Comparative evaluation of network reconstruction methods in high dimensional settings.

Henrique Bolfarine

Dissertation presented to Institute of Mathematics and Statistics of University of São Paulo to obtain the title of Master in Science

Program: Statistics
Advisor: Prof. Dr. Anatoli Iambartsev

During the development of this work the author received financial support from CNPq

São Paulo, April of 2017
Comparação de métodos de reconstrução de redes em alta dimensão.

Henrique Bolfarine

DISSERTAÇÃO APRESENTADA
AO
INSTITUTO DE MATEMÁTICA E ESTATÍSTICA
DA
UNIVERSIDADE DE SÃO PAULO
PARA
OBTENÇÃO DO TÍTULO
DE
MESTRE EM CIÊNCIAS

Programa: Estatística
Orientador: Prof. Dr. Anatoli Iambartsev

Durante o desenvolvimento deste trabalho o autor recebeu auxílio financeiro do CNPq

São Paulo, Abril de 2017
Título do trabalho a ser apresentado à CPG para a dissertação

Esta versão da dissertação contém as correções e alterações sugeridas pela Comissão Julgadora durante a defesa da versão original do trabalho, realizada em 17/04/2017. Uma cópia da versão original está disponível no Instituto de Matemática e Estatística da Universidade de São Paulo.

Comissão Julgadora:

- Prof. Dr. Anatoli Iambartsev (orientador) - IME-USP
- Prof. Dr. André Fujita - IME-USP
- Prof. Dr. Eugene Perchesky - The Institute for Information Transmission Problems, Moscow
Acknowledgements

I would like to thank my parents for their support and patience. I would also like to thank my wife Claudia for her support, incentive, and for sharing the difficulties and joys of paternity. Also I would like to thank my sisters for their inspiration, Bernardo for being such a nice nephew and especially my daughter Olivia, who showed me that life can always be happier and more challenging.

I am also very grateful to my adviser, Anatoly Iambartsev, for accepting my choice of subject, for the ever-present orientation, and the exchange of ideas. I would like also to thank the teachers that where important in this journey. From the Statistics department; Fabio Prates, Márcia D’Elia Branco, Miguel Abadi and Silvia Nagib. From the Computer Science department, Professor Roberto Hirata.

I would like to thank my colleagues from master’s and doctoral degrees; Fabio Marques, Lucas Cabral, Moisés Oliveira, Matheus Marques, Brian Melo, Helton Graziadei, Maicon, Jaime Enrique, Guaraci, and Evandro Makiyama. From everyone I learned something new, and together we have sailed the seas of uncertainty.

It is also appropriate to thank Lina Thomas, Phd candidate, whose work inspired this dissertation, and whose help and ideas where fundamental to execute this work.
Agradecimentos

Agradeço aos meus pais, pelo apoio e pela paciência. Agradeço a minha esposa Claudia pela força, o incentivo para continuar e por compartilhar as dificuldades e alegrias da paternidade. Agradeço também minhas irmãs pela inspiração, ao Bernardo por ser um sobrinho legal e principalmente minha filha, Olivia, que me mostrou que vida sempre pode ser mais feliz e desafiadora.

Agradeço ao meu orientador, Anatoly Iambartsev, por aceitar minha escolha de tema, pela orientação sempre presente e pela troca de idéias. Agradeço também aos professores do IME que foram importantes nessa jornada. Do departamento de Estatística os professores; Fabio Prates, Márcia D’Elia Branco, Miguel Abadi e Silvia Nagib. Do departamento de Ciência da Computação, professor Roberto Hirata.

Agradeço também aos meus colegas de mestrado e doutorado; Fábio Marques, Lucas Cabral, Moisés Oliveira, Matheus Marques, Brian Melo, Helton Graziadei, Maicon, Jaime Enrique, Guaraci e Evandro Makiyama. Com todos aprendi algo novo e juntos navegamos os mares da incerteza.

Cabe aqui também agradecer a Lina Thomas, futura doutora, cujo trabalho inspirou essa dissertação, e cujo auxílio e idéias foram fundamentais na execução deste trabalho.
iv
Abstract

BOLFARINE, H. Comparative evaluation of network reconstruction methods in high dimensional settings. 2017. Dissertation (masters) - Institute of Mathematics and Statistics, São Paulo, 2017.

In the past years, several network reconstruction methods modeled as Gaussian Graphical Model in high dimensional settings where proposed. In this work we will analyze three different methods, the Graphical Lasso (GLasso), Graphical Ridge (GGMridge) and a novel method called LPC, or Local Partial Correlation. The evaluation will be performed in high dimensional data generated from different simulated random graph structures (Erdős-Rényi, Barabási-Albert, Watts-Strogatz), using Receiver Operating Characteristic or ROC curve. We will also apply the methods in the reconstruction of genetic co-expression network for the differentially expressed genes in cervical cancer tumors.

Keywords: Network Reconstruction, Graphical Model, Gaussian Graphical Model, GLasso, GGMridge, LPC, Partial correlation.
Resumo

BOLFARINE, H. Comparação de métodos de reconstrução de redes em alta dimensão. 2017. Dissertação Mestrado - Instituto de Matemática e Estatística, São Paulo, 2017.

Vários métodos têm sido propostos para a reconstrução de redes em alta dimensão, que é tratada como um Modelo Gráfico Gaussiano. Neste trabalho vamos analisar três métodos diferentes, o método Graphical Lasso (GLasso), Graphical Ridge (GGMridge) e um novo método chamado LPC, ou Correlação Parcial Local. A avaliação será realizada em dados de alta dimensão, gerados a partir de diferentes estruturas, a partir de grafos aleatórios (Erdös-Renyi, Barabási-Albert, Watts-Strogatz), usando Receptor de Operação Característica, ou curva ROC. Aplicaremos também os métodos apresentados, na reconstrução da rede de co-expressão gênica para tumores de câncer cervical.

Palavras-chave: Reconstrução de redes, Modelos Gráficos, Modelos Gráficos Gaussianos, GLasso, GGMridge, LPC, correlação parcial.
# Contents

| Section                                                                 | Page |
|-------------------------------------------------------------------------|------|
| Abbreviation                                                            | xiii |
| List of Symbols                                                          | xv  |
| List of Figures                                                          | xvii |
| List of Tables                                                           | xix  |
| 1 Introduction                                                          | 1    |
| 1.1 About this work                                                     | 1    |
| 1.2 Organization of the dissertation                                     | 3    |
| 2 Basic Concepts                                                        | 5    |
| 2.1 Introduction                                                        | 5    |
| 2.2 Multivariate Gaussian distribution                                  | 5    |
| 2.2.1 Definition                                                        | 6    |
| 2.3 Conditional Independence                                            | 7    |
| 2.3.1 Definition                                                        | 8    |
| 2.4 Partial correlation                                                 | 9    |
| 2.4.1 Definition - (Pearson correlation)                                | 9    |
| 2.4.2 Definition - (Partial Correlation)                                | 10   |
| 2.5 Graphs                                                              | 11   |
| 2.5.1 Definition - (Graph)                                              | 11   |
| 2.5.2 Definition - (Sub-graph and Complete Graph)                       | 11   |
| 2.5.3 Definition - (Parent, Child, and Neighbor)                        | 12   |
| 2.5.4 Definition - (Cliques and Degree)                                 | 12   |
| 2.5.5 Graph density                                                     | 13   |
| 2.5.6 Adjacency Graph                                                   | 13   |
| 2.6 Regularization                                                      | 14   |
| 2.6.1 Ridge Regression                                                  | 14   |
| 2.6.2 Lasso                                                             | 15   |
3 Graphical Models
  3.1 Introduction ......................................................... 17
  3.2 Direct Acyclic Graphical Models - DAG .......................... 18
    3.2.1 Factorization - DAG ........................................ 18
  3.3 Undirected Graphical Models ...................................... 19
    3.3.1 Factorization - (Undirect Graphical Models) ............ 20
  3.4 The Markov Properties ............................................ 21
    3.4.1 Pairwise Markov Property (PM) ........................... 21
    3.4.2 Local Markov Property (LM) .............................. 21
    3.4.3 Global Markov Property (GM) ............................ 21
  3.5 Gaussian Graphical Models ........................................ 22

4 Methods ................................................................. 27
  4.1 Introduction ....................................................... 27
  4.2 GLasso ................................................................. 27
  4.3 GGMridge ............................................................... 29
  4.4 Local Partial Correlation (LPC) .................................. 31

5 Simulation ............................................................... 37
  5.1 Introduction ....................................................... 37
  5.2 Graph structures .................................................. 38
    5.2.1 Erdős-Rényi ................................................. 38
    5.2.2 Watts-Strogatz model ...................................... 39
    5.2.3 Barabási-Albert model .................................... 39
  5.3 Simulating the Graph structure ................................... 40
  5.4 ROC curve ............................................................ 41
  5.5 Simulation Results ................................................ 43
    5.5.1 Analysis and Conclusion ................................... 45
    5.5.2 ROC curves - average values .............................. 47
    5.5.3 ROC curves - average and standard deviation .......... 48

6 Application to Data .................................................. 49
  6.1 Introduction ....................................................... 49
  6.2 The Data and Methods ............................................. 50
  6.3 Results of the Application ....................................... 51
  6.4 Analysis and Conclusion ......................................... 52
    6.4.1 Reconstructed Networks .................................... 54

A Support Material ..................................................... 61
  A.1 Chapter 1 ............................................................. 61
    A.1.1 Vector Norm ................................................. 61
### Abbreviation

| Abbreviation | Description                           |
|--------------|---------------------------------------|
| GGM          | Gaussian Graphical Models             |
| GLasso       | Graphical Lasso Algorithm             |
| GGMridge     | Graphical Ridge Algorithm             |
| LPC          | Local Partial Correlation method      |
| OLS          | Ordinary Least Squares                |
| ROC          | Receiver Operating Characteristic     |
List of Symbols

= Defined as
∝ Proportional to
≫ Way greater than
|A| The cardinality of set A
\mathcal{N} Gaussian distribution
\mathcal{U} Uniform distribution
R Correlation matrix
K Precision matrix
P Partial Correlation matrix
# List of Figures

2.1 (a) Directed graph. (b) Undirected complete graph. ......................... 12
2.2 Undirected graph with the 4 maximal cliques indicated. ....................... 12
2.3 Graph $G$, with its adjacency matrix $A_G$. ................................ 14
2.4 Ridge Regression (dashed) and the Lasso regression (dotted). ............. 15

3.1 Snowing causes both the traffic jams and snowmen in the parks. ......... 17
3.2 Example of a DAG. .......................................................... 19
3.3 Undirected graphical model ................................................ 20
3.4 (a) GGM graph structure. (b) Adjacency Matrix ......................... 24
3.5 (a) Estimate $\hat{P}$. (b) Estimate $\tilde{P}$. ............................... 24
3.6 Spread of the Partial Correlation estimates in relation to the sample size. . 25

4.1 Neighborhood selection with LPC. ...................................... 33

5.1 Simulated graph structures .................................................. 37
5.2 Representation of a ROC curve. ........................................... 43
5.3 ROC curves generated using Erdős-Rényi model with. ....................... 47
5.4 ROC curves generated using Watts-Strogatz model with. ................... 47
5.5 ROC curves generated using Barabási-Albert model with. ................... 47
5.6 ROC curves generated using Erdős-Rényi model, with standard deviation. 48
5.7 ROC curve generated using Watts-Strogatz model, with standard deviation. 48
5.8 ROC curve generated using Barabási-Albert model, with standard deviation. 48

6.1 ................................................................................. 51
6.2 Reconstructed Network using LPC method. The nodes represent the genes (color in green), the links are the grey edges. The size of the nodes represent the degrees. The smallest node represent 1 degree the largest node represent 14 degrees. In total there are 658 identified nodes and 1774 edges. We can observe that we don have major hubs and the edges seems evenly distributed, that is we have a large number of nodes with same degree. .................. 54
6.3 Reconstructed Network using **GGMridge** method. The nodes generated are in green and the edges are in gray. The smallest size node represent 1 degree the largest node represent 35 degrees. In total there are 558 identified nodes and 1876 edges. The generated network appears to have more hubs and less nodes with same degree. We can see some concentration in some nodes.  

6.4 Reconstructed Network using **GLasso** method. The nodes generated are in red and the edges are in gray. The smallest size node represent 1 degree the largest node represent 188 degrees. In total there are 630 identified nodes and 1942 edges. From the generated network we can observe the presence of hubs with large number of degrees and a large number of nodes with one degree.  

6.5 Reconstructed Network with the **GLasso** and **LPC** method. The 277 red circles are the nodes identified only by the **GLasso** method, the 377 green parallelograms are the nodes identified by the **LPC** method, and the 281 yellow diamonds are the nodes identified by both the **GLasso** and **LPC** method. The largest node, identified by the **GLasso** has 165 degrees the smallest 3.  

6.6 Reconstructed Network **GGMridge** and **LPC** method. The 60 gray inverted triangles are the nodes identified only by the **GGMridge** method, the 88 green parallelograms are the nodes identified by the **LPC** method, and the 570 blue hexagrams are the nodes identified by both the **GGMridge** and **LPC** method. The largest node, identified by both **GLasso** and **LPC** has 35 degrees the smallest 1. We can observe that the methods have identified most of the nodes in common.  

6.7 Reconstructed Network **GGMridge** and **GLasso** method. The 298 red circles are the nodes identified only by the **GLasso** method, the 370 gray inverted triangles are the nodes identified by the **LPC** method, and the 260 orange triangles are the nodes identified by both the **GLasso** and **LPC** method. The largest node, identified by the **GLasso** has 188 degrees the smallest 3.  

6.8 Comparison of the three methods; **GGMridge**, **GLasso**, and **LPC**. The 298 red circles are the nodes identified only by the **GLasso** method, the 370 gray inverted triangles are the nodes identified by the **LPC** method, and the 260 orange triangles are the nodes identified by both the **GLasso** and **LPC** method. The largest node, identified by the **GLasso** has 188 degrees the smallest 3.
List of Tables

5.1 Used $\gamma$ for ROC$_\gamma$ curves. ............................................. 44
5.2 Mean ROC$_\gamma$ curves by method. ............................................. 44
5.3 Mean ROC$_\gamma$ curves by Structure. ......................................... 44
5.4 Sparsity of the generated graphs. ................................................. 44
5.5 ROC$_\gamma$ areas of the means. ................................................... 45
5.6 Difference of ROC$_\gamma$ areas of the standard deviations. ............ 45

6.1 Results of the methods: number of nodes and edges. ....................... 51
6.2 Maximum and minimum degrees of the methods. ............................ 52
Chapter 1

Introduction

1.1 About this work

In the past years the problem of estimating network structures hidden among variables has increased in attention, especially due available massive data sets from different scientific areas (Meinshausen and Bühlmann, 2010). In this context, Graphical Models (Lauritzen, 1996), emerged as a powerful framework for it’s ability to encode conditional dependencies and for it’s convenience in representing distributions over a graph structure that we can analyse further on.

Although this dissertation is not a rigorous work of bioinformatics, the gene co-expression problem inspires it. This means that the gene expressions are studied through the network perspective. The genes of interest are the nodes and their relations are direct or undirected edges (Schäfer and Strimmer, 2005). This relations are commonly identified using Pearson and Spearman correlations but, due to the characteristics of the variables, this often leads to spurious associations from unknown outside sources. Another problem is that, although there’s a growing amount of data available, the sample size, usually, is still very small (on the order of tens) compared to the number of variables analyzed that are in the order of thousands (Castelo and Roverato, 2006). Data with this characteristics are known as high dimensional (Giraud, 2014). This implies that important associations are often hidden in hundreds, even thousands, of unimportant variables that don’t affect the workings of the overall network. For this high dimensional problem we need a clear and acute filter which will help us manage the information, build a parsimonious model and find relevant associations. To address this issue we use the Graphical Models known as Gaussian Graphical Models
(GGM) (Lauritzen, 1996) (Whittaker, 2009), in which, by assuming that the data has gaussian distribution, a series of interesting properties emerges. For example, the Partial correlation matrix plays a key role, regarded as the strength of the an association between every two genes forming an edge in the network.

We analysed three different methods for estimating the network structure from a GGM, each with a different approach to the high dimensional problem. The first, and most known, is the Graphical Lasso, or GLasso (Friedman et al., 2008), largely based on the optimization method by Banerjee et al. (2008), which applied the ideias from Meinshausen and Bühlmann (2006), using the $\ell_1$ restriction (or $\ell_1$ norm\(^1\)) in the estimates of the Precision matrix, which is the inverse of the Covariance matrix. Since then, other authors worked on the same problem trying to improve the methods performance((Witten et al., 2011) (Mazumder and Hastie, 2012). In our study we used the plain GLasso algorithm provided by the authors in their web page\(^2\). The second method analysed is less known, but proposes an interesting alternative to the Graphical Lasso. Developed by Ha and Sun (2014) based on the work of Schäfer and Strimmer (2005), it estimates the Partial correlation matrix using $\ell_2$ penalty and then perform statistical tests based on empirical distributions. We applied the algorithm called GGMridge, provided by the authors\(^3\). Finally, we will present a method which estimates the Partial correlation neighborhood of two variables, that where selected using Pearson correlation, to learn the network structure, known as Local Partial Correlation or LPC (Thomas et al., 2012). We evaluated the three methods performances using simulation studies in which we applied high dimensional data originated from different random graph structures. We have chosen the well-known Erdős-Rényi model (Erdos and Rényi, 1960), the Watts-Strogatz network, which model social interactions (Watts and Strogatz, 1998), and the Barabási-Albert network (Barabási and Albert, 1999), which models biological interactions, among others. The idea is to test if random generated networks, with different structures affects the methods performance. We analyzed the methods using the Receiver Operating Characteristic or ROC curve, which is a diagnostic test for binary classifiers. We also applied the methods to the data from (Zhai et al., 2007), which has genes with chro-

\(^1\)The definition of this norm and the $\ell_2$ norm can be viewed in the Appendix A.1.1.
\(^2\)statweb.stanford.edu/~tibs/glasso/
\(^3\)cran.r-project.org/web/packages/GGMridge/GGMridge.pdf
mosomal expression of tumorous cervical cells. The 1268 genes used in the reconstruction, have chromosomal aberration which where identified by Mine et al. (2013).

1.2 Organization of the dissertation

We organized the work as follows, in chapter 2, we briefly reviewed some basic concepts in Statistics, Graph Theory, and regularization. In chapter 3 we will present core concepts of Graphical Models and GGM. In chapter 4 we present the chosen methods used in our study; GLasso, GGMridge, and LPC. In chapter 5, we present the methods applied in random graphs and Covariance matrices simulation, and the results and evaluation of the comparison. Finally, in chapter 6, we applied the methods presented in 4, in the cervical tumor data, and compared the results between methods.
Chapter 2

Basic Concepts

2.1 Introduction

In this chapter we will present some important concepts that will be useful throughout this work. We will begin presenting the Multivariate Gaussian Distribution and some of it’s properties, next we will define Conditional Independence in 2.3, Pearson Correlation (2.4.1) and Partial Correlation (2.4.2). Later on, in 2.5, we will present some basic concepts of Graph Theory that will be useful in understanding the ideas presented in chapters 3 and 4. Finally we will introduce some basic concepts of regularization in 2.6, that will be an important tool applied in the methods presented in chapter 4. For the readers already familiar with these topics we suggest skipping this chapter.

2.2 Multivariate Gaussian distribution

The Multivariate Gaussian Distribution plays a pivotal role in this dissertation. It has several properties that makes it an ideal model for gene expression (Schäfer and Strimmer, 2005), which is the main application discussed in chapter 6, among other natural phenomena. It is an easy generalization of the univariate Gaussian distribution, and it is defined by its first and second moments. It has the property that zero correlation implies in independence and pairwise independence implies in total independence. The Multivariate Gaussian Distribution is defined formally in Mardia et al. (1980).
2.2.1 Definition

The random vector $X_V = (X_1, \ldots, X_p)^T$, indexed by the set $V = \{1, 2, \ldots, p\}$ has $p$-variate Gaussian distribution, where $\mu \in \mathbb{R}^p$ is the mean vector and $\Sigma$ is a $p \times p$ dimensional Covariance matrix, that is represented by $X_V \sim \mathcal{N}_V(\mu, \Sigma)$. The density of the Multivariate Gaussian Distribution, given $\Sigma$ is non-singular (this means that $\Sigma$ has an inverse), is given by

$$f(x) = \exp\left(-\frac{1}{2}(x - \mu)^T \Sigma^{-1} (x - \mu)\right) \frac{1}{\sqrt{(2\pi)^p \det(\Sigma)}}$$

where $\det(\Sigma)$ is the determinant of $\Sigma$. The first and second moments of Multivariate Gaussian distribution are $E(X_V) = \mu$ and $\text{Var}(X_V) = \Sigma$.

For our means, it will be instructive to examine the conditional distribution of a variable $X_i$, $i \in V$, given the rest $X_{V-i}$, with $V_i \doteq V \setminus \{i\}$, and $X_V \sim \mathcal{N}_V(\mu, \Sigma)$. In this situation the role of the Precision matrix (Dempster, 1969),

$$K = \Sigma^{-1},$$

will become explicit, allowing us to better understand the methods presented in chapter 4. Suppose $X_V \sim \mathcal{N}_V(0, \Sigma)$, where $\mu = 0$, and $V = \{1, 2, \ldots, p\}$, has Gaussian distribution and we partition $X_V$ as $X_V = (X_i, X_{V-i}^T)^T$ then, the conditional distribution of $X_i$ given $X_{V-i}$ is in the form

$$X_i \mid X_{V-i} \sim \mathcal{N}\left(X_{V-i}, \Sigma^{-1}_{V-i} \Sigma_{V-i}, \Sigma_{ii} - \Sigma_{iV-i} \Sigma_{V-i}^{-1} \Sigma_{V-iV-i} \Sigma_{iV-i}^{-1}\right),$$

with $\Sigma$ as,

$$\Sigma = \begin{pmatrix} \Sigma_{ii} & \Sigma_{iV-i} \\ \Sigma_{V-i} & \Sigma_{V-iV-i} \end{pmatrix},$$

where $\Sigma_{iV-i}$, and $\Sigma_{V-i}$ are $1 \times |V-i|$ and $|V-i| \times 1$ sized vectors and $\Sigma_{V-iV-i}$ is a $|V-i| \times |V-i|$ matrix, where $|A|$ is the cardinality of the set $A$. From 2.2, and $\Sigma K = I$, where $I$ is the
identity matrix, and partitioning $K$ as (2.4) we have

$$
\begin{pmatrix}
\Sigma_{ii} & \Sigma_{iV_{-i}} \\
\Sigma_{V_{-i}i} & \Sigma_{V_{-i}V_{-i}}
\end{pmatrix}
\begin{pmatrix}
\kappa_{ii} & \kappa_{iV_{-i}} \\
\kappa_{V_{-i}i} & K_{V_{-i}V_{-i}}
\end{pmatrix}
= 
\begin{pmatrix}
1 & 0 \\
0 & I_{V_{-i}V_{-i}}
\end{pmatrix}
,$$

(2.5)

which we obtain

$$
-\frac{\kappa_{V_{-i}i}}{\kappa_{ii}} = \Sigma_{V_{-i}V_{-i}} \Sigma_{V_{-i}i}, \quad \kappa_{ii}^{-1} = \Sigma_{ii} - \Sigma_{iV_{-i}} \Sigma_{V_{-i}i} \Sigma_{V_{-i}i}
$$

(2.6)

From the properties (2.5) and (2.6) we have the following decomposition, which can be viewed as a linear regression between $X_i$ and $X_{V_{-i}}$,

$$
X_i = X_{V_{-i}}^T \beta^i + \epsilon_i,
$$

(2.7)

where

$$
\beta^i = -\frac{\kappa_{V_{-i}i}}{\kappa_{ii}}, \quad \epsilon_i \sim \mathcal{N}(0, \kappa_{ii}^{-1}).
$$

(2.8)

From (2.7) and (2.8) we can observe that the elements of the Precision matrix affects directly the relation between $X_i$, and $X_{V_{-i}}$. Next we will present definitions that will relate these results with the concept of Conditional Independence.

### 2.3 Conditional Independence

Conditional Independence is a general form of independence, where $X$ and $Y$ are jointly distributed random variables, where $f(x, y) = f(x)f(y)$. It appears in a number of applications, it is the basis of a subject in Computer Sciences, known as Causality (Pearl, 2009), it appears in some Stochastic Processes (Brémaud, 2013), in Bayesian Statistics (Schervish, 2012), and as will see, plays a central role in Graphical Models.
2.3.1 Definition

Let $X$, $Y$ and $Z$ be random variables with a joint probability distribution. The variables $X$ and $Y$ are said to be Conditionally Independent given $Z$, that we denote as $X \perp \perp Y \mid Z$, if

$$f_{XY\mid Z}(x, y \mid z) = f_{X\mid Z}(x \mid z)f_{Y\mid Z}(y \mid z), \quad \text{whenever} \quad f_{Z}(z) > 0. \quad (2.9)$$

In other words if we know the value $Y$, that doesn’t provide additional information about $X$ given we know $Z$. We can write (2.9) in other ways (Lauritzen, 1996) that is useful in some applications, when the following densities and variables, $X$, $Y$, and $Z$ are well defined, we have that,

$$X \perp \perp Y \mid Z \iff f_{XYZ}(x, y, z) = f_{XY}(x, z)f_{YZ}(y, z)/f_{Z}(z), \quad (2.10)$$

$$X \perp \perp Y \mid Z \iff f_{X\mid YZ}(x \mid y, z) = f_{X\mid Z}(x \mid z), \quad (2.11)$$

$$X \perp \perp Y \mid Z \iff f_{XZ\mid Y}(x, z \mid y) = f_{X\mid Z}(x \mid z)f_{Z\mid Y}(z \mid y), \quad (2.12)$$

$$X \perp \perp Y \mid Z \iff f_{XYZ}(x, y, z) = h(x, z)k(y, z), \quad \text{for some } h, k \quad (2.13)$$

$$X \perp \perp Y \mid Z \iff f_{XYZ}(x, y, z) = f_{X\mid Z}(x \mid z)f_{YZ}(y, z). \quad (2.14)$$

Marginal independence, or independence can be obtained by conditional independence. It is as a particular case of (2.9) and it is denoted by $X \perp \perp Y \mid \emptyset$. In this case $\emptyset$ represents the absence of a variable,

$$X \perp \perp Y \mid \emptyset \iff f_{X\mid Y}(x \mid y) = f_{X}(x), \quad \text{whenever} \quad f_{Y}(y) > 0.$$

Some properties emerge from (2.9) and the relations above, that relates conditional independence in situations where we have more variables, or even sets of variables.

- **Symmetry**: $X \perp \perp Y \mid Z \implies Y \perp \perp X \mid Z$.

- **Decomposition**: $X \perp \perp YW \mid Z \implies X \perp \perp Y \mid Z$.

- **Weak Union**: $X \perp \perp YW \mid Z \implies X \perp \perp Y \mid ZW$.

- **Contraction**: $X \perp \perp Y \mid Z \quad \& \quad X \perp \perp W \mid ZY \implies X \perp \perp YW \mid Z$. 
• **Intersection:** $X \perp W|ZY \quad \& \quad X \perp Y|ZW \Rightarrow X \perp YW|Z$.

The meaning of this properties are found in Pearl (2009) and in Lauritzen (1996), where there is an extensive study is this properties (Graphoid Theory), and their proofs all can be obtained from (2.9), and from the forms; (2.10),(2.11), (2.12), (2.13), and (2.14).

### 2.4 Partial correlation

Partial correlation notion, as will see later on, will help us find if the variables are conditionally independent to each other. This is only valid in certain conditions, that will explore, to build a structure that will explicit the relations seen between the variables in 2.3.

To understand Partial correlation we will start with the Pearson correlation.

#### 2.4.1 Definition - (Pearson correlation)

Pearson correlation coefficient, or known only as correlation, is a measure of linear dependence between two variables, $X$ and $Y$ defined as

$$\rho(X,Y) = \frac{Cov(X,Y)}{\sqrt{Var(X)Var(Y)}}. \quad (2.15)$$

From the Cauchy-Schwartz inequality we can see that $|\rho(X,Y)| \leq 1$. If $\rho(X,Y) = 1$, then we have a total positive linear correlation, when it is $\rho(X,Y) = 0$, then there’s no linear correlation, and if $\rho(X,Y) = -1$, then we have a negative linear correlation. It’s easily seen that from (2.15), $\rho(X,X) = 1$, and $\rho(X,Y) = \rho(Y,X)$. From the multivariate perspective, if we have $X_1, \ldots, X_p$ random variables, then the Correlation matrix is given as follows

$$\rho(X_1, \ldots, X_p) = \begin{pmatrix}
\rho(X_1, X_1) & \cdots & \rho(X_1, X_p) \\
\vdots & \ddots & \vdots \\
\rho(X_p, X_1) & \cdots & \rho(X_p, X_p)
\end{pmatrix}. \quad (2.16)$$
2.4.2 Definition - (Partial Correlation)

Given a set of random variables, Partial Correlation measures the degree of association between two random variables given the rest. Let $X_V$ be random vector, indexed by $V = \{1, \ldots, p\}$, the Partial Correlation between $X_i$ and $X_j$ with $i, j \in V$, given $X_{-ij} \doteq X_{V_{-ij}}$, where $V_{-ij} \doteq V \setminus \{i, j\}$, is the Pearson Correlation between the residuals $\varepsilon_i$ and $\varepsilon_j$ from the regression between $X_i$ and $X_j$ against $X_{-ij}$. That is

$$X_i = \alpha_i + X_{-ij} \beta_i + \varepsilon_i, \quad X_j = \alpha_j + X_{-ij} \beta_j + \varepsilon_j. \tag{2.17}$$

From (2.17), we obtain $\hat{X}_i$ and $\hat{X}_j$, and residuals

$$e_i = X_i - \hat{X}_i, \quad e_j = X_j - \hat{X}_j, \tag{2.18}$$

which gives us the Partial correlation in the form

$$\rho(X_i, X_j | X_{V_{-ij}}) = \rho(e_i, e_j) = \rho(X_i - \hat{X}_i, X_j - \hat{X}_j). \tag{2.19}$$

Let $\rho_{ij,V_{-ij}} \doteq \rho(X_i, X_j | X_{-ij})$, if $\rho_{ij,V_{-ij}} \approx 0$ we have that $X_i$ and $X_j$ are not correlated given the rest of the variables $X_{-ij}$. From (2.19) we have that

$$\rho_{ij,X_{-ij}} = \frac{\rho_{ij} - \rho_{ij,V_{-ij}}}{\sqrt{(1 - \rho_{ij,V_{-ij}}^2)(1 - \rho_{ij,V_{-ij}}^2)}}, \tag{2.20}$$

where $\rho_{ij} = \rho(X_i, X_j)$, and $\rho_{i,V_{-ij}} = \rho(X_i, X_{-ij})$, the proof of (2.20) is found in Takahashi (2008). Another approach for calculating Partial Correlation, when the variables are distributed as 2.2.1, is the inverse method which consists in inverting the correlation matrix $R = \{\rho_{ij}\}$, when is non-singular. We define $Q = R^{-1}$ and the Partial correlation as $P = \{\rho_{ij,X_{-ij}}\}$, from the inverse method we have that $\rho_{ij,V_{-ij}}$ is in the form

$$\rho_{ij,V_{-ij}} = -\frac{q_{ij}}{\sqrt{q_{ii}q_{jj}}}, \quad \text{for } Q = \{q_{ij}\}. \tag{2.21}$$

The demonstration of the equivalence between (2.20) and (2.21), is given in Thomas
The elements of (2.20) are tested for the hypothesis $\rho_{ij,V_{-ij}} = 0$, where the $p-values$ of the test are obtained by the equation

$$
\hat{t}_{ij} = \sqrt{n - 2 - p} \cdot \frac{\hat{\rho}_{ij,V_{-ij}}}{\sqrt{1 - \hat{\rho}^2_{ij,V_{-ij}}}},
$$

(2.22)

We have that (2.22) has $t$-Student distribution with $n - 2 - p$ degrees of freedom (Giraud, 2014). This procedure only makes sense when we have that the sample size $n$ is larger than $|V| = p$. Later, in chapter 3, we will discuss the relation between the Precision matrix of the Multivariate Gaussian distribution with Partial correlation and Conditional Independence.

## 2.5 Graphs

Graph Theory, in mathematics, models pairwise relations between objects. It translates these relations to analysable visual structures. In this section we will review the basics of Graph Theory, for a deeper analysis we suggest (Bollobás, 2013).

### 2.5.1 Definition - (Graph)

We define a graph as pair $\mathcal{G} = (V, E)$, where $V$ is the finite set of vertices or nodes, and the set $E$ is the set of edges, with $V \times V \subseteq E$. A graph is directed if we have that for all $i, j \in V$, if the ordered pair $(i, j) \in E$ then $(j, i) \notin E$. In this case we represent this relation as an arrow between $i$ an $j$, as in Figure 2.1a. If the graph is said to be undirected then, for all $i, j \in V$ we have that, if the ordered pair $(i, j) \in E$ then $(j, i) \in E$, in this case we represent this relation as $\{i, j\} \in E$. This relation is represented visually as a line between $i$ and $j$, as in Figure 2.1b.

### 2.5.2 Definition - (Sub-graph and Complete Graph)

For a subset $U \subseteq V$, we have that $U$ induces a sub-graph $\mathcal{G}_U = (U, E_U)$, where $E_U = E \cap (U \times U)$, is the edge set obtained keeping both end points in $U$. A graph is complete if all nodes are joined by an arrow or line as in Figure 2.1b.
2.5.3 Definition - (Parent, Child, and Neighbor)

If there’s an arrow pointing from \( i \) to \( j \), then \( i \) is a parent of \( j \), the set of all parents of a node is denoted as \( \text{pa}(j) \). If \( j \) is child of \( i \), then there’s an arrow pointing from \( j \) to \( i \), the set of all children of a node, denoted by \( \text{ch}(i) \). If there’s a line between two nodes \( i \) and \( j \), then they are adjacent or neighbors. The set of all neighbors or the neighborhood of a node \( i \), denoted by \( \text{ne}(i) \). In Figure 2.1a we have that the parents of node 2 is the set \( \text{pa}(2) = \{1, 3\} \), the set of children of node 1 is \( \text{ch}(1) = \{2, 3\} \). In Figure 2.1b the neighborhood of node 2 is given by \( \text{ne}(2) = \{1, 3, 4, 5\} \).

2.5.4 Definition - (Clique and Degree)

A clique\(^1\) is complete sub-graph \( G_C \) of \( G \), with \( C \subseteq V \), it said to be maximal if it is not contained in other clique. In Figure 2.2 from (Hastie et al., 2015), we have that the set

\(^1\)From the french *clique* means "gang".
of all maximal cliques \( C \) is given by \( C = \{\{1,2,3\}, \{3,4,5\}, \{4,6\}, \{5,7\}\} \). The degree of a node is defined as the number of edges incident to it, or the number of nodes that are in its neighborhood, that is \( \deg(i) = |\text{ne}(i)| \). In Figure 2.2, node 3 has degree \( \deg(3) = 4 \).

### 2.5.5 Graph density

In graph theory, a **dense graph** is a graph in which the number of edges is close to the maximal amount of edges, or a complete graph as in 2.5.2. The opposite, is a graph with only a few edges, known as a **sparse graph**. We define the graph density as

\[
D = \frac{2|E|}{|V||V|-1}.
\]  

where \( |E| \) is the cardinality edge set, and \( |V| \) the cardinality of the node set. In our simulations we will generate graphs with density of \( D \leq 0.05 \). The idea is to approximate the graphs with real world applications, where the networks are sparse as in gene-expression (Albert, 2005), and social phenomena (Watts and Strogatz, 1998).

### 2.5.6 Adjacency Graph

An **adjacency matrix** \( A_G \) is a square matrix with elements \( \{0, 1\} \) used for representing a finite graph \( G \). If the matrix element is 1, it indicates that there is an edge between two nodes. For an undirected graph with no loops, which is an edge that connects a node to itself, the matrix is symmetric with zeros on the diagonal. For a given graph \( G \) we have that \( A_G = \{a_{ij}\} \) is the adjacency matrix with elements defined as,

\[
a_{ij} = \begin{cases} 
1 & \text{if } \{i,j\} \in E, \text{ and } i,j \in V, \\
0 & \text{otherwise} .
\end{cases}
\]  

(2.24)

Where \( a_{ii} = 0 \), \( a_{ij} = a_{ji} \). In Figure 2.3, we have the graph \( G \) on the left with nodes \( V = \{1,2,3,4\} \), and edge set \( E = \{\{1,2\}, \{1,3\}, \{1,4\}, \{2,4\}\} \), we can see that from (2.24), we have the adjacency matrix \( A_G \) of \( G \) on the right of the picture.
BASIC CONCEPTS

\[ \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix} \]

Figure 2.3: Graph $G$, with its adjacency matrix $A_G$.

In chapter 5, we will apply simulation methods, to generate adjacency matrices for random graph structures.

2.6 Regularization

Regularization in Statistics and Machine learning, refers to the process of introducing additional information in a model in order to prevent overfitting, to induce sparsity in the model, among other uses (Abu-Mostafa et al., 2012). In our case, as we will see later on, inducing sparsity by constraining the parametric space lowering the variance and selecting variables that are relevant to the model will be our interest. We will present two of the most known methods for the case where we have a Linear Model (Friedman et al., 2001). The first technic is known as the Ridge regression, introduced by Hoerl and Kennard (1970), the second technic is the Lasso, introduced by Tibshirani (1996). For the sake of simplicity we will present the methods applied to $OLS^2$ in a case with two variables.

2.6.1 Ridge Regression

From the $OLS$ method we have the following expression

\[ \arg \min_{\beta} \| Y - X\beta \|^2_2 = \arg \min_{\beta} \left\{ \sum_{i=1}^{n} \left( y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 \right\} \], \quad (2.25) \]

\(^2\)Ordinary Least Squares, A.1.2.
The Ridge method adds an $\ell_2$ restriction to the value of the parameters $(\beta_1, \beta_2)$ in the form

$$\|\beta\|_2^2 = \sum_{j=1}^{p} \beta_j^2 \leq t, \quad t > 0. \quad (2.26)$$

From the restriction (2.26), we have that (2.25) is rewritten as

$$\arg\min_{\beta} \left\{ \sum_{i=1}^{n} \left( y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 + \lambda_r \sum_{i=1}^{p} \beta_j^2 \right\}, \quad \lambda_R > 0. \quad (2.27)$$

Where $\lambda_r$ is the Ridge restriction coefficient. There is a two-way relation between $\lambda_r$ in (2.27), and $t$ in (2.26). When $\lambda_r$ increases, there is more restriction in the parameters so $t$ decreases, on the other hand when $\lambda_r$ decreases, we have less restriction, so $t$ increases.

### 2.6.2 Lasso

The Lasso regression is similar to the Ridge method, but instead of the $\ell_2$ restriction, we impose the $\ell_1$ norm, so, for the OLS in (2.25), we have the following restriction

$$\|\beta\|_1 = \sum_{j=1}^{p} |\beta_j| \leq t, \quad t > 0, \quad (2.28)$$
which gives us the following expression

\[
\arg \min_{\beta} \left\{ \sum_{i=1}^{n} \left( y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 + \lambda_L \sum_{i=1}^{p} |\beta_j| \right\}, \quad \lambda_L > 0.
\]

Where \( \lambda_L \) is the Lasso coefficient. In Figure 2.4, the Lasso restriction is the diamond shaped (dotted), and we observe that it restrict the values of \( \beta_1 \) and \( \beta_2 \), so that the value of \( \hat{\beta}_2 \) (square-shaped dot) is zero. Both methods have their benefits and their drawbacks.

The Ridge regression reduces parameters variability and return better predictions for the model. In the other hand it doesn’t do variable selection. Contrary to the Ridge, the Lasso does variable selection, but we are not able to calculate \( p\)-values for the chosen parameters, this means we can’t perform hypothesis testing. In chapter 4 we will apply regularization in different methods, mainly for reconstruction of Networks.
Chapter 3

Graphical Models

3.1 Introduction

In the past decades, Graphical Models emerged as an interesting alternative for understanding complex phenomena in different applied fields. It is useful for understanding the joint distribution of an entire set of random variables and helps explicit conditional dependencies notions, representing them as lines between them. To better understand these dependencies we can use a simple example (Giraud, 2014). When it snows in Paris, the Parisians experience long traffic jams, at the same time a large number of snowmen “appears” in the city parks. We can presume that there’s a strong correlation between the size of the traffic jam, and the number of snowmen in the parks. But it’s known that snowmen don’t cause traffic jams and vice-versa. In fact, this dependency is due to the fact that both events have the same cause, the snow. We can represent this relation in the form of a diagram, or graph, as in Figure (3.1).

Figure 3.1: Snowing causes both the traffic jams and snowmen in the parks.

From the facts presented, we can presume that conditionally to the snow, that is, if we set aside the snow factor, the traffic jams and the snowmen at the park are likely not to be associated. To better understand these conditional relations, in this chapter, we will present
the two most known Graphical Models, the Directed Acyclic Graphs, or DAG model, also known as Bayesian Networks and the Undirected Graphical models, in which we will focus in the case where the variables are Gaussian (GGM). In this chapter we will present the basics of the two models and then concentrate in the GGM.

### 3.2 Direct Acyclic Graphical Models - DAG

A DAG is a direct graph $G = (V,E)$, where $V$ is a set of nodes and $E$ an edge set of ordered pairs $E \subseteq V \times V$, with the properties as seen in 2.5.1. It is acyclic, that is, it has no loops. In Figure 2.1a, from chapter 2, we have a cyclic graph. In a DAG each variables corresponds to a random variable $X_V = (X_1, \ldots, X_p)$, indexed by the set of vertices $V = \{1, \ldots, p\}$. A DAG has the property that is overall density is obtained through the graph structure. It factorizes following Conditional Independence rules 2.3, between the node $X_i$, with $i \in V$, and the rest of it’s parents\(^1\), which gives us the following definition.

#### 3.2.1 Factorization - DAG

Let $G = (V, E)$ be a DAG. If the distribution of the random variables $X_V$ is a Graphical Model according to $G$, with $f$ a positive density, then $f$ factors as

$$f(x_1, x_2, \ldots, x_p) = \prod_{i=1}^{p} f(x_i | x_{pa(i)}). \tag{3.1}$$

Applying 3.1 to the graph structure in Figure 3.2, which is a DAG with nine variables, we have the following factorization,

$$f(x_1, x_2, \ldots, x_9) = f(x_1)f(x_2)f(x_3|x_1)f(x_4|x_2)f(x_5|x_2)$$

$$\times f(x_6|x_3,x_4,x_5)f(x_7)f(x_8|x_6)f(x_9|x_7,x_8).$$

Another way to interpret the relation between random variables in a DAG, is to view it under the perspective of Direct Markov property (Ganguly, 2014), which states that every

\(^1\)Definition given in 2.5.3.
Figure 3.2: Example of a DAG.

A variable is conditionally independent of its non-descendants as in (3.2), where the descend-ants of a node $i$, are the nodes that are reached from $i$ by following some sequence of arrows (i included), that we denote by $\text{de}(i)$. For example, in Figure 3.2 the descendants of the node with index 4 is $\text{de}(4) = \{6, 8, 9\}$.

\[ X_i \perp \perp X_{V \setminus \text{de}(i)} | X_{\text{pa}(i)}, \quad i \in V. \]  \hspace{1cm} (3.2)

So in a DAG, the relation between factorization and conditional independence is embedded in the graph structure. The DAG model is also known as Bayesian Networks, although it necessarily adopt a Bayesian statistics framework (Wasserman, 2013). In general DAG are harder to interpret in to make inference on (Hastie et al., 2015). For more information about DAG we suggest Pearl (2009).

### 3.3 Undirected Graphical Models

An Undirected Graphical Model, also known as Markov Random Field, is a graph structure $\mathcal{G} = (V, E)$, with random variables as nodes with undirect edges, as seen in 2.5.1. In this model we can also factorize the density given the graph structure, but differently from the DAG model the Undirected Graphical model factorizes over the set of maximal cliques $\mathcal{C}$,
as seen in 2.5.4, and instead of using densities \( f \), as in 3.1, it uses positive functions \( \psi_C \), with \( C \in \mathcal{C} \), known as *potentials*. The factorization is given as follows.

### 3.3.1 Factorization - (Undirect Graphical Models)

A probability density function \( f \) over an undirected graph structure \( \mathcal{G} = (V, E) \) is given by

\[
f(x_1, x_2, \ldots, x_p) = \frac{1}{Z} \prod_{C \in \mathcal{C}} \psi_C(x_C),
\]

(3.3)

where \( x_C = (x_i, i \in C) \), and \( Z \) is the normalizing constant known as the *partition function*, which is given by

\[
Z = \sum_{x \in \mathcal{X}} \prod_{C \in \mathcal{C}} \psi_C(x_C).
\]

where the set \( \mathcal{X} \) is the set which the random variables \( X_1, X_2, \ldots, X_p \) are defined. The undirected graph in Figure 5.2, with random variables \( (X_1, X_2, \ldots, X_7) \) as nodes, factors as (3.3), and the set of maximal cliques \( \mathcal{C} = \{\{1, 2, 3\}, \{4, 5, 6\}, \{3, 4\}, \{2, 5\}, \{5, 7\}\} \) resulting in the joint density

\[
f(x_1, x_2, \ldots, x_p) \propto \psi_{123}(x_1, x_2, x_3)\psi_{456}(x_4, x_5, x_6)\psi_{34}(x_3, x_4)\psi_{25}(x_2, x_5)\psi_{57}(x_5, x_7),
\]

(3.4)

for some choice of potential function \( \{\psi_{123}, \psi_{456}, \psi_{34}, \psi_{25}, \psi_{57}\} \).

![Undirected Graphical Model](image)

*Figure 3.3: Undirected graphical model*

Next will present an important property that unites the concept of graphical models and factorization to the conditional independence between the variables.
3.4 The Markov Properties

The Markov property relates the structure of the graph with conditional independence properties. In an Undirected Graphical Models, conditional independence are relations between separated subsets of the nodes in the graph. From these separations we have three Markov properties. The first is the Pairwise Markov Property, that states that for any pair of non-adjacent nodes, they are conditionally independent to the each other given the rest. The second is the Local Markov Property, where a node is conditionally independent to its non-neighborhood given it’s neighborhood. And the last is the Global Markov property which relates conditional independence between sets separated by a given set. For a given finite graph $G = (V, E)$, with joint distribution $X_V$. We define these properties as follows.

3.4.1 Pairwise Markov Property (PM)

For any non-adjacent $i, j \in V$, that is $\{i, j\} \in E$, we have that

$$X_i \perp \perp X_j | X_V \setminus \{i, j\}. \quad (3.5)$$

3.4.2 Local Markov Property (LM)

For any node $i \in V$, we have that

$$X_i \perp \perp X_{V \setminus \text{cl}(i)} | X_{\text{ne}(i)}, \quad (3.6)$$

where $\text{ne}(i)$ is the neighborhood of $i$, as seen 2.5.3, and $\text{cl}(i)$ is the closure of $i$, defined as $\text{cl}(i) = \text{ne}(i) \cup \{i\}$.

3.4.3 Global Markov Property (GM)

For any disjoint subsets $U, B, S$ of $V$, such that $S$ separates $U$ from $B$, then

$$X_A \perp \perp X_B | X_S. \quad (3.7)$$
These properties are not equivalent. The *Global Markov property* implies in the *Local Markov property*, that in turn, implies in the *Pairwise Markov Property*, so

\[(\text{GM}) \implies (\text{LM}) \implies (\text{PM}). \quad (3.8)\]

If we have a positive density \( f(x_1, x_2, \ldots, x_p) > 0 \), then there is an equivalence between (GM), (LM), and (PM), so we have that

\[(\text{GM}) \iff (\text{LM}) \iff (\text{PM}) \quad (3.9)\]

These properties and examples of equivalence are largely discussed in *Lauritzen (1996)*.

Contrary to the DAG, where there is clear equivalence between factorization and the Markov Property (3.2), in the case of Undirected Graphical Models this property is only possible by the Hammersley-Clifford theorem\(^2\), where it is shows that a Gibbs random field satisfies every Markov property as seen in 3.4, and is represented by a Gibbs measure for an appropriate (locally defined) energy function, then we can apply the factorization defined in (3.3). The inverse is also true, with the factorization represented as (F), that gives us the relation in 3.10.

\[(F) \iff (\text{GM}) \iff (\text{LM}) \iff (\text{PM}). \quad (3.10)\]

### 3.5 Gaussian Graphical Models

Gaussian Graphical Models, or GGM, are Graphical Models where the variables associated to the graph structure are Gaussian, as seen in 2.2.1, this fact provides an useful framework for representing network structures, which allows Conditional Independence identification among the variables. In a GGM, \( X_V \), has \( p \)-dimensional Gaussian distribution, with non-singular Covariance matrix \( \Sigma \). The vector \( X_V \), indexed by the set \( V = \{1, \ldots, p\} \), models a graph structure \( \mathcal{G} = (V, E) \). From the Gaussian distribution properties in 2.2.1, it is known that zero Partial correlation, as seen in 2.4.2, between variables \( X_i \) and \( X_j \) with

\(^2\)Hammersley, J. M., Clifford, P. (1971), Markov Fields on Finite Graphs and Lattices”, unpublished, available at http://www.statslab.cam.ac.uk/~grg/books/hammfest/hamm-cliff.pdf.
\( i, j \in V \) is equivalent to conditional independence of the variables given the rest of the variables in \( X_{V\setminus ij} \), where \( V\setminus ij = V \setminus \{i, j\} \). We presented in (2.21), a relation between the correlation matrix \( R \), and the Partial correlation matrix \( P \), this relation is easily extended (Thomas, 2012) to the elements of the Precision matrix \( K = \Sigma^{-1} \) in the form

\[
\rho_{ij, V\setminus ij} = -\frac{\kappa_{ij}}{\sqrt{\kappa_{ii}\kappa_{jj}}}, \quad K = \{\kappa_{ij}\}, \quad i, j \in V. \tag{3.11}
\]

From the the Multivariate Gaussian properties, the density in 2.2.1 can be rewritten as

\[
f(x) = \exp \left( \alpha + \beta^T x - \frac{x^T K x}{2} \right), \quad x = (x_1, x_2, \ldots, x_p), \tag{3.12}
\]

where \( \alpha \) is the normalizing constant, and \( \beta^T x = \Sigma^{-1} \mu \). We can observe that if the elements of the Precision matrix are zero, \( \kappa_{ij} \), we can factorize (3.12) into a form \( f(x_i, x_j, x_{\setminus ij}) = h(x_i, x_{\setminus ij})k(x_j, x_{\setminus ij}) \), for \( i \in V \) and \( x_{\setminus ij} = x_{V\setminus ij} \). From 2.13, we have that \( X_i \) and \( X_j \) are conditionally independent given the rest. This gives us the following relation between the Precision matrix, the Partial correlation and Conditional Independence,

\[
\kappa_{ij} = 0 \Leftrightarrow \rho_{ij, V\setminus ij} = 0 \Leftrightarrow X_i \perp \perp X_j | X_{V\setminus ij}, \quad i, j \in V. \tag{3.13}
\]

From the relation in (3.13), we can estimate the graph structure, through the Partial correlation or Precision matrix. An easy way is to use the OLS estimate of \( \Sigma \), for the case we have \( X_V \sim \mathcal{N}_V(0, \Sigma) \), this estimate is in the form

\[
\hat{\Sigma} = \frac{1}{n} X^T X, \tag{3.14}
\]

where \( X \) is a \( n \times p \) dimensional data matrix. From (3.14) we estimate the Precision matrix \( \hat{K} \), that in turn, from (3.11) gives us Partial Correlation matrix estimation

\[
\hat{P} = -\text{scale}(\hat{K}), \tag{3.15}
\]

where \( \text{scale}(A) = \text{diag}(A)^{-1/2} A \text{diag}(A)^{-1/2} \), with \( A \) a square matrix, and \( \text{diag}(A) \) is the square empty matrix with the diagonal of \( A \) in its diagonal.
In Figure 3.4a we have a graph structure with $|V| = 20$ nodes with its adjacency matrix, represented as a heat-map, in Figure 3.4b. From this adjacency matrix we generated a Covariance matrix, using a procedure that will present in chapter 5, in which we generated a sample of size $n = 200$, $\hat{P}$ is estimated using (3.15). We can observe clearly, that the graph structure in Figure 3.4a is “printed” in the Partial correlation estimate, making the reconstruction of the network possible.

![Graphical Model](image)

**Figure 3.4:** (a) GGM graph structure. (b) Adjacency Matrix

For the case that $n < p$, we have that data is known as high dimensional, that means that $P$ can’t be estimated using (3.15), due to the fact that $\Sigma$ is a singular matrix. If we wish to estimate $P$ in this case we have to use other means. In Figure 3.5b, we have an estimate $\tilde{P}$, from the same data as used for the estimate on Figure 3.5a, but with sample size reduced.

![Covariance Matrices](image)

**Figure 3.5:** (a) Estimate $\hat{P}$. (b) Estimate $\tilde{P}$. 
to $n = 10$. The estimate $\hat{P}$ could only be obtained using the generalized inverse\(^3\) of $\hat{\Sigma}$. We can observe in Figure 3.5b that estimates in $\hat{P}$ are different from the ones in Figure 3.5a, the values are larger than those from Figure 3.5b. In fact if we reduce the sample size of $n$ in relation to $p$, even if $n > p$, the Partial Correlation spread will increase, as in the plot presented Figure 3.6. When $n < 10$ we have the generalized inverse.

![Figure 3.6: Spread of the Partial Correlation estimates in relation to the sample size.](image)

Using the generalized inverse is not the only method for estimating $P$ in high dimensional cases, in fact this method is not very precise since there’s no unicity in generalized inverses. Over the years many methods where developed, examples include Schäfer and Strimmer (2005), Banerjee et al. (2008), and Peng et al. (2009). In the next chapter we will present three of these methods, the *Graphical Lasso*, or GLasso, the method known as GGMridge, and the LPC method.

\(^3\)The generalized inverse is defined in A.2.1.
Chapter 4

Methods

4.1 Introduction

We will briefly present the methods evaluated in chapters 5 and 6. We will start with the **GLasso**, and present its relation to the regularized log-likelihood function, next we will present the **GGMridge** and discuss some aspects of its regularization method, and finally we will present the **LPC** method, that selects the neighborhood of a given variable based on it’s correlation.

4.2 GLasso

The **GLasso** method has its roots on the work of Meinshausen and Bühlmann (2006), in which Gaussian Graphical Models are estimated by fitting a $\ell_1$ penalty for the elements of the Precision matrix $K$ to increase its sparsity. Before we continue it will be useful to present the log-likelihood function for $K$. Suppose $X$ is a $n \times p$ dimensional data matrix with distribution $X \sim V$, as seen in section 3.5, and Precision matrix $K$, it is known (Friedman et al., 2001), that the log-likelihood $L(X; K)$, is in the form

$$L(X; K) = \log \det(K) - tr(SK),$$

where $\det(A)$ is the determinant of $A$, $tr(A)$ is it’s trace, and $S = n^{-1} \sum_{k=1}^{n} X^{(k)}X^{(k)T}$, where $X^{(k)}$, $k \in \{1, \ldots, n\}$ are the lines of $X$, is the empirical covariance matrix. The maximum likelihood estimates (MLE) of (4.1) converges to $K$ as $n$ goes to infinity (Hastie et al., 2015),
but, in practice, several applications have much more variables than samples, \( n \ll p \), which implies that \( S \) is rank degenerated and makes the MLE method unappliable. Banerjee et al. (2008) approached this problem by solving the penalized log-likelihood

\[
\mathcal{L}(X; K) = \log \det(K) - tr(SK) - \lambda ||K||_1
\]

showing that it is convex, and performing optimized methods in a block coordinate fashion (Banerjee et al., 2008). The GLasso method approaches 4.2 by giving the exact maximizer of the penalized log-likelihood in the form

\[
K^{-1} - S - \lambda \text{Sign}(K) = 0.
\]

In (4.3) we have the sub gradient notation (Friedman et al., 2001) for the operator \( \text{Sign}(\cdot) \), defined as \( \text{Sign}(\kappa_{ij}) = \text{sign}(\kappa_{ij}) \) if \( \kappa_{ij} \neq 0 \), else if \( \kappa_{ij} = 0 \), then \( \text{Sign}(\kappa_{ij}) \in [-1, 1] \). From the properties of the precision matrix, \( W = K^{-1} \), and the partition

\[
\begin{pmatrix}
W_{11} & w_{12} \\
w_{21} & w_{22}
\end{pmatrix}
\begin{pmatrix}
K_{11} & K_{12} \\
K_{21} & K_{22}
\end{pmatrix}
= \begin{pmatrix}
I & 0 \\
0 & 1
\end{pmatrix},
\]

we have that (4.3) can be rewritten

\[
W_{11} + s_{12} + \lambda \text{Sign}(\beta) = 0,
\]

where \( w_{11}/\beta = -W_{11}K_{12} \), as derived in (2.8). In Friedman et al. (2008), estimating the system (4.4) is equivalent to estimating equations for the LASSO regression (2.28). The algorithm applies coordinate descent methods (Friedman et al., 2007) to recursively solve (4.4). The procedure is explained in detail in Friedman et al. (2001), we present the algorithm of the GLasso method in Algorithm 1. There are other ways to update \( \hat{\beta}_j \) (Banerjee et al., 2008), but the coordinate descent is the most intuitive to understand. The method updates as follows,

\[
\hat{\beta}_j \leftarrow S(s_{12(j)} - \sum_{k \neq j} w_{11(kj)}\hat{\beta}_k, \lambda_L)/w_{11(j)}
\]
Algorithm 1 GLasso

procedure
Initialize $W = S + \lambda_L I$, (the diagonal of $W$ doesn’t change);

for $j = 1, 2, \ldots, p, 1, \ldots$ until convergence do

1. Partition $W$ into part 1: all but the $j$th row and column and
part 2: the $j$th row and column;
2. Solve $W_{11} \beta - s_{12} + \lambda_L \text{Sign}(\beta) = 0$, using (4.5);
3. Update $w_{12} = W_{11} \hat{\beta}$;

end for
Solve, for $\hat{\kappa}_{12} = -\hat{\beta} \hat{\theta}_{22}$ where $1/\hat{\kappa}_{22} = w_{22} - w_{12}^T \hat{\beta}$.

end procedure

for $j = 1, 2, \ldots, p - 1, 1, 2, \ldots, p - 1, 1, 2, \ldots$, where $s_{12(j)}$ are the elements of the vector $s_{12}$, $w_{11(j)}$ are elements of of the partitioned matrix $W_{11}$ and $S$, is the soft threshold operator

$$S(x, t) = \text{sign}(x)(|x| - t)_+.$$

The coordinate descent method makes the method solve sparse problem with 1000 nodes in less than a minute, and can work easily with different values of $\lambda_L$. But, the are some drawbacks, the method doesn’t estimate correctly the Partial correlations, it “learns” the structure, and there’s a need to apply cross validation methods to select the best coefficient $\lambda_L$. In Bien et al. (2011) there is an example of cross-validation for estimating Covariance matrix, but can also be applied for $K$.

4.3 GGMridge

Contrary to the GLasso method seen in 4.2, that tries to “learn” the graph structure, the GGMridge (Ha and Sun, 2014) estimates the original network by identifying the zero elements of the Partial correlation matrix, and re-estimates the remaining non-zero values.

The idea behind this procedure is to obtain not only the network structure but the true Partial correlation values, that in the gene co-expression perspective measures the strength of the expression between any two genes.
The method basically consists of three stages, in the first stage we obtain a penalized estimation of the Partial correlation matrix using a Ridge “ad-hoc” solution (Hoerl and Kennard, 1970). Next it uses a threshold $\alpha_r \in (0, 1)$ to select the non-zero elements based on mixture distributions related to the hypothesis test of the elements that where estimated. Finally it re-estimates the Partial correlation of the remaining non-zero elements. The \texttt{GGMridge} method is largely based on the work of Schäfer and Strimmer (2005), but instead of using the Ridge “ad-hoc” solution based on generalized inverse of the empirical Covariance matrix $\mathbf{S}$, to guarantee that in high dimensional settings $\mathbf{S}$ would be non-singular.

The method starts by obtaining the estimate of the Partial correlation matrix $\tilde{\mathbf{P}}_{\lambda_r}$ from the empirical Covariance matrix $\mathbf{S}$, obtained from $\mathbf{X}_V$ as seen in 2.2.1 (centered in zero, with spread 1), with $n < p$, in the form

$$\tilde{\mathbf{P}}_{\lambda_r} = - \text{scale}((\mathbf{S} + \lambda_r \mathbf{I}_p)^{-1}), \quad (4.6)$$

where the operator \texttt{scale} is as defined in (3.15). In (4.6), the \texttt{scale} function is applied to $(\mathbf{S} + \lambda_r \mathbf{I}_p)^{-1}$, which is an estimate of the precision matrix $\tilde{\mathbf{K}}_{\lambda_r}$, where $\lambda_r > 0$ is known as the ridge coefficient (actually this “ad-hoc” solution is not the minimization of the log-likelihood function, with $\ell^2$ penalty, a formal solution for this minimization is seen in Warton (2008)), which can be obtained by cross-validation\(^1\), and $\mathbf{I}_p$ is a $p \times p$ identity matrix. The elements $\{\tilde{\rho}_{ij}^{\lambda_r}\} = \tilde{\mathbf{P}}_{\lambda_r}$ are tested using the Fisher $z - transformation$

$$\psi(\tilde{\rho}_{ij}^{\lambda_r}) = \frac{1}{2} \log \left\{ (1 + \tilde{\rho}_{ij}^{\lambda_r})/(1 - \tilde{\rho}_{ij}^{\lambda_r}) \right\}, \quad i, j \in V, \text{ with } i \neq j. \quad (4.7)$$

From (4.7), follows a mixture distribution of $f_0(\psi)$, with $\psi = \psi(\tilde{\rho}_{ij}^{\lambda_r})$, and alternative distribution $f_a(\psi)$, where $H_0 : \rho_{ij}^{\lambda_r} = 0$, $H_a : \rho_{ij}^{\lambda_r} \neq 0$. Due to assumption of sparsity of the original $\mathbf{P}$ we assume that the mixture

$$f(\psi) = \eta_0 f_0(\psi) + \eta_a f_a(\psi), \quad (4.8)$$

have the majority of observations around $H_0$, so we have that $\eta_0 \gg \eta_a$, with $\eta_0 + \eta_a = 1$. We

\(^1\)See supplementary B from (Ha and Sun, 2014).
estimate the distribution of (4.8) using Efron’s central matching method (Efron, 2004) where 
\( f_0(\psi) \) has gaussian distribution \( \mathcal{N}(\mu_0, \sigma_0) \), and \( f_a(\psi) \) is left unknown. Both distributions are 
matched at their central parts (Ha and Sun, 2014) then, using polynomial Poisson regression, 
we estimate \( f(\psi) \). From this distribution we have the \( p \)-values, and the selected non-zero 
estimates trough a threshold \( \alpha_r \) (that can be learned by cross validation along \( \lambda_r^2 \)), used in 
a false discovery rate test (Benjamini and Hochberg, 1995).

This process generates a Partial correlation graph structure where each variable \( X_i \) with \( i \in V \), has a neighborhood of non-zero elements \( \hat{\text{ne}}_i \subset V \). From the properties of the 
partitioned distribution seen in chapter 2, in (2.3), (2.6), (2.7) and (2.8), the estimates of 
the precision matrix \( K \) are

\[
\hat{\kappa}_{ii} = \frac{(n - |\hat{\text{ne}}_i|)}{||X_i - \hat{X}_{\text{ne}_i}\hat{\beta}_{i,\text{ne}_i}||^2_2} \quad \text{and} \quad \hat{\kappa}_{ij} = -\hat{\kappa}_{ii}\hat{\beta}_{i,\text{ne}_i} \hat{\beta}_{i,\text{ne}_j} \tag{4.9}
\]

Based on the sparsity assumption we have that \( |\hat{\text{ne}}_i| < n \), with \( \hat{\beta}_{i,\text{ne}_j} = (X_{\text{ne}_i}^T X_{\text{ne}_i})^{-1} X_{\text{ne}_i} X_i \),
where \( X_{\text{ne}_i} \) is a \( |\hat{\text{ne}}_i| \times n \) submatrix of the data matrix \( X \). Since the solution is not symmetric to 
the elements off-diagonal (4.9), we have that the final estimate is

\[
\hat{\kappa}_{ij} = \hat{\kappa}_{ji} = \text{sign}(\hat{\beta}_{j,\text{ne}_i}) \sqrt{\hat{\kappa}_{ij}\hat{\kappa}_{ij}} \quad \text{for} \quad i \neq j, \tag{4.10}
\]

with \( \hat{\beta}_{j,\text{ne}_i} \) elements of the vector \( \hat{\beta}_{j,\text{ne}_i} \). To better understand the method, we have the 
\textbf{GGMridge} represented in Algorithm 2 in simpler way that presented in Ha and Sun (2014).

\section{4.4 Local Partial Correlation (LPC)}

As seen in section 3.5, it is known, in multivariate models, when \( n \) is smaller than \( p \), that 
most statistical methods are compromised. A popular strategy in bioinformatics, where this 
problem is common, is to compute the standard Pearson correlation between any two genes to

\footnotetext{2}{This method is explained in the supplementary materials of Ha and Sun (2014), Section B.}
Algorithm 2 GGMridge

\textbf{procedure}
1. $\tilde{P}_\lambda = -\text{scale}((S + \lambda r I_p)^{-1})$;
2. Test $H_0 : \rho^\lambda_{ij} = 0$ versus $H_a : \rho^\lambda_{ij} \neq 0$ with
   
   $\psi(\tilde{\rho}^\lambda_{ij}) = 1/2 \log\{(1 + \tilde{\rho}^\lambda_{ij})/(1 - \tilde{\rho}^\lambda_{ij})\}$, for $\{\tilde{\rho}^\lambda_{ij}\} = \tilde{P}_\lambda$;

   Estimates $p$-values from $f(\psi) = \eta_0 f_0(\psi) + \eta_a f_a(\psi)$ as in (Ha and Sun, 2014);
3. If the $p$-values for $\rho^\lambda_{ij}$ for $i < j$, is less than $\alpha_r$ define a neighborhood $\text{ne}_i$ for $X_i$;
4. $\hat{\beta}_i^{\text{ne}_i} = (X^T_{\text{ne}_i} X_{\text{ne}_i})^{-1} X_{\text{ne}_i} X_i$, for $i \in V$;
5. Estimate $\kappa_{ii} = (n - |\text{ne}_i|)/||X_i - X_{\text{ne}_i}\hat{\beta}_i^{\text{ne}_i}||^2_2$ and $\tilde{\kappa}_{ij} = -\kappa_{ii}\hat{\beta}_j^{\text{ne}_j}$;
6. Estimate $\hat{\kappa}_{ij} = \kappa_{ij} \sqrt{||\kappa_{ij}||}$ for $i \neq j$;
7. Return $\hat{P} = -\text{scale}(\hat{K})$.
\textbf{end procedure}

learn a network structure, called “relavance network”, where a missing edge denotes marginal independence. However, high correlation may indicate different interactions between genes (Schafer and Strimmer, 2005).

Inspired by the bioinformatics simple approach came the idea to select “local” partial correlations. What we mean by “locally selected” is that for each Pearson correlated pair chosen through a threshold in it’s $p$-values, we have a neighborhood of nonzero elements. If this neighborhood has more variables than samples, we select fractions of the $n$ highest Pearson correlations, and build a Partial correlation submatrix, that is further tested if the elements are non-zero, which indicates if an edge is present between this pair, which is called this method local Partial correlation or \textbf{LPC} (Thomas et al., 2012). In Figure 4.1 we can observe the neighborhood selected with the Pearson correlation (traced), and then the fraction of this neighborhood being locally selected by the highest correlation values (dark dots), which are the values used for calculating the Partial correlation between $X_i$ and $X_j$.

So, for a multivariate Gaussian distribution $X_V$, as in 2.2.1, for a sample with size $n$, with $p = |V|$ variables, and $n < p$. The method starts by calculating $\rho_{ij} = \rho(X_i, X_j)$, for every $i, j \in V$, which is the Pearson correlation defined in 2.4.1. For each $\hat{\rho}_{ij}$, we perform a hypothesis test $H_0 : \rho_{ij} = 0$ versus $H_a : \rho_{ij} \neq 0$, where the $p$-values $\text{pv}_{ij}$, obtained trough squared $t$-statistics as seen in (2.22). If $\text{pv}_{ij}$ is small, there is an indication that there might
be an interaction between $X_i$ and $X_j$, if $p_{ij}$ is high then there might be no interactions. We then consider a threshold $\alpha \hat{\rho} \in [0, 1]$ to select which values of $\hat{\rho}_{ij}$ that are to be further analyzed. From this selection, we construct a new matrix $\tilde{R}$ such that each row and column is filled with the elements

$$\tilde{R} = \{\tilde{r}_{ij}\}, \quad \tilde{r}_{ij} = \begin{cases} \hat{\rho}_{ij}, & \text{if } p_{ij} < \alpha \hat{\rho}, \text{ for all } i, j \in V; \\ 0, & \text{otherwise}. \end{cases} \tag{4.11}$$

For a fixed $i, j \in V$ such that $\tilde{r}_{ij} \neq 0$, we will define the local neighborhood of $X_i$ and $X_j$ as the subsets of $V$. So, for each $k \in V \setminus \{i\}$ and $l \in V \setminus \{j\}$, with $\tilde{r}_{ik} \neq 0$ we have the subsets

$$\hat{\text{ne}}_{i}^{\text{local}} = \{k \in V \setminus \{i\}; \tilde{r}_{ik} \neq 0\}, \quad \text{and} \quad \hat{\text{ne}}_{j}^{\text{local}} = \{l \in V \setminus \{j\}; \tilde{r}_{jl} \neq 0\}. \tag{4.12}$$

For a fixed pair $\{i, j\} \in V$ with conditions (4.12), we have the local neighborhood of the pair defined as

$$\hat{\text{ne}}_{ij}^{\text{local}} = \{\hat{\text{ne}}_{i}^{\text{local}} \cup \hat{\text{ne}}_{j}^{\text{local}}\} \setminus \{i, j\}.$$  

If $|\hat{\text{ne}}_{ij}^{\text{local}}| > n$, we select the $\lfloor n/2 \rfloor$ (where $\lfloor n/2 \rfloor$ is the greatest integer function), highest correlations associated with $\{i, j\}$, and the local neighborhood associated with this correlations, $\hat{\text{ne}}_{ij}^{LPC} \subset \hat{\text{ne}}_{ij}^{\text{local}}$, where

$$\hat{\text{ne}}_{ij}^{LPC} = \{k \in \hat{\text{ne}}_{ij}^{\text{local}}; \tilde{r}_{\{i,j\}k}^{(1)}, \ldots, \tilde{r}_{\{i,j\}k}^{(\lfloor n/2 \rfloor)}\}. \tag{4.13}$$
where $\hat{r}^{(m)}_{(i,j)k}$ are the sorted correlation values from $m = 1, \ldots, \lfloor n/2 \rfloor$. From 4.13, we determine a local correlation submatrix $\hat{R}^{LPC}$ of $\hat{R}$, with elements $\hat{R}^{LPC} = \{\hat{\rho}^{LPC}_{kl}\}$, with the variables indexed by the set $\ne^{LPC}_{ij} \cup \{i, j\}$, in the form

$$\hat{\rho}^{LPC}_{kl} = \rho(X_{(i,j)}, X_{\ne^{LPC}_{ij}}), \quad \text{with} \quad k,l \in \ne^{LPC}_{ij} \cup \{i, j\}.$$  

From the properties of the Partial correlation matrix seen in (2.21), we define the matrix $\hat{Q}^{LPC} = (\hat{R}^{local})^{-1}$, which gives us the local Partial correlation, or LPC estimate, of $X_i$ and $X_j$, in the form

$$\hat{\rho}^{LPC}_{ij, \ne^{LPC}_{ij}} = \frac{-q^{LPC}_{ij}}{\sqrt{q^{LPC}_{ii}q^{LPC}_{jj}}}, \quad \text{with} \quad \hat{Q} = \{q^{LPC}_{ij}\}.$$  

To test the hypothesis $H_0 : \rho_{ij, \ne^{LPC}_{ij}} = 0$ versus $H_a : \rho_{ij, \ne^{LPC}_{ij}} \neq 0$, that is, to test if there is an edge between $X_i$ and $X_j$ we apply the $z$-transformation,

$$\psi(\hat{\rho}^{LPC}_{ij, \ne^{LPC}_{ij}}) = \frac{1}{2} \log\{(1 + \hat{\rho}^{LPC}_{ij, \ne^{LPC}_{ij}})\}/(1 - \hat{\rho}^{LPC}_{ij, \ne^{LPC}_{ij}}),$$  

for significance level of $\alpha_{LPC}$ the null hypothesis is rejected if

$$\sqrt{n - |\ne^{LPC}_{ij}| - 3.\psi(\hat{\rho}^{LPC}_{ij})} > \Phi^{-1}(1 - \alpha_{LPC}/2),$$  

where $\Phi$ is the cumulative Gaussian distribution $\mathcal{N}(0, 1)$. For more information about the test, we suggest Kendall et al. (1946).

In Algorithm 3, we have the routine for implementing the LPC method, and in B.1.1 we have the R implementation used in the simulations in 5.5, in chapter 5. We can observe from the passages shown in this section, that the LPC method is simple to understand and to implement. But there is a major drawback, that the time the method takes to complete the routine increases exponentially if the number of variables increases. To overcome this problem, we applied parallelization of some of the routines in the method, using the R package doParallel,\(^3\) to try improve its performance.

---

\(^3\)https://cran.r-project.org/web/packages/doParallel/index.html.
Algorithm 3 LPC

procedure
1. Estimate $\hat{\rho}_{ij} = \rho(X_i, X_j)$, for every $i, j \in V$;
2. Calculate the p-values $p_{ij}$ for every $\rho_{ij}$;
3. Initialize $\tilde{R} = \{\tilde{r}_{ij}\}$ an empty matrix, given a threshold $\alpha_{\hat{\rho}}$;

$$\tilde{r}_{ij} = \begin{cases} \hat{\rho}_{ij}, & \text{if } p_{ij} < \alpha_{\hat{\rho}}, \text{ for all } i, j \in V; \\ 0, & \text{otherwise}. \end{cases}$$

for every pair $i, j \in V$, and $\tilde{r}_{ij} \neq 0$
do
1. Define $\tilde{n}_{i, local} = \{k \in V \setminus \{i\}; \tilde{r}_{ik} \neq 0\}$ and $\tilde{n}_{j, local} = \{l \in V \setminus \{j\}; \tilde{r}_{jl} \neq 0\}$;
2. Define $\tilde{n}_{ij, local} = \{\tilde{n}_{i, local} \cup \tilde{n}_{j, local}\} \setminus \{i, j\}$

3. if $|\tilde{n}_{ij, local}| > n$, define $\tilde{n}_{LPC}_{ij} = \{k \in \tilde{n}_{ij, local}; \tilde{r}_{ij}^{(1)}, \tilde{r}_{ij}^{(2)}, \ldots, \tilde{r}_{ij}^{(\lfloor n/2 \rfloor)}\}$;
4. $\hat{\rho}_{ki, LPC} = \rho(X_{\{i,j\}}, X_{\tilde{n}_{LPC}_{ij}})$, with $k, l \in \tilde{n}_{LPC}_{ij} \cup \{i, j\}$
5. $\hat{Q}_{LPC} = (\hat{R}_{local})^{-1}$;
6. $\hat{\rho}_{ij, \tilde{n}_{LPC}_{ij}} = -\hat{q}_{ij, LPC}/\sqrt{\hat{q}_{ii, LPC} \hat{q}_{jj, LPC}}$, with $\hat{Q} = \{\hat{q}_{ij, LPC}\}$
7. Test $H_0 : \rho_{ij, \tilde{n}_{LPC}_{ij}} = 0$ versus $H_a : \rho_{ij, \tilde{n}_{LPC}_{ij}} \neq 0$ with (4.15) and (4.15).
end for
end procedure
Chapter 5

Simulation

5.1 Introduction

In this chapter we will implement and evaluate the methods presented in chapter 4. The idea is to reconstruct different network structures and compare the results using ROC curves. The networks are generated using Random Graph models (Bollobás, 1998). They are random because there’s an underlying probability distribution that generates the graph. We have chosen three known models, the Erdős-Rényi, the Watts-Strogatz, and the Barabási-Albert (Figures 5.1a, 5.1b, 5.1c). The models are generated using random processes that will be further explained in 5.2. For each model we will generate a Covariance matrix from the graph’s adjacency matrix (seen in section 2.5.6) next, we simulate a high dimensional sample from a multivariate Gaussian distribution. The details of this transformation are given in section 5.3. The results of the comparison are given 5.5.

(a) Erdős-Rényi model. (b) Watts-Strogatz model. (c) Barabási-Albert.

Figure 5.1: Simulated graph structures.
5.2 Graph structures

5.2.1 Erdös-Rényi

The Erdös-Rényi model, first introduced by Erdos and Rényi (1960), is a random graph with well-known properties (Bollobás, 1998). The set of edges are distributed equally and independently across the node set with fixed probability. The graph $\mathcal{G}_{ER} = (V, E)$ is obtained by starting with a node set $V = \{1, \ldots, p\}$, in which we connect any two nodes with an edge with probability $p_{ER}$. We will apply this method to generate in 5.3, a Covariance matrix from the adjacency matrix $A_{\mathcal{G}_{ER}}$, in order to perform the comparisons in 5.5. In Algorithm 4, we have a simple method for generating $A_{\mathcal{G}_{ER}}$.

Algorithm 4 Erdös-Rényi

procedure
Initialize $A_{\mathcal{G}_{ER}} = \{a_{ij}\}$, empty matrix and $p_{ER}$:

for $i < j$, for all $i, j \in V$: do

1. $p_{ij} \sim \mathcal{U}(0, 1)$;

2. $a_{ij} = \begin{cases} 
1, & \text{if } p_{ij} < p_{ER}, \\
0, & \text{otherwise}; 
\end{cases}$

end for
$A_{\mathcal{G}_{ER}} = A_{\mathcal{G}_{ER}} + A_{\mathcal{G}_{ER}}^T$;

end procedure

Although the Erdös-Rényi model is relevant due it’s simplicity and known properties, it fails to capture networks seen in real world applications. Mainly by the fact that the probability of finding nodes with large degree, in this process, are practically null. We have chosen this model because it is a standard structure used to evaluate methods of network reconstruction in several publications such as Ha and Sun (2014), Schäfer and Strimmer (2005), and Cai et al. (2016), and others. Also, it will be useful to compare the performances of the selected methods in this work with other methods in similar situations.
5.2.2 Watts-Strogatz model

The Watts-Strogatz model, introduced by Watts and Strogatz (1998), based on the idea of modeling social interactions and other networks in real world applications. It’s name comes from the Watts-Strogatz phenomena, or popularly known as six degrees of separation (Travers and Milgram, 1967). The process behind the Watts-Strogatz network is to begin with a lattice with connected neighbors in to a certain degree, and increase the amount of disorder by “rewiring” the edges with probability $p_{SW}$, and reconnecting it to other edges in the network with equal probability. For this method, in the network simulation, we used the well-known R\(^1\) package Igraph (Csardi and Nepusz, 2006), used mainly for graph applications. The Watts-Strogatz has interesting properties (Watts and Strogatz, 1998), but still, doesn’t form as much nodes with a lot of connections as we see in real world networks. Nevertheless we applied the methods to this random graph in order to evaluate their performances under this structure.

5.2.3 Barabási-Albert model

The Barabási-Albert model, introduced by Barabási and Albert (1999), is based in two main premises, the first is that networks expands continuously by addition of new nodes, the second is that these new nodes attach preferentially to nodes that are already well connected. This means that the probability that a new node connects to existing nodes is not uniform, which allows nodes with high degree to emerge, or hubs, compared to 5.2.1 and 5.2.2. The size of the degrees $k$ in a Barabási-Albert model distributed as power law; $P(k) \sim k^{-\gamma}$, where $2 < \gamma < 3$, is a parameter which indicates how long the tail of the power law’s distribution is (Barabási and Albert, 1999). We simulated the Barabási-Albert network using the preferential attachment principle as a random process. For a network with $p$ nodes, we start with two connected nodes in a graph $G_{SF} = (V, E)$, with $V = \{1, 2\}$ and $\{1, 2\} \in E$. We introduce the next node, 3, which connects with probability

$$ p_j = \frac{\deg(j)}{\sum_{l \in V} \deg(l)}, \quad j \in V, $$

\(^1\)https://cran.r-project.org.
with the nodes in $V$, where $\text{deg}(j)$ is the number of connections of node $j$. The process goes on, with $p - 3$ steps, generating a graph $G_{SF}$ with Barabási-Albert network properties. In Algorithm 5 we have a procedure used for generating the adjacency matrix $A_{G_{SF}}$.

**Algorithm 5 Barabási-Albert**

**procedure**

1. Initialize empty matrix $A_{G_{ER}} = \{a_{ij}\}$;
2. Let $V = \{1, 2\}$, $\{1, 2\} \in E$, and $a_{12} = 1$;
3. for $i = 3, \ldots, p$ do:
   1. $p_j = \frac{\text{deg}(j)}{\sum_{k \in V} \text{deg}(k)}$, $j \in V$;
   2. Select $j \in V$ with probability $p_j$;
   3. if $j < i$, $a_{ji} = 1$, else if $i < j$, $a_{ij} = 1$;
   4. $i \in V$, and $\{i, j\} \in E$;
4. $A_{G_{SF}} = A_{G_{SF}} + (A_{G_{SF}})^T$;

**end procedure**

We felt that the Barabási-Albert model would be interesting in our evaluation, because it seems to appear in a number of real-world networks; the world-wide web, in business networks, transportation networks, gene expression (Albert, 2005), and so on. Another reason, is the network’s property in forming hubs. Recently, the emergence of works with the intention to produce network reconstruction methods mainly applied to Scale-free or Barabási-Albert models, such as Peng et al. (2009), and Liu and Ihler (2011). A version in R for Algorithm 5, is found in the Appendix B.2.1.

### 5.3 Simulating the Graph structure

With the adjacency matrices generated by the processes described in 5.2.1, 5.2.2, and 5.2.3, we will transform $A_G$ into a Covariance matrix $\Sigma_{sim}$, from which we will simulate random samples $X_V \sim \mathcal{N} \left(0, \Sigma_{sim}\right)$. The used method is described in Cai et al. (2016), in which it initially constructs a Precision matrix and then invert it to obtain the Covariance matrix. To construct the Precision matrix, first we need to attribute random values to the
non-zero elements of $A_G$, this process gives us the matrix $\Omega_1$,

$$\Omega_1 = \begin{cases} \omega_{ij} = \omega_{ji} = u_{ij} \cdot \delta_{ij}, & \text{if } a_{ij} = 1 \text{ with } a_{ij} \in A_G, \\ 0, & \text{otherwise}, \end{cases} \quad (5.1)$$

where the variable $u_{ij}$, in (5.1) is a uniform random variable defined in the interval (0.4, 0.8).

The variable $\delta_{ij}$ is a discrete uniform distribution with values in \{-1, 1\}, this value will give us equal amounts of positive and negative values in the Precision matrix. The matrix $\Omega_1$ has zeros in it’s diagonals, this means that is singular, to solve this problem we apply the following transformation

$$\Omega_2 = \Omega_1 + (|\lambda_{\text{min}}(\Omega_1)| + 0.05)I_p,$$

where $\lambda_{\text{min}}(\Omega_1)$ is the minimum eigen value of $\Omega_1$, and $I_p$ is a $p \times p$ identity matrix. This guarantees that $\Omega_2$ is invertable. Next, we have $\Omega^{-1}$, that is the matrix formed by the multiplication

$$\Omega^{-1} = \text{diag}(u)\Omega_2^{-1}\text{diag}(u),$$

where $\text{diag}(u)$, which is a diagonal matrix formed by the $p$-dimensional vector, with uniform distribution $u \sim U_p(1, 5)$. Finally the Covariance matrix is obtained trough $\Sigma_{\text{sim}} = \Omega^{-1}$ from which we simulate the multivariate normal distribution $X_V \sim \mathcal{N}_V(0, \Sigma_{\text{sim}})$, with $n < p$. A routine for this transformation is found in the Appendix B.2.2.

### 5.4 ROC curve

The ROC curve or Receiver Operator Characteristic is a methodology used for the comparison of new diagnostic methods (Johnson and Johnson, 2005), usually binary classifiers. For our case, these classifiers are the methods presented in chapter 4, they are binary because it identifies if there is or not an edge between two nodes. The ROC curve describes the tradeoff between the False Positive Fraction $P(FP)$, which is the probability of the method erroneously identifies an edge, also known as false positives (FP), and the True Positive Fraction $P(TP)$, which is the probability of the method correctly identify an edge, or also known as true positives (TP). In the known literature, the fractions $P(TP)$ and $1 - P(FP)$,
are also referred as *sensitivity* and the *specificity* of the method. For our study the \( FP \), and \( TP \), is summarized by the sets

\[
TP = |\{\text{indetify an edge given } \exists \text{ an edge}\}|,
\]

\[
FP = |\{\text{indetify an edge given } \nexists \text{ an edge}\}|.
\]

The probabilities \( P(FP) \) and \( P(TP) \) are given as

\[
P(FP) = \frac{FP}{N}, \quad P(TP) = \frac{TP}{P},
\]

where \( N \) is the *condition negatives*, which is the number of non-edges in the original network we trying to learn, and \( P \) is the *condition positives*, that is the number of edges in the original network. These quantities rewritten in terms of the of the \( TP \) and \( FP \), and from the *true negatives* \( (TN) \), and *false negatives* \( (FN) \) are defined as

\[
TN = |\{\text{indetify a non-edge given } \nexists \text{ an edge}\}|,
\]

\[
FN = |\{\text{indetify a non-edge given } \exists \text{ an edge}\}|,
\]

which gives us \( N = FP + TN \), and \( P = TP + FN \). The ROC curve is defined as a set of points

\[
\text{ROC}_\gamma = (P(FP_\gamma), P(TP_\gamma)), \quad \gamma \in \mathbb{R},
\]

where \( P(FP_\gamma) \) and \( P(TP_\gamma) \), are the fractions (5.2), where \( FP_\gamma \) and \( TP_\gamma \) where generated given a value \( \gamma \). In our case \( \gamma \) will assume different regularization values in the different methods chosen. In the **GLasso** method, \( \gamma = \lambda_L \), such that \( \lambda_L \in \mathbb{R}^+ \), in the **GGMridge** method we have the threshold \( \gamma = \alpha_r \), which is the *p-value* used for selecting the edges in 4.3, (in step 3. of the Algorithm 2 in chapter 4), and in the **LPC** method we have two values for \( \gamma = (\gamma_1, \gamma_2) \), \( \gamma_1 = \alpha_\rho \) which is the *p-value* used in (4.11), for selecting correlation values, and \( \gamma_2 = \alpha_{LPC} \), for the hypothesis test of the Partial correlation estimates in (4.15), with \( \alpha_\rho, \alpha_{LPC} \in [0, 1] \).

A classifier has perfect learning capability if \( P(TP_\gamma) = 1 \), for all \( P(FP_\gamma) \in [0, 1] \) (the
dotted line in Figure 5.2), and has no diagnostic capability if $P(TP_{\gamma}) = P(FP_{\gamma})$ (the dashed line in Figure 5.2), for all $\gamma \in \mathbb{R}$. In this case, the classifier has the same capability of a random choice between edge and non-edge, with equal probability. A common way two compare two different classifiers is to compare the area under the ROC curve

$$A_{ROC} = \int_0^1 ROC_{\gamma}(\xi)d\xi. \quad (5.4)$$

The methods with higher $A_{ROC}$ are the ones preferred. We can approximate the areas of the generated points (5.3), by interpolating the values, and then integrating it using different numerical methods. In Rosner (2015) an approximation using the trapezoidal rule\(^2\) is used to achieve the value of the areas. In the next section we will construct ROC\(_{\gamma}\) curves using the methods from chapter 4, and use this procedure to evaluate and compare the generated areas.

### 5.5 Simulation Results

In this section we will present the results for the different simulation scenarios. We ran 300 simulations, generating 300 of each graph structure presented in 5.2, and took the average of

\(^2\)The rule is seen in the Appendix A.3.1.
the values obtained through the ROC_{\gamma} curve, for \gamma seen in 5.4. We generated the samples from (p = 50, n = 20), (p = 100, n = 50), and (p = 200, n = 50) using the procedure discussed in 5.3. Later we applied the trapezoidal rule (we used the R package known as pracma\textsuperscript{3}), to the mean ROC_{\gamma} for each method and situation, using \gamma with values shown in Table 5.1. The results for the areas is observed in Table 5.5, the mean of the areas by method in Table 5.2, and the mean of the areas by network structure in Table 5.3. In Table 5.4 we have the sparsity (2.5.5) of the graphs. In Table 5.6, we have the ROC_{\gamma} curve of the areas of between the standard deviations. The ROC_{\gamma} curves properly, with only the averages of the curves, are in 5.5.2. The curves with the averages plus the standard deviation are in 5.5.3.

\begin{itemize}
\item Method \quad \gamma
\item GGMridge \quad \{0,001,\ldots,1; by 0.005\}
\item GLasso \quad \{0,0001,\ldots,40; by 0.1\}
\item LPC \quad \{0,0001,\ldots,0.4; by 0.1\} \cup \{0.6,0.7,0.8,0.9,1\}
\end{itemize}

\textbf{Table 5.1:} Used \gamma for ROC_{\gamma} curves.

\begin{tabular}{|c|c|c|c|}
\hline
\textbf{Methods} & \textbf{GGMridge} & \textbf{GLasso} & \textbf{LPC} \\
\hline
\textbf{mean} & 0.8239 & 0.7516 & 0.7633 \\
\hline
\end{tabular}

\textbf{Table 5.2:} Mean ROC_{\gamma} curves by method.

\begin{tabular}{|c|c|c|c|}
\hline
\textbf{Structures} & \textbf{Erdős-Rényi} & \textbf{Watts-Strogatz} & \textbf{Barabási-Albert} \\
\hline
\textbf{mean} & 0.7975 & 0.7878 & 0.7541 \\
\hline
\end{tabular}

\textbf{Table 5.3:} Mean ROC_{\gamma} curves by Structure.

\begin{tabular}{|c|c|c|c|}
\hline
\textbf{variables, samples} & \textbf{Erdős-Rényi} & \textbf{Watts-Strogatz} & \textbf{Barabási-Albert} \\
\hline
$p = 50, n = 20$ & 0.04 & 0.08 & 0.04 \\
$p = 100, n = 50$ & 0.04 & 0.04 & 0.02 \\
$p = 200, n = 50$ & 0.02 & 0.02 & 0.01 \\
\hline
\end{tabular}

\textbf{Table 5.4:} Sparsity of the generated graphs.

\textsuperscript{3}https://cran.r-project.org/web/packages/pracma/index.html
5.5.1 Analysis and Conclusion

The first thing we might notice in observing the plots in 5.5.2 and 5.5.3, is that the ROC$_\gamma$ curve produced by the GLasso method, contrary to the other methods, doesn’t go all the way to the point (1,1). Investigating this fact we conclude that the restriction coefficient $\lambda_L$, when it is in a certain small value ($\lambda_L \approx 0.0001$), doesn’t generate the estimation that gives us a complete graph (2.5.2), it stops in a structure and stays in that particular result, no matter how close we approximate $\lambda_L$ to zero. Although this might present some problems in generating the ROC$_\gamma$ area, the results presented by the interpolation might in fact inflate the results for the GLasso, so we chosen to continue our analysis in this way.

In the related literature, this problem also appears when the GLasso is evaluated as ROC$_\gamma$ curve in Ha and Sun (2014) and (Peng et al., 2009). Some use other measures related to $FP$, $TP$, $TN$, and $FN$, as seen in 5.4, like the Accuracy and False discovery rate as in (Wang et al., 2012). Other fact we can observe clearly is that GGMridge method gives us
the biggest area under the curve in all cases and different scenarios. This is confirmed by the
data in the Table 5.5. One might think that this method is the best of the three, but with
further analysis we discovered that the form we used to generate the data for the ROC$_\gamma$
might be contributing for this fact. We have generated curves by varying the value of $\alpha_r$ in
other steps of the algorithm, the curves presented we generated before the false discovery
test Benjamini and Hochberg (1995) was applied, and those with different results where
generated by the test. The results of this change tend to produce lower values for the ROC$_\gamma$
area from those found, and increases greatly the standard deviation of the method. Besides
this fact, in overall, we can see that the GGMridge method has the highest value for the
area of ROC$_\gamma$ curve in any situation.

The values of GLasso and LPC are very competitive in all scenarios, in the Barabási-
Albert the GLasso tends to be higher than the LPC, but, from the plots and values we can
see that as the number of values increases the LPC method is approaching the GLasso,
and even is bigger in the case of $(p = 200, n = 50)$. In the Watts-Strogatz network the
LPC method performs better than the GLasso in all cases. And on the Erdős-Rényi the
methods are very close in their performance. One fact that it is worth noticing is that in
Figure 5.4, for $(p = 100, n = 50)$, the LPC produced a curve in which there’s a a drop in
the method’s performance between sensitivity with values in $(0.4, 0.7)$, and 1-specificity with
values in $(0.005, 0.2)$, this might due to the variation of the $\gamma$ used for the method.

In Table 5.4 the sparsity of the graphs that where generated, we can see that in most
cases the density is below 0.05, the only case this doesn’t happen is in the Watts-Strogatz
case for $p = 50$, this happens due the way the network was generated.

From the facts presented we can leave some ideas, and questions for further research;

- Perform the simulations for $p > 200$, in the order of thousands, and evaluate the
  results.

- Improve the LPC method performance.

- What is the right way to produce a ROC$_\gamma$ curve for the GGMridge method?

- How the change in the $\gamma$ used in the LPC method, changes its ROC$_\gamma$ curve.
5.5.2 ROC curves - average values

Figure 5.3: ROC curves generated using Erdős-Rényi model with.

Figure 5.4: ROC curves generated using Watts-Strogatz model with.

Figure 5.5: ROC curves generated using Barabási-Albert model with.
5.5.3 ROC curves - average and standard deviation

Figure 5.6: ROC curves generated using Erdős-Rényi model, with standard deviation.

Figure 5.7: ROC curve generated using Watts-Strogatz model, with standard deviation.

Figure 5.8: ROC curve generated using Barabási-Albert model, with standard deviation.
Chapter 6
Application to Data

6.1 Introduction

Continuing with the chosen methods\(^1\) evaluation, we applied them to a real-world scenario where the reconstructed network results are compared. Although in nature, we can observe different types of complex networks (Barabási and Albert, 1999), most of the methods have the gene co-expression application in mind (Ha and Sun, 2014) (Schafer and Strimmer, 2005), so it is a natural choice for the evaluation.

Briefly, a gene is a segment of DNA (deoxyribonucleic acid), which is a double strand with a sequence of nucleotides or bases (there are four different bases; adenine (A), thymine (T), cytosine (C), and guanine (G), that only can form certain pairs\(^2\)). These bases have all the instructions needed to direct the workings of all living systems. The gene encodes the instruction for the synthesis and regulation of proteins, that determine shape, structure and function of the cell. Although genes are essential in life management, they only comprise 2% of human genome, while the rest are non-code regions, with other functions (Schaefer, 2006). The human genome, contain estimated 19,000 – 20,000 genes (Ezkurdia et al., 2014). It is assumed that the gene expression tend to portray the cell’s function. The aim of gene co-expression analysis is often to identify genes whose expression differs, in cells from a healthy tissue, from those with abnormal behaviors, such as tumorous tissues. By examining this differences, researchers aim to understand the disease development, and to identify what

---

\(^1\)Presented in chapter 4.
\(^2\)The human genome, consists of some 3 billion base pairs (Schaefer, 2006).
genes are involved in such conditions.

The gene co-expression data, in most cases, is high dimensional. As we seen in chapters 3 and 4, dealing with this conditions, from a statistical point a view, is very challenging. We chosen a data set with this characteristics, and performed a simple analysis, in which we counted the nodes, which are the genes, and edges, which are the intensity of the expression between the genes, that where identified in common by the methods. The data used is the gene expression analysis provided by Zhai et al. (2007), which contains the expression of 21 tumor tissue from human Cervical Cancer samples (which is strongly associated with certain types of human papillomaviruses (HPV)), and 10 normal samples. The original data is obtained from the Gene Expression Omnibus\(^3\) repository, with access number GSE7803, using the Affymetrix U133A platform (the format the microarray where generated).

### 6.2 The Data and Methods

From the Zhai et al. (2007) data base, we have the expression of 25,387 genes, from these genes we selected 1,268 based on the work developed by Mine et al. (2013), which identified in the data, the differently expressed genes or DEG, that are genes with expression that differs from normal samples. The study used not only the Zhai et al. (2007) data base, but also 3 other data bases; Biewenga et al. (2008), Pyeon et al. (2007), and Scotto et al. (2008), all of them related to human Cervical Cancer. The study in Mine et al. (2013), aims to understand how these DEG are connected with the HPV virus capacity to induce genomic changes, that not only promote the appearance of tumors, but tumor growth.

In our study, we where only interested in recovering the network structure based on the available data. We didn’t perform any data augmentation, by inserting data from the different studies used in Mine et al. (2013), because it would be necessary further statistical investigation of the data, and choosing the right method to bind them in a single structure. We also didn’t perform any biological analysis, since this kind of analysis require a deeper knowledge biological systems.

In the next section we will compare the results obtained by the methods and present the

---

\(^3\)https://www.ncbi.nlm.nih.gov/geo/
networks that were generated.

6.3 Results of the Application

We applied the data to the methods, with no treatment or normalization\(^4\), besides those that came from the experiment to reduce variance (Zhai et al., 2007). Also we reconstructed the networks not using all the 1,268 genes identified by Mine et al. (2013), but we chosen the restriction coefficients of the methods that, returned approximately half this quantity, with tree times the number of edges\(^5\). The coefficients used and the results generated by the methods is seen in Table 6.1, and in Figure 6.1.

| Method    | Coefficients | number of nodes | number of edges |
|-----------|--------------|-----------------|-----------------|
| GGMridge  | \(\lambda_r = 1, \alpha_r = 0.01\) | 558             | 1876            |
| GLasso    | \(\lambda_L = 0.6\) | 630             | 1942            |
| LPC       | \(\alpha_\beta = 0.1, \alpha_{LPC} = 0.02\) | 658             | 1774            |

Table 6.1: Results of the methods: number of nodes and edges.

In Figure 6.1 we identified the number of genes and edges that the methods identified in common and presented the results as Venn diagrams, with the number of edges in Figure 6.1a and the number of edges in Figure 6.1b. The network images where done using the

\(^4\)The GGMridge method performs the normalization of the data in all cases.

\(^5\)This proportion was achieved informally by conversations with the bioinformatics community.
Cytoscape tool Shannon et al. (2003). The network reconstructed using only one method seen in 6.4.1. The LPC method in Figure (6.2), GGMridge in Figure (6.3), and GLasso in Figure 6.4. In Figure 6.5 we have the network reconstructed using LPC and GLasso, in Figure 6.6 we have LPC and GGMridge, and in Figure 6.7 we have GLasso and the GGMridge. Finally in Figure 6.8 we have the three methods represented together, with the nodes organized in a way as the Venn diagram in Figure 6.1a with the edges preserved. In Table 6.2 we have the minimum and the maximum degrees of the nodes by method. This degrees relates to the size of the nodes of the networks in the Figures. The smaller the size of the node the smaller the degree, and bigger the node also bigger the degree.

| Method                  | min | max |
|-------------------------|-----|-----|
| GGMridge                | 1   | 35  |
| GLasso                  | 1   | 188 |
| LPC                     | 1   | 14  |
| GLasso and LPC          | 3   | 165 |
| GGMridge and LPC        | 1   | 35  |
| GGMridge and GLasso     | 2   | 188 |
| GGMridge, GLasso, and LPC | 3   | 177 |

Table 6.2: Maximum and minimum degrees of the methods.

6.4 Analysis and Conclusion

The Venn diagrams in 6.1a and 6.1b gives us interesting information. We can see that the LPC and GGMridge, excluding the GLasso, share a large number of nodes, 343 in total. If we include the nodes also shared with the GLasso we have 670 common identifications, which is 81.9% of the jointly identified nodes. In Figure 6.6 this is easily seen. On the other hand, the edges are not proportionally shared as the nodes. Only 8.1% of the edges where identified in common. In the GLasso and LPC method this fact becomes more evident, although they share a fair amount of nodes, 54, the number of edges shared is significantly small, only 14. We can conclude that although the methods identify, jointly, a good an amount nodes, the edges tend to form links inside each method. We can observe this fact in other perspective trough the degrees.
From the results in Table 6.2 we can see that the degrees of the methods translates to the Figures 6.2, 6.3, and 6.4. The **LPC** method has nodes with maximum degree 14, this means that the network is more scattered, with few nodes with high degree. In contrast, the network reconstructed by the **GLasso** has a maximum degree of 188, which means that it has identified a hub, this is seen in the network, where the the nodes are concentrated around a group of nodes. The **GGMridge** method has maximum degree of 34, but as we see in Table 6.2, the method identified the node with largest degree from the **Glasso** method, this means that a node with a low degree in one method has high degree in the other. Other fact can observe is that the **Glasso** network is very distinctive from the other methods, since it formed more hubs, this result can also be seen in Figure 6.8. But as we seen, this hubs might not real hubs, since the other methods didn’t recognize these hubs.

From the results obtained we can leave some questions, and future topics for research.

- Examine with more details the recovered structure from the methods, test if it has some topological form.
- Introduce Biological analysis to the results.
- Normalize the data, perform more tests before applying to the method.
- Make the same analysis with the compiled data, from the bases (Biewenga et al., 2008), (Pyeon et al., 2007), and (Scotto et al., 2008). Try to find a way to fit this data without loss of information.
- Try the method to the entire genes, and analyze the results.
- Verify if the **Glasso** hubs, are real hubs.
6.4.1 Reconstructed Networks

Figure 6.2: Reconstructed Network using LPC method. The nodes represent the genes (color in green), the links are the grey edges. The size of the nodes represent the degrees. The smallest node represent 1 degree the largest node represent 14 degrees. In total there are 658 identified nodes and 1774 edges. We can observe that we don have major hubs and the edges seems evenly distributed, that is we have a large number of nodes with same degree.
Figure 6.3: Reconstructed Network using **GGMridge** method. The nodes generated are in green and the edges are in gray. The smallest size node represent 1 degree the largest node represent 35 degrees. In total there are 558 identified nodes and 1876 edges. The generated network appears to have more hubs and less nodes with same degree. We can see some concentration in some nodes.
Figure 6.4: Reconstructed Network using GLasso method. The nodes generated are in red and the edges are in gray. The smallest size node represent 1 degree the largest node represent 188 degrees. In total there are 630 identified nodes and 1942 edges. From the generated network we can observe the presence of hubs with large number of degrees and a large number of nodes with one degree.
Figure 6.5: Reconstructed Network with the GLasso and LPC method. The 277 red circles are the nodes identified only by the GLasso method, the 377 green parallelograms are the nodes identified by the LPC method, and the 281 yellow diamonds are the nodes identified by both the GLasso and LPC method. The largest node, identified by the GLasso, has 165 degrees, the smallest 3.
Figure 6.6: Reconstructed Network GGMridge and LPC method. The 60 gray inverted triangles are the nodes identified only by the GGMridge method, the 88 green parallelograms are the nodes identified by the LPC method, and the 570 blue hexagrams are the nodes identified by both the GGMridge and LPC method. The largest node, identified by both GLasso and LPC has 35 degrees the smallest 1. We can observe that the methods have identified most of the nodes in common.
Figure 6.7: *Reconstructed Network GGMridge and GLasso method.* The 298 red circles are the nodes identified only by the GLasso method, the 370 gray inverted triangles are the nodes identified by the LPC method, and the 260 orange triangles are the nodes identified by both the GLasso and LPC method. The largest node, identified by the GLasso has 188 degrees the smallest 3.
Figure 6.8: Comparison of the three methods; GGMridge, GLasso, and LPC. The 244 red circles are the nodes identified only by the GLasso method, the 27 gray inverted triangles are the nodes identified by the GGMridge method, and the 34 green parallelogram identified by the LPC method. The 33 orange triangles are the nodes identified by both the GLasso and GGMridge method. The 343 blue hexagons are the nodes identified by both the GGMridge and LPC method. The 54 yellow diamonds are the nodes identified by both the GLasso and LPC method, and The 227 light blue squares are the nodes identified by the three methods. We can see that most of the hubs are associated with the GLasso method, and most of the edges are between the LPC and GGMridge methods and the three methods.
Appendix A

Support Material

The subjects presented in this section are ordered as they appear in the text.

A.1 Chapter 1

A.1.1 Vector Norm

A vector norm is defined for a vector \( x^T = (x_1, x_2, \ldots, x_n) \) with \( x \in \mathbb{R}^n \), as a function,

\[
\| \cdot \| : \mathbb{R}^n \rightarrow \mathbb{R}.
\]

The \( \ell_1 \) norm is defined as

\[
\|x\|_1 = \sum_{i=1}^{n} |x_i|,
\]

where \( |\cdot| \) is the module of \( x_i \).

The \( \ell_2 \) norm is defined as,

\[
\|x\|_2 = \sqrt{\sum_{i=1}^{n} |x_i|^2}.
\]

A.1.2 Ordinary Least Squares

In a linear regression, we want to find the model that best fits our samples in the form

\[
y_i = \alpha + x_i \beta + \epsilon_i,
\]

in which \( i = 1, \ldots, n \), is the number of samples and \( \epsilon_i \) is the error term with distribution

\[
\epsilon_i \sim N(0, \sigma^2), \quad \text{with } \sigma^2 \in \mathbb{R}^+ \text{ and } E(\epsilon_i, \epsilon_j) = 0.
\]
We can generalize the problem for \( p \) variables introducing matrix notation.

\[
\begin{cases}
y_1 = \beta_0 + \beta_1 x_{11} + \epsilon_1, \\
\vdots & \vdots \\
y_n = \beta_0 + \beta_1 x_{1n} + \epsilon_n,
\end{cases}
\]

Which give us the form \( Y = X\beta + \epsilon \) where

\[
Y = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}, \quad X = \begin{pmatrix} 1 & x_{11} \\ \vdots & \vdots \\ 11 & x_{1n} \end{pmatrix}, \quad \beta = \begin{pmatrix} \beta_0 \\ \beta_1 \end{pmatrix}, \quad \epsilon = \begin{pmatrix} \epsilon_1 \\ \vdots \end{pmatrix}.
\]

That gives us a **minimization** problem

\[
\arg\min_\beta \|Y - X\beta\|_2^2 = \arg\min_{\beta_0, \beta_1} ((Y - X\beta)\top (Y - X\beta))
\]

where

\[
\|x\|_2^2 = \left(\sqrt{x_1^2 + \cdots + x_n^2}\right)^2 = x\top x,
\]

is the \( L^2 \) norm of the vector \( x \), which gives us

\[
\arg\min_\beta ((Y - X\beta)\top (Y - X\beta)) = \arg\min_{\beta_0, \beta_1} \left( \sum_{i=1}^n (y_i - \beta_0 - \beta_1 x_i)^2 \right)
\]

We can rewrite the matrix term as

\[
(Y - X\beta)\top (Y - X\beta) = Y\top Y - 2Y\top X\beta + \beta\top X\top X\beta.
\]

We observe that we have a from of quadratic equation, which can be minimized using the partial derivative over the parameters to find the inflexion point which will be our minimum.

\[
\frac{\partial RSS}{\partial \beta} = -2X\top Y + 2X\top X\hat{\beta} = 0
\]

\[\Rightarrow X\top X\hat{\beta} = X\top Y\]

\[\hat{\beta} = (X\top X)^{-1}X\top Y\]

This process it’s known as **OLS** or **ordinary least squares**.
A.2 Chapter 4

A.2.1 Generalized inverse

If $A$ is a $m \times n$ matrix, then $G$ is a generalized inverse of $A$ if $G$ is an $n \times m$ matrix with

$$AGA = A.$$ 

The Penrose conditions are used to define different generalized inverses for $A \in \mathbb{R}^{n \times m}$, and $G \in \mathbb{R}^{m \times n}$:

1. $AGA = A$;
2. $GAG = G$;
3. $(AG)^T = AG$;
4. $(GA)^T = GA$;

If $G$ satisfies the first condition, then it is a generalized inverse of $A$. If it satisfies the first two conditions, then it is generalized inverse of $A$. If it satisfies all four conditions, then $G$ is a Moore-Penrose pseudoinverse (Penrose, 1955) of $A$.

A.3 Chapter 5

A.3.1 Trapezoidal rule

The trapezoidal rule approximates the region under a function $f(x)$ as a trapezoid, resulting in its area. So we have that

$$\int_a^b f(x)dx \approx (b - a) \left[ \frac{f(a) + f(b)}{2} \right].$$

For a Numerical implementation in a non-uniform grid we have

$$\int_a^b f(x)dx \approx \frac{1}{2} \sum_{k=1}^{N} (x_{k+1} - x_k)(f(x_{k+1}) + f(x_k)).$$
Appendix B

Algorithms

The subjects presented in this section are ordered as they appear in the text.

B.1 Chapter 4

B.1.1 Local Partial Correlation

```r
# Require the following packages

library("parallel")
library("foreach")
library("doParallel")

### Functions ###

####

filter.rows <- function(x, rows, fun=sd){
  rows <- min(rows, nrow(x))
  val <- apply(x, 1, fun)
  sel.rows <- order(val, decreasing=TRUE)[1:rows]
  return(x[sel.rows, ])
}

just.filter <- function(x, percent, maxRows){
  res <- filter.rows(filter.prop(x, percent), maxRows)
  return(res)
}

# Get probability values for each correlation matrix.
cor.prob = function(X, dfr = nrow(X) - 2) {
  R = cor(X)
  above = row(R) < col(R)
  r2 = R[above]^2
  Fstat = r2 * dfr / (1 - r2)
  R[above] = 1 - pf(Fstat, 1, dfr) #Probability of the F distribution
  R
}

####

# Function for calculating the LPC method

#if(print.lpc == T)
```
print(paste("alpha \alpha", alpha))

nCores = detectCores() - 1
cl = makeCluster(nCores)
registerDoParallel(cl, cores = detectCores() - 1)

#---------------------------------------------
sampleNames <- toupper(names(data))
sampleNames <- gsub("[^\.:;\-,./\|\\]", "", sampleNames)
#---------------------------------------------

if (nLim != -1) {
  data <- just.filter(data, porc, nLim)
  if(print.lpc == T) print("filtering")
} else {
  if(print.lpc == T) print("NOT filtering")
}

dim(data)

# Transpose da data (pxn) to (nxp)
X = t(data)

# Number of variables.
p = ncol(X)

# Size of the sample.
n = nrow(X)

# Warning for thr case n >= p.
if(p <= n){
  print("nothing to do: n >= p.")
  return(FALSE)
}

# Calculate the sample covariance of the data, can we do this?
covX = cov(X)

# Calculates the sample correlation of the data, can we do this?
corX = cor(X)

# Return the pvalue of the correlation matrix.
pvalcor = cor.prob(X)

# New empty matrix
matRoXY = matrix(0, ncol = p, nrow = p)
colnames(matRoXY) = colnames(X)

# Fill the matrix matRoXY with correlation from X
if (alpha==1) {
  for(i in 1:(p-1)) {
    for(j in (i+1):p) {
      matRoXY[i,j] = corX[i,j]
      matRoXY[j,i] = matRoXY[i,j]
    }
  }
} else {
for(i in 1:(p-1)) {
for(j in (i+1):p) {
    matRoXY[i,j] = ifelse(pvalcor[i,j] < alpha, corX[i,j], 0)
    matRoXY[j,i] = matRoXY[i,j]
}
}
#
# Puts zeros in the rest of the matrix where there is NA.
matRoXY[is.na(matRoXY)] = 0
#
# igraph function (transfom into an adjacency graph?)
g1 = graph.adjacency(matRoXY, mode = ("undirected"), weighted = TRUE)
#
# Get the vertices of a igraph object g1, V()
V(g1)$name = c(seq(1:p))
#
# Accessing edges ...
# Get edges from the object g1, list of the kind a->b.
edges = get.adjlist(g1)
#
# Function unlist transforms a list into a single vector
totEdg = unlist(edges)
#
# Generate a matrix with 99 in all positions, why?
Ap = matrix(99, ncol=ncol(X), nrow=ncol(X))
#
# Number of neighbors is equal to the size of the sample.
# I'm fixing the number of neighbors to the size of the sample.
viz = n
t0 = Sys.time()

## foreach(), similar to lapply and loop function from doParallel
## Apply the function to all the elements of X.
lpcData = foreach(k = 1:ncol(X), packages = c("igraph", "corpcor"), .combine = rbind) %dopar%
{  
    # for (k in 1:ncol(X)) {
    # k = 4

    # Get values from the list "edges"
    # Position where the elements is different from zero.
    vertices = edges[[k]]
    if(print.lpc == T){
      'print(k)
      print("------------")
    }

    # Will pass to the line of the matrix
    if (length(vertices) > 0) {
      # (sum(is.na(vertices))!=0)
      for (j in 1:length(vertices)) {
        # print(j)
        # j = 4
        if(vertices[j] > k & matRoXY[k, vertices[j]] !=0) {
          il = as.character(k)
          jl = as.character(vertices[j])

          # Get the neighbors of a node. The function receives the graph,
#the nodes that we want to find the neighbors and how far in the graph we want to search.
#(graph , order , nodes)
vizlist = graph.neighborhood(g1, 1, nodes = c(i1, j1), mode=c("all"))

#Sort the neighbors from the largest to the smallest
unique()
#GET THE COMMON NEIGHBORS
viz = sort(as.integer(unique(c(V(vizlist [[1]]))$name, V(vizlist [[2]]))$name)), decreasing=TRUE))

# Sort the neighbors from the largest to the smallest
unique()

# Downsizing the number of neighbors in case we have we still have more neighbors than observations, but aren't we using the number of the sample size as neighbors.
if(length(viz) >= nViz) {
  # Get the biggest correlation from the two selected nodes.
cor.maiores.i = sort(abs(matRoXY[viz, c(k)])[−k]), decreasing = TRUE)[1:(nViz/2)]
cor.maiores.j = sort(abs(matRoXY[viz, c(vertieces[j])][−(verteices[j])]), decreasing = TRUE)[1:(nViz/2)]

  # Remove the zeros
  cor.maiores.i = cor.maiores.i[cor.maiores.i > 0]
cor.maiores.j = cor.maiores.j[cor.maiores.j > 0]

  # Getting the n/2 biggest correlations.
cor.maiores = sort(abs(unique(c(cor.maiores.j, cor.maiores.i))), decreasing = TRUE)[1:(nViz/2)]
cor.maiores.i = cor.maiores.i[cor.maiores.i %in% cor.maiores]
cor.maiores.j = cor.maiores.j[cor.maiores.j %in% cor.maiores]

  majI = cor.maiores.i[! is.na(cor.maiores.i)]
majJ = cor.maiores.j[! is.na(cor.maiores.j)]

  #— valorem absoluto — zero — erro?
posicao.i = sapply(majI, function(valmajI) {
    which(abs(matRoXY[,c(k)]) == valmajI, useNames = TRUE))
posicao.i = unlist(posicao.i)
posicao.j = sapply(majJ, function(valmajJ) {
    which(abs(matRoXY[, c(verteices[j])]) == valmajJ, useNames = TRUE))
posicao.j = unlist(posicao.j)

  if((length(posicao.i)==0) & (length(posicao.j)==0)) next
  if(length(posicao.j)==0){
    viz = sort(c(k, as.numeric(verteices[j]) , posicao.i))
  } else {
    if (length(posicao.i)==0) viz <- sort(c(k, as.numeric(verteices[j]) , posicao.j))
    else viz = unique(sort(c(k, as.numeric(verteices[j]) , c(posicao.i, posicao.j))))
  }
}
lcl = length(viz)

# Neighbor covariance matrix
Csub = covX[viz, viz]
# \texttt{dim(Csub)}

# \texttt{cor2pcor} computes the pairwise partial correlation.
# Coefficients from either a correlation or a covariance matrix.

# Calculates the Partial correlation from a covariance matrix or corr matrix
\texttt{corr.inv = cor2pcor(Csub)}
\texttt{rownames(corr.inv) = viz}
\texttt{colnames(corr.inv) = viz}

# Performs the z-transformation test
\texttt{p.value = 1-pnorm(abs(sqrt(n-(lcl-2)-3)*}
\texttt{log((1+corr.inv[i1,j1])/(1-corr.inv[i1,j1]))/2))}

# right superior \texttt{lpcData = partial correlation}
# left inferior \texttt{lpcData = p-value}
\texttt{Ap[k,vertices[j]] = p.value}
\texttt{Ap[vertices[j],k] = corr.inv[i1,j1]}

# Calculates the tome that the loop has taken.
\texttt{ti <- Sys.time()}
\texttt{if(print.lpc == T)}
\texttt{print(paste(as.numeric(round(ti-t0,2)), units = "secs")/60,}
\texttt{"min.\$p\$", p, "tempo/p=" , round(as.numeric(ti-t0, units = "secs")/p,2), "s/node")}
\texttt{colnames(lpcData) = rownames(data)}
\texttt{rownames(lpcData) = rownames(data)}

\texttt{if (save) {}
\texttt{writeLpcFile(lpcData, porc, pGenes=p))}
\texttt{}}
# Stop the parallelization routine.
\texttt{stopCluster(cl)}

\section*{B.2 Chapter 5}

\subsection*{B.2.1 Scale-free simulation}

# Number of variables
\texttt{p = 10}

# Size of hubs
\texttt{hubs = 2}

# Generate vector with 1 to p.
\texttt{vec = 1:p}

# Transform vec to characters.
\texttt{charac = as.character(vec)}
# Generate combination of nodes and edges.
comb = combn(rev(charac), 2)
col$pair = ifelse(as.numeric(col$ed1) < as.numeric(col$ed2), paste(col$ed1, col$ed2), paste(col$ed2, col$ed1))
col$edges = numeric((length(vec) * (length(vec) - 1)) / 2)

# Generate Starting hubs.
vec.sel = 1:hubs
temp2 = data.frame(ed1 = temp[1,], ed2 = temp[2,])
col.sel$pair.sel = ifelse(as.numeric(col.sel$ed1) < as.numeric(col.sel$ed2), paste(col.sel$ed1, col.sel$ed2), paste(col.sel$ed2, col.sel$ed1))
col[col$pair %in% col.sel$pair.sel, "edges"] = 1

A = matrix(0, ncol = length(vec), nrow = length(vec))
A[upper.tri(A)] = col$edges

# Generate table edge.counts, that counts the number of edges and control
# the preferential attachment.
edges.count = data.frame(vec)
edges.count$E = numeric(length(vec))
edges.count$edges = numeric(length(vec))
edges.count[edges.count$vec %in% col.sel$ed1, "E"] = 1
edges.count[edges.count$vec %in% col.sel$ed2, "E"] = 1
edges.count[edges.count$vec %in% col.sel$ed2, "E"] = 1
edges.count[edges.count$vec %in% col.sel$ed1, "E"] = 1

edges.count[edges.count$E != 0, "edges"] =
edges.count[edges.count$E != 0, "vec"] =
edges.count$prob = edges.count$E / sum(edges.count$E)
prob = edges.count$prob

# Difference between sets
cmpl.vec = setdiff(vec, vec.sel)
cmpl.vec.temp = compl.vec

# Random Process that generates the preferential attachment.
for (i in compl.vec[1]:(compl.vec[1] + length(compl.vec) - 1)){
    new.node = i
    samp.node = sample(edges.count$edges, 1, prob = edges.count$prob)
    new.pair = ifelse(new.node < samp.node, paste(new.node, samp.node), paste(samp.node, new.node))
    col[col$pair %in% new.pair, "edges"] = 1
    edges.count[edges.count$vec %in% new.node, "E"] = 1
    edges.count[edges.count$vec %in% new.node, "edges"] = new.node
    edges.count[edges.count$vec %in% samp.node, "E"] =
    edges.count[edges.count$vec %in% samp.node, "edges"] =
    edges.count$prob = edges.count$E / sum(edges.count$E)
}

# Resulting adjacency matrix.
A = matrix(0, ncol = length(vec), nrow = length(vec))
\[ A[\text{upper.tri}(A)] = \text{col}\$\text{edges} \]
\[ A = A + t(A) \]

### B.2.2 Matrix transformation

```r
# Sample uniformly select between (0.4, 0.8)
cov = runif(sum(A)/2, epsilon1, epsilon2)

# Sample \{-1,1\} with equal probability.
sign = sample(c(1,-1), size = sum(A)/2, replace = T, prob = c(prob1, prob.menos1))
cov = sign* cov
Omega = A
Omega[which(((upper.tri(A)) * (A!=0)) ! = 0)] = cov
Omega[lower.tri(A)] = 0
Omega = Omega + t(Omega)
Omega = Omega + (0.05 + abs(min(eigen(Omega)$values))) * diag(1, ncol(A), ncol(A))

# Matriz diagonal pxp com uniformes
D = matrix(0, nrow = p, ncol = p)
diag(D) = runif(p, 1, 5)

# Convert precision matrix.
Sigma = D%*%solve(Omega)%*%D
```
Bibliography

Abu-Mostafa et al. (2012) Yaser S Abu-Mostafa, Malik Magdon-Ismail and Hsuan-Tien Lin. Learning from data, volume 4. AMLBook New York, NY, USA:.

Albert (2005) Reka Albert. Scale-free networks in cell biology. Journal of cell science, 118 (21):4947–4957.

Banerjee et al. (2008) Onureena Banerjee, Laurent El Ghaoui and Alexandre d’Aspremont. Model selection through sparse maximum likelihood estimation for multivariate gaussian or binary data. Journal of Machine Learning Research, 9(Mar):485–516.

Barabási and Albert (1999) Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. science, 286(5439):509–512.

Benjamini and Hochberg (1995) Yoav Benjamini and Yosef Hochberg. Controlling the false discovery rate: a practical and powerful approach to multiple testing. Journal of the royal statistical society. Series B (Methodological), pages 289–300.

Bien et al. (2011) Jacob Bien, Robert J Tibshirani et al. Sparse estimation of a covariance matrix. Biometrika, 98(4):807.

Biewenga et al. (2008) Petra Biewenga, Marrije R Buist, Perry D Moerland, Emiel Ver Loren van Themaat, Antoine HC van Kampen, Fiebo JW ten Kate and Frank Baas. Gene expression in early stage cervical cancer. Gynecologic oncology, 108(3):520–526.

Bollobás (1998) Béla Bollobás. Random graphs. Em Modern Graph Theory, pages 215–252. Springer.

Bollobás (2013) Béla Bollobás. Modern graph theory, volume 184. Springer Science & Business Media.

Brémaud (2013) Pierre Brémaud. Markov chains: Gibbs fields, Monte Carlo simulation, and queues, volume 31. Springer Science & Business Media.
Cai et al. (2016) T Tony Cai, Weidong Liu, Harrison H Zhou et al. Estimating sparse precision matrix: Optimal rates of convergence and adaptive estimation. The Annals of Statistics, 44(2):455–488. From page 38, 40

Castelo and Roverato (2006) Robert Castelo and Alberto Roverato. A robust procedure for gaussian graphical model search from microarray data with p larger than n. Journal of Machine Learning Research, 7(Dec):2621–2650. From page 1

Csardi and Nepusz (2006) Gabor Csardi and Tamas Nepusz. The igraph software package for complex network research. InterJournal, Complex Systems:1695. URL http://igraph.org. From page 39

Dempster (1969) AP Dempster. Elements of continuous multivariate analysis. From page 6

Efron (2004) Bradley Efron. Large-scale simultaneous hypothesis testing: The choice of a null hypothesis. Journal of the American Statistical Association, 99:465. From page 31

Erdos and Rényi (1960) Paul Erdos and Alfréd Rényi. On the evolution of random graphs. Publ. Math. Inst. Hung. Acad. Sci, 5(1):17–60. From page 2, 38

Ezkurdia et al. (2014) Iakes Ezkurdia, David Juan, Jose Manuel Rodriguez, Adam Frankish, Mark Diekhans, Jennifer Harrow, Jesus Vazquez, Alfonso Valencia and Michael L Tress. Multiple evidence strands suggest that there may be as few as 19 000 human protein-coding genes. Human molecular genetics, 23(22):5866–5878. From page 49

Friedman et al. (2001) Jerome Friedman, Trevor Hastie and Robert Tibshirani. The elements of statistical learning, volume 1. Springer series in statistics Springer, Berlin. From page 14, 27, 28

Friedman et al. (2007) Jerome Friedman, Trevor Hastie, Holger Höfling, Robert Tibshirani et al. Pathwise coordinate optimization. The Annals of Applied Statistics, 1(2):302–332. From page 28

Friedman et al. (2008) Jerome Friedman, Trevor Hastie and Robert Tibshirani. Sparse inverse covariance estimation with the graphical lasso. Biostatistics, 9(3):432–441. From page 2, 28

Ganguly (2014) Apratim Ganguly. Structure Learning in Locally Constant Gaussian Graphical Models. Phd Thesis, UNIVERSITY OF CALIFORNIA DAVIS. From page 18

Giraud (2014) Christophe Giraud. Introduction to high-dimensional statistics, volume 138. CRC Press. From page 1, 11, 17

Ha and Sun (2014) Min Jin Ha and Wei Sun. Partial correlation matrix estimation using ridge penalty followed by thresholding and re-estimation. Biometrics, 70(3):762–770. From page 2, 29, 30, 31, 32, 38, 45, 49
Hastie et al. (2015) Trevor Hastie, Robert Tibshirani and Martin Wainwright. Statistical learning with sparsity: the lasso and generalizations. CRC Press. From page 12, 19, 27

Hoerl and Kennard (1970) Arthur E Hoerl and Robert W Kennard. Ridge regression: Biased estimation for nonorthogonal problems. Technometrics, 12(1):55–67. From page 14, 30

Johnson and Johnson (2005) Valen E Johnson and Timothy D Johnson. Bayesian analysis of roc data. Handbook of Statistics, 25:821–833. From page 41

Kendall et al. (1946) Maurice George Kendall et al. The advanced theory of statistics. The advanced theory of statistics., (2nd Ed). From page 34

Lauritzen (1996) Steffen L Lauritzen. Graphical models, volume 17. Clarendon Press. From page 1, 2, 8, 9, 22

Liu and Ihler (2011) Qiang Liu and Alexander T Ihler. Learning scale free networks by reweighted L1 regularization. Em AISTATS, pages 40–48. From page 40

Mardia et al. (1980) Kantilal V Mardia, John T Kent and John M Bibby. Multivariate analysis (probability and mathematical statistics), 1980. From page 5

Mazumder and Hastie (2012) Rahul Mazumder and Trevor Hastie. The graphical lasso: New insights and alternatives. Electronic journal of statistics, 6:2125. From page 2

Meinshausen and Bühlmann (2006) Nicolai Meinshausen and Peter Bühlmann. High-dimensional graphs and variable selection with the lasso. The annals of statistics, pages 1436–1462. From page 2, 27

Meinshausen and Bühlmann (2010) Nicolai Meinshausen and Peter Bühlmann. Stability selection. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 72(4):417–473. From page 1

Mine et al. (2013) Karina L Mine, Natalia Shulzhenko, Anatoly Yambartsev, Mark Rochman, Gerdine FO Sanson, Malin Lando, Sudhir Varma, Jeff Skinner, Natalia Volfovsky, Tao Deng et al. Gene network reconstruction reveals cell cycle and antiviral genes as major drivers of cervical cancer. Nature communications, 4:1806. From page 3, 50, 51

Pearl (2009) Judea Pearl. Causality. Cambridge university press. From page 7, 9, 19

Peng et al. (2009) Jie Peng, Pei Wang, Nengfeng Zhou and Ji Zhu. Partial correlation estimation by joint sparse regression models. Journal of the American Statistical Association, 104(486):735–746. From page 25, 40, 45
Penrose (1955) Roger Penrose. A generalized inverse for matrices. *Em Mathematical proceedings of the Cambridge philosophical society*, volume 51, pages 406–413. Cambridge Univ Press. From page 63

Pyeon et al. (2007) Dohun Pyeon, Michael A Newton, Paul F Lambert, Johan A Den Boon, Srikumar Sengupta, Carmen J Marsit, Craig D Woodworth, Joseph P Connor, Thomas H Haugen, Elaine M Smith et al. Fundamental differences in cell cycle deregulation in human papillomavirus–positive and human papillomavirus–negative head/neck and cervical cancers. *Cancer research*, 67(10):4605–4619. From page 50, 53

Rosner (2015) Bernard Rosner. *Fundamentals of biostatistics*. Nelson Education. From page 43

Schaefer (2006) Juliane Schaefer. *Small-Sample Analysis and Inference of Networked Dependency Structures from Complex Genomic Data*. Phd Thesis, lmu. From page 49

Schäfer and Strimmer (2005) Juliane Schäfer and Korbinian Strimmer. An empirical bayes approach to inferring large-scale gene association networks. *Bioinformatics*, 21(6):754–764. From page 1, 2, 5, 25, 30, 38

Schafer and Strimmer (2005) Juliane Schafer and Korbinian Strimmer. Learning large-scale graphical gaussian models from genomic data. *Science of Complex Networks from Biology to the Internet and WWW*, 776:263–276. From page 32, 49

Schervish (2012) Mark J Schervish. *Theory of statistics*. Springer Science & Business Media. From page 7

Scotto et al. (2008) Luigi Scotto, Gopeshwar Narayan, Subhadra V Nandula, Hugo Arias-Pulido, Shivakumar Subramaniyam, Achim Schneider, Andreas M Kaufmann, Jason D Wright, Bhavana Pothuri, Mahesh Mansukhani et al. Identification of copy number gain and overexpressed genes on chromosome arm 20q by an integrative genomic approach in cervical cancer: potential role in progression. *Genes, Chromosomes and Cancer*, 47(9):755–765. From page 50, 53

Shannon et al. (2003) Paul Shannon, Andrew Markiel, Owen Ozier, Nitin S Baliga, Jonathan T Wang, Daniel Ramage, Nada Amin, Benno Schwikowski and Trey Ideker. Cytoscape: a software environment for integrated models of biomolecular interaction networks. *Genome research*, 13(11):2498–2504. From page 52

Takahashi (2008) Daniel Yasumas Takahashi. *Medidas de fluxo de informação com aplicação em neurociência*. Phd Thesis, Universidade de Sao Paulo. From page 10

Thomas et al. (2012) Lina D Thomas, Victor Fossaluza, Anatoly Yambartsev, Julio Michael Stern, Marcelo De Souza Laurentto, Adriano Polpo and Marcio Alves Di-
niz. Building complex networks through classical and bayesian statistics-a comparison. Em *AIP Conference Proceedings*, volume 1490, pages 323–331. AIP. From page 2, 32

**Thomas (2012)** Lina Dornelas Thomas. *Construção de redes usando estatística clássica e Bayesiana-uma comparação*. Phd Thesis, Universidade de São Paulo. From page 10, 23

**Tibshirani (1996)** Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288. From page 14

**Travers and Milgram (1967)** Jeffrey Travers and Stanley Milgram. The small world problem. *Psychology Today*, 1:61–67. From page 39

**Wang et al. (2012)** Hao Wang *et al.* Bayesian graphical lasso models and efficient posterior computation. *Bayesian Analysis*, 7(4):867–886. From page 45

**Warton (2008)** David I Warton. Penalized normal likelihood and ridge regularization of correlation and covariance matrices. *Journal of the American Statistical Association*, 103 (481):340–349. From page 30

**Wasserman (2013)** Larry Wasserman. *All of statistics: a concise course in statistical inference*. Springer Science & Business Media. From page 19

**Watts and Strogatz (1998)** Duncan J Watts and Steven H Strogatz. Collective dynamics of small world networks. *Nature*, 393(6684):440–442. From page 2, 13, 39

**Whittaker (2009)** Joe Whittaker. *Graphical models in applied multivariate statistics*. Wiley Publishing. From page 2

**Witten et al. (2011)** Daniela M Witten, Jerome H Friedman and Noah Simon. New insights and faster computations for the graphical lasso. *Journal of Computational and Graphical Statistics*, 20(4):892–900. From page 2

**Zhai et al. (2007)** Yali Zhai, Rork Kuick, Bin Nan, Ichiro Ota, Stephen J Weiss, Cornelia L Trimble, Eric R Fearon and Kathleen R Cho. Gene expression analysis of preinvasive and invasive cervical squamous cell carcinomas identifies hoxc10 as a key mediator of invasion. *Cancer research*, 67(21):10163–10172. From page 2, 50, 51