NNT : 2016SACLE050

THESE DE DOCTORAT
DE
L'UNIVERSITE PARIS-SACLAY
PREPAREE A
"L'UNIVERSITE EVRY VAL D'ESSONNE"

ECOLE DOCTORALE N°577
Structure et dynamique des systèmes vivants (SDSV)
Spécialité de doctorat : Sciences de la vie et de la santé

Par

M Trung Ha

A multivariate learning penalized method for a joined inference of gene expression levels and gene regulatory networks

Thèse présentée et soutenue à Paris, le 2 décembre 2016 :

Composition du Jury :

M.	Causeur, David	Professeur, AgroCampus Ouest	Président
M.	Bar-Hen, Avner	Professeur du Cnam	Rapporteur
Mme.	Villa-Vialaneix, Nathalie	Chargée de Recherche, INRA	Rapporteur
M.	Picard, Franck	Directeur de Recherche, CNRS	Examinateur
Mme.	Martin-Magniette, Marie-Laure	Directrice de Recherche, INRA	Directrice de thèse
M.	Chiquet, Julien	Chargé de Recherche, INRA	Co-directeur de thèse
M.	Rigaill, Guillem	Maître de Conférences, UEVE	Co-directeur de thèse

Une méthode d'apprentissage multivariée et pénalisée pour l'inférence jointe des niveaux d'expression et des réseaux de régulation génique

Entre plusieurs conditions biologiques, le comportement d'un gène peut être affecté soit dans son niveau d'expression moyen, soit dans sa relation aux autres, caractérisée par les covariances entre gènes. Ces deux questions sont généralement traitées de manière indépendante en statistique, bien qu'elles soient clairement liées. Afin de palier à ces limitations, cette thèse vise à proposer une modélisation unifiée de ces deux questions pour identifier les gènes clés affectés dans leur moyenne et/ou dans leurs interactions. Le modèle principal est le modèle graphique gaussien avec des pénalisations sur les paramètres de la moyenne et de la matrice de précision.

Mots clés: Statistique et application en génomique; Apprentissage statistique; Analyse multivariée; Méthodes pénalisées; Optimisation convexe; Transcriptome.

A multivariate learning penalized method for a joined inference of gene expression levels and gene regulatory networks

When comparing different biological conditions, the expression of a gene might shift. It can be a change in terms of its average expression level characterized by its mean. Or it can be a change in terms of its interactions with other genes characterized by the covariance matrix. These two types of events are usually analysed independently even though they are clearly related. In order to alleviate these limitations, we propose in this thesis a unified strategy to address these two questions and identify key genes affected either in terms of their mean or their interactions with other genes. The main statistical model is the Gaussian graphical model with penalizations on the mean and precision matrix parameters.

Keywords: Statistic and genomic applications; Statistical learning; Multivariate analysis; Penalized method; Convex optimization; Transcriptomic.

Acknowledgement

Staying in France for four years is a long journey to me. During that difficult but also wonderful trip, I have a chance to meet great persons and learn many new things from them.

First of all, I would like to thank my three supervisors: Marie-Laure Martin-Magniette, Julien Chiquet and Guillem Rigaill. I thank you for your teaching, your help and your patience during more than three years. I am deeply appreciated that. I will not forget the first time I saw Marie-Laure when you gave me an interviewed about my thesis subject. I will not forget the first time I met Julien in the room I13 when I had a question about the $optim$ function. And, I will not forget the first time I saw Guillem in the presentation room of the Genopole building for my last rehearsal before I defended my thesis subject to get the financial support.

I would like to thank Avner Bar-Hen and Nathalie Villa-Vialaneix for being reviewers of my Ph.D as well as David Causeur and Franck Picard for being examiners of my Ph.D. Especially, I would like to thank Nathalie Villa-Vialaneix for your very detailed and valuable comments on my manuscript.

I would like to thank Christophe Giraud and Pierre Neuvial for accepting to be members of my thesis committee and for giving me many precious advices. Especially, I would like to thank Christophe Giraud for your valuable advices since I was a student in your master program.

I would like to thank Liliane Bel, Stéphane Robin, and Céline Lévy-Leduc for giving me chances to stay and work in Unit UMR 518 at AgroParisTech.

I would like to thank all members of Unit UMR 518 Eric, Pierre, Marie-Pierre, Sarah, Marie, Félix, Marie,... for their kindness and their help since I started my job in there as an intern. Especially, I would like to thank Julie Aubert and Tristan Mary-Huard, my former supervisors. They are also persons who give me information about this thesis.

I would like to thank the secretaries of the laboratory, Francine Puel and Irène Desecures for your help. Especially, I would like to thank Irène Desecures for your enthusiasm and for helping me to find great places to do my Ph.D defense.

During this time, I also have chances to make friend with great persons.

To my friends at the laboratory, I would like to thank Eleana, you are my first mate at the

lab. You are a great person. Your sharing, caring and helping teach me a lot.

I would like to thank Nadia and Rana. We did make a great unknown office. I know that I will never get hungry when Nadia is in our office, and the cake you bought me in Rana's birthday is still the best cake I have ever had. You are a great mate, thoughtful and it's a pleasure to stay with you in the same office.

To Rana, you are a gentle and great friend. I know that I can totally trust you and ask for your help in my big day. And you are a young and pretty lady, I am sure about that.

I would like to thank Sandra and Monia. Although I do not have much time to talk with you, I know that you are very nice friends. And I am really happy with your gift and the card. It's really cute.

To my Vietnamese friends, I would like to thank Tuan Huynh. You are simply a great friend.

I would like to thank Manh Cuong Ngo. You always give me a hand when I need your help. I've learnt a lot from your thinking about life and I known that you will be successful.

I would like to thank Khue Le for your help. I always feel that you are the most generous person I have ever met. It's a pleasure to be one of your friend.

I would like to thank Binh Vu not only because you are a great roommate. To me, you are also a brother. The nicest thing you did for me is to join me into a great group of persons who you consider as your brothers and sisters.

First, I would like to thank Phuong Tran and Huong Nguyen. I enjoy very much your food, but more importantly the family feeling you bring to me whenever I come to visit you. Especially, I would like to thank Son Tran for giving me the cutest card ever.

I would like to thank Son Dinh for your advices. You are a very funny guy and definitely the best fortune teller I know.

I would like to thank Duong Le. You are very sensitive and a great story teller. I feel that you can make a story from the smallest thing. I actually learn much from the way you talk and care to the others. I would like to thank Long Ha. You are definitely the funniest guy I know. This group is not that fun without you two. And I am really happy to know you.

I would like to thank Van Anh and Thinh Ho. You are my most favourite couple. It's always a joy to stay and talk with you two. I am regret that I don't have many chances to learn the way you make beverages.

I would like to thank my sister (hopefully) Phuong Do. I am really surprise by your knowledge, the way you think about a problem, and especially the energy you bring to the others. I learn much from that.

The final member of the group, I would like to thank Ly Tran. You never talk much, but I can feel that you are a calm, mature and thoughtful person. I wish you health and Godspeed.

I would like to thank my roommate Hau Le. You are a great person. When I was too

much busy with my thesis, you helped me to wash my dishes, encouraged me,... I learn a lot the way you treat people, and I admire your energy.

I would like to thank my recent roommates Kien Nguyen and Thu Nguyen. You are great roommates. Especially, I would like to thank Kien Nguyen for your huge support in my big day.

I would like to thank Diep Tran, Viet Vu, Ky and Nhat for your huge support in my defense.

Finally, I would like to thank my father, my mother and my brother.

It is impossible to say “thanks” enough to all of you.

Preface

My thesis started in October, 2013. It is a part of the project SONATA in Unité de Recherche en Génomique Végétale (URGV) which is now Institute of Plant Sciences Paris-Saclay (ISP2). This thesis was funded by Ecole doctorale des Génomes Aux Organismes (GAO) of university Evry Val d'Essonne which has been united into Ecole doctorale Structure et Dynamique des Systèmes Vivants (SDSV) of university Paris-Saclay since 2015.

During my thesis, I worked at unit UMR 518 at the Institut National de la Recherche Agronomique (INRA) under the supervision of Ph.D Marie-Laure Martin-Magniette, leader of the Genomic Networks group of ISP2 as well as Julien Chiquet, Assistant Professor at university Evry Val-d'Essonne and Guillem Rigaill, Assistant Professor at university Evry Val-d'Essonne.

Motivation

My thesis starts from a requirement to study the gene regulatory network of Arabidopsis thaliana plant in different biological conditions. Reconstructing biological networks such as gene regulatory networks is a very interesting and important question in biology. Indeed such networks should help to better understand the regulatory mechanisms of genes or to identify pathways or subset of genes involved in a particular biological function.

We have datasets about the expression level of Arabidopsis thaliana's genes measured by high-throughput technologies. The development of high-throughput technologies allows to collect huge information on thousands of genes simultaneously. However, in most cases, the number of samples is lower than the number of genes. This problem is known as the "high dimensional problem". In this new setting, standard statistical methods used to answer network inference perform less effectively.

In transcriptomic experiments, differential analysis (looking at the mean expression level of genes) and network inference are typically done separately. In this thesis, we propose to answer both questions at the same time. We think that taking into account the mean expression of genes could improve network inference. Conversely, the improvement in terms of network inference may help to better understand the change of mean expression of

genes between conditions which is also referred as the differential analysis question.

To tackle these two questions, we propose a model and an inference scheme to jointly estimate the network and mean expression differences.

Contributions

The thesis has two main contributions:

1. In terms of theory, we provide a model and an inference scheme which can answer the two questions of network inference and differential analysis simultaneously. Besides, we demonstrate the consistency of our procedure in certain settings and provide a computational strategy to estimate the parameters in practise.
2. In terms of application, we provide a procedure to help the identification of key response genes changing in terms of mean or network connections. These genes may involve in the adaptation of organism when environmental conditions change using transcriptomic data.

Organisation

The thesis contains 5 chapters:

- **Chapter 1:** I present the context of my thesis and some standard methods to answer network inference and differential analysis questions.
- **Chapter 2:** I present our model and some theoretical results.
- **Chapter 3:** I compare our model to standard methods which are used to answer differential analysis and network inference.
- **Chapter 4:** I apply our model to two real datasets.
- **Chapter 5:** I give my conclusion and the perspective of my thesis.

Contents

Chapter 1

Introduction

1 Biological context

1.1 DNA definition

DNA is the abbreviation of deoxyribonucleic acid. The structure of DNA is non-static, and is made of two helical chains called strands. Each strand is a sequence of nucleotides. A nucleotide is made of one or several phosphate groups, a five-carbon sugar and a nucleobase among the four primary nucleobases which are Adenine (A), Thymine (T), Guanine (G) and Cytosine (C). The nucleobases on one strand are complementary to the other:

- an A of the forward strand binds to a T of the reverse strand by two hydrogen bonds,
- a T of the forward strand binds to an A of the reverse strand by two hydrogen bonds,
- a G of the forward strand binds to a C of the reverse strand by three hydrogen bonds,
- a C of the forward strand binds to a G of the reverse strand by three hydrogen bonds.

The structure of DNA is shown in Figure 1.1[1].

For eukaryotic organisms, DNA is stored inside the nucleus of the cell. Unlike eukaryotic organisms, prokaryote organisms store their DNA directly in their cytoplasm. For both types of organism, DNA is the storage of all genetic information. It is transmitted to the offspring during fertilization. In a cell, DNA is transcribed in ribonucleic acid (RNA) which is then translated in protein by the ribosomal machinery.

The order of the nucleotides is non random. The sequence of nucleotides defines regions in DNA. Some of the regions are transcribed in RNA and then translated in proteins, others are called non coding regions. A simple view of the genome is to consider it as a succession of coding genes and none coding regions.

[1] URL: http://slideplayer.com/slide/972373/

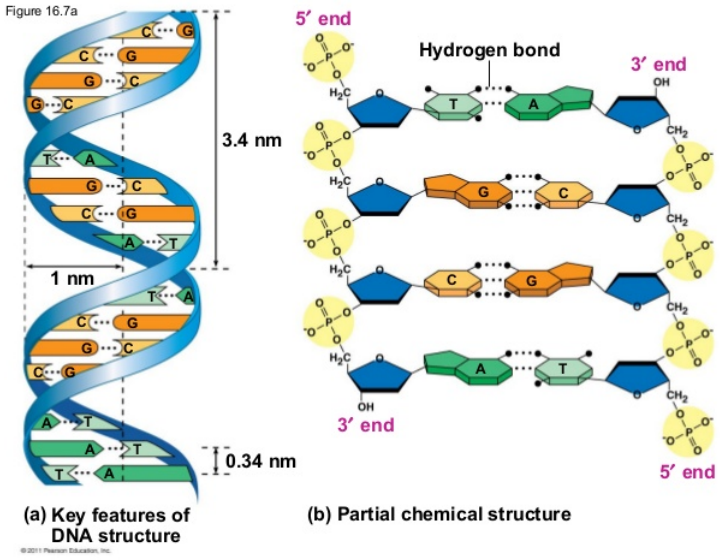


Figure 1.1: Structure of DNA molecule

1.2 Processes for protein synthesis

Protein synthesis is composed of two steps: the transcription and the translation, see Figures 1.2, 1.3[2]. Transcription is the process of copying genetic information from DNA into ribonucleic acid (RNA). Translation is the process in which proteins are created from the RNAs which are taken in charge by ribosomes. In practice, only genes encoding functional RNAs produce proteins.

A gene can be schematically separated into three parts: a promoter, a coding region which stores some genetic information and a termination site. To start the transcription process of a gene, an RNA polymerase comes to the promoter site, and separates the two strands of DNA. The RNA polymerase goes along one strand in the coding region, reads and creates a complementary RNA strand called primary transcript. The primary transcript has a composition very similar to the complementary DNA strand since only the Thymine is replaced with an Uracil. When the RNA polymerase reaches the termination site, the transcription is stopped, and an mRNA molecule is released. Then the RNA polymerase leaves the DNA and the two strands bind again.

After the transcription process, the translation process starts. It consists in decoding mRNA by a ribosome to produce a specific amino acid chain, or a polypeptide. To be specific, the ribosome reads the mRNA three nucleotides at a time to produce an amino acid from the first triplet AUG to a stop codon. When the ribosome reaches this latter, it stops the production of amino acids and the protein is the chain of amino acids linked together.

[2]URL: http://www.lhsc.on.ca/

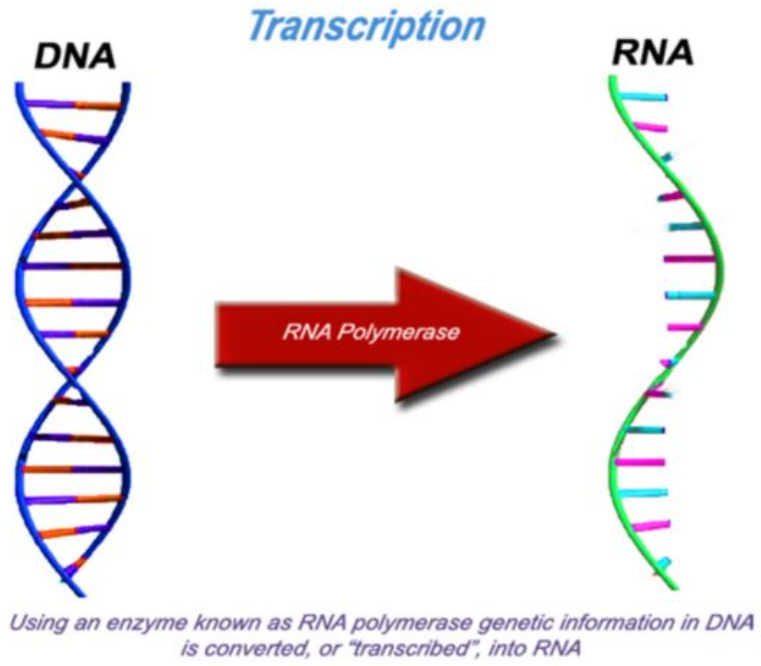


Figure 1.2: Schematic view of transcription process

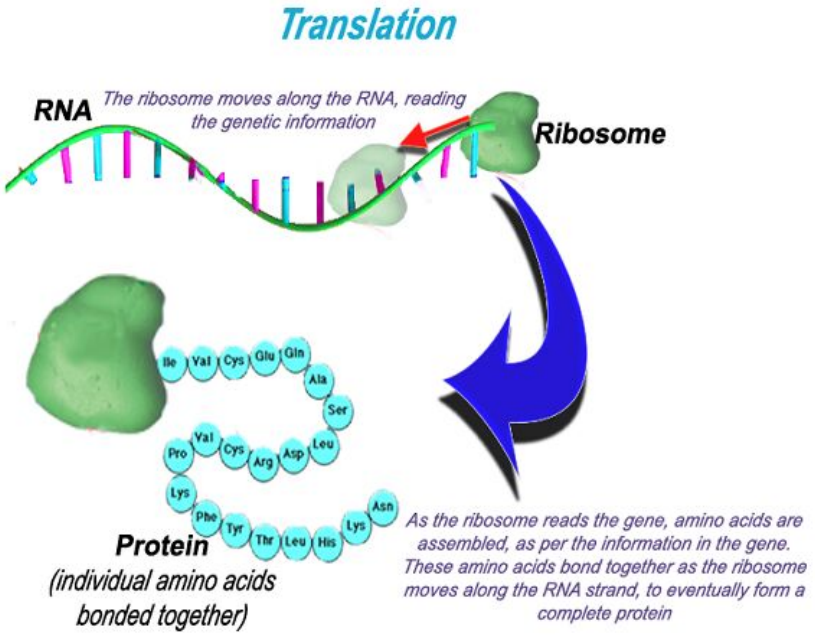


Figure 1.3: Schematic view of translation process

1.3 Gene expression measurement

Transcriptome is the set of all the mRNA molecules present at a given time in a sample. Since 1995, gene expression measurement is performed with high-throughput technologies either by microarrays or RNA sequencing.

Microarrays is a biochip which collects DNA spots attached on a solid surface. Each spot is a cloned DNA sequence corresponding to one gene. Parallely, RNAs are extracted from biological samples, converted into complementary DNAs, amplified and labelled with fluorescent dye. Then they are hybridized on the microarray where they could bind to their complementary DNA presenting in spots. After the process, the abundance of each transcript is measured through the quantification of the fluorescence signal of each spot on the chip. Microarray technique could measure the expression level of thousands of genes simultaneously.

A more recent technology is the RNA sequencing. Its principle is to sequence the transcripts which are sequences of RNA produced by the transcription process. Millions of short sequences with a size between 75 and 100 base pairs, called reads, are generated by a sequencer. These reads are fragments of the transcripts. After a bioinformatics pipeline aiming at attributing the best localization of each read on the genome, expression of each gene is then measured as the number of reads in the genomic region defining the gene. The technique has many advantages compared to microarray technique. For instance, while the list of considered genes are fixed in a microarray, the list of genes is more flexible in RNA sequencing. Therefore, RNA sequencing could detect and measure novel transcripts, or study novel organisms.

In this thesis, we work only with data generated with the microarray technique.

1.4 Objectives of transcriptomic experiments

Thanks to the high-throughput technologies, biologists hope to better understand the role of the genes by comparing several transcriptomes together. Briefly speaking, the comparisons could be organized into two classes.

In the first class, a given condition is considered as a reference and the goal is to evaluate the impact of an another condition on the transcriptome. As an example, plant biologists often compare wild-type plants to a mutant plant in order to identify the role of the mutated genes and how the absence of the mutated genes impacts the expression level of the other genes. In medicine, the reference may be characterized by healthy persons. Their transcriptomes are compared to the ones of ill persons in order to understand the impact of the disease on the gene expression.

The second class of comparison is composed by studies where there is no reference con-

dition. For plants, such studies allow to compare several tissues to identify genes that are specific to a tissue and those which are ubiquitous. In medicine, the studies are used to characterize more precisely subtypes of diseases such as cancer. As an example, in breast cancer, there are several types of cancer defining different transcriptomic profiles. Transcriptomic studies allow to identify genes that could be used to predict the response to the treatment. Typically in cancer, it would be useful to predict which patient will respond to a particular therapy.
Whatever the class of comparison, the two main questions asked by biologists are:

1. Which genes are characteristic of a condition? In other words, which genes are differently expressed between one condition and the others?

2. What are the relationships between genes in a given condition and do they change from one condition to the next?

To answer the first question, one typically runs a differential analysis tool. The goal is to identify if the expression of each gene is altered across the different investigated conditions. Once these genes are identified, biologists collect all available information on these genes in order to understand their role or formulate new biological hypotheses or plan new experiments.
To answer the second question, one may identify relationships between genes by inferring a gene regulatory network. From a biological point of view, a gene regulatory network is composed of a set of genes where some genes called regulators control the expression of other genes. A well-known set of regulators is the set of transcription factors. These genes produce proteins which target the promoters of some other genes in order to activate or repress their expression.

These two questions raise numerous issues in statistic. In the next section, we will present a review about some statistical methods developed to analyse microarray data. These methods will address the two questions: differential analysis and network inference.

2 Methods for the differential analysis

Based on the expression level of genes in several conditions, our objective is to identify which genes show a difference in expression across the conditions. We refer to this question as “univariate differential analysis” when an analysis is performed per gene independently of the other genes.

In some context, however, one wants to consider more than one gene and identify more global changes involving a whole set of genes (typically, genes from a molecular pathway).

We refer to this new question as "multivariate differential analysis". There are many statistical tools to compare more than two conditions. It is usually easier to present the two conditions case and the extension to multi-classes is straightforward. That is why in the following, we describe the methods when measurements are performed in two conditions.

To answer both questions, a natural tool is the hypothesis testing methods which vary according to the framework. In the next section, we describe some of the most widely used methods.

2.1 Univariate analysis

Let X_j^k be a random variable for the expression of gene j in condition k. Let $(x_{1j}^k, \ldots, x_{n_k j}^k)$ be n_k observations of X_j^k. Assume that X_j^k is distributed according to a Gaussian distribution $\mathcal{N}\big((\beta^*)_j^k, ((\sigma^*)_j^k)^2\big)$ and that the X_j^k are independent for $j \in \{1, .., p\}$, where p is the total number of genes under study. For a gene j, the univariate differential analysis to compare two conditions is formulated as

$$\begin{cases} H_{0j}: & (\beta^*)_j^1 = (\beta^*)_j^2 \\ H_{1j}: & (\beta^*)_j^1 \neq (\beta^*)_j^2 \end{cases}$$

It is a two-sided test which requires three steps to be performed:

1. The definition of a test statistic that quantifies the difference of expression between the two conditions.
2. The comparison of the observed value of the test statistic to a distribution to which it fits under the null hypothesis.
3. The definition of a decision rule based on this comparison to reject the null hypothesis or not.

2.1.1 Standard Methods

2.1.1.1 Presentation of the Welch's t-test and the t-test. Assume that the number of observations is large enough for the two conditions to construct a test statistic based on the empirical variances. Let $\overline{\beta}_j^k$ be the empirical estimator of the mean of gene j in condition k

$$\overline{\beta}_j^k = \frac{\sum_{i=1}^{n_k} x_{ij}^k}{n_k}.$$

Let $(S_j^k)^2$ be the connected empirical variance

$$(S_j^k)^2 = \frac{\sum_{i=1}^{n_k} (x_{ij}^k - \overline{\beta}_j^k)^2}{n_k - 1}.$$

The Welch's t-test assumes that the variances are different in the two conditions, then the test statistic is

$$t_j = \frac{\overline{\beta}_j^1 - \overline{\beta}_j^2}{\sqrt{\frac{(S_j^1)^2}{n_1} + \frac{(S_j^2)^2}{n_2}}}.$$

Under the null hypothesis, t_j follows a Student distribution with approximated degrees of freedom given by:

$$\frac{\left(\frac{(S_j^1)^2}{n_1} + \frac{(S_j^2)^2}{n_2}\right)^2}{\frac{(S_j^1)^4}{n_1^2(n_1-1)} + \frac{(S_j^2)^4}{n_2^2(n_2-1)}}.$$

The t-test assumes that the variances are equal across the two conditions. The test statistic is then

$$t_j = \frac{\overline{\beta}_j^1 - \overline{\beta}_j^2}{S_j\sqrt{(\frac{1}{n_1} + \frac{1}{n_2})}},$$

where

$$S_j^2 = \frac{(n_1-1)(S_j^1)^2 + (n_2-1)(S_j^2)^2}{n_1 + n_2 - 2}.$$

Under the null hypothesis, t_j is distributed according to a Student distribution with $(n_1 + n_2 - 2)$ degrees of freedom.

2.1.1.2 Presentation of *limma*. The number of measurements is generally small in a microarray experiment due to the cost of producing biological replicates. Hence the t-test or the Welch's t-test are not powerful. Numerous work have been done to propose alternatives. The most popular method is $limma$ proposed by G. Smyth [1]. In fact, $limma$ could be seen as a combination approach between linear model and Bayes method. To be more specific, it is a Bayesian approach used to squeeze the standard errors in the test statistics toward a common value. The idea is to use a t-statistic with a Bayesian adjusted denominator. The statistic is defined by

$$t_j = \frac{\overline{\beta}_j^1 - \overline{\beta}_j^2}{\tilde{S}_j\sqrt{\frac{1}{n_1} + \frac{1}{n_2}}},$$

where

$$\tilde{S}_j = \frac{d_0 S_0^2 + d_j S_j^2}{d_0 + d_j},$$
$$d_j = n_1 + n_2 - 2,$$

and s_0, d_0 are defined by some prior information assumed on the variance $(\sigma_j^*)^2$:

$$\frac{1}{(\sigma_j^*)^2} \sim \frac{1}{d_0 s_0^2}\chi^2_{d_0}.$$

The estimation of s_0, d_0 is outside the scope of the introduction, see [1] for more details. However, under the null hypothesis, t_j follows a Student distribution with degree of freedom $d_j + d_0$.

2.1.1.3 Presentation of SAM. SAM is a method proposed for microarray data by Tusher *et al* [2]. In details, each gene j is assigned a statistic t_j:

$$t_j = \frac{\overline{\beta}_j^1 - \overline{\beta}_j^2}{s_0 + S_j\sqrt{(\frac{1}{n_1} + \frac{1}{n_2})}},$$

where s_0 is a constant, called fudge factor. Clearly, t_j is very similar to the statistic of the t-test. The only difference is the fudge factor. This additional term helps to compare the statistics t_j together. Indeed, without this fudge factor, t_j depends on the expression level of gene j. Since the value $S_j\sqrt{(\frac{1}{n_1} + \frac{1}{n_2})}$ is often very small for genes with low expression level and high for genes with high expression level, the objective of using s_0 is to minimize the variation of the set $\{t_1, t_2, \ldots, t_p\}$ (See [2] for the implementation).

The genes are ranked according to the SAM statistics into descending order and the observed value of the statistic is compared to the expected value. If the observed value is greater, then SAM identifies a potentially significant change in the expression between the two consider conditions.

2.1.1.4 Non-parametric approach: Wilcoxon test (paired version). Due to the small number of measurements for each gene, one alternative is to use non-parametric test. Hence it is not necessary to specify the distribution of the observations. A possibility is to use the Wilcoxon test proposed by F. Wilcoxon [3]. This requires $n_1 = n_2 = n$. The idea is that if gene j is not differently expressed between two conditions, then the difference between the pairs $(x_{1j}^1 - x_{1j}^2), \ldots, (x_{nj}^1 - x_{nj}^2)$ follows a symmetric distribution around zero. In detail, the procedure is the following:

1. Calculate n pairs $(x_{1j}^1 - x_{1j}^2), \ldots, (x_{nj}^1 - x_{nj}^2)$.
2. Remove all pairs which equal zero. Let n_r be the reduced sample size.
3. Sort the remaining pairs from smallest absolute value to largest absolute value and let R_i denote by the rank of pair i.
4. Compute the statistic $W_j = \sum_{i=1}^{n_r} sgn(x_{ij}^1 - x_{ij}^2)R_i$.

Under the null hypothesis, as n_r increases, W_j tends to a normal distribution with mean 0 and variance $\frac{n_r(n_r+1)(2n_r+1)}{6}$. For small values of n_r, the distribution of W_j is given in a

reference table [3]. Due to its non-parametric nature, the Wilcoxon test is generally less powerful than parametric approaches.

2.1.2 Global risk control in multiple testing

Given a statistical test, there are four possible results as in Table 1.1. In more details, type-I error or false positive is the rejection of H_0, while H_0 is actually true. Type-II error or false negative is the acception of H_0, while H_0 is actually false.

	H_0 Retained	H_0 Rejected	Total
H_0 True	True Negative (TN)	False Positive (FP) Type I Error	T_0
H_0 False	False Negative (FN) Type II Error	True Positive (TP)	T_1
Total	N	P	N+P = T_0 +T_1

Table 1.1: Possibilities in one hypothesis test

Whatever the test statistic used, a p-value is computed for each gene. The p-value is the probability of obtaining a test statistic equal to or more extreme than the observed result when the null hypothesis is true. Since several thousand of genes are tested, it is important to control not only the individual risk but also the global risk. There are several ways for defining the global risk among which the FWER (Family Wise Error Rate) is a natural way for controlling the global risk.

2.1.2.1 Controlling FWER: Bonferroni correction.

Among all the tests, FWER is by controlling the probability for having at least one false positive. More precisely, from Table 1.2, we have:

$$FWER = \mathbb{P}\{\mathrm{V} \geq 1\}.$$

The Bonferroni correction is a method built to control the FWER. If one has p hypotheses, then one way to maintain the FWER is to test each individual hypothesis at a statistical significance level of $1/p$ times the desired maximum overall level. In details, let $H^1, \ldots, H^p$ be a family of hypotheses and $v_1, \ldots, v_p$ their associated p-values. The Bonferroni correction control the FWER at level α if we reject all the hypotheses H_j which satisfy $v_j \leq \frac{\alpha}{p}$.

	H_0 is true	H_1 is true	Total
Reject H_0	Number of false positive (V)	Number of true positives (S)	R
Accept H_0	number of true negatives (U)	number of false negatives (T)	p-R
Total	m_0	p-m_0	p

Table 1.2: Possibilities in multiple testing

The result is clear, and based on the Boole's inequality:

$$FWER = \mathbb{P}\{\mathrm{V} \geq 1\} = \mathbb{P}\{\bigcup_{j=1}^{p}(v_j \leq \frac{\alpha}{p})\} \leq \sum_{j=1}^{p}\mathbb{P}\{v_j \leq \frac{\alpha}{p}\} \leq p\frac{\alpha}{p} = \alpha.$$

2.1.2.2 Controlling the FDR: Benjamini-Hochberg correction. Another type of global risk is the FDR (False Discovery Rate). The FDR is defined by the expected proportion of false positives among the total number of positives. From Table 1.2, we have:

$$\text{FDR} = \mathbb{E}[\frac{V}{R}].$$

Benjamini-Hochberg correction is a method built to control the FDR. In details, let $H^1, \ldots, H^p$ be a family of hypotheses and $v_1, \ldots, v_p$ their corresponding p-values. The method controls the FDR at level α with the following procedure:

- Order the p-values increasingly. Denote them $v_{(1)}, \ldots, v_{(p)}$, the corresponding hypotheses $H^{(1)}, \ldots, H^{(p)}$, where $v_{(j)}$ is the j^{th} smallest p-value.
- Find the largest k such that $v_{(k)} \leq \frac{k}{p}\alpha$.
- Reject all $H^{(1)}, \ldots, H^{(k)}$ and accept the others.

One can prove that FDR=$\mathbb{E}\{\frac{FP}{P}\} \leq \alpha$. There are many different variations of those methods such as the Benjamini-Yekutieli procedure, the Simes procedure and the Sidak procedure which are compared in a paper of Roquain [4]. The Benjamini-Hochberg correction is by far the most popular.

2.1.3 Method comparison

In [5], Jeanmougin *et al* compare empirically, $limma$, t-test, Welch's t-test, SAM, and Wilcoxon in studying gene microarray data. At a fixed controlling level (5%), they measure the FDR and the power of each test. Final results show that $limma$ performs better than the other methods. In [1], Smyth explains that as follows: on the one hand, $limma$ estimates sample variances of genes towards a pooled estimate. On the other hand, the other methods estimate the variance of each gene independently. Therefore, the estimator of $limma$ should be better when the number of observations is small. Indeed, if all variables are totally independent, $limma$ is not better than the other methods. However, the improvement of using $limma$ tells us that taking into account the correlation between variables is a way to improve different analysis, especially, when these variables are truly related.

2.2 Multivariate analysis

In some context, the detection of a change in expression level of one gene is expanded to a set of genes. We refer to this problem as a "multivariate analysis" problem. To study the problem, some methods performs directly a multivariate test on a set of genes such as Hotelling's T^2-test and Jacob *et al*'s test [6]. These approaches perform only one test with the form:

$$\begin{cases} H_0: & (\beta^*)^1 = (\beta^*)^2 \\ H_1: & (\beta^*)^1 \neq (\beta^*)^2 \end{cases}$$

where $(\beta^*)^1$ and $(\beta^*)^2$ are true values of two mean expression vectors in two conditions.

2.2.1 Standard methods

In this section, we represent Hotelling's T^2-test and Jacob *et al*'s test.

2.2.1.1 Presentation of Hotelling's T^2-test.

Denote

$$\begin{aligned} x_i^k =& (x_{i1}^k, \ldots, x_{ip}^k), \\ \overline{\beta}^k =& \frac{1}{n_k} \sum_{i=1}^{n_k} x_i^k = (\overline{\beta}_1^k, \overline{\beta}_2^k, .., \overline{\beta}_p^k). \end{aligned}$$

In the case of a homoscedastic test, we assume that $(\Sigma^*)^1 = (\Sigma^*)^2$. Hence, the pool covariance matrix is estimated by:

$$\Sigma = \frac{(n_1 - 1)\Sigma^1 + (n_2 - 1)\Sigma^2}{n_1 + n_2 - 2},$$

where Σ^1, Σ^2 are the two empirical covariance matrices:

$$\Sigma^k = \frac{1}{n_k - 1} \sum_{i=1}^{n_k} (x_i^k - \overline{\beta}^k)' (x_i^k - \overline{\beta}^k); k \in \{1, 2\}.$$

With this assumption and under the null hypothesis, the following statistic F follows a Fisher distribution:

$$F = \frac{n_1 + n_2 - p - 1}{p(n_1 + n_2 - 2)} T^2 \sim \mathcal{F}_{p, n_1 + n_2 - p - 1},$$

where

$$T^2 = (\overline{\beta}^1 - \overline{\beta}^2)^{\mathsf{T}} \Big(\Sigma \big(\frac{1}{n_1} + \frac{1}{n_2} \big) \Big)^{-1} (\overline{\beta}^1 - \overline{\beta}^2).$$

In the case of a heteroscedastic test, we assume that $(\Sigma^*)^1 \neq (\Sigma^*)^2$. Denote:

$$T^2 = (\overline{\beta}^1 - \overline{\beta}^2)^{\mathsf{T}} \Big(\frac{\Sigma^1}{n_1} + \frac{\Sigma^2}{n_2} \Big)^{-1} (\overline{\beta}^1 - \overline{\beta}^2).$$

In this case, T^2 follows the Hotelling's T-squared distribution $\mathcal{T}^2_{p,\nu}$ under the null hypothesis, where

$$\nu = \frac{tr\left[\left(\frac{\Sigma^1}{n_1}+\frac{\Sigma^2}{n_2}\right)\left(\frac{\Sigma^1}{n_1}+\frac{\Sigma^2}{n_2}\right)\right]+\left[tr\left(\frac{\Sigma^1}{n_1}+\frac{\Sigma^2}{n_2}\right)\right]^2}{\frac{tr\left[\left(\frac{\Sigma^1}{n_1}\right)\left(\frac{\Sigma^1}{n_1}\right)\right]+tr\left[\left(\frac{\Sigma^1}{n_1}\right)\right]}{n_1-1}+\frac{tr\left[\left(\frac{\Sigma^2}{n_2}\right)\left(\frac{\Sigma^2}{n_2}\right)\right]+tr\left[\left(\frac{\Sigma^2}{n_2}\right)\right]}{n_2-1}}.$$

Hotelling's T^2-test is known to perform very well in low dimensional setting. However, its power decreases quickly in high dimensional setting due to a poor estimation of the covariance matrix. The reason is that the empirical estimation of the covariance matrix is ill-conditioned and its inverse matrix is a poor estimator of Σ^{-1}. It is a central problem in GGM methods.

In the next section, we present a method trying to improve Hotelling T^2-test by reducing the number of dimensions.

2.2.1.2 Presentation of Jacob et al 's test.

Jacob *et al* [6] try to relax and make Hotelling's T^2-test more powerful for high dimensional data such as transcriptomic data by using external information.

In some specific situations, on top of the gene expression level, we may have some prior information about existing interactions between genes. Indeed, many published databases about gene networks can be used in differential analysis of genes such as Kyoto Encyclopedia of Genes and Genomes (KEGG; http://www.genome.jp/kegg) or NCI Pathway Integration Database (NCI graphs; http://pid.nci.nih.gov). Although the difference in mean expression between two groups of genes may not be totally related to the gene networks, it should not be entirely contradictory with their structure. The approach of Jacob *et al* [6] is based on this idea and can be summarized as follows:

1. Base on prior information about the gene network, the data is projected into a lower dimension sub-space. As explained below, the sub-space is chosen such that most of the distance between the expression measurements on vectors $(\beta^*)^1, (\beta^*)^2$ is preserved.
2. Apply the Hotelling's T^2-test on the projected data.

More precisely, they consider a network of p genes, represented by a graph $G = (V, E)$, with the node set V and the edge set E. Let $\phi \in \mathbb{R}^p$ denote the difference in mean expression:

$$\phi^* = (\beta^*)^1 - (\beta^*)^2.$$

Jacob *et al* [6] suppose that ϕ is coherent with the graph G, in the sense that it will minimize an energy function $E_G(\phi)$, which is defined on the graph G as follows:

$$E_G(\phi) = \sum_{j=1|d_j\neq 0}^{p}\left(\phi_j - \frac{1}{d_j}\sum_{(j,a)\in E} A_{ja}\phi_a\right)^2,$$

where

$$\begin{aligned}\phi =&(\phi_1, \ldots, \phi_p),\\ A =&(A_{ja}) \text{ is the adjacency matrix of the graph } G,\\ d_j =&\sum_{a=1}^{p} A_{ja}.\end{aligned}$$

Then, they build a space of lower dimension capturing most of the low energy functions. Assume that the sub-space has k dimensions. A base of the sub-space is made by the following procedure:

- Find the vector u_1 that minimizes the energy function:

$$u_1 = \arg\min_{\phi \in \mathbb{R}^d} E_G(\phi).$$

- For all j from 2 to k, find the vector u_j such that:

$$u_j = \begin{cases} arg\min_{\phi \in \mathbb{R}^p} E_G(\phi) \\ \text{such that } u_j \perp u_a \text{ for all } a < j. \end{cases}$$

Then the data are projected on a sub-space made by the k vectors, and the Hotelling's T^2-test is performed on the projected data. Denote $U_{[k]}$ the $p \times k$ matrix made by the k vectors $u_1, \ldots, u_k$. On this lower dimensional sub-space of dimension k, it is possible to perform a Hotelling T^2-test. To do this, we introduce the two quantities:

$$\begin{aligned}N =&\frac{n_1 + n_2 - p - 1}{(n_1 + n_2 - 2)p},\\ T_k^2 =&\frac{n_1 n_2}{n_1 + n_2}(\overline{\beta}^1 - \overline{\beta}^2)' U_{[k]}(U_{[k]}' \Sigma U_{[k]})^{-1} U_{[k]}'(\overline{\beta}^1 - \overline{\beta}^2).\end{aligned}$$

Then, the statistic NT_k^2 follows a F-distribution $\mathcal{F}_{p,n_1+n_2-p-1}$.

Intuitively, the idea is similar to Principle Components Analysis (PCA). On the one hand, PCA makes a k dimension sub-space totally based on the data. On the other hand, Jacob *et al* make a k dimensional sub-space based on some prior information about the network. Therefore, the choice of the network and the sub-space's dimension play an important role in their method.

3 Graphical model for identification of interactions

This section gives a background about graphical models and Gaussian graphical models. They are used in many different frameworks such as image analysis, physics, economics.

One of their applications is the construction of biological regulation networks. We refer to this problem as "network inference" problem.

3.1 General graphical models

A graphical model is a probabilistic model for which a graph is used to represent the conditional dependence structure between random variables. We remind that two random variables X and Y are independent conditionally on a variable Z if

$$\mathrm{P}(X \leq x, Y \leq y | Z = z) = \mathrm{P}(X \leq x | Z = z)\mathrm{P}(Y \leq y | Z = z)$$

for all x, y and z such that $\mathrm{P}(Z = z) > 0$.

In the following, I will consider the special case of random variables with a density function as I will in the end only consider Gaussian Graphical Model. In this special case, the conditional independence can also be defined as follows. Denote $f(X)$ the density function of a random variable X. Assume that all random variables X, Y and Z has a density function. Hence, two random variables X and Y are independent conditionally on a variable Z if

$$f(X, Y|Z) = f(X|Z)\ \ f(Y|Z).$$

Assume g is a graph with p nodes labelled $\{1, .., p\}$. For two nodes a, j, denote

$$a \sim j \Leftrightarrow \text{there is an edge between two nodes } a \text{ and } j \text{ in the graph } g.$$

Hence, neighbours of node j is the set $ne(j) = \{a | a \neq j; a \sim j\}$. In Figure 1.4, we have $ne(1) = \{2, 3\}$.

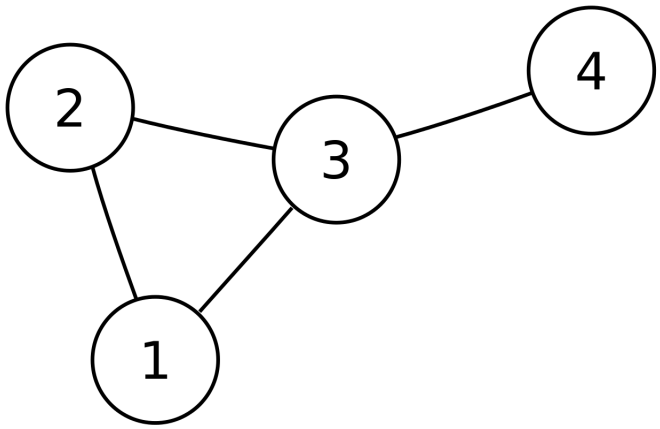


Figure 1.4: An undirected graph.

The law of one random variable $X = (X_1, .., X_p)$ is a graphical model according to the graph g if for all node j, X_j is independent of $\{X_b, b \notin ne(j) \bigcup \{j\}\}$. We denote this relation

as $\mathfrak{L}(X) \sim g$. If the law of the random variable X has a density f, then f satisfies:

$$f(X_1, \ldots, X_p) = \prod_{j=1}^{p} f(X_j | X_{ne(j)}),$$

where $f(X_j|X_{ne(j)})$ is the conditional density of X_j given $X_{ne(j)}$.

3.2 Gaussian graphical model (GGM)

Let $\mathcal{P} := \{1, ..., p\}$ be the set of nodes and $X = (X_1, ..., X_p)$ be a random variable describing a signal over this set. In the standard framework of Gaussian graphical model , X follows a multivariate Gaussian distribution with unknown covariance matrix $\Sigma^* = (\Sigma^*_{ij})_{(i,j)\in\mathcal{P}^2}$:

$$X \sim \mathcal{N}(0_p, \Sigma^*).$$

The covariance matrix $\Sigma^* = \mathbb{E}(XX^{\mathrm{T}})$ is a positive definite symmetric matrix.
A GGM can be associated with the dependency structure between the p variables $\{X_1, ..., X_p\}$ of a Gaussian random vector X. Conversely, the minimum graph g of X could be read directly from the inverse of the covariance matrix $\Theta^* := (\Sigma^*)^{-1}$, also called precision or concentration matrix. More precisely, the graph of X is defined by the symmetric relation:

$$\text{Two nodes } a \text{ and } j \text{ are linked} \Leftrightarrow \Theta^*_{aj} \neq 0.$$

3.3 Inference in GGM

One of the most usual goal of using GGM is to recover an interaction network between variables. In the framework of GGM, the network could be read directly from the adjacency matrix $\mathcal{A}$ of the precision matrix which is defined by:

$$\mathcal{A}_{ij} = \begin{cases} 0 \text{ if } \Theta^*_{aj} = 0 \\ 1 \text{ if } \Theta^*_{aj} \neq 0 \end{cases}$$

Therefore, the goal becomes inferring the precision matrix Θ^*.

In classical contexts where the number of observations n is much larger than the number of variables p, the most usual way to infer the parameters is to use maximum likelihood method. Assume that we have a sample $\{\mathbf{X}^1, ..., \mathbf{X}^n\}$ composed of n i.i.d. replications of X. Denote $\mathbf{X} = (x_{ij})$ as the data matrix of dimension $n \times p$. The row i of matrix $\mathbf{X}$ is $\mathbf{X}^i$. With the empirical covariance matrix $\mathbf{S}_n = \mathbf{X}^T\mathbf{X}/n$, the maximal likelihood estimator (MLE) is:

$$\hat{\Theta}^{\text{mle}} = arg \min_{\Theta\in\mathcal{S}_p^+} L(\Theta) = arg \min_{\Theta\in\mathcal{S}_p^+} -\log det(\Theta) + Trace[\Theta\mathbf{S}_n],$$

where $\mathcal{S}_p^+$ is the set of $p \times p$ positive definite matrices. When $n \geq p$, this problem has a unique solution which is $(\mathbf{S}_n)^{-1}$.

However, MLE does not work well in the high-dimensional setting. First, MLE gives us more than one solution when $n \ll p$. Second, in some specific contexts, people usually assume that the true set of direct relationships between variables is small. In other words, the true interaction network of variables is sparse. However, MLE always gives back complete networks where all variables are connected. Therefore, using MLE is not a good choice in this specific context. To resolve this problem, many sparse methods have been introduced for GGM.

3.3.1 Graphical-Lasso (Glasso)

Yuan and Lin [7]; Barnejee *et al* [8] propose the graphical-Lasso estimator:

$$\hat{\Theta}^{Glasso} = arg \min_{\Theta \in \mathcal{S}_p^+} E^{Glasso} = arg \min_{\Theta \in \mathcal{S}_p^+} L(\Theta) \ + \lambda_1 \sum_{1 \leq j < a \leq p} |\Theta_{ja}|.$$

The tuning parameter λ_1 controls the sparsity of $\hat{\Theta}^{Glasso}$. The bigger λ_1, the more sparse the matrix $\hat{\Theta}^{Glasso}$. When λ_1 tends to zero, $\hat{\Theta}^{Glasso}$ tends to the MLE estimator $\hat{\Theta}^{\text{mle}}$. The graphical-Lasso estimator is always symmetric, positive definite which are nice and desirable properties of a precision matrix estimator.

3.3.2 Neighborhood selection (NS)

Meinshausen and Buhlmann [9] propose a method to fill the gap between GGM and linear regression model. They made the important remark that, in the GGM framework with centered data, one has:

$$X_j = \sum_{a \neq j} (\theta^*)_{ja} X_a + \epsilon_j, \tag{1.1}$$

where $\theta^*_{ja} = -\Theta^*_{ja}/\Theta^*_{jj}$ and $\epsilon_j \sim \mathcal{N}(0, 1/\Theta^*_{jj})$. By the definition of the adjacency matrix , we can recover the adjacency matrix of Θ^* from entries $(\theta^*)_{ja}$.
Hence, For each random variable X_j, its set of neighbors is estimated by the support of vector:

$$\hat{\theta}_j^{NS} = arg \min_{\theta_j \in \mathbb{R}^{p-1}} \|\mathbf{X}_j - \mathbf{X}_{\setminus j}\theta_j\|^2 \ + \ \lambda_1 \|\theta_j\|_1.$$

Indeed, $\hat{\theta}_j^{NS}$ is the estimator of column j of matrix θ^* deprived of its j^{th} entry. A full estimator of matrix θ^* is then the combination of p optimization problems for p random variables X_j.

Empirically, NS is better than $Glasso$ at edge detection in some types of data [10] [11]. In addition, instead of solving a big $lasso$ problem as $Glasso$, NS boils down to p smaller

$lasso$ problems. This is the reason why NS is better than $Glasso$ from a computational perspective. However, the NS estimator cannot guarantee the symmetric property of the network. Meinshausen and Buhlmann suggest AND or OR rules which are both consistent when n goes to infinity to solve the problem.

3.3.3 Sparse Partial correlation estimation (SPACE)

A variation of neighbourhood selection is SPACE proposed by Peng *et al* [12]. Note that, the partial correlation $(\rho^*)_{ij} = -\Theta^*_{ij}/\sqrt{\Theta^*_{ii}\Theta^*_{jj}}$. From equation 2.1, one has:

$$X_j = \sum_{a \neq j} (\rho^*)_{ja} \sqrt{\frac{\Theta^*_{aa}}{\Theta^*_{jj}}} X_a + \epsilon_j.$$

Hence, they would like to estimate ρ^* and diagonal elements of matrices Θ^* by minimizing:

$$(\hat{\rho}, \hat{\Theta})^{space} = arg \min_{\substack{\theta \in \mathcal{S}_p^+ \\ \Theta_{11}, \Theta_{22}, .., \Theta_{pp}}} L^{space}(\rho, \Theta_{11}, \Theta_{22}, .., \Theta_{pp}) \; + \; \lambda_1 \|\rho\|_1,$$

where

$$L^{space}(\rho, \Theta_{11}, \Theta_{22}, .., \Theta_{pp}) = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{p} \left(x_{ij} - \sum_{a=1}^{p} \rho_{ja} \sqrt{\frac{\Theta_{aa}}{\Theta_{jj}}} x_{ia} \right)^2.$$

Hence, the network is inferred from the support of ρ. In fact, the SPACE estimator could be seen as a hybrid version of Glasso and NS estimators. It has the form of a linear regression problem as NS. However, it guarantees a symmetric and positive definite estimator for the covariance matrix. Moreover, it takes into account an estimation of the diagonal elements of precision matrix Θ^* as Glasso.

Although there are several differences between Glasso, NS and SPACE, the three methods are in a same framework of multivariate normal distribution.

3.4 Inference of a multiple GGM

In more complicated situations with multiple tasks or conditions, we may have multiple networks, where each of them is associated to one task. In some situation, if we consider that data in different tasks are totally independent, the associated networks will be independent and usually much different. In that case, multi-task is not different from single task. However, in most of the cases, there is always some kind of connections between data and networks between tasks. For instance, we have data about expression level of a set of genes in different experimental conditions. Even if the conditions are very different the gene networks most probably share similarities. This is the reason why, beside of many methods for GGM in single task framework, we also have many other methods for GGM in the multi-task framework.

3.4.1 Definition of a multi-task GGM

The framework of GGM in multi-task for centered data has been described by Chiquet *et al* [13], Danaher *et al* [14], Mohan *et al* [15]. In this section, we describe again this framework.

Sometimes, data concerning the same variables could be collected from different sources. For instance, gene expression levels are observed in different experimental conditions. Each condition is equivalent to one task. Assume that we have K distinct tasks. Each observation in a task k is the measurement of a p-dimensional Gaussian random vector X^k. Suppose that all observations are independent. Moreover, if they are in the same task, they share the same distribution. In other words,

$$X^k = (X_1^k, X_2^k, .., X_p^k) \sim \mathcal{N}\Big(0_p, (\Sigma^*)^k\Big),$$

where $(\beta^*)^k \in \mathbb{R}^p$ is the mean vector and $(\Sigma^*)^k$ is the $p \times p$ covariance matrix corresponding to task k. With K tasks, one has K distinct matrices $(\Sigma^*)^k$; $k \in \{1, .., K\}$. We denote:

$$\Sigma^* = \Big((\Sigma^*)^1, (\Sigma^*)^2, .., (\Sigma^*)^K\Big).$$

Each observation i of a random variable X^k has the form

$$x_i^k = (x_{i1}^k, x_{i2}^k, .., x_{ip}^k) \in \mathbb{R}^p,$$

where $i \in \{1, .., n_k\}$, and n_k states for the number of observations in the task k. Hence, data in task k is a $n_k \times p$ matrix denoted by:

$$\mathbf{X}^k = \begin{pmatrix} x_1^k \\ .. \\ x_{n_k}^k \end{pmatrix},$$

and the total number of observations is $n = n_1 + .. + n_K$.

In this multi-task Gaussian framework, from the given data $\mathbf{X} = (\mathbf{X}^1, .., \mathbf{X}^K)$, we aim to infer K sparse graphs. Each graph corresponds to the conditional dependencies among p variables of vector X^k in task k. Let $\Theta^* = \Big((\Theta^*)^1, .., (\Theta^*)^K\Big) = \Big(\big((\Sigma^*)^1\big)^{-1}, .., \big((\Sigma^*)^K\big)^{-1}\Big)$ be the precision matrices vector; for each task k, the non-zero entries of $(\Theta^*)_{aj}^k$ describes a conditional dependency between the variables X_a^k and X_j^k, thus defining the graph $\mathcal{G}^k$ of conditional dependencies of X^k. In this framework, the maximal likelihood estimator is

$$(\hat{\beta}^{\text{mle}}, \hat{\Theta}^{\text{mle}}) = arg \max_{\substack{\Theta^k \in \mathcal{S}_p^+ \\ \beta^k \in \mathbb{R}^p}} \sum_{k=1}^{K} \Big(\log det(\Theta^k) - Trace[\Theta^k \mathbf{S}_n^k] \Big), \tag{1.2}$$

where $\mathbf{S}_n^k$ is the empirical covariance matrix function:

$$\mathbf{S}_n^k = \frac{1}{n_k}(\mathbf{X}^k)^T\mathbf{X}^k. \tag{1.3}$$

The equation 1.2 corresponds to the case where the inference is performed independently within each condition.

3.4.2 Sparse methods for GGM in multi-task framework

In this section, we describe sparse methods in the multi-task framework. First of all, all methods developed for single task framework could be used in the multi-task framework. In the case where data in different tasks are independent, using these methods is straightforward.

However, in many cases, data in different tasks are not independent, for instance, gene expression data in multiple tasks may share a very similar regulatory network. Therefore, if we use data between tasks independently, we may lose information. This is a serious problem especially in the high-dimensional setting. In this section, we make a brief overview of the $Glasso$ and NS in this new framework. Then, we describe some more sophisticated methods which share information between tasks.

3.4.2.1 Glasso. In the multi-task framework, the $Glasso$ estimator is:

$$\hat{\Theta}^{Lasso} = arg \max_{\Theta^k \in \mathcal{S}_p^+} E^{GLasso} = arg \max_{\Theta^k \in \mathcal{S}_p^+} L^{Lasso}(\Theta) \ + \ \Omega_1^{Lasso}(\Theta),$$

where

$$L^{Lasso}(\Theta) = \sum_{k=1}^{K} \Big(\log det(\Theta^k) - Trace\{\Theta^k \mathbf{S}_n^k\} \Big),$$

$$\Omega_1^{Lasso}(\Theta) = -\sum_{k=1}^{K} \lambda_{1k} \|\Theta^k\|_1 = -\sum_{k=1}^{K} \lambda_{1k} \sum_{1 \le j < a \le p} |\Theta_{ja}^k|,$$

3.4.2.2 Neighborhood selection. Similar to the single GGM framework with centered data, in the multi-task case we have:

$$X_j^k = \sum_{a \neq j} (\theta^*)_{ja}^k X_a^k + \epsilon_j^k. \tag{1.4}$$

Then we estimate θ^* by:

$$\hat{\theta}^{NS} = arg \min_{\theta \in \mathcal{S}_p^+} E^{NS} = arg \min_{\theta \in \mathcal{S}_p^+} L^{NS}(\theta) \ + \ \Omega_1^{NS}(\theta),$$

where

$$L^{NS}(\theta) = \frac{1}{2}\sum_{k=1}^{K}\sum_{i=1}^{n_k}\sum_{j=1}^{p}\Big(x_{ij}^k - \sum_{a\neq j}\theta_{ja}^k x_{ia}^k\Big)^2,$$

$$\Omega_1^{NS}(\theta) = \sum_{k=1}^{K}\lambda_{1k}\|\theta^k\|_1.$$

3.4.2.3 Group Lasso. Based on [16], the $Group-Lasso$ estimator in the multi-task is:

$$\hat{\Theta}^{group-lasso} = arg\max_{\Theta^k\in\mathcal{S}_p^+} E^{group-lasso} = arg\max_{\Theta^k\in\mathcal{S}_p^+} L^{group-lasso}(\Theta) + \Omega_1^{group-lasso}(\Theta),$$

where

$$L^{group-lasso}(\Theta) = \sum_{k=1}^{K}\Big(\log det(\Theta^k) - Trace\{\Theta^k\mathbf{S}_n^k\}\Big),$$

$$\Omega_1^{group-lasso}(\Theta) = -\lambda_1\sum_{a\neq j}\Big(\sum_{k=1}^{K}(\Theta_{aj}^k)^2\Big)^{1/2}.$$

The $Group-lasso$ is a mixed norm that encourages sparse solutions with respect to groups. Each group contains K entries such as $\{\Theta_{aj}^1,..,\Theta_{aj}^K\}$. Either estimators of all entries in one group are zero or all of them are non-zero.

The geometric interpretation of $Group-lasso$ is shown in Figure 1.5 [3].

3.4.2.4 Cooperative Lasso. Similar to the $Group-Lasso$ estimator, we have:

$$\hat{\Theta}^{coop-lasso} = arg\max_{\Theta^k\in\mathcal{S}_p^+} E^{coop-lasso} = arg\max_{\Theta^k\in\mathcal{S}_p^+} L^{coop-lasso}(\Theta) + \Omega_1^{coop-lasso}(\Theta),$$

where

$$L^{coop-lasso}(\Theta) = L^{group-lasso}(\Theta),$$

$$\Omega_1^{coop-lasso}(\Theta) = -\lambda_1\sum_{a\neq j}\Big(\sum_{k=1}^{K}(\Theta_{aj}^k)^2\Big)_+^{1/2} - \lambda_1\sum_{a\neq j}\Big(\sum_{k=1}^{K}(-\Theta_{aj}^k)_+^2\Big)^{1/2},$$

where $(u)_+ = max(u,0)$. The $Cooperative-lasso$ not only encourages sparse solutions with respect to groups as group lasso, but also encourage the similar sign of elements in a same group. Moreover, to enable the inference of different networks, let say (k,c), we must have some (a,j) such that $\Theta_{aj}^k \neq \Theta_{aj}^c$. This event occurs with probability zero with the group-lasso [7]. $Cooperative-lasso$ cures the problem by either shrink all the positive (negative) elements in one group to zero, and keep the other negative (positive) elements. The geometric interpretation of $Cooperative-lasso$ is showed in Figure 1.6[4]

[3] These figures are taken from Chiquet *et al* [13]

[4] These figures are taken from Chiquet *et al* [13].

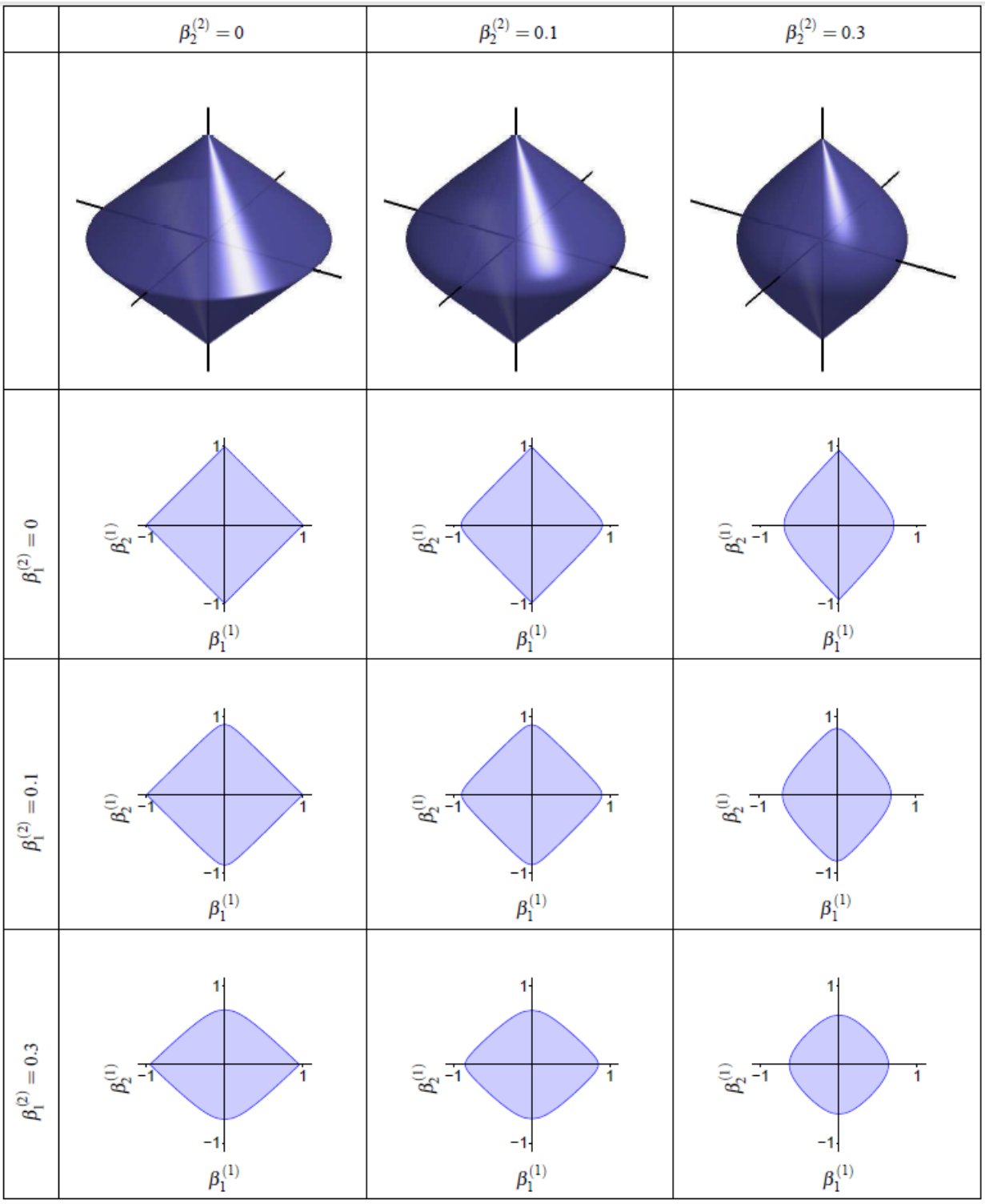


Figure 1.5: Representations of the admissible set for the Group-LASSO penalty for a problem with two tasks and two features. Top row: cuts of the unit ball through $(\beta_1^{(1)}, \beta_1^{(2)}, \beta_2^{(1)})$ for various values of $\beta_2^{(2)}$, where $\beta_1^{(1)}, \beta_1^{(2)}$ span the horizontal plane, and $\beta_2^{(1)}$ is on the vertical axis; bottom rows: cuts through $(\beta_1^{(1)}, \beta_2^{(1)})$ for various values of $\beta_1^{(2)}$ and $\beta_2^{(2)}$.

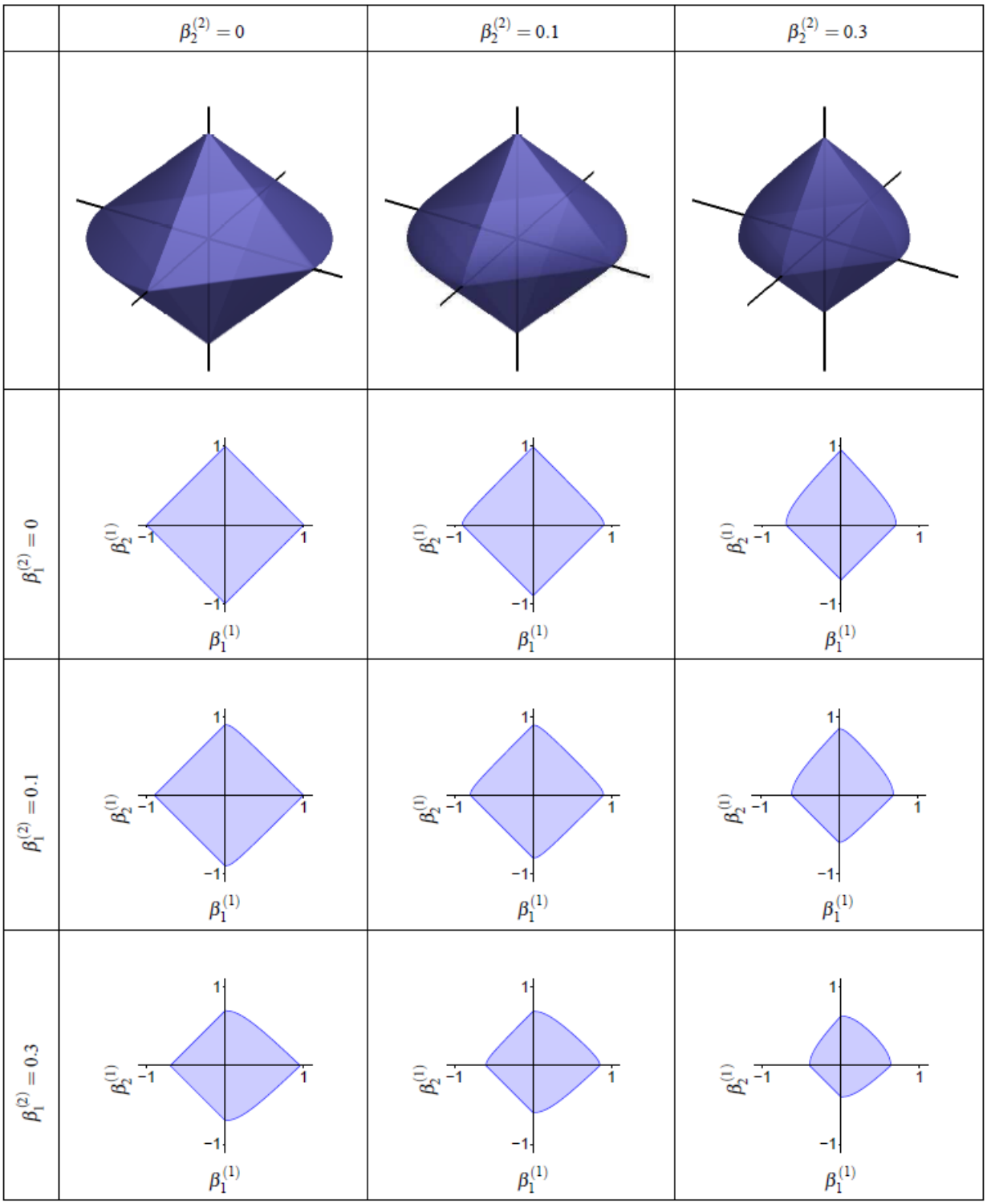


Figure 1.6: Representations of the admissible set for the Cooperative-LASSO penalty for a problem with two tasks and two features. Top row: cuts of the unit ball through $(\beta_1^{(1)}, \beta_1^{(2)}, \beta_2^{(1)})$ for various values of $\beta_2^{(2)}$, where $\beta_1^{(1)}, \beta_1^{(2)}$ span the horizontal plane, and $\beta_2^{(1)}$ is on the vertical axis; bottom rows: cuts through $(\beta_1^{(1)}, \beta_2^{(1)})$ for various values of $\beta_1^{(2)}$ and $\beta_2^{(2)}$.

3.5 Remarks about the implementation of the methods

Beyond the performance of the methods in terms of estimation, one important aspect of these methods is their implementation. In this section, we describe several packages used to solve these problems. I categorize these packages into two categories: general Lasso estimation and GGM estimation.

3.5.1 Packages for general lasso estimation

1. $glmnet$
   - Pros:
     - Fast computation,
     - Possible to put weights on each estimated parameter,
     - Provides the solution path, where the solution path is the value of Lasso estimator for each value of the tuning parameters.
   - Cons:
     - Single task package,
     - Does not compute directly the GGM but a general form of the $Lasso$.
2. $genlasso$
   - Pros:
     - Provides the solution path,
     - Takes into account complex design matrix,
     - Possible to put weights on each estimated parameter.
   - Cons:
     - Single task package,
     - Numerically unstable and slow computation,
     - Does not directly compute the GGM.
3. $elasticnet$
   - Pros:
     - Fast computation,
     - Provides the solution path.
   - Cons:
     - Single task package,
     - Does not compute directly the GGM,
     - Not possible to put weights on each estimated parameter.

3.5.2 Packages for graphical Gaussian models

1. *glasso*
   - Pros:
     - Estimates parameters of Lasso and NS directly on GGM,
     - Possible to put weights on each estimated parameter.
   - Cons:
     - Single task package,
     - Does not provide the solution path.
2. *huge*
   - Pros:
     - Fast computation,
     - Computes the GGM directly.
   - Cons:
     - Single task package,
     - Not possible to put weights on each estimated parameter.
     - Does not provide the solution path.
3. *space*
   - Pros:
     - Fast computation.
   - Cons:
     - Single task package,
     - Dose not provide the solution path,
     - Not possible to put weights on each estimated parameter,
4. *simone*
   - Pros:
     - Multi-task package,
     - Computes the GGM directly,
     - Provides the solution path.
   - Cons:
     - Slow computation.

4 Fused ANOVA method - The beginning idea of the thesis

As we described above, the two questions of differential analysis and network inference are solved independently. However, in many specific contexts, sharing information between the two problems actually could improve them both. For instance, it is well-known that the mean expression level of genes and the genes regulatory network have a strong relation. The remaining challenge is how we can answer the two questions simultaneously.

The idea of this thesis started from a paper of Chiquet *et al* [17]. In this paper, they rewrite hypothesis test methods under a regression form. Let us first give more details about their method which is called Fused ANOVA.

In the method, the framework is slightly different compared to the original framework of GGM. In fact, it is a classical one-way ANOVA setup:

$$Y_{ik} = \beta_k + \epsilon_{ik}, \quad \epsilon_{ik} \sim \mathcal{N}(0, \sigma^2_{\underline{i}k}),$$

where Y_{ik} is the intensity of a continuous random variable for samples i in condition k, and β_k is the mean parameter of condition k. Denote by K the number of conditions, n_k the number of sample in condition k and $n = \sum_k n_k$ the total sample size.

Their goal is to test the differences between β_k from Y_{ik}. They rewrite the problem as a minimization problem on the objective function :

$$E(\beta) = \sum_{k=1}^{K} \sum_{i=1}^{n_k} (Y_{ik} - \beta_k)^2 + \lambda \sum_{k,l} w_{kl} |\beta_k - \beta_l|,$$

where $\beta = (\beta_1, ..., \beta_K)$ and the weights w_{kl} may be interpreted as a prior on the differences between the means of two conditions. This problem encourages the absolute differences between β_k to be small: the larger the λ is, the smaller the differences will be. But, what is the link between this optimization problem and the hypothesis test method? The answer stands at the "fusion time" of two mean parameters between two conditions. As we know, when λ increases, mean parameter is getting closer, and the "first time" when two mean parameters β_k and β_l are fused is:

$$\lambda_{kl} = \frac{Y_{\cdot}^{(k)} - Y_{\cdot}^{(l)}}{w_{kl}(1/n_1 + 1/n_2)},$$

where $Y_{\cdot}^{(k)} = \sum_{i=1}^{n_k} Y_{ik}^{(k)}/n_k$ and $Y_{\cdot}^{(l)} = \sum_{i=1}^{n_l} Y_{il}^{(l)}/n_l$.

If we choose $w_{kl} = \frac{1}{\sqrt{1/n_k + 1/n_l}}$, we recognize the statistic of the t-test. Similarly, if $w_{kl} = \sqrt{\frac{s_k^2 + s_l^2}{1/n_k + 1/n_l}}$, where s_k^2 and s_l^2 are empirical variances of the data in tasks k and l, we can recover the Welch's t-test. By changing w_{kl}, we can recover many test statistics such as Welch's t-test, ANOVA. In short, finding the fusion time of the optimization problem equals to finding the statistic test methods. There are other choices for the weights such as:

- Trivial weights: $w_{kl} = 1$,
- Default weights: $w_{kl} = n_k n_l$,
- Exponential weights: $w_{kl} = n_k n_l exp(-\gamma(Y_{\cdot}^{(k)} - Y_{\cdot}^{(l)})^2)$,
- Adaptive weights: $w_{kl} = n_k n_l (Y_{\cdot}^{(k)} - Y_{\cdot}^{(l)})^{-\gamma}$.

However, they are used for different purposes (e.g. better visualization, oracle properties). Therefore, we will not mention them here in details.

Our idea is to propose a unified framework in the context of multivariate Gaussian distributions. Similar to Fused ANOVA and penalised regression for GGM, our problem will be under the form of one optimization problem and can be solved by an alternate convex search algorithm.

In Chapter 2, we present our model and some theoretical results. In Chapter 3, using numerical simulations, we compare our model to standard methods which are used to answer differential analysis and network inference. In Chapter 4, we apply our model to two real datasets. Finally, in Chapter 5, we give our conclusion and the perspective of this thesis.

Chapter 2

Statistical model for coupling network inference and differential analysis

Suppose that we have a dataset with gene expression levels from multiple experimental conditions with several replicates per condition. This dataset could be used to treat either a problem of multiple network inference or a problem of differential analysis. In this chapter, we introduce a statistical model to treat these two tasks jointly. After describing the model, we develop a penalized strategy for the inference, accompanied with an effective algorithm. We also prove the consistency of our estimator and show how it can be used to answer two interesting problems of genomic data analysis introduced in Chapter 1.

1 Model description

Our model is built in a multi-task framework of GGM for uncentered data. More precisely, we consider the setting in which one has access to observations from K distinct tasks. Each observation in a task k is the measurement of a p-dimensional Gaussian random vector X^k. We suppose that, all observations are independent. Moreover, if they are in the same task, they have identical distribution. We have:

$$X^k = (X_1^k, X_2^k, .., X_p^k) \sim \mathcal{N}\Big((\beta^*)^k, (\Sigma^*)^k\Big),$$

where $(\beta^*)^k \in \mathbb{R}^p$ is the mean vector and $(\Sigma^*)^k$ is the $p \times p$ covariance matrix corresponding to task k. With K tasks, one has K distinct couples $(\beta^*)^k$ and $(\Sigma^*)^k$; $k \in \{1, .., K\}$. We denote:

$$\beta^* = \Big((\beta^*)^1, (\beta^*)^2, .., (\beta^*)^K\Big),$$
$$\Sigma^* = \Big((\Sigma^*)^1, (\Sigma^*)^2, .., (\Sigma^*)^K\Big).$$

Each observation i of a random variable X^k is written:

$$x_i^k = (x_{i1}^k, x_{i2}^k, .., x_{ip}^k) \in \mathbb{R}^p,$$

where $i \in \{1, .., n_k\}$, and n_k states for the number of observations in task k. Hence, data in task k is a $n_k \times p$ matrix denoted by:

$$\mathbf{X}^k = \begin{pmatrix} x_1^k \\ .. \\ x_{n_k}^k \end{pmatrix}.$$

The total data is $\mathbf{X} = (\mathbf{X}^1, .., \mathbf{X}^K)$ and the total number of observations is $n = n_1 + .. + n_K$.

Following these notations, the expression level of gene j in condition k is described by the following bilinear model, arising from Gaussian vector analysis:

$$X_j^k = (\beta_j^*)^k + \sum_{a \neq j} (\theta^*)_{ja}^k \Big(X_a^k - (\beta_a^*)^k \Big) + \epsilon_j^k, \tag{2.1}$$

where $(\theta^*)_{ja}^k = -\Big((\Sigma^*)^{-1}\Big)_{ja}^k / \Big((\Sigma^*)^{-1}\Big)_{jj}^k$ and $\epsilon_j^k \sim \mathcal{N}(0, \sigma^2)$, where σ is a positive unknown constant.

In words, the expression level of gene j in condition k (or equivalently, task k) mainly depends on two factors: first, its mean expression in the current condition, namely $(\beta^*)^k$; second, its relationships with the other genes in the same condition, namely $\sum_{a \neq j} (\theta^*)_{ja}^k \Big(X_a^k - (\beta_a^*)^k \Big)$.

There are several motivations for this model. The multivariate Gaussian setting is flexible enough to catch the most important trends of gene expression data. Moreover, the vectors of means $(\beta^*)^k$ are directly interpretable in terms of expression and the matrices $(\theta^*)^k$ are interpretable in terms of interaction. Beside, most of the considered data are actually log ratio data, for which the normality assumption holds.

This model has many advantages. On the bright side, it is pretty simple, easy to study and inherits many results from previous works due to its similarity with the linear model. However, the bilinear form of our model requires additional developments both in terms of computation and estimation.

Let us now specify some additional notations for the purpose of statistical inference: we consider n_k observations of the vector $X^k = (X_1^k, \ldots, X_p^k)^T \in \mathcal{R}^p$ in condition k. Let $x_i^k = (x_{i1}^k, x_{i2}^k, .., x_{ip}^k)$ be the i^{th} observation. Hence, the data related to the condition k is a $n_k \times p$ matrix

$$\mathbf{X}^k = \begin{pmatrix} x_1^k \\ .. \\ x_{n_k}^k \end{pmatrix}.$$

The total number of observations is $n = \sum_{k=1}^K n_k$ and the whole $n \times p$ data matrix is

$\mathbf{X} = (\mathbf{X}^1, \ldots, \mathbf{X}^K)$. Note that, from equation 2.1, we have

$$x_{ij}^k = (\beta_j^*)^k + \sum_{a \neq j} (\theta^*)_{ja}^k \left(x_{ia}^k - (\beta_a^*)^k \right) + \epsilon_{ij}^k. \tag{2.2}$$

In this setting, we address two biological-oriented problems by using estimated parameters of the model.

1. For network inference, we aim to infer K sparse graphs. In each task k, the inferred graph corresponds to the conditional dependencies among the p variables of the vector X^k. Namely, non-zero entries of $(\theta^*)_{ja}^k$ indicate the conditional dependency between variables X_j^k and X_a^k, and thus define the graph $\mathcal{G}^k$ of conditional dependencies between a pair of variables in X^k. Hence, the K graphs describe K interaction networks between genes in K different biological conditions.

2. For differential analysis, variables $(X_j^1, .., X_j^K)$ are gene expression levels of gene j in K different biological conditions. We aim to detect changes in the mean expression of each gene in the K conditions. In terms of parameters of the model, this can be measured by detecting differences between the K vectors $\{(\beta^*)^1, \ldots, (\beta^*)^K\}$.

The next section introduces a strategy for estimating the parameters of the model in the high dimensional setting.

2 Parameters estimation

The general idea to estimate $(\beta^*)^k$ and $(\theta^*)^k$ is to optimize a criteria composed by a loss function plus some penalties reflecting our priors on the parameters. Possible choices for the penalties are numerous and were introduced in Chapter 1, Section 3.3 and Section 3.4. In the data that we are considering, the interaction networks are sparse. Moreover, many of the vectors $\{(\beta^*)^1, \ldots, (\beta^*)^K\}$ share similar elements. Therefore, we construct a penalized criterion that encourages sparsity of the concentration matrix and similarities between mean vectors. Our estimators are defined as the argmin of the following objective function $\{(\hat{\beta}_{\lambda_2}^k, \hat{\theta}_{\lambda_1}^k)\}_{k=1,\ldots,K} = \arg\min_{\beta^k, \theta^k} E(\beta, \theta\ , \lambda_1, \lambda_2; \mathbf{X})$ with

$$\begin{aligned} E(\beta, \theta\ , \lambda_1, \lambda_2; \mathbf{X}) =& \frac{1}{2} \sum_{k=1}^{K} \sum_{i=1}^{n_k} \sum_{j=1}^{p} \left(x_{ij}^k - \beta_j^k + \sum_{a \neq j} \theta_{ja}^k \beta_j^k - \sum_{a \neq j} \theta_{ja}^k x_{ia}^k \right)^2 + \\ & \lambda_1 \sum_{k=1}^{K} \sum_{k=1}^{p} \sum_{a \neq j} \omega_{ja}^k |\theta_{ja}^k| + \lambda_2 \sum_{\substack{k,l=1 \\ k<l}}^{K} \sum_{j=1}^{p} \varrho_j^{kl} |\beta_j^k - \beta_j^l|, \end{aligned} \tag{2.3}$$

where λ_1, λ_2 are well chosen tuning parameters corresponding to the two penalty parts, and $\omega_{ja}^k, \varrho_j^{kl}$ are given weights. From a biological point of view, the first penalty controls the sparsity of the interaction networks, while the second penalty controls the similarity of the gene expression levels between the different biological conditions.

If not specified, we use trivial weights where all weights $\omega_{ja}^k = 1$ and all weights $\varrho_j^{kl} = 1$ except for some particular scenarios. From a computation point of view, with the trivial weights, the objective function is simpler and easier to manipulate. In particular, it is easier to minimize the objective function in that case. However, Zou [18] showed that, with these trivial weights, Lasso and fused-Lasso estimators could give back inconsistent estimators. Therefore, from the theoretical point of view, the choice of the weights is important and may have some practical consequences. We will come back to this problem in the theoretical part of this chapter (Section 3).

2.1 Optimization

In this paragraph, we discuss the optimization of Problem (2.3). Note that for a fixed couple (λ_1, λ_2), the function $E(\beta, \theta\ , \lambda_1, \lambda_2; \mathbf{X})$ is biconvex (that is, convex in β for fixed θ and convex in θ for fixed β). Hence, (β^*, θ^*) can be estimated by an alternate convex search algorithm (ACS) [19]. The general idea of the algorithm can be summarized as follows. We fix either all θ^k or all β^k, then estimate the others and iterate the whole process. The loop continues until the two sequences of estimators converge, with the stopping condition defined by

$$\sum_{k=1}^{K}\sum_{j=1}^{p}\sum_{a=1}^{p}\left|(\hat{\theta}_{ja}^k)^{(t+1)} - (\hat{\theta}_{ja}^k)^{(t)}\right| + \sum_{k=1}^{K}\sum_{j=1}^{p}\left|(\hat{\beta}_{j}^k)^{(t+1)} - (\hat{\beta}_{j}^k)^{(t)}\right| < \epsilon^0,$$

where ϵ^0 is a given small number, and $\left((\hat{\theta}_{ja}^k)^{(t)}, (\hat{\beta}_{j}^k)^{(t)}\right)$ are our estimators at the t^{th} step. The pseudo code of the algorithm is the following:

ACS - Algorithm

Initial values: $(\hat{\beta}_j^k)^{(0)} = (\sum_{i=1}^{n_k} x_{ij})/n_k$ for all $j \in \{1, .., p\}$ and $k \in \{1, .., K\}$.

While (stopping criteria not met) **do**

- Fixed all $\hat{\beta}^{(t)}$, estimate $\hat{\theta}^{(t)} = \arg\min_{\theta} E(\theta; \hat{\beta}^{(t)})$.
- Fixed all $\hat{\theta}^{(t)}$, estimate $\hat{\beta}^{(t+1)} = \arg\min_{\beta} E(\beta; \hat{\theta}^{(t)})$.

end

2.1.1 Algorithm and discussion

In 2007, Jochen et al [26] showed the convergence of the ACS algorithm. However, they also point out that ACS may reach a local optimum but not a global optimal value. To the best of our knowledge, there is one algorithm to find the global optimal value called global optimization algorithm (GOP). The algorithm is developed by Floudas and Visweswaran [27]. However, it is almost hopeless to use it in high-dimensional data due to its running time. Several improvement of GOP have been considered involving the structure of the given biconvex problem, but there are many remaining challenges. Therefore, in this thesis we use the ACS algorithm, and we should be aware that the output values of ACS may be not the global optimal values.

We now explain in more details the estimations of β^* and θ^*.

2.1.2 Estimation of θ^*

For fixed values of $(\lambda_1, \lambda_2, \beta)$, our objective function is equivalent to

$$E_1(\theta) = \frac{1}{2}\sum_{k=1}^{K}\sum_{i=1}^{n_k}\sum_{j=1}^{p}\Big[\big(x_{ij}^k - \beta_j^k\big) - \sum_{a\neq j}\theta_{ja}^k\big(x_{ia}^k - \beta_a^k\big)\Big]^2 + \lambda_1\sum_{k=1}^{K}\sum_{k=1}^{p}\sum_{a\neq j}\omega_{ja}^k|\theta_{ja}^k|.$$

In this case, minimizing $E_1(\theta)$ is a Lasso problem [16]. The default form of the weighted Lasso problem is

$$\arg\min_{\tilde{\Theta}} \frac{1}{2}\|\tilde{y} - \mathcal{X}\tilde{\Theta}\|_2^2 + \lambda\|\tilde{\omega}\circ\tilde{\Theta}\|_1,$$

where $\tilde{y}$ is an observation vector, $\mathcal{X}$ is a given matrix, λ is the tuning parameter, $\tilde{\Theta}$ is the vector of variables, and "$\circ$" is the element-by-element multiplication. Our purpose is to explain to the reader what are the $\tilde{y}, \mathcal{X}, \tilde{\Theta}, \tilde{\omega}, \lambda$ in our own problem. In fact, we have:

- $\tilde{y} = (x_{11}^1 - \beta_1^1, \ldots, x_{1p}^1 - \beta_p^1, \ldots, x_{n_1 1}^1 - \beta_1^1, \ldots, x_{n_1 p}^1 - \beta_p^1, \ldots, x_{11}^K - \beta_1^K, \ldots, x_{1p}^K - \beta_p^K, \ldots, x_{11}^K - \beta_1^K, \ldots, x_{n_K p}^K - \beta_p^K)$. We can view $\tilde{y}$ as a vector which is made from all elements of all data matrices $\mathbf{X}^k$ after centring by β. Consequently, the size of vector $\tilde{y}$ in our case is 1 by $\big(n_1 p + \cdots + n_K p\big)$, or simply, 1 by $(n \times p)$;
- $\tilde{\Theta} = \big(\theta_{12}^1, \ldots, \theta_{1p}^1, \ldots, \theta_{p1}^1, \ldots, \theta_{p(p-1)}^1, \ldots, \theta_{12}^K, \ldots, \theta_{1p}^K, \ldots, \theta_{p1}^K, \ldots, \theta_{p(p-1)}^K\big)$.
  We can view $\tilde{\Theta}$ as a vector which is made from all the off diagonal elements of all matrices θ^k. Consequently, the size of vector $\tilde{\Theta}$ in our case is $\big(p(p-1)K\big)$ by 1;
- $\lambda = \lambda_1$;

- $\tilde{\omega} = \big(\omega^1_{12}, \ldots, \omega^1_{1p}, \ldots, \omega^1_{p1}, \ldots, \omega^1_{p(p-1)}, \ldots, \omega^K_{12}, \ldots, \omega^K_{1p}, \ldots, \omega^K_{p1}, \ldots, \omega^K_{p(p-1)}\big)$, with the size 1 by $\big(p \times (p-1) \times K\big)$;
- $\mathcal{X}$ is a $(n \times p)$ by $(p(p-1) \times K)$ matrix defined as follows. First, denote the row vectors

$$A^k_{ij} := (x^k_{i1} - \beta^k_1, \ldots, x^k_{i(j-1)} - \beta^k_{j-1}, x^k_{i(j+1)} - \beta^k_{j+1}, \ldots, x^k_{ip} - \beta^k_p).$$

These are 1 by $(p-1)$ vectors. Then, we create $p \times (p-1)p$ matrices from these vectors:

$$B^k_i = \begin{bmatrix} A^k_{i1} & 0 & \cdots & 0 \\ 0 & A^k_{i2} & \cdots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & A^k_{ip} \end{bmatrix}.$$

Then, we make K matrices of dimension $(n_k p) \times (p(p-1))$ by stacking the B^k_i:

$$C^k = \begin{bmatrix} B^k_1 \\ B^k_2 \\ \vdots \\ B^k_{n_k} \end{bmatrix}$$

and we finally have the $np \times p(p-1)K$ matrix

$$\mathcal{X} = \begin{bmatrix} C^1 & 0 & \cdots & 0 \\ 0 & C^2 & \cdots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & C^K \end{bmatrix}.$$

We choose the R package $huge$ [20] to minimize $E_1(\theta; \beta)$. There are many good R packages to solve the default $Lasso$ problem such as $huge$ [20], $glmnet$ [21], $LARS$ [22]. Among them, the package $huge$ has a simpler user interface for our problem due to the similar framework. We only need to give the package $huge$ the data matrix $\mathbf{X}$ as input value.

2.1.3 Estimation of β^*

For fixed values of $\lambda_1, \lambda_2, \theta$, our objective function is equivalent to

$$E_2(\beta) = \frac{1}{2} \sum_{k=1}^{K} \sum_{i=1}^{n_k} \sum_{j=1}^{p} \Big(\big[x^k_{ij} - \sum_{a \neq j} \theta^k_{ja} x^k_{ia}\big] - \beta^k_j + \sum_{a \neq j} \theta^k_{ja} \beta^k_a \Big)^2 + \lambda_2 \sum_{\substack{k,l=1 \\ k<l}}^{K} \sum_{j=1}^{p} |\beta^k_j - \beta^l_j|.$$

In this case, minimizing $E_2(\beta)$ boils down to a generalized-Lasso problem [23] (redefined below). Particularly, if $K = 2$, the generalized-Lasso problem can be called Fused-Lasso. To solve this problem, we use the $genlasso$ package [24] or the $FusedLasso$ package [25]. In the case $K = 2$, besides of using the two packages, we will show that minimizing $E_2(\beta)$ (a Fused-Lasso problem) also can be seen as a $Lasso$ problem. We consider this alternative to optimize our problem as another option.

2.1.3.1 Estimating with K $\geq$ 2. Similar to Section 2.1.1, we remind the reader of the form of a generalized-Lasso with the original notations. Then we explain what are the corresponding variables in our problem.

The default form of the generalized-Lasso problem is

$$\arg\min_{\tilde{\beta}} \frac{1}{2}\|\tilde{y} - \mathcal{X}\tilde{\beta}\|_2^2 + \lambda\|D\tilde{\beta}\|_1,$$

where $\tilde{y}$ is an observation vector, $\mathcal{X}$ is a given matrix, λ is the tuning parameter, $\tilde{\beta}$ is the vector of coefficients and D is a given matrix. The problem is called generalized-Lasso also because we can encourage others structural constraints on β through the matrix D, instead of just pure sparsity as in the standard $Lasso$. Our problem is an example of a different type of structure where some coefficients in $\tilde{\beta}$ tend to be very similar to each other. Here, we have

- $\tilde{y}$ is a 1 by $(n \times p)$ vector:

$$\begin{aligned}\tilde{y} = (&x_{11}^1 - \sum_{a\neq 1}\theta_{1a}^1 x_{1a}^1, \ldots, x_{1p}^1 - \sum_{a\neq p}\theta_{pa}^1 x_{1a}^1, \ldots, x_{n_1 1}^1 - \sum_{a\neq 1}\theta_{1a}^1 x_{n_1 a}^1, \ldots,\\ &x_{n_1 p}^1 - \sum_{a\neq p}\theta_{pa}^1 x_{n_1 a}^1, \ldots, x_{11}^K - \sum_{a\neq 1}\theta_{1a}^K x_{1a}^K, \ldots, x_{1p}^K - \sum_{a\neq p}\theta_{pa}^K x_{1a}^K, \ldots,\\ &x_{n_K 1}^K - \sum_{a\neq 1}\theta_{1a}^K x_{n_K a}^K, \ldots, x_{n_K p}^K - \sum_{a\neq p}\theta_{pa}^K x_{n_K a}^K);\end{aligned}$$

- $\tilde{\beta} = (\beta_1^1, \ldots, \beta_p^1, \beta_1^2, \ldots, \beta_p^2, \ldots, \beta_1^K, \ldots, \beta_p^K)$. We can view $\tilde{\beta}$ as a vector which is made from all elements of all vectors β^k. Therefore, the size of vector $\tilde{\beta}$ is $\big(pK\big)$ by 1;

- $\lambda = \lambda_2$;

- D is a $(pK(K-1)/2) \times (Kp)$ matrix defined as follows. First, we denote $p+1$ vectors:

$e_0 = (\underbrace{0,\dots,0}_{p \text{ zero elements}})$; $e_1 = (1,\underbrace{0,\dots,0}_{p-1 \text{ zero elements}})$;...; $e_p = (\underbrace{0,\dots,0}_{p-1 \text{ zero elements}},1)$. Then, we let

$$D_j^1 = \underbrace{\left[\begin{array}{ccccc} e_j & -e_j & e_0 & \cdots & e_0 \\ e_j & e_0 & -e_j & \cdots & e_0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ e_j & e_0 & e_0 & \cdots & -e_j \end{array}\right]}_{K \text{ vectors of size } p} \Big\} K-1 \text{ lines},$$

$$D_j^2 = \underbrace{\left[\begin{array}{cccccc} e_0 & e_j & -e_j & e_0 & \cdots & e_0 \\ e_0 & e_j & e_0 & -e_j & \cdots & e_0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ e_0 & e_j & e_0 & e_0 & \cdots & -e_j \end{array}\right]}_{K \text{ vectors of size } p} \Big\} K-2 \text{ lines},$$

$$\vdots$$

$$D_j^{K-1} = \underbrace{\left[\begin{array}{cccccc} e_0 & e_0 & \cdots & e_0 & e_j & -e_j \end{array}\right]}_{K \text{ vectors of size } p} \Big\} 1 \text{ lines}.$$

Then, we introduce the matrix D_j of dimension $K(K-1)/2 \times Kp$

$$D_j = \left[\begin{array}{c} D_j^1 \\ D_j^2 \\ \vdots \\ D_j^{K-1} \end{array}\right],$$

and we finally det the matrix D of dimension $pK(K-1)/2 \times Kp$

$$D = \left[\begin{array}{c} D_1 \\ D_2 \\ \vdots \\ D_p \end{array}\right];$$

- $\mathcal{X}$ is a $(np) \times (Kp)$ matrix defined as follows. Let

$$A^k = \left[\begin{array}{cccc} 1 & -\theta_{12}^k & \cdots & -\theta_{1p}^k \\ -\theta_{21}^k & 1 & \cdots & -\theta_{2p}^k \\ \vdots & & \ddots & \vdots \\ -\theta_{p1}^k & -\theta_{p2}^k & \cdots & 1 \end{array}\right].$$

Then, we make the matrix

$$B^k = \left.\begin{bmatrix} A^k \\ A^k \\ \vdots \\ A^k \end{bmatrix}\right\} n_k \text{ times},$$

and we finally have

$$\mathcal{X} = \begin{bmatrix} B^1 & 0 & \cdots & 0 \\ 0 & B^2 & \cdots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & B^K \end{bmatrix}.$$

Thanks to this writing, we can use the R package $genlasso$ or $FusedLasso$ to solve our problem.

2.1.3.2 The case of K = 2.

The default form of a fused-Lasso problem is

$$\arg\min_{\tilde{\beta}} \frac{1}{2}\|\tilde{y} - \mathcal{X}\tilde{\beta}\|_2^2 + \lambda \sum_{j=2}^{p} |\tilde{\beta}_j - \tilde{\beta}_{j-1}|.$$

Hence, the fused-Lasso problem equals a generalized-Lasso problem with matrix

$$D = \begin{bmatrix} 1 & -1 & 0 & \cdots & 0 \\ 0 & 1 & -1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 & -1 \end{bmatrix},$$

of dimension $p - 1 \times p$.

Even if we can use the package $genlasso$ or $FusedLasso$ in this case, sometimes they do not run really fast. Indeed, our own experience shows that a Lasso problem is solved faster than a fused-Lasso problem. Hence, our idea is to rewrite the problem as a Lasso problem. When $K = 2$, the loss function is:

$$E_2(\beta) = \frac{1}{2}\sum_{k=1}^{2}\sum_{i=1}^{n_k}\sum_{j=1}^{p}\left(\left[x_{ij}^k - \sum_{a\neq j}\theta_{ja}^k x_{ia}^k\right] - \beta_j^k + \sum_{a\neq j}\theta_{ja}^k\beta_a^k\right)^2 + \lambda_2\sum_{j=1}^{p}|\beta_j^1 - \beta_j^2|.$$

The corresponding Lasso form is

$$\arg\min_{\tilde{\beta}} \frac{1}{2}\|\tilde{y} - \mathcal{X}\tilde{\beta}\|_2^2 + \lambda\|\tilde{\omega}\circ\tilde{\beta}\|_1,$$

with

- $\tilde{y}$ is 1 by $(n \times p)$ vector:

$$\begin{aligned}\tilde{y} = (&x^1_{11} - \sum_{a\neq 1} \theta^1_{1a} x^1_{1a}, \ldots, x^1_{1p} - \sum_{a\neq p} \theta^1_{pa} x^1_{1a}, \ldots, x^1_{n_1 1} - \sum_{a\neq 1} \theta^1_{1a} x^1_{n_1 a}, \ldots,\\ &x^1_{n_1 p} - \sum_{a\neq p} \theta^1_{pa} x^1_{n_1 a}, \ldots, x^K_{11} - \sum_{a\neq 1} \theta^K_{1a} x^K_{1a}, \ldots, x^K_{1p} - \sum_{a\neq p} \theta^K_{pa} x^K_{1a}, \ldots,\\ &x^K_{n_K 1} - \sum_{a\neq 1} \theta^K_{1a} x^K_{n_K a}, \ldots, x^K_{n_K p} - \sum_{a\neq p} \theta^K_{pa} x^K_{n_K a});\end{aligned}$$

- $\tilde{\beta}$ is a $2p$ by 1 vector:

$$\tilde{\beta} = (\beta^1_1, \ldots, \beta^1_p, \beta^1_1 - \beta^2_1, \ldots, \beta^1_p - \beta^2_p).$$

  Note that $(\beta^1_1, \ldots, \beta^1_p, \beta^2_1, \ldots, \beta^2_p) = M\tilde{\beta}$, where

$$M = \left[\begin{array}{cccc|cccc} 1 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 & 0 & 0 & \cdots & 0 \\ \hline 1 & 0 & \cdots & 0 & -1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 & 0 & -1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 & 0 & 0 & \cdots & -1 \end{array}\right]$$

  is a $2p \times 2p$ matrix. We need this remark to latter construct the matrix $\mathcal{X}$;

- $\lambda = \lambda_2$;

- $\tilde{\omega}$ is a 1 by $(2p)$ vector:

$$\omega = (\underbrace{0, \ldots, 0}_{p \text{ elements}}, \underbrace{1, \ldots, 1}_{p \text{ elements}}).$$

  With this vector, we only put weights to force the fusion but not the sparsity;

- $\mathcal{X}$ is a (np) by $(2p)$ matrix defined as follows. Let

$$A^k = \begin{bmatrix} 1 & -\theta^1_{12} & \cdots & -\theta^1_{1p} \\ -\theta^1_{21} & 1 & \cdots & -\theta^1_{2p} \\ \vdots & & \ddots & \vdots \\ -\theta^1_{p1} & -\theta^1_{p2} & \cdots & 1 \end{bmatrix}.$$

Then, we make the matrix

$$B^k = \left[\begin{array}{c} A^k \\ A^k \\ \vdots \\ A^k \end{array}\right] \Big\} n_k \text{ times .}$$

and we finally have:

$$\mathcal{X} = \left[\begin{array}{cc} B^1 & 0 \\ 0 & B^2 \end{array}\right] M.$$

Note that, using the procedure, we estimate the vector $(\beta_1^1, \ldots, \beta_p^1, \beta_1^1 - \beta_1^2, \ldots, \beta_p^1 - \beta_p^2)$. However, we can easily recover $(\beta_1^1, \ldots, \beta_p^1, \beta_1^2, \ldots, \beta_p^2)$ from it.

2.2 Calibration of the tuning parameters

Picking λ_1 and λ_2 (or the model selection issue) is a difficult and not new question, for which many answers exist in the literature such as Bayesian information criterion (BIC), rotation information criterion (RIC), stability approach to regularization selection (stars), stability selection method by Meinchausen & Buhlman and cross-validation. Regarding our own experience, there is no universal best choice. In some situations, a given procedure works well, but easily fails in some others. Hence, we decide to access the performance of our model on the full regularization path. This is a common practise in the literature. Although the procedure takes much time, we have a better view on the performances that our method can achieve and on its limit. We will go back to the model selection issue in more details in Chapter 3, dedicated to numerical experiments.

3 Theoretical properties

In this section, we provide a sufficient condition to guarantee asymptotic normality and selection consistency of our estimators when the number of observations n grows to infinity.

The main idea is to choose proper values for the weights and the tuning parameters. Namely, let γ be a positive number, and let $(\tilde{\beta}^k, \tilde{\theta}^k)$ be a $\sqrt{n_k}$-consistent estimators of (β^k, θ^k). For instance, we can use ordinary least square estimator (OLS) as in [18].

Besides, our choice of tunings parameters depends on the numbers of observations n_k. The choice λ_2 depends only on the total number of observation n, while the choice λ_1 is a bit more complicated and depends on each individual value n_k. In more detail, our estimators

are defined as

$$\left(\hat{\beta}^n, \hat{\theta}^n\right) := \left(\hat{\beta}^{1n}, \ldots, \hat{\beta}^{Kn}, \hat{\theta}^{n_1}, \ldots, \hat{\theta}^{n_K}\right) = \arg\min_{\theta^k, \beta^k} E(\beta, \theta, \lambda_{1n_k}, \lambda_{2n}; \mathbf{X})$$

$$= \arg\min_{\theta^k, \beta^k} \frac{1}{2} \sum_{k=1}^{K} \sum_{i=1}^{n_k} \sum_{j=1}^{p} \left(x_{ij}^k - \beta_j^k + \sum_{a \neq j} \theta_{ja}^k \beta_j^k - \sum_{a \neq j} \theta_{ja}^k x_{ia}^k \right)^2 +$$

$$\sum_{k=1}^{K} \lambda_{1n_k} \sum_{k=1}^{p} \sum_{a \neq j} \omega_{ja}^k |\theta_{ja}^k| + \lambda_{2n} \sum_{k,l=1}^{K} \sum_{j=1}^{p} \varrho_j^{kl} |\beta_j^k - \beta_j^l|.$$

where $\omega_{ja}^k = |(\tilde{\theta})_{ja}^k|^{-\gamma}$, $\varrho_j^{kl} = |(\tilde{\beta})_j^k - (\tilde{\beta})_j^l|^{-\gamma}$.

For appropriate choices of λ_{1n_k} and λ_{2n}, our theorem guarantees the asymptotic normality and the selection consistency of our estimators. Before stating the theorem, we need some additional notations to set the asymptotic (hypotheses) under which it is valid. In details, we set

- $(\beta_{\setminus j}^k)$ the vector β^k deprived of its j^{th} element;
- $\hat{\beta}^k, \hat{\theta}^k$ our estimators when number of observations in the k^{th} task is n_k;
- $\overline{\beta}^k$ the vector of empirical means;
- $\beta_{\mathcal{S}}$ the vector formed by all elements different from zero of vector β which we call support vector of vector β;
- $M_{\setminus j \setminus j}$ the matrix M deprived of its j^{th} row and j^{th} column, and $M[i,.]$, $M[,j]$ the i^{th} row, j^{th} column of M;
- $\mathcal{S}_u = \{j | j \in \{1,..,p\}, u_j \neq 0\}$ and $\mathcal{S}_u^C = \{j | j \in \{1,..,p\}, u_j = 0\}$ for u any arbitrary vector. In words, $\mathcal{S}_u$ is the index set of elements in support set of vector u, while $\mathcal{S}_u^C$ is the complement set of $\mathcal{S}_u$;
- $\mathcal{S}_{(\theta^*)^k} = \{(j,a,k) | j,a \in \{1,..,p\}; k \in \{1,..,K\}; (\theta^*)_{ja}^k \neq 0\}$ and $\mathcal{S}_{(\theta^*)^k}^C = \{(j,a,k) | j,a \in \{1,..,p\}; k \in \{1,..,K\}; (\theta^*)_{ja}^k = 0\}$;
- $(\theta^*)_{\mathcal{S}_{(\beta^*)^k}}^k$ the matrix $(\theta^*)^k$ with rows and columns restricted to the support set of vector $(\beta^*)^k$;
- $|\mathcal{S}|$ the cardinality of $\mathcal{S}$ for $\mathcal{S}$ any arbitrary set.
- $\mathcal{B} = \left\{ (j,k,l) | (\beta^*)_j^k = (\beta^*)_j^l; k,l \in \{1,..K\}; j \in \{1,..,p\} \right\}$.
- Finally, let us introduce an important notation. We would like to rewrite the vector $\beta^* = \left((\beta^*)^1, .., (\beta^*)^K\right)$ in a new and more comprehensive form. In details, for

$k, l \in \{1, .., K\}$ and $j \in \{1, .., p\}$, if we have $(\beta^*)^k_j = (\beta^*)^l_j$ and $k < l$, we remove the element $(\beta^*)^l_j$ from vector β^*. After the process, we obtain a new vector β^*_{co}. Moreover, there exist an matrix M_β such that

$$\beta^* = \beta^*_{co} M_\beta.$$

Denote M_β^{-1} as the pseudo inverse matrix of M_β.

With these notations, we assume that for all k, one has

(A1) $\quad n = c_k n_k,$

(A2) $\quad \lim_{n_k \to \infty} \frac{1}{n_k} \Big(\mathbf{X}^k - 1_{n_k} \big((\beta^*)^k \big)^T \Big)^T \Big(\mathbf{X}^k - 1_{n_k} \big((\beta^*)^k \big)^T \Big) = C^k,$

where c_k is a given positive real number and C_k is a semi positive definite matrix. Consequently, we have

$$\lim_{n_k \to \infty} \frac{1}{n_k} \Big(\mathbf{X}^k - 1_{n_k} \big((\beta^*)^k \big)^T \Big)^T_{\setminus j \setminus j} \Big(\mathbf{X}^k - 1_{n_k} \big((\beta^*)^k \big)^T \Big)_{\setminus j \setminus j} = C^k_{\setminus j \setminus j}.$$

Theorem 3.1 *Let $\gamma > 1$ be a real positive number. For all $k \in \{1, .., K\}$, suppose that $\lambda_{1n_k}/\sqrt{n_k} \to 0$, $\lambda_{1n_k} n_k^{(\gamma-1)/2} \to \infty$, $\lambda_{2n}/\sqrt{n} \to 0$, $\lambda_{2n} n^{(\gamma-1)/2} \to \infty$ and (A1), (A2) are satisfied, then our estimators have the following properties*

- *Asymptotic normality*

$$\sqrt{n_k} \Big((\hat{\theta}^{n_k})^k_{\mathcal{S}_{(\theta^*)^k}} - (\theta^*)^k_{\mathcal{S}_{(\theta^*)^k}} \Big) \to \mathcal{N}\Big(0, \sigma^2 \begin{pmatrix} (C^k_{\setminus 1 \setminus 1})^{-1}_{\mathcal{S}_{(\theta^*)^k[1,.]}} & .. & 0 \\ .. & .. & .. \\ 0 & .. & (C^k_{\setminus p \setminus p})^{-1}_{\mathcal{S}_{(\theta^*)^k[p,.]}} \end{pmatrix} \Big).$$

$$\sqrt{n} \Big((\hat{\beta}^n) - (\beta^*) \Big) \to \Big(M_\beta \, \Theta \, M_\beta^T \Big)^{-1} M_\beta \, \mathcal{N}\Big(0, \sigma^2 \Theta\Big) M_\beta^T \text{ , where}$$
$$\Theta = diag(\Theta^1, .., \Theta^K) \text{ and}$$
$$\Theta^k = c_k \big[Id_p - (\theta^*)^k \big]^T \big[Id_p - (\theta^*)^k \big].$$

- *Selection consistency*

$$\lim_{n_k \to \infty} P\Big(supp\{\hat{\theta}^{n_k}\} = supp\{(\theta^*)^k\} \Big) = 1.$$

3.1 Sketch of the proof

Due to the length of our proof, we first describe informally its main lines:

- We introduce a convex function $V^n(u,v)$ whose unique minimizer $(\hat{u}^n, \hat{v}^n)$ is the difference between our estimators and the true values (β^*, θ^*):

$$(\hat{u}^n, \hat{v}^n) = (\hat{\beta}^n, \hat{\theta}^n) - (\beta^*, \theta^*).$$

  The convex function $V^n(u,v)$ is composed by K sub convex functions, each of whom corresponding to one task, with minimizer given by $(\hat{u}^{kn}, \hat{v}^{n_k})$. We will explain these functions more clearly throughout the proof.

- Under the assumptions of the theorem, we study the asymptotic behavior of $V^n(u,v)$ when all n_k tends to infinity. We show that it converges to a convex function $V(u,v)$ whose optimal value can be defined in an explicit form.

- Applying results of Geyer [28], Knight and Fu [29], we show that $(\hat{u}^{kn/\sqrt{n}}, \hat{v}^{n_k}/\sqrt{n_k})$, the difference between the true values and our estimators for each k converges to the optimum value of function $V(u,v)$.

- Then, we prove that $(\hat{u}^{kn}, \hat{v}^{n_k})$ follows a normal distribution with known covariance matrix. As a consequence, we obtain the asymptotic normality.

- Finally, we prove the selection consistency by means of the asymptotic normality.

3.2 Proof of Theorem 3.1

3.2.1 Asymptotic normality

Let

$$\beta_j^k = (\beta^*)_j^k + \frac{u_j^k}{\sqrt{n}}, \quad \theta_{ja}^k = (\theta^*)_{ja}^k + \frac{v_{ja}^k}{\sqrt{n_k}}$$

and consider the function

$$\begin{aligned}
\Psi_n(u,v) =& L_n(u,v) + \Omega_{1n}(v) + \Omega_{2n}(u) \\
=& \sum_{k=1}^{K}\sum_{i=1}^{n_k}\sum_{j=1}^{p}\Big|x_{ij}^k - \Big((\beta^*)_j^k + \frac{u_j^k}{\sqrt{n}}\Big) + \sum_{a\neq j}\Big((\theta^*)_{ja}^k + \frac{v_{ja}^k}{\sqrt{n_k}}\Big)\Big((\beta^*)_a^k + \frac{u_a^k}{\sqrt{n}}\Big) \\
& - \sum_{a\neq j}\Big((\theta^*)_{ja}^k + \frac{v_{ja}^k}{\sqrt{n_k}}\Big)x_{ia}\Big|_2^2 + \sum_{k=1}^{K}\lambda_{1n_k}\sum_{j=1}^{p}\sum_{a\neq j} w_{ja}^k\Big|(\theta^*)_{ja}^k + \frac{v_{ja}^k}{\sqrt{n_k}}\Big| \\
& + \lambda_{2n}\sum_{k,l=1}^{K}\sum_{j=1}^{p}\varrho_j^{kl}\Big|(\beta^*)_j^k + \frac{u_j^k}{\sqrt{n}} - (\beta^*)_j^l - \frac{u_j^l}{\sqrt{n}}\Big|,
\end{aligned}$$

where, $\omega_{ja}^k = |(\tilde{\theta})_{ja}^k|^{-\gamma}$, $\varrho_j^{kl} = |(\tilde{\beta})_j^k - (\tilde{\beta})_j^l|^{-\gamma}$ and $\tilde{\theta}, \tilde{\beta}$ are ordinary least square estimators of θ^* and β^*. Denote $(\hat{u}^n, \hat{v}^n) = (\hat{u}^{n_1}, .., \hat{u}^{n_K}, \hat{v}^{n_1}, .., \hat{v}^{n_K})$ as the argmin of $\Psi_n(u,v)$, one has

$$\hat{\beta}^{kn} = (\beta^*)^k + \hat{u}^n/\sqrt{n} \text{ and } \hat{\theta}^{n_k} = (\theta^*)^k + \hat{v}^{n_k}/\sqrt{n_k}. \tag{2.4}$$

Therefore, $(\hat{u}^n, \hat{v}^n)$ is also the argmin of function $V^n(u,v)$ defined by

$$\begin{aligned} V^n(u,v) =& \Psi_n(u,v) - \Psi_n(0,0) \\ =& \underbrace{\left[L_n(u,v) - L_n(0,0)\right]}_{T^1(n)} + \sum_{k=1}^{K}\sum_{j=1}^{p}\sum_{a\neq j} \underbrace{\left[\frac{\lambda_{1n_k}}{\sqrt{n_k}} w_{ja}^k \sqrt{n_k}\left(\left|(\theta^*)_{ja}^k + \frac{v_{ja}^k}{\sqrt{n_k}}\right| - \left|(\theta^*)_{ja}^k\right|\right)\right]}_{T_{ja}^2(n_k)} \\ &+ \sum_{k,l=1}^{K}\sum_{j=1}^{p} \underbrace{\left[\lambda_{2n}\varrho_j^{kl}\left(\left|(\beta^*)_j^k + \frac{u_j^k}{\sqrt{n}} - (\beta^*)_j^l - \frac{u_j^l}{\sqrt{n}}\right| - \left|(\beta^*)_j^k - (\beta^*)_j^l\right|\right)\right]}_{T_j^3(n)}. \end{aligned}$$

We will consider the behaviors of the three terms $T^1(n), T_{ja}^2(n_k), T_j^3(n_k, n_l)$ successively when all n_k and n tend to infinity.

3.2.1.1 $\mathbf{T_{ja}^2(n_k)}$.

We note that

- If $(\theta^*)_{ja}^k \neq 0$, then $\omega_{ja}^k \longrightarrow \left((\theta^*)_{ja}^k\right)^{-\gamma}$ and $\sqrt{n_k}\left(\left|(\theta^*)_{ja}^k + \frac{v_{ja}^k}{\sqrt{n_k}}\right| - \left|(\theta^*)_{ja}^k\right|\right) \longrightarrow v_{ja}^k sign((\theta^*)_{ja}^k)$. By Slutsky's theorem, $T_{ja}^2(n_k) \xrightarrow{n_k\to\infty} 0$.
- If $(\theta^*)_{ja}^k = 0$, using the $\sqrt{n_k}$-consistency property of $\tilde{\theta}_{ja}^k$, we have
  - If $v_{ja}^k = 0$, then $T_{ja}^2(n_k) \xrightarrow{n_k\to\infty} 0$,
  - If $v_{ja}^k \neq 0$, then $T_{ja}^2(n_k) \xrightarrow{n_k\to\infty} +\infty$.

3.2.1.2 $\mathbf{T_j^3(n)}$.

We now study $T_j^3(n)$ term

- If $(\beta^*)_j^k \neq (\beta^*)_j^l$ or $\left((\beta^*)_j^k = (\beta^*)_j^l \text{ and } u_j^k = u_j^l\right)$, then $T_j^3(n) \xrightarrow{n\to\infty,} 0$.
- Otherwise, $T_j^3(n) \xrightarrow{n\to\infty,} \infty$.

3.2.1.3 $\mathbf{T^1(n)}$.

Finally, we consider the first term $T^1(n)$. Let

$$M_{ij}^k = -\frac{u_j^k}{\sqrt{n}} - \sum_{a\neq j}\frac{v_{ja}^k}{\sqrt{n_k}} x_{ia}^k + \sum_{a\neq j}\left((\theta^*)_{ja}^k \frac{u_a^k}{\sqrt{n}} + \frac{v_{ja}^k}{\sqrt{n_k}}(\beta^*)_a^k + \frac{u_a^k v_{ja}^k}{\sqrt{nn_k}}\right).$$

Hence,

$$T^1(n) = \sum_{k=1}^{K}\sum_{i,j} M_{ij}^k\left(2\epsilon_{ij}^k + M_{ij}^k\right) = \sum_{k=1}^{K}\Big(\sum_{i,j} 2\epsilon_{ij}^k\, M_{ij}^k + \sum_{i,j}(M_{ij}^k)^2\Big),$$

where ϵ_{ij}^k is the values of ϵ_j^k for the i^{th} observation. We first study the term $\sum_{i,j} \epsilon_{ij}^k \, M_{ij}^k$, then the term $\sum_{i,j}(M_{ij}^k)^2$. One has

$$\begin{aligned}
\sum_{i,j} \epsilon_{ij}^k \, M_{i,j}^k &= \sum_{k=1}^{K} \sum_{i=1}^{n_k} \sum_{j=1}^{p} \epsilon_{ij}^k \left(- \frac{u_j^k}{\sqrt{n}} - \sum_{a \neq j} \frac{v_{ja}^k}{\sqrt{n_k}} x_{ia}^k + \sum_{a \neq j} \left[(\theta^*)_{ja}^k \frac{u_a^k}{\sqrt{n}} + \frac{v_{ja}^k}{\sqrt{n_k}} (\beta^*)_a^k + \frac{u_a^k v_{ja}^k}{\sqrt{n n_k}} \right] \right) \\
&= \sum_{k,i,j} \epsilon_{ij}^k \left(\left[- \frac{u_j^k}{\sqrt{n}} + \sum_{a \neq j} (\theta^*)_{ja}^k \frac{u_a^k}{\sqrt{n}} \right] - \sum_{a \neq j} \frac{v_{ja}^k}{\sqrt{n_k}} \big(x_{ia}^k - (\beta^*)_a^k \big) + \sum_{a \neq j} \frac{u_a^k v_{ja}^k}{\sqrt{n n_k}} \right) \\
&= \sum_{k,i,j} \underbrace{\epsilon_{ij}^k \left[- \frac{u_j^k}{\sqrt{n}} + \sum_{a \neq j} (\theta^*)_{ja}^k \frac{u_a^k}{\sqrt{n}} \right]}_{S_1^k} + \sum_{k,i,j} \underbrace{\epsilon_{ij}^k \left[- \sum_{a \neq j} \frac{v_{ja}^k}{\sqrt{n_k}} \big(x_{ia}^k - (\beta^*)_a^k \big) \right]}_{S_2^k} \\
&\quad + \sum_{k,i,j} \underbrace{\epsilon_{ij}^k \left[\sum_{a \neq j} \frac{u_a^k v_{ja}^k}{\sqrt{n n_k}} \right]}_{S_3^k} \\
&= \sum_k S_1^k + \sum_k S_2^k + \sum_k S_3^k.
\end{aligned}$$

We now compute the terms S_1^k, S_2^k, S_3^k successively.

For S_1^k terms, we note that

$$\sum_{i=1}^{n_k} \frac{\epsilon_{ij}^k}{\sqrt{n}} \longrightarrow \mathcal{N}(0, \frac{n_k \sigma^2}{n}) = \mathcal{N}(0, c_k \sigma^2)$$

In other words, $\sum_{i=1}^{n_k} \frac{\epsilon_{ij}^k}{\sqrt{n_k}} \sim \epsilon_j^k = \mathcal{N}(0, c_k \sigma^2)$. Hence

$$\begin{aligned}
S_1^k = \sum_{i,j} \epsilon_{ij}^k \left(- \frac{u_j^k}{\sqrt{n}} + \sum_{a \neq j} (\theta^*)_{ja}^k \frac{u_a^k}{\sqrt{n}} \right) &= \sum_{j=1}^{p} \left(\sum_{i=1}^{n_k} \frac{\epsilon_{ij}^k}{\sqrt{n}} \right) \left(- u_j^k + \sum_{a \neq j} (\theta^*)_{ja}^k u_a^k \right) \\
&\longrightarrow \sum_{j=1}^{p} \epsilon_j^k \left(- u_j^k + \sum_{a \neq j} (\theta^*)_{ja}^k u_a^k \right) \\
&\longrightarrow u^k \Big(Id_p - (\theta^*)^k \Big) \Big(\epsilon_1^k, .., \epsilon_p^k \Big)^T \\
&\longrightarrow u^k \mathcal{N} \Big(0, \sigma^2 \Theta^k \Big), \qquad (2.5)
\end{aligned}$$

where $\Theta^k = c_k \big[Id_p - (\theta^*)^k \big]^T \big[Id_p - (\theta^*)^k \big]$.

For S_2^k, let Z^k be the $(p-1)p$-dimension vector formed by all elements of matrix v^k

except its diagonal elements. One has

$$S_2^k = \sum_{i,j} \epsilon_{ij}^k \Big(- \sum_{a \neq j} \frac{v_{ja}^k}{\sqrt{n_k}} \big(x_{ia}^k - (\beta^*)_a^k \big) \Big) = - \sum_{i,j,a \neq j} \epsilon_{ij} \frac{v_{ja}}{\sqrt{n}} \Big(x_{ia}^k - (\beta^*)_a^k \Big) \xrightarrow{n_k \to \infty} - Z^k \mathcal{N}\Big(0, \sigma^2 \mathcal{C}^k\Big), \tag{2.6}$$

where

$$\mathcal{C}^k = \begin{pmatrix} C^k_{\setminus 1 \setminus 1} & .. & 0 \\ .. & .. & .. \\ 0 & .. & C^k_{\setminus p \setminus p} \end{pmatrix} \text{ is a } (p-1)p \times np \text{ matrix,}$$

and ϵ is the np-dimension vector formed by all elements ϵ_{ij}^k.

For S_3^k, with fixed values of (u_a^k, v_{ja}^k) and note that $\sum_{i=1}^{n_k} \frac{\epsilon_{ij}^k}{\sqrt{n n_k}} \to 0$ when n_k goes to infinity, we have

$$S_3^k = \sum_{j=1}^{p} \big(\sum_{i=1}^{n_k} \frac{\epsilon_{ij}^k}{n_k} \big) \sum_{a \neq j} u_a^k v_{ja}^k \to 0. \tag{2.7}$$

From (2.5)(2.6)(2.7), one has

$$2 \sum_{i,j} \epsilon_{ij}^k \, M_{ij}^k = -2 (u^k, v^k)^T \begin{pmatrix} W_1^k \\ W_2^k \end{pmatrix}, \tag{2.8}$$

where

$$W_1^k = \mathcal{N}\Big(0, \sigma^2 \Theta^k\Big), \quad W_2^k = \mathcal{N}\Big(0, \sigma^2 \mathcal{C}^k\Big).$$

Now, we compute $\sum_{i,j} (M_{ij}^k)^2$. One has

$$\sum_{i,j} (M_{ij}^k)^2 = \sum_{i,j} \Bigg(\underbrace{\Big[- \frac{u_j^k}{\sqrt{n}} + \sum_{a \neq j} (\theta^*)_{ja}^k \frac{u_a^k}{\sqrt{n}} + \sum_{a \neq j} \frac{u_a^k v_{ja}^k}{\sqrt{n n_k}} \Big]}_{\mathrm{T1}_{ij}^k} - \underbrace{\Big[\sum_{a \neq j} \frac{v_{ja}^k}{\sqrt{n_k}} \Big(x_{ia}^k - (\beta^*)_a^k \Big) \Big]}_{\mathrm{T2}_{ij}^k} \Bigg)^2$$

$$= \sum_{i,j} \Big(\mathrm{T1}_{ij}^k - \mathrm{T2}_{ij}^k \Big)^2$$

$$= \sum_{i,j} (\mathrm{T1}_{ij}^k)^2 - 2 \, \mathrm{T1}_{ij}^k \, \mathrm{T2}_{ij}^k + (\mathrm{T2}_{ij}^k)^2.$$

When n_k grows to infinity, we have

$$\sum_{i,j} (\mathrm{T1}_{ij}^k)^2 \longrightarrow u^k \Theta^k (u^k)^T, \ \sum_{i,j} 2 \, \mathrm{T1}_{ij}^k \, \mathrm{T2}_{ij}^k \longrightarrow 0, \ \sum_{i,j} (\mathrm{T2}_{ij}^k)^2 \longrightarrow Z^k \mathcal{C}^k (Z^k)^T.$$

Therefore,

$$\sum_{i,j}(M_{ij}^k)^2 \longrightarrow (u^k, Z^k)\begin{pmatrix}\Theta^k & 0\\ 0 & \mathcal{C}^k\end{pmatrix}(u^k, Z^k)^T. \tag{2.9}$$

From (2.8)(2.9), we have

$$T^1(n) \xrightarrow{n\to\infty} \sum_{k=1}^{K}(u^k, Z^k)\begin{pmatrix}\Theta^k & 0\\ 0 & \mathcal{C}^k\end{pmatrix}(u^k, Z^k)^T - 2(u^k, v^k)\begin{pmatrix}W_1^k\\ W_2^k\end{pmatrix}. \tag{2.10}$$

3.2.1.4 Global behavior of $\mathbf{V^n}$. From the behaviors of three terms $T_1(n)$, $T_{ja}^2(n_k)$ and $T_j^3(n_k, n_l)$, we see that $V^n(u, v)$ tends to a convex function $V(u, v)$ defined by

- If $v_{ja}^k = 0$ for all $(j, a, k) \in \mathcal{S}_{(\theta^*)^k}^C$ and $u_j^k = u_j^l$ for all $(j, k, l) \in \mathcal{B}$ then

$$V(u,v) = (uM_\beta^{-1}, Z^1_{\mathcal{S}_{(\theta^*)^1}}, .., Z^K_{\mathcal{S}_{(\theta^*)^K}})\text{Mat}_1(uM_\beta^{-1}, Z^1_{\mathcal{S}_{(\theta^*)^1}}, .., Z^K_{\mathcal{S}_{(\theta^*)^K}})^T - 2(uM_\beta^{-1}, Z^1_{\mathcal{S}_{(\theta^*)^1}}, .., Z^K_{\mathcal{S}_{(\theta^*)^K}})\text{Mat}_2,$$

where

$$\text{Mat}_1 = \left[\begin{array}{c|c} M_\beta\begin{bmatrix}\Theta^1 & & & \\ & \Theta^2 & & \\ & & \ddots & \\ & & & \Theta^K\end{bmatrix}M_\beta^T & 0 \\ \hline 0 & \begin{matrix}\mathcal{C}_{\mathcal{S}_{(\theta^*)^1}} & & & \\ & \mathcal{C}_{\mathcal{S}_{(\theta^*)^2}} & & \\ & & \ddots & \\ & & & \mathcal{C}_{\mathcal{S}_{(\theta^*)^K}}\end{matrix}\end{array}\right],$$

$$\text{Mat}_2 = \begin{bmatrix} M_\beta\begin{bmatrix}W_1^1\\ \cdots \\ W_1^K\end{bmatrix}M_\beta^T \\ (W_2^1)_{\mathcal{S}_{(\theta^*)^1}} \\ \cdots \\ (W_2^K)_{\mathcal{S}_{(\theta^*)^K}}\end{bmatrix},$$

and

$$\mathcal{C}_{\mathcal{S}_{(\theta^*)^k}} = \begin{pmatrix}(C^k_{\backslash 1\backslash 1})^{-1}_{\mathcal{S}_1} & .. & 0\\ .. & .. & ..\\ 0 & .. & (C^k_{\backslash p\backslash p})^{-1}_{\mathcal{S}_p}\end{pmatrix}.$$

- Otherwise, $V(u, v) = +\infty$.

Because function $V^n(u,v)$ is convex and $V(u,v)$ has unique optimal value which is

$$(\hat{u},\hat{v}) = (\hat{u}^k,\hat{v}^k) = (\text{Mat}_1)^{-1}\text{Mat}_2.$$

More precisely,

$$\begin{aligned}
\hat{u} =& \Big(M_\beta \ \text{diag}(\Theta^1,..,\Theta^K)\ M_\beta^T\Big)^{-1} M_\beta (W_1^1,..,W_1^K)^T M_\beta^T,\\
\hat{v}^k =& \Big(\hat{v}^k_{\mathcal{S}_{(\theta^*)^k}}, \hat{v}^k_{\mathcal{S}^C_{(\theta^*)^K}}\Big)\\
=& \Big((\mathcal{C}_{\mathcal{S}_{(\theta^*)^k}})^{-1}(W_2^1)_{\mathcal{S}_{(\theta^*)^k}}, 0\Big).
\end{aligned}$$

We apply the epi-convergence results of Geyer(1994), Knight and Fu(2000), and we have

$$(\hat{u}^n, \hat{v}^n_{\mathcal{S}_{\theta^*}}) \to (\hat{u},\hat{v}).$$

Hence, from equation (2.4) we have the asymptotic normality of our estimators.

3.2.2 Selection consistency

Now we prove the second part of the theorem which is the consistency of the support of our estimators $\hat{\theta}$. From the asymptotic normality of the support part, for all $(\theta^*)^k_{ja} \neq 0$, we have $\hat{\theta}^k_{ja} \to (\theta^*)^k_{ja}$. Hence, our estimators detect all the true different zero values (true edges). Now we just have to show that, our estimators do not detect wrong edges. In other words, for all variable X_b^k that do not belong to $\mathcal{S}_j^k$-the neighbourhood set of X_j^k, we must verify that

$$P(X_b^k \in \mathcal{S}_j^k) \to 0, \quad \forall b \notin \mathcal{S}_j^k.$$

Suppose that $X_b^k \in \mathcal{S}_j^k$, hence, $\hat{\theta}^k_{jb} \neq 0$. Following the KKT conditions, we have

$$\frac{\partial E(\beta,\theta;\mathbf{X})}{\partial \theta^k_{jb}}(\hat{\beta},\hat{\theta}) = 0.$$

Hence,

$$\sum_{i=1}^{n_k}\left(x_{ij}^k - \hat{\beta}_j^k - \sum_{a\neq j}\hat{\theta}^k_{ja}\big(x^k_{ia} - \hat{\beta}^k_a\big)\right)(x^k_{ib} - \hat{\beta}^k_b) = \lambda_{1n_k} w^k_{jb}.$$

Consequently,

$$\frac{1}{\sqrt{n_k}}\sum_{i=1}^{n_k}\left(x_{ij}^k - \hat{\beta}_j^k - \sum_{a\neq j}\hat{\theta}^k_{ja}\big(x^k_{ia} - \hat{\beta}^k_a\big)\right)(x^k_{ib} - \hat{\beta}^k_b) = \frac{\lambda_{1n_k} w^k_{jb}}{\sqrt{n_k}}.$$

Now will we show that the left hand side is bounded while the right hand side tends to $+\infty$. This would be a contradiction, that would proof the selection consistency. For the right hand

side, from our initial hypothesis on λ_{1n_k}, one has

$$\frac{\lambda_{1n_k} w_{jb}^k}{\sqrt{n_k}} = \frac{\lambda_{1n_k}}{\sqrt{n_k}} n_k^{\gamma/2} \frac{1}{|\sqrt{n_k}\tilde{\theta}_{bj}^k|^\gamma} = \infty \times\ O(1) \to \infty.$$

On the other hands, note that $x_{ij}^k = (\beta^*)_j^k + \sum_{a \neq j} (\theta^*)_{ja}^k \big(x_{ia}^k - (\beta^*)_a^k\big) + \epsilon_{ij}^k$. Consequently, the left hand side satisfies

$$\begin{aligned}
&\frac{1}{\sqrt{n_k}} \left| \sum_{i=1}^{n_k} \left[(\beta^*)_j^k - \hat{\beta}_j^k + \sum_{a \neq j} \Big((\theta^*)_{ja}^k - \hat{\theta}_{ja}^k \Big) x_{ia}^k - \sum_{a \neq j} \Big((\theta^*)_{ja}^k (\beta^*)_a^k - \hat{\theta}_{ja}^k \hat{\beta}_a^k \Big) + \epsilon_{ij}^k \right] \Big(x_{ib}^k - \hat{\beta}_b^k \Big) \right| \\
=&\frac{1}{\sqrt{n_k}} \left| \sum_{i=1}^{n_k} \left[(\beta^*)_j^k - \hat{\beta}_j^k + \sum_{a \neq j} \Big((\theta^*)_{ja}^k - \hat{\theta}_{ja}^k \Big) \Big(x_{ia}^k - (\beta^*)_a^k \Big) \right. \right. \\
&\qquad \left. \left. - \sum_{a \neq j} \Big(\hat{\theta}_{ja}^k (\beta^*)_a^k - \hat{\theta}_{ja}^k \hat{\beta}_a^k \Big) + \epsilon_{ij}^k \right] \Big(x_{ib}^k - \hat{\beta}_b^k \Big) \right| \\
\leq& \underbrace{\frac{1}{\sqrt{n_k}} \left| \Big((\beta^*)_j^k - \hat{\beta}_j^k \Big) \sum_{i=1}^{n_k} \Big(x_{ib}^k - \hat{\beta}_b^k \Big) \right|}_{Z_1} + \underbrace{\frac{1}{\sqrt{n_k}} \left| \sum_{i=1}^{n_k} \left(\sum_{a \neq j} \Big((\theta^*)_{ja}^k - \hat{\theta}_{ja}^k \Big) (x_{ia}^k - (\beta^*)_a^k) \right) \Big(x_{ib}^k - \hat{\beta}_b^k \Big) \right|}_{Z_2} \\
&+ \underbrace{\frac{1}{\sqrt{n_k}} \left| \sum_{a \neq j} \Big(\hat{\theta}_{ja}^k (\beta^*)_a^k - \hat{\theta}_{ja}^k \hat{\beta}_a^k \Big) \sum_{i=1}^{n_k} \Big(x_{ib}^k - \hat{\beta}_b^k \Big) \right|}_{Z_3} + \underbrace{\frac{1}{\sqrt{n_k}} \left| \sum_{i=1}^{n_k} \epsilon_{ij}^k \Big(x_{ib}^k - \hat{\beta}_b^k \Big) \right|}_{Z_4}.
\end{aligned}$$

We now show that the 4 terms Z_1, Z_2, Z_3, Z_4 are bounded. Using the asymptotic normality results obtained in the Section 3.2.1, we have

- First term

$$Z_1 = \left| \sqrt{n_k} \Big((\beta^*)_j^k - \hat{\beta}_j^k \Big) \right| \left| \frac{\sum_{i=1}^{n_k} (x_{ib}^k - \hat{\beta}_b^k)}{n_k} \right| \longrightarrow 0.$$

- Second term

$$\begin{aligned}
Z_2 =& \sum_{a \neq j} \left| \sqrt{n_k} \Big((\theta^*)_{ja}^k - \hat{\theta}_{ja}^k \Big) \right| \left| \Big(x_{ia}^k - (\beta^*)_a^k \Big) \frac{\sum_{i=1}^{n_k} (x_{ib}^k - \hat{\beta}_b^k)}{n_k} \right| \\
\longrightarrow& \sum_{a \neq j} \left| \sqrt{n_k} \Big((\theta^*)_{ja}^k - \hat{\theta}_{ja}^k \Big) \right| \mathcal{C}_{ab}^k = \mathcal{N}\Big(0, C_2\Big), \text{where } C_2 \text{ is a constant.}
\end{aligned}$$

- Third term

$$Z_3 = \sum_{a \neq j} \left| \sqrt{n_k} \Big((\beta^*)_j^k - \hat{\beta}_j^k \Big) \right| \left| \hat{\theta}_{ja}^k \right| \left| \frac{\sum_{i=1}^{n_k} (x_{ib}^k - \hat{\beta}_b^k)}{n_k} \right|$$
$$\longrightarrow \sum_{a \neq j} \left| \sqrt{n_k} \Big((\beta^*)_j^k - \hat{\beta}_j^k \Big) \right| \left| \hat{\theta}_{ja}^k \right| 0 = 0.$$

- Fourth term

$$Z_4 \to \frac{1}{\sqrt{n_k}} \mathcal{N}(0, 4 \sum_{i=1}^{n_k} \|x_{ib}^k - (\beta^*)_b^k\|^2 \sigma^2).$$

Therefore, the left hand side go to infinity, while the right hand side is controlled. Hence,

$$P(X_b^k \in \mathcal{S}_j^k) \leq P(\frac{\partial E(\beta, \theta; \mathbf{X})}{\partial \theta_{jb}^k}(\hat{\beta}, \hat{\theta}) = 0) \to 0.$$

As a conclusion, we have the selection consistency of our $\hat{\theta}$.

4 Usability towards gene expression analysis

In this section, we explain how our model can be used to answer both questions of differential analysis and network inference from a practical point of view. We explore in details several set-ups arising naturally in genomics. First, when we consider one single gene, independently of the other genes. Second, we consider a set of genes.

At the single gene level, we aim to answer the question that, given one gene and two conditions, is the gene expressed differentially between the two conditions? We refer to this question as "univariate differential analysis". In case of more than two conditions, we perform pairwise conditions tests for each gene. For instance, if one has data about one gene in three conditions labelled 1, 2 and 3, we perform 3 tests to detect changes in the mean of this gene between conditions 1 and 2, 2 and 3, 3 and 1.

In the more complicated case where we consider a set of genes, we wish to answer two questions. First, is there any gene in the set whose mean expression changes between two given conditions? This question could be stated in another way: are the two vectors formed by the mean expression levels of all genes in the considered set different? We refer to this question as "multivariate differential analysis". Second, what is the interaction network between the genes in this set? We call this question "network inference".

The following part explains ways to answer these questions at the two corresponding levels.

4.1 Single gene analysis

4.1.1 Univariate differential analysis

To answer the univariate differential analysis, we choose to create a priority list. This list consists of triplets (j, k, l) indicating that gene j changes its mean expressions between conditions k, l with a given probability. We sort the list according to this probability. Actually, for a fixed value of λ_1, the priority list may be obtained from the ACS algorithm by varying λ_2 from infinity to zero. Note that, the bigger the λ_2, the more the mean expression values fuse between the conditions. For each triplet (j, k, l), we assign a value of λ_2 called the fusion time. The fusion time of (j, k, l) indicates the smallest values of λ_2 making two estimators of $(\beta^*)^k_j$ and $(\beta^*)^l_j$ equal. Therefore, if the fusion time of one triplet (j, k, l) is big, it means that gene j has a high probability to differ between the two conditions (k, l). We can sort the list base on the corresponding fusion times of triplets.

Note that this priority list is associated with a given λ_1. In an easy scenario where we know the covariance matrix or the network, there is nothing to do with θ and its corresponding penalty part. Nevertheless, in most of the cases, prior information about the network is very limited. Therefore, we would try every possible values of λ_1 to see which one leads to the best results. To sum up, we proceed as follows:

Procedure to find priority list for univariate analysis

1. Generate a list of λ_1 from zero to infinity.
2. For each fixed value of λ_1:
   - Find the fusion times of all triplets by varying values of λ_2 and using ACS algorithm.
   - Make the priority list of this λ_1 based on sorting the fusion times in decreasing order.

Final result will be several lists of triplets. Each list corresponds to one λ_1. In each list, the top triplets are the most important triplets.

4.2 Gene set analysis

4.2.1 Multivariate differential analysis

At this level, we provide a priority list of all the couples (k, l), where the first couple (k, l) corresponds to a pair of conditions in which mean expression vectors of all genes in the set

has the highest possibility to change among all the couples of two conditions. In term of fusion time, the first couple (k, l) corresponds to the first value of λ_2 making mean expression level vector in one condition different from any other vectors in the remaining conditions.

Procedure to find the priority list for multivariate differential analysis is similar to the univariate case. The major difference is that, instead of detecting one triplet by comparing two mean values of one gene between two conditions, we now detect a couple by comparing

two vectors of means of a set of genes in two conditions. The procedure is detailed here:

Procedure to find priority list for multivariate analysis

1. Generate a list of λ_1 from zero to infinity.
2. For each fixed value of λ_1:
   - Find the fusion times of all couples by varying values of λ_2 and using ACS algorithm.
   - Make the priority list of this λ_1 based on sorting the fusion times in decreasing order.

4.2.2 Network inference

Again, we build on the idea of a priority list, but now in terms of edge detection. For a fixed value of λ_2, we vary λ_1 from zero to positive infinity. Note that, the bigger the λ_1, the less the edges in the network corresponding to $(\hat{\theta})^k$. For each triplet (j, a, k), we assign a value of λ_1 called the shrinking time. The shrinking time of (j, a, k) indicates the smallest values of λ_1 making estimator of $(\theta^*)^k_{ja}$ equal to zero. Therefore, if the shrinking time of a triplet (j, a, k) is big, it means that there is a high probability of link between genes (j, a) in condition k. For each couple (λ_1, λ_2), we use ACS algorithm to estimate $(\theta^*)^k$. Hence, we detect a new edge and add them into the priority list. An edge could be labelled by one triplet (j, a, k) where j, a denote two genes and k a condition. Therefore, each value of λ_2 gives us a priority list. This strategy avoids choosing λ_2 which is known to be a difficult problem. In detail, we derive the priority list as follows:

Procedure to find priority list in network inference

1. Generate a list of λ_2 from zero to infinity.
2. For each fixed value of λ_2:
   - Find the shrinking times of all triplets by varying values of λ_1 and using ACS algorithm.
   - Make the priority list based on sorting the shrinking times in decreasing order.

---

5 Conclusion of Chapter 2

In Chapter 2, we described our model and proposed an algorithm to estimate its parameters. From a theoretical perspective, we prove the consistency of our estimators. In practice, we propose a procedure for using our model to answer the two questions of differential analysis and network inference.

Chapter 3

Numerical experiments

The goal of this chapter is to compare the performances of our method with some other methods which are widely used to perform differential analysis and network inference. We first describe our simulation protocol. Then we explain how we compare the different methods and finish by discussing the results obtained.

1 Simulation procedure and experimental design

For both differential analysis experiment and network inference experiment, we simulate data in a similar fashion. From one experiment to the next, we change the values of some parameters such as the size of the data matrices or the number of tasks (or biological conditions). We do this for three reasons.

First, we want to study the behavior of all methods when we give them more and more information. This is a very standard goal in statistics, and we simply increase the number of observations to do that.

Second, we need to vary these parameters to balance both the statistical and the computational complexity of the experiment. Depending on the goal of the experiment (network inference or differential analysis), the studied data could be easy or very hard to analyse. For example, in our experiments, the network inference task is usually more complicated than the differential analysis task for two main reasons. First, the number of parameters to estimate is much larger in network inference experiments. Second, differential analysis methods such as the t-test and the ANOVA usually have simpler computation. These methods run fast even for datasets containing thousand of genes. On the other hand, methods to infer network such as the graphical lasso and the neighbourhood selection usually use an iterative algorithm and sometime have a long runtime, especially for datasets with large number of genes. Therefore, a dataset containing thousands of genes can be relatively easy to analyse in the context of differential analysis. However, it may turn to be very difficult to analyse in the context of network inference.

Third, we vary the simulation parameters to better compare and assess the differences between the methods. For example, in our differential analysis experiments, in an easy scenario when we have some prior information about the network, many methods work well. In contrast, in some hard scenarios like when we do not have any prior information, all considered methods fail. If we want to compare them, we have to find new settings in which the differences between the methods are clearer and easier to evaluate.
In the following section, we describe the main steps to simulate our datasets. Detailed values of the parameters in each experiment are summarized in Table 3.1.

1.1 Protocol to simulate data

Our protocol is in 4 steps:

1. Choose the number of tasks (conditions) K, and the number of variables (genes) p.
2. Choose the number of observations (samples) n_k for each task k.
3. Generate a mean vector and a covariance matrix $\left((\beta^*)^k, (\Sigma^*)^k\right)$ for each task k.
   - For the **mean vectors,** we always fix mean vector in the first condition to zero, i.e $(\beta^*)^1 = 0$. Then, we use a parameter μ to control the similarity between $(\beta^*)^1$ and the mean vectors of the other tasks $(\beta^*)^k$. For instance, if μ equals 50%, half of the elements of $(\beta^*)^k$ equal zero for all $k \neq 1$. For each task, the set of zero elements is chosen randomly. All other elements of vector $(\beta^*)^k$ are set equal to $1/2$.
   - For the **covariance matrices,** all $(\Sigma^*)^k$ are set equal. We generate them using the R package $huge$. As the choice of the network is also important, we consider 4 network types proposed by the $huge$ R package which are called band, cluster, hub, and random. An example of each network type is shown in Figure 3.1. In detail, the covariance matrices are generated by the R function

     $$(\Sigma^*)^k \leftarrow huge.generator(2, p, \text{type of network})\$sigma.$$

     We note that the number 2 in the above function could be replaced by any positive integer number. In fact, the R function $huge.generator$ is made to generate a dataset following a centered normal distribution. The covariance matrix of this normal distribution is a by-product of this function. The number of observations of the simulated dataset is chosen by the first parameter in this function. In our simulation, we only recover the covariance matrix generated by this function.

     In fact, given the network (denoted by the adjacency matrix $\mathcal{A}$) and its type, generating the corresponding covariance matrix is not a trivial work. In the $huge$ package, to obtain a positive definite precision matrix, the smallest eigenvalue of

$0.3 \times \mathcal{A}$ is computed. We denote it as e. Then authors set the precision matrix equal to $0.3 \times \mathcal{A} + (|e| + 0.1 + 0.1)I$, where I is the identity matrix. The covariance matrix is then computed to generate multivariate normal data.

4. We generate the simulated data for each task using a multivariate normal distribution $\mathcal{N}((\beta^*)^k, (\Sigma^*)^k)$. In R, we use the package $mtvnorm$ with the following code:

$$\mathbf{X}^k \leftarrow rmvnorm\Big(n_k, mean = (\beta^*)^k, sigma = (\Sigma^*)^k\Big).$$

1.2 Importance of the network type

In a network, the degree of a node is defined as the number of node directly connected to it. The maximum degree of all nodes is usually called the sparsity parameter of the network [30] and we denote it by d. A problem is in a ultra-high dimensional setting if $d[1 + log(p/d)]$ is larger than $n/2$. In [30], the author also demonstrated that any network inference procedure fails for data in a ultra-high dimensional setting in the sense that their minimax risks blow up in this setting. Therefore, the choice of parameters such as the sparsity and the number of nodes for each network is very important. The value of d can be controlled in the function $huge.generator$. In all of our simulations, we always choose d less than 4 and avoid the ultra-high dimensional setting.

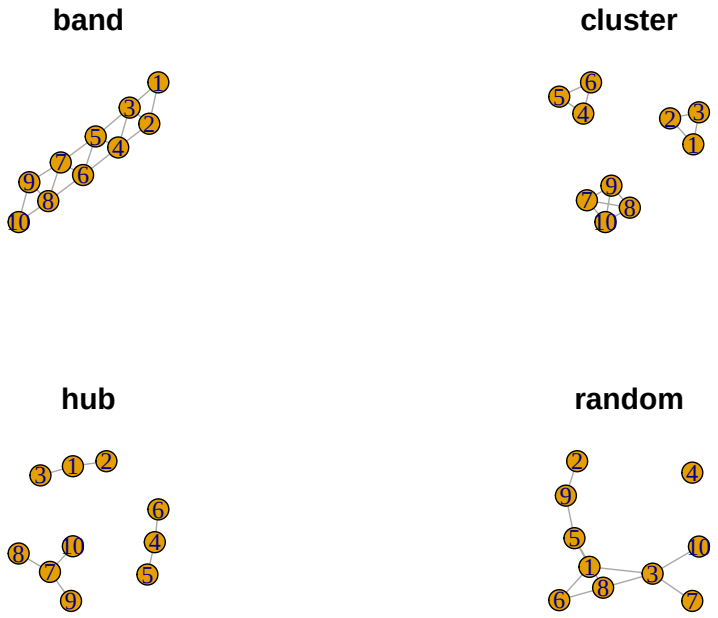


Figure 3.1: Example of 4 types of network simulated by $huge$ R package. Each of network contains 10 nodes. The sparsity parameter of each network is less than 4.

Another aspect we study is the type of network. As discussed, we choose 4 types of network. The idea is to consider both structured networks such as band, cluster and hub; and random networks. See Figure 3.1 for a graphical representation:

- The band network case is illustrated in Figure 3.1 (top left) with nodes labelled from 1 to 10. Each three consecutive nodes are connected and make a band. The sparsity parameter d is 4 (for example, node 3 is connected to nodes 1, 2, 4 and 5).
- The cluster network case is illustrated in Figure 3.1 (top right). We have 3 different clusters. In each cluster, all genes are connected. In Figure 3.1, we have $d = 3$ (the node 8 for example is connected to nodes 7, 9 and 10).
- The hub network case is illustrated in Figure 3.1 (bottom left). We also have different clusters. However, in each cluster, there is a central hub gene and all other genes are linked to it. For instance, in the cluster, $\{1, 2, 3\}$, gene 1 is the central hub gene and genes 2,3 are linked to it. In that case, we have $d = 3$ (for example, the node 7 is linked to nodes 8, 9 and 10).
- The random network case is illustrated in Figure 3.1 (bottom right). Each node is randomly linked to some other nodes. The nodes are chosen randomly but the number of nodes must be less than 4. In Figure 3.1, we have $d = 4$ (the node 3 has the highest degree. It is linked to nodes 1, 7, 8 and 10).

From a biological point of view, structured networks have received a lot of attention [31], [32], [33]. That is why we consider them in our analysis. Even though it is thought that "band" and "random" networks rarely appear in practice, we think that this case is interesting at least as a reference.

In the next section, we consider two cases. Either the number of conditions K is equal to 2 or greater than 2. In both cases, we analyse the data at two levels, single gene and gene set in order to address either the gene differential analysis or network inference problems.

Table 3.1: Summary table of all experiments

Number of task	Analysis level	Type of experiment	Type of network	Scenario	p	n	Methods	Number of dataset
2	Single gene	Univariate analysis	Band, Cluster, Hub, Random	1. Given covariance matrix	50	n1 = n2 in {10,20,30,50}	t-test, ANOVA, Wilcoxon, Sam, Limma	100 per one couple (n1,n2)
				2. Given network				
				3. No prior information			Limma,	
	Set of gene	Multivariate analysis		1. Given covariance matrix	20	n1 = n2 in {12,15,20}	Hotelling-T^2 test, Jacob et al's test	6000 per one couple (n1,n2)
				2. Given network				
				3. No prior information	20	n1 = n2 in {12,15,20}	Hotelling-T^2 test	1000 per one couple (n1,n2)
					50	n1 = n2 in {10,20}	None	1000 per one couple (n1,n2)
		Network inference		1. Given mean vectors 2. No prior information	50	n1 = n2 = 20	Graphical lasso	100 per one couple (n1,n2)
5,8,10	Set of gene	Network inference		1. Given mean vectors 2. No prior information	50	n1 = n2 = 20	Graphical lasso	100 per one couple (n1,n2)

2 Differential analysis

We consider the case where the number of conditions K is 2. In the case where $K \geq 3$, we obtain very similar results.

2.1 Results for one gene at a time analysis

In this section, we study the univariate gene differential analysis problem.

2.1.1 Experiment set up

In this experiment, we choose $p = 50$, all n_k are equal and take a value in the set $\{10, 20, 30, 50\}$, the similarity parameters $\mu = 50\%$. With 4 options for n_k and 4 network types, we consider a total of 16 cases. For each case, we simulated 100 datasets.

2.1.2 Competitors

We compare our method to five hypothesis testing methods, namely the Welch's t-test, the ANOVA, the limma, the Wilcoxon test and the SAM method. All these methods are introduced in Chapter 1, Section 2.1. We compare the Receiver Operating Characteristic (ROC) and Area Under the Curve (AUC) of all these methods for different types of simulated datasets.

2.1.3 Three scenarios and prior information

To get more insights about the performance of our method and evaluate the importance of knowing the network for the univariate differential analysis, we create the three following scenarios:

- **Scenario 1:** The covariance matrices $(\Sigma^*)^k$ or the precision matrices $((\Sigma^*)^k)^{-1}$ are known. In this scenario, we have perfect prior knowledge about the gene network. Both edges and amplitudes of interactions are known. Therefore, we know $(\theta^*)^k$ directly from $(\Sigma^*)^k$:

$$(\theta^*)^k_{ja} = \frac{(\Theta^*)^k_{ja}}{(\Theta^*)^k_{jj}}, \text{ where } (\Theta^*)^k = \left((\Sigma^*)^k\right)^{-1}.$$

  Thus, we assign λ_1 to zero and simply consider a grid of λ_2 to infer $(\beta^*)^k$. For each value of λ_2, we only estimate mean vectors $(\beta^*)^k$. This scenario is a reference scenario. It provides an upper bound on the performance of our method.

- **Scenario 2:** The adjacency matrix of the precision matrix (i.e. the network) is known. This scenario is a bit closer to what we usually have in practice. In this scenario, we know exactly the positions of all edges in the network. Hence, we set λ_1 equal to 1 and put very large weights on unconnected edges, and zero weights on connected edges. More precisely, if $\mathcal{A}^k$ is the adjacency matrix corresponding to the network in condition k, we choose

$$\lambda_1 = 1, \quad \omega_{ja}^k = \begin{cases} +\infty & \text{if } \mathcal{A}_{ja}^k = 0 \text{ (In our R code, we set it to } 10^6) \\ 0 & \text{if } \mathcal{A}_{ja}^k = 1 \end{cases}.$$

  Then, we make a grid of λ_2. For each fixed value of λ_2, we can estimate (β^*, θ^*).

  In practice, we never have full information about the network, but possibly a part of it. We consider this scenario as the second reference scenario. In this scenario, due to the lack of information about amplitude of interactions, we expect our method to perform worse than in the first scenario.

- **Scenario 3:** We have no prior information about the covariance matrix (or precision matrix). This is the most difficult scenario. In practice, at least for small biological networks, we are often somewhere between scenario 2 and 3. In this hard scenario, we have to make a two dimensional grid for λ_1 and λ_2 as we need to infer both $((\beta^*)^k, (\theta^*)^k)$. For each couple (λ_1, λ_2) on the grid, we get an estimator $(\hat{\beta}^k, \hat{\theta}^k)$.

2.1.4 Priority list

For each simulated dataset and each method, we order the genes in a priority list. For our method, the priority list is made by ordering genes by fusion times in decreasing order. The fusion times are created by the procedure described in Chapter 2, Section 4.1.1. Each fusion time values is labelled by a triplet (j, k, l), where j is a gene and k, l are two biological conditions (or tasks). For example, if the fusion time takes value 1, it means that the minimal value of λ_2 to fuse the two estimators of $(\beta^*)_j^k$ and $(\beta^*)_j^l$ is 1. The higher the fusion time is, the higher the difference between the two estimators of $(\beta^*)_j^k$ and $(\beta^*)_j^l$ is. To be specific, assume that we have two triplets (j_1, k_1, l_1) with rank r_1 and (j_2, k_2, l_2) with rank r_2. If $r_1 < r_2$, then the difference of gene j_1 between the two conditions k_1 and l_1 is considered more significant than the difference of gene j_2 between two conditions k_2 and l_2. Consequently, if our method detects $(\beta^*)_{j_2}^{k_2} \neq (\beta^*)_{j_2}^{l_2}$, then automatically it will also detect $(\beta^*)_{j_1}^{k_1} \neq (\beta^*)_{j_1}^{l_1}$. For the five hypothesis testing methods, the priority list is made by ordering the p-values increasingly. Similar to our method, each p-value is assigned to a triplet (j, k, l). The main difference is that the smaller the p-value is, the more significant the difference between the two estimators of $(\beta^*)_j^k$ and $(\beta^*)_j^l$.

In what follows, we are in the case where $K = 2$ tasks, therefore we get only p scores, as we only perform p tests. In the general case with arbitrary values of K, we perform pairwise condition tests and obtain $K(K-1)p/2$ scores.

2.1.5 ROC and AUC

Given a priority list of triplets, it is possible to make a ROC curve for each method. Each triplet corresponds to a point on the ROC curve (that gives the performance of the approach if this triplet is the last detected by the method). The horizontal coordinate of this point is the false positive rate (FPR), and the vertical coordinate of this point is the true positive rate (TPR). These coordinates are computed by the following procedure:

1. For each triplet (j, k, l), we consider all triplets in the priority list having higher rank than (j, k, l) and itself as detected triplets by the method.

2. For this set of detected triplets, we compute

$$\text{TPR}(j,k,l) = \frac{\text{Number of detected triplets that are true}}{\text{Number of triplets corresponding to a differential expression}},$$
$$\text{FPR}(j,k,l) = \frac{\text{Number of detected triplets that are wrong}}{\text{Number of triplets corresponding to a non differential expression}}.$$

From this, we make the ROC curves and compute the AUC of each curve as represented in Figure 3.2 for instance. The AUC takes values in the range $[0, 1]$. The bigger the AUC of a method, the better the detection.

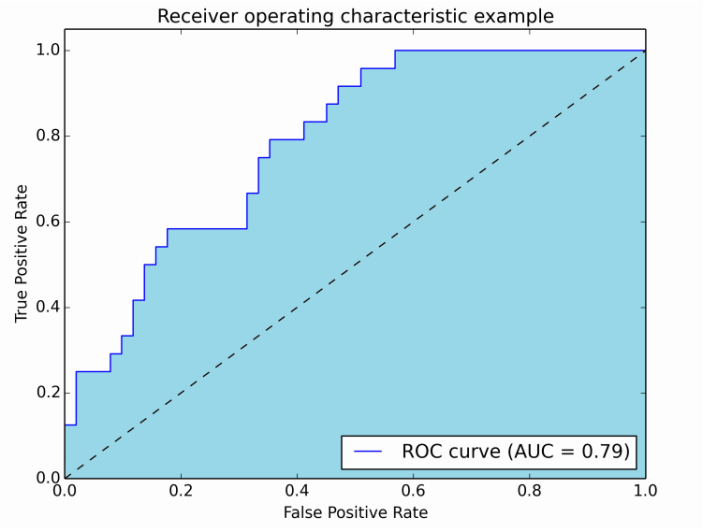


Figure 3.2: Example of a ROC curve and its AUC. The AUC takes its value between 0 and 1. The dash line presents the random guess.

Note that each ROC curve corresponds to one simulated dataset. As we have 100 datasets, we have 100 AUC values for each method. From that, we can make boxplots for the 6 methods.

We now look at the performances of all methods in the 3 different scenarios introduced in Section 2.1.1.

2.1.6 Univariate analysis results

2.1.6.1 Results with known covariance matrix. In this scenario, we study the performance of our method when full information about the covariance matrix is given. The results obtained are very similar for all types of network. For the cluster network, results of the AUC for 4 choices of n_k are shown in Figure 3.3.

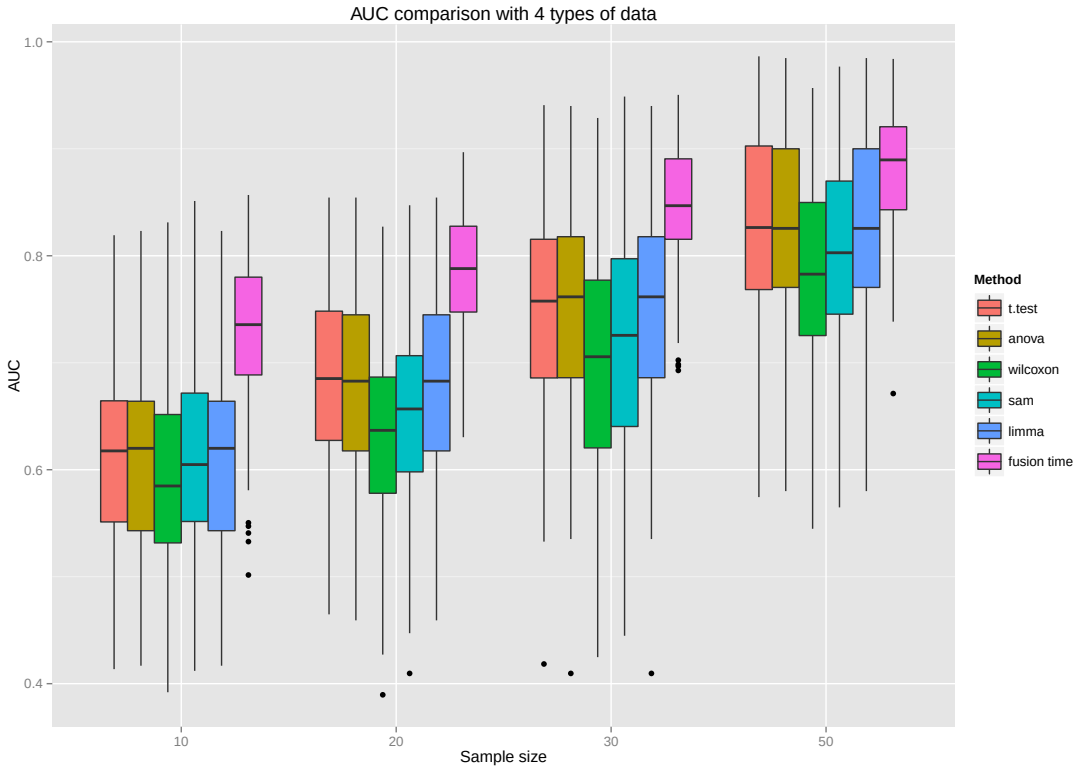


Figure 3.3: Scenario 1: covariance matrix $(\Sigma^*)^k$ is known. Here, we compare the AUC of 6 methods t-test, ANOVA, wilcoxon, sam, limma and our method (fusion time). We simulated data with a number of observations is $n_k \in \{10, 20, 30, 50\}$. Each box represents the AUC of one method for one value of n_k over 100 simulations.

Clearly, our method (the pink box) is better than the other methods for all choices of n_k. This is expected because our method has a big advantage in this scenario. More precisely, while other methods only take into account the variance of each gene independently, our method take into account the covariance matrices. We also see that the fewer the observations, the higher the difference between the pink box and the others. This result somewhat justifies

the importance of knowing the covariance matrix in complex situation as it improves gene differential analysis.

As a conclusion, prior information on the network is important even for the univariate differential analysis, especially when we have few observations. All methods are very good when we have abundant observations. For other types of network, their results are shown in Appendix A.

2.1.6.2 Result with known network. In this scenario, we study the performance of our method when we provide the adjacency matrix of the precision matrix. For cluster networks, results are shown in Figure 3.4.

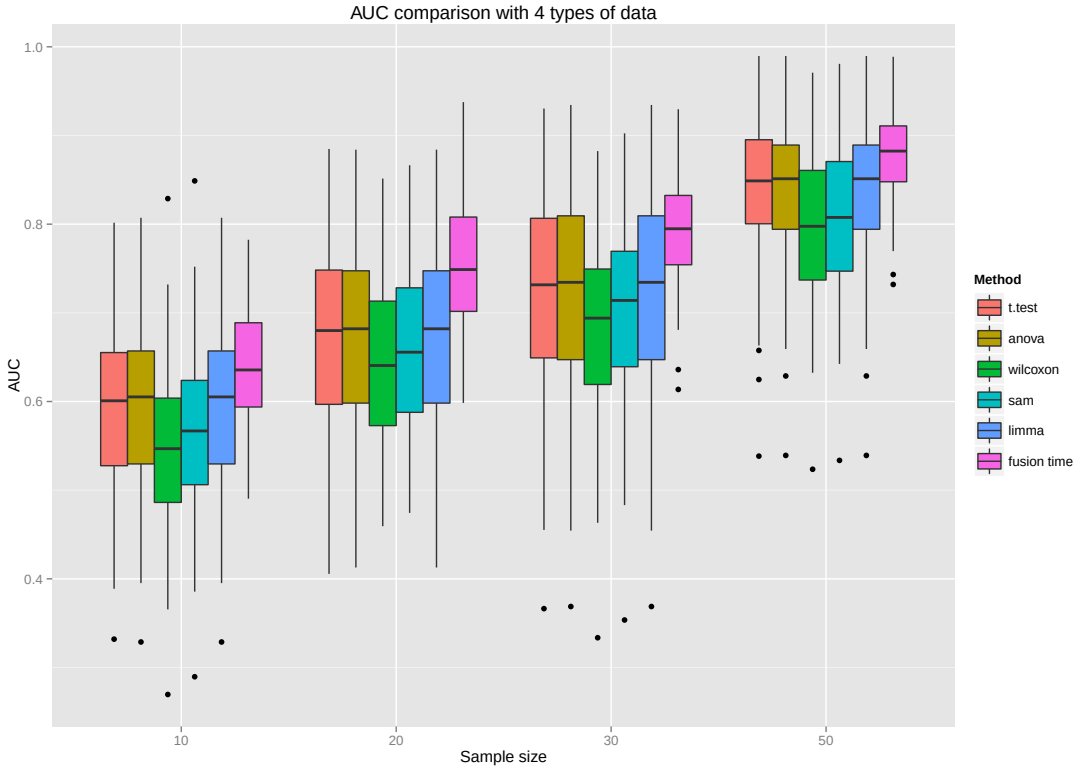


Figure 3.4: Scenario 2: network is known. Here, we compare the AUC of 6 methods t-test, ANOVA, wilcoxon, sam, limma and our method (fusion time). We simulated data with a number of observations is $n_k \in \{10, 20, 30, 50\}$. Each box represents the AUC of one method for one value of n_k over 100 simulations.

As expected, our method (the pink box) is still better than the others for every choices of n_k. Note that, the result of our method is a bit worse than in the first scenario. This is explained by the lack information on the true value of each element in the covariance matrix.

These results emphasizes one more time the importance of the network in the univariate differential analysis. The results for other types of network are similar and shown in Appendix A.

2.1.6.3 Results with no prior information about the covariance matrix. In the most difficult scenario, we need to infer both the network and the mean expression vectors. To do this, we create a grid of 8 values for λ_1 following a log scale. For each dataset and each values of λ_1, we can compute an AUC . We compare these AUC to the AUC of $limma$ since $limma$ is a standard method for this kind of problem. In addition, we use the AUC of our method in the first scenario as another control. Hence, we have 10 boxes in a plot. Results for the cluster network for $n_k \in \{10, 20, 30, 50\}$ are shown in Figures 3.5, 3.6, 3.7. Results of other types of network are similar and shown in Appendix A.

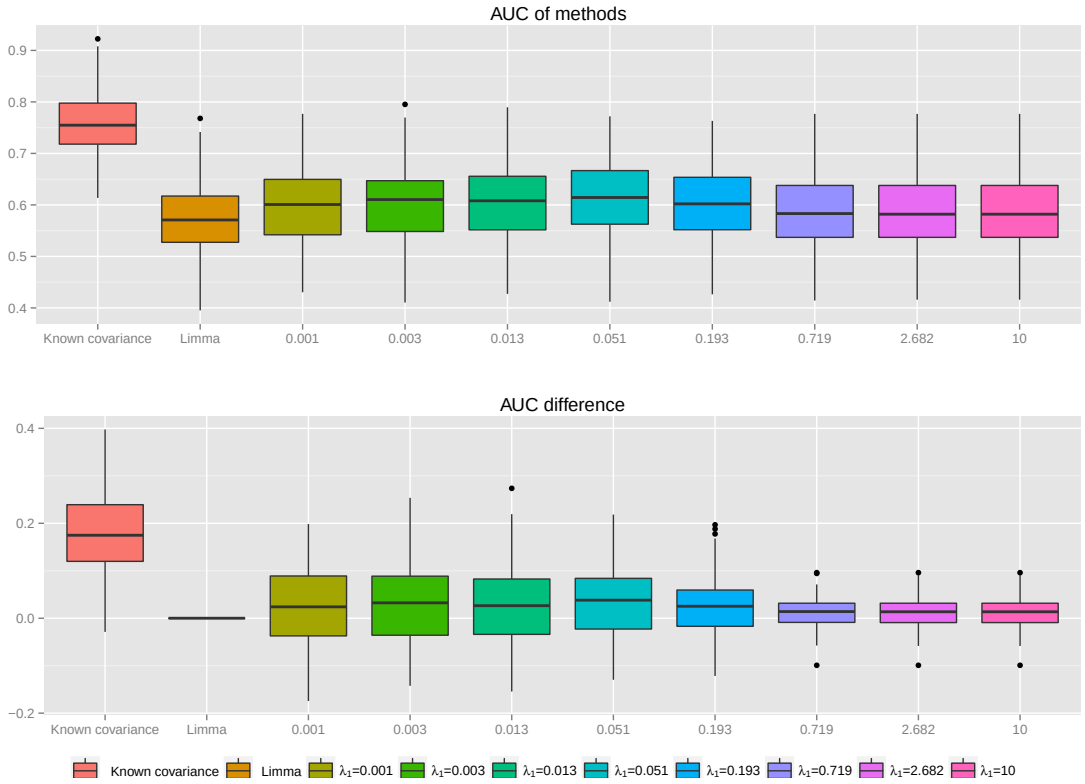
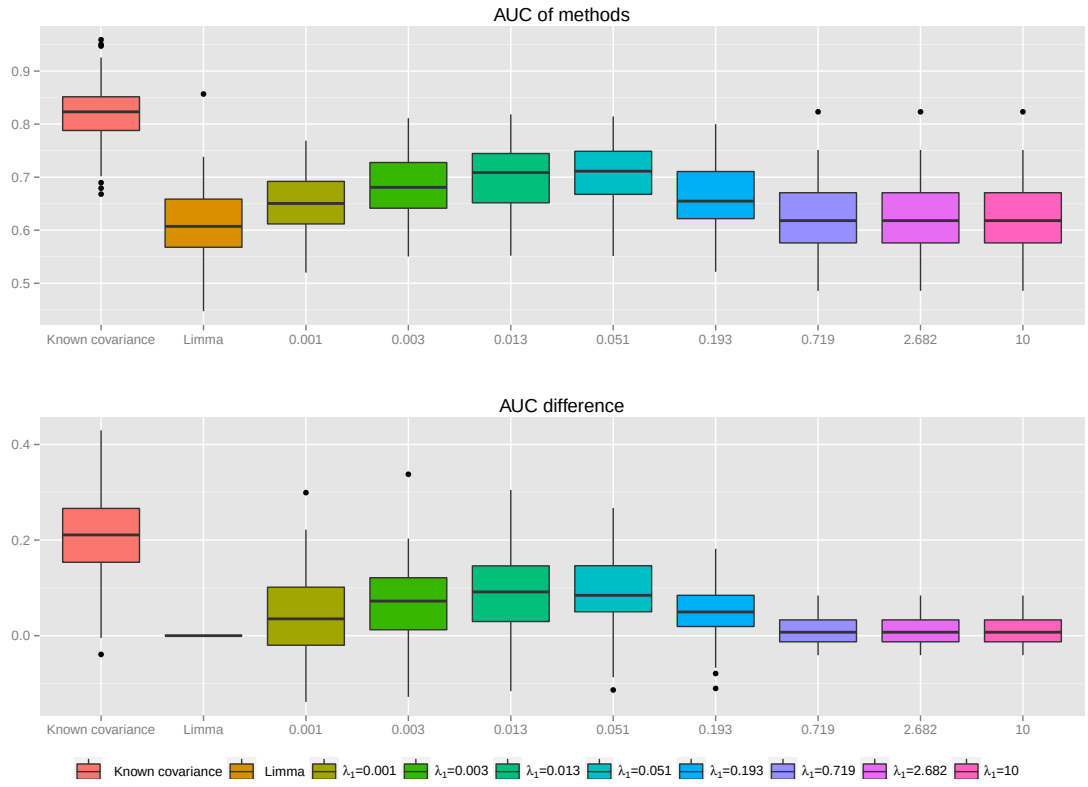
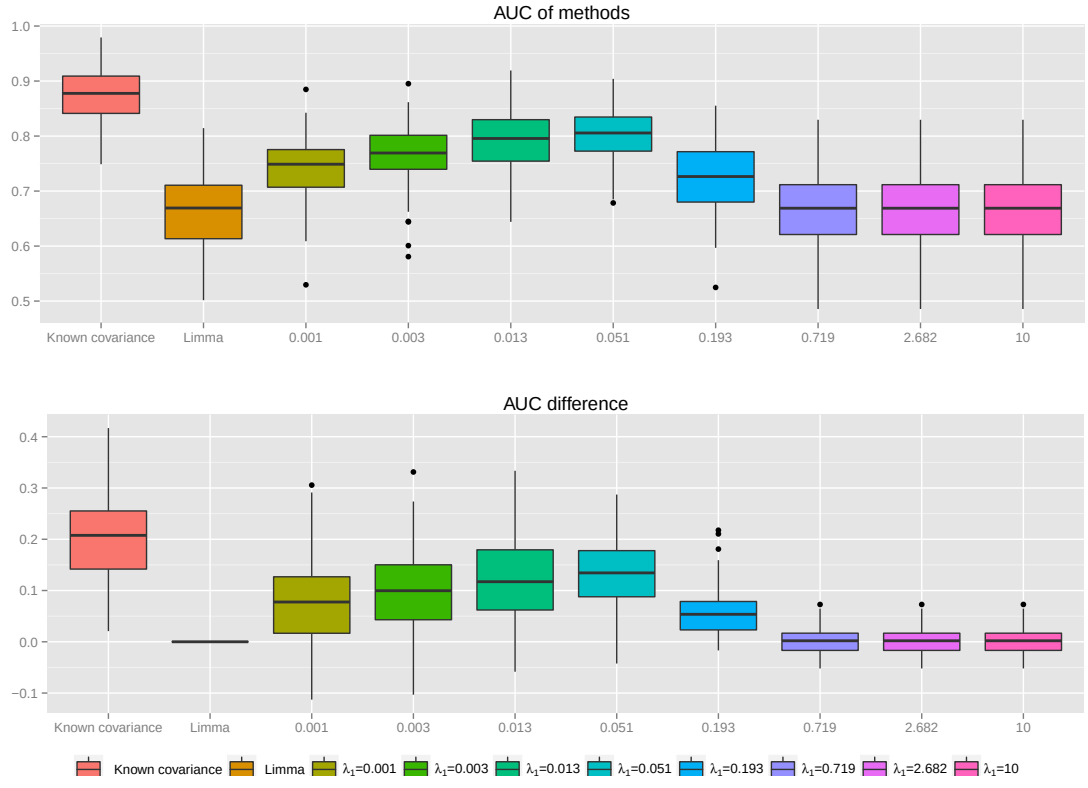


Figure 3.5: Scenario 3 with $n_k = 10$: no prior information about the covariance matrix. AUC comaprison between $limma$ method, our method with given covariance matrix (Scenario 1), our method in Scenario 3 for 8 values of λ_1 from zero to infinity. We are in the CLUSTER network setting. The number of observations n_k is 10 and the number of genes p is 50. The first figure is the AUC boxplot of all methods. The second figure is the difference between the AUC of all methods and the AUC of limma. Therefore, the second boxplot is zero.



(a) $n_k = 20$



(b) $n_k = 30$

Figure 3.6: Scenario 3 with $n_k = 20$ or $n_k = 30$: no prior information about the covariance matrix. AUC comaprison between $limma$ method, our method with given covariance matrix (Scenario 1), our method in Scenario 3 for 8 values of λ_1 from zero to infinity. We are in the CLUSTER network setting. The number of observations n_k are 20 in figure (a) and 30 in figure (b); the number of genes p is 50.

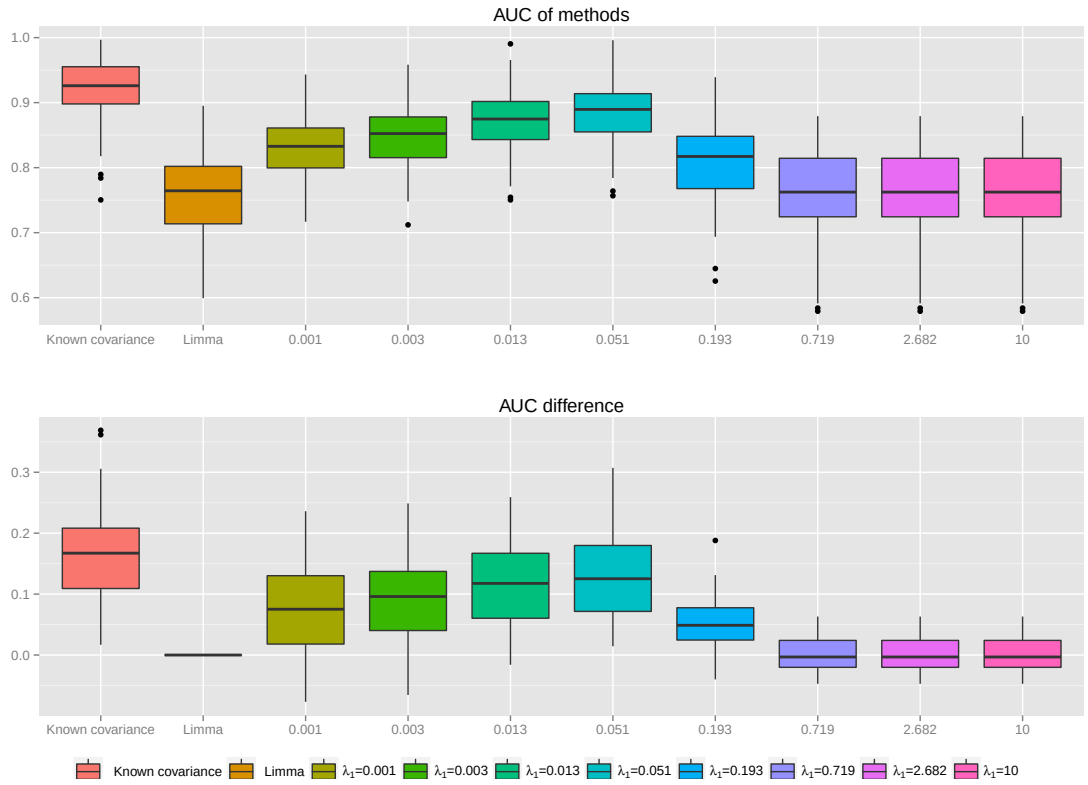


Figure 3.7: Scenario 3 with $n_k = 50$: no prior information about the covariance matrix. AUC comaprison between $limma$ method, our method with given covariance matrix (Scenario 1), our method in Scenario 3 for 8 values of λ_1 from zero to infinity. We are in the CLUSTER network setting. The number of observations n_k is 50 and the number of genes p is 50. The first figure is the AUC boxplot of all methods. The second figure is the difference between the AUC of all methods and the AUC of limma. Therefore, the second boxplot is zero.

For all choices of n_k, the first box (the result of our method in Scenario 1) is naturally always the best boxplot. It is the best result which we could hope with our method. In this third scenario, comparing our method with different values of λ_1 to $limma$ method is difficult when we have too few observations $n_k = 10$. There might be a small advantage to our method but it is not very significant. However, in other cases where $n_k = \{20, 30, 50\}$, there are always several values of λ_1 for which our method works really well compared to $limma$. In fact for $n_k = 50$, the best result of our approach is not that far from the result of our approach with given covariance matrix (the mean value of AUC is 0.85 compared to 0.92).

The value of λ_1 for which we get the best results is 0.051 and is intermediate. This is a good and expected result. From a technical point of view, recalling that our estimator is the

minimizer of the loss function

$$E(\beta,\theta) = \frac{1}{2}\sum_{k=1}^{K}\sum_{i=1}^{n_k}\sum_{j=1}^{p}\left(x_{ij}^k - \beta_j^k + \sum_{a\neq j}\theta_{ja}^k\beta_j^k - \sum_{a\neq j}\theta_{ja}^k x_{ia}^k\right)^2 + \lambda_1\sum_{k=1}^{K}\sum_{k=1}^{p}\sum_{a\neq j}|\theta_{ja}^k| + \lambda_2\sum_{\substack{k,l=1\\k<l}}^{K}\sum_{j=1}^{p}|\beta_j^k - \beta_j^l|.$$

It is clear that large values of λ_1 lead us to very sparse networks, and large enough λ_1 can lead us to an empty network. Recall that $limma$ assumes that all genes are independent, thus the network is empty. Therefore, we expect that results of our approach should be close to results of $limma$ when the value of λ_1 is large enough. This is also what we observe if we look at the boxplots for extreme values of λ_1 on the grid (0.719, 2.682 and 10) because these values of λ_1 lead us to an empty network. For any choice of n_k, we can see that these boxplots are very close to the boxplot of $limma$. In our simulations, we use a sparse but not empty network. Therefore, we expect that for intermediate values of λ_1, we recover part of the true network and this should help differential analysis.

This result is very satisfying for two reasons. First, it emphasizes again the importance of knowing the network for differential analysis. Second, it shows that this is the case even when we have no prior knowledge about the network. Of course these are only numerical simulations. We will illustrate in Chapter 4 the advantage of our approach on 2 real datasets.

2.2 Result for gene set analysis

In this section, we study the multivariate gene differential analysis problem.

2.2.1 Experimental set up

One of our competitor is the Hotelling-T^2 test. To perform the Hotelling-T^2 test, it is required that the total number of observations $n = \sum_{k=1}^{K} n_k$ must be greater than p. Therefore, in this experiment we set the number of genes p smaller than the sum of all n_k.
We choose $p = 20$, while all n_k are equal and take a value in the set $\{12, 15, 20\}$. With 3 options of n_k and 4 types of network, we consider a total of 12 cases. For each case, we made 3000 simulated datasets where the mean vector is changed between the two conditions $\left((\beta^*)^1 \neq (\beta^*)^2\right)$ and 3000 simulated datasets where the mean vector is not changed $\left((\beta^*)^1 = (\beta^*)^2\right)$. In all experiments, the similarity parameters μ is always set to 50% which means that in the case $\left((\beta^*)^1 \neq (\beta^*)^2\right)$, there is exact $p/2$ values of j such that $\left((\beta^*)_j^1 = (\beta^*)_j^2\right)$ for $j \in \{1, .., p\}$.

2.2.2 Competitors

We compare our method to two hypothesis testing methods. One is the Hotelling-T^2 test and one is the projection test of Jacob *et al* [13]. Recall that, when we perform Jacob et al's test, we have to choose the dimension (d) of subspaces in which the data is projected (see Chapter 1, Section 2.2.1.2). We will perform Jacob et al's test for three options of $d \in \{2, 3, 4\}$. Hence, we actually have three versions of this test. We will first compare our approach to the Hotelling-T^2 test then with Jacob et al's test.

2.2.3 Scenarios

we consider the three scenarios that we considered for the the univariate analysis case introduced in Section 2.1.3.

- **Scenario 1:** The covariance matrix is known.
- **Scenario 2:** The network is known.
- **Scenario 3:** We have no prior information about the network.

2.2.4 Priority list

In this multivariate differential analysis experiment, we have 6000 simulated datasets labelled by $\{1, 2, \ldots, 6000\}$. For each simulated dataset, instead of finding a priority list for each gene as in univariate analysis, we find only 1 fusion time. This value is obtained by the procedure described in Chapter 2, Section 4.2.1. This fusion time is labelled by a triplet (id, k, l), where id is the label of each simulated dataset and k, l are two conditions. For example, in a simulated data, if the fusion time takes value 1, it means that the minimal value of λ_2 to fuse the two estimators of $(\beta^*)^k$ and $(\beta^*)^l$ is 1. Then, the priority list is made by ordering these 6000 values decreasingly.
Similarly, for each of the other tests, the priority list is made by ordering the 6000 p-values increasingly.

Note that we are in the case where $K = 2$ tasks. Therefore, we have 1 value for each simulated dataset. In the general case with an arbitrary value for K, we perform pairwise comparison for each method. Hence, we obtain $K(K-1)p/2$ scores per simulated dataset. For instance, if $K = 3$, we have three fusion times per simulated dataset: one for the fusion time of the estimators of $(\beta^*)^1$ and $(\beta^*)^2$, one for the fusion time of the estimators of $(\beta^*)^1$ and $(\beta^*)^3$ and one for the fusion time of the estimators of $(\beta^*)^2$ and $(\beta^*)^3$.

2.2.5 ROC

Unlike the procedure to compute ROC and AUC in the univariate analysis, our procedure in the multivariate analysis only makes a ROC curve. The way we make ROC curves is fairly similar to what we did for the univariate analysis case. More precisely,

- We assign each triplet (id, k, l) to a point on the ROC curve. Then we consider all triplets having higher rank than (id, k, l) and itself in the priority list as detected triplets by the method.
- We then compute the coordinate of this point as:

$$\text{TPR}(id,k,l) = \frac{\text{Number of detected triplets that are true}}{\text{Number of triplets corresponding to a differential expression}},$$
$$\text{FPR}(id,k,l) = \frac{\text{Number of detected triplets that are wrong}}{\text{Number of triplets corresponding to a non differential expression}}.$$

From this, we make the ROC curve. In the next sections, we compare our method to each test methods in different scenarios.

2.2.6 Comparison between Hotelling-T^2 test and our method

2.2.6.1 Result with known covariance matrix. In this scenario, we want to compare our method to the Hotelling-T^2 test when the total information about the covariance matrix is given. In this scenario, Hotelling-T^2 is used with the given covariance matrix instead of the empirical covariance matrix. The results obtained depend on the network type. For band, cluster and hub networks, the results are similar. Results of cluster network are shown in Figure 3.8. Results of random network is different and shown in Figure 3.9.
Overall, our approach is better than the Hotelling-T^2 test. Clearer results are obtained for band, cluster and hub; and the hardest case is the random one in which we observe a clear difference only for sufficiently large n_k.

These results show that taking into account the covariance matrix could improve the multivariate differential analysis. However, depending on the type of network and the number of observations, the amplitude of the improvement could vary. With networks which do not have a solid structure such as random network, the improvement is only clear when the number of observations is large enough.

2.2.6.2 Result with known network. In this scenario, we want to compare our method and Hotelling-T^2 test when only the adjacency matrix of the precision matrix is given. In this scenario, the results obtained are only slightly different for different types of network. For

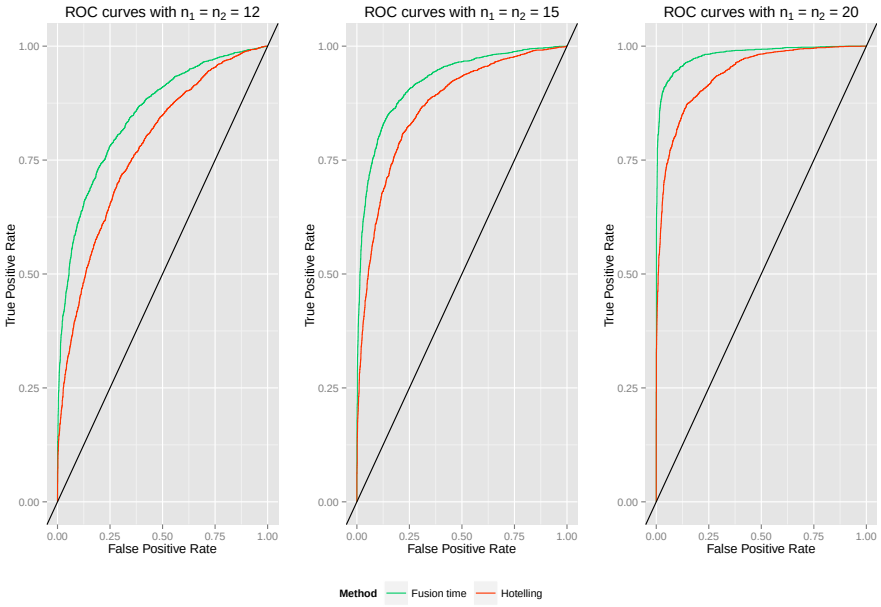


Figure 3.8: Scenarion 1: covariance matrix is given. ROC of two methods Hotelling-T^2 and Fusion time for HUB network. The number of observations n_k takes a value in the set $\{12, 15, 20\}$ and the number of genes is $p = 20$.

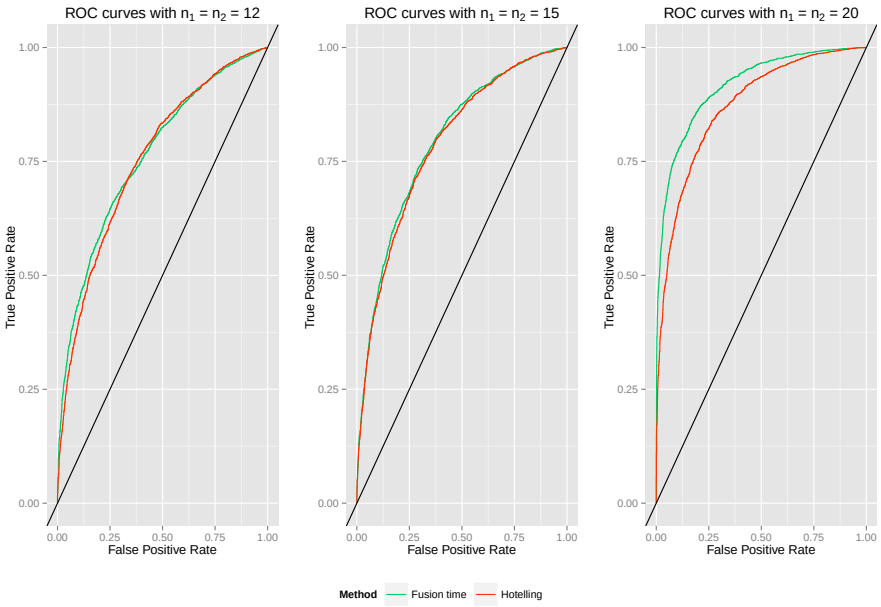


Figure 3.9: Scenarion 1: covariance matrix is given. ROC of two methods Hotelling-T^2 and Fusion time for RANDOM network. The number of observations n_k takes a value in the set $\{12, 15, 20\}$ and the number of is genes $p = 20$.

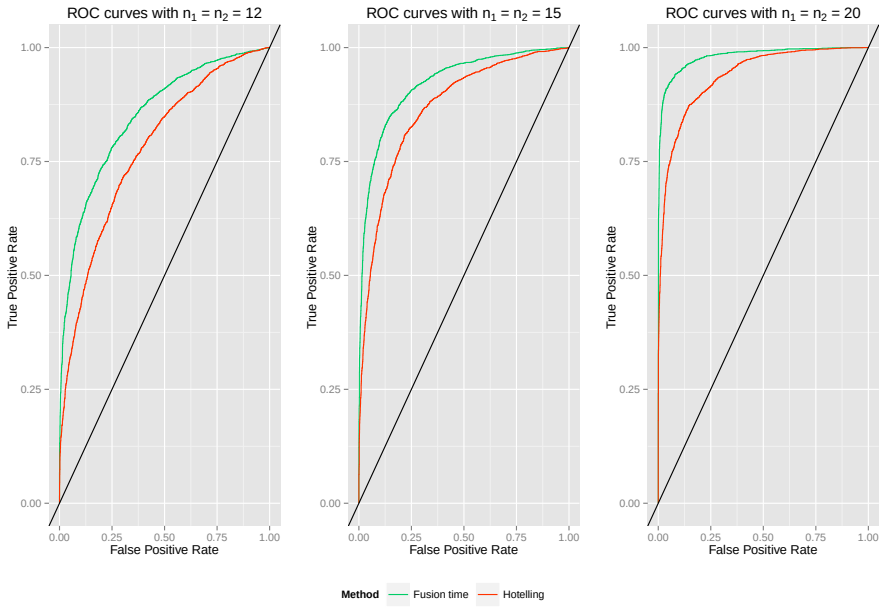


Figure 3.10: Scenario 2: network is given. ROC of two methods Hotelling-T^2 and Fusion time for CLUSTER network. The number of observations n_k takes a value in the set $\{12, 15, 20\}$ and the number of genes is $p = 20$.

all choices of networks and n_k, our method is still better than Hotelling-T^2 test. However, our method performs worse than in the first scenario. This is expected because we give our method less prior information about the covariance matrix. Results for cluster are shown in Figure 3.10. Results for band and hub networks are similar. As expected, results for random network is different and shown in Figure 3.11.

2.2.6.3 Results with no prior information about covariance matrix case.

In this scenario, we want to compare our method to Hotelling-T^2 test when we do not have any prior information about the network. We choose a list of values of λ_1 following a log scale to find the best result obtained by our method.

We also add ROC curve of our method when we know the covariance matrix as a control. For all type of network, results are similar. Results for cluster networks are shown in Figure 3.12.

Hotelling-T^2 test performs well in this scenario compare to our method. However, in the case $n_k = 12$ and $n_k = 20$, our method work slightly better than Hotelling-T^2 test for some values of λ_1. It emphasizes the fact that Hotelling-T^2 test work well when we have many observations but lose quickly its power in high dimensional setting. Compare to the given covariance matrix scenario, the Hotelling-T^2 test performs worse. This is expected as we give the method no information about the network. However, in general, the Hotelling-T^2 test is clearly better than our method. Although our method does not perform as we

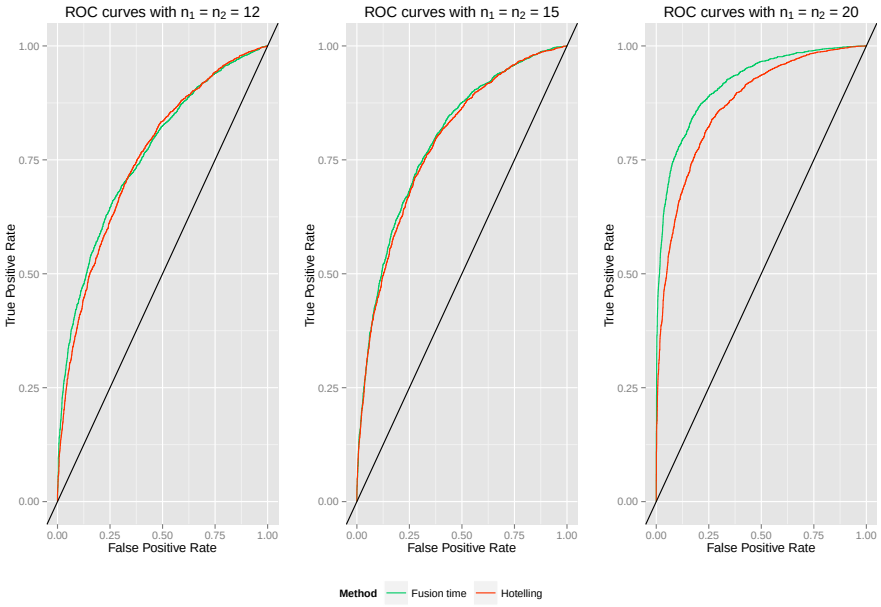


Figure 3.11: Scenario 2: network is given. ROC of two methods Hotelling-T^2 and Fusion time for RANDOM network. The number of observations n_k takes a value in the set $\{12, 15, 20\}$ and the number of genes is $p = 20$.

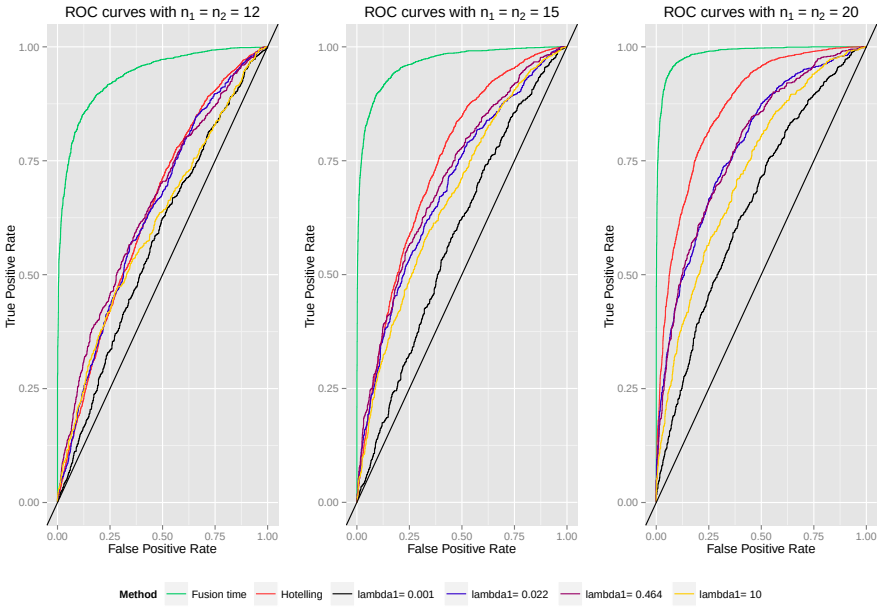


Figure 3.12: Scenario 3: No prior information about the network. ROC of our method with given covariance matrix, ROC of Hotelling-T^2 test and our method with different values of λ_1. The number of observations n_k takes a value in the set $\{12, 15, 20\}$ and the number of genes is $p = 20$.

expected, we still find some good signals. More precisely, in all cases, the best choice of λ_1 is an intermediate values on the grid. It suggests that our idea still works in this case, but the choice of λ_1 should be more sophisticated. For instance, we should take different values of λ_1 on different edges. Of course, a more sophisticated choice of tuning parameters requires more computations, but in some real problems, we may have prior information about the network. Therefore, the computations could be easier and our method may work much better.

Recall that Hotelling-T^2 test requires the total number of observations $n_1 + n_2 \geq p$. This type of data is not high dimensional data. It is well known that in omics datasets, the number of genes is usually much bigger than the total number of observations. Therefore, in the next part, we consider a scenario where we also have no prior information about the network, but the number of observations is such that $n_1 + n_2 < p$.

2.2.6.4 Results with no prior information about covariance matrix in high dimensional data. In this setting, we set the number of genes $p = 50$, while the number of observations n_k takes a value in the set $\{10, 20\}$. Due to these new values, Hotelling-T^2 test cannot be performed. Our goal in this setting is to measure the performance of our method in the high dimensional setting. As usual, we add the result of our method with given covariance matrix as a control. Results for four types of network are similar. The results of cluster network are shown in Figure 3.13.

When we have few observations $n_k = 10$, our method performs poorly for all choices of λ_1. However, if we increase the value of n_k to 20, the blue curve corresponding to $\lambda_1 = 0.02$ performs much better compared to the other curves. This is a good and expected result because it is an intermediate value.

2.2.6.5 Hotelling-T^2 test vs our method, the conclusion. Both Hotelling-T^2 test and our method perform better when we provide them more prior information. In the two first scenarios, when prior information about the covariance matrix are given to both methods, our method outperforms Hotelling-T^2 tests. In the third scenarios, when we have no prior information and many observations, Hotelling-T^2 outperform our method in most of the scenarios. However, when we have few observations, only our method can perform on the dataset and it works pretty well for some choices of λ_1.

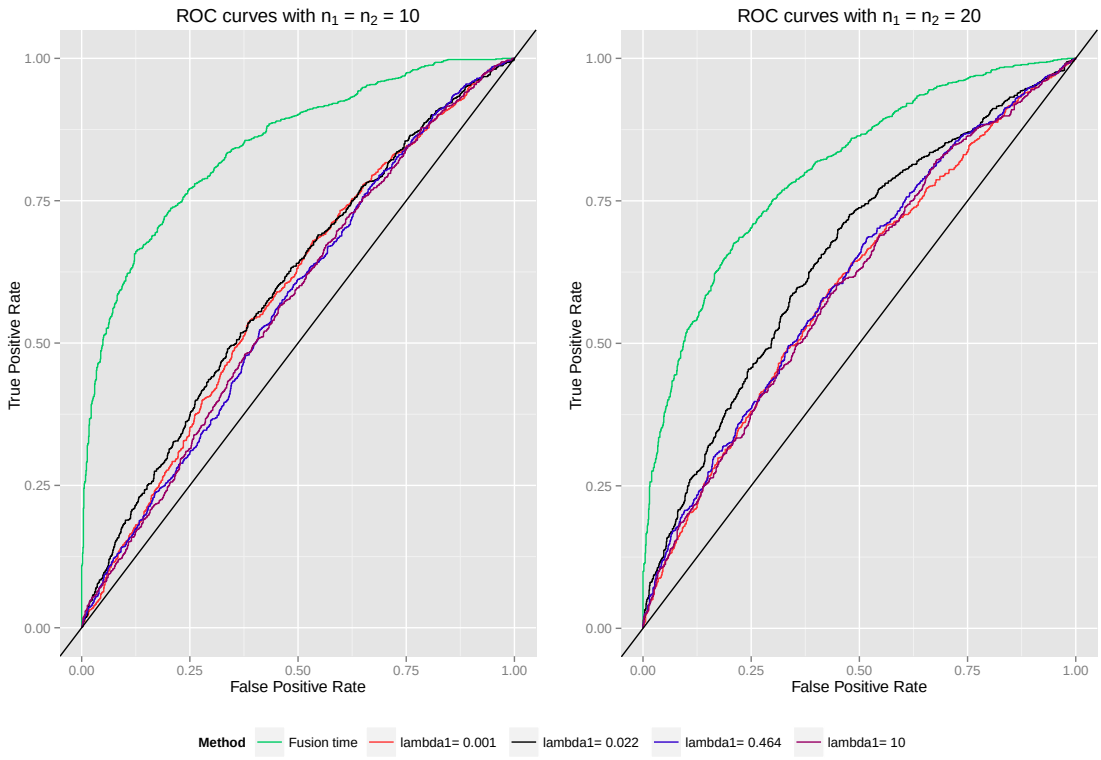


Figure 3.13: Scenario 3: No prior information about the network. ROC of our method with given covariance matrix and our method with different values of λ_1. The number of observations n_k takes a value in the set $\{10, 20\}$ and the number of genes is $p = 50$.

2.2.7 Comparison between Jacob et al's test and our method

The method of Jacob *et al* can only be performed when we know the covariance matrix (Scenario 1) or the network (Scenario 2). Therefore, in this comparison we only consider Jacob et al's test and our method in the two first scenarios. Recall that for Jacob et al's test, we choose $d \in \{2, 3, 4\}$, where d is the dimension of the subspace in which the data is projected. In this comparison, results of different types of networks are different.

- For band networks, results are shown in Figure 3.14. All methods work very well in this case. Our method is better than Jacob et al' test when the covariance matrix is provided, but it is worse when we only give it the network information.
- For cluster networks, results are shown in Figure 3.15. We obtain similar results to the band network case.
- For hub networks, results are shown in Figure 3.16. In this case, our method is always better than Jacob *et al*'s test. Moreover, the differences between the methods is clearer and bigger than in the band network and in the cluster network cases.

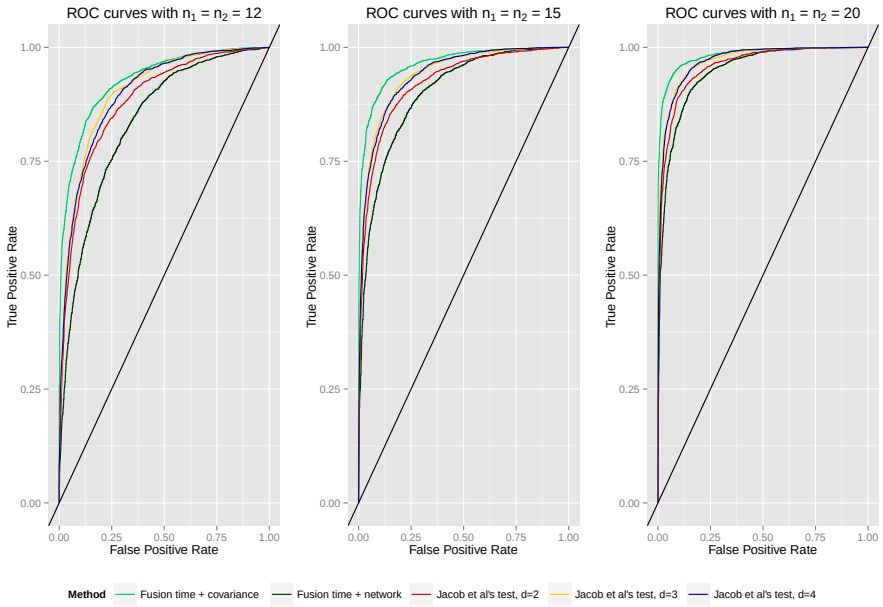


Figure 3.14: ROC curves of our method with given covariance matrix or given network methods and ROC curves of Jacob et al'test. Type of network is BAND. The number of observations n_k takes a value in the set $\{12, 15, 20\}$ and the number of genes is $p = 20$.

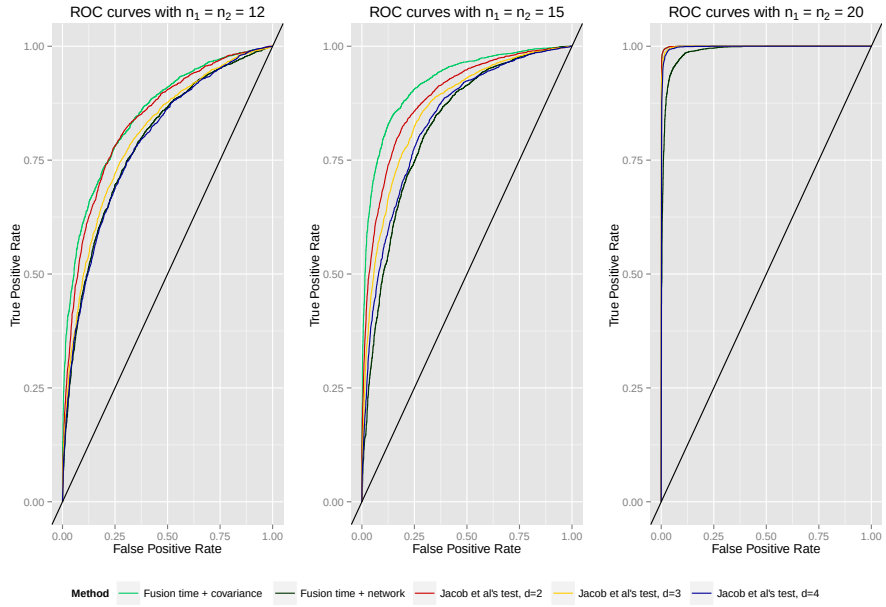


Figure 3.15: ROC curves of our method with given covariance matrix or given network methods and ROC curves of Jacob et al'test. Type of network is CLUSTER. The number of observations n_k takes a value in the set $\{12, 15, 20\}$ and the number of genes is $p = 20$.

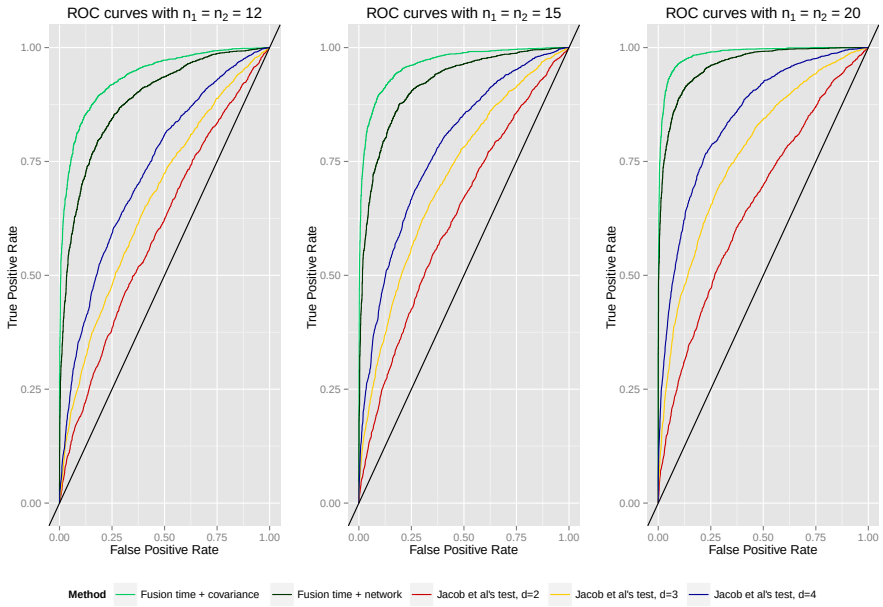


Figure 3.16: ROC curves of our method with given covariance matrix or given network methods and ROC curves of Jacob *et al*'s test. Type of network is HUB. The number of observations n_k takes a value in the set $\{12, 15, 20\}$ and the number of genes is $p = 20$.

- For random networks, results are shown in Figure 3.17. We obtained similar results to the hub network case. Our method is clearly better than Jacob *et al*'s test. However, as expected, all methods do not work as good as in the other network case. It emphasizes one more time that the random network case is a hard case.

2.2.7.1 Jacob *et al*'s test vs our method, the conclusion. Depending on the type of network, our method could be better or worse than Jacob *et al*'s test. However, in the case where our method performs worse, the results obtained are close to the results of Jacob *et al*'s test. On the other hand, when our method performs better, the differences between the methods is larger. Furthermore, the approach of Jacob *et al* requires the tuning of an extra parameters d. If we choose d too small, the sub-space could not capture important information of the data which leads to a poor differential analysis result. However, if d is too large, we fall again into a high-dimensional setting. To our best knowledge, the choice of d is still an open question.

2.3 Results in case of having more than 2 conditions

In this setting, to perform the experiments on the univariate analysis and the multivariate analysis, we actually do it pairwise on conditions. Then, we combine all the results. Hence, the results obtained is not different compared to the case where $K = 2$.

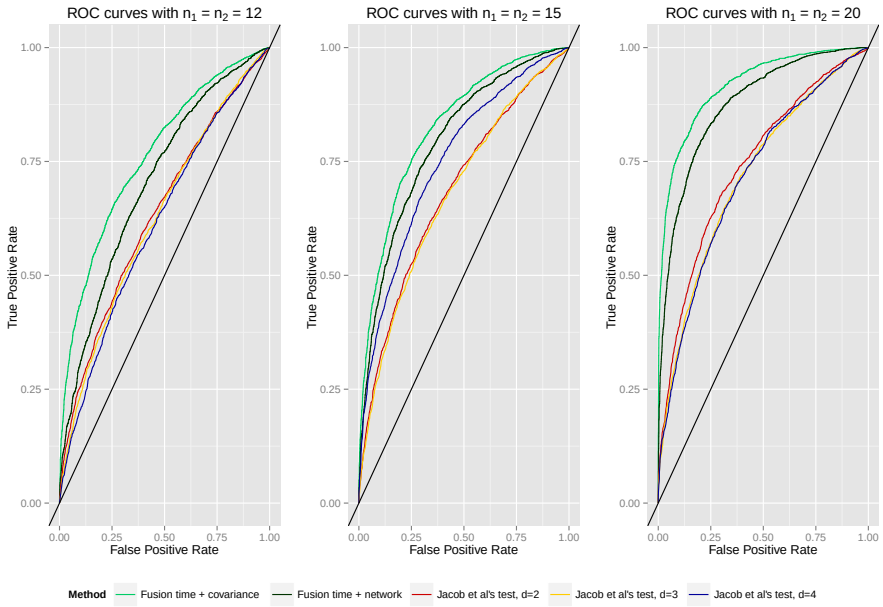


Figure 3.17: ROC curves of our method with given covariance matrix or given network methods and ROC curves of Jacob *et al*'s test. Type of network is RANDOM. The number of observations n_k takes a value in the set $\{12, 15, 20\}$ and the number of genes is $p = 20$.

3 Network inference

3.1 Inference network when having 2 conditions

3.1.1 Experiment set up

In the experiment, we choose $p = 50$, while all n_k are equal and take a value in the set $\{5, 10, 20\}$. The similarity parameters μ is always set to 50% in all experiments. With 3 options of n_k and 4 types of network, we consider 12 cases. For each case, we create 100 simulated datasets.

3.1.2 Competitors

We compare our method to the graphical lasso method. The graphical lasso is used on centered data. Therefore, we create two scenarios. First, we have no prior information about the mean vector. Therefore, to use the graphical lasso, we normalize data with the empirical mean vector. Second, we know the true value of the mean vector. Hence, we can use the graphical lasso on data normalized by the true mean vector.
We expect that results of the first approach are worse than results of the second approach. Results of the two approaches can also be considered as a reference and we expect that results of our method are in between results of the two scenarios.

3.1.3 Priority list

In this experiment, we choose a grid of 8 values for λ_2 following a log scale. For each value of λ_2 in the grid and for each simulated dataset, we order the edges of the network in a priority list. For our method, the priority list is made by ordering shrinking times of edges in the network. The shrinking times are created by the procedure described in Chapter 2, Section 4.2.2. Each shrinking time values is labelled by a triplet (j, a, k), where j, a are two genes and k is a condition. For example, if the shrinking time of (j, a, k) is 1, it means that the minimal value of λ_1 to shrink to zero the estimator of $(\theta^*)^k_{ja}$ is 1. The bigger the shrinking time is, the less probability $(\theta^*)^k_{ja}$ shrinks to 0. Therefore, if we have two triplets (j_1, a_1, k_1) with rank r_1 and (j_2, a_2, k_2) with rank r_2 and $r_1 < r_2$, then the shrinking probability of $(\theta^*)^{k_1}_{j_1a_1}$ is smaller that the shrinking probability of $(\theta^*)^{k_2}_{j_2a_2}$. Consequently, if our method detects one edge $(\theta^*)^{k_2}_{j_2a_2} \neq 0$, then automatically it will detect the edge $(\theta^*)^{k_1}_{j_1a_1} \neq 0$. In other words, if (j_2, a_2, k_2) is detected, then (j_1, a_1, k_1) is also detected. For the two approaches with graphical lasso, we create a similar priority list.

3.1.4 ROC and AUC

Again, given a priority list of triplets, it is possible to make the ROC curve for each method. Each triplet corresponds to a point on the ROC curve. The horizontal coordinate of this point is the false positive rate (FPR), and the vertical coordinate of this point is the true positive rate (TPR). These coordinates are computed by the following procedure:

- For each triplet (j, a, k), we consider all triplets in the priority list having higher rank than (j, a, k) and itself as detected triplets by the method.
- We then compute the coordinate of this point as

$$\text{TPR}(j, a, k) = \frac{\text{Number of detected triplet that are true}}{\text{Number of edges}},$$
$$\text{FPR}(j, a, k) = \frac{\text{Number of detected triplet that are wrong}}{\text{Number of blank edges}},$$

  where (j, k, l) is a blank edge if $(\theta^*_{aj})^k = 0$.

From this, we make the ROC curve. Because we have 100 simulated data, we obtain 1200 ROC curves. From them, we can make 12 boxplots for the 12 methods. Result of the graphical lasso approach on normalized data by the true mean (resp. empirical mean) is the first boxplot (resp. the second boxplot). Result of the graphical lasso approach on normalized data by the empirical mean is the second box. Results of our methods with 8 different values of λ_2 from zero in infinity are the 8 last boxplots.

3.1.5 The results

For all simulated data with different sizes and types of network, results are very similar. Results of the cluster network are shown in Figure 3.18.

Obviously, the graphical lasso with true mean is better than all others, while our method is just equivalent to the graphical lasso with empirical mean for some values of λ_2. In fact, the results of the graphical lasso with the true mean is not too far from the others. It means that in this setting, the value of the mean vectors does not have a big role regarding network inference.

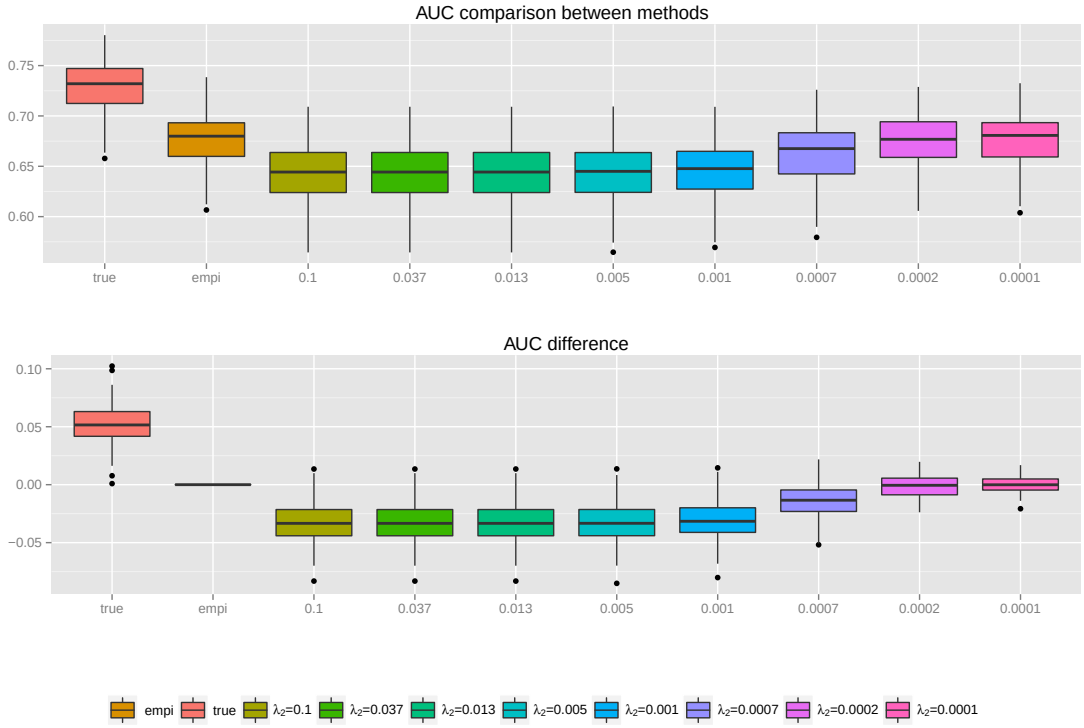


Figure 3.18: AUC of different methods in band network case. Graphical lasso approach with data normalized by the true mean (first box) have the best result. Results of other methods are very close. The first figure is AUC values, the second figure is AUC values of all method minus AUC of Graphical lasso with data normalized by the empirical mean (second box). The number of observations is $n_k = 20$. The number of genes is $p = 50$.

3.2 Results when having more than 2 conditions

Here we study the network inference problem when we have more than 2 conditions. In this setting, we found that the mean vector becomes more important for network inference.

3.3 Role of mean vectors in inferring network

To get a clearer picture about the impact of mean vectors on network inference when the number of conditions K vary, we try different values of $K \in \{2, 5, 8, 10\}$ with $p = 50$, while n_k depends on the number of tasks. Namely,

- If $K = 2$, we choose all $n_k = 20$.
- If $K = 5$, we choose all $n_k = 8$.
- If $K = 8$, we choose all $n_k = 5$.
- If $K = 10$, we choose all $n_k = 4$.

Hence, whatever the number of tasks, the total number of observations is always 40. For each type of network, and each value of K, we create 100 simulated datasets. Results for all types of network are similar. Results for cluster network case are shown in Figure 3.19.

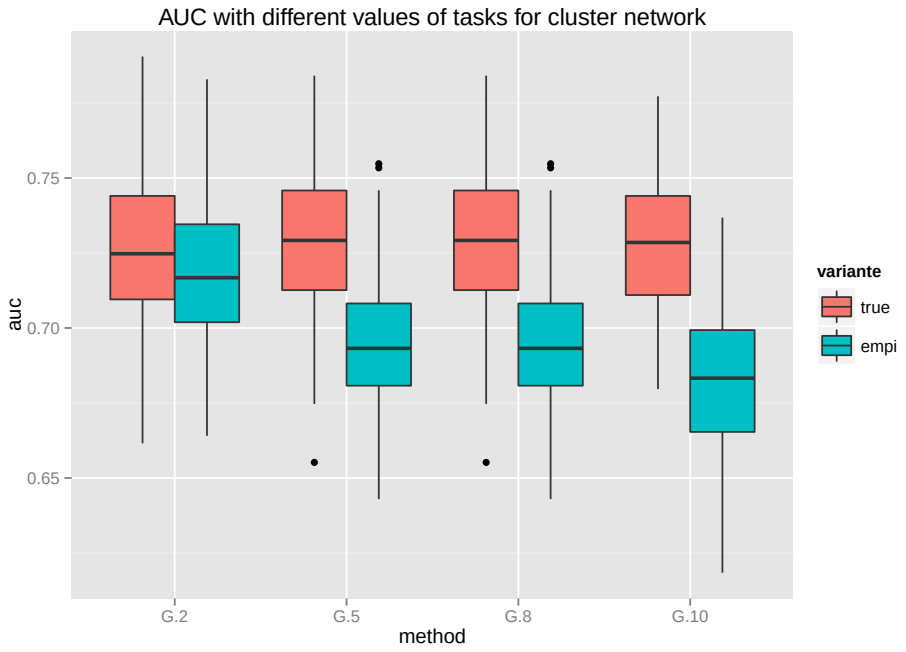


Figure 3.19: Results of graphical lasso with true and empirical means for different values of K. The number of tasks takes a value in the set $\{2, 5, 8, 10\}$. The number of total observations is $n = 40$ for all choices of K.

Clearly, when we increase the number of conditions K, the difference between the graphical lasso with the true mean vector and the graphical lasso with the empirical mean vector is larger. For all choices of K, results of the graphical lasso with the true mean vector are similar, because the number of total observations is always 40. However, when we use the graphical lasso with the empirical mean vector, the results obtained become much worse.

The bigger the number of conditions, the less accurate the empirical estimator of mean vector since we have less observations for each task. In figure **??**, we can see that mean vectors has some impacts on network inference experiments. However, the effect of estimating the mean on the estimation of the network seems small compare to the effect of estimating the network on the estimation of the mean as illustrated in differential analysis experiments. I think that the main reason is because of the number of parameters. The total parameters of mean vectors is $K * p$ while the total number parameters of network is $K * p * (p-1)/2$. Therefore, maybe we need to increase K to obtain clearer impacts of mean vector on the network.

3.4 Inference network when having 10 conditions

Because the difference between using the true mean vector and the empirical mean vector is very clear when we choose K big enough, we will consider the case where $K = 10$. We choose a number of genes $p = 50$ and all n_k equal to 2. Among all values of λ_2 on the grid, we make the boxplot for the best choice of $\lambda_2 = 0.1$. The results for all types of network are similar. Results for cluster network are shown in Figure 3.20. Results for other types of network are shown in Appendix A.

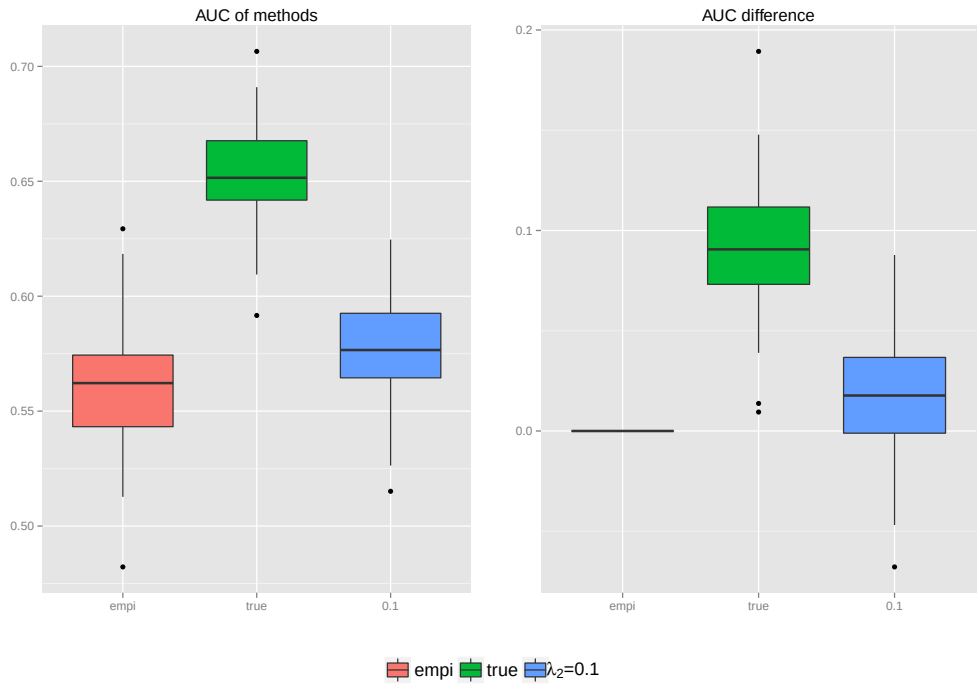


Figure 3.20: AUC of different methods in CLUSTER network case. Graphical lasso approach with data normalized by the empirical mean (first boxplot), Graphical lasso approach with data normalized by the true mean (second boxplot), and our method with the best choice of $\lambda_2 = 0.1$ (third boxplot). The total number of observations is $n = 40$. The number of genes is $p = 100$.

With a good choice of λ_2, our method works better than the graphical lasso with the empirical mean. Although the difference is only about 3%, in term of network inference, we think that it is a good result. In genomic datasets, it contains thousand of genes. Even when we only consider a sub network of hundreds of genes, the number of possible edges is about ten thousands. Hence, our approach gives a better estimator result for 300 edges.

4 Conclusion of Chapter 3

In this chapter, we try to evaluate the performance of our method for differential analysis and network inference. In my opinion, there is no fair comparison. It is always possible to find a setting in which one method performs better than another. However, we tried to simulate typical settings and simple scenarios. From the results obtained, we expect that our method could work well on high dimensional data. Moreover, it could improve both differential analysis and network inference compared to previous methods.

Chapter 4

Application to real data

The goal of this chapter is to illustrate how our method work on real datasets. We consider two publicly available datasets, one on breast cancer (Guedj *et al* [34]) and one on Arabidopsis thaliana (CATdb, Gagnot *et al* [35]). Using these two datasets, we try to illustrate that looking at both the mean transcription level and the network of genes using our model is useful and could lead to interesting biological conclusions. In particular, genes whose behaviour change both in mean and network between two biological conditions could be very interesting.

1 General goals and set up

1.1 Our goal

Our model could be used to infer both mean expression and gene interactions. When comparing two biological conditions we can thus hope to detect two type of differences: changes in the mean or in the gene network. From this, we define four types of genes as illustrated by the Table 4.1.

	Change in Mean expression	No change in Mean expression
Change in Network	MN (change in both Mean and Network)	0N (no change in Mean but change in Network)
No change in Network	M0 (Change in Mean expression but no change in Network)	00 (no change in both Mean and Network)

Table 4.1: 4 groups of genes categorized by our method.

1.2 Statistical tools

For both datasets, we have two biological conditions. Therefore, we denote the dataset $\mathbf{X} = (\mathbf{X}^1, \mathbf{X}^2)$. Assume that $\mathbf{X}^1 \sim \mathcal{N}((\beta^*)^1, (\Sigma^*)^1)$, $\mathbf{X}^2 \sim \mathcal{N}((\beta^*)^2, (\Sigma^*)^2)$. In the two conditions, the vectors $(\beta^*)^1, (\beta^*)^2$ are the mean expression vectors of genes, while the two

covariance matrices are $(\Sigma^*)^1, (\Sigma^*)^2$ and their inverse matrices can be interpreted as the interaction between genes. By estimating parameters $\big((\beta^*)^1, (\beta^*)^2, (\Sigma^*)^1, (\Sigma^*)^2\big)$, we aim to build measures in order to evaluate the changes between the two conditions at the gene level.

1.2.1 Building the measures

In our model, we need to select a value for both λ_1 and λ_2. This is a different question. Therefore, we consider a different approach to overcome this issue. Essentially, we measure the differences between estimators in different condition for each fixed couple (λ_1, λ_2). Then we integrate these differences over a large grid of λ_1 and λ_2. This has the disadvantage of being computationally intensive. However, we do not have to select values for tuning parameters (λ_1, λ_2). More precisely, our procedure is as follows:

1. We create a two dimensional grid of (λ_1, λ_2).
2. For each couple (λ_1, λ_2), we perform the ACS algorithm (Chapter 2, Section 2.1) to estimate $\big((\beta^*)^1, (\beta^*)^2, (\Sigma^*)^1, (\Sigma^*)^2\big)$, then we measure the mean change and the network change of each gene (we will explain in more details this step in the next part).
3. The overall change of each gene is the integration of results obtained with each couple (λ_1, λ_2).

In the next sections, we give more details about each step in this procedure.

1.2.1.1 Building the grid.

For our approach, the denser the grid, the better it is. However, we do not want to consider a too dense grid due to the running time. Therefore we make a grid $\mathcal{G}$ which is:

- Dense enough. A good grid should capture most status of the network and the fusion of the mean vectors. For instance, the status of the network should be from almost empty (no edge) to complete (full edge). Similarly, the status of the fusion should be from complete fusion (all genes are fused) to almost no fusion (no genes are fused).
- Avoid too dense graphs (corresponding to small λ_1) because this is not expected (we expect a sparse network) and the running time is particularly long for these values.

Following these principles, we choose an 8 by 8 grid for each dataset. For both datasets, we fixed the exact range of the grid manually looking at how dense the network was depending on the values of λ_1.

1.2.1.2 Building measures with fixed values of (λ_1, λ_2)

1.2.1.2.1 Mean change measure. For a fixed value of (λ_1, λ_2) and a gene j, we have an estimation of its mean expression for the two conditions. We measure the change in the mean expression of gene j as the absolute difference:

$$\textbf{l1.fused}(j, \lambda_1, \lambda_2) = |\hat{\beta}_j^1 - \hat{\beta}_j^2|.$$

We could also have considered other measures such as l_0 or l_2 based measures:

$$\textbf{l0.fused}(j, \lambda_1, \lambda_2) = |sign(\hat{\beta}_j^1 - \hat{\beta}_j^2)|,$$
$$\textbf{l2.fused}(j, \lambda_1, \lambda_2) = (\hat{\beta}_j^1 - \hat{\beta}_j^2)^2.$$

In practice, we found that results on mean change were not too dependent on this choice. Hence, we use only the l_1-based measure.

1.2.1.2.2 Network change measures. Similarly, we measure the change in network of gene j by:

$$\textbf{l1.shrink}(j, \lambda_1, \lambda_2) = \sum_{a=1, a\neq j}^{p} |\hat{\theta}_{ja}^1 - \hat{\theta}_{ja}^2|.$$

For the network, we also consider the l_0 version as it gives substantially different results in practice.

$$\textbf{l0.shrink}(j, \lambda_1, \lambda_2) = \Big| \sum_{\substack{a=1 \\ a\neq j}}^{p} |sign(\hat{\theta}_{ja}^1)| - \sum_{\substack{a=1 \\ a\neq j}}^{p} |sign(\hat{\theta}_{ja}^2)| \Big|.$$

Overall, the network change measures for gene j will be higher or lower depending if the interaction of gene j with other genes varies a lot or not.

1.2.1.3 Integral Measures.

The integral measure is the sum of all measures corresponding to the couples (λ_1, λ_2) on the grid. In details, for each gene j, we have three measures:

$$\textbf{l1.fused.integration}(j) = \sum_{(\lambda_1, \lambda_2) \in \mathcal{G}} \textbf{l1.fused}(j, \lambda_1, \lambda_2),$$
$$\textbf{l1.shrink.integration}(j) = \sum_{(\lambda_1, \lambda_2) \in \mathcal{G}} \textbf{l1.shrink}(j, \lambda_1, \lambda_2),$$
$$\textbf{l0.shrink.integration}(j) = \sum_{(\lambda_1, \lambda_2) \in \mathcal{G}} \textbf{l0.shrink}(j, \lambda_1, \lambda_2).$$

The three measures will be used to interpret our results in a two dimensional graph.

1.2.2 Discussion about the measures

For the network inference, we proposed two difference measures based on either l_0 norm or l_1 norm. We believe that they do not give exactly the same information. Using l_0 based measure, we put more emphasis on the network topology. We want to identify genes whose neighbors have change between the two conditions. This measure is interesting but we might miss important variation in the amplitude of the interactions. In order to have more specific detections based on these quantities, we also consider an l_1 based measure.

In the next sections, we will study two real datasets. We start with breast cancer data.

2 Breast cancer data

The breast cancer dataset was well studied in [34]. Furthermore, several papers have been published on breast cancer and genes interactions is relatively well characterized (e.g [36],[37],[38]). With our method, we expect to replicate these well-known results and maybe find some other interesting genes.

2.1 Biological context

2.1.1 Breast cancer

Breast cancer is a type of cancer developing from breast tissue. There are many causes of breast cancer involving genetic, environment, nutrition. One of the genetic reason is involving the female sex hormone called oestrogen. Oestrogen is responsible for mediating breast development. Many breast cancers rely on supplies of the hormone oestrogen to grow. Oestrogen can control the procedure by activating oestrogen receptors (ERs). ERs are a group of proteins. Once activated by oestrogen, the ERs bind to DNA in the nucleus and regulate the activity of some genes. Hence, ERs are also called transcription factors. ERs are encoded either by the gene ESR1 or the gene ESR2, resulting in two forms ERα and ERβ. In about 70% of breast cancer cases, ERs are over-expressed. The breast cancer cases are referred as "ER-positive". The rest are referred as "ER-negative".

Many analyses have been done on breast cancer and the goal of these analyses are very different. One typical problem people want to study is to characterize the differences between different groups of breast tumours, typically ER- and ER+. Looking at the mean expression, one can get thousands of differentially expressed genes. Biological interpretation of so many differences is difficult. Our hope is that looking at the network will help to pin-point important genes or important interactions that are changing between the two conditions.

2.1.2 Description of the dataset

The dataset [34] comprises 537 primary breast cancer transcriptomes on Affymetrix U133-Plus2.0 arrays. For each sample, the expression level of 54,675 genes are measured. Among these 537 tissues, 375 tissues are ER-positive and 162 tissues are ER-negative. The gene expression was normalized by Robust Multi-array Average (RMA) method [39] .

However, we do not consider the whole set of genes, but just a small subset for two reasons. First, running the code on the whole gene set is possible but it takes a very long running time. Second, Verzelen [30] shows that inference of a network on so many genes is not possible. Because of these practical and theoretical reasons, we only study a set of 200 genes.

This subset of genes contains the 160 genes with the highest variance and 40 genes chosen randomly. We make 10 subsets of gene like this. All subsets share the same 160 top genes, but the 40 remaining genes are not the same. On the one hand, taking the most variant genes is fairly common in gene expression analysis, typically to perform sample clustering. Therefore, we fix the list of 160 highest variance genes in all subsets. Hopefully, genes with high variance are biologically relevant. On the other hand, by selecting some genes at random, we do not expect enrichment for biologically relevant genes. In some sense, the scores which we get with those random genes can be used as a reference or a control.

Another well-known approach for selecting genes is "lossy screening" [40] but we do not consider it here.

2.2 Set up and results

2.2.1 Building the grid

In our R code, we create an 8×8 grid. The values of λ_1 and λ_2 are chosen by the R functions:

$$lambda1.list \leftarrow 10\text{^}seq(0, log10(1e-2), len = 8)$$
$$lambda2.list \leftarrow 10\text{^}seq(-2, log10(1e-4), len = 8)$$

In more details, we have

$$lambda1.list = \{1,\ 0.51,\ 0.26,\ 0.13,\ 0.07,\ 0.03,\ 0.019,\ 0.01\},$$
$$lambda2.list = \{0.01,\ 0.005,\ 0.002,\ 0.001,\ 0.0007,\ 0.0003,\ 0.00019,\ 0.0001\}.$$

With this grid, we capture most statuses of the network and the fusion status of the mean vectors. In details, all genes are fused when λ_2 equals 10^{-2}; while half of the genes are fused when λ_2 equals 10^{-4}. Regarding the network aspect, we capture most status of θ^1, θ^2 from all their elements equal zero (no edge) to 10266 edges presence over $p \times (p-1)/2$ = 19900 possible edges. It is about 51% of edge presence.

2.2.2 Results

We apply our strategy to the ten subsets of genes. The obtained results are very similar. The result of one subset is shown in Figures 4.1. Results of some other subsets are in Appendix B. In each figure, I split the genes in four categories: 00, M0, 0N and MN as illustrated in Table 4.1. Indeed, choosing important genes is also a selection problem. In this context, because we knew some important genes such as ESR1, we decide to select genes whose measurements are close to the measurements of those important genes. We have several comments on the results:

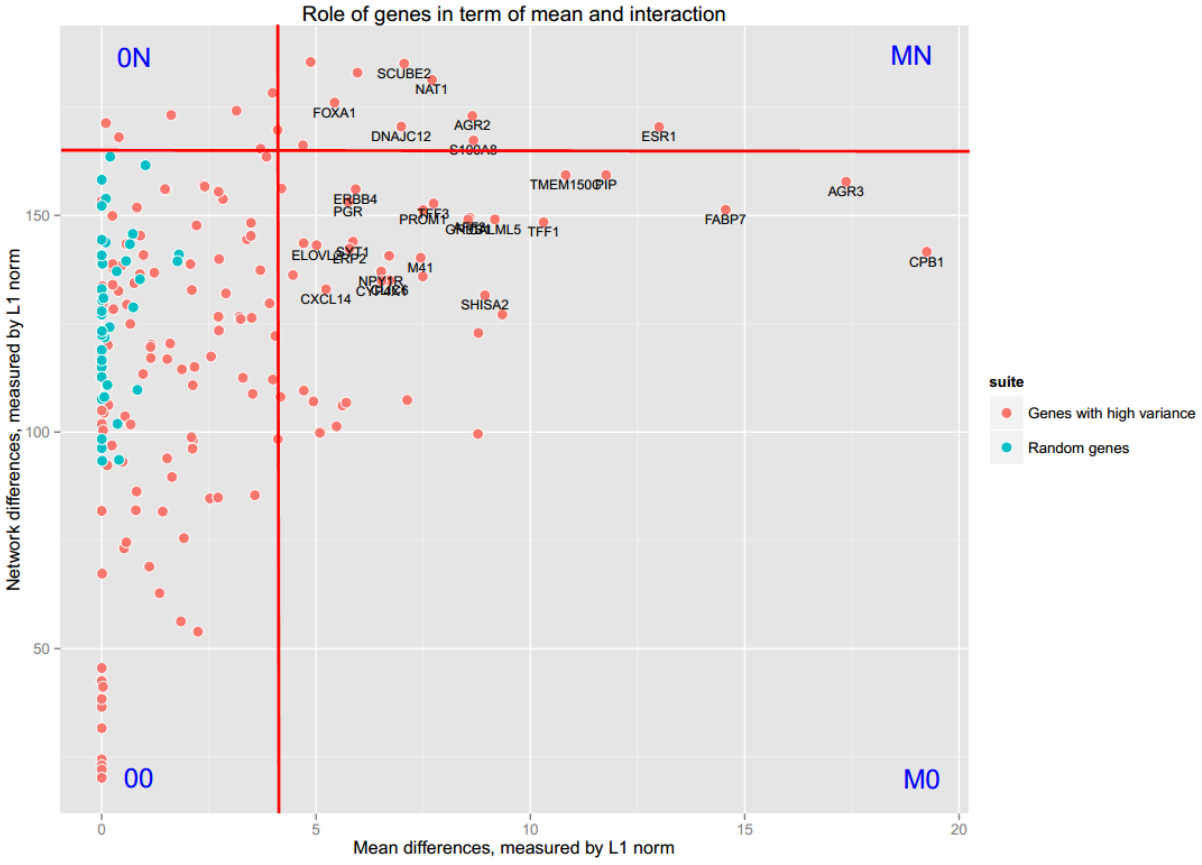


Figure 4.1: The roles of genes evaluated by different types of measure.

1. Most of random genes are concentrated on the left of Figure 4.1. Although their network measurement is quite high, they almost do not change their mean expression between the two conditions. The concentration of random genes shows their similar behaviors and guarantees that they do not affect too much the genes on the top right of the figure (which we think that they are important genes). However, we detect several highly variant genes in the bottom left of the figure. It suggests that, some highly

variant genes may not be important in both term of mean expression and interaction.

2. It is not surprising that ESR1 is one of the most significant genes in both terms (mean and interaction). Besides, we also detect
   - AGR3 which is significantly associated with oestrogen α [36].
   - FABP7 which is an inhibitor of proliferation of breast tumour cells [37].
   - CPB1 about which we do not have any biological information.
3. In term of network interaction, the most significant genes are:
   - SCUBE2 which is a breast tumour suppressor [38].
   - NAT1 which have a positive correlation with ER+ [41].
   - FOXA1 and AGR2 which are involved in ER regulation [42].
4. Denote the network score of gene j in the condition k as

$$\textbf{network.score}(j,k) = \sum_{a=1,a\neq j}^{p} (|\hat{\theta}^k_{ja}|).$$

   We found that important genes tends to make more interactions with other genes when ER appears. This change is shown in Figure 4.2.

The fact that our approach is able to pint-point some well-known regulators and interaction in breast cancer data is very satisfying and suggest that indeed looking at both mean and variance might be a key to investigate new genomic datasets.

In the next section, we use our approach again on the Arabidopsis thaliana dataset.

3 Arabidopsis thaliana data

3.1 Biological context

3.1.1 Arabidopsis thaliana

Arabidopsis thaliana is a plant which has been widely studied in genetics and genomics. This plant is small (25 cm tall), has a short life cycle (60 days) and a fairly small genome (about 135 megabase pairs). It is a model organism.

Our study is a part of a project of the team "Genomic networks" which started in 2010 at INRA. The main goal of this project is to identify biological functions of genes in Arabidopsis thaliana. Due to the huge number of genes, it is impossible to consider all of them at the same time. Therefore, a priority list of genes should be given before we start an in-depth study for a subset of genes. Our goal is to propose a priority list of genes by statistical methods.

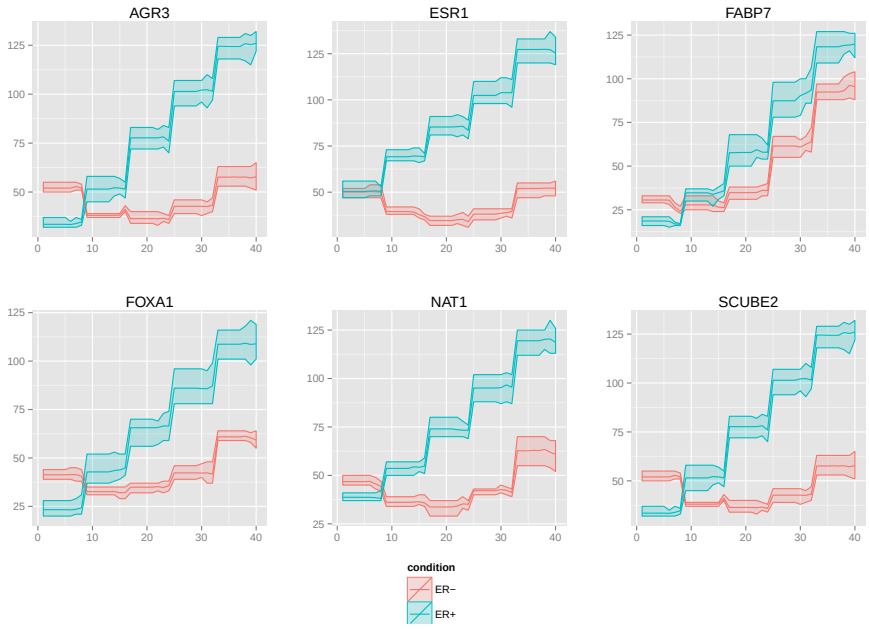


Figure 4.2: Network scores of top genes in the two conditions. Each sub figure corresponds to one gene. In each sub figure, the horizontal axis presents the order of couples (λ_1, λ_2) (e.g the first couple is $(lambda1.list[1], lambda2.list[1])$, the second couple is $(lambda1.list[1], lambda2.list[2])$, etc). For each point on the horizontal axis, we plot the network score of this gene under the ER- condition in red and under the ER+ condition in blue.

3.1.2 Description of the dataset

The dataset comprises 70 Arabidopsis thaliana transcriptomes. For each tissue, we have the log ratio expression level of 24,576 genes. Among the 70 tissues, 35 are leaf and 35 are root. The log ratio expression level were measured as follows. Each log-ratio is obtained as a comparison of Arabidopsis thaliana developped under the nitrogen starvation condition. The dataset were extracted from CATdb (see Gagnot *et al* [35]) and the co-expression was studied in Zaag *et al* [43]. The gene expressions were measured with a 2 color microarray and technical biases were removed with a LOWESS correction [44]. Nevertheless, in this study, they did not consider the dataset according to the tissues. In our study, the goal is to identify genes which are different between roots and leaves of Arabidopsis thaliana at the mean expression, network interaction or both levels.

Similar to the breast cancer dataset, we only consider a subset of 500 genes with the highest variance. Besides, we try a re-sampling method: cross-validation to assess the reproducibility of our approach. From the dataset of 500 genes, we create 20 sub datasets by removing five random observations in each condition. Hence, we consider a total of 21 datasets.

3.2 Set up and results

3.2.1 Building the grid

Similar to the grid of breast cancer data, we choose a grid with size 8×8 grid. The values of λ_1 and λ_2 are chosen by the R functions:

$$lambda1.list \leftarrow 10\text{^}seq(1, log10(1e-2), len = 8)$$
$$lambda2.list \leftarrow 10\text{^}seq(-3, log10(1e-6), len = 8)$$

In more details, we have

$$lambda1.list = \{10,\ 3.72,\ 1.38,\ 0.51,\ 0.19,\ 0.07,\ 0.02,\ 0.01\},$$
$$lambda2.list = \{10^{-3},\ 3.10^{-4},\ 10^{-4},\ 5.10^{-5},\ 10^{-5},\ 7.10^{-6},\ 2.10^{-6},\ 10^{-6}\}.$$

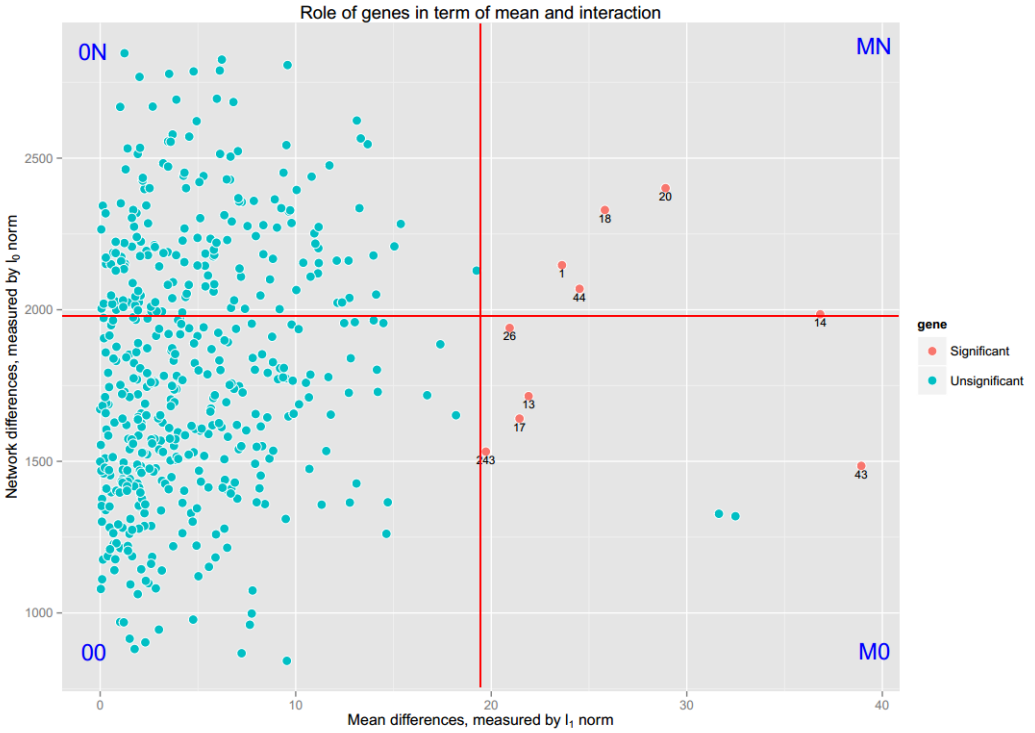
Most status are also captured with the grid. In details, all genes are fused when λ_2 equals 10^{-3}; while half of genes are fused when λ_2 equals 10^{-6}. In the network aspect, we capture most status of θ^1, θ^2 from all their elements equal zero (no edges) until 19250 edges presence over $p \times (p-1)/2$ = 124750 possible edges. It is about 15% of the edge presence.

3.2.2 Results

Using our approach for the 21 datasets, our obtained results are coherent. We give the result for the whole datasets with two types of network change measures in Figures 4.3 and 4.4. Results of some other datasets are in Appendix B.

Similar to the case of breast cancer dataset, we spilt the genes in four categories: 00, M0, 0N and MN. In this context, because we do not have prior information. In Figure 4.3, we decide to select genes whose measurements are higher than one half of the maximum measurements of all genes. In Figure 4.4, we select genes whose measurements in terms of mean are higher than one half of the maximum measurements of all genes and genes whose measurements in terms of network are higher than three quarters of the maximum measurements of all genes. We consider red genes in the figures as important genes. For two network change measures, the results obtained are different. However, they still share some same detections such as genes 14, 20 and 44. Therefore, we also count the number of a gene detected by our method in the 21 datasets by either l_0 or l_1 measure. The results are shown in Figures 4.5.

Notably, three genes 14 (AT1666390), 20 (AT1634060) and 44 (AT5607990) are detected by our method in every dataset or by any type of measure. Unfortunately, we do not have any prior information about these genes yet. However, we expect that these genes are important and should be analysed furthermore.



(a)

Figure 4.3: The roles of genes evaluated by l_0 network change measure and l_1 mean change measure.

(a)

Figure 4.4: The roles of genes evaluated by l_1 network change measure and l_1 mean change measure.

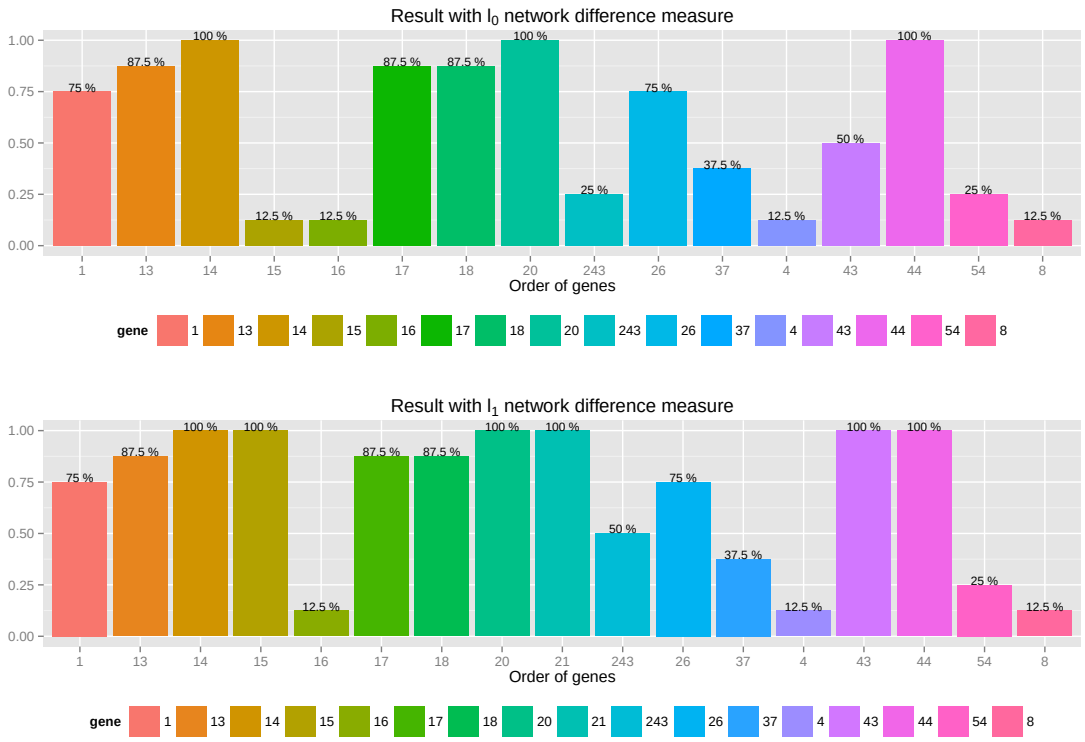


Figure 4.5: Percentage of detecting as a significant gene of top genes.

4 Conclusion of Chapter 4

In this chapter, we apply our method on two real datasets. For breast cancer dataset, we obtained similar results as previous methods. This is what we expected. For the Arabidopsis thaliana method, we obtain a list of several genes which are highly different compared to the others both in terms of mean expression level and in terms of network. We expect that these genes take important roles in the functioning of leaf and root at Arabidopsis thaliana.

Chapter 5

Conclusion

During the 3 years of my thesis, I mainly focused on sparse regularization methods for high-dimensional data. The goal is to find a method which improves both differential analysis and network inference in transcriptomic data. The more I tried to improve the results, the more I realize that how hard to get close to the true result. Why is that?

In my opinion, the core of sparse regularization methods is the method Lasso. Since the publication of Lasso in 1996, many Lasso's variations have been created. I category the variations into two classes. The first class of methods were created in order to fit Lasso with each individual studying context. Using only Lasso penalty part is not enough in these contexts and people add some other penalty parts to address specific requirements. For instance, they are Fused Lasso, elasticnet, group-Lasso and cooperative-Lasso. A motivating example of using the Fused Lasso is in gene expression studying. In the research, people expect that the mean expression level of some given genes are close. Hence, they add the fused part beside the Lasso part. The second class of methods was created to improve Lasso in a very original context. They are methods such as adaptive Lasso, adaptive Fused Lasso. These methods improve Lasso in the statistical sense. For instance, theoretically, adaptive Lasso will give consistent estimators, while Lasso may not.

However, no matter how all of these variations of Lasso change, the core idea is using an $l_1 - norm$ regularization part which is very sensitive to the change of the data, especially in the high-dimensional setting. Therefore, we should forget about quantitative estimations of parameters. Instead of that, qualitative estimators is something which is much more realistic to do. This thesis proposes a novel approach which aims to that goal. As discussed in Chapters 3 and 4, if we have a dataset about genes. We do not try to answer questions such as what is the mean expression level of one genes or what is the exact relations between genes? Instead of that, we provide a procedure which helps to visualize the importance of genes both in terms of mean expression and interaction. This is the second main contribution of this thesis.

The first main contribution of this thesis is a new model which can solve differential analysis and network inference for multi-task datasets. This model is raised from studying gene expression data in multiple conditions and aims to address two very standard questions in biology. Because it is generalized from the original graphical Gaussian model, this model could be well adapted to many other types of data. We also implemented the model in an R package which is available soon.

In term of theory, by using the new model, we showed the importance and correlation between the mean vector and the covariance matrix in terms of estimating. This is the third contribution of this thesis.

In conclusion, getting a picture about variables' structure in high-dimensional data is a challenging work. I believe that we never have enough information, but we have enough sources of data. Our work is to create a link between the sources. Although the data from different sources will be heterogeneous and hard to find the relation, if we may find them, we will obtain a better view on the problem.

Appendix A

I give the results for simulated data in Chapter 3.

2.1.6 Univariate analysis results

2.1.6.1 Results with known covariance matrix

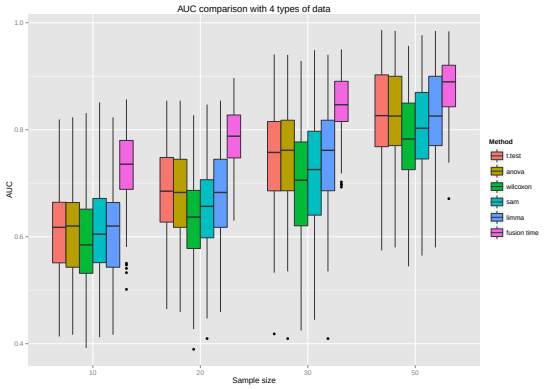


Figure 5.1: Scenario 1: covariance matrix $(\Sigma^*)^k$ is known. Here, we compare the AUC of 6 methods t-test, anova, wilcoxon, sam, limma and our method (fusion time). We are in the CLUSTER network setting. We simulated data with a number of observations is $n_k \in \{10, 20, 30, 50\}$. Each box represents the AUC of one method for one value of n_k over 100 simulations.

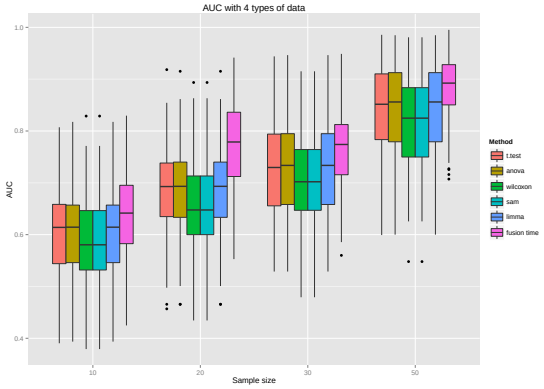


Figure 5.2: Scenario 1: covariance matrix $(\Sigma^*)^k$ is known. Here, we compare the AUC of 6 methods t-test, anova, wilcoxon, sam, limma and our method (fusion time). We are in the RANDOM network setting. We simulated data with a number of observations is $n_k \in \{10, 20, 30, 50\}$. Each box represents the AUC of one method for one value of n_k over 100 simulations.

2.1.6.2 Results with known network

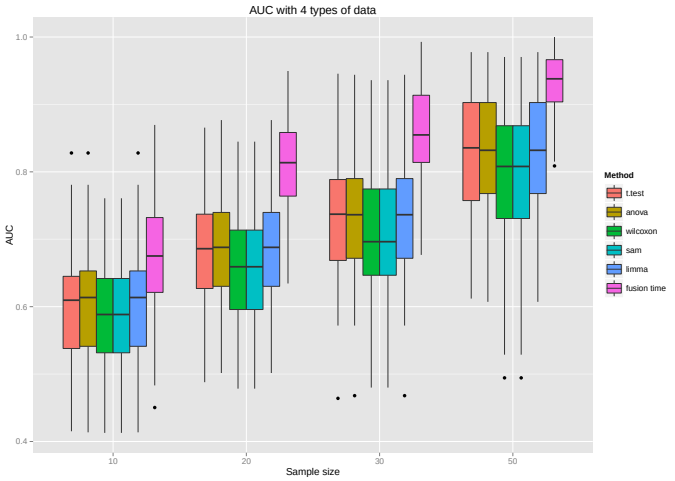


Figure 5.3: Scenario 2: network is known. Here, we compare the AUC of 6 methods t-test, anova, wilcoxon, sam, limma and our method (fusion time). We are in the BAND network setting. We simulated data with a number of observations is $n_k \in \{10, 20, 30, 50\}$. Each box represents the AUC of one method for one value of n_k over 100 simulations.

2.1.6.3 Results with no prior information about the covariance matrix

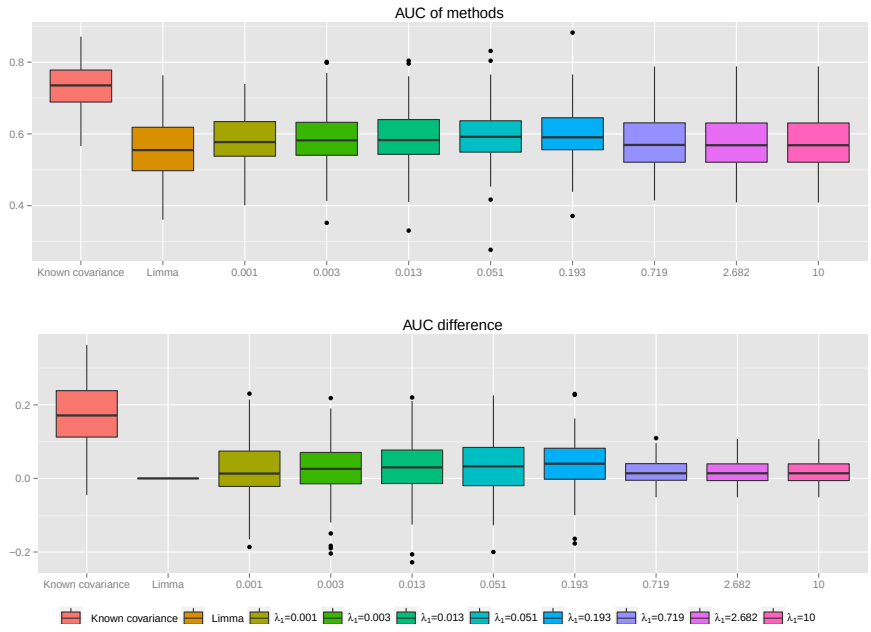


Figure 5.4: Scenario 3 with $n_k = 10$: no prior information about the covariance matrix. AUC comaprison between $limma$ method, our method with given covariance matrix (Scenario 1), our method in Scenario 3 for 8 values of λ_1 from zero to infinity. We are in the BAND network setting. The number of observations n_k is 10 and the number of genes p is 50. The picture first picture is the AUC boxplot of all methods. The second picture is the difference between the AUC of all methods and the AUC of limma. Therefore, the second boxplot is zero.

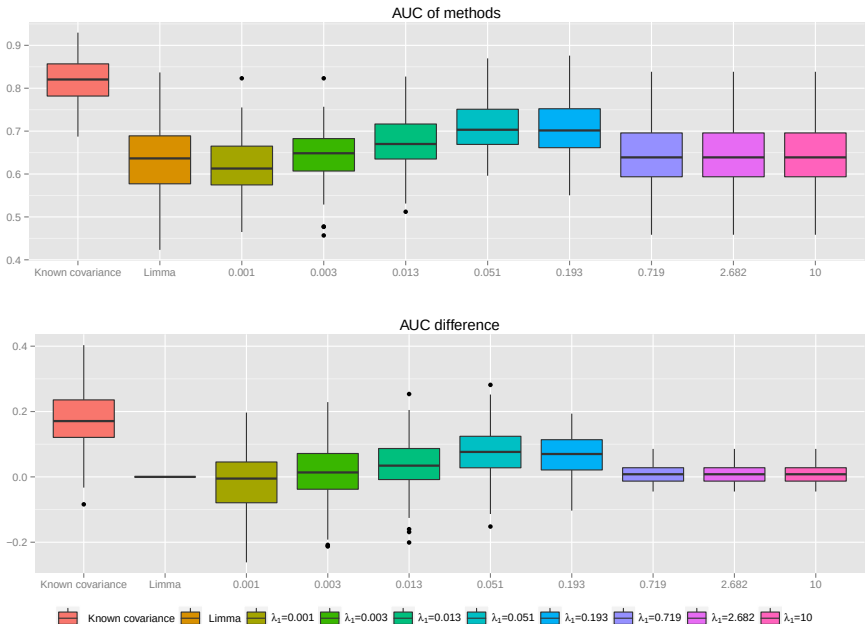


Figure 5.5: Scenario 3 with $n_k = 10$: no prior information about the covariance matrix. AUC comaprison between $limma$ method, our method with given covariance matrix (Scenario 1), our method in Scenario 3 for 8 values of λ_1 from zero to infinity. We are in the BAND network setting. The number of observations n_k is 20 and the number of genes p is 50. The picture first picture is the AUC boxplot of all methods. The second picture is the difference between the AUC of all methods and the AUC of limma. Therefore, the second boxplot is zero.

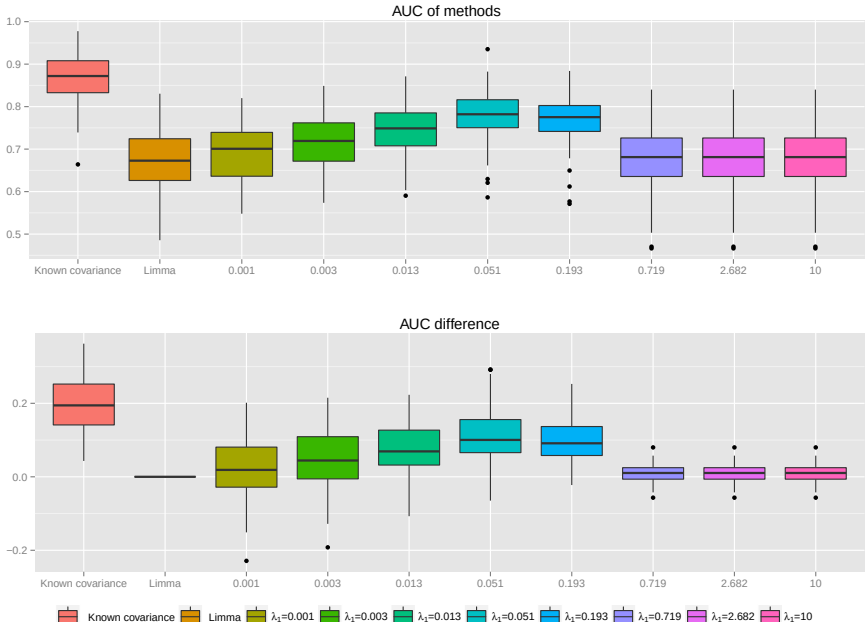


Figure 5.6: Scenario 3 with $n_k = 10$: no prior information about the covariance matrix. AUC comaprison between $limma$ method, our method with given covariance matrix (Scenario 1), our method in Scenario 3 for 8 values of λ_1 from zero to infinity. We are in the BAND network setting. The number of observations n_k is 30 and the number of genes p is 50. The picture first picture is the AUC boxplot of all methods. The second picture is the difference between the AUC of all methods and the AUC of limma. Therefore, the second boxplot is zero.

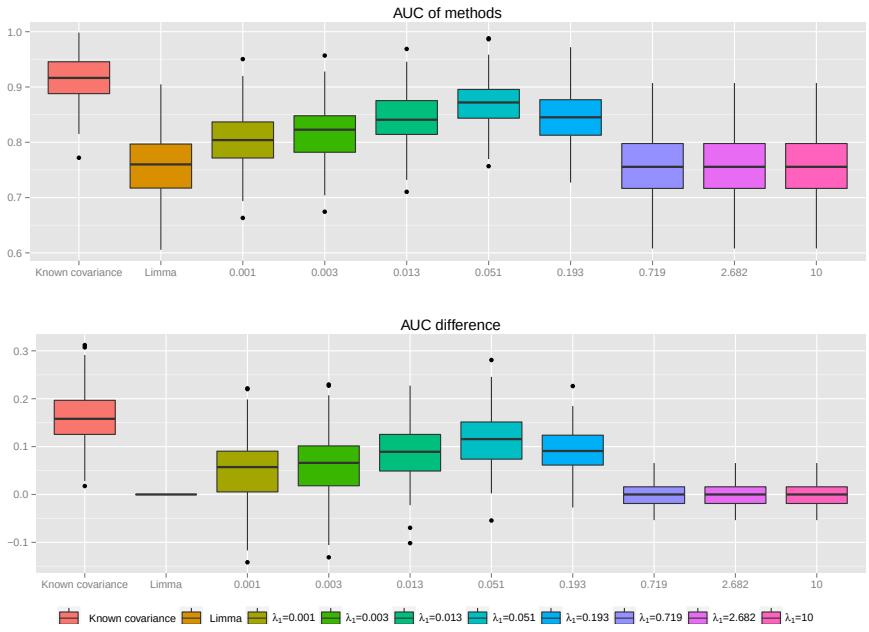


Figure 5.7: Scenario 3 with $n_k = 10$: no prior information about the covariance matrix. AUC comaprison between $limma$ method, our method with given covariance matrix (Scenario 1), our method in Scenario 3 for 8 values of λ_1 from zero to infinity. We are in the BAND network setting. The number of observations n_k is 40 and the number of genes p is 50. The picture first picture is the AUC boxplot of all methods. The second picture is the difference between the AUC of all methods and the AUC of limma. Therefore, the second boxplot is zero.

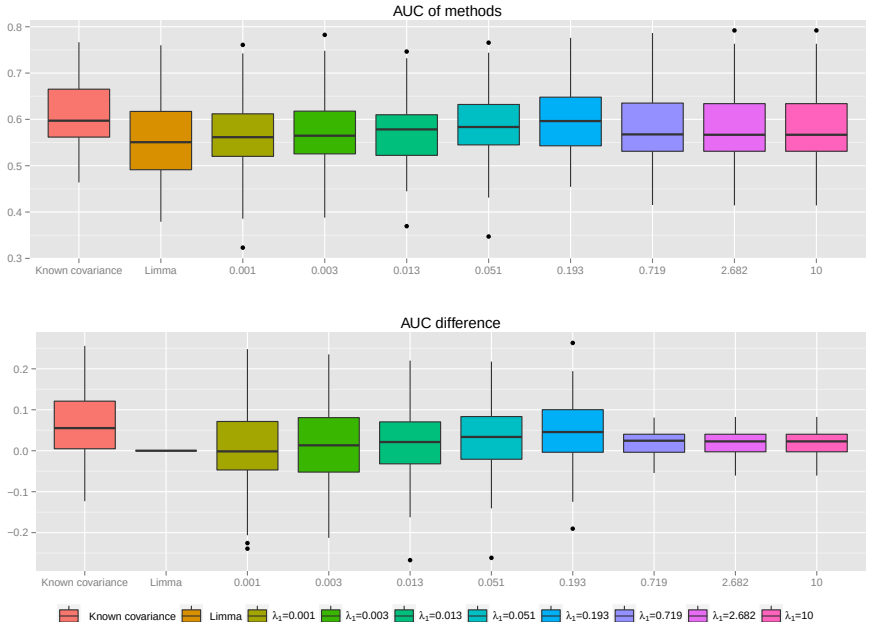


Figure 5.8: Scenario 3 with $n_k = 10$: no prior information about the covariance matrix. AUC comaprison between $limma$ method, our method with given covariance matrix (Scenario 1), our method in Scenario 3 for 8 values of λ_1 from zero to infinity. We are in the HUB network setting. The number of observations n_k is 10 and the number of genes p is 50. The picture first picture is the AUC boxplot of all methods. The second picture is the difference between the AUC of all methods and the AUC of limma. Therefore, the second boxplot is zero.

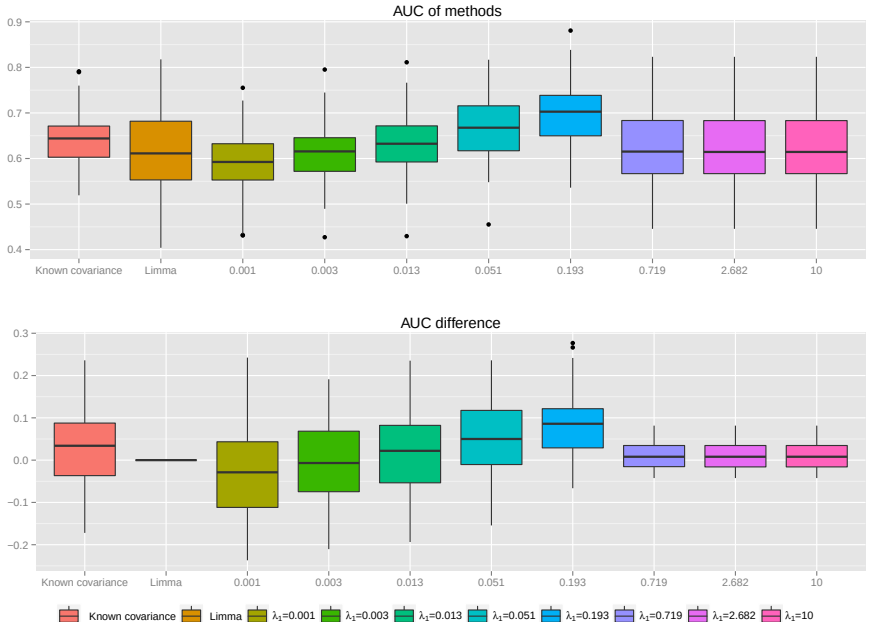


Figure 5.9: Scenario 3 with $n_k = 10$: no prior information about the covariance matrix. AUC comaprison between $limma$ method, our method with given covariance matrix (Scenario 1), our method in Scenario 3 for 8 values of λ_1 from zero to infinity. We are in the HUB network setting. The number of observations n_k is 20 and the number of genes p is 50. The picture first picture is the AUC boxplot of all methods. The second picture is the difference between the AUC of all methods and the AUC of limma. Therefore, the second boxplot is zero.

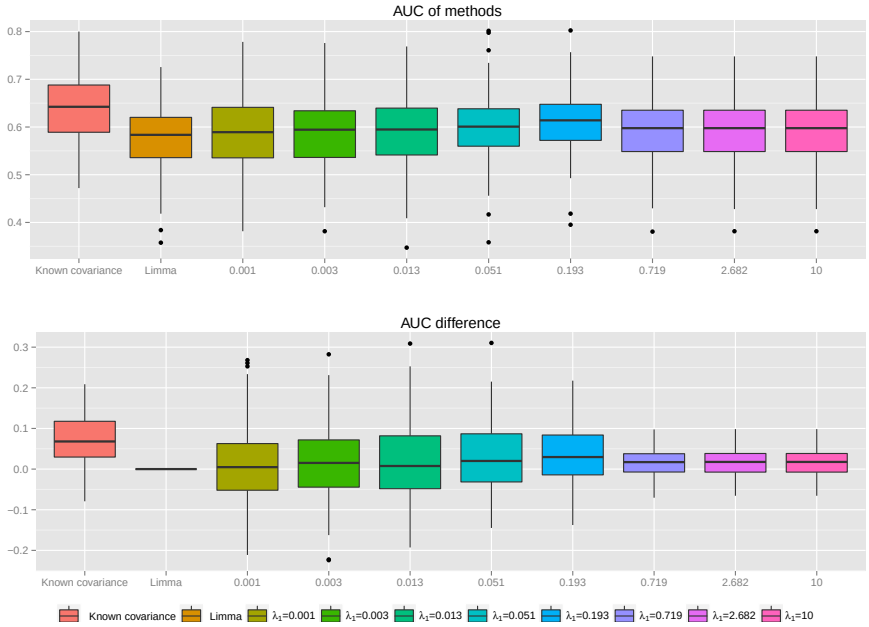


Figure 5.10: Scenario 3 with $n_k = 10$: no prior information about the covariance matrix. AUC comaprison between $limma$ method, our method with given covariance matrix (Scenario 1), our method in Scenario 3 for 8 values of λ_1 from zero to infinity. We are in the RANDOM network setting. The number of observations n_k is 10 and the number of genes p is 50. The picture first picture is the AUC boxplot of all methods. The second picture is the difference between the AUC of all methods and the AUC of limma. Therefore, the second boxplot is zero.

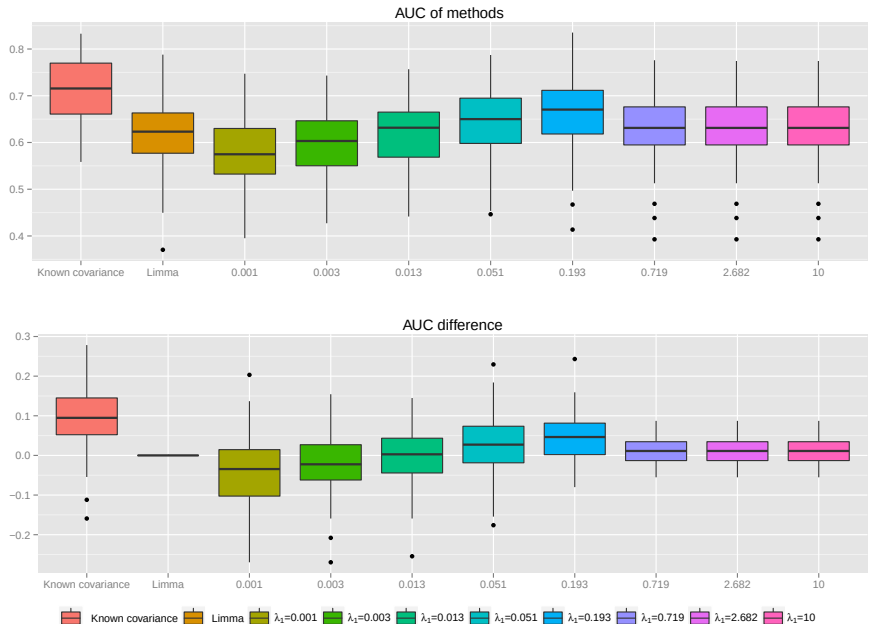


Figure 5.11: Scenario 3 with $n_k = 10$: no prior information about the covariance matrix. AUC comaprison between $limma$ method, our method with given covariance matrix (Scenario 1), our method in Scenario 3 for 8 values of λ_1 from zero to infinity. We are in the RANDOM network setting. The number of observations n_k is 20 and the number of genes p is 50. The picture first picture is the AUC boxplot of all methods. The second picture is the difference between the AUC of all methods and the AUC of limma. Therefore, the second boxplot is zero.

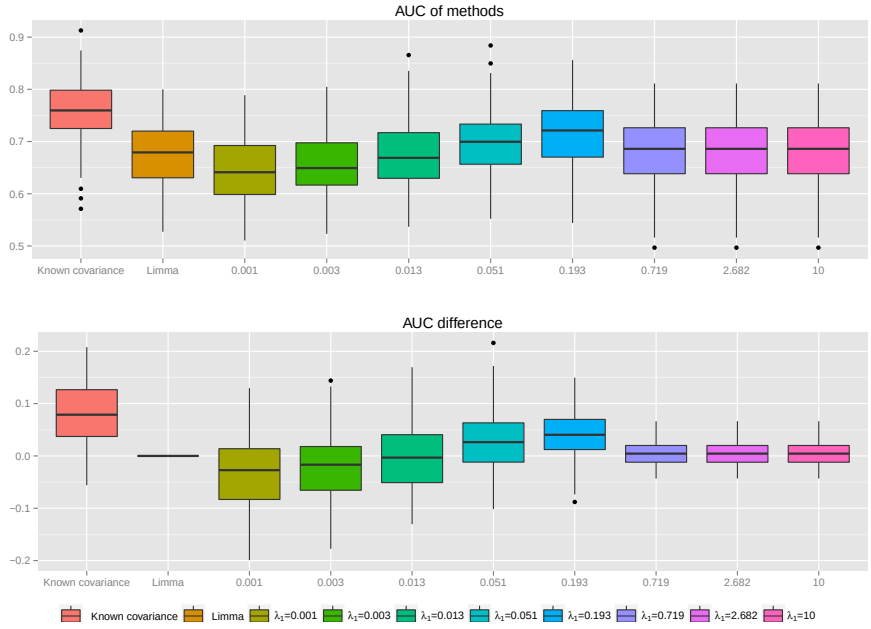


Figure 5.12: Scenario 3 with $n_k = 10$: no prior information about the covariance matrix. AUC comaprison between $limma$ method, our method with given covariance matrix (Scenario 1), our method in Scenario 3 for 8 values of λ_1 from zero to infinity. We are in the RANDOM network setting. The number of observations n_k is 30 and the number of genes p is 50. The picture first picture is the AUC boxplot of all methods. The second picture is the difference between the AUC of all methods and the AUC of limma. Therefore, the second boxplot is zero.

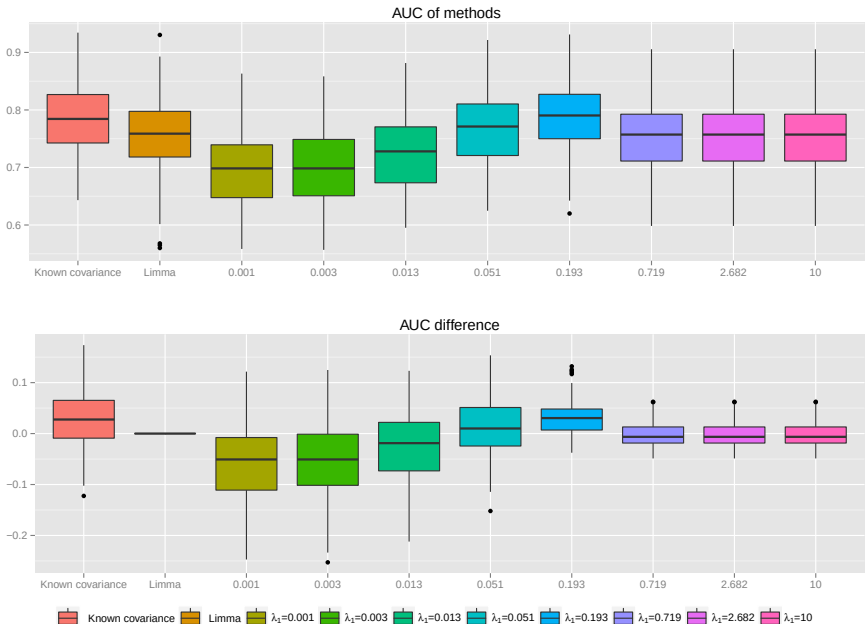


Figure 5.13: Scenario 3 with $n_k = 10$: no prior information about the covariance matrix. AUC comaprison between $limma$ method, our method with given covariance matrix (Scenario 1), our method in Scenario 3 for 8 values of λ_1 from zero to infinity. We are in the RANDOM network setting. The number of observations n_k is 50 and the number of genes p is 50. The picture first picture is the AUC boxplot of all methods. The second picture is the difference between the AUC of all methods and the AUC of limma. Therefore, the second boxplot is zero.

3.4 Inference network when having 10 conditions

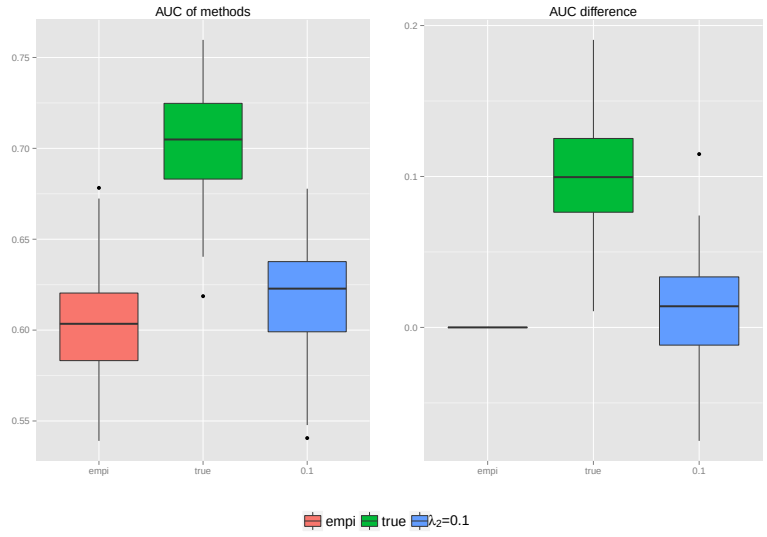


Figure 5.14: AUC of different methods in BAND network case. Graphical lasso approach with data normalized by the empirical mean (first boxplot), Graphical lasso approach with data normalized by the true mean (second boxplot), and our method with the best choice of $\lambda_2 = 0.1$ (third boxplot). The total number of observations is $n = 40$. The number of genes is $p = 100$.

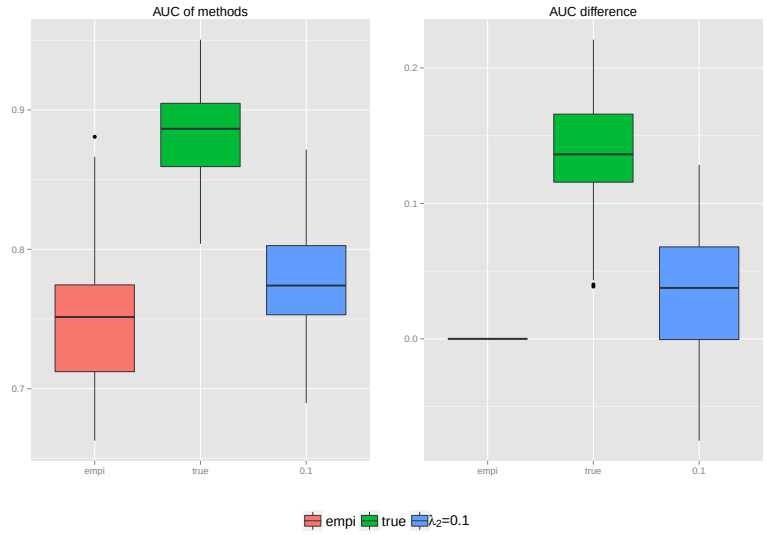


Figure 5.15: AUC of different methods in HUB network case. Graphical lasso approach with data normalized by the empirical mean (first boxplot), Graphical lasso approach with data normalized by the true mean (second boxplot), and our method with the best choice of $\lambda_2 = 0.1$ (third boxplot). The total number of observations is $n = 40$. The number of genes is $p = 100$.

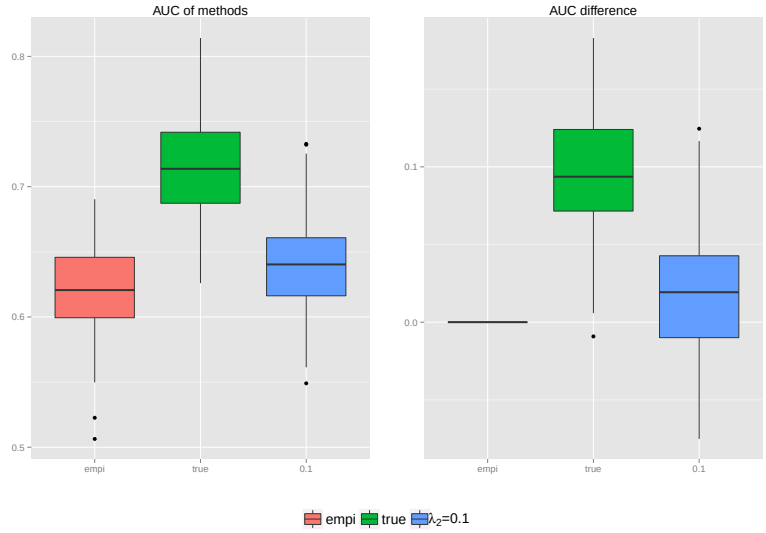


Figure 5.16: AUC of different methods in RANDOM network case. Graphical lasso approach with data normalized by the empirical mean (first boxplot), Graphical lasso approach with data normalized by the true mean (second boxplot), and our method with the best choice of $\lambda_2 = 0.1$ (third boxplot). The total number of observations is $n = 40$. The number of genes is $p = 100$.

Appendix B

I give the results for two real datasets in Chapter 4.

2 Breast cancer data

Figure 5.17: The roles of genes evaluated by different types of measure.

Figure 5.18: The roles of genes evaluated by different types of measure.

Figure 5.19: The roles of genes evaluated by different types of measure.

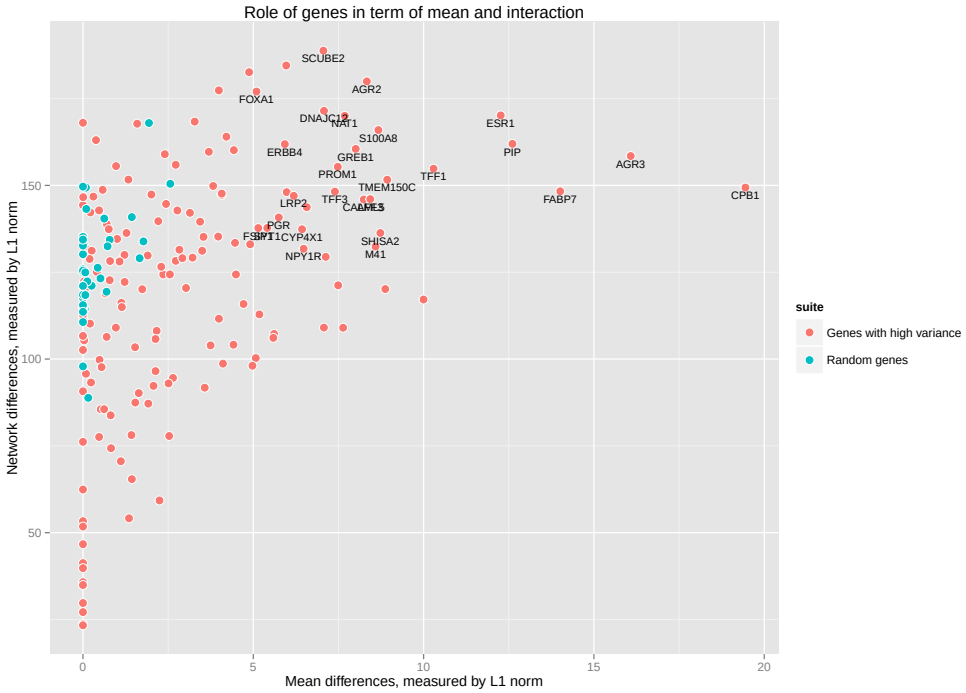


Figure 5.20: The roles of genes evaluated by different types of measure.

Figure 5.21: The roles of genes evaluated by different types of measure.

3 Arabidopsis thaliana data

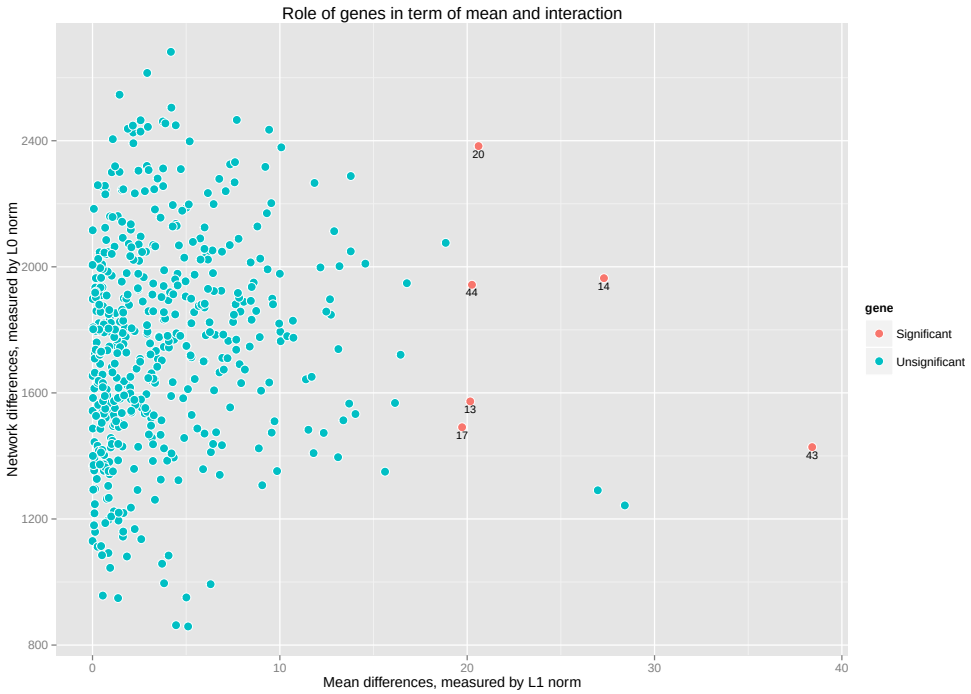


Figure 5.22: The roles of genes evaluated by different types of measure.

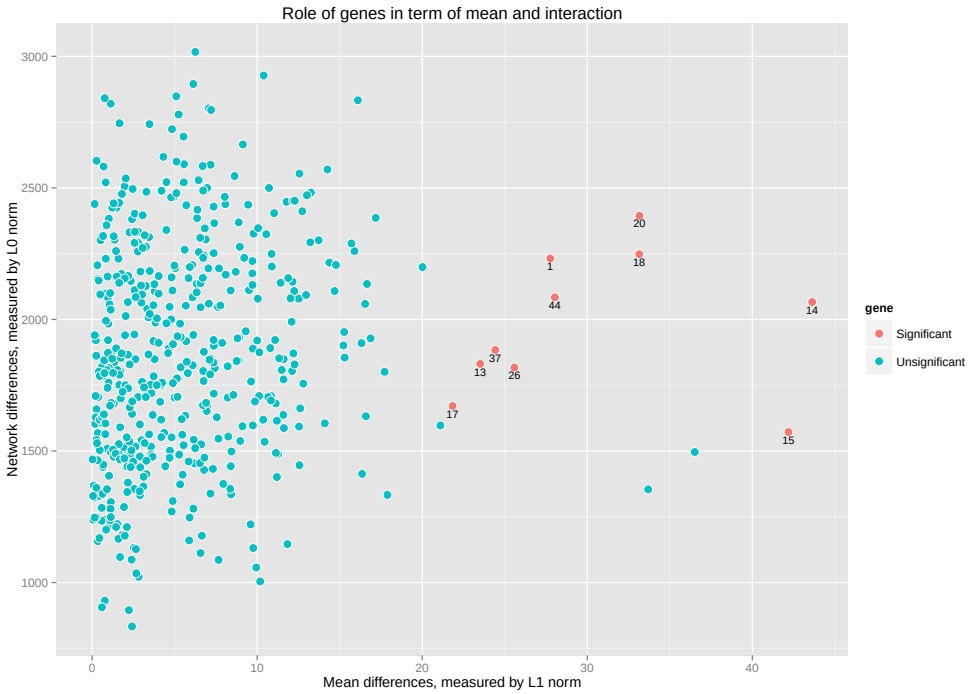


Figure 5.23: The roles of genes evaluated by different types of measure.

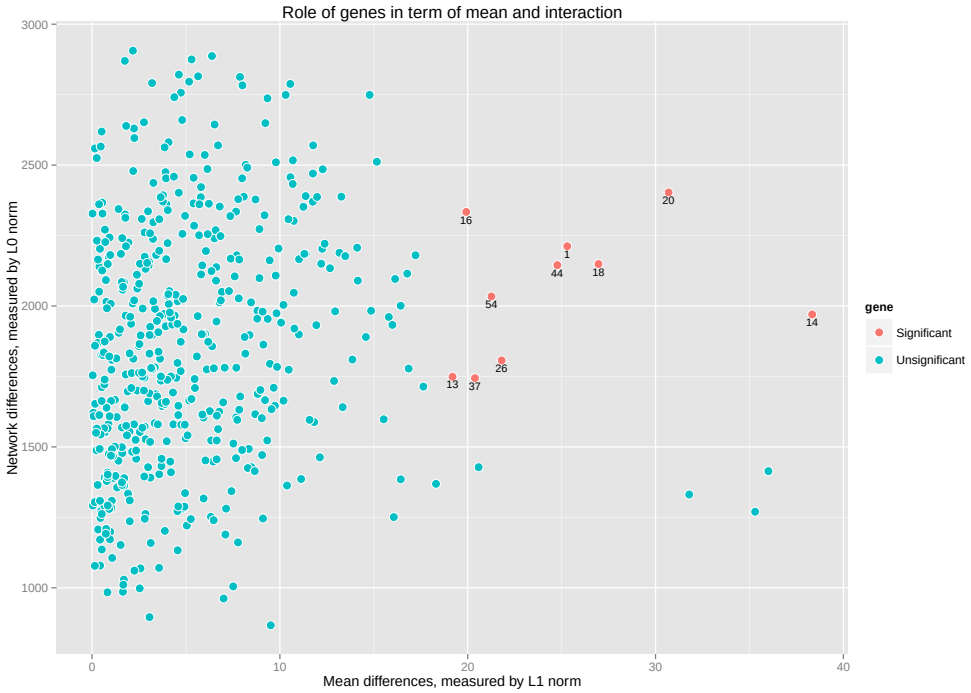


Figure 5.24: The roles of genes evaluated by different types of measure.

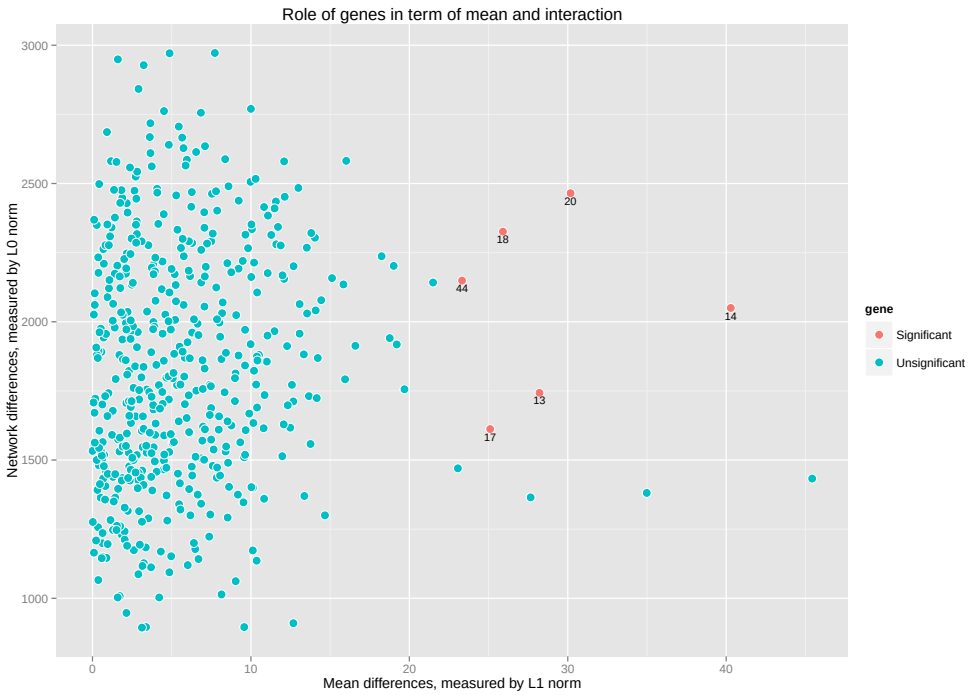


Figure 5.25: The roles of genes evaluated by different types of measure.

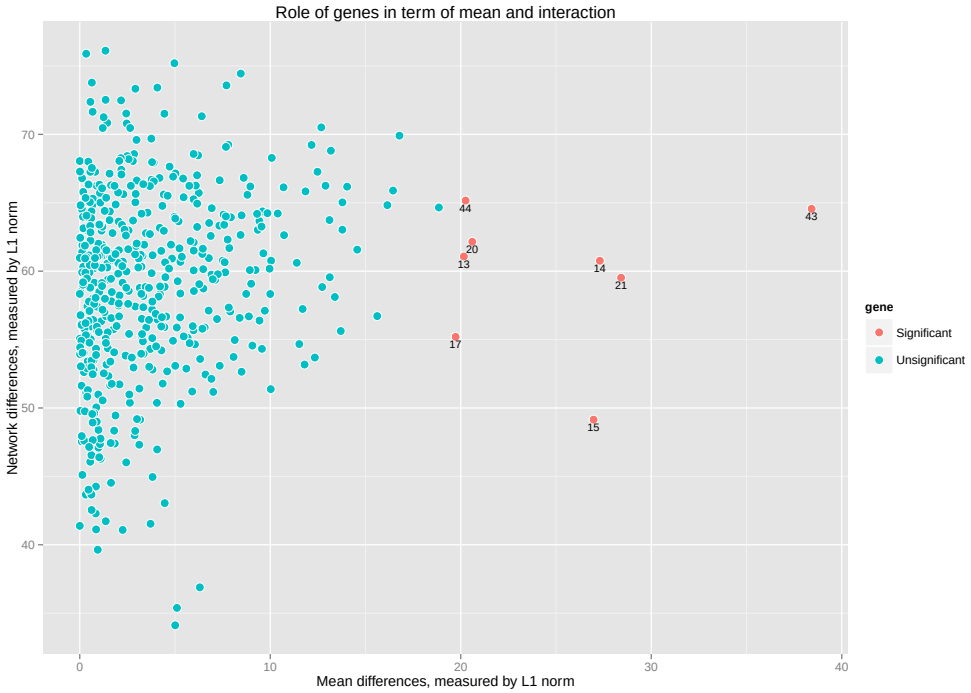


Figure 5.26: The roles of genes evaluated by different types of measure.

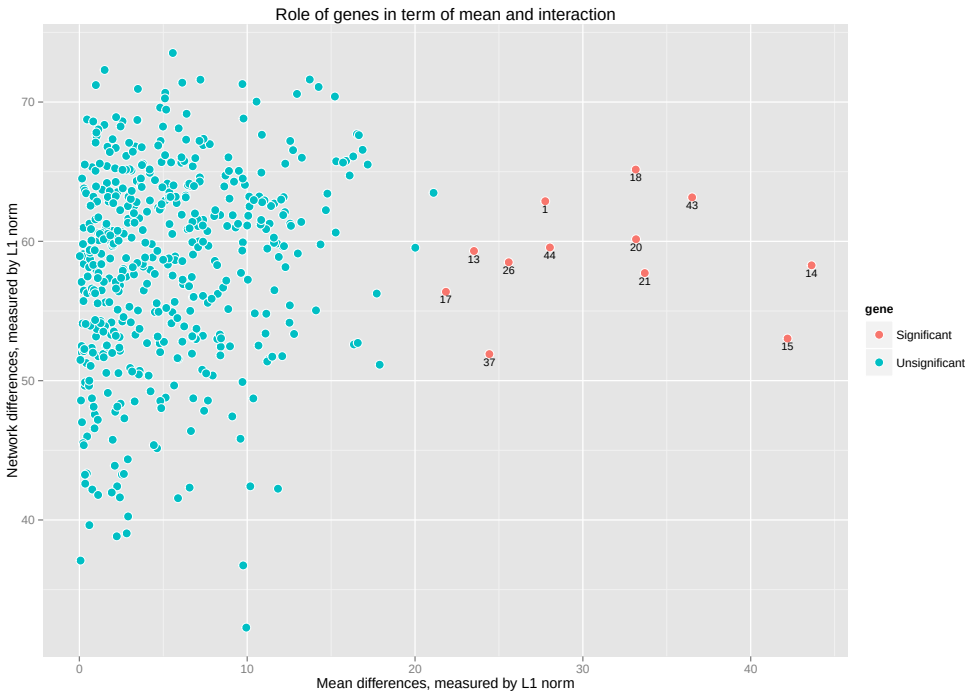


Figure 5.27: The roles of genes evaluated by different types of measure.

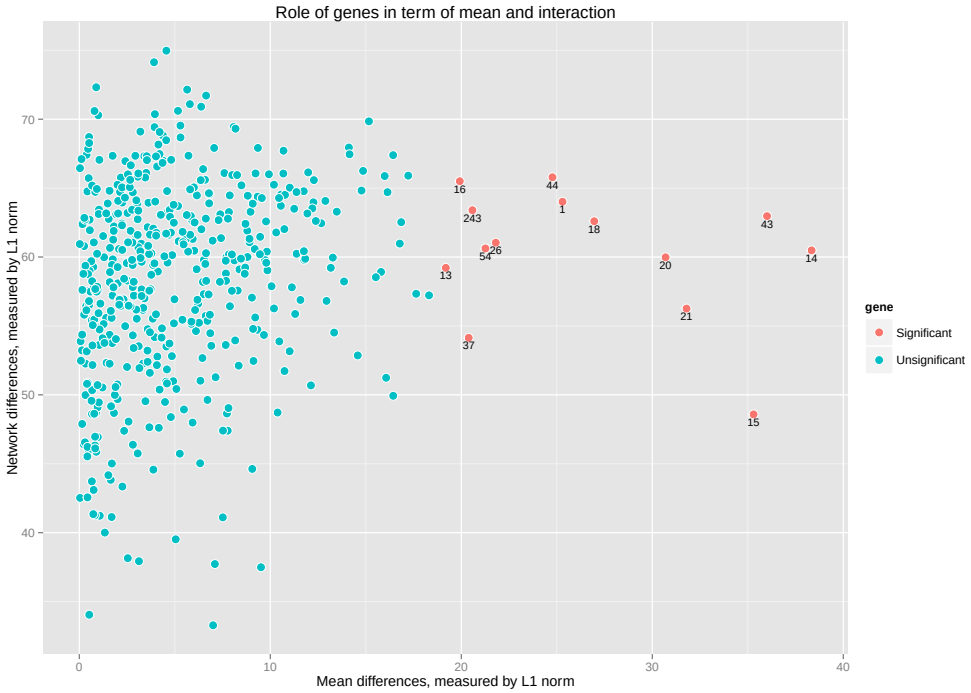


Figure 5.28: The roles of genes evaluated by different types of measure.

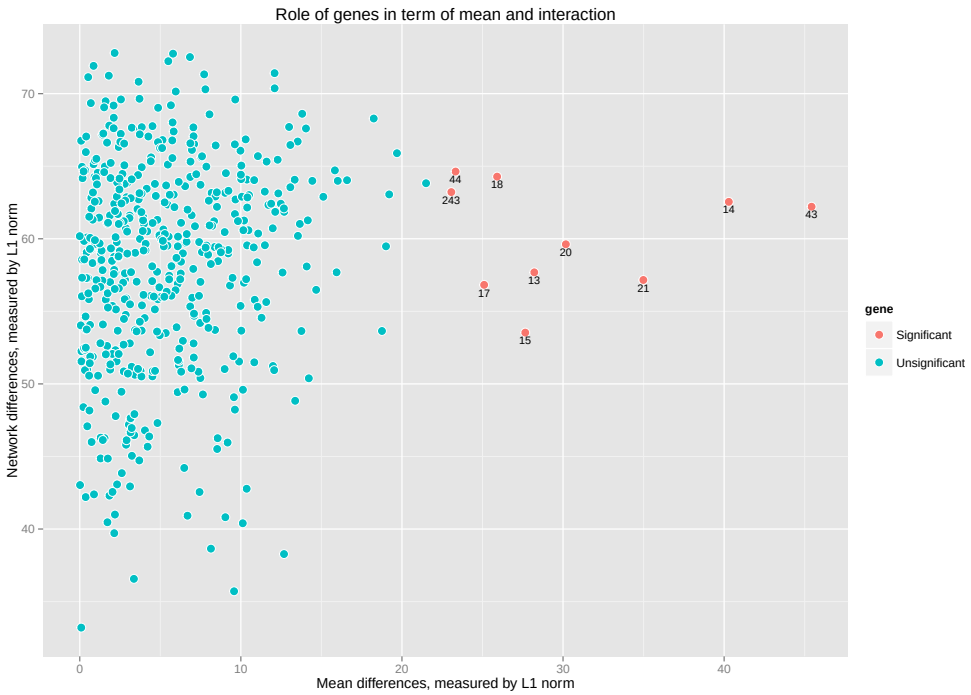


Figure 5.29: The roles of genes evaluated by different types of measure.

Bibliography

[1] Gordon K. Smyth. *Linear Models and Empirical Bayes Methods for Assessing Differential Expression in Microarray Experiments*. Statistical applications in genetics and molecular biology, 2004.

[2] Virginia Goss Tusher, Robert Tibshirani, and Gilbert Chu. *Significance analysis of microarrays applied to the ionizing radiation response* PNAS 2001 98: 5116-5121, (Apr 24).

[3] Wilcoxon, Frank. *Individual comparisons by ranking methods*, Biometrics Bulletin. 1 (6): 80–83. 1945.

[4] Etienne Roquain. *Type I error rate control in multiple testing: a survey with proofs.* 2010. Journal de la Société Française de Statistique. Vol. 152 No. 2 3-38, 2011.

[5] M. Jeanmougin, A. Reynies, L. Marisa, C. Paccard, G. Nuel, M. Guedj *Should We Abandon the t-Test in the Analysis of Gene Expression Microarray Data: A Comparison of Variance Modeling Strategies* . PIOS ONE, 2010. URL: http://dx.doi.org/10.1371/journal.pone.0012336.

[6] L. Jacob, P. Neuvial, S. Dudoit. *More power via graph-structured tests for differential expression of gene networks.* The Annals of Applied Statistics, 2012, Vol. 6.

[7] Ming Yuan and Yi Lin. *Model selection and estimation in regression with grouped variables*, J. R. Statist. Soc. B (2006)68, Part 1, pp. 49–67.

[8] Banerjee. O, El Ghaoui. L, d'Aspremont. A. *Model selection through sparse maximum likelihood estimation for multivariate Gaussian or binary data.* Journal Mach Learn Res 9:485–516, 2008.

[9] Nicolai Meinshausen,Peter Buhlmann. *High-dimensional graphs and variable selection with the Lasso.* Ann. Statist.Volume 34, Number 3 (2006), 1436-1462.

[10] Villers. F, Schaeffer. B, Bertin. C, Huet. S. *Assessing the validity domains of graphical Gaussian models in order to infer relationships among components of complex biological systems.* Stat Appl Genet Mol Biol 7(2), 2008.

[11] Rocha. GV, Zhao. P, Yu. B. *A path following algorithm for sparse pseudo-likelihood inverse co-variance estimation (splice).* URL: http://arxiv.org/abs/0807.3734. 2008.

[12] Jie Peng, Pei Wang, Nengfeng Zhou, Ji Zhu. *Partial correlation estimation by joint sparse regression models.* J Am Stat Assoc. 2009 Jun 1; 104(486): 735–746.

[13] Julien Chiquet, Yves Grandvalet, Christophe Ambroise. *Inferring Multiple Graphical Models.* Statistics and Computing, October 2011, Volume 21, Issue 4, pp 537–553.

[14] Danaher.P, Wang.P, Witten. D.M. *The joint graphical lasso for inverse covariance estimation across multiple classes.* Journal of the Royal Statistical Society. Series B: Statistical Methodology, 76(2), 373-397.

[15] Karthik Mohan, Mike Chung, Seungyeop Han, Daniela Witten, Su-in Lee, Maryam Fazel. *Structured Learning of Gaussian Graphical Models.* Advances in Neural Information Processing Systems 25 (NIPS 2012)

[16] Robert Tibshirani. *Regression shrinkage and selection via the lasso.* (1996). J. Roy. Statist. Soc. Ser.B58 267–288.MR1379242.

[17] Julien Chiquet, Pierre Gutierrez, Guillem Rigaill *Fast tree inference with weighted fusion penalties.* Journal of Computational and Graphical Statistics, 2016.

[18] Hui Zou. *The adaptive lasso and its oracle properties.* Journal of the American Statistical Association. Volume 101, Number 476, 2006.

[19] De Leeuw, J *Block relaxation algorithms in statistics.* Information Systems and Data Analysis Part of the series Studies in Classification, Data Analysis, and Knowledge Organization pp 308-324, 1994.

[20] Tuo Zhao, Han Liu, Kathryn Roeder, John Lafferty and Larry Wasserman (2014). huge: High-dimensional Undirected Graph Estimation. R package version 1.2.6. http://CRAN.R-project.org/package=huge

[21] Jerome Friedman, Trevor Hastie, Robert Tibshirani (2010). *Regularization Paths for Generalized Linear Models via Coordinate Descent.* Journal of Statistical Software, 33(1), 1-22. URL http://www.jstatsoft.org/v33/i01/.

[22] Trevor Hastie, Brad Efron (2013). *LARS: Least Angle Regression, Lasso and Forward Stagewise.* The Annals of Statistics 2004, Vol. 32, No. 2, 407–499.

[23] Ryan J. Tibshirani, Jonathan Taylor *The Solution Path of the Generalized Lasso* The Annals of Statistics, Volume 39, Number 3.

[24] Taylor B. Arnold and Ryan J. Tibshirani (2014). genlasso: Path algorithm for generalized lasso problems. R package version 1.3. http://CRAN.R-project.org/package=genlasso

[25] Holger Hoefling (2014). FusedLasso: Solves the generalized Fused Lasso. R package version 1.0.6. http://CRAN.R-project.org/package=FusedLasso

[26] Gorski Jochen, Pfeuffer Frank, Klamroth Kathrin *Biconvex Sets and Optimization with Biconvex Functions - A Survey and Extensions.* Mathematical Methods of Operations Research. December 2007, Volume 66, Issue 3, pp 373–407.

[27] C.A. Floudas and V. Visweswaran. *A Global Optimization Algorithm (GOP) For Certain Classes of Nonconvex NLPs : I. Theory.* N.J. 08544-5263, 1990.

[28] C. J. Geyer. *On the asymptotics of constrained M-estimation.* The annals of Statistic, Volume 22, Number 4 (1994), 1993-2010.

[29] K. Knight,W. Fu. *Asymtptics for the Lasso-type estimators.* The annals of Statistic, Volume 28, Number 5 (2000), 1356-1378.

[30] Verzelen. N. *Minimax risks for sparse regressions: Ultra-high dimensional phenomenons.* Electron. J. Statist. Volume 6 (2012), 38-90.

[31] Georgios A Pavlopoulos, Maria Secrier, Charalampos N Moschopoulos, Theodoros G Soldatos, Sophia Kossida, Jan Aerts, Reinhard Schneider, Pantelis G Bagos. *Using graph theory to analyze biological networks.* BioData Min. 2011; 4: 10. Published online 2011 Apr 28. doi: 10.1186/1756-0381-4-10.

[32] Michael P. H. Stumpf, Carsten Wiuf and Robert M. May. *Subnets of scale-free networks are not scale-free: Sampling properties of networks.* PNAS, vol. 102 no. 12 (2005), 4221–4224.

[33] Albert-Laszlo Barabasi, Reka Albert. *Emergence of Scaling in Random Networks* Science, 15 Oct 1999, Vol. 286, Issue 5439, pp. 509-512.

(2003) Soc. Industrial Appl. Math. Rev. 45 , 167–256.

[34] M. Guedj, L. Marisa, A. de Reynies, B. Orsetti, R. Schiappa, F. Bibeau, G. MacGrogan, F. Lerebours, P. Finetti, M. Longy, P. Bertheau, F. Bertrand, F. Bonnet, A. L. Martin, J. P. Feugeas, I. Bieche, J. Lehmann-Che, R. Lidereau, D. Birnbaum, F. Bertucci, H. de The, et C. Theillet. *A refined molecular taxonomy of breast cancer.* Oncogene, 31(9):1196–1206, Mar 2011.

[35] Gagnot S., Tamby J.P., Martin-Magniette M.L., Bitton F., Taconnat L., Balzergue S., Aubourg S., Renou J.P., Lecharny A., Brunaud V. *CATdb: a public access to Arabidopsis transcriptome data from the URGV-CATMA platform.* Nucleic Acids Res. 2008;36:D986-D990.

[36] Obacz J, Brychtova V, Podhorec J, Fabian P, Dobes P, Vojtesek B, Hrstka R. *Anterior gradient protein 3 is associated with less aggressive tumors and better outcome of breast cancer patients.* OncoTargets and Therapy. Volume 8. 24 June 2015 Volume 2015:8 Pages 1523—1532.

[37] Alshareeda AT, Rakha EA, Nolan CC, Ellis IO, Green AR. *Fatty acid binding protein 7 expression and its sub-cellular localization in breast cancer.* Breast Cancer Res Treat. 2012 Jul;134(2):519-29. doi: 10.1007/s10549-012-2083-8. Epub 2012 May 6.

[38] Lin YC, Lee YC, Li LH, Cheng CJ, Yang RB. *Tumor suppressor SCUBE2 inhibits breast-cancer cell migration and invasion through the reversal of epithelial-mesenchymal transition.* J Cell Sci. 2014 Jan 1;127(Pt 1):85-100. doi: 10.1242/jcs.132779. Epub 2013 Nov 8.

[39] Sirizarry. RA, Hobbs, B. Collin, F. Beazer-Barclay, YD. Antonellis, KJ. Scherf, U. Speed. TP. *Exploration, normalization, and summaries of high density oligonucleotide array probe level data.* Biostatistics. 4 (2): 249–64, 2003.

[40] Tuo Zhao, Han Liu, Kathryn Roeder, John Lafferty, and Larry Wasserman *The huge Package for High-dimensional Undirected Graph Estimation in R.* J Mach Learn Res. 2012 Apr; 13: 1059–1062.

[41] Wakefield L, Robinson J, Long H, Ibbitt JC, Cooke S, Hurst HC, Sim E. *Arylamine N-acetyltransferase 1 expression in breast cancer cell lines: a potential marker in estrogen receptor-positive tumors.* Genes Chromosomes Cancer. 2008 Feb;47(2):118-26.

[42] Tricia M Wright, Suzanne E Wardell, Jeff S Jasper, James P Stice, Rachid Safi, Erik R Nelson, and Donald P McDonnell. *Delineation of a FOXA1/ER-alpha/AGR2 regulatory loop that is dysregulated in endocrine therapy-resistant breast cancer.* MDOI: 10.1158/1541-7786.MCR-14-0195, December 2014.

[43] Zaag R, Tamby JPh, Guichard C, Tariq Z, Rigaill G, Delannoy E, Renou JP, Balzergue S, Mary-Huard T, Aubourg S, Martin-Magniette ML, Brunaud V. *GEM2Net: from gene expression modeling to -omics networks, a new CATdb module to investigate Arabidopsis thaliana genes involved in stress response.* Nucl. Acids Res. (28 January 2015) 43 (D1): D1010-D1017. doi: 10.1093/nar/gku1155.

[44] John A Berger, Sampsa Hautaniemi, Anna-Kaarina Järvinen, Henrik Edgren, Sanjit K Mitra and Jaakko Astola. *Optimized LOWESS normalization parameter selection for DNA microarray data.* BMC Bioinformatics20045:194 DOI: 10.1186/1471-2105-5-194.

[45] Cho-Jui Hsieh, Matyas A. Sustik, Inderjit S. Dhillon, Pradeep Ravikumar.*Sparse Inverse Covariance Matrix Estimation Using Quadratic Approximation.* Advances in Neural Information Processing Systems 24, 2011, p. 2330-2338.

[46] Christophe Giraud.*Introduction to High-Dimensional Statistics.* December 17, 2014 by Chapman and Hall/CRC.

université PARIS-SACLAY | ÉCOLE DOCTORALE Structure et dynamique des systèmes vivants (SDSV)

Titre : Une méthode d'apprentissage multivariée et pénalisée pour l'inférence jointe des niveaux d'expression et des réseaux de régulation génique

Mots clés : Statistique et application en génomique; Apprentissage statistique; Analyse multivariée; Méthodes pénalisées; Optimisation convexe; Transcriptome.

Résumé : Entre plusieurs conditions biologiques, le comportement d'un gène peut être affecté soit dans son niveau d'expression moyen, soit dans sa relation aux autres, caractérisée par les covariances entre gènes. Ces deux questions sont généralement traitées de manière indépendante en statistique, bien qu'elles soient clairement liées. Afin de palier à ces limitations, cette thèse vise à proposer une modélisation unifiée de ces deux questions pour identifier les gènes clés affectés dans leur moyenne et/ou dans leurs interactions. Le modèle principal est le modèle graphique gaussien avec des pénalisations sur les paramètres de la moyenne et de la matrice de précision.

Title : A multivariate learning penalized method for a joined inference of gene expression levels and gene regulatory networks

Keywords : Statistic and genomic applications; Statistical learning; Multivariate analysis; Penalized method; Convex optimization; Transcriptomic.

Abstract : When comparing different biological conditions, the expression of a gene might shift. It can be a change in terms of its average expression level characterized by its mean. Or it can be a change in terms of its interactions with other genes characterized by the covariance matrix. These two types of events are usually analysed independently even though they are clearly related. In order to alleviate these limitations, we propose in this thesis a unified strategy to address these two questions and identify key genes affected either in terms of their mean or their interactions with other genes. The main statistical model is the Gaussian graphical model with penalizations on the mean and precision matrix parameters.