ABSTRACT

YANG, MENG. Selected Topics in Statistical Computing and Genomic Data Analysis. (Under the direction of Eric C. Chi and Jung-Ying Tzeng).

Biclustering approaches are able to recover checkerboard structure accurately and efficiently like Spectral Biclustering and Convex Biclustering. As we know, outliers are common in the practical problems and they will seriously bias the estimation sometimes. A natural question is: Are we still able to recover the checkerboard structure accurately and efficiently even with the appearance of the outliers? This article aims to answer this question by incorporating the minimum distance estimator into the biclustering problem assuming the outliers follow a mean shift outlier model. It has been shown that minimum distance estimation (L2E) is more robust than maximum likelihood estimation (MLE) in the parametric modeling problem. Moreover, L2E has been applied in various areas and shown a strong resistance against outliers. Motivated by these previous work, we hope the newly constructed method can not only reveal the hidden checkerboard structure behind a data matrix but also has good resistance to the outliers. In the Chapter 2, we demonstrate the advantages of our approach via simulations on both artificial data and real microarray lung cancer data and the disadvantages.

Integrative multi-omics analyses, which integrate complementary level of information from different molecular events, has great potentials to detecting novel disease genes and pathways, enhancing risk prediction and elucidating disease mechanisms. The importance of jointly modeling multi-omics data has been recognized, with one major focus on leveraging the relationships among different platforms and identifying gene-sets associated with clinical outcomes. A common strategy for the gene-set integrative analysis is to regress the clinical outcomes on all genomic variables in a gene set. However, the joint modeling methods encounter the challenges of high-dimensionality inference, especially the sample size is usually moderate either due to research resources or missing data. In this work, we propose a tensor-based framework to enhance model efficiency while retaining the variable-wise resolution. By accounting for the inherent matrix
structure of an individual’s multi-omics data to reduce the number of parameters, the proposed tensor methods naturally incorporate the relationship among genomic effects and boost the efficiency of high-dimensional modeling. We explore the variable-specific test procedure under tensor regression framework. We derive an alternative variance formula of the coefficient estimator that does not depend on the permutation matrix as required in previous work, which permits a systematic expression of the variance formula in computational coding. We evaluate the performance of the tensor-based test using simulations and demonstrate its utility by applying to the datasets from the Cancer Genome Atlas (TCGA).
DEDICATION

To my parents.
BIOGRAPHY

The author grew up in Beijing, the capital of China. He spent his four years in Nankai University, an famous university in Tianjin, China, to obtain his bachelor’s degree in Financial Engineering. Meanwhile, based on his interest, he also obtained a minor in applied Mathematics by completing all of the main courses and several selective courses in the Mathematics department in Nankai University. After the graduation in summer in 2012, he came to U.S. to pursue his Master and Ph.D. degrees in North Carolina State University. He majored in Financial Mathematics in the Mathematics department in North Carolina State University, but one year later, he realized he has lost his interest in Finance and more interested in Statistics than any other subjects because he believed it is tool letting the data talk that matters most. Therefore, he decided to change his major to Statistics. Having decided to change major, He worked very hard to learn Statistics as much as possible while he was in Financial Mathematics in North Carolina State University. After two years hard working, he finally managed to get an admission to the Master program in Statistics in North Carolina State University but also successfully complete his courses in Financial Mathematics program. One year later after passing the Basic exam, he transferred from the Master program to the Doctoral program in Statistics in North Carolina State University.

During his amazing but stressing five years in Doctoral program, he not only roughly completed the dissertation projects for dissertation, but also he did consulting work for a post-doc in Electrical Engineering department regarding the experimental design issue and got involved in other projects. Learning how to do research is not easy task for him since he had no experience at all before joining the Doctoral problem. He spent years to learn how to do research. But still, he never gave up and finally came the defense date. Despite of those difficulties, he did System Program Analyst intern in RENCI, a computer company belonging to the University of North Carolina at Chapel Hill in summer in 2016. One year later in the summer, he did the Data Analyst intern in LENOVO. Starting from May 2018, he became a GRA student working with
SAS during the summer in 2018 and the following Fall semester till graduation. He obtained a return offer as Research Statistician Developer, where he joined a small team working the Advanced Modeling in SAS. He defended his dissertation on 12/14/2018.
ACKNOWLEDGEMENTS

I would like to thank my two main advisors, Eric Chi and Jung-Ying Tzeng, for all of their guidance, knowledge, patience, understanding over the last few years. I would also like to thank Arvind Saibaba and Wenbin Lu for serving on my committee.

Dr. Tzeng effectively served as a co-advisor. I very much appreciate her careful and detailed guidance during the whole project. The project is not as easy as we expect before we start since it has many numerical issues when computing the variance and optimizing. I was thinking to give up several times but I am fortunate that I did not and have her as one of co-advisor. Not only did she guide me how to do research and solve potential issues, but also she also taught me how to keep calm and continue working without stressing too much. Moreover, she always encouraged me even though I made some careless mistakes or reported work with poor quality. She emphasized the importance of documentations, which can be a great help for a years-long project. When I applied internship, she and her husband also help me with my resume and provided many invaluable advice and finally I grabbed the chance of being a GRA student in SAS, a great company everyone in Statistics department in NCSU longing for. I am very fortunate that Dr. Chi joined the NC State faculty and served also as a co-advisor on my committee. I really appreciate his emphasis on Statistics Computing, Software Developer and Mathematics. Since I lack research experience before joining the Doctoral program, he focuses on teaching me how to be a good independent researcher during the project. He told what the real researcher is supposed to do and shared many of his previous experiences to me when he was a student. Besides, he also told me how to review papers and shared many of his precious previous working samples to me. Working with him has definitely made me a more effective researcher.

There are numerous other current and former NC State staff and faculty members I would like to thank for the knowledge and support they provided me during graduate school. I would like to thank Tao Pang, Jeffery Scroggs and Donald Martin for their recommendation letters
for letting me get admission from Master program of NC State University. It is their trust that made me have a chance to learn more about Statistics and changed my life. I had a wonderful time when I was involved Dr. Martin’s wonderful courses, which laid a solid foundations in basic statistics inference theory and time series theory. Not only did Dr. Martin provided me good courses, but also he also shared me his experience when he was applying tenure-track professor position when he just graduated when he saw me stressed and worried about application to the Statistics department being rejected. It is my great honor to meet him when I study aboard and I appreciate what he did to me when I was apply both Master and Doctorate programs. I would also want to thank Kim Weems for her advice and patience when I was TA in her courses. Dr. Weems gave me many advice of being a good TA and also she also provided me many information regarding the questions about application of the Master and Doctorate programs in NC State University. Lastly, I want to thank Alison McCoy for all that she has done for me.

Last and most importantly, I truly believe that I would not manage to finish my degree without my unbelievably amazing family and friends, whom I am immensely and eternally grateful for. Above all, I would like to thank my wife, Die Hu, who supported me both mentally and Materially over the years when I was struggling in the Doctorate program. I would also like to thank my Mother, Aihua Meng, who has given me such a huge leg up in life, and for that I will be forever grateful. In additional, I would like to thank my friend, Yueyang Huang, a current Ph.D. student in Bioinformatics who provided me endless supports in terms of biology and also many invaluable advices regarding all kinds of things whenever I lost. I would like to especially thank Guojing Hou, a formal Ph.D. student in Nuclear Engineering, and his wife, Yanming Li, a formal Ph.D. student in Physics, for helping me with my personal statement. I would not have had a wonderful and touchy personal statement without your help and I will be grateful eternally. I can still feel the passion after reading the personal statement written years ago. Also, I would like to thank my two private English tutors for helping me with my speaking, Chris Goodings and Susan Edleburg. I have gained my confidence while speaking English and now I can communicate in English fluently. I would not have achieved this without your help. You
have my gratitude. Finally, I would like also to thank my two personal trainers, Ricky McCoy and Christina Tallman, who provided me many advice regarding body fat reduction and muscle build.
TABLE OF CONTENTS

LIST OF TABLES ........................................................................ x

LIST OF FIGURES ...................................................................... xi

Chapter 1 Introduction ................................................................. 1
  1.1 Standard Linear Regression Model ...................................... 3
  1.2 Penalized Linear Regression Model ..................................... 6
  1.3 Biclustering And Convex Biclustering ................................. 8
  1.4 Multi-Platform Genomic Data Analysis ............................... 10

Chapter 2 Robust Biclustering: A method based on the Minimum Distance Estimator .................................................. 14
  2.1 Introduction ..................................................................... 14
  2.2 A Brief Introduction to the Parametric Minimum $L_2$ Error Criterion .................................................. 18
  2.3 A Brief Introduction to MM Algorithm .............................. 20
  2.4 $L_2$ Error Biclustering Criterion and Its Properties .......... 21
  2.5 Connection to the M-estimator ........................................... 26
  2.6 Estimation of Biclusters with LEBRA .............................. 27
  2.7 Robustness of LEBRA ...................................................... 30
  2.8 Convergence of LEBRA ................................................... 31
  2.9 Model Selection ............................................................ 32
  2.10 Simulation Studies .......................................................... 34
  2.11 Application to Genomics ................................................. 40
  2.12 Discussion ................................................................. 46

Chapter 3 Hypothesis Testing for Tensor Regression with Application to Gene-set Integrative Analysis of Multi-omics Data .................................................. 48
  3.1 Abstract ..................................................................... 48
  3.2 Introduction ................................................................. 49
  3.3 Methods ..................................................................... 52
    3.3.1 Notation and Preliminaries ...................................... 52
    3.3.2 Tensor Regression Model for Integrative Gene-Set Analysis .................................................. 54
    3.3.3 Estimation ............................................................ 55
    3.3.4 Association Test for Single Genomic Variables ........ 58
  3.4 Simulation Studies .......................................................... 59
    3.4.1 Performance of Detecting Causal Variables ............ 60
    3.4.2 Performance of Detecting Causal Variables ............ 61
  3.5 Real Data Analysis .......................................................... 66
  3.6 Discussion ................................................................. 69
  3.7 Future work ............................................................... 72

Chapter 4 TCGA Data Processing Pipeline ........................................ 73
### LIST OF TABLES

| Table 3.1 | Performance of detecting causal genomic variables of tensor regression (TR), linear model (LM), and LASSO with effect size $C = 0.25$, sample size $n = 500$ and $B$ with dimension $3 \times 74$ based on 50,000 replications. For TR and LM, a variable is selected as significant if its p-value is less than $\alpha = 0.05/(3 \times 74) = 0.00023$. | 63 |
| Table 3.2 | This table shows the simulation results for "I" under 1000 replications, 74 genes and 500 sample size. Bonferroni correction is not used. | 65 |
| Table 3.3 | This table shows the simulation results for "I", "Random" and "T" under 1000 replications, 170 genes, 500 sample size and an appropriate effect size. Bonferroni correction is not used. | 66 |
| Table 3.4 | KEGG_P53_SIGNALING_PATHWAY (R = 3) | 67 |
LIST OF FIGURES

Figure 1.1 Blue line is the regression line based on clean data. Red line is the regression line based on contaminated data .......................................................... 5
Figure 1.2 The objective functions in $\mathbb{R}^2$ of Ridge Regression and Lasso and their support from two angles. The region lies in dark red cylinder is the support of ridge regression, yet the dark green cuboid is the support for lasso. The purple dot is the OLS estimate. The contours of OLS objective function is in rainbow color. Whatever approaches is looking for the global minimum of the OLS objective function, yet penalized regression approaches are looking for the global minimum within the support defined by $\mathcal{J}$. .................................................. 7
Figure 1.3 Different groups are labeled with different colors. We have $n$ objects and $p$ features. The data matrix is a $n$-by-$p$ matrix .......................................................... 8
Figure 1.4 Two categories under the "joint-modeling" based analysis: tensor based (left side) and transformation based (right side). There are other categories such as iBAG (model based, Wang, Baladandayuthapani, Morris, Broom, Manyam, and Do (2012)). .......................................................... 12
Figure 2.1 Compete MM procedure looking for the minimum value of $x_\infty = \arg\min_x f(x)$, The thick solid curve is $f(x)$. The dash curve are majorants based on each update $x_i$ ........................................................................................................... 21
Figure 2.2 Row Size: 100, 30, 30, 40; Column size: 50, 60, 50, 40; No outliers ............... 36
Figure 2.3 Row Size: 10, 100, 50, 40; Column size: 10, 100, 50, 40; No outliers ............... 37
Figure 2.4 Row Size: 100, 30, 30, 40; Column size: 60, 60, 60, 50. The number of outliers: 4000 .................................................................................................................. 38
Figure 2.5 Row Size: 10, 100, 50, 40; Column size: 10, 100, 50, 40. The number of outliers: 4000 .................................................................................................................. 39
Figure 2.6 Snap shots of the LEBRA solution path of the lung cancer dataset as the penalty parameter $\lambda$ increases. All of the matrices have been forced to have the same column order and row order for comparison purpose. It shows the whole trend of the LEBRA solution path from the estimate with not enough strong penalty strength to the over-smooth constant estimate. ......................... 41
Figure 2.7 Performance of COBRA, LEBRA and SpBC under the measure of Rand Index, Variational Information, Adjusted Rand Index and Normalized Mutual Information .................................................................................................................. 43
Figure 2.8 Running time of the previous simulation experiment needed by COBRA, LEBRA and SpBC .................................................................................................................. 44
Figure 2.9 Performance of COBRA, LEBRA and SpBC for different magnitude of outliers .................................................................................................................. 45
Figure 3.1 Reverse "I" shape ($R = 1$); "T" shape ($R = 2$); "Random" shape ($R = 2$, similar to I but with different rank). .......................................................... 61
Figure 3.2 Bias of variance estimates for the tensor coefficient estimates with bias = estimated variance – empirical variance. The 3 panels from left to right represent 3 different signal patterns in B: "I" shape (R = 1), "Random" shape (R = 2), "T" shape (R = 2); . Orange boxplots are biases for B_{pg} = 0 and blue boxplots are biases for B_{pg} ≠ 0. 62

Figure 4.1 Interpretation of the sample ID (https://docs.gdc.cancer.gov) 75

Figure 4.2 DNA methylation overview. Methylation occurs at CpG dinucleotides regulated by DNA methyltransferases (Dnmts). Dnmt1 is predominantly associated with methylation of hemimethylated CpG dinucleotides while Dnmt3a and Dnmt3b contribute to de novo methylation. 76

Figure 4.3 The Structure of data dnamethy_UCEC.Rdata 76

Figure 4.4 First section of methylation data processing: data cleaning process. Green rectangular represents intermediate files generated throughout the first section. Light blue rectangular represents the bash script files implementing each sub step. Detailed bash scripts will be provided in the Appendix C. 78

Figure 4.5 Second section of methylation data processing: splitting and selection. Green rectangular represents intermediate files generated throughout the first section. Yellow rectangular represents the bash script files implementing each sub step. Detailed bash scripts will be provided in the Appendix C. 79

Figure 4.6 “Central dogma” of molecular biology describes two steps: (1) Transcription: a segment of DNA is transcribed into either coding RNA or non-coding RNA; (2) Translation: coding RNA is further translated into polypeptide then protein. 80

Figure 4.7 The structure of transcripts_UCEC.Rdata 81

Figure 4.8 Transcripts data set process procedure. Blue rectangular represents the file that processes the whole process in the dashed border rectangular. Green rectangular represents the files that are needed or produced during or after the whole data process procedure. The detailed R script are given in the Appendix C. 82

Figure 4.9 Changes in order and number of genes A, B, C and D vary the human genome, which makes us different. The right bottom is the copy number variations (CNVs) 84

Figure 4.10 CNVs data processing and splitting procedure 86

Figure A.1 Additional Measures for Figure 2.2. Row Size: 100, 30, 30, 40; Column size: 50 60 50 40. The number of outliers = 0. Other five measures show the similar results as the four measures we reported in the paper. 109

Figure A.2 Additional Measures for Figure 2.3. Row Size: 10 100 50 40; Column size: 10 100 50 40. The number of outliers = 0. Other five measures show the similar results as the four measures we reported in the paper. 110

Figure A.3 Additional Measures for Figure 2.5. Row Size: 10, 100, 50, 40; Column size: 10, 100, 50, 40. The number of outliers: 4000. We can see JI has the similar problem as VI, which has been discussed. 111
Figure A.4 Additional Measures for Figure 2.4. Row Size: 100, 30, 30, 40; Column size: 60, 60, 60, 50. The number of outliers: 4000. We can see JI has the similar problem as VI, which has been discussed.

Figure A.5 Additional Measures for Figure 2.7. JI behaves similar to VI and the reason has been discussed. Other additional measures exhibit similar results to the four we reported.

Figure A.6 Additional Measures for Figure 2.9. Other five measures show the similar results as the four measures we reported in the paper.

Figure B.1 order-1 tensor: vector; order-2 tensor: matrix; order-3 tensor: which is a cute and we do not have specific names for any higher dimensional array.

Figure B.2 Box product vs Kronecker product.

Figure B.3 Rank selection results for BIC and AIC of Shape "T". Index is the indices of 1000 replicaitons.

Figure B.4 "Triangle" shape with true rank 3.

Figure B.5 Rank selection results for BIC and AIC of Shape "Triangle". Index is the indices of 1000 replicaitons.
Chapter 1

Introduction

People compute. Restaurant owners always seek to predict how much customers will consume to know how much food they need to prepare. Investors aim to create the best portfolios to maximize their profits while maintaining a relatively low risk. Furthermore, meteorologists recently adjusted the parameters of their own models to predict the possible path of Hurricane Florence.

Nature computes. The solar system tends to seek balance by adjusting the position of each of its celestial bodies. Rays of light always find the routes that minimize their travel time. Moreover, green plants face the direction of sunlight.

Statisticians have always been frequent users of computing resources, which makes computing such an important tool in statistics. The branch of Statistics that focuses on computational research is called Statistical Computing. Statistical Computing is one of the most important areas in Statistics recently. This is because the implementation is always a required and inevitable topic regarding any of the statistical theories. People originally used calculators for statistical computing to find the mean and variance as well as solve other elementary equations in the past. With the rapid development of computers and AI, machines rather than humans have been performing more and more statistical calculations such as predicting, fitting, and testing. Statistical Computing is an old but novel direction in statistics. It is old because the studies regarding computation in statistics have been around for decades. However, it is novel because
there are more potentialities since the advent of increasingly advanced technologies that have made what were originally impossible issues in statistical computing solvable. For example, computers that are strong enough can generate random numbers so that the Monte Carlo method becomes a feasible and simple way to solve many problems, e.g. evaluating the integral of a certain function. MCMC has now become an important research area in Statistical Computing.

The main goal of the Statistical Computing is to optimize an objective function accurately and efficiently. Given a practical problem, the procedure of figuring out the objective function, constraints, and variables of interest is called modeling. It is important to think twice before constructing any models. If the model is too simple, the solution will lose its validity in the application. However, if the model is too complexed, it might take years to figure out the solution which makes the modeling pointless. The forms of the objective function not only reflects the goal and the problem of interest of the researcher but also the definition of being a "good" model. A good model can make the optimization procedure much easier and scalable which will be explained in the next paragraph.

Once the model has been established. An algorithm is needed according to the model. The procedure of optimizing an objective function created in the modeling procedure is called an algorithm. Simply put, an algorithm is an instruction for human or machines telling what should be done in every single iteration, it is like a blueprint. There does not exist a universal algorithm that can solve all of the models accurately and efficiently. Different algorithms should be applied properly when it comes to different models. In addition, there is always a tradeoff between efficiency and accuracy. Develop a proper algorithm is an art of design. A proper algorithm can not only utilize the structure of the given problem but also cater to the research interest. Particularly, some algorithms are good at giving an accurate answer while other algorithms might be capable of providing a less accurate answer but at an extremely fast speed. The decision of using which kinds of an algorithm depends on the purpose of the study. If prediction accuracy is our main interest, we might choose the one giving the accurate answer. However, if we are only interested in a rough idea of the solutions, we might just choose the one giving answers faster.
A comprehensive understanding of the problems is needed to design a proper algorithm.

After the decision of choosing an algorithm being made, it is also important to let the machine be able to recognize the solution. There are many ways to let a machine do so. Empirically, relative or absolute changing rate of the objective function or the distance between the current estimate and previous estimate can be checked. Also, more rigorously, KKT condition can be applied as well.

My research area is related to the Statistical Computing. More specifically, my research mainly constitutes two topics. One is the application of the low-rank tensor regression to TCGA data analysis. More specifically, we studied the parameter-wise hypothesis testing procedure by investigating the behavior of the test statistics under the tensor regression framework. The other topic discussed the robustness of the biclustering algorithm. The objective function has been redesigned so that the resulting algorithm has stronger resistance to the outliers than traditional approaches. Besides, a simple algorithm has been developed to optimize the newly formulated objective function. Both projects are closely related to the optimization issue in linear regression, penalized linear regression, and their extensions. In the following section, I will briefly give an overview regarding linear regression, penalized regression and their usage in my projects.

1.1 Standard Linear Regression Model

Consider the standard linear regression model for \(i\)-th observation, \(i = 1, \ldots, n\),

\[
y_i = \mathbf{x}_i^T \mathbf{\beta} + \epsilon_i
\]  

(1.1)

where \(n\) is the sample size, \(y_i, i = 1, 2, \ldots, n\) is the response variable for \(i\)-th sample. If we have \(p\) predictors/variables, \(\mathbf{x}_i \in \mathbb{R}^p\). \(\mathbf{x}_i\) is the realization of all \(p\) predictors for \(i\)-th sample whose \(j\)-th element is \(x_{ij}\). \(\mathbf{\beta}\) is the \(p\)-by-1 unknown parameter vector. The goal of the algorithm solving standard linear regression model is to estimate \(\mathbf{\beta}\). In matrix notation, we can re-write the (1.1)
into a compact form:

$$y = X\beta + \epsilon \quad (1.2)$$

where

$$y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix},$$

$$X = \begin{bmatrix} x_{11} & x_{12} & x_{13} & \ldots & x_{1p} \\ x_{21} & x_{22} & x_{23} & \ldots & x_{2p} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & x_{n3} & \ldots & x_{np} \end{bmatrix},$$

$$\beta = \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_p \end{bmatrix},$$

$$\epsilon = \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_n \end{bmatrix}.$$

$y$ is an $n$-by-$1$ response variable. It can be any continuous type. For example, $y$ can be any traits in genetic data analysis like status, pattern or growth. We call an $n$-by-$p$ matrix, $X$, design matrix. Each column of the design matrix represents a realization of a predictor, specifically, a genetic variable in genetic data analysis. We are interested in the association analysis between a certain trait and a group of genes. In such case, each predictor can be interpreted as a measure of a gene, which is a column of a design matrix. $\epsilon$ is the random error vector following i.i.d. gaussian distribution with mean zero and variance $\sigma^2$. $\beta$ is the $p$-by-$1$ parameter vector. The typical method to estimate the parameter vector $\beta$ is called Ordinary Least Square (OLS), whose goal is to estimate $\beta$ by compute the optimal point of the following optimization problem:

$$\hat{\beta} = \arg \min_{\beta \in \mathbb{R}^p} \frac{1}{2} ||y - X\beta||_2^2 \quad (1.3)$$

The intuition behind (1.3) is to let the model take care of all the samples $\{y_i, x_i^T\}_{i=1}^n$ by minimizing the squared distance of each observation, $(y_i - x_i^T\beta)^2$. After setting the first order derivative to be zero, (1.3) has the well-known solution, $\hat{\beta} = (X^TX)^{-1}X^Ty$. Although the solution from OLS solving standard linear regression model is substantially neat and elegant, it still have many drawbacks. One of the well-known drawback of OLS estimate is that it lacks the resistance to the outliers since the OLS is based on the Euclidean Distance, $|| \bullet ||^2$. Several outliers can seriously bias the estimates, which can be seen clearly in the following figure:
In Figure 1.1, the blue solid line is the standard linear regression model based on the data with outliers labeled with blue triangles, yet the red solid line is the standard linear regression model based on the data without outliers labeled with red dots. It is clearly shown that the blue line is dragged towards those outliers. The other drawback of an OLS estimate is that it does not work when the number of parameters is larger than the sample size, that is $n < p$. There are various ways to understand the issues caused by $n < p$. $p > n$ will cause the Gram matrix, $X^TX$ rank deficient. As a result, the OLS estimate is not unique and its stability is weak due to the inversion of a matrix that is close to singular. One way to improve the standard linear regression model is to add additional constraints. So the corresponding optimization problem becomes:

$$\hat{\beta} = \arg \min_{J(\beta) \leq t} \frac{1}{2} ||y - X\beta||^2_2$$

(1.4)

(1.4) searches solution within a certain region specified by $J$, yet OLS seeks the solution in the whole space. The unconstraint form of (1.4) is called the penalized regression model and will be
1.2 Penalized Linear Regression Model

The Equivalent unconstraint form of (1.5) can be written as

$$\hat{\beta} = \arg\min_\beta \frac{1}{2} ||y - X\beta||_2^2 + \lambda J(\beta)$$  \hspace{1cm} (1.5)

where $\lambda \in \mathbb{R}_+$ is called the tuning or regularization parameter. $\lambda$ controls the strength of the penalty. The larger the $\lambda$ is, the stronger the penalty is. A large value of $\lambda$ is equivalent to set a small value of $t$ in (1.4). Thus, when $\lambda = 0$, the penalized regression model is equivalent to the standard linear regression model. when $\lambda = \infty$, it means the model actually ignores the squared loss part of the objective function, $\frac{1}{2} ||y - X\beta||_2^2$. The form of $J$ is a user specified and depends on the model purpose. The most common two choices are $J = || \cdot ||_2^2$ and $J = || \cdot ||_1$ and they are called ridge regression and LASSO, respectively. Lasso and Ridge regression are looking for the "small" $\hat{\beta}$ in terms of $|| \cdot ||_2^2$ or $|| \cdot ||_1$. The differences of lasso and ridge regression are the shape of support, which can be easily visualized in Figure 1.2.

The typical optimization procedure for solving the ridge regression is the same as the procedure for solving a standard linear regression model since the ridge penalty term can be absorbed into the squared loss part. It can be easily seen in the following simple algebra:

$$\min_\beta \frac{1}{2} ||y - X\beta||_2^2 + \lambda ||\beta||_2^2$$

$$= \min_\beta \frac{1}{2} ||y - X\beta||_2^2 + \lambda ||0 - I\beta||_2^2$$

$$= \min_\beta \frac{1}{2} \left( \begin{pmatrix} y \\ 0 \end{pmatrix} - \begin{pmatrix} X \\ \lambda I \end{pmatrix} \beta \right)^2$$

$$= \min_\beta \frac{1}{2} ||y^* - X^*\beta||_2^2$$
Figure 1.2: The objective functions in $\mathbb{R}^2$ of Ridge Regression and Lasso and their support from two angles. The region lies in dark red cylinder is the support of ridge regression, yet the dark green cuboid is the support for lasso. The purple dot is the OLS estimate. The contours of OLS objective function is in rainbow color. Whatever approaches is looking for the global minimum of the OLS objective function, yet penalized regression approaches are looking for the global minimum within the support defined by $\mathcal{J}$.

where $\mathbf{y}^* = \begin{pmatrix} \mathbf{y} \\ 0 \end{pmatrix}$ and $\mathbf{X}^* = \begin{pmatrix} \mathbf{X} \\ \mathbf{X} \lambda \mathbf{I} \end{pmatrix}$. Ridge regression has been converted to a standard linear regression.

As for lasso, there is no explicit solution in general (we do only when the columns of the $\mathbf{X}$ are orthogonal to each other). However, there are many iterative approaches works quite well for solving it such as proximal gradient algorithm, ADMM, MM, even gradient descent algorithm plus some additional conditions during the iteration), which have been discussed in details in many of the textbooks. Therefore we will omit them in the introduction section.

The objective function of the standard linear regression and the penalized regression can be extended to more general fitting settings.

$$
\min_{\mathbf{B}} \frac{1}{2} \| \mathbf{Y} - \mathbf{X}\mathbf{B} \|_F^2 + \lambda \mathcal{J} (\mathbf{B})
$$

(1.6)
Many of the problem can be converted into an optimization problem using (1.6) as the objective function such as matrix approximation, low-rank decomposition, and image smoothing. One of the most important extensions is the convex formulated clustering and biclustering problem.

1.3 Biclustering And Convex Biclustering

Speaking of Biclustering, it is requisite to cover some basics of clustering at first. Clustering is definitely an important fundamental data mining technique. The goal of the clustering is to group a set of objects according to some certain similarity measure or rule such as the Euclidean distance. Clustering has been applied in many areas such as image recognition and unsupervised genetic data analysis. Mathematically, the standard clustering problem can be formulated as follows: Suppose we have \( n \) objects, \( \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n \). We aim to cluster \( n \) objects according to some similarity measure based on \( p \) features, which means each of the \( \mathbf{x}_i \) is a \( p \)-by-1 vector whose \( j \)-th element is the score in terms of \( j \)-th feature. Due to such formulation, we can form a \( n \)-by-\( p \) data matrix, \( \mathbf{X} \in \mathbb{R}^{n \times p} \), whose rows are samples and columns are features. The interpretations of rows and columns can vary. For example, rows can also represents different experimental conditions. The goal of clustering in terms of the data matrix setting is just to group rows according to some similarity measure based on the given \( p \)-features and shown in Figure 1.3.

Figure 1.3: Different groups are labeled with different colors. We have \( n \) objects and \( p \) features. The data matrix is a \( n \)-by-\( p \) matrix.
The view of clustering problem as grouping the rows of a data matrix make it much easier to generalize to a concept of biclustering.

Biclustering is a problem of clustering seeking homogeneous groups simultaneously in both columns and rows. Unlike the clustering looking for latent structure hinged on the whole feature space, Biclustering allows the existence of the grouping based on the subspace of the whole feature space. Simply put, biclustering clusters a data matrix along both rows and columns simultaneously, thus the name "Bi-clustering".

There has been an influx of application examples of biclustering such as voting data (Hartigan, 1972), Collaborative filtering recommendation system (Elnabarawy, Wunsch, and Abdelbar, 2016) and Lung cancer data (Yun and Yi, 2013; Chen, Zou, Lu, and Chen, 2014; Chi, Allen, and Baraniuk, 2017), as reviewed in Xie, Ma, Fennell, Ma, and Zhao (2018). Meanwhile, there has been also plenty of biclustering algorithms suggested, as summarized in Prelić, Bleuler, Zimmermann, Wille, Bährmann, Gruissem, Hennig, Thiele, and Zitzler (2006); Eren, Deveci, Kütüktünc, and Çatalyürek (2012); Oghabian, Kilpinen, Hautaniemi, and Czeizler (2014); Pontes, Giráldez, and Aguilar-Ruiz (2015); Roy, Bhattacharyyya, and Kalita (2015). In the genomic data analysis, the most popular biclustering algorithm is the clustered dendrogram. Hierarchical clustering (Hastie, Tibshirani, and Friedman, 2009) was performed on the patients (columns) and on the genes (rows), respectively. As pointed out by Chi et al. (2017), the clustered dendrogram, as a biclustering algorithm, has two properties that make it less appropriate for generating reproducible results. First, a greedy search algorithm is used to construct the dendrogram so that the clustered dendrogram will provide a local optima result in terms of a certain error criterion. Therefore, the algorithm depends heavily on the initial setting. The algorithm is also not robust in the sense that small perturbations in the data can lead to completely different biclustering result, not to mention outliers.

The underlying issue of the many biclustering algorithms is the non-convex formulation of the error criterion. In order to solve this issue, a convex formulated biclustering algorithm was proposed by Chi et al. (2017) under a checkerboard latent structure assumption. It has a convex,
non-constraint objective error criterion:

$$\min_U \frac{1}{2} \|X - U\|_F^2 + \lambda J(U) \tag{1.7}$$

where $U$ is a $n$-by-$p$ matrix needed to be estimated, $X$ is a $n$-by-$p$ data matrix. (1.7) is a special case of (1.6). It can be viewed as a penalized regression but with an identity design matrix and multiple response variables. The penalty $J$ steers the solution toward a checkerboard pattern. (1.7) is convex as a function of $U$. Consequently, it has a unique solution, which makes the estimate from (1.7) does not depend on the initial setting. Moreover, Chi et al. (2017) proved the solution to (1.7) is jointly continuous in $(X, \lambda)$. Continuity in $U$ makes (1.7) more stable against small perturbations. However, (1.7) still lack resistance to the outliers. As shown in Figure 1.1, outliers can still bias the solution to (1.7) due to the characteristic of Euclidean distance as a measure of similarity. The chapter 2 of this dissertation proposed a new formulation of the biclustering objective error criterion while sacrificing the convexity in exchange for the great robustness under the checkerboard pattern assumption.

Except for biclustering, association study in genetic data analysis is also an important direct application of standard linear regression and penalized regression problem.

### 1.4 Multi-Platform Genomic Data Analysis

Understanding the genetic variations behind complex traits has always been a goal for many researchers. The advent of advanced technologies in data generated from multiple levels of biological systems containing DNA sequence data, RNA expression levels data, methylation pattern data, other epi-genetics markers has been generating a larger and larger amount of data sets. Such a revolution in the availability of the data sets across all platforms holds great promise. Yet, the gap between the speed of data generation and the in-depth analysis of those multi-platform data sets are increasing and being mentioned frequently (Ebrahim, Brunk, Tan, O’brien, Kim, Szubin, Lerman, Lechner, Sastry, Bordbar, et al. (2016)). A lot of approaches has
been proposed to detect the hidden genetic variation behind complex traits. For example, DNA sequences variation can be detected by the linkage analysis and through association studies. Moreover, the association studies between phenotypes and variations in other high-throughput platform datasets such as RNA levels, methylation pattern sequencing, and protein variation comes more and more popular and is treated as routine. At first, each platform of the data has been analyzed separately to look for the association between the phenotypes and genetic variation, yet with these methods for multiple platforms data sets, we have already assembled different pieces of puzzles from each platform of data into a big picture. Still, there are much of complicated biological process behind certain trait remains unclear, which could be due to the current disadvantages of the methodologies for multi-platform data sets.

Development of the methodologies for analyzing multiple platforms data sets is relatively slow and quite challenging for several reasons. One of the reasons is that the sample size is always small compared to the number of genes. Although costs per unit measurement are going down due to the new data collection method, the cost of the total volume of data continues to increase. Therefore, the ratio of sample size to the number of parameters is always problematic. There has already been a rich amount of studies regarding methodologies analyzing multiple platform datasets try to solve bad ratio issue of the sample size to the number of parameters. They can be roughly classified into two categories: (a) "meta"-based methods, which combine the results of multiple single-gene (Louhimo and Hautaniemi (2011); Xiong, Mukherjee, and Furey (2014)), single gene-set, or single platform analyses (Ma and Huang (2009); Ma, Huang, and Song (2011); Lin, Zhang, Li, He, Deng, and Wang (2014)); and (b) "joint-modeling"-based methods, which regress the clinical outcomes on all genomic variables from multiple platforms in a gene set (e.g., Tyekucheva, Marchionni, Karchin, and Parmigiani (2011); Huang, Huang, Shia, and Ma (2012); Wang et al. (2012); Hu and Tzeng (2014); Chu and Huang (2017)). Model-based approaches tend to outperform meta-based approaches because they conduct simultaneous data integration across genes and platforms and account for the biological relationship among genomic variables. However, model-based methods encounter the challenges of high dimensional
genomic variables, especially the sample size is usually moderate either due to research resources or missing data. Various strategies have been proposed to address the issue of high-dimensional genomic variables, e.g., Bayesian approaches (e.g., Wang et al. (2012)), penalization regressions (e.g., Liu, Huang, and Ma (2013); Shi, Liu, Huang, Zhou, Shia, and Ma (2014); Ma and Huang (2009); Ma et al. (2011); Jiang, Shi, Zhao, Krauthammer, Rothberg, and Ma (2016)), and dimension-reduction based methods, e.g., principal component analysis (PCA) Meng, Zeleznik, Thallinger, Kuster, Gholami, and Culhane (2016). Joint-modeling based methods can be further classified into (1) "tensor-based" and (2) "transformation-based" approaches depends on the approaches of aligning data platforms.

Figure 1.4: Two categories under the "joint-modeling" based analysis: tensor based (left side) and transformation based (right side). There are other categories such as iBAG (model based, Wang et al. (2012)).

Risk of losing information is always one of the ongoing issue for multiple platforms genomic data analysis. Information can be lost when it flows through multiple steps in a multi-stated
analysis and depends on the exact procedure in each step. "Meta-based" methods have risking of losing information when it assumes the independence among different platforms, genes, or gene sets. Even in "joint-modeling"-based analysis that are based on a transformation, an inappropriate transformation of the data will project the data into a space with lower dimensions but incomplete information.

The most intuitive way to conduct the so-called "joint-modeling"-based analysis by pooling all of the datasets together and analyze them as a whole. One way to implement this is to use tensor. We can simply align the design matrices of multiple platforms into a big order-3 tensor directly. Such a strategy will make us face the bad ratio of sample size to the number of parameters directly. In chapter 3, we will try to provide an answer belonging to this category by utilizing the tensor concept directly along with low-rank approximation to alleviate the issue of the bad ratio of sample size to the number of parameters. Other than that issue, data integration is also challenging. Different platforms of data were quantified under different studies with different levels and standards. For example, the methylation data is measured under CpG level while the transcript data is measured on the transcript level, a more refined level than the gene level. Sloppy processing and integration of the data under different scales will mask the signal. Chapter 4 addresses the data processing and integration issues by forming an alignment workflow written in R and Bash script.
Chapter 2

Robust Biclustering: A method based on the Minimum Distance Estimator

2.1 Introduction

Biclustering, as known as co-clustering or two-mode clustering, is one of the most commonly used statistical tools in the analysis of high-dimensional datasets. Biclustering identifies and groups homogeneous objects but it allows the existence of sub-clusters within each bicluster. Biclustering has been used in a wide array of areas. For example, in text mining, biclustering can be used to detect subgroups of documents with similar properties with respect to a subgroup of words (Dhillon, 2001). In addition, in gene expression data, biclustering can be used to identify a subgroup of genes with similar conditions. A complete review for the biclustering with application to gene expression data can be found in (Beatriz Pontes, Raul Giraldez, 2015).

In this chapter, we assume the data can be partitioned into a checkerboard-like pattern. Many of the early biclustering approaches under the checkerboard-like assumption can cast as an optimization problem with hard thresholding constraints (Kluger, 2003). Mathematically,
this suggests an optimization problem:

$$\min_{U \in \mathbb{R}^{n \times p}} \frac{1}{2} ||X - U||_F^2$$

$$\sum_{i<j} 1_{U(:,j) \neq U(:,i)} \leq t$$

$$\sum_{i<j} 1_{U(j,:) \neq U(i,:)} \leq t$$

(2.1)

where $X \in \mathbb{R}^{n \times p}$, $U$ is the hidden checkerboard-like structure (a block matrix each of whose block is a constant matrix) and $t$ is the largest possible number of row/column clusters that allowed in the model. For example, for a 2-by-2 data matrix case, the hidden checkerboard-like structure $U$ looks like:

$$U = \begin{bmatrix} \mu_{11} 1_1^T & \mu_{12} 1_1^T \\ \mu_{21} 1_2^T & \mu_{22} 1_1^T \end{bmatrix}$$

where the size of $1_i, i = 1, \ldots, 4$ depends the size of bicluster and the scalar $\mu_{ij}$ is the centroid of the $(i, j)$ bicluster.

However, 2.1 is very sensitive to not only outliers but also even a small perturbation in the data. One of the reason is that hard thresholding constraints themselves are too strict since it dose not allow any “grey” area. Soft thresholding constraints have been proposed as an alternative in order to improve the stability of 2.1. As indicated by its name, soft threshold constraints relaxes the criterion by allowing a small difference, $\epsilon$. It suggests the following modified version of 2.1:

$$\min_{U \in \mathbb{R}^{n \times p}} \frac{1}{2} ||X - U||_F^2$$

$$\sum_{i<j} 1_{||U(:,j) - U(:,i)||_2 \leq \epsilon} \leq t$$

$$\sum_{i<j} 1_{||U(j,:) - U(i,:)||_2 \leq \epsilon} \leq t$$

(2.2)
Then the equivalent non-constraint weighted version of 2.2 can be written as:

$$\min_{U \in \mathbb{R}^{n \times p}} \frac{1}{2} ||X - U||_F^2 + \lambda \left( \sum_{i<j} \omega_{ij} ||U(:, j) - U(:, i)||_2 + \sum_{i<j} \tilde{\omega}_{ij} ||U(j, :) - U(i, :)||_2 \right)$$

where $\omega_{ij}$ and $\tilde{\omega}_{ij}$ are the weights to adjust the penalty strength of each pair-wise difference.

The very initial idea of the biclustering comes from the term “Direct Clustering” introduced by Hartigan (1972) with application to the republican vote data. The term “Biclustering” is later used by Mirkin. But the term “Biclustering” became more popular after Cheng and Church (2000) applying biclustering approach on a gene expression data set, one of the most important literatures in the gene expression biclustering field. Afterward, biclustering has been extensively studied and there are huge amount of work such as Plaid model Lazzeroni and Owen (2002), Spectral Biclustering Kluger, Basri, Chang, and Gerstein (2003), FLOC Yang, Wang, Wang, and Yu (2003), Coupled Two-way Iterative Method (CTWC) Getz, Levine, and Domany (2000), Non-smooth Non-Negative Matrix Factorization Carmona-Saez, Pascual-Marqui, Tirado, Carazo, and Pascual-Montano (2006), SSVD Lee, Shen, Huang, and Marron (2010), Sparse Biclustering Tan and Witten (2014) and COBRA Chi et al. (2017). Compared to clustering, biclustering has many advantages. One of the most important advantages over clustering is much more flexible which allows the existence of sub-group. For more detailed list of Biclustering work, please see the review papers written by Madeira and Oliveira (2004); Busygin, Prokopyev, and Pardalos (2008); Pontes et al. (2015)

While extensive efforts have been put on accelerating and stabilizing the biclustering algorithms, noticeably less attentions have been given to extending these methods to handle extreme perturbations, which is called “outliers.” A simple motivation to study robustness is from a natural question: given a data matrix with randomly located outliers, are we still able to recover the checkerboard-like pattern accurately and efficiently?

There are roughly two categories of approaches capable of dealing with outliers: preprocessing approach and automated approach. Preprocessing approach consists of two steps. The first step
identifies outliers by applying some techniques before applying any of the biclustering approaches to the dataset. The second step is to implement a biclustering algorithm on the result of the first step. This approach has a risk of losing information while throwing away so-called “outliers.” A more dedicated way is to use Robust Principal Component Analysis Partridge and Jabri (2000); Wright, Ganesh, Rao, Peng, and Ma (2009); Candès, Li, Ma, and Wright (2011) by assuming the outliers are the minority compared to sample size and any non-zero entries in the sparse matrix of the results will be treated as outliers. But the ugly part of this more dedicated approach is that we have to choose a tuning parameter. Additionally, all those approaches will vary the actual data set based on different pre-selected outlier-detectors. As a result, the actual data that a biclustering algorithm applied to will be very subjective. The major advantage of the automated approach is that the actual data set we are going to work with will not vary and it has less risk of losing information compared to abandon parts of the data labeled “outlier.” It is an automated process combining dealing with outliers and implementing biclustering algorithm. The approach we are going to present in this chapter is an automated approach.

Many of the error criterions (the objective function of a biclustering minimization problem) of the biclustering algorithm can be formulated as a Frobenius norm that measure the disparity between an estimation and dataset along with penalty terms that impose the sparsity on the solution (there are many other types of the biclustering, i.e., graph theory based). Usually, biclustering approaches are not doing great job when we have outliers since most of them are based on least square based which is known sensitive to atypical observations. Least square as a measure for distance actually aims to take care of all of the data: normal data and outliers as well. A natural extension of the current biclustering approach to gain robustness is to find an approach only take care of the normal data while ignoring all or part of the outliers. In this chapter, we construct such kind of the automated approach via the minimum distance approach. We will use the following model to construct the outliers.

Suppose we are given a data matrix $X$ following a mean shift outlier model with a normal
error term:

$$X = U + C + E$$  \hspace{1cm} (2.3)\

where $U$ is the ground truth having a checkerboard-like structure, $C$ is a sparse matrix that contains randomly located outliers, $E$ is a noise matrix that follows a multivariate distribution. When $E \sim N(0, \sigma^2 I)$, the data matrix $X$ follows a normal distribution with mean $U + C$ and variance $\sigma^2 I$. Particularly, when $C = 0$, there are a massive number of classical approaches that are capable of recovering the estimation that has the checkerboard-like structure accurately and efficiently. Are we still able to recover the checkerboard structure, $U$ as good as the results from the classical approaches while we have not only $E$ but also $C$? we will try to give an answer to this question.

The mean shift outlier model has been widely used as a start of many outlier related analysis. For example, it was used earlier for the outliers detection problem using the linear model Wei and Fung (1999); McCann and Welsch (2007); She and Owen (2011); Candès et al. (2011). Non-zero $C$ cases are not very uncommon in the real world. For example, tissue samples may be mislabeled or contaminated due to some careless reasons. It is also possible that the tissue sample dataset may exhibit different statistical properties under different situations that are not of interest. As a result, these outliers or non-constant statistical properties may bias biclusters results dramatically.

2.2 A Brief Introduction to the Parametric Minimum $L_2$ Error Criterion

We start with a brief introduction to $L_2$E, since this is an important motivation for the construction of LEBRA error criterion function. The idea of minimizing distance error criterion is definitely not a new idea. It has been around for a long time and its robustness has been shown by Beran (1977); Donoho and Liu (1988). Then, Hellinger distance, a special case of minimum
distance error criterion, is used to estimate the bin width of the histogram in the nonparametric least squares cross-validation problem Rudemo (1982); Bowman (1984). Motivated by its robustness and and its usage in estimating bin width, Scott (2001) proposed a robust parametric unbiased estimation by minimizing Integrated Squared Error (ISE) which is also a special case of the minimum distance criterion. The derivation of $L_2E$ in the parametric modeling is given below:

Consider minimizing the following ISE:

$$
\int \left[ f(x|\theta) - f(x|\theta_0) \right]^2 dx
$$

After some algebra, we have:

$$
\hat{\theta}_{L_2E} = \arg\min_{\theta} \int \left[ f(x|\theta) - f(x|\theta_0) \right]^2 dx = \arg\min_{\theta} \int \left[ f^2(x|\theta) - 2f(x|\theta)f(x|\theta_0) + f^2(x|\theta_0) \right] dx
$$

$$
= \arg\min_{\theta} \int f^2(x|\theta) dx - 2 \int f(x|\theta)f(x|\theta_0) dx
$$

$$
= \arg\min_{\theta} \int f^2(x|\theta) dx - 2E_{\theta_0}f(x|\theta)
$$

$$
= \arg\min_{\theta} \int f^2(x|\theta) dx - \frac{2}{n} \sum_{i=1}^{n} f(x_i|\theta)
$$

where $f$ is assumed statistical model for the data, $\theta_0$ is the true value for the unknown parameter $\theta$ and $\theta_{L_2E}$ is the $L_2E$ obtained from minimizing ISE. Scott (2001) presented a normal density example: Suppose $x \sim N(\mu, 1)$ and we are interested in estimating the unknown parameter $\mu$ given a set of observations. According to the formula derived above, the $L_2E$ is given below.
along with the MLE for comparison purpose.

\[ \hat{\mu}_{\text{MLE}} = \text{argmax}_\mu \sum_{i=1}^{n} \log \phi(x_i|\mu,1) \]  
(2.4)

\[ \hat{\mu}_{L_2E} = \text{argmin}_\mu \frac{1}{2\sqrt{\pi}} - \frac{2}{n} \sum_{i=1}^{n} \phi(x_i|\mu,1) \]  
(2.5)

Note the difference between MLE and \( L_2E \), MLE is to maximize the product of the densities while \( L_2E \) maximizes the sum of the densities. Scott (2001) also showed the robustness of \( L_2E \) by showing boundedness of the influence function of \( L_2E \) while the influence function of MLE is unbounded.

### 2.3 A Brief Introduction to MM Algorithm

MM algorithm Hunter and Lange (2004) generalized the idea of EM algorithm. The basic idea of the MM algorithm is to convert a hard problem (probably not convex) to a sequence of simpler sub-problems and we will see how this works later. The term “MM” can be interpreted as Minorization-Minimization or Majorization-Maximization depending on whether we are looking for a minimum or a maximum. A hallmark of the MM algorithm is its numerical stability because every iteration will be guaranteed to make progress toward the ground truth. Suppose we are going to find the minimum of a convex function \( f(\theta) \). A function \( g \) is called a majorant of \( f \) at \( \hat{\theta} \) if we have the following conditions:

\[ f(\theta) \leq g(\theta|\hat{\theta}) \]  
(2.6)

\[ f(\hat{\theta}) = g(\hat{\theta}|\hat{\theta}) \]  
(2.7)

In other word, the surface of the majorant lies above the surface of a objective function that we are interested in working with. Two surfaces intersects if and only if at \( \theta = \hat{\theta} \). Suppose \( \hat{\theta} \) is where we are for now. With MM algorithm, we are guaranteed to make progress via simply minimizing the majorant \( g \) instead of \( f \). Thus the MM algorithm contains two steps: (1) construct the
majorant at each iteration; (2) minimizing the majorant conditioning on the current estimate. A complete MM algorithm procedure in $\mathbb{R}^2$ is illustrated in the following figure.

![Figure 2.1](image)

Figure 2.1: Complete MM procedure looking for the minimum value of $x_\infty = \arg \min_x f(x)$, The thick solid curve is $f(x)$. The dash curve are majorants based on each update $x_i$

## 2.4 L$_2$ Error Biclustering Criterion and Its Properties

Motivated by the robustness of L$_2$E shown in the parametric modeling field, we are hoping to introduce its robustness into Biclustering problem. Assume a ground truth $U$ has a checkerboard structure. This is because, in the biclustering setting, our goal is to cluster the rows and the columns of a data matrix simultaneously. If we reorder the rows and the columns of data matrix according to the bicluster assignment, there will be a checkerboard pattern, i.e. a block matrix with each block a constant submatrix. Denote the data matrix as $X \in \mathbb{R}^{n \times p}$ consisting of $n$ samples generated from a $p$-dimensional space. Suppose the latent checkerboard structure is generated by the $R \times C$ where $R$ is the indices of the groups of rows, $C$ the indices of the groups of columns and $\times$ is the Cartesian product. Then each entry of $X$, $x_{ij}$, can be decomposed as
\[ x_{ij} = \mu_0 + \mu_{rc} + \epsilon_{ij} \] where \( \mu_{rc} \) is the centroid of \( r \)-th row group, \( R_r \), and \( c \)-th column group, \( C_c \), \( \mu_0 \) is the overall baseline mean shared by every entry and \( \epsilon_{ij} \sim N(0, \sigma^2) \). Then each entry of the data matrix follows a normal distribution with mean, \( \mu_0 + \mu_{rc} \) and variance, \( \sigma^2 \). Suppose we have \( K_r \) row groups and \( K_c \) column groups in total, we have \( p = \sum_{i=1}^{K_r} |C_i| \) and \( n = \sum_{i=1}^{K_c} |R_i| \) Denote the latent checkerboard structure matrix, \( U \), then we have \( U = \{ U_{rc} \}_{(r,c) \in R \times C} \) where \( U_{rc} = \mu_{rc} 1_{|R_r|} 1_{|C_c|}^T \in \mathbb{R}^{|R_r| \times |C_c|} \).

Motivated by (2.5), we propose to identify the latent checkerboard structure by minimizing the following penalized ISE. Denote \( F_\lambda: \mathbb{R}^{n \times p} \times \mathbb{R}^{++} \to \mathbb{R} \), then \( F_\lambda \) is the combination of distance measure \( \psi: \mathbb{R}^{n \times p} \times \mathbb{R}^{++} \to \mathbb{R} \) and regularization term \( J: \mathbb{R}^{n \times p} \to \mathbb{R} \):

\[
F_\lambda(U, \tau) = \begin{align*}
\psi(X - U, \tau) + \lambda J(U) \\
= \frac{\tau}{2\sqrt{\pi}} - \frac{\tau}{np} \sqrt{\frac{\pi}{2}} \sum_{i,j} e^{-\frac{1}{\tau}(x_{ij} - u_{ij})^2_2} + \lambda J(U)
\end{align*}
\]

(2.8) (2.9)

where \( J(U) = \sum_{i<j} \omega_{ij} ||U(:,i) - U(:,j)||_2 + \sum_{i<j} \tilde{\omega}_{ij} ||U(i,:) - U(j,:)||_2 \), \( \omega_{ij} = \frac{k}{ij} \) \( \tilde{\omega}_{ij} = \phi(||X(:,i) - X(:,j)||_2^2) \). The weights is the product of two terms: an indicator and a Gaussian kernel measuring the similarity. The idea of weights here follows the same idea of weights in COBRA setting. The first term \( \frac{k}{ij} \) is 1 if \( j \) is among \( k \)'s \( k \)-nearest-neighbors or vice versa and zeros otherwise, which will impose sparsity of the weights Chi et al. (2017).

Now we have transformed a biclustering problem into a penalized optimization problem. A penalized optimization problem is similar to an ordinary optimization problem. The only difference is the measure quantifying the similarity between \( X \) and \( U \). An ordinary penalized optimization problem uses norm while LEBRA error criterion uses ISE. Without penalty term, it is not hard to notice that the optimal estimation is \( \hat{X} \) itself, i.e., \( \hat{U} = X \). The penalty term biased the estimation toward the matrices that have latent checkerboard structure as much as possible. The penalty term is a mapping \( J: \mathbb{R}^{n \times p} \to \mathbb{R} \) that penalizes the pairwise differences among rows and the pairwise differences among the columns of the estimates simultaneously.
The penalty parameter $\lambda$ tunes the tradeoff between the ISE and the penalty term. The weights $\omega_{ij} = \omega_{ji}$ and $\tilde{\omega}_{ij} = \tilde{\omega}_{ji}$ are non-negative and will be explained in details later.

The regularization term $\mathcal{J}(U)$ or similar ideas have been widely used in many papers that aim to impose sparsity. Suppose $\mathcal{J}(U)$ only regularizes rows(columns), if all of weights are the same, the LEBRA error criterion becomes a summation of measure of distance with a fused penalty on columns, which is very similar to the idea of convex clustering described in Pelckmans, De Brabanter, Suykens, and De Moor (2005). If we set $\psi = || \cdot ||$ and a non-uniform weights are, say, Gaussian kernel, the LEBRA error criterion is similar to the relaxed version of hierarchical clustering in Hocking, Joulin, Bach, and Vert (2011) and the extensions described in Lindsten, Ohlsson, and Ljung (2011). Moreover, If we add $\mathcal{J}(U)$ regularizes both rows and columns, the LEBRA error criterion will be the same as the error criterion in COBRA Chi et al. (2017).

The regularization parameter controls how strong the strength is to impose checkerboard structure on $U$. If $\lambda = 0$, there is no strength of imposing any checkerboard pattern, then the best estimate of $U$ would be trivially $X$, the data matrix itself. If $\lambda = +\infty$, it indicates that the strength of imposing the checkerboard structure on $U$ is infinitely strong. In other word, it is not allowed the existence of any different clusters in both rows and columns, then the result would be the simplest checkerboard structure if all the weights of the pairwise differences among both rows and columns are not zeros, a constant matrix. Then, $U$ will be a constant matrix if $\forall (i, j), \epsilon_{ij}^k = 1$. The approach of choosing a balanced $\lambda$ will be discussed shortly.

Sparse Gaussian kernel has been used to construct both the weights term $\omega_{ij}$ and distance metric. As for weights, it is mainly because of computation reasons. In fact, any kernel $\kappa(x_i, x_j)$ will do. The idea of using kernel is to make the cluster path sensitive to the local density in the data Hocking et al. (2011); Lindsten et al. (2011); Chi et al. (2017). As for the distance measure in the ISE, the usage of Gaussian kernel comes from an implicit assumption that the data matrix follows a Gaussian process. We will see this implicit assumption in the following discussion.

We will provide an interpretation about the formulation of LEBRA error criterion. It is very
useful for developing an intuition about the robustness of the LEBRA. Recall the LEBRA error criterion, given fixed $\lambda$,

$$F_\lambda(U, \tau) = \frac{\tau}{2\sqrt{\pi}} - \frac{\tau}{np\sqrt{2\pi}} \sum_{i,j} e^{-\frac{1}{2}\tau^2(x_{ij} - u_{ij})^2} + \lambda \underbrace{J(U)}_{\text{penalty}}$$

(2.10)

Since $J(U)$ imposes a checkerboard structure on the estimate, $\hat{U}$. Without losing generality, suppose we have already known that there are total number of $K$ biclusters. Then we can restrict our attention only to the matrices that have $K$ squares structure denoted as $\Gamma_K \subseteq \mathbb{R}^{n \times p}$. In other word, we no longer need the term, $J(U)$ and we have the following equivalent problem.

$$\arg\min_{r \in \mathbb{R}, U \in \mathbb{R}^{n \times p}} F_\lambda(U, \tau) = \arg\min_{r \in \mathbb{R}, U \in \Gamma_K} \psi(X - U, \tau) \quad (2.11)$$

Notice that the equivalent problem on the right side has no penalty term. For convenient purpose, we let the indices of all of the biclusters in $\hat{U}$ follow column major. Therefore, given total number of $K = K_rK_c$ biclusters, the location of $x_{rc}$ can be expressed in terms $K, K_r, K_c$. The corresponding row index is $(K \mod K_r)$ and the corresponding column index is $(K \mod K_c)$. Denote $\epsilon_K = \{(i, j)|x_{ij} \in (R \times C)_K\}$ be the bicluster index set where $(R \times C)_K$ is the bicluster
in the \((K \mod K_c)\)th column group and the \((K \mod K_r)\)th row group.

\[
\psi(X - U, \tau) = \frac{\tau}{2\sqrt{\pi}} - \frac{\tau}{np} \sqrt{\frac{2}{\pi}} \sum_{k=1}^{K} \sum_{(i,j) \in \epsilon_k} e^{-\frac{1}{2} \tau^2 (x_{ij} - u_k)^2} \tag{2.12}
\]

\[
= \sum_{k=1}^{K} \left( \frac{\tau}{2K\sqrt{\pi}} - \frac{\tau}{np} \sqrt{\frac{2}{\pi}} \sum_{(i,j) \in \epsilon_k} e^{-\frac{1}{2} \tau^2 (x_{ij} - u_k)^2} \right) \tag{2.13}
\]

\[
= \sum_{k=1}^{K} \left( \frac{\tau}{2K\sqrt{\pi}} - \frac{\tau}{Kq} \sqrt{\frac{2}{\pi}} \sum_{(i,j) \in \epsilon_k} e^{-\frac{1}{2} \tau^2 (x_{ij} - u_k)^2} \right) \tag{2.14}
\]

\[
= \frac{1}{K} \sum_{k=1}^{K} \left( \frac{\tau}{2\sqrt{\pi}} - \frac{\tau}{q} \sqrt{\frac{2}{\pi}} \sum_{(i,j) \in \epsilon_k} e^{-\frac{1}{2} \tau^2 (x_{ij} - u_k)^2} \right) \tag{2.15}
\]

(2.16)

Set \(\tau = \frac{1}{\sigma}\) and \(u_k = \mu_k\) (To be consistent with the notations in Scott (2001)), then (2.16) becomes

\[
\frac{1}{K} \sum_{k=1}^{K} \left( \frac{1}{2\sigma\sqrt{\pi}} - \frac{1}{\sigma q} \sqrt{\frac{2}{\pi}} \sum_{(i,j) \in \epsilon_k} e^{-\frac{1}{2} \tau^2 (x_{ij} - \mu_k)^2} \right) \tag{2.17}
\]

The term in the bracket are the same as the error criterion in parametric modeling problem under Gaussian model described in Scott (2001) for every single \(k\). The equation (2.17) indicates that LEBRA will reduce to a parametric modeling problem within each of \(K\) biclusters using Gaussian distribution model under certain conditions. In other word, LEBRA can be seen as a combination of multiple parametric modeling problems within each bicluster given the number of biclusters under Gaussian modeling assumption. This is exactly where the implicit Gaussian error assumption comes from. Only if we assume Gaussian model can we get the LEBRA error criterion. \(\tau\) can be interpreted as the inverse of the standard error if every data entry within each bicluster has the same standard error. Additionally, think of above derivation in a reverse way, it actually shows how we come up with the form of the error criterion in LEBRA. That is also one way of explaining the reason why LEBRA inherits the robustness of the minimum
distance estimator.

One interesting relationship need to be noticed is that there is an inverse relationship between $\lambda$ and $\tau$. Also the range of $\lambda$ is actually lower bounded. To see this, let us assume $\lambda = 0$, then $\hat{U}$ would be $X$ itself. Then every single entry of $X$, $x_{ij}$, belongs to its own bicluster. The variance with each bicluster with one entry is zero. Since $\tau$ can be interpreted as an inverse of the standard deviation. Then $\tau = +\infty$ in this case. Therefore, in order for LEBRA return a meaningful solution, there exists a lower bound $B$ such that $\lambda \in (B, \infty)$

2.5 Connection to the M-estimator

From the previous section, we know the formulation of LEBRA is closely related to the Minimum Distance Estimator. Actually, there is another way to gain the intuition of the formulation of the LEBRA objective function. Yohai and Zamar (1988) defined a new class of rho function satisfying the following six conditions:

1. $\rho(0) = 0$
2. $\rho(-u) = \rho(u)$
3. $0 \leq u \leq v$ implies that $\rho(u) \leq \rho(v)$
4. $\rho(t)$ is continuous
5. Let $a = \sup \rho(u)$; then $0 < a < \infty$
6. If $\rho(u) < a$ and $0 \leq u < v$, then $\rho(u) < \rho(v)$

Apparently, the $\rho$ function in the ISE part of LEBRA objective function satisfies all the conditions except the first one, I would call its a generalized version of the $\rho$ function. Recall the the ISE part:

$$\frac{\tau}{2\sqrt{\pi}} - \frac{\tau}{np} \sqrt{\frac{2}{\pi}} \sum_{i,j} e^{-\frac{1}{4}\tau^2(x_{ij} - u_{ij})^2}$$  \hspace{1cm} (2.18)
After some algebra, we have:

$$\frac{\tau}{np} \sum_{i,j} \rho(\tau r_{ij}) \quad (2.19)$$

where $\rho(t) = \frac{1}{2} - \sqrt{\frac{2}{\pi}} e^{-\frac{1}{2}t^2}$. In our case, $\rho(0) = \frac{1}{2} - \sqrt{\frac{2}{\pi}}$. Our problem as a scaled version of MM estimate (Yohai, 1987) from the M estimate. It has been applied in many other fields as a robust alternative. For example, the robust ridge regression (RRR) replaced the Euclidean distance with a specific form of $\rho(t)$ function along with some constants need to be computed correspondingly. Besides Ridge Regression, Robust LASSO applied the same idea to the original LASSO problem but using Tukey’s Biweight Criterion. Whatever the issues are, robustness comes from the boundness of the $\rho(t)$. For detailed explanation, please check out the High Breakdown-Point Estimates of Regression by Yohai and Zamar (1988).

### 2.6 Estimation of Biclusters with LEBRA

Block relaxation method is a class of fixed point methods. It has a very long and complicated history. For more detailed discussion about block relaxation method, please see De Leeuw (1994). In this article, we only focus on the application of block relaxation method on our problem. Recall the formulation of LEBRA problem, it contains two variables: $U$ and $\tau$, we are going to use block relaxation method to minimize $U$ and $\tau$ alternatively.

From now on, $X$ is a $n \times p$ fixed data matrix with $n$ rows and $p$ columns. The weights $W$ and $\tilde{W}$ are only based on the rows and columns of $X$, therefore they are fixed as well. The LEBRA can be seen as a bivariate function on $\mathbb{R}^{n \times p} \times \mathbb{R}^{++}$. Block relaxation method will require solving subproblems on $\mathbb{R}^{n \times p}$ and $\mathbb{R}^{++}$, respectively. We are going to illustrate the approach of minimizing each of them.

**Updating $\tau$:** Recall the LEBRA error criterion is literally an univariate function of $\tau$, given $\tilde{U}$. Thus, many approaches can be used here. We simply use Golden Search Algorithm (GSA) to find the minimum value with respect to $\tau$ by assuming $\tau$ is somewhere between zero and some
reasonable large number value. Estimate $\tau$ is to solving the following minimization problem:

\[
\hat{\tau} = \arg\min_\tau \frac{\tau}{2\sqrt{\pi}} - \frac{\tau}{np\sqrt{\pi}} \sum_{i,j} e^{-\frac{1}{2} \tau^2 (x_{ij} - u_{ij})^2} + \lambda J(U) \tag{2.20}
\]

\[
\hat{\tau} = \arg\min_\tau \frac{\tau}{2\sqrt{\pi}} - \frac{\tau}{np\sqrt{\pi}} \sum_{i,j} e^{-\frac{1}{2} \tau^2 (x_{ij} - u_{ij})^2} \tag{2.21}
\]

**Updating $U$:** The update procedure of LEBRA for $U$ is actually a COBRA problem on a "modified data." COBRA is developed by Chi et al. (2017), which transfers an ordinary biclustering problem into a penalized convex optimization problem. The COBRA error criterion is defined as following:

\[
\text{COBRA}_\lambda(X) = \arg\min_U \frac{1}{2} \|X - U\|_F^2 + \lambda J(U) \tag{2.22}
\]

where the notations are defined as the same as we did for the LEBRA. COBRA plays an important role in the estimation of LEBRA, and we will see in details in the later discussion.

Fixed $\tau$, LEBRA error criterion can be viewed as an univariate function of $U$. Then we have the following proposition:

**Proposition 2.6.1** Given fixed $\tau$, the ISE part of the LEBRA error criterion has a bounded hessian matrix.

The proof of Proposition 2.6.1 will be given in Appendix A.1. Given Proposition 2.6.1, Proposition 2.6.2 tells us how to find a majorant of $F_\lambda$ and the way to solve it. The proofs of them will be given in Section A.1.

**Proposition 2.6.2** Given fixed $\tau$, a majorant of $F_\lambda$, $G_\lambda : \mathbb{R}^{n \times p} \to \mathbb{R}$, has the form:

\[
G_\lambda(U|\tilde{U}) = \frac{\sqrt{2} \tau L}{np\sqrt{\pi}} \left( \frac{1}{2} \|U - X^*\|_F^2 + \gamma(\lambda)J(U) \right) + \tilde{C} \tag{2.23}
\]
where $\tilde{C}$ is a constant not related to either $U$ or $\tau$,

$$\gamma(\lambda) = \frac{np\sqrt{\pi}}{\tau\sqrt{2L}} \lambda$$  

(2.24)

$$X^* = \tilde{U} + \tilde{K}$$  

(2.25)

$$\tilde{K} = \left\{ \frac{\tau^2}{L}(x_{ij} - \tilde{u}_{ij})e^{-\frac{\tau^2(x_{ij} - \tilde{u}_{ij})^2}{2}} \right\}_{(i,j)} \in \mathbb{R}^{n \times p}$$  

(2.26)

It turns out that the majorant coincides with the form of the COBRA error criterion. Therefore, minimizing majorant with respect to $U$ in each LEBRA iteration is just a COBRA problem biclustering over the “modified” data $X^* = \tilde{U} + \tilde{K}$ with a tuning parameter $\gamma(\lambda)$.

As for $L$, we can find the estimation of $L$ by the following algebra:

$$\nabla^2 \left( -e^{-\frac{\tau^2(x_{ij} - \bullet)^2}{2}}(u_{ij}) \right) = \tau^2 e^{-\frac{\tau^2(x_{ij} - u_{ij})^2}{2}} \left( 1 - \tau^2(u_{ij} - x_{ij})^2 \right) \leq \tau^2 e^{-\frac{\tau^2(x_{ij} - u_{ij})^2}{2}} \leq \tau^2$$

$$= L$$

The robustness of LEBRA lies in the formulation of $G_\lambda$ and we will explain this in details later.

**L₂E BiclusteRing Algorithm (LEBRA):** After we have an algorithm to minimize two subproblems, we are now ready to have LEBRA:

**Algorithm 1 L₂E BiclusteRing Algorithm (LEBRA)**

1: **procedure** LEBRA($U^{(0)}, \tau^{(0)}$) \> Initialize $U^{(0)}$ and $\tau^{(0)}$
2: \> **repeat**
3: \> $\tau^{(i+1)} = \text{GSA}(\tau^{(i)}, U^{(i)})$
4: \> $U^{(i+1)} = \text{COBRA}_{\gamma(\lambda)}(U^{(i)})$  \> $\triangleright \gamma(\lambda) = \frac{np\sqrt{\pi}}{\tau^{(i+1)}\sqrt{2L}}\lambda$
5: \> **until** convergence
6: **end procedure**

We can see LEBRA uses COBRA iteratively but dynamically changing the value of the tuning parameter in the error criterion of COBRA based on the solution path of $U^{(i)}$. Moreover,
the more we are close to the ground truth, LEBRA behaves more closer to an algorithm with an Euclidean loss function. For convenient purpose, let us assume $\sigma = 1$. Recall the Proposition 2.6.1, under certain conditions of Proposition 2.6.1, the error criterion in LEBRA can be written into (2.27). Now let us apply quadratic Taylor expansion to the term $-e^{-(x_{ij} - u_k^2)}$ in (2.27) as a function of $x_{ij}$ at $x_{ij} = u_k$ if $(i,j) \in \epsilon_k$. Then (2.27) can be rewritten further into the following form approximately

$$\frac{1}{K} \sum_{k=1}^{K} \left( \frac{1}{2\sqrt{\pi}} - \frac{1}{q} \sqrt{\frac{2}{\pi}} \left| x_{\epsilon_k} - u_k \mathbf{1} \right|^2 \right)$$

(2.27)

where $u_{\epsilon_k} \in \mathbb{R}^{\mid \epsilon_k \mid \times 1}$. (2.27) indicates LEBRA is equivalent to a scaled Euclidean distance based biclustering algorithm when we are closer enough to the ground truth. The difference is a high order term of the distance between the data matrix and the ground truth. In other word, LEBRA gains strong resistance to outliers while sacrificing many good properties such as convexity, but its local behavior is similar to a scaled COBRA, a convex continuous problem.

### 2.7 Robustness of LEBRA

The robustness of LEBRA is very intuitive and it lies in the formulation of $G_\lambda$. We have already mentioned its robustness actually comes from the usage of minimum distance estimator. From algorithm aspect, we can also have a good understanding where the robustness comes from and how LEBRA actually works. After some simple algebra, we can simply write the quadratic majorant in the following form:

$$G_\lambda(U|\tilde{U}) = \frac{\tau L}{np\sqrt{2\pi}} \| U - (\tilde{Q} \odot X + \tilde{U} \odot (\mathbf{1}^T - \tilde{Q})) \|_F^2 + \lambda J(U)$$

(2.28)

where $\odot$ denotes element-wise matrix multiplication, $\tilde{Q} = Q(\tilde{U}) = \{exp(-\frac{x_{ij}^2}{2}(x_{ij} - \tilde{u}_{ij}))\}_{i,j} \in \mathbb{R}^{n \times p}$ and we will use $\tilde{Q}$ notation throughout the paper. Throughout the algorithm, $\tilde{Q}$ is a
function of the previous update of $U$ and $\tilde{Q}$ tends to be zero when there are relatively large disagreement between the data matrix and previous estimation, while $\tilde{Q}$ is much closer to one when there is a perfect agreement between the data matrix and the previous estimation.

The essence of the LEBRA is actually a COBRA but more dynamic in terms of the data matrix that it works on throughout the algorithm. The data matrix LEBRA is working over keeps adjusting as the LEBRA proceeds. LEBRA uses the convex combination of the data matrix $X$ and the previous update $U$ instead of $X$ all the way like COBRA does. As LEBRA iterates, LEBRA not only dynamically updates the tradeoff between the real data matrix and tuning parameter via $\tilde{Q}$ that LEBRA is working on, but it also adjusts the values of $\tilde{Q}$ itself throughout the whole algorithm.

An outlier at certain location within a certain bicluster tends to have relatively large disagreement with its centroid. In this case, such disagreement tends to produce a small value of $\tilde{Q}_{ij}$. Given small $\tilde{Q}_{ij}$, this indicates LEBRA algorithm tends to “trust” estimation from algorithm more than the data matrix itself at certain $i$th row and $j$th column. In other word, LEBRA is just trying to cluster the data using “important” part instead of using all data. And the weights control how important each entry is. This idea is very similar to the idea of median or trimmed mean as a substitutes for average value to estimate the mean. The only difference here is that the definition of importance is defined based on the Gaussian kernel and the data driven while the trimmed mean and the median are very subjective.

### 2.8 Convergence of LEBRA

With the compactness assumption of the LEBRA sequence $\{\theta(k)\}_{k>0} = \{\tau(k), U(k)\}_{k>0}$, we are able to prove that there exists a convergent subsequence of the LEBRA sequence guarantee to converge a critical point. The LEBRA convergence theorem is given below:

**Theorem 1** *LEBRA Convergence Theorem*: Given an algorithm on $\Xi$, $\theta^{(0)} = (\tau^{(0)}, U^{(0)}) \in$
Ξ and LEBRA mapping φ, assuming a bounded sequence \( \{\theta^{(k)}\}_{k=1}^\infty \) is generated which satisfies

\[
\theta^{(k+1)} = \phi(\theta^{(k)})
\]  
(2.29)

Then either the LEBRA stops at the point where a solution is identified or there exists such a k so that for all \( k + j (j \geq 1) \) there is a convergent subsequence of \( \{\theta^{(i)}\}_{k=0}^\infty \) whose limit is the fixed point of LEBRA mapping, φ.

The proof of Theorem 1 are given in Appendix A.1.

2.9 Model Selection

Not all biased models are better than the unbiased model, the goal of using penalized model is to sacrifice the unbiased property in exchange a good reduction in estimation variance. This is why we need an appropriate approach that is capable of selecting the best model. Model selection is always a complicated problem for any problems. So is in both clustering and biclustering problem. For the simplicity, we will use random hold-out cross-validation method.

Random Hold-out Cross-validation: We will randomly select a set of entries from the data matrix and put their index pairs as a hold out set. The corresponding entries in a hold-out set will be taken out and impute with zeros instead. LEBRA will be implemented for a sequence of tuning parameters and choose the specific tuning parameter that can minimize the prediction error for entries in the hold-out set. This idea was first proposed by Wold (1978) for model selection in principal component and analysis and has been used more recently to select tuning parameters in matrix completion problem Mazumder, Hastie, and Tibshirani (2010) and convex biclustering problem as well Chi et al. (2017). The difference between LEBRA and COBRA in choosing tuning parameter is that the distance function we used. Euclidean distance is used for COBRA while the ISE is used for LEBRA. Denote a hold-out index pair \( \Theta \subset \Omega = \{1, 2, \ldots, n\} \times \{1, 2, \ldots, p\} \) where \( n \) is the sample size and \( p \) is the number of parameters. Denote a projection operator as \( P_\Theta(\bullet) \) such that \( P_\Theta(X) = \{x_{ij}I_\Theta\}_{i,j} \) where \( I_\Theta \) is an indicator function.
Following Chi et al. (2017)’s convention, we choose a small hold out set relative to the original data, say 10% of the data. Then we minimize the following criterion over \( U \) and \( \tau \):

\[
\frac{\tau}{2\sqrt{\pi}} - \frac{\tau}{np}\sqrt{\frac{2}{\pi}} \sum_{i,j} e^{-\frac{1}{2}\tau^2(P_{\Theta c}(x_{ij}) - P_{\Theta c}(u_{ij}))^2} + \lambda J(U) \tag{2.30}
\]

for a sequence of \( \lambda \)'s, \( \Lambda \), ordered from smallest to the largest. We choose the \( \gamma \) minimizing the prediction error over the hold out set \( \Theta \), i.e.

\[
\lambda^* = \arg\min_{\lambda \in \Lambda} \frac{\tau_0}{2\sqrt{\pi}} - \frac{\tau_0}{np}\sqrt{\frac{2}{\pi}} \sum_{i,j} e^{-\frac{1}{2}\tau_0^2(P_{\Theta c}(x_{ij}) - P_{\Theta c}(u_{ij}))^2} \tag{2.31}
\]

where \( \tau_0 \) is the estimation from solving the hold out minimization.

**Solving the Random Hold-out Cross-validation Problem:** The problem of minimizing (2.30) can be viewed as a LEBRA version matrix completion problem. LEBRA version is because of the difference in \( \phi \). Recall the traditional matrix completion problem uses the Euclidean function as \( \phi \) while LEBRA uses Integrated Squared Error. The following pseudo-code summarizes a simple algorithm for reducing the LEBRA version matrix completion problem to solving a sequence of LEBRA problems.

**Algorithm 2** LEBRA with missing data

1: \( \textbf{procedure} \) LEBRA HOLD OUT(\( U^{(0)} \), \( \tau^{(0)} \)) \( \triangleright \) Initialize \( U^{(0)} \) and \( \tau^{(0)} \)
2: \( \textbf{repeat} \)
3: \( \text{\textbf{end procedure}} \)

```
\begin{align*}
M &= P_{\Theta c}(X) + P_{\Theta}(U^{(i)}) \\
(\tau^{(i+1)}, U^{(i+1)}) &= LEBRA(\tau^{(i)}, M)
\end{align*}
```

The estimation from the previous step will be used to fill in the missing entries of the current LEBRA problem. By doing so, the hold-out problem has boiled down to just a regular LEBRA
problem for each iteration. This approach is the same as the model selection model described in Chi et al. (2017), similar to the soft-impute matrix completion problem in Mazumder et al. (2010). The difference is that they use Euclidean distance while we are using Integrated Squared Error. For each iteration, the subproblem is actually a MM problem Lange, Hunter, and Yang (2000). Please see WEB APPENDIX D in Chi et al. (2017) for the proof of the derivation of the algorithm of solving the LEBRA with missing value.

2.10 Simulation Studies

In this section, we compare LEBRA with COBRA and Sparse Biclustering (SpBC). Nine measures have been used to assess the quality of biclustering estimation. But because of the limitation of the paper length, we have only reported four of them. The results in terms of the other five measures along with the four measures shown in this section can be found in Appendix A.2. Those four are: Rand Index (RI), Adjusted Rand Index (ARI), the Variation of Information (VI) and the Normalized Mutual Information (NMI). COBRA is implemented in the R package: cvxbiclustr; SpBC is implemented in the R package: sparseBC. COBRA solves the biclustering problem by minimizing a convex error criterion with the Frobenius norm as the measure of the distance of the matrix in the loss function combined with a penalty term penalizing the pairwise differences among rows and columns, respectively, while the SpBC method minimizes the same loss function as COBRA does but imposes the $l_1$ penalty on the number of centroids instead. All parameters are selected according to the documentation written in the corresponding R package.

Rand Index (RI) or Rand measure, named after William M. Rand, is a measure of the similarity between two biclustering estimations. It is the ratio of the occurrence of total agreements of two biclustering estimations of the total possible pairs. The range of the Rand Index is from 0 to 1. 1 indicates one biclustering estimation is completely consistent with the other biclustering estimation while 0 means one biclustering estimation is completely against the other one. The Adjusted Rand Index (ARI) is a corrected-for-chance version of the RI. ARI can be any number
less than \( \log(K) \), where \( K \) is the maximum number of biclusters. The difference between ARI and RI is ARI not only counts the agreements between two biclustering estimations, but also makes an adjustment for the chance purpose. Both RI and ARI are pair-counting-based measures. The Variation of Information (VI) or Shared Information Distance is based on the concepts of entropy and information. It is the difference between the information shared by two biclustering estimations and the total amount of information carried by each biclustering estimation. If two biclustering estimations are completely the same, we can expect the shared information would be the same as the information carried by each of them, i.e. \( VI = 0 \). Unlike RI and ARI, VI has been proved to be a metric that defines the distance between two biclustering estimations. The triangle property and symmetry of VI are extremely useful when interpreting the VI scores. In addition, the advantage of having a bounded metric with a known bound is that we can have the definition of “small” in terms of VI. Meilä (2007) found the nearest neighbor of a clustering estimation in terms of the VI metric is clustering result obtained by splitting or merging small clusters, which is consistent with our intuitive definition of “small”. Therefore by using VI, we know what is called a “small” change of a biclustering result. The Normalized Mutual Information (NMI) shares the same idea as VI. The difference between them is that VI defines the distance between shared information and the totally information carried by each estimation while NMI does the same but in terms of ratio of shared information over the total information carried by each estimation. Like VI, NMI is an information theory based non-negative measure as well. If two biclustering estimations agree with each other, the shared information would be the same as the information carried by themselves, i.e., \( NMI = 1 \), while \( NMI = 0 \) if they are totally against each other, i.e. no shared information. Thus NMI is between 0 and 1 as well like RI. Since the NMI is normalized, we can use it to compare different number of clusters as well and eliminate the problem that entropy tends to increase the number of biclusters. The detailed introduction about these measures and the additional five measures can be found in the Appendix A.3.

Our simulated data matrix is based on a \( 200 \times 200 \) matrix with 400 outliers, which is 1% of
the total same size. Suppose the ground truth has four row clusters and four column clusters, sixteen biclusters in total. For each bicluster, we simulated the data by $N(\mu_{rc}, \sigma^2)$, where $\mu_{rc}$ is the centroid for the bicluster located in $r$th row cluster and $c$th column cluster. Moreover, $\sigma^2$ is the variation of each bicluster. The mean parameter $\mu_{rc}$ can be any numbers with relative same distance for each $r$ and $c$. In our simulation study, we pick $\mu_{rc}$ to be a bunch of different integers. The noise level is set to be a number in the set $\sigma^2 \in \{1, 50, 100, 500\}$. The locations of outliers are randomly chosen. The values of the outliers is generated from a uniform distribution $U(4\sigma_0^2, 5\sigma_0^2)$, where $\sigma_0^2$ is the variance of the ground truth plus the variance within each bicluster.

Figure 2.2 compared their performances when there are no contamination, i.e., that is $C = 0$. The sizes of row clusters are 100, 30, 30 and 40, while the sizes of the column clusters are 50, 60, 50 and 40. Without outliers, we can see Sparse Biclustering, LEBRA and COBRA have the

![Figure 2.2: Row Size: 100, 30, 30, 40; Column size: 50, 60, 50, 40; No outliers](image)
same performance levels when the noise level is less than 50. As we increased the noise level, the Sparse Biclustering approach did a slightly better job than LEBRA and COBRA did. Moreover, LEBRA is more effective than COBRA when the noise level is at 100 but less effective than SpBC under the same condition. But overall, based on this experiment, Sparse Biclustering is preferred when there are no outliers. A possible reason that Sparse Biclustering beats LEBRA and COBRA might be due to the maximum number of cluster selection parameters we have set in this simulation. The maximum number of row clusters and column clusters are set to be 10 for timing purposes. Theoretically it should be set up to the number of rows and the number of columns, respectively, which might consequently favor the results of the Sparse Biclustering approach. Figure 2.3 shows the results when the sizes of biclusters are more uneven than with the previous case. The sizes of the row clusters are 10, 100, 50 and 40, and the sizes of the

![Figure 2.3](image)

Figure 2.3: Row Size: 10, 100, 50, 40; Column size: 10, 100, 50, 40; No outliers
column clusters are 10, 100, 50 and 40 as well.

When there are outliers; that is $C \neq 0$, the story is slightly different. Suppose the true underlying checkerboard structure is still a 4-by-4 block structure. The row structure and column structure are the same as the conditions in using in Figure 2.2.

Figure 2.4: Row Size: 100, 30, 30, 40; Column size: 60, 60, 60, 50. The number of outliers: 4000

Figure 2.4 shows the performances of three approaches in terms of RI, ARI, VI and NMI when 1% of the sample size are contaminated. It is observed from ARI and ARI that LEBRA shows a great resistance against outliers while Sparse Biclustering is totally broken with the appearance of the outliers. COBRA has stronger ability to handle outliers than SpBC does but not as well as LEBRA does. For the other two measures, we need to be very careful in interpreting the results, the details discussion will be provided later. Now we changed our bicluster size more uneven than the previous case, both column and row structure are 10, 100, 50 and 40. The result
can be found in Figure 2.5.

![Graph showing ARI, NMI, RI, and VI measures against noise level for COBRA, LEBRA, and SpBC.]

**Figure 2.5**: Row Size: 10, 100, 50, 40; Column size: 10, 100, 50, 40. The number of outliers: 4000

We can see the overall trend of even size case and not even size case are almost the same except the exact values are a bit different. COBRA in uneven tends to have worse values than it does in even case. The results of LEBRA is totally same and the results of SpBC are meaningless since it is broken when there are outliers.

Now, let us discuss the interpretation of VI and NMI results for both even and non-even cases when the data matrix is contaminated. It is observed from NMI that SpBC outperforms the other two approaches at all noise level and its performance does not relate to the noise level at all. For VI, the performance of SpBC is confusing as well. It outperforms COBRA when the noise level is at 100 and dominates LEBRA after the noise level is 500. The overall trend of the
values of \( VI \) for SpBC is a flat line. Its performance seems not relate to the noise level as well. In fact, the result that SpBC returns are just a constant matrix. The total number of biclusters is one. Equivalently, the result of Sparse Biclustering returned is just a random guess. We will provide a small example here to show how this random guess affect the metric.

Suppose we have a 4-by-3 matrix, we have four biclusters in total. Let us label them 1 to 4. Then the assumed vectorized bicluster assignment vector is \( \mathbf{v} = [1, 1, 2, 2, 3, 3, 4, 4, 3, 3, 4, 4] \). Suppose \( \mathbf{v}_1 \) is a flawless vectorized bicluster assignment vector given by some perfect biclustering approach. Then the \( VI(\mathbf{v}, \mathbf{v}_1) = 0 \) as expected, which means the information carried by \( \mathbf{v} \) and \( \mathbf{v}_1 \) are the same as their shared information, i.e. \( \mathbf{v} \equiv \mathbf{v}_1 \). Now suppose \( \mathbf{v}_2 = [1, 1, 2, 1, 1, 2, 2, 2, 2, 3, 4, 4] \) is a vectorized assignment vector given by some regular biclustering algorithm. We can see there are some wrong assignments but this algorithm is trying to find the hidden structure. The \( VI(\mathbf{v}, \mathbf{v}_2) = 2.13 \). Now, suppose we are given an uniform assignment, i.e., \( \mathbf{v}^* = [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1] \). Clearly, this uniform assignment is not even trying, but the \( VI(\mathbf{v}, \mathbf{v}^*) = 1.92 \).

### 2.11 Application to Genomics

To evaluate the performance of LEBRA in the real application, we choose to apply LEBRA to the lung cancer data studied by Lee et al. (2010), Tan and Witten (2014) and Chi et al. (2017). We have chosen 500 genes with the greatest variance from the 12625 genes. There are four subgroups in total among subjects.

Firstly we will discuss how the solution \( U_\lambda \) changes as we increase the value of the penalty parameter, \( \lambda \). Figure 2.6 shows several snapshots of the entire solution path of the LEBRA. There seems like no obvious checkerboard structure as \( \lambda \) is small. This is because the strength of the penalty parameter forcing different cells together is not strong enough. As \( \lambda \) increases, the strength of imposing a checkerboard structure on the estimate is stronger and stronger. After reach certain threshold value of \( \lambda \), more homogeneous cells merged into the same bicluster. All of the cells in the dataset fusssed into the same bicluster, i.e. a constant matrix. If \( \lambda \) is large
Figure 2.6: Snap shots of the LEBRA solution path of the lung cancer dataset as the penalty parameter \( \lambda \) increases. All of the matrices have been forced to have the same column order and row order for comparison purpose. It shows the whole trend of the LEBRA solution path from the estimate with not enough strong penalty strength to the over-smooth constant estimate. It shows the whole trend of the LEBRA solution path from the estimate with not enough strong penalty strength to the over-smooth constant estimate.

Secondly, we conduct a simulation experiment based on the lung cancer data with artificial outliers to see the performance of all three methods under different percentage of outliers. The performance refers to the consistency between the estimation on the clean lung cancer data and the estimations on the contaminated lung cancer data. Notice the consistency we have just mentioned is not the statistical consistency for an unbiased estimator. A biclustering algorithm is robust if the estimation based on the contaminated data are not quite different from the estimation based on the clean data. The simulation is conducted under four different percentage of outliers, they are 1%, 5%, 10% and 15%. Under each percentage of outliers, we simulate 50 replicates. The total size of the data is 500 times 56, 28000 entries in total. Therefore, 1% of the matrix represents 280 entries of the data. Outliers are those cells whose values are outside 5 to 6 times standard deviation of the clean lung cancer data and are generated under an uniform distribution. Other parameters are properly chosen accordingly. The COBRA used in this simulation experiment is the adaptive COBRA. For detailed explanation of adaptive COBRA, please see Chi et al. (2017). Total nine measures are used in this simulation experiment.
Four of them are shown here. Please see the Appendix for the full numerical results of nine measures.

The detailed numerical results are shown in Figure 2.7. It is observed that LEBRA shows great resistance to outliers compared to the other two approaches. COBRA has the weakest resistance against the outliers. SpBC is more robust than COBRA but still not as robust as LEBRA. The reason why SpBC outperforms COBRA may be due to the usage of $l_1$ penalty imposed on the number of centroids. Outliers are supposed to create more small biclusters but sometimes the strength of outliers is much weaker than the strength of the penalty so that the estimation might be affected but not significantly. Another possible reason might be the upper bound of the maximum number of column clusters and row clusters we had set for the sparse biclustering algorithm. The maximum number of column clusters and row clusters for SpBC are set to be 10 in this simulation experiment, which might have provided “additional” information to SpBC so that it can outperform the COBRA to some extent. It is also observed that VI indicates COBRA is more robust than SpBC when the percentage of outliers is less than 10%. This is against the conclusions of the three other measures. The reason for this is the same as the reasoning we have conducted in the simulation section. Over-smoothed constant matrix estimations arrive at a higher value for VI than dedicated wrong estimations. In fact, the COBRA estimation is a constant matrix when the number of outliers is less than 10%; this is because the real signal of lung cancer data is too weak to be detected given the appearance of outliers of relatively large magnitude. However, when there is more than 10% of the outliers, COBRA not only fails to detect the true signal but also the COBRA estimation is seriously biased by the outliers. As a result, COBRA tends to place each of the outliers in its own bicluster. This is why the value of VI at the contamination rate of 5% is quite different from the value of VI at the contamination rate of 10%.

The running time for three approaches based on 50 replications under four different percentage of outliers are shown in Figure 2.8. SpBC(20,20) means the maximum number of row clusters and column clusters are set to be 20 while SpBC in default we set the upper bounds to be 10.
Figure 2.7: Performance of COBRA, LEBRA and SpBC under the measure of Rand Index, Variational Information, Adjusted Rand Index and Normalized Mutual Information

The simulation is conducted on the server that has the configuration: DELL R815 Quad Processor AMD Opteron 16 core 2.3 GHz machines with 512GB RAM each running 64Bit Ubuntu Linux Version 16.04. The running time results are graphically presented in Figure 2.7. It is clear that COBRA is the fastest approach compared to the slowest LEBRA. The running time of SpBC is faster than LEBRA but still slower than COBRA. Although the LEBRA is slower than SpBC, the running timing of LEBRA is very close to the running time needed by Sparse Biclustering. In addition, the running time of Sparse Biclustering increases dramatically as we increase the upper bound of the maximum possible row clusters and column clusters. Moreover, the performances of SpBC are almost the same under both cases. Therefore, without additional information on the maximum number of row clusters and column clusters, we should not set a small value for the upper bound of the number of row clusters and column clusters. The reason
we use 10 and 20 here is just for convenient purpose. Therefore Sparse Biclustering is supposed to run longer time if there are not further information regarding the maximum number of row clusters and column clusters.

Figure 2.8: Running time of the previous simulation experiment needed by COBRA, LEBRA and SpBC

In the final, we increase the magnitude of the outliers gradually to see how the performance of three approaches change. The corresponding numerical results are shown in Figure 2.9. The LEBRA estimations barely change as we increase the magnitude of the outliers, while the other two approaches’ estimations are biased dramatically by the larger outliers. In fact, if the magnitude of the outliers are not big enough, LEBRA estimation will be twisted towards to outliers as well. This is as expected because not big enough outliers might be called normal as well, which will confuse the algorithm and twist the estimation. This happened not only for LEBRA
but also for COBRA and SpBC as well. This is why we start the simulation experiment at “reasonable" range of magnitude of the outliers, say five times of the standard deviation of the original lung cancer data. Overall, LEBRA outperforms COBRA and SpBC. The estimation of

![Figure 2.9: Performance of COBRA, LEBRA and SpBC for different magnitude of outliers](image)

LEBRA barely change as we increase the magnitude of outliers, which shows a strong robustness of LEBRA. For SpBC and LEBRA, the agreement between the estimation over the clean data and contaminated data keeps decreasing as an increase in the magnitude of the outliers. There are some disagreements under different measures. For VI, the values of VI for SpBC actually decreases as we increase the magnitude of the outliers, which is against our intuition. Let us first see what happened. SpBC provides 2-by-2 checkerboard structure as an estimation at level 5 while a constant matrix at level 10. As we have explained earlier, VI tends to penalize harder on an under-smooth result than a over-smooth result. This explains why the value of VI of a
constant matrix might be higher than the value of the VI of a 2-by-2 checkerboard structure. Also notice there is a big increase in the values of VI of COBRA estimations at this range. COBRA gives an constant matrix estimation at level 5. But when the outlier level increases up to 10, COBRA gives a very messy estimation tending to put every single outlier into a bicluster while still keeping the checkerboard structure. This is because a row with outliers is too “pricy" for COBRA to be fused with other rows when the magnitude of the outliers are big enough.

### 2.12 Discussion

Our proposed method for biclustering, LEBRA can be considered as an iteratively reweighted COBRA. Internally, LEBRA is iteratively solving subproblem using COBRA over the convex combination of original data matrix and the intermediate estimation at each iteration. As for weights, they are determined by the Gaussian kernel and being updated throughout the LEBRA. The most important of the improvement of LEBRA over existing biclustering methods is its outstanding robustness against the outliers. It has been observed from our simulations that the biclustering estimation from LEBRA are almost as the same as the estimations on the clean lung cancer data, while the estimations from other biclustering method biased more or less by the appearance of the outliers. As we have explained in the previous section, one way to understand its robustness is the convex combination of the data and intermediate estimation. The values of the Gaussian kernel determines how much the LEBRA should “trust" data. Abnormal entries will be granted smaller weights while larger weights on the intermediate estimations.

Robustness does not come for free. There are many weaknesses in LEBRA as the prices for the robustness. Firstly, the non-convexity of the LEBRA error criterion is one of the biggest drawback. There is no guarantee the LEBRA will converge to a global minimum. However, locally, when the intermediate estimation is very close to the ground truth, Taylor expansion of the LEBRA criterion has shown the behavior of LEBRA is similar to the behavior of a scaled COBRA problem. In other word, as long as we are close enough to the ground truth, the behavior of LEBRA is not that bad. Secondly, like the COBRA, the checkerboard structure is
also a drawback. Each observation will only be assigned to exactly one bicluster and overlapping biclusters has been excluded for simplicity purpose, while frameworks allowing for overlapping biclusters might be much more flexible. Finally, The corresponding mathematical theories are hard to prove because of the complicated formulation of LEBRA error criterion. As for now, we are only able to prove the fact when the index of the LEBRA sequence is bigger than a finite threshold value, there exists a subsequences in LEBRA sequence will converge to a stationary point given a compactness of LEBRA sequence. The compactness for now has to be assumed. This is because the LEBRA error criterion is not coercive (It will not blow up if the arguments blow up). But empirically, we haven’t seen any cases that LEBRA sequences are unbounded based on all of the simulations have been done in this article for now.
Chapter 3

Hypothesis Testing for Tensor Regression with Application to Gene-set Integrative Analysis of Multi-omics Data

3.1 Abstract

Integrative multi-omics analyses, which integrate complementary level of information from different molecular events, has great potentials to detecting novel disease genes and pathways, enhancing risk prediction and elucidating disease mechanisms. The importance of jointly modeling multi-omics data has been recognized, with one major focus on leveraging the relationships among different platforms and identifying gene-sets associated with clinical outcomes. A common strategy for the gene-set integrative analysis is to regress the clinical outcomes on all genomic variables in a gene set. However, the joint modeling methods encounter the challenges of high-dimensionality inference, especially the sample size is usually moderate either due to research resources or missing data. In this work, we propose a tensor-based framework to enhance model
efficiency while retaining the variable-wise resolution. By accounting for the inherent matrix structure of an individual’s multi-omics data to reduce the number of parameters, the proposed tensor methods naturally incorporate the relationship among genomic effects and boost the efficiency of high-dimensional modeling. We explore the variable-specific test procedure under tensor regression framework. We derive an alternative variance formula of the coefficient estimator that does not depend on the permutation matrix as required in previous work, which permits a systematic expression of the variance formula in computational coding. We evaluate the performance of the tensor-based test using simulations and demonstrate its utility by applying to the datasets from the Cancer Genome Atlas (TCGA).

3.2 Introduction

Integrative multi-omics studies consider the molecular events at different levels, e.g., DNA variations, epigenetic marks, transcription events, metabolite profiles, and clinical phenotypes. With recent technological advances, an increasing number of projects, e.g., The Cancer Genome Atlas (TCGA), International Cancer Genome Consortium (ICGC), the Encyclopedia of DNA Elements (ENCODE), and GTEx Project, have measured multiple genomic features on the same samples. By incorporating complementary level of information, integrative analyses of multi-platform data have helped to identify novel disease genes and pathways (Assié, Letouzé, Fassnacht, Jouinot, Luscap, Barreau, Omeiri, Rodriguez, Perlemoine, René-Corail, et al. (2014)), enhance risk prediction (Seoane, Day, Gaunt, and Campbell (2013)), elucidate disease mechanisms (Chow, Pramparo, Winn, Barnes, Li, Weiss, Fan, Murray, April, Belinson, et al. (2012)) derive mechanistic insight of trait-associated DNA variants (Mo, Wang, Seshan, Olshen, Schultz, Sander, Powers, Ladanyi, and Shen (2013)), and identify plasma biomarkers when tissue samples are difficult to obtain (Bowler, Bahr, Hughes, Lutz, Kim, Coldren, Reisdorph, and Kechris (2013)).

One major focus of integrative multi-omics analysis has been on studying the relationships among different platforms and identify regulatory modules or gene-sets that are associated
with (Liu et al. (2013); Shi et al. (2014); Tyekucheva et al. (2011); Wang et al. (2012); Xiong, Ancona, Hauser, Mukherjee, and Furey (2013)) or predictive of (Bennett, Xiong, Mukherjee, and Furey (2012); Seoane et al. (2013); Srivastava, Wang, Manyam, Ordonez, and Baladandayuthapani (2013)) clinical outcomes (Kristensen, Lingjærde, Russnes, Vollan, Frigessi, and Børresen-Dale (2014)). Gene-set analysis with multi-platform data take all omics levels of the gene set into account, and the missing signals in one platform might be captured when combined with other platforms. By assessing gene effects in a functional context (e.g., pathways and biological processes), gene set integrative analysis improves the detectability, reproducibility and interpretability of significant findings and facilitates the construction of follow-up biological hypotheses (Sass, Buettner, Mueller, and Theis (2013); Tyekucheva et al. (2011); Xiong et al. (2013)).

Gene-set integrative approaches can be roughly classified into (a) "meta"-based methods, which combine the results of multiple single-gene (Louhimo and Hautaniemi (2011); Xiong et al. (2014)), single gene-set, or single platform analyses (Ma and Huang (2009); Ma et al. (2011); Lin et al. (2014)) and (b) "joint-modeling"-based methods, which regress the clinical outcomes on all genomic variables from multiple platforms in a gene set (e.g., Tyekucheva et al. (2011); Huang et al. (2012); Wang et al. (2012); Hu and Tzeng (2014); Chu and Huang (2017)). Joint modeling approaches tend to outperform meta-based approaches (Tyekucheva et al. (2011); Huang et al. (2012); Wang et al. (2012); Hu and Tzeng (2014)) because they conduct simultaneous data integration across genes and platforms and account for biological relationship among genomic variables. However, model-based methods encounter the challenges of high dimensional genomic variables, especially the case where the sample size is usually moderate either due to research resources or missing data. Various strategies have been proposed to address the issue of high-dimensional genomic variables, e.g., Bayesian approaches (e.g., Wang et al. (2012)), penalization regressions (e.g., Liu et al. (2013); Shi et al. (2014); Ma and Huang (2009); Ma et al. (2011); Jiang et al. (2016)), and dimension-reduction based methods, e.g., principal component analysis (PCA) as discussed in (Meng et al. (2016)).
In this work, we propose an alternative strategy based on tensor regression framework (Zhou, Li, and Zhu (2013)) to enhance modeling efficiency in gene-set integrative analysis. A tensor is a multi-dimensional array (e.g., a vector is an order-1 tensor and a matrix is an order-2 tensor). Because an individual’s gene-set data from multi-platforms have a $P \times G$ matrix structure, where $P$ (or $G$) is the total number of platforms (or genes), the gene-set data of the entire samples form an order-3 ($P \times G \times n$) data tensor. Consequently, the regression coefficients form a $P \times G$ matrix (denoted by $B$ hereafter) and we can utilize the matrix structure of $B$ to improve the statistical inference. Specifically, we explore the potential low rank structure of $B$ induced by biological relationship among genomic variables so to use less degrees of freedoms to model the multi-platform variables. Comparing to PCA-based methods, which only output pathway-level associations, the tensor-based methods are able to retain the variable-wise resolution during dimension reduction and reveal associations at gene and platform levels. Comparing to penalized-based regressions, e.g., imposing lasso penalty on the vectorized regression coefficients (Liu et al. (2013); Shi et al. (2014)), tensor-based modeling gains additional model efficiency by accounting for the inherent structure among genomic effects to reduce the number of parameters. More importantly, because the tensor model can achieve dimensional reduction even if the coefficient matrix $B$ has a non-sparse structure, our inference framework remain useful under polygenic etiology, where the signal sparsity can be low due to the likely involvement of many small-effect genes, rather than a few strong-effect genes.

Tensor-based modeling has been used in a variety of applications and demonstrated utility, e.g., association of neuroimaging data (Zhou and Li (2014); Zhou et al. (2013)), identification of gene-gene interactions (Hung, Lin, Chen, Wang, Huang, and Tzeng (2016); Wu, Huang, and Ma (2018)), integrations of multiple expression data sets (Li, Dai, Kang, and Zhou (2014); Omberg, Golub, and Alter (2007)) and identification of co-expression modules among multi-platform data (Li, Liu, Zhang, Li, Waterman, and Zhou (2011)). However, current tensor-based methods focus mainly on predictive models (Zhou et al. (2013); Li, Suk, Shen, and Li (2016); Wu et al. (2018)) and only a few works deal with signal detections. These methods, which focus on detecting
gene-gene or gene-environment interaction signals, are either based on variable selection and not providing p values or are designed to detect global signals. For example, Hung et al. (2016) consider rank-1 tensor interaction model as a screening tool; Wu et al. (2018) use penalization techniques to select significant gene-gene interactions; and Hung and Jou (2016) derive a global interaction test for tensor regression.

In this work, we use the tensor regression (Zhou et al. (2013)) as a framework to generalize the traditional regression from 2-dimension data (e.g., \( n \times PG \)) to 3-dimensional data (e.g., \( n \times P \times G \)). Aiming to generate inferences results that can help to comprehend the relationship between genomic variations and outcomes, we explore the variable-specific testing procedure under the tensor regression framework. Finally, we derive an alternative variance formula of the tensor coefficient estimator that does not depend on the permutation matrix as required in the previous work (Zhou et al. (2013)). The new variance formula avoids the non-trivial, case-by-case specification of the permutation matrix and permits a systematic expression of the variance formula in computation coding. We provide an R package that implements the proposed variable-specific tensor regression test.

### 3.3 Methods

#### 3.3.1 Notation and Preliminaries

In this section, we give a brief introduction to tensor and define the relevant operations used throughout the article. Because the data of an individual in gene-set multi-platform analysis is a matrix (i.e., order-2 tensor), we will focus our illustration on order-2 tensors for clarity purposes, although the results can be straightforwardly extended to higher order.

Denote an order-\( J \) tensor as \( \mathcal{X} \in \mathbb{R}^{p_1 \times p_2 \times \ldots \times p_J} \), where \( p_j \) is the dimension of the \( j \)-th vector space. Operator \( \text{vec}(\bullet): \mathbb{R}^{p_1 \times p_2 \times \ldots \times p_J} \rightarrow \mathbb{R}^{\prod_j p_j} \) is a function mapping an order-\( J \) tensor from \( \mathbb{R}^{p_1 \times p_2 \times \ldots \times p_J} \) to a vector of length \( \prod_{j=1}^{J'} p_j \) in \( \mathbb{R}^{\prod_j p_j} \) following a column major. For example,
assume \( A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \), then \( \text{vec}(A) \) reshapes \( A \) into a vector as:

\[
\text{vec}(A) = \begin{bmatrix} a_{11} \\ a_{21} \\ a_{12} \\ a_{22} \end{bmatrix}
\]

Although there are several types of multiplications for tensors, here we focus on the inner product, Kronecker product and Box product as defined below. To define these products, consider matrix \( A \in \mathbb{R}^{m_1 \times n_1} \) and matrix \( B \in \mathbb{R}^{m_2 \times n_2} \). First, the inner product of \( A \) and \( B \) is defined as:

\[
\langle A, B \rangle = \text{vec}(A)^T \text{vec}(B)
\]

Also we have the following result by the cyclic property of trace:

\[
\langle A, BC \rangle = \text{tr}(A^T BC) = \text{tr}(CA^T B) = \langle C^T, A^T B \rangle,
\]

which will be used when deriving our estimation algorithm in Section 3.3.3.

Given matrices \( A, B \) and \( X \) of appropriate dimensions, an important equality between \( \text{vec}(\bullet) \) operator and the Kronecker product is:

\[
\text{vec}(AXB) = \left( B^T \otimes A \right) \text{vec}(X).
\]

On the other hand, in matrix calculus, one often encounters the following when \( \text{vec}(X^T) \) is
involved in a function and one has to take the function’s derivative with respect to \( \text{vec}(X) \):

\[
\text{vec}(AX^TB) = \left( B^T \otimes A \right) \text{vec}(X^T) = \left( B^T \otimes A \right) T \text{vec}(X),
\]

where \( T \) is a permutation matrix adjusting the positions of entries in \( X \) so that \( T \text{vec}(X) = \text{vec}(X^T) \). The involvement of permutation matrix \( T \) is tedious to deal with for several reasons: (1) There is no exact close-form presentation for \( T \) but \( T \) will be involved in the variance of the coefficient estimators of the tensor regression (e.g., Zhou et al. (2013)) (2) The involvement of the permutation matrix results in higher computational cost because additional time is required for the matrix multiplication with \( T \) and the storage of permutation matrix takes substantial amount of space even in the sparse format when the data size is large.

To bypass the permutation matrix, we consider the Box product defined as follows. Consider \( A \in \mathbb{R}^{m_1 \times n_1} \) with entries \( A_{ij} \) and \( B \in \mathbb{R}^{m_2 \times n_2} \) with entries \( B_{\ell k} \). Their Box product, denoted by \( C = A \boxtimes B \), is a \( m_1 m_2 \times n_1 n_2 \) matrix with its entries:

\[
C_{(i-1)m_2 + \ell, (k-1)n_1 + j} = A_{ij} B_{\ell k}.
\]

In Appendix B.1, we provide a couple of examples to illustrate the difference between Kronecker product and Box product. With Box product, we can rewrite Equation (3.2) as below and avoid the permutation matrix.

\[
\text{vec}(AX^TB) = \left( B^T \boxtimes A \right) \text{vec}(X)
\]

3.3.2 Tensor Regression Model for Integrative Gene-Set Analysis

Consider a dataset of \( n \) samples. Let \( Y = (y_1, y_2, \ldots, y_n)^T \) be the continuous clinical outcome vector of interest. The multi-platform data of the \( n \) samples are stored in an order-3 tensor, \( \mathcal{X} \in \mathbb{R}^{P \times G \times n} \), where \( P \) is the number of platforms and \( G \) is the number of genes. Let \( X_i \) be the
$i$-th slice of $\mathcal{X}$ with respect to the third order, i.e., $\mathcal{X}(:,;i)$. Then $X_i$ is the design matrix for the $i$-th sample and $\mathcal{X} = \{X_i\}_{i=1,...,n}$. Also define $z_i$ the covariate vector of sample $i$ including the intercept. In multi-platform analysis, the effects of different platforms for a gene and/or the effects of different genes within a platform can be highly structured due to the interrelationships among different levels of molecular events. This implies that the coefficient matrix of the genomic variables, denoted by $B$, is likely to have a low-rank structure. Therefore, we posit the following tensor regression model to study the integrative gene-set effects of multi-platform:

$$y_i = z_i^T \beta + \langle X_i, B \rangle + \epsilon_i \text{ with } B = B_1B_2^T,$$

(3.3)

where $\langle \bullet, \bullet \rangle$ denotes the inner product, $B \in \mathbb{R}^{P \times G}$ is the parameter matrix for the gene-set genomic variables; $\beta$ is the parameter vector of the covariates; and $\epsilon_i$ is the error term for $i$-th sample following a normal distribution with mean zero and variance $\sigma^2$; $B_1 \in \mathbb{R}^{P \times R}$, $B_2 \in \mathbb{R}^{G \times R}$, and $R \leq \text{min}(P,G)$. Model (3.3) changes the number of genomic parameters from $P \times G$ to $R \times (P + G)$. When $R < \text{min}(P,G)$, the tensor regression uses the inherent structure among genomic effects to reduce the parameter dimension. When $R = \text{min}(P,G)$, the rank of matrix $B$ is equal to the maximal possible rank of a $P \times G$ matrix. Under this scenario, model (3.3) is equivalent to the linear regression with vectorized $B$, and the tensor regression is just a compact and structural formulation of the linear regression. A proof of the equivalence has been shown in Appendix B.4.

### 3.3.3 Estimation

Zhou et al. (2013) provides a general procedure for estimating the tensor parameters. In our case (matrix regression with $P$-by-$G$-by-$n$ tensor), we derive the detailed estimation procedure according to Zhou’s estimation procedure. In matrix regression case, the tensor rank decomposition of a matrix is just $B_1 \Lambda B_2^T$ where $\Lambda$ is a diagonal matrix. Zhou’s general estimation procedure, we should update $B_1$, $B_2$ and $\Lambda$ cyclically. However, in matrix regression case, we
can absorb $\Lambda$ into $B_1$ and $B_2$ so that we can omit the additional step updating $\Lambda$

For estimating $B_1$ and $B_2$, recall that the term involved is $\langle X_i, B_1 B_2^T \rangle$. By the cyclic property of the trace estimator, we have:

$$\langle X_i, B_1 B_2^T \rangle = \langle X_i B_2, B_1 \rangle = \langle X_i^T B_1, B_2 \rangle$$

Consequently, the response vector $y$ is linear in $B_1$ given $B_2$ and linear in $B_2$ given $B_1$ for Gaussian response variable.

$$E(y_i | X_i, z_i) = z_i^T \beta + \langle X_i, B_1 B_2^T \rangle$$

$$= z_i^T \beta + \langle X_i, B_2, B_1 \rangle = z_i^T \beta + \langle \{A(B_2^{(j)})\}, B_1 \rangle$$

$$= z_i^T \beta + \langle X_i^T B_1, B_2 \rangle = z_i^T \beta + \langle \{B_1^{(j+1)}\}, B_2 \rangle,$$

where $[\bullet]_i$ is the $i$-th row of $\bullet$, $A(B_2) = \begin{bmatrix} \text{vec}(X_1 B_2)^T \\ \text{vec}(X_2 B_2)^T \\ \vdots \\ \text{vec}(X_n B_2)^T \end{bmatrix}$, $B_1 = \begin{bmatrix} \text{vec}(X_1^T B_1)^T \\ \text{vec}(X_2^T B_1)^T \\ \vdots \\ \text{vec}(X_n^T B_1)^T \end{bmatrix}$, $B_2^{(j)}$ is the estimates of $B_2$ in the $j$th iteration. (3.4) and (3.4) show that we can update either $B_1$ or $B_2$ by solving a simple least square problem. Consequently, we obtain the following estimation algorithm:

**Algorithm 3** Tensor Regression in order-3 case for Gaussian outcome variables (TR-G)

1. procedure TR-G($B_1^{(0)}, B_2^{(0)}, \beta^{(0)})$
2. repeat
3. $\beta^{(j+1)} = LM(\beta | y, Z, B_1^{(j)}, B_2^{(j)})$
4. $B_1^{(j+1)} = LM(B_1 | y, \{Z, A(B_2^{(j)})\}, \beta^{(j+1)})$
5. $B_2^{(j+1)} = LM(B_2 | y, \{Z, B(B_1^{(j+1)})\}, \beta^{(j+1)})$
6. until convergence
7. end procedure
In Algorithm 1, \( \text{LS}(B_1 \mid Y, X, A, \ldots) \) indicates a least square problem with respect to \( B_1 \) given a response vector \( Y \) and a design matrix \( X \), conditioning on \( A \) or other variables. The estimation procedure of the tensor regression coefficients is a series of least square problems. The definition of a standard least square (LM) problem is defined as following:

\[
LM(\beta \mid y, X, \text{anything that are fixed here}) = \min_\beta \frac{1}{2} ||y - X\beta||^2
\]

Notice that model (3.3) is a non-convex model with respect to \( B_1 \) and \( B_2 \) as a whole but it is convex in either \( B_1 \) or \( B_2 \). Such function is called a piece-wise convex function and its optimality conditions have been widely discussed (e.g., Tsevendorj (2000); Laude, Wu, and Cremers (2017)). Once the model is not convex, there is no guarantee to get a global minima. Therefore, local minima and non-reproducibility are always two main issues of a non-convex model. In our estimation procedure, we simply implement our algorithms multiple times with multiple random initial values and choose the one with the minimum objective value as the final estimate.

The parameterization in the tensor regression model is non-identifiable, and additional constraints are needed when updating \( B_1^{(i)} \) and \( B_2^{(i)} \). The non-identifiability issue can arise from two complications: (1) permutations and (2) scaling. To see this, consider an orthogonal matrix \( O \) such that \( OO^T = O^TO = I \). Then given the same \( B \), multiple solutions are available:

\[
B = B_1B_2^T = B_1OO^TB_2^T.
\]

Both \( B_1, B_2 \) and \( B_1O, O^TB_2^T \) are solutions to the original problem. If \( O \) is a scalar, it is referred to as scaling; if \( O \) is a orthogonal matrix, it is referred to as permutations. The constraints that we impose restrict \( B_1 \) and \( B_2 \) to take the following forms:

\[
B_1 = \begin{bmatrix} I_R \\ B_{12} \end{bmatrix} \quad \text{and} \quad B_2 = \begin{bmatrix} B_{21} \\ B_{22} \end{bmatrix}
\]
such that $B_1B_2^T = B$, where $I_R \in \mathbb{R}^{R \times R}$ is an identity matrix, $B_{12} \in \mathbb{R}^{(P-R) \times R}$, $B_{21} \in \mathbb{R}^{R \times R}$, and $B_{22} \in \mathbb{R}^{(G-R) \times R}$.

One crucial task in low-rank tensor regression is to choose an appropriate rank for the model. To identify the optimal rank $R$ of the tensor model, we first perform model fitting using Algorithm 1 for a given rank $R = 1, \cdots, \min(P, G)$. We consider two rank selection methods, (a) using Akaike information criterion (AIC), i.e., deviance + $k_r$, and (b) using Bayesian information criteria (BIC), i.e., deviance + $k_r \log(n)$, where deviance is $\frac{1}{2} \sum_{i=1}^{n}(y_i - z_i^T \hat{\beta} - \langle X_i, \hat{B}_1\hat{B}_2^T \rangle)^2$, and $k_r$ is the degree of freedom in the rank-$r$ model. As shown in the Section B.2, we found that AIC performs better than BIC in identifying the correct rank.

### 3.3.4 Association Test for Single Genomic Variables

To assess the significance of the effect of gene $g$ in platform $p$, we use a Wald’s test for $H_0 : B_{pg} = 0$ based on the tensor model (3.3). The test statistic is $T_{pg} = \hat{B}_{pg} / s.e.(\hat{B}_{pg})$, where $\hat{B}_{pg}$ is the tensor model estimate obtained from Algorithm 1 and $s.e.(\hat{B}_{pg})$ is its standard error with the formula given in the proposition below. The proposition also shows that $T_{pg}$ follows a normal distribution.

**Proposition 3.3.1** Given a tensor regression $y_i = z_i^T \beta + \langle X_i, B_1B_2^T \rangle + \epsilon_i$ for $i = 1, \ldots, n$ and $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$, we have

$$vec(\hat{B}_1\hat{B}_2^T) \sim \mathcal{N}
\left(vec(B_1B_2^T), \ (I \otimes B_1)V_2(B_1^T \otimes I) + (B_2 \otimes I)V_1(B_2^T \otimes I) + 2(I \otimes B_1)I_{B_1B_2} (B_2^T \otimes I)\right)$$

(3.4)
where

\[
V_1 = \sigma^2 (I_{B_1} - I_{B_2}B_1[I_{B_2}]^{-1}I_{B_1}B_2)^{-1}
\]
\[
V_2 = \sigma^2 (I_{B_2} - I_{B_1}B_2[I_{B_1}]^{-1}I_{B_2}B_1)^{-1}
\]
\[
I_{B_1B_2} = \frac{1}{\sigma^2} \sum_{i=1}^{n} vec(X_i^TB_1)vec(X_iB_2)^T
\]
\[
I_{B_1} = \frac{1}{\sigma^2} \sum_{i=1}^{n} vec(X_iB_2)vec(X_iB_1)^T
\]
\[
I_{B_2} = \frac{1}{\sigma^2} \sum_{i=1}^{n} vec(X_i^TB_1)vec(X_i^TB_1)^T
\]

The proof is given in Appendix B.3. We note that Zhou et al. (2013) describes the asymptotic property of the parameter matrix. (3.4) can be viewed as a special case focusing on matrix-covariate regressions. The proposition provides an explicit form of variance formula that does not depend on the permutation matrices and shows how the variance depends on the low-rank components $B_1, B_2$ and the data $(y, X)$. Moreover, the introduction of the box product makes the variance formula more intuitive: we can see the variance formula consists three parts; (1) variance of $B_1$ (2) variance of $B_2$ and (3) the co-variance of $B_1$ and $B_2$.

### 3.4 Simulation Studies

We conduct simulations to evaluate the performance of the proposed tensor regression (TR in short) for identifying important genomic variables. We consider two benchmark methods that represent the two common strategies applied on vectorized genomic variables: (1) linear regression (LM in short) and (2) penalized regression via lasso (LASSO in short) using BIC to select the tuning parameter. LM is conducted using R the package 'MASS' and LASSO is conducted using the R package 'msgps'.

We generate the design matrix of an individual based on the pathway, "REACTOME PROCESSING OF CAPPED INTRON CONTAINING PRE MRNA" as defined in MSigDB, with
data obtained from the TCGA breast cancer dataset as detailed in Hu and Tzeng (2014). The dataset consists of copy number variation (CNV), methylation and RNA-Seq values of 74 genes from 530 samples, and the values are standardized to have mean 0 and standard deviation 1 within each platform.

To reduce the complications caused by correlations among genes when evaluating true positives and false positives of the variable selection results, we simulate the genomic values \( X_i \) using the following procedure. For a given gene \( g, g = 1, \ldots, 74 \), we randomly select an individual’s platform design vector of length \( P = 3 \), and repeat this sampling process with replacement until we obtain the data for \( n = 500 \) samples. The data dimension of \( (n, P, G) = (500, 3, 74) \) is considered so to evaluate the performance of TR compared to LM and LASSO. We also consider a higher data dimension of \( (n, P, G) = (500, 3, 170) \) by sampling with replacement from the 74 genes; this scenario allows us to evaluate the performance of TR when only TR and LASSO are feasible.

Given an individual’s genomic data \( X_i \), we simulate the outcome value \( y_i \) from the model

\[
y_i = z_i^T \beta + \langle X_i, B \rangle + \epsilon_i,
\]

where the error term \( \epsilon_i \) and the \( 5 \times 1 \) covariate vector \( z_i \) are simulated from standard Normal distribution, \( \beta = (1, 1, 1, 1, 1)^T \), and non-zero entries of coefficient matrix \( B \) are generated from Normal with mean \( C \) and standardized deviation \( C^2/4 \). We consider three signal patterns of \( B \) (i.e., the shape of the non-zero coefficients in \( B \)) as shown in Figure 3.1): (1) vertical rectangular shape of \( B \) with rank 1, which is referred to as the "I" shape and represents a few causal genes with non-zero effects in all platforms; (2) reverse T shape of \( B \) with rank 2, which is referred to as the "T" shape and represents a few master CNVs and methylations affecting the expressions of multiple genes; (c) random shape of \( B \) with rank 2, which is referred to as the "Random" shape.

### 3.4.1 Performance of Detecting Causal Variables

Next we evaluate the performance of the estimated variance obtained by evaluating Equation (3.4) at the estimated \( B_{pg} \). As tensor estimation is based on the non-convex optimization, we
are aware of numerical issues in real practice, such as that the estimates may not be the global optimizer and result in unstable variance estimates. Therefore besides considering multiple initial values to increase the chance of reaching the global minima. We also adapt the following strategy to improve the numerical performance of the variance estimates: For a given (replicated) dataset, we first obtain the tensor estimate (denoted as $\hat{B}_{org}^{pg}$) and compute the variance based on it, i.e., $V\left(\hat{B}_{pg}; \hat{B}_{org}^{pg}\right)$. We then obtain additional 9 bootstrap samples, with each of which we compute tensor estimates (denoted as $\hat{B}_{pg}^b$ with $b = 1, \cdots, 9$) and compute the variance based on them, i.e., $V\left(\hat{B}_{pg}; \hat{B}_{pg}^b\right)$’s. The final variance estimate, which is denoted as $V\left(\hat{B}_{pg}\right)$, is the median of the 10 variance estimates (i.e., $V\left(\hat{B}_{pg}\right) = \text{median}\{\hat{B}_{org}^{pg}, \hat{B}_{pg}^1, \cdots, \hat{B}_{pg}^9\}$).

Figure 3.2 shows the bias, i.e., $V(\hat{B}_{pg}) - V_{\text{emp}}(\hat{B}_{pg})$, for $B_{pg} = 0$’s and $B_{pg} \neq 0$’s under different B shapes and sample sizes. We see that the variances tend to be slightly overestimated when $n = 500$; the bias decreases as sample size increases. When the sample size increases to 1000 or 2000, the bias magnitude decreases and the bias distribution emerges to be symmetric around zero.

### 3.4.2 Performance of Detecting Causal Variables

We evaluate the false positive rates (FPR) and true positive rate (TPR) of LM, TR and LASSO in detecting associated genomic variables under different signal patterns of B with the setting of $C = 0.25$, $n = 500$, $P = 3$ and $G = 74$. We also consider a composite metric called $g$ measure (Powers (2011)) to assess the overall selection performance. The $g$ measure is the
Figure 3.2: Bias of variance estimates for the tensor coefficient estimates with bias= estimated variance−empirical variance. The 3 panels from left to right represent 3 different signal patterns in B: "I" shape \( (R = 1) \), "Random" shape \( (R = 2) \), "T" shape \( (R = 2) \); . Orange boxplots are biases for \( B_{pq} = 0 \) and blue boxplots are biases for \( B_{pq} \neq 0 \).
geometric mean of the TPR and 1–FPR with 1 being the best and 0 being the worst. FPR is obtained by first computing the fraction of selected variables among the null variables (i.e., $B_{pg} = 0$) in each replication, and then averaging across a total of 50,000 replications. TPR is obtained by first computing the fraction of selected variables among all causal variable (i.e., $B_{pg} \neq 0$) in each replication, and then averaging across the 50,000 replications. For LM and TR, a variable is selected if the p-value of a variable is less than the Bonferroni-corrected threshold $0.05/(3 \times 74) = 2.3 \times 10^{-4}$, and for LASSO, a variable is selected if the lasso coefficient is not 0.

Table 3.1: Performance of detecting causal genomic variables of tensor regression (TR), linear model (LM), and LASSO with effect size $C = 0.25$, sample size $n = 500$ and $B$ with dimension $3 \times 74$ based on 50,000 replications. For TR and LM, a variable is selected as significant if its p-value is less than $\alpha = 0.05/(3 \times 74) = 0.00023$.

<table>
<thead>
<tr>
<th>Shape</th>
<th>Method</th>
<th>I</th>
<th>T</th>
<th>Random</th>
</tr>
</thead>
<tbody>
<tr>
<td>False Positive Rate</td>
<td>TR</td>
<td>0.00005</td>
<td>0.00021</td>
<td>0.00006</td>
</tr>
<tr>
<td></td>
<td>LM</td>
<td>0.00028</td>
<td>0.00028</td>
<td>0.00027</td>
</tr>
<tr>
<td></td>
<td>LASSO</td>
<td>0.11180</td>
<td>0.20504</td>
<td>0.17488</td>
</tr>
<tr>
<td>True Positive Rate</td>
<td>TR</td>
<td>0.99149</td>
<td>0.77192</td>
<td>0.99987</td>
</tr>
<tr>
<td></td>
<td>LM</td>
<td>0.55695</td>
<td>0.73417</td>
<td>0.68924</td>
</tr>
<tr>
<td></td>
<td>LASSO</td>
<td>0.70510</td>
<td>0.88540</td>
<td>0.86720</td>
</tr>
<tr>
<td>$g$ Measure</td>
<td>TR</td>
<td>0.99571</td>
<td>0.87850</td>
<td>0.99990</td>
</tr>
<tr>
<td></td>
<td>LM</td>
<td>0.79137</td>
<td>0.83896</td>
<td>0.84590</td>
</tr>
<tr>
<td></td>
<td>LASSO</td>
<td>0.74619</td>
<td>0.83008</td>
<td>0.85672</td>
</tr>
</tbody>
</table>

Table 3.1 summarize the results of FPR, TPR and $g$ measures. LM has the FPRs around the nominal level while TR has over-conservative FPRs, which is not unexpected due to the over-estimated variance. However, TR still has higher TPR than LM. Under all 3 shapes, LASSO has higher FPR than LM and TR and higher TPR than LM; LASSO has higher TPRs than TR under Shape T, and lower TPR than TR under Shape I and Random. Accounting for both FPR and TPR and judging based on $g$ measures, TR has the best overall performance, while LASSO and LM have similar overall performance.
We also examine the impact of the effect size $C$ on the method performance, with $C = 1$, 0.25 and 0.125. Table 3.2 shows the results for the 3 different shapes of $B$ based on 1000 replications. In these tables, a variable is selected as significant in TR and LM if the p-value is less than 0.05. We observe a very similar pattern as observed in Table 3.1 (i.e., TR has too conservative FPRs and the best overall performance), with the only exception under Shape T with the small effect size $C = 0.125$, under which LASSO has the best performance. We observe that when the effect size is small, TR does not have enough resolution to recognize the "T" shape and select a correct rank, which results in high FPRs. TR appears to be more robust to smaller effect size with Shape I and Random since the shape is simple and less resolution is needed. To understand this, imagine we have a circle shape in a more complicated case, the estimated shape will be more like square if sample size is not enough and wrong lower rank has been selected.

Finally, we consider the scenario of $P = 3$ and $G = 170$ with sample size $n = 500$, under which LM is infeasible due to the large number of parameters. Table 3.3 shows the performance of TR and LASSO with effect size $C = 1$ based on 1000 replications. The number of non-zero $B_{pg}$'s stays the same as Table 3.2. We see that TR and LASSO have comparable FPRs but the TPR of TR is higher than LASSO, resulting in a superior overall performance of TR. We note that in this case, the rank selection range for TR is 1 and 2 instead of from 1 to $P$, because larger number of genes makes $R = 3$ case infeasible.
Table 3.2: This table shows the simulation results for "I" under 1000 replications, 74 genes and 500 sample size. Bonferroni correction is not used.

<table>
<thead>
<tr>
<th>Measure (Shape : I)</th>
<th>Method</th>
<th>C = 1</th>
<th>C = 0.25</th>
<th>C = 0.125</th>
</tr>
</thead>
<tbody>
<tr>
<td>False Positive Rate</td>
<td>TR</td>
<td>0.04</td>
<td>0.03</td>
<td>0.03</td>
</tr>
<tr>
<td></td>
<td>LM</td>
<td>0.05</td>
<td>0.05</td>
<td>0.05</td>
</tr>
<tr>
<td></td>
<td>LASSO</td>
<td>0.12</td>
<td>0.09</td>
<td>0.08</td>
</tr>
<tr>
<td></td>
<td>TR</td>
<td>0.91</td>
<td>0.91</td>
<td>0.64</td>
</tr>
<tr>
<td>True Positive Rate</td>
<td>LM</td>
<td>0.80</td>
<td>0.51</td>
<td>0.25</td>
</tr>
<tr>
<td></td>
<td>LASSO</td>
<td>0.88</td>
<td>0.71</td>
<td>0.48</td>
</tr>
<tr>
<td></td>
<td>TR</td>
<td>0.93</td>
<td>0.94</td>
<td>0.79</td>
</tr>
<tr>
<td>g Measure</td>
<td>LM</td>
<td>0.87</td>
<td>0.70</td>
<td>0.49</td>
</tr>
<tr>
<td></td>
<td>LASSO</td>
<td>0.88</td>
<td>0.80</td>
<td>0.66</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Measure (Shape : T)</th>
<th>Method</th>
<th>C = 1</th>
<th>C = 0.25</th>
<th>C = 0.125</th>
</tr>
</thead>
<tbody>
<tr>
<td>False Positive Rate</td>
<td>TR</td>
<td>0.02</td>
<td>0.02</td>
<td>0.17</td>
</tr>
<tr>
<td></td>
<td>LM</td>
<td>0.05</td>
<td>0.05</td>
<td>0.05</td>
</tr>
<tr>
<td></td>
<td>LASSO</td>
<td>0.19</td>
<td>0.17</td>
<td>0.15</td>
</tr>
<tr>
<td></td>
<td>TR</td>
<td>0.78</td>
<td>0.76</td>
<td>0.68</td>
</tr>
<tr>
<td>True Positive Rate</td>
<td>LM</td>
<td>0.77</td>
<td>0.69</td>
<td>0.54</td>
</tr>
<tr>
<td></td>
<td>LASSO</td>
<td>0.91</td>
<td>0.88</td>
<td>0.78</td>
</tr>
<tr>
<td></td>
<td>TR</td>
<td>0.87</td>
<td>0.86</td>
<td>0.74</td>
</tr>
<tr>
<td>g Measure</td>
<td>LM</td>
<td>0.86</td>
<td>0.81</td>
<td>0.72</td>
</tr>
<tr>
<td></td>
<td>LASSO</td>
<td>0.86</td>
<td>0.85</td>
<td>0.81</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Measure (Shape : Random)</th>
<th>Method</th>
<th>C = 1</th>
<th>C = 0.25</th>
<th>C = 0.125</th>
</tr>
</thead>
<tbody>
<tr>
<td>False Positive Rate</td>
<td>TR</td>
<td>0.04</td>
<td>0.04</td>
<td>0.03</td>
</tr>
<tr>
<td></td>
<td>LM</td>
<td>0.05</td>
<td>0.05</td>
<td>0.05</td>
</tr>
<tr>
<td></td>
<td>LASSO</td>
<td>0.16</td>
<td>0.11</td>
<td>0.08</td>
</tr>
<tr>
<td></td>
<td>TR</td>
<td>1.00</td>
<td>0.99</td>
<td>0.62</td>
</tr>
<tr>
<td>True Positive Rate</td>
<td>LM</td>
<td>0.74</td>
<td>0.46</td>
<td>0.21</td>
</tr>
<tr>
<td></td>
<td>LASSO</td>
<td>0.88</td>
<td>0.67</td>
<td>0.44</td>
</tr>
<tr>
<td></td>
<td>TR</td>
<td>0.98</td>
<td>0.97</td>
<td>0.78</td>
</tr>
<tr>
<td>g Measure</td>
<td>LM</td>
<td>0.84</td>
<td>0.66</td>
<td>0.45</td>
</tr>
<tr>
<td></td>
<td>LASSO</td>
<td>0.86</td>
<td>0.77</td>
<td>0.64</td>
</tr>
</tbody>
</table>

TR: Tensor Regression; GLM: Generalized Linear Model.
Table 3.3: This table shows the simulation results for "I", "Random" and "T" under 1000 replications, 170 genes, 500 sample size and an appropriate effect size. Bonferroni correction is not used.

<table>
<thead>
<tr>
<th>Shape</th>
<th>Method</th>
<th>I</th>
<th>T</th>
<th>Random</th>
</tr>
</thead>
<tbody>
<tr>
<td>False Positive Rate</td>
<td>TR</td>
<td>0.01</td>
<td>0.02</td>
<td>0.01</td>
</tr>
<tr>
<td></td>
<td>LASSO</td>
<td>0.00</td>
<td>0.02</td>
<td>0.01</td>
</tr>
<tr>
<td>True Positive Rate</td>
<td>TR</td>
<td>1.00</td>
<td>1.00</td>
<td>0.98</td>
</tr>
<tr>
<td></td>
<td>LASSO</td>
<td>0.86</td>
<td>0.29</td>
<td>0.67</td>
</tr>
<tr>
<td>g Measure</td>
<td>TR</td>
<td>0.98</td>
<td>0.99</td>
<td>0.98</td>
</tr>
<tr>
<td></td>
<td>LASSO</td>
<td>0.93</td>
<td>0.53</td>
<td>0.81</td>
</tr>
</tbody>
</table>

TR: Tensor Regression.

3.5 Real Data Analysis

We perform pathway-based multi-platform analysis on the TCGA Uterine Corpus Endometrial Carcinoma (UCEC) dataset. UCEC (and EC in short) is one of the most common cancers of the female reproductive system among American women; different omics analysis has provided insights on the pathobiologic features and molecular biomarkers for UCEC (e.g., Chen, O’mara, Thompson, Painter, Attia, Black, Brinton, Chanock, Chen, Cheng, et al. (2016); Fan, Wang, Fu, Yang, Lin, Fan, and Wen (2018) ). In this analysis, we focus on the tumor protein p53 (TP53, a tumour suppressor gene) signaling pathway from Kyoto Encyclopedia of Genes and Genomes (KEGG), and evaluate the effects of the genomic variables in the TP53 pathway, including copy number variation (CNVs), DNA methylation and transcriptome profiling, on the tumor invasion percentage of EC. TP53 signaling pathway is exceedingly well studied in a variety of cancers. Mutated TP53 has been found predominantly in serous EC (89%) and serous ovarian (94.6%), suggesting the pivotal importance of TP53 in tumorigenesis (Kandoth, McLellan, Vandin, Ye, Niu, Lu, Xie, Zhang, McMichael, Wyczalkowski, et al. (2013)).

The data considered consist of 560 samples and can be downloaded directly from the National Cancer Institute Genomic Data Commons (GDC) Data Portal at https://portal.gdc.cancer.gov/projects/TCGA-UCEC. The CNVs, DNA methylation and transcripts data are first aligned
into the gene level. Then within each platform, the gene values across all genes are normalized with mean 0 and variance 1. We next retrieve the gene data that are involved in TP53 signaling pathway and available in all 3 platforms. For the outcome variable (tumor invasion percentage of EC), we remove 1 individual whose \( p_i > 100\% \). We perform a logit transformation by first computing \( p_i^* \), i.e., \( p_i^* = p_i + 10^{-5} \) if \( p_i = 0 \), \( p_i^* = p_i - 10^{-5} \) if \( p_i = 1 \), and \( p_i^* = p_i \) otherwise and then taking transformation to obtain \( y_i = \log\left(\frac{p_i^*}{1-p_i^*}\right) \). Variable is used as the response variable in the multiplatform analysis using tensor regression (TR), linear model (LM) and LASSO. The results are shown in Table 3.4.

### Table 3.4: KEGG_P53_SIGNALING_PATHWAY (R = 3)

<table>
<thead>
<tr>
<th>Methods</th>
<th>Platforms</th>
<th>External Gene Symbol</th>
</tr>
</thead>
</table>
| TR      | Methylation | CASP8, CCNB1, CCND1, CCND3  
                  |  
                  | CCNE1, CCNG1, CYCS, DDB2  
                  |  
                  | PPM1D, RRM2B, SESN1, TP53I3, TP73  
                  |  
                  | BBC3, BID, CASP3, CCND2, CCNE2  
                  |  
                  | CCNG2, CD82, CDK1, CDKN2A  
                  |  
                  | CYCS, GADD45G, MDM2, SERPINB5  
                  |  
                  | SESN1, SESN3, SIAH1, THBS1  
                  |  
                  | CCND2, CCNE2, CHEK1, DDB2  
| TRANS   | CNVs       | FAS, IGF1, PERP, SIAH1  
                  |  
                  |  
                  | TNFRSF10B, TP73  
| LASSO   | Methylation | -  
|        | CNVs       | CCND1, CD82  
|        | TRANS      | -  
| LM      | Methylation | CASP8, CCNB1, CCND1, CCND3  
                  |  
                  | CCNE1, CCNG1, CYCS, DDB2  
                  |  
                  | PPM1D, RRM2B, SESN1, TP53I3, TP73  
                  |  
                  | BBC3, BID, CASP3, CCND2, CCNE2  
                  |  
                  | CCNG2, CD82, CDK1, CDKN2A  
                  |  
                  | CYCS, GADD45G, MDM2, SERPINB5  
                  |  
                  | SESN1, SESN3, SIAH1, THBS1  
                  |  
                  | CCND2, CCNE2, CHEK1, DDB2  
| TRANS   | CNVs       | FAS, IGF1, PERP, SIAH1  
                  |  
                  |  
                  | TNFRSF10B, TP73  

TRANS: Transcriptome profiling
The TR model, with rank = 3 determined by AIC, identifies 17 genes at CNV level, 13 genes at DNA methylation level and 10 genes at the transcriptome level associated with EC tumor invasion percentage. Since TR rank is 3, we expect TR to have the same results as LM and this is what we observed. LASSO identifies two gene signals at the CNV level, i.e., \( CCND1 \) and \( CD82 \). These two genes are also identified by TR (and LM), with \( CCND1 \)'s signal from methylation and \( CD82 \)'s signal from CNV. \( CD82 \), known as a metastasis suppressor, inhibits cancer cell migration (Liu and Zhang (2006)). Loss of \( CD82 \) protein expression was observed in advanced and metastatic EC (Liu, Dong, Chen, Hsieh, Ho, Hung, Lu, and Chiou (2003)). The multi-platform analysis by TR, LM and LASSO suggest that loss of \( CD82 \) protein expression may be related to CNV level of variations. Additional investigations will be helpful to elucidate the underlying mechanisms. \( CCND1 \) has been reported overexpressed (protein level) in EC (Nikaido, Toshio, et al., 1996, Quddus, M. Ruhul, et al., 2002, Soslow, Robert A., et al., 2000, Nan, Fangfang, et al., 2009). In the multi-platform analysis, TR (and LM) find that methylation of \( CCND1 \) is associated with EC while LASSO finds that CNV in \( CCND1 \) is associated with EC. Further analysis will be helpful to understand which molecular events may be more relevant to the over-expressions of \( CCND1 \) protein in EC.

We group the signals identified by TR (and LM) into four types to facilitate the discussions of their biological relevance: (a) \( CYCS \) and \( SESN1 \) are identified for signals from CNV and DNA methylation; (b) \( DDB2 \) and \( TP73 \) have signals from DNA methylation and transcription; (c) \( CCND2 \), \( CCNE2 \) and \( SIAH1 \) have signals from CNV and transcription; and (d) the rest genes have signals from one single platform, either from CNV, methylation or gene expression.

For (a), frequent \( TP53 \) mutations were observed in EC (Levine, Network, et al. (2013)), and it has been reported that \( TP53 \) activates the expression of \( SESN1 \) in response to DNA damage (Nakano and Vousden (2001); Velasco-Miguel, Buckbinder, Jean, Gelbert, Talbott, Laidlaw, Seizinger, and Kley (1999)). In additional, \( TP53 \) also activates the expression of \( BBC3 \) (Nakano and Vousden (2001); Velasco-Miguel et al. (1999)). \( BBC3 \) binds to \( BCL2 \) and induces release of \( CYCS \) proteins, which lead to induction of apoptosis in endometrial cancer cell lines (Nakano
and Vousden (2001); Velasco-Miguel et al. (1999)). Thus, aberrant DNA methylation and copy number of CYCS and SESN1 might contribute to the tumorigenesis of EC through indifference to the activation induced by TP53. For (b), transcription of DDB2 is downregulated in UCEC with mutated TP53 (Parikh et al.[79]). TP73 is homologous to TP53 in terms of structure and functionality (Aqeilan, Pekarsky, Herrero, Palamarchuk, Letofsky, Druck, Trapasso, Han, Melino, Huebner, et al. (2004)). Thus, abnormal DNA methylation of DDB2 and TP73 might influence their expressions leading to failure of apoptosis. For (c), over-abundance of beta-catenin in nucleus promotes transcription of many oncogenes (Shang, Hua, and Hu (2017)), and SIAH1 is essential to degrade beta-catenin by a phosphorylation-independent way (Herbst and Kolligs (2007)). CCND2 has also been reported to contribute to tumorigenesis in a variety of cancer including ovarian cancer possibly via over-abundance of beta-catenin in nucleus. In addition, CCND2 and CCNE2 are related to cell cycle in the sense that cell cycle phase is controlled by cyclin-dependent kinase family whose activation depends on availability of cyclins (Murray (2004)). Deregulation in cell-cycle checkpoints is a common feature in many human cancers (Kastan and Bartek (2004); Malumbres and Barbacid (2009)). CCND2 has been reported to be involved in cell cycle control in tumors cells in female reproductive system such as ovarian germ cell tumors (Deshpande, Sicinski, and Hinds (2005)). CCNE2 transcripts have also been shown to be significantly lower in ovarian endometrioid adenocarcinoma (Steg, Wang, Blanquicett, Grunda, Eltoum, Wang, Buchsbaum, Vickers, Russo, Diasio, et al. (2006)). Thus, atypical copy number of CCNE2, CCND2 and SIAH1 may affect their expressions causing abnormalities in cell cycle control. Signals in group (d) also have direct or indirect supports regarding to their relevance with EC and we organize the information in Appendix B.5.

3.6 Discussion

we have described in detail the parameter-wise hypothesis testing (instead of global hypothesis testing) under the tensor regression framework for multiple types of platforms of data under the Gaussian model. We introduced the concept of box product, a permutated version of kronecker
product, into the variance formula derivation, whose merits have been explained in details. Moreover, we investigated the behavior of the test statistics of hypothesis testing under different effect sizes, shapes of the parameters matrix $B$ as well as different number of parameters. TR-based approaches have multiple benefits. First, TR-based approaches can take advantage of the latent net-work structure among genes in the same pathway even such network might cause serious multi-collinearity issue leading to an unstable estimate. Second, TR-based approaches are intuitive since they are essentially linear regressions but being transformed into higher order Tensors. Together with low rank assumption, TR-based approaches can reduce the number of parameters dramatically given the rank is small. Having realized so many benefits, there has been large amount of applications of TR-based approaches in all kinds of fields, i.e., neuro-imaging. Particularly in genomic study, there are many methods looking for the "significant" genes such as LASSO, iBAG and MTGDR. The term "significant" has double quotes is because its definition varies under different models. The selected genes are called "significant" in terms of variable selection procedure under LASSO framework while "significant" means the significance in terms of p-value under hypothesis testing settings. Except for the variable selection, the study of the hypothesis testing under the tensor regression framework is always needed since p-value is one of the most widely used decision making parameters and it is always appreciated and demanded that if tensor regression can be applied as easy as linear regression.

The data we discussed throughout the article focuses on only three platforms of data on a order-3 tensor design matrix. We still see the power gain in rank 1 and rank 2 situations in the simulation study as shown in Table 3.2. Even in rank 3 scenario, TR-based analysis will continue to be useful since it is just structural linear regression model as explained earlier. Cancer comics data has become "larger" and "larger". The "large" is in terms of two aspects. The sample size is growing, so is the number of platforms. At this moment in the genomic study, even though the number of platforms are still very few because of the limitations in technologies, one of the most important goal of this article is to conceptualize the potentials of TR-based approaches and reveal the behavior of the test statistics under the commonly used hypothesis test framework.
We believe the TR-based approaches will shine even more when we have deeper understanding of cancers since we can construct our TR-based models based on a much higher order and meaningful design matrix and more availability of the data.

Even the TR-based approaches can reduce the number of parameters, but still, its feasibility depends on the ratio of the sample size to the number of parameters since the internal algorithm optimizing its objective function is still an iterative linear regression procedure which has been explained in the estimation section. To see this, denote the number of parameters as $G$, $R$ is the rank of the parameter matrix $B$ and the sample size is $n$. LM is not going to work whenever $n < G$. For TR-based approaches with a low rank assumption, situations are different. The algorithm of solving a TR-based approaches involves solving two GLM problems internally, each of which requires only $PR$ and $GR$ parameters, respectively, which is much smaller than the required sample size from GLM. This is why the introduction of the TR-based approaches lower the requirement of the ratio of the sample size to the number of parameters. Thus, in order to make TR-based approaches feasible, the sample size required by TR-based approaches must satisfy $n \geq \max\{PR, GR\}$. In our case, $P = 3$, which is smaller than $G$. Hence the condition for TR becomes $n \geq GR$. The feasibility conditions of TR-based approaches coincide with the feasibility conditions of LM when $R = 3$. For example, suppose $R = 2$, the range that TR model works but GLM does not work needs $\frac{n}{3} < p_2 \leq \frac{n}{2}$. In our experiment, suppose $n = 500$, then $167 < G \leq 250$. If $G < 167$, both LM and TR-based approaches will work. Similarly, suppose $R = 1$, $n = 500$, then $167 < G \leq 500$. Suppose we chose $G = 170$. In such case, GLM does not work for "random" and "T" case. The maximum possible rank selection for TR-based approaches is two.

Every coin has two sides. TR-based approaches have many limitations as well. First of all, being transformed into a more structural format (higher order tensor), the numerical property such as convergence and stability become much more complicated, which we have already seen during the simulation procedure and binary case even harder to discuss than the situations in
Gaussian case. Many procedures have already been done to stabilize the estimations. In addition, low rank assumption transformed a convex problem into a non-convex problem which suffers non-global minima guaranteed issue. Second of all, the definition of rank of higher order tensor are unclear. The definition of rank in terms of a matrix is pretty straightforward but for higher order tensor, we mainly use the definition of rank from the CANDECOMP/PARAFAC(CP) decomposition. Third of all, it is suitable for follow-up analysis that is once a pathway-level significance is identified (as the rank of $\mathbf{B} = 0$ is undefined). Forth of all, how frequently and strong of the magnitude the biological mechanism will be reflected on the low-rank phenomenon remains unknown. At last, rank selection is the key for getting a properly controlled FPR. An inflated FPR will be observed if the wrong rank is selected but how serious the inflated FPR depends on many other factors as well, such as the magnitude of the smallest absolute value of the non-zero eigenvalues and the shape of the parameter matrix.

3.7 Future work

In this article, we only investigate the behavior of the test statistics under Gaussian type response variable. However, Gaussian type response variable is not as common as the binary type response variable. Binary type response variable is even more reasonable in the genomic study since we alway interested in the association analysis between the appearance of a certain phenotype ($y_i = 0$ or 1) of interest and the genetic variations of the $i$-th participant. As pointed out in the previous section, TR-based approaches introduce the numerical issue and such issue is even amplified when we are under the logistic TR-based framework. We did conduct the hypothesis testings using the derived variance formula under the different logit function, but the results are not very ideal. We also tried to use Gaussian TR-based frame work (latent variable) for the binary type response variable, but results are still not very good and an inflation of FTR has been observed, which is not acceptable. Even though, we will still provide the details how we derive the algorithm for binary response variable under the Gaussian TR-based framwork for future study.
"Larger" and "larger" data have been generated across all fields in life science. The term of "large" has two aspects: on one hand, the size of the data keeps growing, on the other hand, the number of platforms of data has been increasing as well. Such growth in both aspects not only leads to the demand for more comprehensive statistical techniques: Integrated genetic data analysis, but also a great challenge in both data analysis and data process. Integrated multi-platform genetic data analysis is a concept of analyzing the genomic data across different platforms instead of implementing analysis platform by platform or sample by sample. Such an integrative analysis has shown its advantage against the traditional analysis tools, i.e., (Wang et al. (2012); Ma and Huang (2009); Jiang et al. (2016)). Meanwhile, the data processing before any analysis procedures is no longer trivial since it is always problematic to align data across different studies measured at different levels. For example, large data need a more sophistical tool to pre-process the data. Even, different measure units need to be aligned into the same levels, e.g., the methylation data at CpG level and CNV data measured at gene levels. Careless process procedure can result in "biased biological knowledge."

The importance of processing data has been realized recently by many studies (i.e., Palsson and Zengler (2010)) which split the integrated multi-platform genomic data analysis into four steps: (1) Data generation; (2) Data processing; (3) Data integration; (4) Data Analysis. The
importance of data processing and integration are discussed in the previous paragraph, which is the reason the process steps are discussed in a separate chapter. In chapter 3, three platforms of data are integrated and analyzed: methylation data, CNVs data and transcripts data. In this chapter, we reveal the details how the data process is conducted in chapter 3 and form a pipeline using both R and Bash scripts to convert these data to the desired resolution efficiently and effectively.

Before the discussion of details in processing each platform of the data, we need a brief introduction to the sample ID used in TCGA dataset. TCGA dataset measurement is based on tissue samples so that multiple tissue samples can come from the same participant. But in the chapter 3, we are interested in participants as units of interest instead of tissue samples. Hence, before conducting any analysis, the original sample ID needs to be reworked from the bar-code level to participant level. For example, the original sample ID from TCGA datasets is

TCGA-PG-A6IB-01A-21D-A31V-05

it is truncated into

TCGA-PG-A6IB-01

The truncated part is the bar codes of different tissue samples on the same participant. The detailed interpretation is shown in Figure 4.1.

4.1 Methylation Data Processing Pipeline

DNA methylation is a biological process of chemically modifying a DNA sequence in the context of CG dinucleotide (CpG site). CpG sites are regions of DNA where a cytosine nucleotide is followed by a guanine nucleotide, which explains what "C" and "G" stand for. "p" means the
Figure 4.1: Interpretation of the sample ID
(https://docs.gdc.cancer.gov)

$PO_4^-$ that connects cytosine nucleotide and guanine nucleotide.

The methylation dataset used in chapter 3 is the Illumina Human Methylation 450K downloaded from TCGA data portal through *bioconductor*. The downloaded object named *dnamethy_UCEC.Rdata* is stored into a list format in R and its structure is shown in Figure 4.3.

Six variables are contained in *dnamethy_UCEC.Rdata*. (1) *query_ucec_dnamethy* stored the query information from TCGA data portal; (2) *methy.measure* is a matrix format that has 484577 rows and 485 columns. Each row has a corresponding CpG ID and each column has a sample ID; (3) *methy.patient.info* contains the bio information for each patient such as race, BMI and age; (4) *methy.compound.info* is a list including the additional information for each CpG such as location, strand and their external gene symbol (5) *methy.UCEC.sumy* is the RangedSummarizedExperiment class variable summarizing all the information; (6) *methy.UCEC* is the matrix format combining the *methy.measure* and *methy.compound.info*. *methy.UCEC* is the main list whose first ten entries store ad-
Figure 4.2: DNA methylation overview. Methylation occurs at CpG dinucleotides regulated by DNA methyltransferases (Dnmts). Dnmt1 is predominantly associated with methylation of hemimethylated CpG dinucleotides while Dnmt3a and Dnmt3b contribute to de novo methylation.

Figure 4.3: The Structure of data `dnamethy_UCEC.Rdata`
ditional information on the CpG sites. The remaining entries in `methy.UCEC` stores the methylation level labeled by the truncated sample ID, `TCGA-PG-A6IB-01`, for example. The interpretation of the sample ID has been shown in Figure 4.1. Except for `methy.UCEC`, the other important variable is `methy.compound.info`. One of its information is the external gene symbol. Hence, each entry of `methy.UCEC` has a corresponding position in the `methy.compound.info`.

In the tensor regression analysis of chapter 3, we are interested in the associations between a phenotype of interest and genetic variations under gene levels labeled by external gene symbols. However, the methylation data set is measured based on CpG sites, a more refined level than the gene level. Therefore, we need to convert the methylation data set from the CpG sites level to the gene level. Conversion from CpG sites level to gene level is not straightforward. Denote A, B, C, three external gene symbols, there are four possible external gene symbols for each CpG site (CpG ID): (1) A;A;A; (2) A;B;A;C (3) A;A;B;B;C;C. Given a CpG ID, if the corresponding external gene symbol is missing, which is the case (1), it means the CpG site is located in an intergenic region, a region between two genes. This is why the external gene symbol is missing. Thus, any methylation data that are not labeled with any external gene symbols are discarded. As for case (2), the number of replications reflects the number of possible versions of the gene A. Hence, we can just take the unique value when one CpG ID labeled with a unique but repeating external gene symbol as in case (2). Case (3) and (4) will be discussed together. As discussed in case (2), the number of replications indicate different versions. Moreover, different version numbers of a gene, A arise from the change of the set of transcripts linked to gene A (http://useast.ensembl.org/info/genome/stable_ids/index.html). What makes case (3) and (4) special is it contains many external gene symbols. This means a certain CpG ID can lie in an overlapping region of gene A, gene B, and gene C. One CpG entry should be expanded into three entries with same CpG ID but different external gene symbols.

The methylation data process contains two sections. The first section is called data cleaning process. The first section converts the TCGA methylation data set from CpG level to gene level
Figure 4.4: First section of methylation data processing: data cleaning process. Green rectangular represents intermediate files generated throughout the first section. Light blue rectangular represents the bash script files implementing each sub step. Detailed bash scripts will be provided in the Appendix C.
under certain rules. Those rules can be summarized as (1) records with missing external gene symbols will be omitted. (2) records under same external gene symbols will be averaged. The whole data process steps is shown in Figure 4.4. The results are twenty-two files stored by the chromosome information.

The second section of the methylation dataset processing pipeline is called splitting and selection. It splits the data according to the tissue sample type and then selects genes within the certain pathway of interest. The detailed procedure is described in Figure 4.5. The resulting

![Diagram](image)

**Figure 4.5:** Second section of methylation data processing: splitting and selection. Green rectangular represents intermediate files generated throughout the first section. Yellow rectangular represents the bash script files implementing each sub step. Detailed bash scripts will be provided in the Appendix C.
methylation data is at gene level and all of the genes belong to the same pathway.

4.2 Transcriptome Profiling Data Processing Pipeline

Transcription is the first step of the gene expression, in which a strand of DNA sequence is copied into a new complementary RNA sequence called transcript by transcription factors and enzyme RNA polymerase. A total of 62.1% and 74.7% of the human genome (Djebali, Davis, Merkel, Dobin, Lassmann, Mortazavi, Tanzer, Lagarde, Lin, Schlesinger, et al., 2012) get involved into transcription. The transcripts, as the resulting RNA of the transcription process, can be either coding or non-coding (ncRNA). Coding RNA refers to a RNA molecule which is translated into a protein and serves as templates during translation while the ncRNA,

Figure 4.6: “Central dogma” of molecular biology describes two steps: (1) Transcription: a segment of DNA is transcribed into either coding RNA or non-coding RNA ;(2) Translation: coding RNA is further translated into polypeptide then protein.
many non-ribosomal, non-mitochondrial RNAs, refers to a RNA molecule that is not translated into any protein. ncRNA is as important as coding RNA. It has a variety of functions including forming ribosome subunits (rRNA), carrying amino acids (tRNA), and regulation of gene expression. Besides, ncRNAs is involved in many human diseases including Alzheimer’s disease, Schizophrenia, Cardiovascular disease, cancer, and so forth (Wahlestedt, 2013). The transcription process with the translation process has been shown in Figure 4.6. The transcript data used in Chapter 3 is downloaded from TCGA data portal through bioconductor named transcripts_UCEC.Rdata. And its structure is in Figure 4.7. Six variables are contained in transcripts_UCEC.Rdata. (1) query.ucec.transcripts stored the query information from TCGA data portal; (2) trans.measure is a matrix format that has 56963 rows and 587 columns. Each row has a corresponding ensemble ID and each column has a sample ID; (3) trans.patient.info contains the bio information for each patient such as race, BMI and age; (4) trans.compound.info is a list including the additional information original ensemble gene ID, external gene ID, location and strand. (5) trans.UCEC.sumy is the RangedSummarizedExperiment class variable summarizing all the information; (6) trans.UCEC is the matrix format combining the trans.measure and trans.compound.info. Similar to methylation data set, trans.UCEC is the main list. The first entry stores external ensemble ID with the versioning number. The remaining entries in trans.UCEC stores the gene expression level by the sample ID. The interpretation of the sample ID has been truncated into and the reason has been discussed in the previous paragraph. Besides trans.UCEC, the other important variable is trans.compound.info, one of whose information is the external gene symbol. The
Figure 4.8: Transcripts data set process procedure. Blue rectangular represents the file that processes the whole process in the dashed border rectangular. Green rectangular represents the files that are needed or produced during or after the whole data process procedure. The detailed R script are given in the Appendix C.
length of external gene symbol entry in the `trans.compound.info` is different from each entry of `trans.UCEC`. This is because some ensemble gene ID has no corresponding external gene symbols given the version of the downloaded data. Thus, for convenience purpose, those entries are removed even though there exist corresponding external gene symbol in a newer version of the data. So as to map the transcript data to external gene symbol level, we will remove the versioning number and take the average of those entries whose external ensemble gene ID and sample ID is the same. The whole procedure of processing transcripts data is summarized into Figure 4.8. The resulting transcripts data is labeled with external gene symbol under the same pathway.
4.3 CNV Data Processing Pipeline

Genetic variation refers to the differences in the DNA sequence among individuals. There are two types of genetic variations: single nucleotide polymorphisms and structural genetic variation. Copy number variations (CNVs) refers one of the four structural genetic variations (inversion, deletion, insertion and CNVs shown in Figure 4.9). It is the phenomenon that the copy number of a particular DNA sequence is different among individuals.

![Diagram of structural genomic variants](image)

Figure 4.9: Changes in order and number of genes A, B, C and D vary the human genome, which makes us different. The right bottom is the copy number variations (CNVs)

| CNV_UCEC.Rdata | The CNVs dataset downloaded from the TCGA (processed already). After some preprocess procedures, we have two files first: (1) CNV; (2) range.clean. All the cleaning steps in this documentation start from these two files. The detailed contents of these files are posted in Appendix C. CNV.txt is a table that contains six variables: (1) sample ID; (2) chromosome; (3) start; (4) end; (5) Num Probes; (6) Segment Mean. range.clean.txt is a table with four variables: (1) start; (2) end; (3) Ensemble gene ID; (4) Gene symbol (external gene symbol). "Sample ID" is participant based. "Chromosome", "start" and "end" are the location information regarding certain CNV. "Num Probes" is the number of probes in
the region of DNA sequence that has CNVs. Probe is a DNA or RNA fragment used to search for a particular gene or other DNA sequence. This is how we detect whether we have different copies of certain gene. "Segment mean" is $\log_2(x/2)$ transformed raw data of CNVs data entries. The raw data of CNVs data set is 0, 1 or 2.

The goal of the CNV data cleaning pipeline is to convert the CNVs data set from the probe level to the gene level following the two tables above. The most important step for converting CNVs data is to match external gene symbol to the original CNV data by the locations. The raw CNVs data is log$_2$ transformed. In our matching procedure, for now, we will have to choose a $d$ value such that only those data that bigger than $\log_2((2 + d)/2)$ or smaller than $\log_2((2 - d)/2)$ will be kept. Besides, it also requires that the probe density larger than 0.001, where probe density is the ratio of the number of probes to the length of the certain gene. The overall CNV processing procedure has been shown in Figure 4.10:
Figure 4.10: CNVs data processing and splitting procedure

The diagram illustrates the steps involved in processing and splitting CNVs (Copy Number Variations) data. The process starts with loading CNV data from a file, `CNV.txt`, and then splitting this data into 22 files according to chromosome information.

Each chromosome file is then processed with a script named `CNV_clean_step1.sh`, which includes a data snapshot containing chromosome and gene information. This step is followed by merging files according to gene symbol.

Subsequent steps involve further processing and splitting, culminating in a gene list in a pathway.

The diagram is a flowchart detailing the computational steps involved in the CNVs data processing pipeline.
REFERENCES


Xie, J., Ma, A., Fennell, A., Ma, Q., and Zhao, J. (2018), “It is time to apply biclustering: a comprehensive review of biclustering applications in biological and biomedical data,” *Briefings in bioinformatics*.


Appendix A

Robust Biclustering: A method based on the Minimum Distance Estimator

A.1 Proofs

In this section we give proofs of the propositions and theorems within the paper.

Proposition A.1.1 Given fixed $\tau$, the ISE part of the LEBRA error criterion has a bounded hessian matrix.

Proof: The ISE part of the LEBRA error criterion is a sum of functions with form $-e^{-x^2}$. Hence we only need to prove $-e^{-x^2}$ has a bounded second derivative. We will prove this by the simple algebra below:

$$\nabla^2 \left( -e^{-x^2} \right) = 2e^{-x^2}(1 - x^2) \leq 2e^{-x^2} \leq 2$$

Therefore, the Hessian of $f(U)$ is upper bounded.

Proposition A.1.2 Given fixed $\tau$, a majorant of $F_\lambda$, $G_\lambda : \mathbb{R}^{n \times p} \to \mathbb{R}$, has the form:

$$G_\lambda(U|\tilde{U}) = \frac{\sqrt{2\pi}L}{np\sqrt{\pi}} \left( \frac{1}{2}||U - X^*||_F^2 + \gamma(\lambda)\mathcal{J}(U) \right) + \tilde{C} \quad (A.1)$$
where $\tilde{C}$ is a constant that is not related to either $U$ or $\tau$,

\[
\gamma(\lambda) = \frac{np\sqrt{\pi}}{\tau\sqrt{2L}} \lambda \\
X^* = \tilde{U} + \tilde{K} \\
\tilde{K} = \left\{ \frac{\tau^2}{L} (x_{ij} - \tilde{u}_{ij}) e^{-\frac{\tau^2(x_{ij} - \tilde{u}_{ij})^2}{2}} \right\}_{(i,j)} \in \mathbb{R}^{n \times p}
\]

**Proof:** Let’s denote $f(U) = \psi(X - U, \tau)$ whenever we are only interested in $U$. Recall the LEBRA error criterion, we need to find the upper bound on its second derivative with respect to $U$ so that we can apply theorem above to find the quadratic majorant on the term $-e^{-\frac{\tau^2(x_{ij} - \tilde{u}_{ij})^2}{2}}$. Then the majorant is supposed to be:

\[
F_\lambda(U) = f(U) + \lambda J(U) \\
= \frac{\tau}{2\sqrt{\pi}} - \frac{\tau}{np} \sqrt{\frac{2}{\pi}} \sum_{i,j} e^{-\frac{\tau^2(x_{ij} - \tilde{u}_{ij})^2}{2}} + \lambda J(U) \\
\leq \frac{\tau}{np} \sqrt{\frac{2}{\pi}} \sum_{i,j} \left( -e^{-\frac{\tau^2(x_{ij} - \tilde{u}_{ij})^2}{2}} - \tau^2(x_{ij} - \tilde{u}_{ij}) e^{-\frac{\tau^2(x_{ij} - \tilde{u}_{ij})^2}{2}} (u_{ij} - \tilde{u}_{ij}) + \frac{L}{2} (u_{ij} - \tilde{u}_{ij})^2 \right) + \lambda J(U) \\
= G_\lambda(U|\tilde{U})
\]

where $\tilde{U}$ is a constant, $G_\lambda(U|\tilde{U}) : \mathbb{R}^{n \times p} \rightarrow \mathbb{R}$ and $G_\lambda(\tilde{U}|\tilde{U}) = F_\lambda(\tilde{U})$ if only if $U = \tilde{U}$. Solving $G_\lambda(U|\tilde{U})$ is actually a COBRA problem Chi et al. (2017). We can write $G_\lambda(U|\tilde{U})$ into COBRA problem by completing the square:
\[ G_\lambda(U|U) = \frac{\tau L}{np\sqrt{2\pi}} \sum_{i,j} \left( u_{ij} - \tilde{u}_{ij} - \frac{\tau^2}{L} (x_{ij} - \tilde{x}_{ij}) e^{-\frac{\tau^2(x_{ij} - \tilde{x}_{ij})^2}{2}} \right)^2 \\
+ \lambda J(U) + \tilde{C} \]

\[ = \frac{\tau L}{np\sqrt{2\pi}} \| U - (\tilde{U} + \tilde{K}) \|_F^2 + \lambda J(U) + \tilde{C} \tag{A.5} \]

\[ = \frac{\sqrt{2}\tau L}{np\sqrt{\pi}} \left( \frac{1}{2} \| U - (\tilde{U} + \tilde{K}) \|_F^2 + \frac{np\sqrt{\pi}}{\tau \sqrt{2L}} \lambda J(U) \right) + \tilde{C} \]

\[ = \frac{\sqrt{2}\tau L}{np\sqrt{\pi}} \left( \frac{1}{2} \| U - X^* \|_F^2 + \gamma(\lambda) J(U) \right) + \tilde{C} \tag{A.6} \]

**Theorem 2** LEBRA Convergence Theorem: Given an algorithm on \( \Xi \), \( \theta(0) = (\tau(0), U(0)) \in \Xi \) and LEBRA mapping \( \phi \), assuming a bounded sequence \( \{\theta(k)\}_{k=1}^\infty \) is generated which satisfies

\[ \theta(k+1) = \phi(\theta(k)) \tag{A.7} \]

Then either the LEBRA stops at the point where a solution is identified or there exists such a \( k \) so that for all \( k + j (j \geq 1) \) there is a convergent subsequence of \( \{\theta(i_k)\}_{k=0}^\infty \) whose limit is the fixed point of LEBRA mapping, \( \phi \).

**Proof:** The LEBRA convergence theorem results from a direct application of the generalization of Zangwill (1969). This proof is just to prove three required conditions (1) Compactness (2) Decreasingness (3) Continuity. Since both \( \{(U(i), \tau(i))\}_{i \in \{1, \ldots, n\}} \) are bounded, then we are able to find a compact set \( S \) such that \( \{(U(i), \tau(i))\}_{i \in \{1, \ldots, n\}} \subset S \). Then the compact condition of the Zangwill’s Convergence Theorem is automatically satisfied. Updating \( U \) given \( \tau \) is a MM algorithm whose majorant is constructed by quadratic upper bound theorem and the non-increasing property of the MM algorithm is a known fact. Moreover, we update \( \tau \) using Golden Search Algorithm (GSA) to look for the stationary point of the first order derivative with respect to \( \tau \), which is also non-increasing. Now we need to prove such \( \tau \) always exists: indeed, since the penalty part is a function of \( U \), so we may omit the penalty
term while working only with \( \tau \). It is not hard to show that

\[
\lim_{\tau \to +\infty} \frac{\tau}{2\sqrt{\pi}} \frac{\sqrt{\frac{2}{\pi}}}{n} \sum_{i,j} e^{-\frac{1}{2} \tau^2 (x_{ij} - u_{ij})^2} = +\infty
\]

(A.8)

\[
\lim_{\tau \to 0} \frac{\tau}{2\sqrt{\pi}} \frac{\sqrt{\frac{2}{\pi}}}{n} \sum_{i,j} e^{-\frac{1}{2} \tau^2 (x_{ij} - u_{ij})^2} = 0
\]

(A.9)

The remaining work is to prove for any \( U \), we can always find a \( 0 < \tau < \infty \) that makes

\[
\frac{\tau}{2\sqrt{\pi}} \frac{\sqrt{\frac{2}{\pi}}}{n} \sum_{i,j} e^{-\frac{1}{2} \tau^2 (x_{ij} - u_{ij})^2} \leq 0
\]

(A.10)

After some algebra, (12) can be written as \( 1 \leq \frac{2\sqrt{2}}{np} \sum_{i,j} e^{-\frac{1}{2} \tau^2 (r_{ij})^2} \). Then we have \( 2\sqrt{2} e^{-\frac{1}{2} \tau^2 r^2} \leq \frac{2\sqrt{2}}{np} \sum_{i,j} e^{-\frac{1}{2} \tau^2 (r_{ij})^2} \) where \( r = \text{max}(r_{ij}) \). In order to check whether such \( \tau \) exist, we need to set \( 2\sqrt{2} e^{-\frac{1}{2} \tau^2 r^2} \geq 1 \). After some algebra, we have \( \tau^2 \in (0, \frac{\ln 8}{r^2}] \), then \( \tau \in (0, \sqrt{\frac{\ln 8}{|r|}}) \) given \( \tau \in (0, \infty) \).

The bound upper bound for \( \tau \) here is very conservative.

Therefore, the LEBRA mapping \( \phi \), as a composition of two non-increasing mappings, is also a non-increasing mapping, i.e., the decreasing property of Zangwill’s theorem is satisfied.

Now, we prove the closeness of LEBRA mapping. Since the majorant is strictly convex as a function of \( U \), the update set for each iteration with respect to \( U \) is a singleton. Then we can conclude that LEBRA mapping, \( \phi \), is a point to point mapping. Therefore, we only need to prove \( \phi \) is a continuous mapping. LEBRA mapping for updating \( U \) may be written as the minimizer of a proximal operator: \( \text{prox}_{\lambda J}((X \odot Q + U \odot (11^T - Q))) \). LEBRA mapping for updating \( U \) can be seen as a composition of a proximal mapping and a continuous mapping. It is known that the proximal mapping is continuous and \( Q \) is a continuous as a function of \( U \) as well. So, LEBRA mapping for updating \( U \) is continuous. As for \( \tau \), the LEBRA mapping for updating \( \tau \) is continuous since the derivative of LEBRA criterion with respect to \( \tau \) is continuous.

Therefore, The LEBRA mapping is a continuous mapping. The closeness condition of Zangwill’s
Convergence Theorem has been shown.

**Proposition A.1.3** The fixed point of LEBRA mapping as a function of $\mathbf{U} \neq \mathbf{0}$ coincides with the stationary point of the LEBRA criterion as a function of $\mathbf{U} \neq \mathbf{0}$.

**Proof:** We are going to discuss $\tau$ and $\mathbf{U}$ separately. $\tau$ part is self-proved this is because we are using the stationary point for updating $\tau$. Therefore, the proof mainly regards $\mathbf{U}$. Denote $\tilde{\mathbf{U}}$ a fixed point of LEBRA mapping. Recall when we are deriving (A.6), we have intermediate result (A.5). For the convenient purpose, we will start with (A.5). Denote

$$g(\mathbf{U}|\tilde{\mathbf{U}}) = \frac{\tau L}{np\sqrt{2\pi}}||\mathbf{U} - (\tilde{\mathbf{U}} + \tilde{\mathbf{K}})||_F^2$$ (A.11)

Then we have an alternative form for the majorant $G$:

$$G(\mathbf{U}|\tilde{\mathbf{U}}) = g(\mathbf{U}|\tilde{\mathbf{U}}) + \lambda J(\mathbf{U}) + \tilde{C}$$ (A.12)

Notice here $G$ here is no subscript. This is because (A.5) appears before the definition of $\gamma(\lambda)$. So the subscript of $G$ has been omitted on purpose. Since $G$ is a convex function, then an equivalent condition for $\tilde{\mathbf{U}}$ to be the fixed point of the LEBRA mapping with respect to $\mathbf{U}$ is:

$$0 \in \partial G(\tilde{\mathbf{U}}|\tilde{\mathbf{U}})$$
$$0 \in \partial \left( g(\mathbf{U}|\tilde{\mathbf{U}}) + \lambda \partial J(\tilde{\mathbf{U}}) \right) (\tilde{\mathbf{U}})$$
$$0 \in \partial g(\tilde{\mathbf{U}}|\tilde{\mathbf{U}}) + \lambda \partial J(\tilde{\mathbf{U}})$$ (A.13)
$$0 \in \{ \nabla g(\tilde{\mathbf{U}}|\tilde{\mathbf{U}}) \} + \lambda \partial J(\tilde{\mathbf{U}})$$ (A.14)

Recall the construction of $G(\mathbf{U}|\tilde{\mathbf{U}})$,

$$G(\mathbf{U}|\tilde{\mathbf{U}}) = f(\tilde{\mathbf{U}}) + \langle \nabla f(\tilde{\mathbf{U}})|\mathbf{U} - \tilde{\mathbf{U}} \rangle + \frac{L^2}{2}||\mathbf{U} - \tilde{\mathbf{U}}||^2 + \lambda J(\mathbf{U})$$
Taking sub-differential on both side of the above equation, we have

\[ \partial G(U|\tilde{U}) = \{ \nabla f(\tilde{U}) + L^2(U - \tilde{U}) \} \oplus \lambda \partial J(U) \quad (A.15) \]

Combined (A.15) with (A.13), we end up having:

\[ \{ \nabla g(U|\tilde{U}) \} \oplus \lambda \partial J(U) = \{ \nabla f(\tilde{U}) + L^2(U - \tilde{U}) \} \oplus \lambda \partial J(U) \]

Let \( U = \tilde{U} \), we have:

\[ \{ \nabla g(\tilde{U}|\tilde{U}) \} \oplus \lambda \partial J(\tilde{U}) = \{ \nabla f(\tilde{U}) \} \oplus \lambda \partial J(\tilde{U}) \quad (A.16) \]

To sum up:

\[
\begin{align*}
0 & \in \partial G(\tilde{U}|\tilde{U}) \\
0 & \in \{ \nabla g(\tilde{U}|\tilde{U}) \} \oplus \lambda \partial J(\tilde{U}) \\
0 & \in \{ \nabla f(\tilde{U}) \} \oplus \lambda \partial J(\tilde{U}) \\
0 & \in \partial^o F_\lambda(\tilde{U})
\end{align*} \quad (A.17)
\]

where \( \partial^o \) denotes the Clarke subdifferential operator. \( \partial^o F_\lambda(\tilde{U}) \) is a singleton when \( \tilde{U} \neq 0 \).

**Proposition A.1.4** For any direction \( V \), the directional derivative of the LEBRA error criterion at the fixed point of the LEBRA mapping is non-negative.

**Proof:** We only need to prove \( U \) part. Since the summation \( G = g + \lambda J + \tilde{C} \) is proper and convex, then for any direction \( V \in \mathbb{R}^{n \times p} \), the directional derivative of \( G'(U; V) \) is well-defined at the stationary point \( \tilde{U} \). Thus we have

\[
\begin{align*}
G'(\tilde{U}; V) &= \langle \nabla g(\tilde{U}|\tilde{U})|V \rangle + \lambda J'(\tilde{U}; V) \\
&= \langle \nabla f(\tilde{U})|V \rangle + \lambda J'(\tilde{U}; V)
\end{align*} \quad (A.18)
\]
where $\tilde{U}$ is a stationary point of LEBRA algorithm. (A.19) is because of (A.16). Recall the relationship between the directional derivative and the sub-differential of a convex function, we have:

$$J'(\tilde{U}|V) \geq -\frac{\langle \nabla f(\tilde{U})|V \rangle}{\lambda}$$

After some algebra, we get:

$$0 \leq \lambda J'(\tilde{U}; V) + \langle \nabla f(\tilde{U})|V \rangle = F'_\lambda(\tilde{U}|V) \quad (A.20)$$

for a stationary point $\tilde{U}$ at any direction $V$
A.2 Additional Results

Figure A.1: Additional Measures for Figure 2.2. Row Size: 100, 30, 30, 40; Column size: 50 60 50 40. The number of outliers = 0. Other five measures show the similar results as the four measures we reported in the paper.
Figure A.2: Additional Measures for Figure 2.3. Row Size: 10 100 50 40; Column size: 10 100 50 40. The number of outliers = 0. Other five measures show the similar results as the four measures we reported in the paper.
Figure A.3: Additional Measures for Figure 2.5. Row Size: 10, 100, 50, 40; Column size: 10, 100, 50, 40. The number of outliers: 4000. We can see JI has the similar problem as VI, which has been discussed.
Figure A.4: Additional Measures for Figure 2.4. Row Size: 100, 30, 30, 40; Column size: 60, 60, 60, 50. The number of outliers: 4000. We can see JI has the similar problem as VI, which has been discussed.
Figure A.5: Additional Measures for Figure 2.7. JI behaves similar to VI and the reason has been discussed. Other additional measures exhibit similar results to the four we reported.
Figure A.6: Additional Measures for Figure 2.9. Other five measures show the similar results as the four measures we reported in the paper.
A.3 MEASURES OF CLUSTERING SIMILARITY

In this section, we will provide a detailed description of each clustering measure we have already used. They all belong to external criteria of clustering quality.

Rand Index: Consider a set of \( n \) elements \( X = \{x_1, x_2, \ldots, x_n\} \) and two clustering results of \( X \) compare. Suppose one clustering result is \( U = \{U_1, \ldots, U_r\} \), which clustered \( X \) into \( r \) clusters. The other is \( V = \{V_1, \ldots, V_s\} \), which clustered \( X \) into \( s \) clusters. The Rand Index (Rand, 1971) defines the similarity of two clustering result \( U \) and \( V \) are to each other by computing the “agreements” and “disagreements” of them. Let us define the following: \( a \), the number of pairs of elements in \( X \) that are in the same subset in \( U \) and in the same subset in \( V \); \( b \), the number of pairs of elements in \( X \) that are in different subsets in \( U \) and in different subsets in \( V \); \( c \), the number of pairs of elements in \( X \) that are in the same subset in \( U \) and in different subsets in \( V \); \( d \), the number of pairs of elements in \( X \) that are in different subsets in \( U \) and in the same subset in \( V \).

The Rand Index is:

\[
RI = \frac{a + b}{a + b + c + d} = \frac{a + b}{\binom{n}{2}}
\]  

(A.21)

The sum \( a + b \) can be interpreted as the agreement of two clustering results, \( U \) and \( V \) and the sum \( c + d \) can be interpreted as the disagreement of them. Rand Index tells us the probability two clusterings results will agree on a randomly chosen pair.

Adjusted Rand Index (ARI) Hubert and Arabie (1985): is the corrected-for-chance version of the Rand Index. The purpose of Adjusted Rand Index is to scale it in an interpretable way. 0 is “as well as random”, less than 0 is worse, and close to 1 is good. One of the problem with the Rand Index is that a completely random result under certain scenario can achieve a higher score than a clustering result from a certain algorithm.

It might be easier to understand from a contingency table:
Then, the Adjusted Rand Index is:

\[
ARI = \frac{\sum_{ij} C_{n_{ij}}^2 - [\sum_i C_{a_i}^2 \sum_j C_{b_j}^2]/C_n^2}{\frac{1}{2} [\sum_i C_{a_i}^2 + \sum_j C_{b_j}^2] - [\sum_i C_{a_i}^2 \sum_j C_{b_j}^2]/C_n^2}
\]  \hfill (A.22)

where \(C_{n_{ij}}^j\) is the combination of choosing \(j\) elements from \(n\) elements.

**Entropy:** The definition of the total entropy of a clustering result \(U\) on \(X\) is:

\[
H(U) = \sum_{i=1}^r \frac{|U_i|}{n} H(U_i)
\]  \hfill (A.23)

where \(|\bullet|\) computes the cardinality of a given set and \(H(U_i)\) is the entropy of the \(i\)th cluster. The definition of the entropy for the \(i\)th cluster, \(H(U_i)\), is given below:

\[
H(U_i) = -\sum_{U_i} \frac{|U_i|}{n_i} \log_2 \frac{|U_i|}{n_i}
\]  \hfill (A.24)

where \(n_i\) is the count of points in the \(i\)th cluster (truth).

**Variation of Information (VI) and Normalized Variation of Information (NVI)**

Meilă (2007) is a information based measure of clustering quality and closely related to mutual information as well. Following the conventions defined when we are describing the concept of Rand Index,

\[
VI(U, V) = H(U) + H(V) - 2I(U, V)
\]  \hfill (A.25)

\[
= [H(U) - I(U, V)] + [H(V) - I(U, V)]
\]  \hfill (A.26)
Informally, we can understand variation of information as the distance between the total information possessed by two clustering estimations and the shared information by those two. It is positive and bounded by the logarithm of the size of a dataset. If they totally agree with each other, the shared information would be same as the information brought by either one. Then the distance would be zero, i.e. $VI = 0$. We can expect a large value for two complete different clustering estimations. We can also write the $VI$ into a summation of two differences. The first term corresponds to the amount of information about $U$ that we lose while the other term corresponds to the amount of information about $V$ we still have to gain when we are going from $U$ to $V$. Unlike other measures, $VI$ is a metric. As a result, we are able to have the definition of “small" by using $VI$. Normalized Variation of Information is a normalized version of $VI$. Regular $VI$ can provide meaningful comparison between two clustering results from a same dataset but it will not be very helpful when it comes to different datasets. NVI makes two clustering estimations from different datasets comparable by rescaling $VI$ to an interval $[0, 1]$.

**Purity** is a simple and transparent evaluation measure. It is a measure of the extent to which clusters contain a single class. Its calculation can be thought of as follows: For each cluster, count the number of data points from the most common class in said cluster. Now take the sum over all clusters and divide by the total number of data points.

Given two sets of cluster estimations, $U$ and $V$, Purity can be defined as follows:

$$
\frac{1}{N} \sum_{V_i \in V} \max_{U_j \in U} |U_j \cap V_i|
$$

(A.27)

The sum and max operations are over all possible $i$ and $j$. Note that this measure doesn’t penalize the number of clusters. So for example, a purity score of 1 is possible by putting each data point in its own cluster.

**Jaccard Index** Ben-Hur, Elisseeff, and Guyon (2001) is another measure to evaluate the similarity between two cluster estimations. It takes value between 0 and 1. 0 means two cluster estimations has no agreements at all while 1 means they perfectly agree with each other.
The formation definition of Jaccard Index is given below:

Given two cluster estimations, \( U \) and \( V \)

\[
J(U, V) = \frac{|U \cap V|}{|U \cup V|}
\]  
(A.28)

This is simply the number unique elements common to both sets is divided by the total number of unique elements in both \( U \) and \( V \). Notice Jaccard Index is bounded below by zero but has no upper bound. It is very similar to the definition of Rand Index, however it disregards the pairs of elements that are in different clusters for both clusterings.

**Mirkin Metric** is also known as Equivalent Mismatch Distance. It is a variation of the Rand Index and defined by

\[
M(U, V) = \sum_{i=1}^{r} |U_i|^2 + \sum_{j=1}^{s} |V_j|^2 - \sum_{i,j} |U_i \cap V_j|^2
\]  
(A.29)

Like the Variation of Information, Mirkin metric is a metric as well. This metric is closely related to the Hamming distance between certain binary vector representations of each partition Mirkin, Mucic, Storhoff, and Letsinger (1996). Recall the Hamming distance of two vectors of same length is the number of positions that two elements from two vectors are different.

**Normalized Mutual Information** is an information theory based non-negative measure as well. If two clustering estimations agree with each other, the shared information would be the same as the information carried by themselves then Normalized Mutual Information is one while Normalized Mutual Information is zero if they are totally against each other, i.e. no shared information. Thus Normalized Mutual Information is between 0 and 1. The idea of Normalized Mutual Information is very similar to the idea of Normalized Variation of Information, the only difference is they used different way to express “difference”. Normalized Mutual Information uses ratio while Normalized Variation of Information uses minus. Since the NMI is normalized, we can use it to compare different number of clusters as well and eliminate the problem that entropy tends to increase the number of clusters. The detailed discussion about these measures
will be given in the appendix. The formal definition is given below:

Given two cluster estimations, \( U \) and \( V \), the Normalized Mutual Information is computed by:

\[
NMI(U, V) = \frac{2I(U, V)}{H(U) + H(V)}
\]  
(A.30)

The definitions of function \( I \) and \( H \) are the same as the previous section.
Appendix B

Hypothesis Testing for Tensor Regression with Application to Gene-set Integrative Analysis of Multi-omics Data

B.1 Background on Tensor, Kronecker product and Box product

In this section, we will give a brief introduction to tensor since it plays an important role in the TR-based model. Notations that are mentioned here will be used throughout the article. For simplicity, we restrict our discussions only on the related concepts of tensor and operations for order-3 or lower dimensional tensor. Complete discussion regarding tensor are available in other materials. Tensor is generalized concept of the matrix and vector. 1D tensor is called vector; 2D tensor is called matrix shown in Figure B.1. When the dimension is over three, we do not have specific name like matrix and vector do.

Kronecker product, denoted by $\otimes$, is a generalization of the outer product; it is an operator on two matrices of arbitrary size and yields a block matrix. For $A \in \mathbb{R}^{m_1 \times n_1}$ and $B \in \mathbb{R}^{m_2 \times n_2}$,
their Kronecker product is a $m_1m_2 \times n_1n_2$ block matrix:

$$A \otimes B = \begin{bmatrix} a_{11}B & \cdots & a_{1n_1}B \\ \vdots & \ddots & \vdots \\ a_{m_1}B & \cdots & a_{m_1n_1}B \end{bmatrix}. $$

For $A \in \mathbb{R}^{m_1 \times n_1}$ with entries $A_{ij}$ and $B \in \mathbb{R}^{m_2 \times n_2}$ with entries $B_{\ell k}$, their box product, the permutated kronecker product, denoted by $C = A \boxtimes B$, is a $m_1m_2 \times n_1n_2$ matrix with its entries:

$$C_{(i-1)m_2 + \ell, (k-1)n_1 + j} = A_{ij}B_{\ell k}. $$

Below we use a simple example considering two $2 \times 2$ matrices, $A$ and $B$, to illustrate the difference between the Kronecker product and Box product.

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}, B = \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix}.$$
The Kronecker product is:

\[
A \otimes B = \begin{bmatrix}
    a_{11}b_{11} & a_{11}b_{12} & a_{12}b_{11} & a_{12}b_{21} \\
    a_{11}b_{21} & a_{11}b_{22} & a_{12}b_{12} & a_{12}b_{22} \\
    a_{21}b_{11} & a_{21}b_{12} & a_{22}b_{11} & a_{22}b_{12} \\
    a_{21}b_{21} & a_{21}b_{22} & a_{22}b_{21} & a_{22}b_{22} \\
\end{bmatrix}, \quad \text{(B.1)}
\]

and the Box product is:

\[
A \boxtimes B = \begin{bmatrix}
    a_{11}b_{11} & a_{11}b_{12} & a_{12}b_{11} & a_{12}b_{21} \\
    a_{11}b_{21} & a_{12}b_{12} & a_{11}b_{22} & a_{12}b_{22} \\
    a_{21}b_{11} & a_{22}b_{11} & a_{21}b_{12} & a_{22}b_{12} \\
    a_{21}b_{21} & a_{22}b_{21} & a_{21}b_{22} & a_{22}b_{22} \\
\end{bmatrix}.
\]

In the case of 2x2 matrices, the Box product is the the Kronecker product with the second column and the third column switched. A more complicated example and neater expression of box product and kronecker product: Suppose we rewrite A and B into row vectors.

\[
A = \begin{bmatrix}
    a_{11} & a_{12} & a_{13} \\
    a_{21} & a_{22} & a_{23} \\
\end{bmatrix} = \begin{bmatrix}
    a_1 & a_2 & a_3 \\
\end{bmatrix}
\]

\[
B = \begin{bmatrix}
    b_{11} & b_{12} \\
    b_{21} & b_{22} \\
    b_{31} & b_{32} \\
    b_{41} & b_{42} \\
\end{bmatrix} = \begin{bmatrix}
    b_1 & b_2 \\
\end{bmatrix}
\]
The Kronecker product is:

\[
A \otimes B = \begin{bmatrix}
a_1 \otimes b_1 & a_1 \otimes b_2 & a_2 \otimes b_1 & a_2 \otimes b_2 & a_3 \otimes b_1 & a_3 \otimes b_2 \\
a_{11}b_{11} & a_{11}b_{12} & a_{12}b_{11} & a_{12}b_{12} & a_{13}b_{11} & a_{13}b_{12} \\
a_{11}b_{21} & a_{11}b_{22} & a_{12}b_{21} & a_{12}b_{22} & a_{13}b_{21} & a_{13}b_{22} \\
a_{11}b_{31} & a_{11}b_{32} & a_{12}b_{31} & a_{12}b_{32} & a_{13}b_{31} & a_{13}b_{32} \\
a_{11}b_{41} & a_{11}b_{42} & a_{12}b_{41} & a_{12}b_{42} & a_{13}b_{41} & a_{13}b_{42} \\
a_{21}b_{11} & a_{21}b_{12} & a_{22}b_{11} & a_{22}b_{12} & a_{23}b_{11} & a_{23}b_{12} \\
a_{21}b_{21} & a_{21}b_{22} & a_{22}b_{21} & a_{22}b_{22} & a_{23}b_{21} & a_{23}b_{22} \\
a_{21}b_{31} & a_{21}b_{32} & a_{22}b_{31} & a_{22}b_{32} & a_{23}b_{31} & a_{23}b_{32} \\
a_{21}b_{41} & a_{21}b_{42} & a_{22}b_{41} & a_{22}b_{42} & a_{23}b_{41} & a_{23}b_{42} \\
\end{bmatrix}
\]

and the Box product is:

\[
A \boxtimes B = \begin{bmatrix}
b_1 \otimes a_1 & b_1 \otimes a_2 & b_1 \otimes a_3 & b_2 \otimes a_1 & b_2 \otimes a_2 & b_2 \otimes a_3 \\
\end{bmatrix}
\]

\[
= \begin{bmatrix}
a_{11}b_{11} & a_{12}b_{11} & a_{13}b_{11} & a_{11}b_{12} & a_{12}b_{12} & a_{13}b_{12} \\
a_{11}b_{21} & a_{12}b_{21} & a_{13}b_{21} & a_{11}b_{22} & a_{12}b_{22} & a_{13}b_{22} \\
a_{11}b_{31} & a_{12}b_{31} & a_{13}b_{31} & a_{11}b_{32} & a_{12}b_{32} & a_{13}b_{32} \\
a_{11}b_{41} & a_{12}b_{41} & a_{13}b_{41} & a_{11}b_{42} & a_{12}b_{42} & a_{13}b_{42} \\
a_{12}b_{11} & a_{22}b_{11} & a_{23}b_{11} & a_{21}b_{12} & a_{22}b_{12} & a_{23}b_{12} \\
a_{12}b_{21} & a_{22}b_{21} & a_{23}b_{21} & a_{21}b_{22} & a_{22}b_{22} & a_{23}b_{22} \\
a_{12}b_{31} & a_{22}b_{31} & a_{23}b_{31} & a_{21}b_{32} & a_{22}b_{32} & a_{23}b_{32} \\
a_{12}b_{41} & a_{22}b_{41} & a_{23}b_{41} & a_{21}b_{42} & a_{22}b_{42} & a_{23}b_{42} \\
\end{bmatrix}
\]

The comparison of box product and kronecker product in terms of inverse, transpose and
distribution low is shown below:

\[(A \otimes B)(C \otimes D) = AC \otimes BD\]
\[(A \boxtimes B)(C \boxtimes D) = AC \boxtimes BD\]
\[(A \otimes B)^{-1} = A^{-1} \otimes B^{-1}\]
\[(A \boxtimes B)^{-1} = B^{-1} \boxtimes A^{-1}\]
\[(A \otimes B)^T = A^T \otimes B^T\]
\[(A \boxtimes B)^T = B^T \boxtimes A^T\]

A more intuitive comparison is shown in Figure B.2:

Figure B.2: Box product vs Kronecker product
B.2 Rank Selection: AIC vs BIC

As explained in the article, the rank selection is crucial for the validation of the hypothesis testing. A correct rank selection will provide a good FPR, while a wrong rank selection can result in an inflated FPR. Therefore, in the Appendix B, we examine the performance of our way of computing AIC and BIC through synthetic numerical examples. The data used in this section is generated in the same way as what we did in the Section 3.4.1. We would like to see if BIC is too conservative in terms of rank selection. Our first example is a parameter matrix with "T" shape under different values of $C$ since the true rank of "T" is 2. The rank selection results of "T" shape is shown in Figure B.3.

We can see that AIC is able to select rank 2 correctly when $C = 1$ while BIC are still at rank 1.

Now, suppose we now consider the high rank case with "triangle" shape (similar to T shape but with different row length for each row, shown in Figure Figure B.4) under different magnitude. Sample size $n$ is 500, the number of genes $G$ is 74. Figure B.5 shows the rank estimation results. The x-axis represents total indices of 1000 replication.

We can see that AIC is more sensitive than BIC. The true rank of triangle shape parameter matrix B is 3. At $C = 0.5$, AIC has already chosen the correct rank 2 while BIC still stick to rank 1. At $C = 1$, AIC has already chosen rank correctly but BIC gives the wrong rank. When the signal of data is strong enough ($C = 1$), correct rank is chosen by both AIC and BIC.

Therefore AIC is better than BIC under our settings.
Figure B.3: Rank selection results for BIC and AIC of Shape "T". Index is the indices of 1000 replicaitons
B.3 Proof of Prop 1

The loglikelihood function is:

\[
h(B_1, B_2, \beta) = \frac{-n}{2} \log(2\pi) - \frac{n}{2} \log(\sigma^2) - \sum_{i=1}^{n} \frac{1}{2\sigma^2} (y - z_i^T \beta - \langle B, X_i \rangle)^2
\]

\[
= \frac{-n}{2} \log(2\pi) - \frac{n}{2} \log(\sigma^2) - \sum_{i=1}^{n} \frac{1}{2\sigma^2} (y - z_i^T \beta - \langle B_1B_2^T, X_i \rangle)^2
\]
Figure B.5: Rank selection results for BIC and AIC of Shape "Triangle". Index is the indices of 1000 replications.
\[ \nabla_\beta h = \frac{1}{\sigma^2} \sum_{i=1}^{n} (y_i - z_i^T \beta - \langle B, X_i \rangle) z_i \]

\[ \nabla^2_\beta h = -\frac{1}{\sigma^2} \sum_{i=1}^{n} z_i z_i^T \]

\[ I_\beta = \frac{1}{\sigma^2} \sum_{i=1}^{n} z_i z_i^T \]

\[ \nabla^2_{\beta,B_1} h = -\frac{1}{\sigma^2} \sum_{i=1}^{n} z_i vec(X_i B_2)^T \]

\[ I_{\beta,B_1} = \frac{1}{\sigma^2} \sum_{i=1}^{n} z_i vec(X_i B_2)^T \]

\[ \nabla^2_{\beta,B_2} h = -\frac{1}{\sigma^2} \sum_{i=1}^{n} z_i vec(X_i^T B_1)^T \]

\[ I_{\beta,B_2} = \frac{1}{\sigma^2} \sum_{i=1}^{n} z_i vec(X_i^T B_1)^T \]

\[ \nabla_{B_1} h = \frac{1}{\sigma^2} \sum_{i=1}^{n} (y_i - z_i^T \beta - \langle B, X_i \rangle) vec(X_i B_2) \]

\[ \nabla^2_{B_1} h = -\frac{1}{\sigma^2} \sum_{i=1}^{n} vec(X_i B_2) vec(X_i B_2)^T \]

\[ I_{B_1} = \frac{1}{\sigma^2} \sum_{i=1}^{n} vec(X_i B_2) vec(X_i B_2)^T \]

\[ \nabla_{B_2} h = \frac{1}{\sigma^2} \sum_{i=1}^{n} (y_i - z_i^T \beta - \langle B, X_i \rangle) vec(X_i^T B_1) \]

\[ \nabla^2_{B_2} h = -\frac{1}{\sigma^2} \sum_{i=1}^{n} vec(X_i^T B_1) vec(X_i^T B_1)^T \]

\[ I_{B_2} = \frac{1}{\sigma^2} \sum_{i=1}^{n} vec(X_i^T B_1) vec(X_i^T B_1)^T \]

\[ \nabla^2_{B_1,B_2} h = -\frac{1}{\sigma^2} \sum_{i=1}^{n} vec(X_i B_2) vec(X_i^T B_1)^T \]

\[ I_{B_1,B_2} = \frac{1}{\sigma^2} \sum_{i=1}^{n} vec(X_i B_2) vec(X_i^T B_1)^T \]
According to asymptotic distribution of MLE, we have: (without considering \( \beta \) for simplicity)

\[
\begin{align*}
\text{vec}(\hat{B}_1) & \sim \mathcal{N}\left( \text{vec}(B_1), (I_{B_1} - I_{B_2}B_1[I_{B_2}]^{-1}I_{B_1}B_2)\right)^{-1} = V_1 \\
\text{vec}(\hat{B}_2) & \sim \mathcal{N}\left( \text{vec}(B_2), (I_{B_2} - I_{B_1}B_2[I_{B_1}]^{-1}I_{B_2}B_1)\right)^{-1} = V_2
\end{align*}
\]

Then we can have the asymptotic distribution of \( B_1B_2^T \)

\[
\begin{align*}
\hat{B}_1\hat{B}_2^T - B_1B_2^T &= \hat{B}_1\hat{B}_2^T - \hat{B}_1\hat{B}_2^T + \hat{B}_1\hat{B}_2^T - B_1B_2^T \\
&= \hat{B}_1(\hat{B}_2 - B_2)^T + (\hat{B}_1 - B_1)B_2^T
\end{align*}
\]

\[
\begin{align*}
\text{vec}(\hat{B}_1(\hat{B}_2 - B_2)^T) &= (I \otimes \hat{B}_1)\text{vec}(\hat{B}_2 - B_2) \quad \text{(B.2)} \\
\text{vec}((\hat{B}_1 - B_1)B_2^T) &= (B_2 \otimes I)\text{vec}(\hat{B}_1 - B_1)
\end{align*}
\]

Since \( \hat{B}_1 \xrightarrow{p} B_1 \), then by Slutsky’s theorem, we have

\[
\text{vec}(\hat{B}_1(\hat{B}_2 - B_2)^T) = (I \otimes \hat{B}_1)\text{vec}(\hat{B}_2 - B_2) \sim \mathcal{N}\left( 0, (I \otimes B_1)V_2(B_1^T \otimes I) \right)
\]

\[
\text{vec}((\hat{B}_1 - B_1)B_2^T) = (B_2 \otimes I)\text{vec}(\hat{B}_1 - B_1) \sim \mathcal{N}\left( 0, (B_2 \otimes I)V_1(B_2^T \otimes I) \right)
\]

\[
\begin{align*}
\text{cov}\left((I \otimes \hat{B}_1)\text{vec}(\hat{B}_2 - B_2), (B_2 \otimes I)\text{vec}(\hat{B}_1 - B_1)\right) \\
= (I \otimes \hat{B}_1)\text{cov}\left(\text{vec}(\hat{B}_2 - B_2), \text{vec}(\hat{B}_1 - B_1)\right)(B_2^T \otimes I) = (I \otimes \hat{B}_1)I_{B_1B_2} (B_2^T \otimes I)
\end{align*}
\]
Thus as $n \to \infty$, we have:

$$\sqrt{n} \vec{\hat{B}T_1 - B_1B_2^T} \xrightarrow{D} \mathcal{N} \left( 0, \ n \left[ (I \otimes B_1)V_2(B_1^T \otimes I) + (B_2 \otimes I)V_1(B_2^T \otimes I) + 2(I \otimes B_1)I_{B_2}B_2 \ (B_2^T \otimes I) \right] \right)$$

### B.4 EM algorithm for binary data

#### The Derivation of EM Algorithm

Suppose $y$ is a $n$-by-1 response vector whose $i$th element is $y_i$. $z$ is a $n$-by-1 latent vector with $i$th element, $z_i$. The parameter is defined by $\beta$. Now, we will work on the log likelihood function first:

$$l(\beta) = \sum_{i=1}^{n} \log(p(y_i|\beta))$$

$$= \sum_{i=1}^{n} \log \left( \sum_{z_i} p(y_i, z_i|\beta) \right)$$

$$= \sum_{i=1}^{n} \log \left( \sum_{z_i} Q_i(z_i) \frac{p(y_i, z_i|\beta)}{Q_i(z_i)} \right)$$

$$= \sum_{i=1}^{n} \log \left( E_{z_i \sim Q_i} \left[ \frac{p(y_i, z_i|\beta)}{Q_i(z_i)} \right] \right)$$

$$\geq \sum_{i=1}^{n} E_{z_i \sim Q_i} \left[ \log \left( \frac{p(y_i, z_i|\beta)}{Q_i(z_i)} \right) \right]$$

$$= \sum_{i=1}^{n} \sum_{z_i} Q_i(z_i) \left[ \log \left( \frac{p(y_i, z_i|\beta)}{Q_i(z_i)} \right) \right]$$

$$\equiv T(\beta, \{Q_i\}_{i=1,...,n})$$

where $Q_i$ is a density function related to $z_i$. In order to construct a legit lower bound of log likelihood function $l(\beta)$, we require $l(\beta) \geq T(\beta, \{Q_i\}_{i=1,...,n})$ and "$\equiv$" correct only if $\beta = \beta_t$. Moreover, "$\equiv$" means Jensen’s inequality takes "$=$" as well. So $Q_i$ has to be something that
can make the ratio \( \frac{p(y_i, z_i | \beta)}{Q_i(z_i)} \) is a constant as a function of \( z_i \). Therefore, we take

\[
Q_i(z_i) = \frac{p(y_i, z_i | \beta)}{\sum_{z_i} p(y_i, z_i | \beta)}
\]

Now notice \( \sum_{z_i} Q_i(z_i) = 1 \) indicates \( Q_i(z_i) \) is a valid density function; (2) When \( \beta = \beta_t \), we have:

\[
T(\beta_t, \{Q_i\}_{i=1,...,n}) = \sum_{i=1}^{n} \left[ \sum_{z_i} Q_i(z_i) \right] \log p(y_i | \beta_t) = \sum_{i=1}^{n} \log p(y_i | \beta_t) = l(\beta_t)
\]

Noticed that the \( \beta \) doesn’t appear as an argument of function \( Q_i \) is because the value of \( \beta \) is known as \( \beta_t \). In addition, \( Q_i \) can be further simplified into \( p(z_i | y_i, \beta_t) \), which is a distribution of the latent variable conditional observed data and the value of \( \beta \) at \( t \)-th iteration.

Two steps of EM algorithm can be summarized as following:

**Q step:**

\[
Q_i(z_i) = \frac{p(y_i, z_i | \beta_t)}{\sum_{z_i} p(y_i, z_i | \beta_t)}
\]

**M step:**

\[
\beta_{t+1} = \arg \max_{\beta} T(\beta, \{Q_i\}_{i=1,...,n})
\]

**Proof of Equivalence between LM and TR when \( R = 3 \)**

Recall we will need to adjust \( B_1 \) and \( B_2 \) to address the identifiability issue. Recall the constraints that we impose restrict \( B_1 \) and \( B_2 \) to take the following forms:

\[
B_1 = \begin{bmatrix} I_R \\ B_{12} \end{bmatrix} \quad \text{and} \quad B_2 = \begin{bmatrix} B_{21} \\ B_{22} \end{bmatrix}
\]
When $R = 3$, the previous restricted $B_1$ and $B_2$ becomes

\[ B_1 = I_3 \]
\[ B_2 = B \]

This is because we need to keep the outer product of $B_1$ and $B_2$ unchange. However, $B_1$ with constraints becomes a full identity matrix when the rank is 3 and the corresponding adjustment make the $B_2$ with constraints is just $B$ trivially.

**An Application To The Probit Regression Example**

In addition to the setting described in the first section regarding the observed data and latent data. The relationship between the latent dat and the observed data is $y_i = I(z_i > 0)$ and $z = X\beta + \epsilon$, where $\epsilon \sim \mathcal{N}(0, \sigma^2 I_n)$ where $z$ is a n-by-1 vector. **Q step:**

\[ Q_i(z_i) = p(z_i|y_i) = TN_+ I(y_i = 1) + TN_- (1 - I(y_i = 1)) \]

where $TN_+$ and $TN_-$ are truncated normal distribution variables whose domain are $\mathbb{R}^+$ and $\mathbb{R}^-$ $\cup \{0\}$, respectively. **M step:**

\[
T(\beta, \{Q_i\}_{i=1,..,n}) = \sum_{i=1}^{n} E_{z_i \sim Q_i} \left[ \log \left( \frac{p(y_i, z_i|\beta)}{Q_i(z_i)} \right) \right]
= \sum_{i=1}^{n} E_{z_i \sim Q_i} \left[ \log \left( \frac{p(z_i|\beta)p(y_i|z_i, \beta)}{Q_i(z_i)} \right) \right]
= \sum_{i=1}^{n} E_{z_i \sim Q_i} \left[ \log(p(z_i|\beta)) + \log(p(y_i|z_i, \beta)) - \log(Q_i(z_i)) \right]
= \sum_{i=1}^{n} E_{z_i \sim Q_i} \left[ \log(p(z_i|\beta)) + 0 \right] + C
= \sum_{i=1}^{n} E_{z_i \sim Q_i} \left[ \log(p(z_i|\beta)) \right] + C
\]
where \( C \) is a constant as a function \( \beta \) since we are only interested in any terms with the parameter, \( \beta \). 0 is because if you take the expectation, for different value of \( z_i \), \( y_i \) is a constant, 1 or 0. In other word, \( y_i \)'s density function is constant, 1, if the value of \( z_i \) is given. In our case, \( Q_i = Q \). Thus:

\[
\beta_{t+1} = \arg\max_{\beta} T(\beta, Q) \\
= \arg\max_{\beta} \sum_{i=1}^{n} E_{z_i \sim Q_i} \left[ \log(p(z_i|\beta)) \right] + C \\
= \arg\max_{\beta} \sum_{i=1}^{n} E_Q \left[ \log(p(y_i|\beta)) \right] \\
= \arg\max_{\beta} \sum_{i=1}^{n} E_Q \left[ \frac{1}{2} ||z_i - x_i^T \beta||^2 \right] \\
= \arg\max_{\beta} E_Q \left[ \frac{1}{2} ||z - X\beta||^2 \right]
\]

The first order condition for stationary point is:

\[
E_Q \left[ \nabla_{\beta} \left( \frac{1}{2} ||z - X\beta||^2 \right) \right] = X^T (E_Q z - X\beta) = 0
\]

Thus

\[
\beta_{t+1} = (X^T X)^{-1} X^T E_Q z
\]
We have already known $E_Qz$ is a truncated normal distribution. The domain depends on the value of $y_i$. Therefore, for any $i = 1, \ldots, n$,

\[
E(z_i|y_i = 1, \beta_t) = x_i^T \beta_t + \frac{\phi(x_i^T \beta_t)}{1 - \Phi(-x_i^T \beta_t)}
\]

\[
E(z_i|y_i = 0, \beta_t) = x_i^T \beta_t - \frac{\phi(x_i^T \beta_t)}{\Phi(-x_i^T \beta_t)}
\]

Then we have already got the EM algorithm for probit regression model.

**An Application To The Probit Tensor Regression Example**

Most of things are the same as the case for regular regression model. The difference is $z_i = \langle X_i, B \rangle + \epsilon_i$. The algorithm can be summarized as follows: **Iterate:** Start from $B_t$:

\[
E(z_i|y_i = 1, B_t) = \langle X_i, B_t \rangle + \frac{\phi(\langle X_i, B_t \rangle)}{1 - \Phi(-\langle X_i, B_t \rangle)}
\]

\[
E(z_i|y_i = 0, B_t) = \langle X_i, B_t \rangle - \frac{\phi(\langle X_i, B_t \rangle)}{\Phi(-\langle X_i, B_t \rangle)}
\]

\[
B_{t+1} = TS(E_Qz, X)
\]

where $X = \{X_i\}_{i=1,\ldots,n}$, $TS$ is our tensor regression solver that is ready for Gaussian data case.

**Modifications to the Application of EM algorithm to Probit Tensor Regression**

**Scenario and Implementation Details**

Recall the the derived EM algorithm in the previous section, we can split the step using tensor regression solver updating $B$ into two separate updates. This is because we conduct low rank approximation technique to $B$ so that $B$ can be written as $B_1B_2^T$. Therefore the complete modified EM algorithm to binary case can be summarized as follows: Suppose the sample size is $n$,
Iterate: Start from $B_t$: for any $i = 1, 2, \ldots, n$

\[
E(z_i|y_i, \beta_t) = \langle X_i, B_t \rangle + \frac{\phi(\langle X_i, B_t \rangle)}{y_i - \Phi(-\langle X_i, B_t \rangle)}
\]

\[
[B_1]_{t+1} = LM(EQz, X_i, [B_2]_t)
\]

\[
[B_2]_{t+1} = LM(EQz, X_i, [B_1]_{t+1})
\]

\[
[B]_{t+1} = [B_1]_{t+1}[B_2]_{t+1}^T
\]

---

Algorithm 4 TR-EM

1: procedure TR($B_1^{(0)}$, $B_2^{(0)}$, $\beta^{(0)}$) \hspace{1cm} \triangleright \text{Initialize $B_1^{(0)}$, $B_2^{(0)}$ and $\beta^{(0)}$}
2: repeat
3: \hspace{1cm} $E(M_j^{(i+1)}|y_j, \beta^{(i)}, B^{(i)}) = \langle X_j, B \rangle + \frac{\phi(\langle X_j, B^{(i)} \rangle)}{y_j - \Phi(-\langle X_j, B^{(i)} \rangle)}$ \hspace{1cm} \triangleright \text{For } j = 1, \ldots, N
4: \hspace{1cm} $\beta^{(i+1)} = LM(EM^{(i+1)}, Z|B_1^{(i)}, B_2^{(i)})$
5: \hspace{1cm} $B_1^{(i+1)} = LM(EM^{(i+1)}, A(B_2^{(i)})|\beta^{(i+1)}, B_2^{(i)})$
6: \hspace{1cm} $B_2^{(i+1)} = LM(EM^{(i+1)}, B(B_1^{(i+1)})|\beta^{(i+1)}, B_1^{(i+1)})$
7: \hspace{1cm} $B^{(i+1)} = B_1^{(i+1)}(B_2^{(i+1)})^T$
8: until convergence
9: end procedure

where $LM(y, X, \ldots)$ indicates a least square problem with a continuous response variable $y$, a design matrix $X$ and conditional on some covariates. The advantage of using TR-EM is to solve a binary problem using a simple least square tensor regression model.
### B.5 Additional Information for Real Data Analysis Results

<table>
<thead>
<tr>
<th>Gene</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>BBC3</strong></td>
<td>TP53 activates the expression of BBC3; BBC3 binds to BCL2 and induces release of CYCS proteins, which lead to induction of apoptosis (Nakano and Vousden (2001)).</td>
</tr>
<tr>
<td><strong>BID</strong></td>
<td>BID binds to BAX and BCL2, and BID is also both structurally and functionally related to BAX (Billen et al. (2009)). Moreover, the cellular BCL2/ BAX ratio is important in regulating apoptosis, while the ratio is lower in EC (Sedlak et al. (1995); Hanada et al. (1995); Vaskivuo et al. (2002)).</td>
</tr>
<tr>
<td><strong>CASP3</strong></td>
<td>Overexpression of MicroRNA-101 can decrease the proliferation of endometrial cancers cells (Konno et al. (2014)). Activity of CASP3 will increase during the process.</td>
</tr>
<tr>
<td><strong>CASP8</strong></td>
<td>The activation of both CASP3 and CASP8 contribute to the reduced viability of endometrial cancers cells in TRAIL-induced apoptosis and in BAX and FAS-induced apoptosis (Yeramian et al. (2013))</td>
</tr>
<tr>
<td><strong>CCNB1</strong></td>
<td><em>CCNB1</em> is a critical factor in proliferation and differentiation of endometrial cells (Tang et al. (2009)). Mutation in <em>CCNB1</em> was frequently observed in endometrioid carcinomas (type 1 EC) (Guan et al. (2011)).</td>
</tr>
<tr>
<td><strong>CCND3</strong></td>
<td><em>CCND3/CDK4</em> phosphorylates retinoblastoma (RB) proteins and then regulated the G(1)/S transition. Phosphorylated RB was observed in <em>BAX</em> and <em>FAS</em>-induced apoptosis in endometrioid carcinomas (Yeramian et al. (2013)).</td>
</tr>
<tr>
<td><strong>CCNE1</strong></td>
<td><em>CCNE1</em> has shown to be overexpressed in EC (Cassia et al. (2003)).</td>
</tr>
<tr>
<td><strong>CCNG1 &amp; CCNG2</strong></td>
<td>Cyclin G was overexpressed in breast and prostate cancer (Reimer et al. (1999)). <em>CCNG2</em> transcription was upregulated by nodal signaling which has been shown to suppress proliferation in human ovarian cancer cells (Fu and Peng (2011)).</td>
</tr>
<tr>
<td><strong>CDK1</strong></td>
<td>MicroRNA miR-302 inhibits the tumorigenicity of endometrial cancer cells by inhibition of <em>CCND1</em> and <em>CDK1</em> (Yan et al. (2014)).</td>
</tr>
<tr>
<td><strong>CDKN2A</strong></td>
<td>Loss of function of <em>CDKN2A</em> (also called as p16INK4a) has been proposed to contribute to endometrial carcinogenesis (Dedes et al. (2011)).</td>
</tr>
<tr>
<td><strong>CHEK1</strong></td>
<td>MicroRNA-449a exerted its tumor suppressor function by targeting <em>CDC25A</em> in endometrial cancer cells (Ye et al. (2014)). <em>CDC25</em> regulates cell cycle through <em>CHEK1</em> dependent signaling pathway.</td>
</tr>
<tr>
<td><strong>FAS</strong></td>
<td>Transcription of <em>FAS</em> is downregulated in endometrial carcinoma tissues (Das et al. (2000)).</td>
</tr>
<tr>
<td><strong>GADD45G</strong></td>
<td>Transcription of <em>GADD45G</em> is downregulated in late stage of endometrial cancer compared to early stage of cancer (Wong et al. (2007)).</td>
</tr>
<tr>
<td><strong>IGF1</strong></td>
<td>Evaluation of <em>IGF1</em> in serum has been observed in patients with endometrial cancer (Kreuzinger et al. (2015)). Moreover, insulin can inhibit GFBP-1 mRNA and protein expression resulting in elevated levels of free IGF-1, which can thus promote the development of EC (Mu et al. (2012); Lathi et al. (2005)).</td>
</tr>
<tr>
<td><strong>PPM1D</strong></td>
<td>High frequency of loss-of-function mutations of <em>PPM1D</em> was observed in uterine corpus (Dudgeon et al. (2013)).</td>
</tr>
<tr>
<td><strong>SEPRINB5</strong></td>
<td>Maspin, also called as <em>SEPRINB5</em>, is overexpressed during the progression of endometrioid EC (Tsuji et al. (2007)).</td>
</tr>
<tr>
<td><strong>SESN3</strong></td>
<td>Methylation of <em>SESN3</em> associates with EC (Izmirli et al. (2013)).</td>
</tr>
<tr>
<td><strong>THBS1</strong></td>
<td><em>THBS1</em> tends to be overexpressed in esophageal squamous cell carcinoma (Zhou et al. (2009)).</td>
</tr>
<tr>
<td><strong>TNFRSF10B</strong></td>
<td><em>TNFRSF10B</em> is overexpressed in endometrioid adenocarcinoma (Gottwald et al. (2013)).</td>
</tr>
<tr>
<td><strong>TP53I3</strong></td>
<td><em>TP53I3</em> (also called as PIG3): low frequency of decreased expression observed in breast and lung carcinomas (Gorgoulis et al. (2004)).</td>
</tr>
<tr>
<td><strong>RRM2B</strong></td>
<td><em>RRM2B</em> has reported to be involved in tumorigenesis of several cancers (Wang et al. (2009)).</td>
</tr>
<tr>
<td><strong>PREP</strong></td>
<td>Elevated <em>PREP</em> mRNA transcripts in ovarian and colorectal tumors compared with normal tissue (Myöhänen et al. (2012)).</td>
</tr>
</tbody>
</table>
C.1.1 Methylation_clean0.sh

# Meng Yang
# myang13@ncsu.edu
# This code is to merge cpg vs gene table with gene list in
# a pathway we are going to get a gene in a certain
# pathway vs cpg table
# /bin/sh
# remove double quotes sign in gene_vs_cpg.txt table

sed 's/"//g' gene_vs_cpg.txt > gene_vs_cpg_noquotes.txt
sed 's/"//g' methydata_nomissing.txt > methydata_nomissing_noquotes.txt
sed 's/"//g' gene_list_all.txt > gene_list_noquotes.txt
sed 's/"//g' methydata.txt > methydata_noquotes.txt
C.1.2 Methylation_clean1.sh

Listing C.1: bash version

```
# Meng Yang
# myang13@ncsu.edu
#
# This code is to reform the cpg_gene_symbol table
# The output file is the gene_vs_cpg table that only genes
# within certain pathway will be displayed.
#
# All of files will have no double quotes
#
# Snapshot of the file:
# cg18174089,MAPK8
# cg08866557,MAPK8
# cg10308810,MAPK8
# cg20525763,MAPK8
# cg05241265,MAPK8
# cg07103124,MAPK8
# cg24283186,MAPK8
# cg26830985,MAPK8
# cg14881393,MAPK8
# cg01512155,MAPK8
# cg09359802,MAPK8
# cg06178786,MAPK8
# cg19872284,MAPK8
# cg19612574,MAPK8
```
```
# cg13474011,MAPK8
# cg03883275,MAPK8
# cg07259660,MAPK8
# cg06272678,PIK3R3
# cg15076218,PIK3R3
# cg17533684,PIK3R3
# cg02281178,PIK3R3
# cg21579472,PIK3R3
# cg01028869,PIK3R3
# cg18818048,PIK3R3
# cg17169489,PIK3R3
# cg15837086,PIK3R3

# ./Methy_clean_step1.sh gene_vs_cpg_noquotes.txt |
# sort > genelist_vs_cpg.txt

/bin/sh

input_file1_name=$1 # cpg vs gene table
output_file1_name=$2 # any output file name will do
awk 'BEGIN { FS = OFS = "\"," } 
#(NR == FNR) {range[$0]; next}
# read all files in the first file into range
{
    split($0, arry, /,/) # split key with , and store in array [arry[2], arry[3]
    split(arry[5], arrygene, /;/) # split key with ; and store in array [arry[2], arry[3]
```
j = 0;
for(key in arrygene){
    if (!(arrygene[key] in duplicateArray)){
        duplicateArray[arrygene[key]] = 1
    }
    #print arrygene[key], duplicateArray[arrygene[key]]
};
for (j in duplicateArray){
    if (j) { print $1, j, $2, $3}
}
duplicateArray
}

C.1.3 Methylation_clean2.sh

Listing C.2:  bash version

# Meng Yang
# myang13@ncsu.edu
# /bin/sh
#
# This is code is to put gene symbol behind sample ID
# and cpg ID to generate the file like
#
#TCGA-A5-A1OH-01A,cg00004533,0.93119284220995,chr21, GeneA
#TCGA-A5-A1OH-01A,cg00021028,0.0651497265000608,chr21, GeneA
#TCGA-A5-A1OH-01A,cg00025591,0.704287120114766,chr21, GeneB

143
input_file1_name=$1 #chx.txt
#input_file2_name=$2 #genelist vs cpg .txt
#output_file1_name=$3

awk '"
BEGIN { FS = OFS = "\," }

NR>1{
    a = $1","$2
    arr[a] += $3
    count[a] += 1
    choo[a] = $4
}

END{
for (a in arr) {
    print a,arr[a] / count[a],choo[a]
}
'
$input_file1_name # > $output_file1_name

C.1.4 Methy\_pathway\_selection.sh

Listing C.3: bash version

```
# Meng Yang
# myang13@ncsu.edu
#
# This code is to subset the raw data
# according the gene list of a certain path way
# All of files will have no double quotes
#
# /bin/sh
input_file1_name=$1
# cleand raw gene level methylation data for each choromosome
input_file2_name=$2
output_file1_name=$3
# gene list in a certain pathway
awk '\
BEGIN { FS = OFS = ",," } 
(NR == FNR) {range[$0]; next}
# read all files in the first file into range
{
```
for (key in range)
{
    split(key, arry, /,/
    # split key with , and store in array arry[2], arry[3]
    #print arry[1],arry[2], arry[3]
    if ($1 == arry[2]){
        print arry[1],arry[2], arry[3],arry[4]
    }
}
'} $input_file1_name $input_file2_name > $output_file1_name

C.1.5 Methy_sampleID_splitting.sh

Listing C.4: bash version

Meng Yang
myang13@ncsu.edu

This code is to subset the raw data
according the gene list of a certain path way
All of files will have no double quotes TCGA-BS-A0T9-01A

/bin/sh
input_file1_name=$1
# clean raw gene level methylation data for each chromosome
#$pathway_name=$2
# gene list in a certain pathway
awk '\"
BEGIN { FS = OFS = "," }

#(NR == FNR) {range[$0]; next}
# read all files in the first file into range
{
    split($0, arry, /,/)  
    # split key with , and store in array arry[2], arry[3]
    split(arry[1], sampleID, /-/)
    print >> ("methylation_tensor_regression_pathway

    /methylation_cleandata"sampleID[4].".txt")

}

' $input_file1_name #$pathway_name

C.2 Codes For CVNs Data Processing Pipeline

C.2.1 CNV_clean_step0.sh

Listing C.5: bash version

# Meng Yang
# myang13@ncsu.edu
# /bin/sh
# This file contains the bash code that
# removes the double quotes of the data file
#

sed 's//g' gene_list KEGG_CHEMOKINE_SIGNALING_PATHWAY.txt
>
> gene_list KEGG_CHEMOKINE_SIGNALING_PATHWAY_noquotes.txt
# Meng Yang
# myang13@ncsu.edu
# /bin/sh
input_file1_name=$1
input_file2_name=$2
output_file1_name=$3

awk 'BEGIN { FS = OFS = "," } (NR!=1 && NR == FNR) {range[$0]; next}
# read all files in the first file
{
    for (key in range){
        #print key
        split (key, arry, /,/) # split key with , and store in array [arry[2], arry[3]

        if (!(arry[4] < $4 || $5 < arry[3])) {
            # if intersection
            a = arry[4] - $4 + 1
            b = $5 - arry[3] + 1
            len = (a < b)? a : b
            density = $6/len
        }
    }
}'
if (density >= 0.001 && ($7 > 0.1375035||$7 < -0.1520031)) {
    # [log(2.2/2, base = 2), log(1.8/2, base = 2)]
    print $2, $7, arry[5], arry[6], len
}
}
# print arry[5], arry[2], $0
}
'
$input_file1_name $input_file2_name > $output_file1_name

C.2.3 CNV_clean_step2.sh

Listing C.7: bash version

# Meng Yang
# myang13@ncsu.edu
# /bin/sh
input_file1_name=$1
output_file_name=$2

awk '"
BEGIN{
FS=OFS="","}
{
    b = $1$3
    if (a[b]<$5){
        a[b]=$5
        data[b]=$0
    }
}
C.2.4 CNV_clean_step3.sh

Listing C.8: bash version

```bash
# Meng Yang
# myang13@ncsu.edu
# /bin/sh
input_file1_name=$1
input_file2_name=$2
output_file1_name=$3

awk '"
BEGIN { FS = OFS = "," }
(NR!=1 && NR == FNR) {range[$0]; next}
# read all files in the first file
{
   for (key in range){
      #print key
      #split (key, arry, /,/) # split key with , and store in array [arry[2], arry[3]
```
#print arry[4]
if (key == $4) { # if intersection
    print key, $1, $2
}
}
'
$input_file1_name $input_file2_name > $output_file1_name

C.3 Codes For Transcripts Data Processing Pipeline

C.3.1 transcripts_preprocessing.R

# Reading data from local
load("/Users/mengyang1/Documents/Project/Realdata_Tzeng/transcript/transcripts_UCEC.Rdata")

#trans.UCEC$'TCGA-A5-A0VO-01A-21R-A109-07'

# because of some issues.
# Not All of the transcript ID has
# corresponding gene symbol, We will deal
# with them in the preprocess procedure.
pathway_name = "KEGG_CHEMOKINE_SIGNALING_PATHWAY"

# truncate the sample ID #
name1 <- names(trans.UCEC)
for (i in 2:588 ){
    name1 <- substr(name1, 1, 15)
}

names(trans.UCEC) <- name1

# transcript ID to gene ID 

trans.UCEC$X1 <- substr(trans.UCEC$X1, 1, 15)

# transcript ID to gene ID

ind <- trans.UCEC$X1 %in% trans.compound.info$ensembl_gene_id

# indices for those transcript ID that has gene ID

# clean the data

# Remove those data that has no gene ID 

# This is the gene mapping version we used in this
# dataset has no information for certain transcript ID.
# Thus we will remove those
# transcript has no corresponding gene symbols.

trans.UCEC.clean <- list()
for (i in 1:(length(trans.UCEC)-1)){
    trans.UCEC.clean[[i+1]] <- trans.UCEC[[i+1]][ind]
}
trans.UCEC.clean[[1]] <- trans.UCEC$X1[ind]
names(trans.UCEC.clean) <- names(trans.UCEC)
name1 <- names(trans.UCEC.clean)
# trans.UCEC.clean contains those values
# that has corresponding gene ID in the TCGA dataset
n <- length(names(trans.UCEC.clean)) - 1
nn <- length(trans.UCEC.clean$'TCGA-A5-A0VO-01')

en_vs_gene <- cbind(trans.compound.info$external_gene_name,
                     trans.compound.info$ensembl_gene_id)

# match ense ID with gene symbol ID #
# hash table #
ret <- en_vs_gene[,1]
names(ret) <- en_vs_gene[,2]
aa <- ret[trans.UCEC.clean$X1]
trans.UCEC.clean$X1 <- aa
names(trans.UCEC.clean$X1) <- NULL

# Given Gene list in a certain pathway, we need to find the subset #

#transcriptdata.txt is the sample ID
# vs gene vs data matrix within certain pathway
#the remaining work is to do in bash script