{G}^{-1} \tilde{\boldsymbol{b}}_{\boldsymbol{g}}/\sigma^2 + \lambda_G \mathbb{1}_{\boldsymbol{\Delta_g \neq 0}} \Big\} \\ \equiv &\sum_g f_g \end{aligned}$$

Optimizing (4.10) (with $\boldsymbol{D}, \boldsymbol{\Sigma_\epsilon}$ and $\boldsymbol{\beta}$ treated as fixed) is the same as optimizing (4.7) first and then compute $\tilde{\boldsymbol{b}}$ based on the penalized maximum likelihood estimates of $\boldsymbol{\Delta}$ and $\boldsymbol{\delta}$. This is because the predicted random effect $\tilde{\boldsymbol{b}}$ is a function of $\boldsymbol{\Delta}$ and $\boldsymbol{\delta}$, and that using similar argument as the invariance of MLE, the $\hat{\tilde{\boldsymbol{b}}}$ based on the penalized maximum likelihood estimates $\hat{\boldsymbol{\Delta}}$ and $\hat{\boldsymbol{\delta}}$ is also the penalized maximum likelihood estimate.

**Definition 4.1. Induced penalized likelihood function** Let $Q$ be the penalized likelihood function of $\theta$. Given a value $\eta$, define the induced penalized likelihood function $Q^*$ as

$$Q^*(\eta|\boldsymbol{x}) = \sup_{\{\theta:\tau(\theta)=\eta\}} Q(\theta|\boldsymbol{x})$$

**Proposition 4.2.** *If $\hat{\theta}$ is the penalized MLE of $\theta$, then for any function $\tau(\theta)$, the penalized MLE of $\tau(\theta)$ is $\tau(\hat{\theta})$.*

*Proof.*

$$Q^*(\hat{\eta}|\boldsymbol{x}) = \sup_{\eta} Q^*(\eta|\boldsymbol{x}) = \sup_{\eta} \sup_{\{\theta:\tau(\theta)=\eta\}} Q(\theta|\boldsymbol{x}) = \sup_{\theta} Q(\theta|\boldsymbol{x}) = Q(\hat{\theta}|\boldsymbol{x})$$

Meanwhile,

$$\begin{aligned} Q(\hat{\theta}|\boldsymbol{x}) = &\sup_{\{\theta:\tau(\theta)=\tau(\hat{\theta})\}} Q(\theta|\boldsymbol{x}) \\ &= Q^*(\tau(\hat{\theta})|\boldsymbol{x}), \quad \text{by the definition of the induced penalized likelihood function.} \end{aligned}$$

$\square$

### 4.3.2 Transform the objective function into a sum of truncated quadratic functions

Since the objective function is separable, we optimize (4.10) in parallel based on each group of observations and focus on the following function in subsequent discussion:

(4.11)

$$\begin{aligned} f_g(\tilde{\boldsymbol{b}}_{\boldsymbol{g}}, \boldsymbol{\delta}_{\boldsymbol{g}}, \boldsymbol{\Delta}_{\boldsymbol{g}}) =&[\boldsymbol{y}_{\boldsymbol{g}} - \boldsymbol{X}_{\boldsymbol{g}}\boldsymbol{\beta} - \boldsymbol{Z}_{\boldsymbol{g}}(\tilde{\boldsymbol{b}}_{\boldsymbol{g}} + \boldsymbol{\Delta}_{\boldsymbol{g}}) - \boldsymbol{\delta}_{\boldsymbol{g}}]^T \boldsymbol{R}_{\boldsymbol{g}}^{-1} [\boldsymbol{y}_{\boldsymbol{g}} - \boldsymbol{X}_{\boldsymbol{g}}\boldsymbol{\beta} - \boldsymbol{Z}_{\boldsymbol{g}}(\tilde{\boldsymbol{b}}_{\boldsymbol{g}} + \boldsymbol{\Delta}_{\boldsymbol{g}}) - \boldsymbol{\delta}_{\boldsymbol{g}}]/\sigma^2 + \\ &\lambda_O \|\boldsymbol{\delta}_{\boldsymbol{g}}\|_0 + \tilde{\boldsymbol{b}}_{\boldsymbol{g}}^T \boldsymbol{G}^{-1} \tilde{\boldsymbol{b}}_{\boldsymbol{g}}/\sigma^2 + \lambda_G \mathbb{1}_{\boldsymbol{\Delta}_{\boldsymbol{g}} \neq \boldsymbol{0}} \end{aligned}$$

#### 4.3.2.1 Estimate $\Delta_g, \tilde{b}_g$ and $\delta_g$

If $\boldsymbol{\Delta_g} \neq \mathbf{0}$ then $f_g$ is minimized at $\hat{\tilde{\boldsymbol{b}}}_{\boldsymbol{g}} = \mathbf{0}$ and

$$\min f_g = \min_{\boldsymbol{\Delta_g},\boldsymbol{\delta_g}} [\boldsymbol{y_g} - \boldsymbol{X_g\beta} - \boldsymbol{Z_g\Delta_g} - \boldsymbol{\delta_g}]^T \boldsymbol{R_g}^{-1}[\boldsymbol{y_g} - \boldsymbol{X_g\beta} - \boldsymbol{Z_g\Delta_g} - \boldsymbol{\delta_g}]/\sigma^2 + \lambda_O\|\boldsymbol{\delta_g}\|_0 + \lambda_G$$

$$\equiv \min_{\boldsymbol{t_g},\boldsymbol{\delta_g}} h_{g-robust}(\boldsymbol{t_g},\boldsymbol{\delta_g}) + \lambda_G, \text{where}$$

$$h_{g-robust}(\boldsymbol{t_g},\boldsymbol{\delta_g}) = [\boldsymbol{y_g} - \boldsymbol{X_g\beta} - \boldsymbol{Z_g t_g} - \boldsymbol{\delta_g}]^T \boldsymbol{R_g}^{-1}[\boldsymbol{y_g} - \boldsymbol{X_g\beta} - \boldsymbol{Z_g t_g} - \boldsymbol{\delta_g}]/\sigma^2 + \lambda_O\|\boldsymbol{\delta_g}\|_0$$

If $\boldsymbol{\Delta_g} = \mathbf{0}$ then

$$\min f_g = \min_{\tilde{\boldsymbol{b}}_{\boldsymbol{g}},\boldsymbol{\delta_g}} [\boldsymbol{y_g} - \boldsymbol{X_g\beta} - \boldsymbol{Z_g}\tilde{\boldsymbol{b}}_{\boldsymbol{g}} - \boldsymbol{\delta_g}]^T \boldsymbol{R_g}^{-1}[\boldsymbol{y_g} - \boldsymbol{X_g\beta} - \boldsymbol{Z_g}\tilde{\boldsymbol{b}}_{\boldsymbol{g}} - \boldsymbol{\delta_g}]/\sigma^2 + \lambda_O\|\boldsymbol{\delta_g}\|_0 + \tilde{\boldsymbol{b}}_g^T\boldsymbol{G}^{-1}\tilde{\boldsymbol{b}}_{\boldsymbol{g}}/\sigma^2$$

$$\equiv \min_{\boldsymbol{t_g},\boldsymbol{\delta_g}} h_{g-robustMix}(\boldsymbol{t_g},\boldsymbol{\delta_g}), \text{where}$$

$$h_{g-robustMix}(\boldsymbol{t_g},\boldsymbol{\delta_g}) = [\boldsymbol{y}_g - \boldsymbol{X_g\beta} - \boldsymbol{Z_g t_g} - \boldsymbol{\delta_g}]^T \boldsymbol{R_g}^{-1}[\boldsymbol{y_g} - \boldsymbol{X_g\beta} - \boldsymbol{Z_g t_g} - \boldsymbol{\delta_g}]/\sigma^2 + \lambda_O\|\boldsymbol{\delta_g}\|_0 + \boldsymbol{t}_{\boldsymbol{g}}^T\boldsymbol{G}^{-1}\boldsymbol{t_g}/\sigma^2$$

Then the penalized MLE for $\tilde{\boldsymbol{b}}_{\boldsymbol{g}}, \boldsymbol{\Delta_g}$ and $\boldsymbol{\delta_g}$ are:

(4.12)

$$(\hat{\tilde{\boldsymbol{b}}}_{\boldsymbol{g}}, \hat{\boldsymbol{\Delta}}_{\boldsymbol{g}}, \hat{\boldsymbol{\delta}}_{\boldsymbol{g}}) = \begin{cases} (\mathbf{0}, \hat{\boldsymbol{t}}_{\boldsymbol{g-robust}}, \hat{\boldsymbol{\delta}}_{\boldsymbol{g-robust}}), & \text{if } \min h_{g-robustMix} - \min h_{g-robust} \geq \lambda_G \\ (\hat{\boldsymbol{t}}_{\boldsymbol{g-robustMix}}, \mathbf{0}, \hat{\boldsymbol{\delta}}_{\boldsymbol{g-robustMix}}), & \text{otherwise} \end{cases}$$

where

$$\begin{cases} (\hat{\boldsymbol{t}}_{\boldsymbol{g-robust}}, \hat{\boldsymbol{\delta}}_{\boldsymbol{g-robust}}) = \arg\min h_{g-robust}(\boldsymbol{t_g},\boldsymbol{\delta_g}) \\ (\hat{\boldsymbol{t}}_{\boldsymbol{g-robustMix}}, \hat{\boldsymbol{\delta}}_{\boldsymbol{g-robustMix}}) = \arg\min h_{g-robustMix}(\boldsymbol{t_g},\boldsymbol{\delta_g}) \end{cases}$$

$\hat{t}_{g-robust}$ can be viewed as estimates from gene-wise robust regression using M-estimation and $\hat{\delta}_{g-robust}$ is the vector indicating outlying observations ($\hat{\delta}_{gi} = 0$ for non-outliers). On the other hand, $\hat{t}_{g-robustMix}$ can be viewed as moderated gene-wise robust regression because of the distribution assumed among all the random effects.

In another words, $\hat{t}_{g-robustMix}$ is shrunk toward 0. When comparing the Mahalanobis distances between $\hat{t}_{g-robustMix}$ and $\hat{t}_{g-robust}$ (results not shown here), the former is always greater than the latter.

With respect to optimizing the two functions, $h_{g-robust}$ and $h_{g-robustMix}$ are non-convex because of the $L_0$ penalties. Although they are in general hard to solve, if we assume that the observations in each group are independent, i.e., $\boldsymbol{R_g}$ is a diagonal matrix with the diagonal elements as $r_{g1}, ..., r_{gn_g}$, the objective functions can be further reduced to a sum of truncated quadratic functions, which is shown in the next section.

#### 4.3.2.2 Transform $h_g$ to sum of truncated convex functions

When $\boldsymbol{R_g}$ is diagonal, $h_{g-robust}$ and $h_{g-robustMix}$ can be further separated by each observation:

$$\begin{cases} h_{g-robust}(\boldsymbol{t_g}, \boldsymbol{\delta_g}) = \sum\limits_i [(y_{gi} - \boldsymbol{X_{gi}\beta} - \boldsymbol{Z_{gi}t_g} - \delta_{gi})^2/\sigma^2 r_{gi} + \lambda_O \|\delta_{gi}\|_0] \\ h_{g-robustMix}(\boldsymbol{t_g}, \boldsymbol{\delta_g}) = \sum\limits_i [(y_{gi} - \boldsymbol{X_{gi}\beta} - \boldsymbol{Z_{gi}t_g} - \delta_{gi})^2/\sigma^2 r_{gi} + \lambda_O \|\delta_{gi}\|_0] + \boldsymbol{t_g^T G^{-1} t_g}/\sigma^2 \end{cases}$$

If $\delta_{gj} = 0$, then

$$\begin{cases} h_{g-robust} = & [(y_{gj} - \boldsymbol{X_{gj}\beta} - \boldsymbol{Z_{gj}t_g})^2/\sigma^2 r_{gj}] + \\ & \sum\limits_{i \neq j} [(y_{gi} - \boldsymbol{X_{gi}\beta} - \boldsymbol{Z_{gi}t_g} - \delta_{gi})^2/\sigma^2 r_{gi} + \lambda_O \|\delta_{gi}\|_0] \\ h_{g-robustMix} = & [(y_{gj} - \boldsymbol{X_{gj}\beta} - \boldsymbol{Z_{gj}t_g})^2/\sigma^2 r_{gj}] + \\ & \sum\limits_{i \neq j} [(y_{gi} - \boldsymbol{X_{gi}\beta} - \boldsymbol{Z_{gi}t} - \delta_{gi})^2/\sigma^2 r_{gi} + \lambda_O \|\delta_{gi}\|_0 + \boldsymbol{t_g^T G^{-1} t_g}/\sigma^2] \end{cases}$$

If $\delta_{gj} \neq 0$, then both $h_{g-robust}$ and $h_{g-robustMix}$ are minimized when $\hat{\delta}_{gj} = y_{gj} -$

$\boldsymbol{X_{gj}\beta} - \boldsymbol{Z_{gj}t_g}$ and

$$\begin{cases} h_{g-robust} = \lambda_O + \sum\limits_{i \neq j}[(y_{gi} - \boldsymbol{X_{gi}\beta} - \boldsymbol{Z_{gi}t_g} - \delta_{gi})^2/\sigma^2 r_{gi} + \lambda_O \|\delta_{gi}\|_0] \\ h_{g-robustMix} = \lambda_O + \sum\limits_{i \neq j}[(y_{gi} - \boldsymbol{X_{gi}\beta} - \boldsymbol{Z_{gi}t_g} - \delta_{gi})^2/\sigma^2 r_{gi} + \lambda_O \|\delta_{gi}\|_0 + \boldsymbol{t_g^T G^{-1} t_g}/\sigma^2] \end{cases}$$

Therefore, the penalized maximum likelihood estimate for the $\delta_{gi-robust}$ and $\delta_{gi-robustMix}$ are:

$$\hat{\delta}_{gi-robust} = \begin{cases} 0, \quad \text{if } [y_{gi} - \boldsymbol{X_{gi}\beta} - \boldsymbol{Z_{gi}}(\boldsymbol{\hat{t}_{g-robust}})]^2/\sigma^2 r_{gi} < \lambda_O \\ y_{gi} - \boldsymbol{X_{gi}\beta} - \boldsymbol{Z_{gi}}(\boldsymbol{\hat{t}_{g-robust}}), \quad \text{otherwise} \end{cases} \tag{4.13}$$

$$\hat{\delta}_{gi-robustMix} = \begin{cases} 0, \quad \text{if } [y_{gi} - \boldsymbol{X_{gi}\beta} - \boldsymbol{Z_{gi}}(\boldsymbol{\hat{t}_{g-robustMix}})]^2/\sigma^2 r_{gi} < \lambda_O \\ y_{gi} - \boldsymbol{X_{gi}\beta} - \boldsymbol{Z_{gi}}(\boldsymbol{\hat{t}_{g-robustMix}}), \quad \text{otherwise} \end{cases} \tag{4.14}$$

With (4.13) and (4.14), we can re-write $h_{g-robust}$ and $h_{g-robustMix}$ as:

$$\begin{cases} h_{g-robust}(\boldsymbol{t_g}) = \sum\limits_i \min\{(y_{gi} - \boldsymbol{X_{gi}\beta} - \boldsymbol{Z_{gi}t_g})^2/\sigma^2 r_{gi}, \lambda_O\} \\ h_{g-robustMix}(\boldsymbol{t_g}) = \sum\limits_i \min\{(y_{gi} - \boldsymbol{X_{gi}\beta} - \boldsymbol{Z_{gi}t_g})^2/\sigma^2 r_{gi}, \lambda_O\} + \boldsymbol{t_g^T G^{-1} t_g}/\sigma^2 \end{cases} \tag{4.15}$$

The above derivation showed that $\hat{\delta}_{gi-robust}$ and $\hat{\delta}_{gi-robustMix}$ are functions of $\boldsymbol{t_{g-robust}}$ and $\boldsymbol{t_{g-robustMix}}$, and $h_{g-robust}$ and $h_{g-robustMix}$ can be further reduced to functions of $\boldsymbol{t_{g-robust}}$ and $\boldsymbol{t_{g-robustMix}}$. In addition, since both functions are sums of truncated quadratic functions ( $\boldsymbol{t_g^T G^{-1} t_g}$ can be seen as being truncated at infinity), they can be solved by the algorithm proposed in [30].

### 4.3.3 Summary of estimating $\Delta$ and $\delta$

In summary, with independent observations in each group, the original objective function as in (4.7) can be broken down to $m$ parallel optimization problems, one for each group. For each group of observations, we first solve:

(4.16)
$$\begin{cases} \hat{\boldsymbol{t}}_{\boldsymbol{g-robust}} = \arg\min \quad h_{g-robust} = & \underset{\boldsymbol{t_g}}{\arg\min} \quad \sum_i \min\{(y_{gi} - \boldsymbol{X_{gi}\beta} - \boldsymbol{Z_{gi}t_g})^2/\sigma^2 r_{gi}, \lambda_O\} \\ \hat{\boldsymbol{t}}_{\boldsymbol{g-robustMix}} = \arg\min \quad h_{g-robustMix} = & \underset{\boldsymbol{t_g}}{\arg\min} \quad \sum_i \min\{(y_{gi} - \boldsymbol{X_{gi}\beta} - \boldsymbol{Z_{gi}t_g})^2/\sigma^2 r_{gi}, \lambda_O\} + \\ & \boldsymbol{t_g^T G^{-1} t_g}/\sigma^2 \end{cases}$$

and then we have

(4.17)
$$(\hat{\hat{\boldsymbol{b}}}_{\boldsymbol{g}}, \hat{\boldsymbol{\Delta}}_{\boldsymbol{g}}, \hat{\boldsymbol{\delta}}_{\boldsymbol{g}}) = \begin{cases} (\boldsymbol{0}, \hat{\boldsymbol{t}}_{\boldsymbol{g-robust}}, \hat{\boldsymbol{\delta}}_{\boldsymbol{g-robust}}), & \text{if } \min h_{g-robustMix} - \min h_{g-robust} \geq \lambda_G \\ (\hat{\boldsymbol{t}}_{\boldsymbol{g-robustMix}}, \boldsymbol{0}, \hat{\boldsymbol{\delta}}_{\boldsymbol{g-robustMix}}), & \text{otherwise} \end{cases}$$

where

$$(4.18) \qquad \hat{\delta}_{gi-robust} = \begin{cases} 0, & \text{if } [y_{gi} - \boldsymbol{X_{gi}\beta} - \boldsymbol{Z_{gi}}(\hat{\boldsymbol{t}}_{\boldsymbol{g-robust}})]^2/\sigma^2 r_{gi} < \lambda_O \\ y_{gi} - \boldsymbol{X_{gi}\beta} - \boldsymbol{Z_{gi}}(\hat{\boldsymbol{t}}_{\boldsymbol{g-robust}}), & \text{otherwise} \end{cases}$$

$$(4.19) \quad \hat{\delta}_{gi-robustMix} = \begin{cases} 0, & \text{if } [y_{gi} - \boldsymbol{X_{gi}\beta} - \boldsymbol{Z_{gi}}(\hat{\boldsymbol{t}}_{\boldsymbol{g-robustMix}})]^2/\sigma^2 r_{gi} < \lambda_O \\ y_{gi} - \boldsymbol{X_{gi}\beta} - \boldsymbol{Z_{gi}}(\hat{\boldsymbol{t}}_{\boldsymbol{g-robustMix}}), & \text{otherwise} \end{cases}$$

If we define the conditional residual as $e_{gi} = y_{gi} - \boldsymbol{X_{gi}\hat{\beta}} - \boldsymbol{Z_{gi}(\hat{\hat{b}}_g + \Delta_g)}$, then for an outlying observation, the estimated outlying quantity, $\hat{\delta}_{gi}$, will pick up the error term and be equivalent to the estimated conditional residual, $\hat{e}_{gi} = y_{gi} - \boldsymbol{X_{gi}\hat{\beta}} - \boldsymbol{Z_{gi}(\hat{\hat{b}}_g +}$

| | Non-outlying observations | Outlying observations |
|---|---|---|
| Non-outlying effect | $\hat{\tilde{\boldsymbol{b}}}_{\boldsymbol{g}} = \hat{\boldsymbol{t}}_{\boldsymbol{g-robustMix}}$ <br> $\hat{\boldsymbol{\Delta}}_{\boldsymbol{g}} = \boldsymbol{0}$ <br> $\hat{\delta}_{gi} = 0$ | $\hat{\tilde{\boldsymbol{b}}}_{\boldsymbol{g}} = \hat{\boldsymbol{t}}_{\boldsymbol{g-robustMix}}$ <br> $\hat{\boldsymbol{\Delta}}_{\boldsymbol{g}} = \boldsymbol{0}$ <br> $\hat{\delta}_{gi} = \hat{\delta}_{gi-robustMix} = \hat{e}_{gi}$ |
| Outlying effect | $\hat{\tilde{\boldsymbol{b}}}_{\boldsymbol{g}} = \boldsymbol{0}$ <br> $\hat{\boldsymbol{\Delta}}_{\boldsymbol{g}} = \hat{\boldsymbol{t}}_{\boldsymbol{g-robust}}$ <br> $\hat{\delta}_{gi} = 0$ | $\hat{\tilde{\boldsymbol{b}}}_{\boldsymbol{g}} = \boldsymbol{0}$ <br> $\hat{\boldsymbol{\Delta}}_{\boldsymbol{g}} = \hat{\boldsymbol{t}}_{\boldsymbol{g-robust}}$ <br> $\hat{\delta}_{gi} = \hat{\delta}_{gi-robust} = \hat{e}_{gi}$ |

Table 4.1: Summary of the estimates with corresponding outlyingness

$\hat{\boldsymbol{\Delta}}_{\boldsymbol{g}}$). We summarize the estimates with corresponding outlyingness in table (4.1).

## 4.4 Determine $\lambda_O$ and $\lambda_G$

The tuning parameter $\lambda_O$ and $\lambda_G$, in general, cannot be determined readily by cross-validation procedures. For example, when an observation has a significant prediction error, it could be caused by either biased parameter estimates or that the observation itself is an outlier [45]. Consider again the estimates resulted from the penalized MLE in (4.18) and (4.19). We can see that finding the $\lambda_O$ is equivalent to finding suitable cutoffs to define outlying observations. The criteria in (4.18) and (4.19) use conditional residuals to select outliers, and we will show that this is similar to determining the cutoffs based on the estimated distribution of the conditional residuals [40].

For $\lambda_G$, we resort to the estimated distribution of the predicted random effects and use the Mahalanobis distances of these effects to form a one-dimensional criterion to detect outlying effects. In the following sections, we first define the conditional residual, the empirical BLUP and their distribution under the proposed model (4.4). Then we show that we could use these two distributions to define cutoffs for outliers.

### 4.4.1 Variance of the conditional residuals and predicted random effects

Based on the proposed model in (4.4), we have

$$cov(y) = \sigma^2(ZDZ^T + \Sigma_\epsilon) = \sigma^2\Sigma_y$$

Let $H$ be the Cholesky decomposition of $\Sigma_y$ such that $HH^T = \Sigma_y$, then

$$H^{-1}(y - Z\Delta - \delta) = H^{-1}X\beta + H^{-1}Zb + H^{-1}\epsilon.$$

Also, since

$$cov(H^{-1}y) = H^{-1}\sigma^2\Sigma_y(H^{-1})^T = \sigma^2 I_N,$$

the best linear unbiased estimator for $\beta$ is therefore

$$\begin{aligned}\hat{\beta} &= [(H^{-1}X)^T H^{-1}X]^{-1}(H^{-1}X)^T H^{-1}(y - Z\Delta - \delta)\\ &= (X^T\Sigma_y^{-1}X)^{-1}X^T\Sigma_y^{-1}(y - Z\Delta - \delta)\\ &= T(y - Z\Delta - \delta)\end{aligned}$$

where we define $\boldsymbol{T = (X^T\Sigma_y^{-1}X)^{-1}X^T\Sigma_y^{-1}}$, following the settings in [40].

For the empirical BLUP of $b$, because we don't have closed-form solutions for $\hat{\Delta}$ and $\hat{\delta}$, we treat $\Delta$ and $\delta$ as known in the process of driving the variance-covariance matrix for $b$ and then plug in $\hat{\Delta}$ and $\hat{\delta}$ to the variance-covariance matrix formula for $b$. The drawback of this approach is that we will underestimate the variation of $\boldsymbol{b}$ since we don't take into account the fact that $\hat{\Delta}$ and $\hat{\delta}$ are estimated. However, if $\hat{\Delta}$ and $\hat{\delta}$ are close to the true values, the estimation for the variance-covariance matrix for $b$ can be very close to the truth. The empirical BLUP is defined as:

$$\begin{aligned}\hat{\hat{b}} &= DZ^T\Sigma_y^{-1}(y - X\hat{\beta} - Z\Delta - \delta)\\ &= DZ^T\Sigma_y^{-1}[y - Z\Delta - \delta - XT(y - Z\Delta - \delta)]\\ &= DZ^T\Sigma_y^{-1}(I - XT)(y - Z\Delta - \delta)\\ &= DZ^TQ(y - Z\Delta - \delta)\end{aligned}$$

where we define $\boldsymbol{Q = \Sigma_y^{-1}(I - XT)}$, also following [40].

For the conditional residual, we also treat $\Delta$ as known and define it as:

$$\begin{aligned}e &= y - X\hat{\beta} - Z(\hat{\hat{b}} + \Delta)\\ &= y - Z\Delta - \delta - XT(y - Z\Delta - \delta) - ZDZ^TQ(y - Z\Delta - \delta) + \delta\\ &= (\Sigma_yQ - ZDZ^TQ)(y - Z\Delta - \delta) + \delta\\ &= \Sigma_\epsilon Q(y - Z\Delta - \delta) + \delta\end{aligned}$$

Note that $Q$ is a symmetric matrix ($Q^T = Q$) and that $XT$ is a projection matrix ($(XT)^2 = XT$). Therefore, $I - XT$ is also a projection matrix. These lead to

$$\begin{aligned}var(e) &= \Sigma_\epsilon Q(\sigma^2\Sigma_y)Q\Sigma_\epsilon\\ &= \sigma^2\Sigma_\epsilon Q\Sigma_\epsilon\end{aligned}$$

### 4.4.2 Determine the cutoffs for conditional residuals and predicted random effects

Based on the previous derivation, the conditional residuals have a normal distribution with mean $\mathbf{0}$ and estimated variance as $\widehat{var(e)}$=$\hat{\sigma}^2\boldsymbol{\hat{\Sigma}_\epsilon\hat{Q}\hat{\Sigma}_\epsilon}$, where $\hat{Q} = \hat{\Sigma_y}^{-1}(I - X\hat{T})$ and $\hat{T} = (X^T\hat{\Sigma_y}^{-1}X)^{-1}X^T\hat{\Sigma_y}^{-1}$. Let $p_{gi}$ denote the diagonal element of the matrix $\boldsymbol{\hat{\Sigma}_\epsilon\hat{Q}\hat{\Sigma}_\epsilon}$ then $\hat{\sigma}^2p_{gi}$ is the estimated variance of the conditional residual of the observation $gi$. The estimated standardized conditional residual therefore is:

$$e^*_{gi} = \frac{\hat{e}_{gi}}{\hat{\sigma}\sqrt{p_{gi}}}$$

where $\hat{e}_{gi} = y_{gi} - X_{gi}\hat{\beta} - Z_{gi}(\hat{\tilde{b}}_g + \hat{\Delta}_g)$. A cutoff for an outlying observation can be defined as

$$|e^*_{gi}| > Z_\alpha. \tag{4.20}$$

$\alpha$ is an user-defined probability that an observation conforming to the specified distribution having its standardized conditional residual falling beyond the cutoff $Z_\alpha$, which is a quantile of the normal distribution under normality assumption. Expanding (4.20), we will have a criterium for outlying observations similar to that based on the penalized maximum likelihood in (4.18) and (4.19):

$$\hat{e}^2_{gi} = [y_{gi} - X_{gi}\hat{\beta} - Z_{gi}(\hat{\tilde{b}}_g + \hat{\Delta}_g)]^2$$
$$= \begin{cases} [y_{gi} - X_{gi}\hat{\beta} - Z_{gi}\hat{t}_{g-robustMix}]^2 > \chi^2_{\alpha,1}\hat{\sigma}^2 p_{gi}, & \text{if the random effect is not outlying.} \\ [y_{gi} - X_{gi}\hat{\beta} - Z_{gi}\hat{t}_{g-robust}]^2 > \chi^2_{\alpha,1}\hat{\sigma}^2 p_{gi}, & \text{if the random effect is outlying.} \end{cases}$$

If we define $w_{gi} = p_{gi}/r_{gi}$ and let $\lambda_O = \chi^2_{\alpha,1}$ then the above can be written as

$$\hat{e}^2_{gi} = \begin{cases} [y_{gi} - X_{gi}\hat{\beta} - Z_{gi}\hat{t}_{g-robustMix}]^2/\hat{\sigma}^2 r_{gi} > w_{gi}\lambda_O, & \text{if the random effect is not outlying.} \\ [y_{gi} - X_{gi}\hat{\beta} - Z_{gi}\hat{t}_{g-robust}]^2/\hat{\sigma}^2 r_{gi} > w_{gi}\lambda_O, & \text{if the random effect is outlying.} \end{cases}$$

The loss function of individual observations:

$$\min\{(\text{conditional residual})^2/\sigma^2, \lambda_O w_{gi}\}$$

$^*w_{gi}\sigma^2$ =variance of conditional residual

We can see that the above criterium leads to a penalized regression model with a weighted penalty term, $\lambda_O \sum_g \sum_i w_{gi} \|\delta_{gi}\|_0 = \lambda_O \langle w, \|\delta\|_0 \rangle$.

For outlying random effects, we resort to the estimated distribution of the random effects, which has a multivariable normal distribution with a mean $\mathbf{0}$ and an estimated variance $\hat{G}$. Since the random effects can have more than one dimension, an one-dimensional cutoff for the outlying predicted random effects can be defined by the Mahalanobis distances:

$$M_g = \hat{\boldsymbol{t}}^{\boldsymbol{T}}_{\boldsymbol{g-robustMix}} (\hat{\sigma}^2 \hat{G}_g)^{-1} \hat{\boldsymbol{t}}_{\boldsymbol{g-robustMix}} > \chi^2_{q,\alpha}, \tag{4.21}$$

where $G_g$ is the $g^{th}$ diagonal block of $G$, $q$ is the dimension of the random effects, and $\alpha$ is an user-defined probability. It describes the tolerated chance of a random effect conforming to the specified distribution having its Mahalanobis distance go beyond the cutoff. In other words, it is the tolerated false positive rate for outlier detection. We further define

$$k_g = \min h_{g-robustMix} - \min h_{g-robust} - M_g.$$

$k_g$ can be viewed as the increased sum of truncated squared standardized conditional residuals when we use the moderated, shrunk robust regression estimates based on $h_{g-robustMix}$ instead of that based on $h_{g-robust}$. Because estimates based on $h_{g-robustMix}$ are "predicated" random effects rather than true group-specific regression coefficients, artificially generated outliers due to the increased conditional residual from moderated coefficients are not desirable. In particular, we don't want the percentage of outliers generated from moderated estimates to go beyond the threshold $\alpha$ that the user defines, which is also the false positive rate. Therefore, a practical criterium for $k_g$ can be set as:

$$k_g < (\chi^2_{\alpha,1} - \chi^2_{0.5,1}) * \alpha * n_g,$$

where $\chi^2_{\alpha,1} - \chi^2_{0.5,1}$ approximates the amount of increased squared standardized residual when a typical observation become outlying. The above leads to a practical setting for $\lambda_G$ as $\chi^2_{\alpha,q} + (\chi^2_{\alpha,1} - \chi^2_{0.5,1}) * \alpha * n_g$

In summary, the search for analytic expressions of $\lambda_O$ and $\lambda_G$ based on the estimated distribution of conditional residuals and random effects gives rise to a penalized regression model with weighted penalties as the following:

(4.22)

$$\begin{aligned} -2\log f(\boldsymbol{y}) =& \log|2\pi\sigma^2\boldsymbol{\Sigma_y}| + (\boldsymbol{y} - \boldsymbol{X\beta} - \boldsymbol{Z\Delta} - \boldsymbol{\delta})^T\boldsymbol{\Sigma}_{\boldsymbol{y}}^{-1}(\boldsymbol{y} - \boldsymbol{X\beta} - \boldsymbol{Z\Delta} - \boldsymbol{\delta})/\sigma^2 + \\ & \lambda_O\langle w, \|\delta\|_0\rangle + \lambda_G \sum_g \mathbb{1}_{\Delta_g \neq \boldsymbol{0}} \end{aligned}$$

where $\boldsymbol{\Sigma_y} = (\boldsymbol{ZDZ^T} + \boldsymbol{\Sigma_\epsilon})$

$\lambda_O = \chi^2_{\alpha,1}$

$\lambda_G = \chi^2_{\alpha,q} + (\chi^2_{\alpha,1} - \chi^2_{0.5,1}) * \alpha * n_g$

$\alpha$: an user-defined probability allowing a conformer being identified as an outlier. It is the false positive rate of outlier detection.

$w$: a vector of weights such that $\sigma^2 w$ are variances of conditional residuals for $y$.

## 4.5 Dilation factor

The random effect variance $\sigma^2 G$ and the error variance $\sigma^2$ obtained through the R package "lme4" are based on the filtered observations with the detected outliers removed, which follow a truncated normal distribution. We found that the actual variance is, in fact, the variance of the truncated data multiplied by a constant.

*Proposition* 4.3. Suppose that $X_1, X_2, ..., X_n \sim N_q(\mathbf{0}, \Sigma)$. Let $\Sigma^* = cov(X|X^T\Sigma^{-1}X \leq k)$, the conditional covariance based on truncated data. Then it can be shown that

$$\Sigma^* = a\Sigma, \text{where a is a scalar.}$$

In particular, when $q = 2$,

$$a = \frac{2 - (k+2)\exp(-k/2)}{2\alpha}$$

where $\alpha = P(z^Tz \leq k)$, $z \sim N_p(\mathbf{0}, I_p)$

*Proof.* Let $\Gamma\Gamma^T$ be the Cholesky decomposition of $\Sigma$ such that $\Sigma = \Gamma\Gamma^T$. Then we have

$$\Sigma^{-1} = (\Gamma\Gamma^T)^{-1} = (\Gamma^T)^{-1}\Gamma^{-1}$$

Let $z = \Gamma^{-1}X$, then $z \sim N_p(0, I_p)$ and $X = \Gamma z$. In addition,

$$X^T\Sigma^{-1}X = X^T(\Gamma^{-1})^T\Gamma^{-1}X = z^Tz$$

Therefore,

$$\begin{aligned}
\Sigma^* &= cov(X|X^T\Sigma^{-1}X \leq k) \\
&= cov(\Gamma z|z^Tz \leq k) \\
&= E[\Gamma zz^T\Gamma^T|z^Tz \leq k] + E[\Gamma z|z^Tz \leq k]E[\Gamma z|z^Tz \leq k]^T \\
&= \Gamma E[zz^T|z^Tz \leq k]\Gamma^T
\end{aligned}$$

Let $\alpha = P(z^Tz \leq k), A = \{z : z^Tz \leq k\}$ and $f$ be the density of $z$ then

$$E[zz^t|z^Tz \leq k] = \int_A zz^T\frac{f(z)}{\alpha}dz,$$

Using the polar coordinate, we set $z = (rcos\theta, rsin\theta)$, then

$$zz^T = r^2 \begin{bmatrix} cos^2\theta & sin\theta cos\theta \\ sin\theta cos\theta & sin^2\theta \end{bmatrix} = r^2 M(\theta)$$

$$f(z) = f(r,\theta) = r\exp(-0.5r^2)/2\pi$$

$$\begin{aligned} E[zz^T|z^Tz \leq k] &= \frac{1}{\alpha}\int_0^{\sqrt{k}}\int_0^{2\pi} r^2 M(\theta) f(r,\theta) d\theta dr \\ &= \frac{2-(k+2)\exp(-k/2)}{2\alpha} I_2 \end{aligned}$$

□

## 4.6 Simulation Study

There are two simulating scenarios: the first one mimics our problem for which there are limited observations for each gene, but the number of genes is large. We generated simulated datasets with ten paired samples for each gene and 2000 genes for each replicate. In the second scenario, we also created ten paired samples for each gene but had only 20 genes for each replicate. In both situations, there are 20 percents of outlying measurements for each gene and 20 percents of outlying genes. The outlier detection threshold variable $\alpha$ is set to be 0.05, which is equivalent to an allowance of the swamping (false positive) rate of 0.05, for both individual measurements and genes.

In the generated data sets, the fixed intercept is 0, and the fixed slope is 1. The random intercepts follow a normal distribution with mean 0 and variance 3; the random slopes follow a normal distribution with mean 0 and variance 1. The covariance between the random intercepts and random slopes is -1.2. The errors follow the normal distribution with mean 0 and standard deviation 1. The single independent variable has a uniform distribution between -15 and 15.

For outlying data, each random effect was generated from one of the equally likely

| | Masking rate (%) | | Swamping rate (%) | | Run time |
|---|---|---|---|---|---|
| | observation | random effect | observation | random effect | (minutes) |
| naive | 0.0000 | 0.0000 | 1.2500 | 6.2500 | 0.0013 |
| proposed | 0.0000 | 0.0000 | 3.1250 | 0.0000 | 0.0798 (using 6 CPUs) |

Table 4.2: Outlier detection for small data sets composed of 20 genes, each with10 paired measurements. The results are based on the medians of the 100 replicates. For the proposed method, the ideal masking (false negative) rate is 0 and the swamping (false positive) rate should be close to the user-defined threshold $\alpha$, which is 5% in the simulation setting.

distributions: (1) bivariate normal distribution with mean (10,8), variances (5,2) and covariance 1.2; (2) two independent exponential distributions: (*exp*(0.3)+7, *exp*(0.5)-4); (3) two independent exponential distributions: (*exp*(0.3)-10, *exp*(0.5)-6); (4) two independent exponential distributions: (*exp*(0.3)-12, *exp*(0.5)+15). The individual outliers within each group have the covariate X generated from $uniform(10, 20)$ and an additional deviation term $\delta$ generated from *exp*(0.3)+10 such that these outliers would have large influences.

We compared the performance of the proposed method with a naive method that:

1. Detects outlying observations by gene-wise robust regression using the R function "lmRob" from the package "robust". The function generates initial estimates through S-estimation and then produces the final estimates by MM estimates. The error standard deviation common for all genes is estimated by the median of the residuals from gene-wise robust regressions.

2. Detects outlying random effects and provide estimates the location and scale of the multivariate distribution of the random effects using the R function "covRob" from the package "robust".

For each scenario we generate 100 replicates and the results are listed in Table 4.2 and Table 4.4.

| | Masking rate (%) | | Swamping rate (%) | | Run time (minutes) |
|---|---|---|---|---|---|
| | observation | random effect | observation | random effect | |
| naive | 0.5750 | 0.2500 | 1.1312 | 6.4375 | 0.0852 |
| proposed | 0.2250 | 0.0000 | 4.4062 | 4.7500 | 3.5925 (using 6 CPUs) |

Table 4.3: Outlier detection for data sets composed of 2000 genes, each with10 paired measurements. The results are based on the medians of the 100 replicates. For the proposed method, the ideal masking (false negative) rate is 0 and the swamping (false positive) rate should be close to the user-defined threshold $\alpha$, which is 5% in the simulation setting.

| | Fixed effect | | Random effect variance-covariance matrix | | | Error S.D. |
|---|---|---|---|---|---|---|
| | Intercept | Slope | $\Sigma_{11}$ | $\Sigma_{12}$ | $\Sigma_{22}$ | |
| True Value | 0 | 1 | 3 | -1.2 | 1 | 1 |
| Naive | -0.0183 | 1.0184 | 4.0732 | -1.4709 | 1.2740 | 1.4205 |
| Proposed | -0.0615 | 1.0374 | 3.1556 | -1.0603 | 1.0020 | 1.1443 |

Table 4.4: Parameter estimation by the naive method and the proposed method based on data sets composed of 20 genes, each with 10 paired measurements. The results are based on the medians of the 100 replicates. .

| | Fixed effect | | Random effect variance-covariance matrix | | | Error S.D. |
|---|---|---|---|---|---|---|
| | Intercept | Slope | $\Sigma_{11}$ | $\Sigma_{12}$ | $\Sigma_{22}$ | |
| True Value | 0 | 1 | 3 | -1.2 | 1 | 1 |
| naive | 0.0143 | 0.9998 | 4.4315 | -1.6779 | 1.4218 | 1.4321 |
| Proposed | 0.0007 | 1.0019 | 2.9926 | -1.2101 | 1.0144 | 1.1190 |

Table 4.5: Parameter estimation by the naive method and the proposed method based data sets composed of 2000 genes, each with 10 paired measurements. The results are based on the medians of the 100 replicates. .

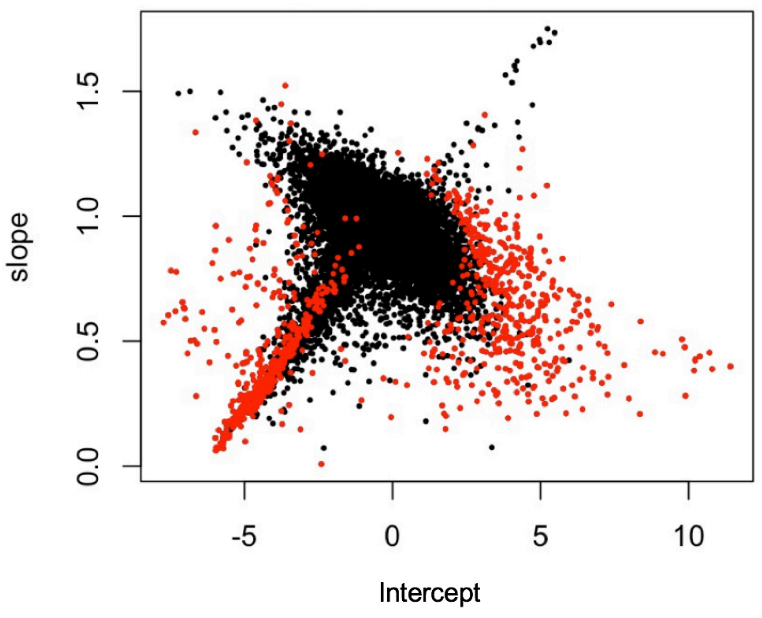

Figure 4.2: Illustration of outlier detection for a mixed effect by the proposed method in real RNA-Seq data. There are paired Poly(A) and Capture RNA-Seq measurements for 18,000 genes from 100 subjects. We use a mixed model to the predict Poly(A) measurements based on a single predictor, the Capture RNA-Seq measurements with gene-specific random intercepts and slopes. Because the outlying gene-specific effects will be estimated as zeros in the proposed method, we use estimated coefficients from the simple regression in to demonstrate the relative positions of the detected outlying random effects (red) and normal random effects (black).

## 4.7 Application to RNA-Seq data

We apply the proposed algorithm to the RNA-seq data with 100 subjects and 18,000 genes. Because the most critical variable, RNA integrity number (RIN), correlates poorly with the level of RNA degradation at low RINs. We only use the Capture RNA-Seq measurements as the predictor of the Poly(A) RNA-Seq measurements. Each gene is a group in the proposed robust mixed effect model and has its random intercept and random slope. The result is illustrated in Figure 4.2 with the detected outlying genes marked in red color.

A considerable number of random effects in the right upper quadrant, left upper quadrant and the lower center portions of the figure do not conform to the specified bivariate normal distribution but are not detected as outliers. We examine some

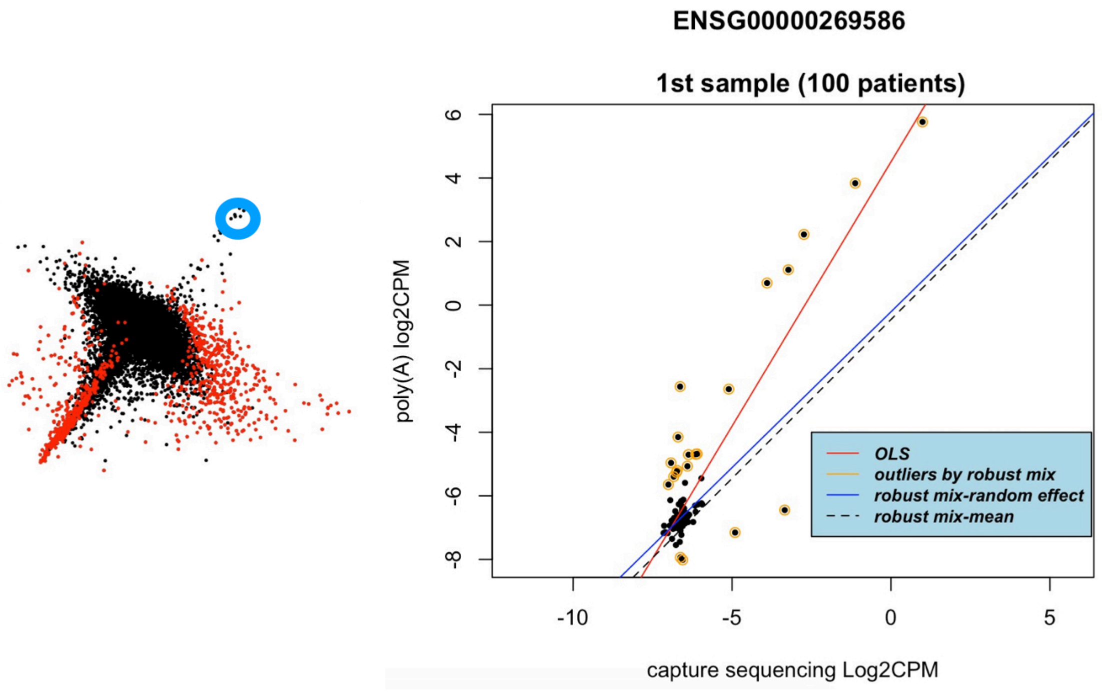

Figure 4.3: The scatter plot of Capture and Poly(A) RNA-Seq measurements from one of the undetected outlying genes whose OLS estimates lies in the right upper quadrant in Figure 4.2. The gene-specific OLS estimates the standard error to be 1.1 while the common standard error assumed by the proposed method across all genes is 0.46. There are also unequal variances among observations within a gene.

of these genes with undetected outlying effects, such as those on the right upper quadrant as in Figure 4.3, which have relatively larger slopes. We find that a lot of the observations with high variance are identified as outliers (circled by yellow color), and therefore the estimate for the gene-specific random effect is based on a cluster of observations with low variance, which is close to the population and thus the gene is not identified despite the overall aberrant pattern. This is because the proposed method assumes a common error variance across different genes and also constant variance for all observations within a gene. Take the illustrated gene as an example, the gene-specific OLS estimates the standard error to be 1.1 while the common standard error assumed by the proposed method across all genes is 0.46.

Figure 4.4 and Figure 4.5 illustrate another example of outlying genes not being

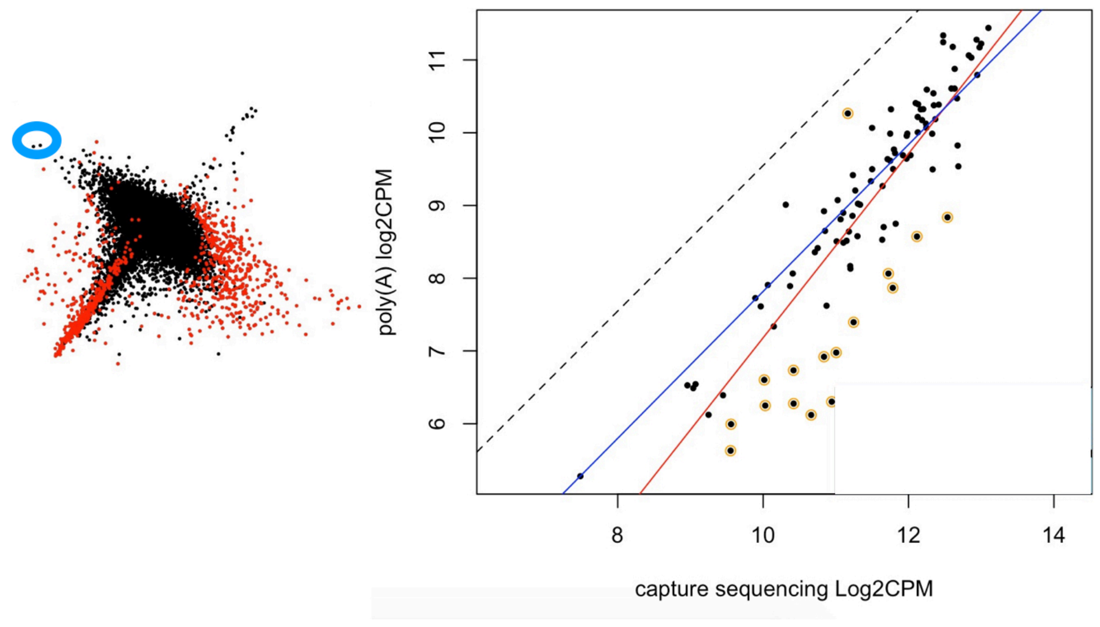

Figure 4.4: The scatter plot of Capture and Poly(A) RNA-Seq measurements from one of the undetected outlying genes whose OLS estimates lies in the left upper quadrant in Figure 4.2. The gene-specific OLS estimates the standard error to be 0.72 while the common standard error assumed by the proposed method across all genes is 0.46. There are also unequal variances among observations within a gene.

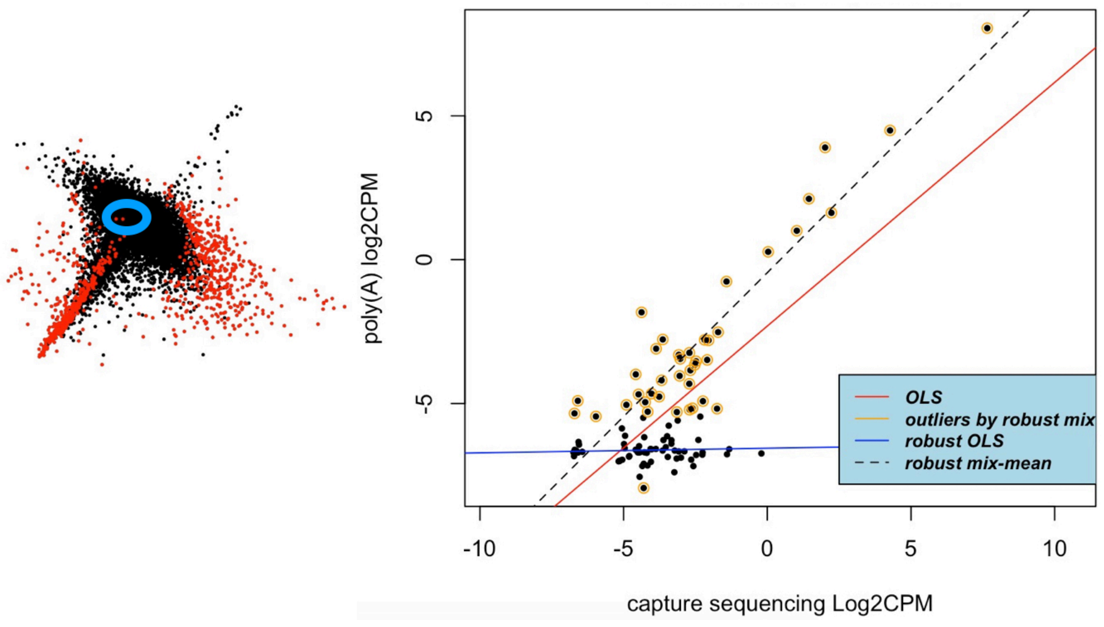

Figure 4.5: The scatter plot of Capture and Poly(A) RNA-Seq measurements from one of the detected genes whose OLS estimates are not outlying. The gene-specific OLS estimates the standard error to be 1.9 while the common standard error assumed by the proposed method across all genes is 0.46. There are also unequal variances among observations within a gene.

detected (false negative) and an example of identified genes whose OLS estimates are not outlying (false positive). For these genes, their gene-specific error variances are greater the assumed common error variance across all genes. Besides, there are unequal variances among the observations within the gene.

## 4.8 Discussion

Tuning the parameters $\lambda_O$ and $\lambda_G$ is in general difficult, as large prediction errors could be due to biased estimates or the observation being outlying. However, we found that under the assumption of normal distribution, setting $\lambda_O = \chi^2_{\alpha,1}$ and $\lambda_G = \chi^2_{\alpha,q} + (\chi^2_{\alpha,1} - \chi^2_{0.5,1}) * \alpha * n_g$ gives satisfactory results, especially when the separation between outliers and normal data is clear.

Because the objection function for the penalized MLE is non-convex, the optimization of the proposed method is also in general difficult. We used a specialized algorithm to minimize the sum of truncated quadratic functions. This algorithm provides exact solutions for low-dimensional problems and generates a fast local solution for high-dimensional problems. Also, we were able to separate the objection function by groups of observations. Therefore the computation can run in parallel, and the runtime can be reduced linearly.

Regarding the robust estimation of the parameters, we find the proposed method more accurate than the naive method when the number of observations within a group and the number of groups is small. However, we expect that as the number of observations and the number of groups grows larger, the precision of the naive method will improve with the required computing resources remaining minimal.

Finally, the drawback of the proposed algorithm is that we assume a common error variance across all groups and a constant error variance among observations within a group. This leads to false positives and false negatives for outlier detection when these assumptions do not hold, as illustrated in the real RNA-Seq data. The potential future extension includes theoretical properties of the estimation scheme and incorporating group-specific variances and unequal variances among observations within a group into the proposed method.

# CHAPTER V

# Discussion

The mean-shift model with sparse estimation with $L_0$ penalty is a versatile tool for outlier detection and robust estimation in the regression [45] and linear mixed model. We also demonstrated that the problem of parameter tuning could be regarded as finding the threshold for defining the outliers under the specified distribution. However, accuracy for such algorithms depends on accuracy in the estimation of the error variances and covariances of the random effects. As illustrated in the application to real RNA-Seq data, when the assumption about the variances such as homoscedasticity in a group or a common error variance shared by all groups, are not met, there could be considerable false negatives and false positives.

In the proposed robust mixed model, we estimated the outliers and variances in two iterative steps: we first identify the outliers using a penalized objective function and then estimated the error variance and random effect covariance based on the filtered dataset, which has the detected outliers removed. We repeated the two steps until the fixed effects, the error variance and the covariance converged. This iterative approach generated effective and efficient estimation in simulation studies. However, we have not fully understood its theoretical properties, including convergence, consistency and statistical efficiency.

Potential future work includes studying the theoretical properties of this iterative estimation scheme and incorporating group-specific variances and mechanisms for heteroscedasticity in the outlier detection model.

# Appendix

## A1 Appendix for Chapter II

### A1.1 Application on detecting differential gene expression with $\ell_0$-penalized models

The idea of using the $\ell_0$ penalty for variable selection can also be applied to the detection of differentially expressed genes from RNA sequencing data. The problem is discussed in detail in [23], and we briefly summarize the approach here. Given $S$ experimental groups each with $n_s$ biological samples, we would like to compare the expression levels of $m$ genes measured in the samples. Let $\mu_{si}$ be the mean expression level of gene $i$ (on the log-scale) in group $s$, $d_{sj}$ be the scaling factor (e.g., sequencing depth or library size on the log-scale) for sample $j$ in group $s$, and $\sigma_i^2$ be the variance of expression level of gene $i$ (on the log-scale). Assuming a linear model on the observed data $x_{sij} \sim N(\mu_{si} + d_{sj}, \sigma_i^2)$, the problem is to identify genes that are differentially expressed across the groups. To do so, assuming $\{\sigma_i\}_{i=1}^m$ are known, reparametrizing $\mu_{si}$ as $\mu_i = \mu_{1i}, \gamma_{si} = \mu_{si} - \mu_{1i}, s = 1, \ldots, S$, the $\ell_0$-penalized negative log-likelihood function of the model is

$$f(\mu, \gamma, d) = \sum_{i=1}^{m} \frac{1}{2\sigma_i^2} \sum_{s=1}^{S} \sum_{j=1}^{n_s} (x_{sij} - \mu_i - \gamma_{si} - d_{sj})^2 + \sum_{i=1}^{m} \alpha_i 1(\sum_{s=1}^{S} |\gamma_{si}| > 0) \tag{5.1}$$

Where $\{\alpha_i\}_{i=1}^m$ are tuning parameters. It is shown in [23] that (5.1) can be solved as follows

$$d'_{sj} = (\textstyle\sum_{i=1}^m (x_{sij} - x_{si1})/\sigma_i^2)/(\sum_{i=1}^m 1/\sigma_i^2), s = 1, \ldots, S$$

$$\mu'_{si} = (1/n_s) \textstyle\sum_{j=1}^{n_s} (x_{sij} - d'_{sj}), s = 1, \ldots, S$$

$$d_1 = 0$$

$$d_2, \ldots, d_S = \underset{d_2,\ldots,d_S}{\arg\min} \sum_{i=1}^m \min\left(g(d_2, \ldots, d_S), \alpha_i\right)$$

$$\text{where } g(d_2, \ldots, d_S) = \frac{1}{2\sigma_i^2} \left\{ \sum_{s=1}^S n_s(\mu'_{si} - d_s)^2 - \frac{1}{n} \left[ \sum_{s=1}^S (n_s(\mu'_{si} - d_s)) \right]^2 \right\}$$

$$d_{sj} = d_s + d'_{sj}, s = 1, \ldots, S$$

$$\gamma_{si} = \begin{cases} 0 & \text{if } g(d_2, \ldots, d_S) < \alpha_i \\ \mu'_{si} - \mu'_{1i} - d_s & \text{otherswise} \end{cases}$$

$$\mu_i = \begin{cases} (1/n) \sum_{s=1}^S n_s(\mu'_{si} - d_s) & \text{if } g(d_2, \ldots, d_S) < \alpha_i \\ \mu'_{1i} & \text{otherwise} \end{cases}$$

where the only computationally intensive step is to minimize a sum of truncated quadratic functions in $d_2, \ldots, d_S$

$$d_2, \ldots, d_S = \underset{d_2,\ldots,d_S}{\arg\min} \sum_{i=1}^m \min\{g(d_2, \ldots, d_S), \alpha_i\}.$$

Methods for choosing $\{\alpha_i\}_{i=1}^m$ and for estimating $\{\sigma_i^2\}_{i=1}^m$, as well as experiments on simulated and real data, are given in [23].

## A1.2 Algorithms described in Section 2.3

**Algorithm 1** A general algorithm for minimizing (2.6).

```
procedure ALGORITHM.GENERAL($f_1, \dots, f_n$)
    for $i = 1 : n$ do
        Find region $C_i$ such that $f_i(\mathbf{x}) \leq 0$ on $C_i$.
    end for
    Find all the pieces $\{A_j\}_{j=1}^m$ in the partition of $R^d$ formed by $\{C_i\}_{i=1}^n$.
    $s \leftarrow 0$.
    for $j = 1 : m$ do
        Find the set of functions $\{f_k\}_{k \in I_j}$ that are not truncated on $A_j$.
        $s \leftarrow \min\{s, \min_{\mathbf{x}} \sum_{k \in I_j} f_k(\mathbf{x})\}$.
    end for
    return $s$.
end procedure
```

**Algorithm 2** An algorithm for minimizing (2.6) in 1-D.

```
procedure ALGORITHM.1D($f_1, \dots, f_n$)
    for $i = 1 : n$ do
        Find the interval $C_i = [l_i, r_i] \subset R$ such that $f_i(x) \leq 0$ on $C_i$.
    end for
    Order all the $2n$ end-points of $\{C_i\}_{i=1}^n$ along the real line as $p_1 < \cdots < p_{2n}$.
    $s \leftarrow 0, I \leftarrow \emptyset$.
    for $j = 1 : 2n$ do
        if $p_j$ is the left end-point of an interval $C_k$ then
            Add $k$ to set $I$.
        else if $p_j$ is the right end-point of an interval $C_k$ then
            Remove $k$ from set $I$.
        end if
        $s \leftarrow \min\{s, \min_{x} \sum_{k \in I} f_k(x)\}$.
    end for
    return $s$.
end procedure
```

**Algorithm 3** An algorithm for minimizing (2.6) in 2-D.

```
procedure ALGORITHM.2D($f_1, \ldots, f_n$)
    for $i = 1 : n$ do
        Find $C_i \subset R^2$ such that $f_i(\mathbf{x}) \leq 0$ on $C_i$.
        Find $\partial C_i$, the boundary $C_i$.
    end for
    $s \leftarrow 0$.
    for $i = 1 : n$ do
        Find all the intersection points of $\partial C_i$ and $\partial C_k, k \neq i$.
        Sort all the intersection points along $\partial C_i$ clockwise as $\mathbf{p}_1, \ldots, \mathbf{p}_{n_i}$.
        Find a point $\mathbf{p}$ between $\mathbf{p}_1$ and $\mathbf{p}_{n_i}$ on $\partial C_i$.
        $I \leftarrow \{k : \mathbf{p} \in C_k\}, J \leftarrow I \setminus \{i\}$.
        for $j = 1 : n_i$ do
            if $\mathbf{p}_j$ is the intersection point of $\partial C_i$ and $\partial C_k$ and $k \in I$ then
                Remove $k$ from sets $I$ and $J$.
            else if $\mathbf{p}_j$ is the intersection point of $\partial C_i$ and $\partial C_k$ and $k \notin I$ then
                Add $k$ to sets $I$ and $J$.
            end if
            $s \leftarrow \min\{s, \min_{\mathbf{x}} \sum_{k \in I} f_k(\mathbf{x}), \min_{\mathbf{x}} \sum_{k \in J} f_k(\mathbf{x})\}$.
        end for
    end for
    return $s$.
end procedure
```

**Algorithm 4** A cyclic coordinate descent algorithm for minimizing (2.6) in high-dimensional settings.

```
procedure ALGORITHM.HIGH-D($f_1, \ldots, f_n$)
    Initialize $\mathbf{x}$ as $\mathbf{x}_0$.
    while true do
        for $j = 1 : d$ do
            Fix all $x_k, k \neq j$, minimize the objective function as a univariate function of $x_j$ using Algorithm 2.
        end for
        if the change in $\mathbf{x}$ since the last iteration is less than a given tolerance level then
            return $\mathbf{x}$.
        end if
    end while
end procedure
```

### A1.3 The Θ-IPOD algorithm for robust linear regression

**Algorithm 5** The Θ-IPOD algorithm for robust linear regression, adapted from Algorithm 2 in [45].

```
procedure Θ-IPOD($\mathbf{X} \in R^{n\times p}, \mathbf{y} \in R^n, \boldsymbol{\lambda} > 0 \in R^n, \boldsymbol{\gamma}^{(0)} \in R^p$, and threshold operator $\Theta(\cdot;\cdot)$
which is taken as the hard-threshold operator $\Theta_h(\cdot;\cdot)$ in our paper)
    $\boldsymbol{\gamma} \leftarrow \boldsymbol{\gamma}^{(0)}, \mathbf{H} \leftarrow \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T, \mathbf{r} \leftarrow \mathbf{y} - \mathbf{H}\mathbf{y}$.
    while true do
        $\boldsymbol{\gamma} \leftarrow \Theta_h(\mathbf{H}\boldsymbol{\gamma} + \mathbf{r}; \sqrt{\boldsymbol{\lambda}})$.
        if the change in $\boldsymbol{\gamma}$ since the last iteration is less than a given tolerance level then
            return $\hat{\boldsymbol{\gamma}} \leftarrow \boldsymbol{\gamma}$ and $\hat{\boldsymbol{\beta}} \leftarrow (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T(\mathbf{y} - \hat{\boldsymbol{\gamma}})$.
        end if
    end while
end procedure
```

### A1.4 The difference of convex (DC) functions algorithm

Following [2], we rewrite our objective function for sum of truncated quadratic functions

$$f(\mathbf{x}) = \sum_{i=1}^{n} \min\left(\frac{1}{2}\mathbf{x}^T\mathbf{A}_i\mathbf{x} + \mathbf{x}^T\mathbf{b}_i + c_i, \lambda_i\right)$$

as $f(\mathbf{x}) = f_1(\mathbf{x}) - f_2(\mathbf{x})$, where

$$f_1(\mathbf{x}) = \sum_{i=1}^{n} \frac{1}{2}\mathbf{x}^T\mathbf{A}_i\mathbf{x} + \mathbf{x}^T\mathbf{b}_i + c_i$$

is a quadratic function, and

$$f_2(\mathbf{x}) = \sum_{i=1}^{n} \left(\frac{1}{2}\mathbf{x}^T\mathbf{A}_i\mathbf{x} + \mathbf{x}^T\mathbf{b}_i + c_i - \lambda_i\right)_+ .$$

Then, the DC algorithm iteratively minimizes a convex majorization of $f(\cdot)$ by replacing $f_2(\cdot)$ with its linear approximation at $\mathbf{x}^k$, until converge. That is,

$$\mathbf{x}^{k+1} = \arg\min_{\mathbf{x}} \left\{ f_1(\mathbf{x}) - [\nabla f_2(\mathbf{x}^k)]^T(\mathbf{x} - \mathbf{x}^k) \right\},$$

where $\nabla f_2(\mathbf{x}^k)$ is the gradient of $f_2(\mathbf{x})$ evaluated at $\mathbf{x}^k$, and we have

$$\nabla f_2(\mathbf{x}^k) = \sum_{i=1}^{n} 1\left(\frac{1}{2}\mathbf{x}^T\mathbf{A}_i\mathbf{x} + \mathbf{x}^T\mathbf{b}_i + c_i > \lambda_i\right)(\mathbf{A}_i\mathbf{x}^k + \mathbf{b}_i).$$

Therefore,

$$\mathbf{x}^{k+1} = \arg\min_{\mathbf{x}} \sum_{i=1}^{n} \frac{1}{2}\mathbf{x}^T\mathbf{A}_i\mathbf{x} + \mathbf{x}^T\mathbf{b}_i + c_i -$$

$$\sum_{i=1}^{n} 1\left(\frac{1}{2}\mathbf{x}^T\mathbf{A}_i\mathbf{x} + \mathbf{x}^T\mathbf{b}_i + c_i > \lambda_i\right)(\mathbf{A}_i\mathbf{x}^k + \mathbf{b}_i)^T(\mathbf{x} - \mathbf{x}^k)$$

for which we only need to minimize a quadratic function, and the solution exists in closed-form.

### A1.5 The iterative marginal optimization (IMO) algorithm for signal and image restoration

Following [41], we rewrite our objective function

$$f(\mathbf{x}) = \sum_{i=1}^{d}(x_i - y_i)^2 + w\sum_{i=1}^{n}\min\{x_i - x_{i+1})^2, \lambda\}.$$

as

$$f(\mathbf{x}) = ||\mathbf{H}\mathbf{x} - \mathbf{y}||^2 + w\sum_{i=1}^{n}\min\{(\boldsymbol{\phi}_i^T\mathbf{x})^2, \lambda\},$$

where $n = d - 1, \mathbf{H} = \mathbf{I}_n$ is an identity matrix, $\boldsymbol{\Phi} = (\boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_n)^T \in R^{n\times d}$ with $\phi_{i,i} = -1, \phi_{i,i+1} = 1$ and otherwise $\phi_{i,j} = 0$ for all $i$ and $j$. We then minimize $f(\mathbf{x})$ using the following iterative algorithm proposed in [41], where $\Theta_h(\cdot;\cdot)$ is the hard-threshold operator.

**Algorithm 6** The iterative marginal optimization (IMO) algorithm for signal and image restoration, Adapted from Algorithm 1 in [41].

```
procedure THRESHOLD($\mathbf{y} \in R^d, \boldsymbol{\Phi} \in R^{n\times d}, \boldsymbol{\lambda} > 0 \in R^n$)
    $\mathbf{x} \leftarrow \mathbf{y}$.
    while true do
        $\mathbf{b} \leftarrow \boldsymbol{\Phi}\mathbf{x}$.
        $\mathbf{a} \leftarrow \Theta_h(\mathbf{b}; \sqrt{\boldsymbol{\lambda}})$.
        $\mathbf{z} \leftarrow w\boldsymbol{\Phi}^T\mathbf{a}$.
        $\mathbf{x} \leftarrow (\mathbf{H}^T\mathbf{H} + w\boldsymbol{\Phi}^T\boldsymbol{\Phi})^{-1}(\mathbf{H}^T\mathbf{y} + \mathbf{z})$.
        if the change in $\mathbf{x}$ since the last iteration is less than a given tolerance level then
            return $\mathbf{x}$.
        end if
    end while
end procedure
```

## A1.6 Proofs

*Proof of Proposition 2.1.*

To minimize (2.3),

$$f(\boldsymbol{\beta},\boldsymbol{\gamma}) = \sum_{i=1}^{n}(y_i - \mathbf{x}_i^T\boldsymbol{\beta} - \gamma_i)^2 + \lambda\sum_{i=1}^{n} 1(\gamma_i \neq 0),$$

notice that the minimization with respect to $\boldsymbol{\gamma}$ can be performed componentwise. For each $\gamma_i$, if $\gamma_i = 0$, we have

(5.2)
$$f(\boldsymbol{\beta},\gamma_1,\ldots,\gamma_i=0,\ldots,\gamma_n) = \sum_{j\neq i}\left\{(y_j - \mathbf{x}_i^T\boldsymbol{\beta} - \gamma_j)^2 + \lambda 1(\gamma_j \neq 0)\right\} + (y_i - \mathbf{x}_i^T\boldsymbol{\beta})^2.$$

On the other hand, if $\gamma_i \neq 0$, we have

$$f(\boldsymbol{\beta},\gamma_1,\ldots,\gamma_i\neq 0,\ldots,\gamma_n) = \sum_{j\neq i}\left\{(y_j - \mathbf{x}_i^T\boldsymbol{\beta} - \gamma_j)^2 + \lambda 1(\gamma_j \neq 0)\right\} + (y_i - \mathbf{x}_i^T\boldsymbol{\beta} - \gamma_i)^2 + \lambda,$$

which is minimized at $\gamma_i = y_i - \mathbf{x}_i^T\boldsymbol{\beta}$, that is,

$$(5.3)\quad f(\boldsymbol{\beta},\gamma_1,\ldots,\gamma_i = y_i - \mathbf{x}_i^T\boldsymbol{\beta},\ldots,\gamma_n) = \sum_{j\neq i}\left\{(y_j - \mathbf{x}_i^T\boldsymbol{\beta} - \gamma_j)^2 + \lambda 1(\gamma_j \neq 0)\right\} + \lambda.$$

Comparing (5.2) with (5.3), it is easy to see that we should choose $\gamma_i = 0$ if $(y_i - \mathbf{x}_i^T\boldsymbol{\beta})^2 < \lambda$ and $\gamma_i = y_i - \mathbf{x}_i^T\boldsymbol{\beta}$ othersise. Plugging the value of $\gamma_i$ into (2.3), we have

$$\begin{aligned} f(\boldsymbol{\beta},\boldsymbol{\gamma}) &= \sum_{i=1}^{n}\left[(y_i - \mathbf{x}_i^T\boldsymbol{\beta})^2 1\{(y_i - \mathbf{x}_i^T\boldsymbol{\beta})^2 < \lambda\} + \lambda 1\{(y_i - \mathbf{x}_i^T\boldsymbol{\beta})^2 \geq \lambda\}\right] \\ &= \sum_{i=1}^{n}\min\{(y_i - \mathbf{x}_i^T\boldsymbol{\beta})^2, \lambda\} \end{aligned}$$

which is the objective function $g(\boldsymbol{\beta})$ in Proposition 2.1. □

*Proof of Proposition 2.2.*

Similar to the proof of Proposition 2.1, for the objective function in (2.4), if $\gamma_i = 0$, the $i$-th summand becomes $b(\mathbf{x}_i^T\boldsymbol{\beta}) - (\mathbf{x}_i^T\boldsymbol{\beta})y_i$. Otherwise, if $\gamma_i \neq 0$, the $i$-th summand becomes $b(\mathbf{x}_i^T\boldsymbol{\beta} + \gamma_i) - (\mathbf{x}_i^T\boldsymbol{\beta} + \gamma_i)y_i + \lambda$, which is minimized when

$$y_i = \frac{\partial b(\mathbf{x}_i^T\boldsymbol{\beta} + \gamma_i)}{\partial \gamma_i} = g^{-1}(\mathbf{x}_i^T\boldsymbol{\beta} + \gamma_i) \Rightarrow \mathbf{x}_i^T\boldsymbol{\beta} + \gamma_i = g(y_i)$$

which makes the $i$-th summand become $\lambda^* := b(g(y_i)) - g(y_i)y_i + \lambda$. The objective function can then be rewritten as:

$$\sum_{i=1}^{n} \min\{b(\mathbf{x}_i^T\boldsymbol{\beta}) - (\mathbf{x}_i^T\boldsymbol{\beta})y_i, \lambda^*\}$$

which completes the proof. $\square$

*Proof of Proposition 2.4.*

Let $b_1, \ldots, b_n$ be $n$ Boolean variables, i.e., each $b_k$ only takes one of two possible values: TRUE or FALSE. For a 3-SAT problem $P$, suppose its formula is

$$f(b_1, \ldots, b_n) = c_1 \wedge \cdots \wedge c_m,$$

where $\wedge$ is the logical OR operator, and $\{c_i\}_{i=1}^m$ are the clauses[1] of P with

$$c_i = (l_{i1} \vee l_{i2} \vee l_{i3}),$$

where $\vee$ is the logical AND operator, and $\{l_{ij}\}_{i=1}^m, j \in \{1, 2, 3\}$, are literals of $P$. Each literal $l_{ij}$ is either a variable $b_k$ for which $l_{ij}$ is called a positive literal, or the negation of a variable $\neg b_k$ for which $l_{ij}$ is called a negative literal. Without loss of generality, suppose that each clause consists of exactly three literals, and that the three literals in each clause correspond to three distinct variables. The 3-SAT problem $P$ concerns about the satisfiability of $f(b_1, \ldots, b_n)$, i.e., whether there exists

[1]A clause is a disjunction of literals or a single literal. In a 3-SAT problem each clause has exactly three literals.

a possible assignment of values of $b_1, \ldots, b_n$ such that $f(b_1, \ldots, b_n) = \text{TRUE}$.

We reduce the 3-SAT problem $P$ to the minimization of a sum of truncated convex functions $g(\mathbf{x}) : R^n \to R$ as follows. Let $\mathbf{x} = (x_1, \ldots, x_n) \in R^n$ with each $x_k$ corresponds to a $b_k$ such that $b_k = \text{TRUE}$ if and only if $x_k > 0$. For each clause $c_i = (l_{i1} \vee l_{i2} \vee l_{i3})$ of $P$, define a sum of seven truncated convex functions

$$g_i(\mathbf{x}) = \sum_{t=1}^{7} \min(g_{it}(\mathbf{x}), 1)$$

where

$$g_{it}(\mathbf{x}) = \begin{cases} 0 & \text{if } \mathbf{x} \in S_{it1} \cap S_{it2} \cap S_{it3} \\ \infty & \text{otherwise} \end{cases}$$

where $S_{itj}$ is one of the two half-spaces defined by $x_k > 0$ and $x_k \leq 0$, respectively, where $x_k$ is the variable corresponding to $l_{ij}$, that is, $l_{ij} = b_k$ or $l_{ij} = \neg b_k$. We choose $S_{itj}$ as the half-space defined by $x_k > 0$ if and only if $(b(j,t) - \frac{1}{2})$ has the same sign as $l_{ij}$, where $b(j,t)$ is the $j$-th digit (from left to right) of $t$ when $t \in \{1, \ldots, 7\}$ is represented as three binary digits. For instance, for a clause $c_i = (b_1 \vee \neg b_2 \vee b_3)$, we have

$$g_{i1}(\mathbf{x}) = \begin{cases} 0 & \text{if } x_1 \leq 0, x_2 > 0, x_3 > 0 \\ \infty & \text{otherwise} \end{cases}$$

and

$$g_{i7}(\mathbf{x}) = \begin{cases} 0 & \text{if } x_1 > 0, x_2 \leq 0, x_3 > 0 \\ \infty & \text{otherwise} \end{cases}$$

Since all the half-spaces, as well as their intersections, are convex sets, all the $g_{it}(\mathbf{x})$'s are convex functions. Furthermore, since the regions in which $g_{it}(\mathbf{x}) = 0, t \in \{1, \ldots, 7\}$, are disjoint, it is easy to verify that $g_i(\mathbf{x})$ can only take one of two possible values

$$g_i(\mathbf{x}) = \begin{cases} 6 & \text{if } c_i \text{ is satisfied by the assigned values of } b_1, \dots, b_n \\ 7 & \text{otherwise} \end{cases}$$

where we choose $b_k = \text{TRUE}$ if and only if $x_k > 0$. The reduction is then completed by noticing that the 3-SAT problem $P$ is satisfiable if and only if the minimum value of the function $g(\mathbf{x}) = \sum_{i=1}^{m} g_i(\mathbf{x})$ is $6m$, and that it is easy to see that the reduction can be done in polynomial time.

□

*Proof of Proposition 2.5.*

On one hand, we have

$$\min_{\mathbf{x}} \sum_{i=1}^{n} \min\{f_i(\mathbf{x}), 0\} = \min_{j} \min_{\mathbf{x} \in A_j} \sum_{k \in I_j} f_k(\mathbf{x}) \geq \min_{j} \min_{\mathbf{x}} \sum_{k \in I_j} f_k(\mathbf{x}), \tag{5.4}$$

On the other hand, we have

$$\begin{aligned} \min_{j} \min_{\mathbf{x}} \sum_{k \in I_j} f_k(\mathbf{x}) &\geq \min_{j} \min_{\mathbf{x}} \sum_{k \in I_j} \min\{f_k(\mathbf{x}), 0\} \\ &\geq \min_{j} \min_{\mathbf{x}} \sum_{i=1}^{n} \min\{f_i(\mathbf{x}), 0\} \\ &= \min_{\mathbf{x}} \sum_{i=1}^{n} \min\{f_i(\mathbf{x}), 0\} \end{aligned} \tag{5.5}$$

Putting (5.4) and (5.5) together, we have

$$\min_{\mathbf{x}} \sum_{i=1}^{n} \min\{f_i(\mathbf{x}), 0\} = \min_{j} \min_{\mathbf{x}} \sum_{k \in I_j} f_k(\mathbf{x}).$$

□

## A1.7 Supplementary figures and tables

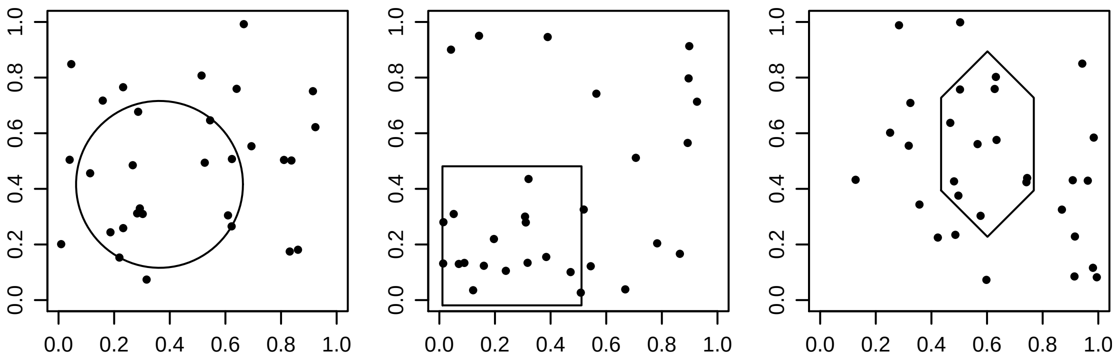

Figure A1: Placement of different convex shapes to cover the maximum number of points uniformly sampled from the unit square.

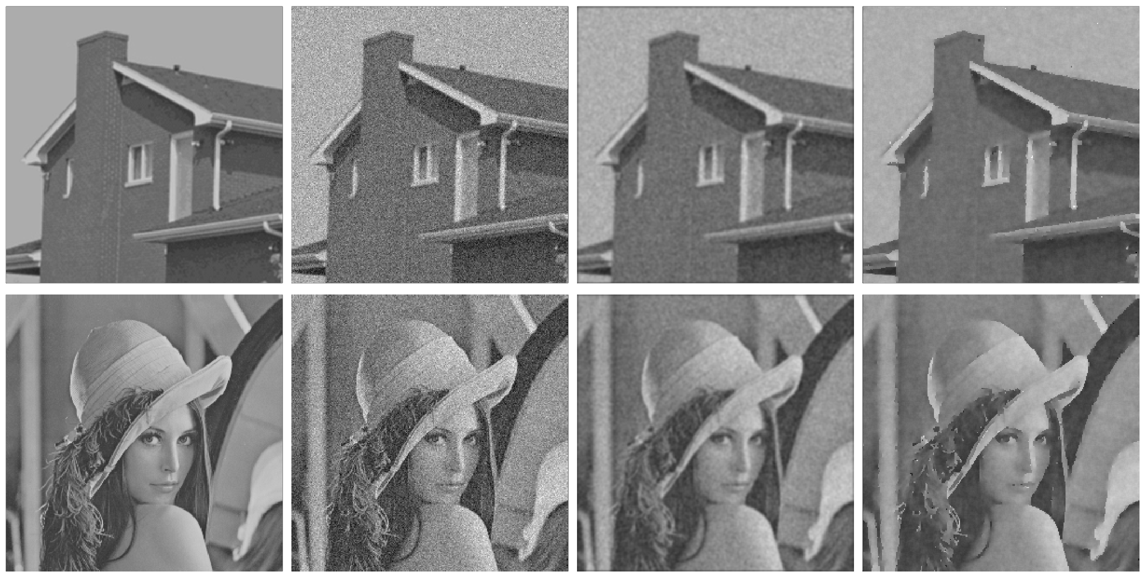

Figure A2: Restoration of images. For each row, from left to right: original image, image with Gaussian noise added, image restored using Gaussian smoothing with a $5 \times 5$ kernel and image restored using proposed algorithm.

Table A1: Comparison of different methods for outlier detection in simple linear regression. The table shows the leverages of outliers (L), percents of outliers (O%) and mean percents of masking and swamping for all the methods using 100 simulated replicates. The standard errors of the means are given in parentheses.

Masking

| L | O% | MM | LTS | GY | IPOD | Proposed |
|---|---|---|---|---|---|---|
| 0 | 5 | 0.6 (0.3) | 0.6 (0.3) | 1.0 (0.4) | 0.6 (0.3) | 0.8 (0.4) |
| 0 | 10 | 2.3 (0.5) | 2.1 (0.5) | 2.4 (0.5) | 2.0 (0.5) | 2.0 (0.5) |
| 0 | 20 | 1.8 (0.3) | 1.8 (0.3) | 1.9 (0.3) | 1.5 (0.3) | 1.4 (0.3) |
| 0 | 30 | 3.1 (0.3) | 2.6 (0.3) | 3.2 (0.3) | 2.3 (0.3) | 2.3 (0.3) |
| 0 | 45 | 11.1 (0.5) | 4.8 (0.3) | 9.8 (0.4) | 2.5 (0.3) | 2.5 (0.3) |
| 0 | 60 | 33.2 (0.5) | 24.3 (0.5) | 33.0 (0.5) | 34.4 (0.7) | 2.8 (0.4) |
| 20 | 5 | 2.0 (0.6) | 2.4 (0.7) | 2.0 (0.6) | 1.8 (0.6) | 1.8 (0.6) |
| 20 | 10 | 3.1 (0.6) | 2.8 (0.6) | 3.1 (0.6) | 2.7 (0.5) | 2.8 (0.5) |
| 20 | 20 | 4.8 (0.7) | 2.8 (0.4) | 3.7 (0.5) | 3.2 (0.5) | 2.8 (0.4) |
| 20 | 30 | 14.1 (1.0) | 5.7 (0.6) | 8.9 (0.7) | 5.5 (0.7) | 3.9 (0.4) |
| 20 | 45 | 34.5 (0.7) | 13.8 (0.8) | 29.6 (0.9) | 18.0 (1.4) | 7.5 (0.9) |
| 20 | 60 | 34.0 (0.7) | 36.1 (0.5) | 34.9 (0.6) | 35.0 (0.5) | 24.1 (1.3) |

Swamping

| L | O% | MM | LTS | GY | IPOD | Proposed |
|---|---|---|---|---|---|---|
| 0 | 5 | 1.0 (0.1) | 1.4 (0.1) | 1.0 (0.1) | 1.1 (0.1) | 1.0 (0.1) |
| 0 | 10 | 1.1 (0.1) | 1.2 (0.1) | 1.1 (0.1) | 1.2 (0.1) | 1.2 (0.1) |
| 0 | 20 | 1.3 (0.1) | 1.3 (0.1) | 1.3 (0.1) | 1.4 (0.1) | 1.3 (0.1) |
| 0 | 30 | 1.3 (0.1) | 1.3 (0.1) | 1.5 (0.2) | 1.4 (0.1) | 1.3 (0.2) |
| 0 | 45 | 19.0 (1.1) | 2.9 (0.3) | 15.1 (0.8) | 1.4 (0.1) | 1.3 (0.1) |
| 0 | 60 | 96.1 (0.6) | 73.2 (1.4) | 95.5 (0.6) | 90.5 (1.8) | 2.0 (0.6) |
| 20 | 5 | 1.5 (0.1) | 1.8 (0.1) | 1.4 (0.1) | 1.5 (0.1) | 1.5 (0.1) |
| 20 | 10 | 1.1 (0.1) | 1.3 (0.1) | 1.1 (0.1) | 1.2 (0.1) | 1.3 (0.1) |
| 20 | 20 | 1.2 (0.1) | 1.1 (0.1) | 1.1 (0.1) | 1.1 (0.1) | 1.1 (0.1) |
| 20 | 30 | 4.0 (0.4) | 1.6 (0.2) | 1.9 (0.2) | 1.6 (0.1) | 1.4 (0.1) |
| 20 | 45 | 30.2 (0.8) | 4.5 (0.3) | 20.7 (0.7) | 8.6 (0.9) | 2.4 (0.3) |
| 20 | 60 | 44.9 (0.9) | 34.8 (0.9) | 42.8 (0.9) | 33.0 (1.3) | 13.5 (1.1) |

Table A2: Stopping criteria of the simulation studies in Sections 2.4.2 and 2.4.4

| | Maximal number of function evaluations | Maximal number of iterations | Tolerance | Maximal steps when no improvement in the estimate |
|---|---|---|---|---|
| DIRECT | $10^4$ | - | $10^{-8}$ | - |
| StoGO | $10^4$ | - | $10^{-8}$ | - |
| SA | $10^4$ | $10^4$ | - | $10^6$ |
| PSO | $10^4$ | $10^4$ | $10^{-8}$ | - |
| IMO/DC | | $10^4$ | $10^{-8}$ | - |
| Proposed (high-D) | | $10^4$ | $10^{-8}$ | - |

Table A3: Comparison of different algorithms for global optimization of the sum of 50 randomly generated truncated quadratic functions in 2-D. The table shows the complexities of the functions ($C$) as well as mean success rates (in percents) and running times (in seconds) for all the methods using 100 simulated replicates. The standard errors of the means are given in parentheses.

Success Rate

| C | DIRECT | StoGO | SA | PSO | DC | Proposed |
|---|---|---|---|---|---|---|
| 1 | 100.0 (0.0) | 86.0 (3.5) | 98.0 (1.4) | 99.0 (1.0) | 27.0 (4.5) | 100.0 (0.0) |
| 5 | 99.0 (1.0) | 74.0 (4.4) | 97.0 (1.7) | 93.0 (2.6) | 9.0 (2.9) | 100.0 (0.0) |
| 10 | 88.0 (3.3) | 57.0 (5.0) | 85.0 (3.6) | 72.0 (4.5) | 1.0 (1.0) | 100.0 (0.0) |

Running Time

| C | DIRECT | StoGO | SA | PSO | DC | Proposed |
|---|---|---|---|---|---|---|
| 1 | 0.45 (0.01) | 2.76 (0.02) | 0.40 (0.00) | 2.29 (0.05) | 0.50 (0.03) | 3.07 (0.03) |
| 5 | 0.42 (0.01) | 2.62 (0.07) | 0.39 (0.00) | 2.66 (0.08) | 2.95 (0.29) | 2.62 (0.03) |
| 10 | 0.44 (0.05) | 2.41 (0.02) | 0.37 (0.00) | 2.82 (0.12) | 8.04 (0.71) | 2.35 (0.03) |

# Bibliography

[1] Arvind Agarwal and Hal Daumé III. Generative kernels for exponential families. In *AISTATS*, pages 85–92, 2011.

[2] Le Thi Hoai An and Pham Dinh Tao. Solving a class of linearly constrained indefinite quadratic problems by dc algorithms. *Journal of global optimization*, 11(3):253–285, 1997.

[3] Gill Barequet, Matthew Dickerson, and Petru Pau. Translating a convex polygon to contain a maximum number of points. *Computational Geometry*, 8(4):167–179, 1997.

[4] Douglas Bates and Dirk Eddelbuettel. Fast and elegant numerical linear algebra using the RcppEigen package. *Journal of Statistical Software*, 52(5):1–24, 2013.

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

[6] Emmanuel J Candès and Terence Tao. The power of convex relaxation: Near-optimal matrix completion. *Information Theory, IEEE Transactions on*, 56(5):2053–2080, 2010.

[7] Bernard Marie Chazelle and Der-Tsai Lee. On a circle placement problem. *Computing*, 36(1-2):1–16, 1986.

[8] Guanhua Chen, Donglin Zeng, and Michael R Kosorok. Personalized dose finding using outcome weighted learning. *Journal of the American Statistical Association*, 111(516):1509–1521, 2016.

[9] Marcin Cieslik et al. The use of exome capture rna-seq for highly degraded rna with application to clinical cancer sequencing. *Genome Research*, 2015.

[10] Stephen A Cook. The complexity of theorem-proving procedures. In *Proceedings of the third annual ACM symposium on Theory of computing*, pages 151–158. ACM, 1971.

[11] Chao Cui et al. Infectious disease modeling and innate immune function in zebrafish embryos. *Methods in Cell Biology*, 105:273–308, 2011.

[12] Mark De Berg, Marc Van Kreveld, Mark Overmars, and Otfried Cheong Schwarzkopf. Computational geometry. In *Computational geometry*, pages 1–17. Springer, 2000.

[13] D. Eberly. Intersection of ellipses, 2000. `https://www.geometrictools.com/Documentation/IntersectionOfEllipses.pdf`, Last Modified: 06/23/2015.

[14] Dirk Eddelbuettel. *Seamless R and C++ Integration with Rcpp*. Springer, New York, 2013. ISBN 978-1-4614-6867-7.

[15] Dirk Eddelbuettel and Romain François. Rcpp: Seamless R and C++ integration. *Journal of Statistical Software*, 40(8):1–18, 2011.

[16] Jianqing Fan and Runze Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American statistical Association*, 96(456):1348–1360, 2001.

[17] Irène Gannaz. Robust estimation and wavelet thresholding in partially linear models. *Statistics and Computing*, 17(4):293–310, 2007.

[18] Alexandros Georgogiannis. Robust k-means: a theoretical revisit. In *Advances in Neural Information Processing Systems*, pages 2883–2891, 2016.

[19] Daniel Gervini and Victor J Yohai. A class of robust and fully efficient regression estimators. *Annals of Statistics*, pages 583–616, 2002.

[20] Eligius MT Hendrix, G Boglárka, et al. *Introduction to nonlinear and global optimization*. Springer New York, 2010.

[21] Guifeng Jiang and D. Jed Harrison. mrna isolation in a microfluidic device for eventual integration of cdna library construction. *The Analyst*, 125, 2000.

[22] Hui Jiang and Julia Salzman. A penalized likelihood approach for robust estimation of isoform expression. *Statistics and Its Interface*, 8:437–445, 2015.

[23] Hui Jiang and Tianyu Zhan. Unit-free and robust detection of differential expression from rna-seq data. *Statistics in Biosciences 9 (1), 178-199*, 2017.

[24] Donald R Jones, Cary D Perttunen, and Bruce E Stuckman. Lipschitzian optimization without the lipschitz constant. *Journal of Optimization Theory and Applications*, 79(1):157–181, 1993.

[25] Richard M Karp. *Reducibility among combinatorial problems*. Springer, 1972.

[26] Shota Katayama and Hironori Fujisawa. Sparse and robust linear regression: An optimization algorithm and its statistical properties. *arXiv preprint arXiv:1505.05257*, 2015.

[27] Eija Korpelainen et al. *RNA-seq Data Analysis*. CRC Press, 2015.

[28] Charity W Law et al. voom: precision weights unlock linear model analysis tools for rna-seq read counts. *Genome Biology*, 2014.

[29] Yoonkyung Lee, Steven N MacEachern, and Yoonsuh Jung. Regularization of case-specific parameters for robustness and efficiency. *Statistical Science*, pages 350–372, 2012.

[30] Tzu-Ying Liu and Hui Jiang. Minimizing sum of truncated convex functions and its applications. *Journal of Computational and Graphical Statistics*, 2018.

[31] Kaj Madsen and Serguei Zertchaninov. *A new branch-and-bound method for global optimization*. IMM, Department of Mathematical Modelling, Technical Universityof Denmark, 1998.

[32] RARD Maronna, Douglas Martin, and Victor Yohai. *Robust statistics*. John Wiley & Sons, Chichester. ISBN, 2006.

[33] Lauren McCann and Roy E Welsch. Robust variable selection using least angle regression and elemental set sampling. *Computational Statistics & Data Analysis*, 52(1):249–257, 2007.

[34] Abraham Mehrez and Alan Stulman. The maximal covering location problem with facility placement on the entire plane. *Journal of Regional Science*, 22(3):361–365, 1982.

[35] R Garey Michael and S Johnson David. Computers and intractability: a guide to the theory of np-completeness. *WH Free. Co., San Fr*, 1979.

[36] Nasser M Nasrabadi, Trac D Tran, and Nam Nguyen. Robust lasso with missing and grossly corrupted observations. In *Advances in Neural Information Processing Systems*, pages 1881–1889, 2011.

[37] Mila Nikolova. Thresholding implied by truncated quadratic regularization. *IEEE Transactions on Signal Processing*, 48(12):3437–3450, 2000.

[38] Mila Nikolova. Energy minimization methods. In *Handbook of mathematical methods in imaging*, pages 139–185. Springer, 2011.

[39] Mila Nikolova, Michael K Ng, and Chi-Pan Tam. Fast nonconvex nonsmooth minimization methods for image restoration and reconstruction. *IEEE Transactions on Image Processing*, 19(12):3073–3088, 2010.

[40] Juvencio Santos. Nobre and Julio da Motta Singer. Residual analysis for linear mixed models. *Biometrical Journal*, 2007.

[41] Javier Portilla, Antonio Tristán-Vega, and Ivan W Selesnick. Efficient and robust image restoration using multiple-feature l2-relaxed sparse analysis priors. *IEEE Transactions on Image Processing*, 24(12):5046–5059, 2015.

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

[43] Jürgen Richter-Gebert. *Perspectives on projective geometry: A guided tour through real and complex geometry*. Springer Science & Business Media, 2011.

[44] Peter J Rosseeuw and Annick M Leroy. Robust regression and outlier detection. *Wiley Series in Probability and Mathematical Statistics, New York: Wiley*, 1, 1987.

[45] Yiyuan She and Art B Owen. Outlier detection using nonconvex penalized regression. *Journal of the American Statistical Association*, 2012.

[46] Xiaotong Shen, Wei Pan, and Yunzhang Zhu. Likelihood-based selection and

sharp parameter estimation. *Journal of the American Statistical Association*, 107(497):223–232, 2012.

[47] Julie Tibshirani and Christopher D Manning. Robust logistic regression using shift parameters. In *ACL (2)*, pages 124–129, 2014.

[48] Rand R Wilcox. Robust testing procedures. *Encyclopedia of Statistics in Behavioral Science*, 2005.

[49] Daniela M Witten. Penalized unsupervised learning with outliers. *Statistics and its Interface*, 6(2):211, 2013.

[50] Yang Xiang, Sylvain Gubian, Brian Suomela, and Julia Hoeng. Generalized simulated annealing for global optimization: the gensa package. *R Journal*, 5(1):13–28, 2013.

[51] Victor J Yohai. High breakdown-point and high efficiency robust estimates for regression. *The Annals of Statistics*, pages 642–656, 1987.

[52] Mauricio Zambrano-Bigiarini and Rodrigo Rojas. A model-independent particle swarm optimisation software for model calibration. *Environmental Modelling & Software*, 43:5–25, 2013.

[53] Anatoly Zhigljavsky and Antanas Žilinskas. *Stochastic global optimization*, volume 9. Springer Science & Business Media, 2007.