Graphical models for estimating dynamic networks

Antonino Abbruzzo

Publisher: Wöhrmann print service
www.wps.nl

Rijksuniversiteit Groningen

Graphical models for estimating dynamic networks

Proefschrift

ter verkrijging van het doctoraat in de
Wiskunde en Natuurwetenschappen
aan de Rijksuniversiteit Groningen
op gezag van de
Rector Magnificus, dr. E. Sterken,
in het openbaar te verdedigen op
dinsdag 10 april 2012
om 11.00 uur

door

Antonino Abbruzzo

geboren op 5 december 1978
te Ribera, Italië

Promotores:	Prof. Dr. E.C. Wit
	Prof. Dr. A.M. Mineo
Beoordelingscommissie:	Prof. Dr. A.C.D. van Enter
	Prof. Dr. E.R. van den Heuvel
	Prof. Dr. G. Kauermann

ISBN 978-90-367-5430-9
e-ISBN 978-90-367-5433-0

Contents

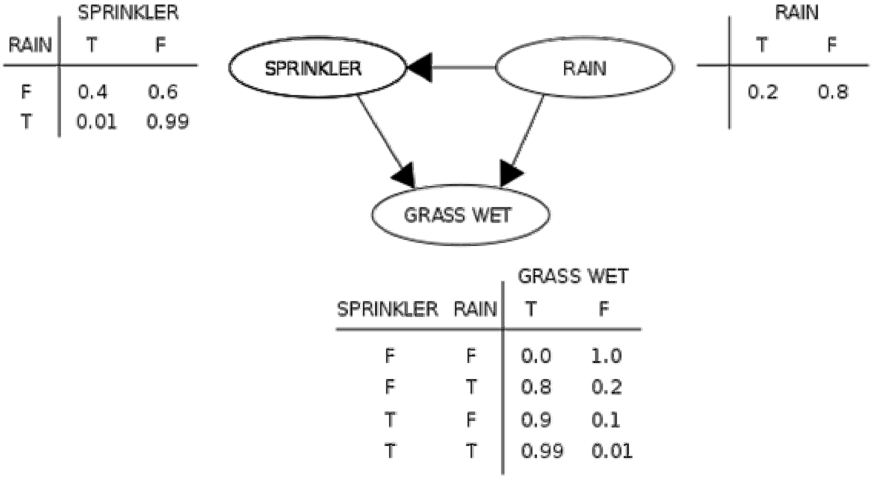
Chapter 1

Introduction

1.1 Introduction

Graphical models are powerful tools for analysing relationships between a large number of random variables. Their fundamental importance and universal applicability are due to a number of factors. Graphs can be used to represent conditional dependence between random variables. Moreover, their structure is also modular which means that complex graphs can be built up from many simple graphs or components. This modular structure is useful to describe complex systems in a simple way. Graphical models have been applied in many fields of science such as molecular biology, social science and marketing data analysis. We can divide graphical models in directed graphical models, e.g. Bayesian networks (Jensen, 1996; Neapolitan, 2004) and undirected graphical models, e.g. Gaussian graphical models (Whittaker, 1990; Lauritzen, 1996).

Directed graphical models can be represented by directed acyclic graphs, where a set of nodes is linked by directed links. For example consider two events that cause grass to be wet: either the sprinkler is on or it is raining. Also, suppose that the rain has a direct effect on the use of the sprinkler (namely that when it rains, the sprinkler is usually not turned on). Then the situation can be modelled with a Bayesian network (see Figure 1.1). All three variables have two possible values, T (which stands for true) and F (which stands for false). The model can answer questions like ”What is the probability that it is raining, given the grass is wet?” by



	SPRINKLER	
RAIN	T	F
F	0.4	0.6
T	0.01	0.99

RAIN	
T	F
0.2	0.8

		GRASS WET	
SPRINKLER	RAIN	T	F
F	F	0.0	1.0
F	T	0.8	0.2
T	F	0.9	0.1
T	T	0.99	0.01

Figure 1.1. A simple example of a Bayesian network. Nodes represent a real situation where grass, rain and a sprinkler are considered. Grass is wet if it is raining or if the sprinkler is on. Rain has a directed effect on the use of the sprinkler. This situation is modelled by directed links in the graph. Tables nearby the nodes represent conditional probabilities.

using the conditional probability formula and summing over all nuisance variables:

$$\begin{aligned}
\mathrm{P}(R=T \mid G=T) \quad = \quad & \frac{\mathrm{P}(G=T,R=T)}{\mathrm{P}(G=T)} = \\
& \frac{\sum_{S\in\{T,F\}}\mathrm{P}(G=T,S,R=T)}{\sum_{S,R\in\{T,F\}}\mathrm{P}(G=T,S,R)} = \\
& \frac{(0.99\times 0.01\times 0.2)_{TTT}+(0.8\times 0.99\times 0.2)_{TFT}}{0.00198_{TTT}+0.288_{TTF}+0.1584_{TFT}+0_{TFF}} \approx \\
& 0.35
\end{aligned}$$

In equation (1.1) the joint probability function is used to calculate each iteration of the summation function. If, on the other hand, we wish to answer an interventional question: ”What is the likelihood that it would rain, given that we wet the grass?” the answer would be governed by the post-intervention joint distribution function

$$\mathrm{P}(S,R|do(G=T))=P(S|R)P(R)$$

obtained by removing the factor $\mathrm{P}(G|S,R)$ from the pre-intervention distribution. As expected, the likelihood of rain is unaffected by the action:

$$\mathrm{P}(R|do(G=T)) = P(R).$$

If, moreover, we wish to predict the impact of turning the sprinkler on, we have

$$P(R,G|do(S=T)) = P(R)P(G|R,S=T),$$

with the term $P(S=T|R)$ removed, showing that the action has an effect on the grass but not on the rain. See Pearl (2000) for more details on the *do* operator.

Another example of using directed graphs for modelling non-symmetric relationship is about representing Bayesian regression models.With non-symmetric relationships we mean that one variable, usually called “dependent variable”, can be predicted from several independent variables. The program `WinBugs`, which is developed to solve numerically Bayesian models where posterior distributions cannot be written in closed form, can be used to build Bayesian models. In Bayesian modelling, parameters are assumed to be random variables and they can represented by nodes in the graph. Relations among the parameters can be represented by directed links. Figure 1.2 shows a directed graph. The regression example shows how easy it is to represent a model with directed graphs. The great advantage is that one can generate a code which represents a particular Bayesian model. Figure 1.3 shows the Winbugs code for the graph in Figure 1.2.

However, directed graphs cannot represent all possible situations. For example, suppose that examination marks are collected for a class of N students, and the examination marks consist of Mechanics, Vectors, Statistics, Analysis and Algebra. Assume that Mechanics, Vectors and Algebra are depended and Statistics, Mathematics and Algebra are depended, but Mechanics and Vectors are conditional independent of Mathematics and Statistics given Algebra. This situation can be represented by an undirected graphical model (see Figure 1.4). Bayesian networks cannot be used to represent such situation since cycles are presents in the graph and no direction can be assumed.

In this thesis, we want to address the problem of estimating dynamic graphs for time-course genetic experiments, i.e. the study of relations among thousands genes. We start by considering a small example to get used to the notation. Suppose that three expression level genes over two time points have been collected. Assume that the genes are conditional depended within the time points but only self-self dependences are present over time points. Self-self relations represent conditional dependences between the same gene over two different temporal points. A possible representation of such graph is given in Figure 1.5.

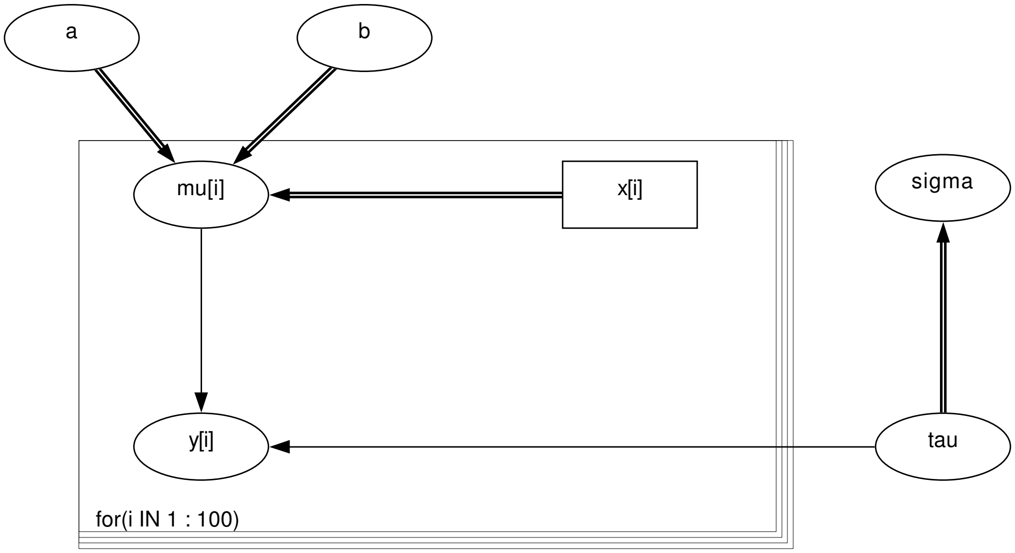


Figure 1.2. Bayesian regression model represented with a directed acyclic graph. The $n = 100$ observations are assumed to follow a Gaussian distribution with parameters *mu* and *tau*. The parameter *tau* represents the precision which is the inverse of the variance parameter *sigma*. The parameter $mu_i = a + b * x_i$ represents the conditional expectation of y_i given x_i, where x_i are independent variables. The hyper-parameters are assumed to be Gaussian with mean zero and precision small. The precision for a and b are assumed to be small such that uniform information is assumed on those hyper-parameters. Variables inside the rectangle are vector variables of length 100.

Gene expression is a temporal process. Different proteins are required (and synthesized) for different functions and under different conditions. Even under stable conditions, due to the degradation of proteins, mRNA is transcribed continuously and new proteins are generated. This process is highly regulated. One of the most important ways in which the cell regulates gene expression is by using feedback loops. Some of the proteins are transcription factors (TFs). These proteins regulate the expression of other genes (and possibly, their own expression) by either initiating or repressing transcription. For example, when cells are faced with a new condition, such as starvation (Natarajan *et al.*, 2001), infection (Nau *et al.*, 2002) or stress (Gasch *et al.*, 2000), they react by activating a new expres-

```
model;
{
  for( i in 1 : 100 ) {
    y[i] ~ dnorm(mu[i],tau)
  }
  for( i in 1 : 100 ) {
    mu[i] <- alpha + beta * x[i]
  }
  tau ~ dgamma(0.001,0.001)
  a ~ dnorm( 0.0,1.0E-6)
  b ~ dnorm( 0.0,1.0E-6)
  sigma <- 1 / tau
}
```

Figure 1.3. WinBugs code for the Bayesian multivariate linear regression model represented in Figure 1.2.

sion program. In many cases, the expression program starts by activating a few TFs, which in turn activate many other genes that act in response to the new condition. Taking a snapshot of the expression profile following a new condition can reveal some of the genes that are specifically expressed under the new condition. However, in order to determine the complete set of genes that are expressed under these conditions, and to determine the interaction between these genes, it is necessary to measure gene expression levels form time-course microarray experiments. Then, the pathway and networks that were changed from a state (time t) to another one (time $t+1$) can be inferred. Note that dynamic genetic networks are usually sparse, i.e. most of the genes are not connected in the network.

Bayesian networks (BNs) have been used to estimate the structure between multiple interaction quantities such as expression levels of different genes (Friedman *et al.*, 2000). However, they suffer a major limitation, i.e no cycle can be estimated. Moreover, BNs perform poorly on sparse graphs as shown by Husmeier (2003). The first limitation of BNs can be overcome. Dynamic Bayesian networks have been proposed to estimate directed graphs with cycles (Murphy, 2002;

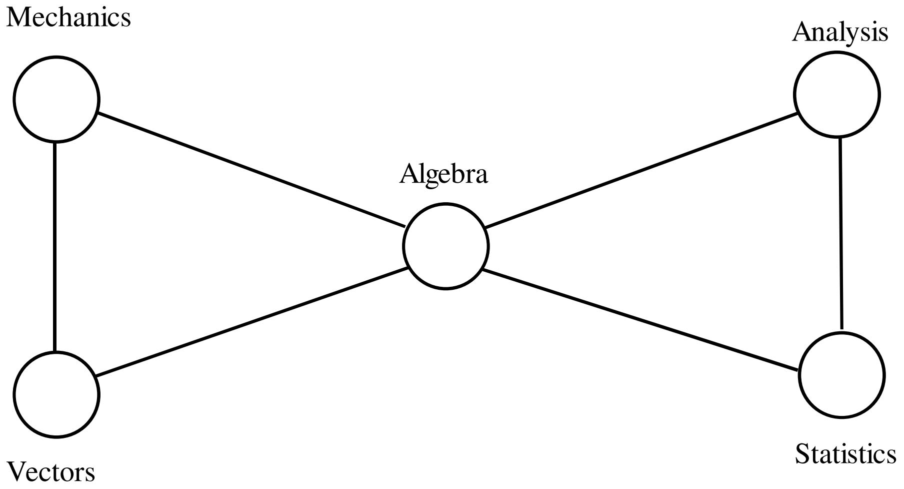


Figure 1.4. Undirected graph representing a model of conditional independence among five subjects: Mechanics, Vectors, Statistics, Analysis and Algebra.

Ghahramani, 1998; Perrin *et al.*, 2003). However, no direction can be assumed for relationships within time points. For example, gene_1 and gene_2 are dependent but no direction can be establish unless an experiment on those specific genes has been carried out under "similar" conditions.

Gaussian graphical models can be adapted to estimate dynamic graphs. The main advantage for GGMs is that the precision matrix, i.e. the inverse of the covariance matrix, represents the conditional independence. When Gaussian graphical models are applied in order to study large networks, they typically fail because the number of variables is much greater than the number of observations. Recently, penalized Gaussian graphical models have been proposed to estimate static networks in high-dimensional studies because of their statistical properties and computational tractability. The main idea in this thesis is to extend penalized Gaussian graphical models to estimate structured dynamic graphs.

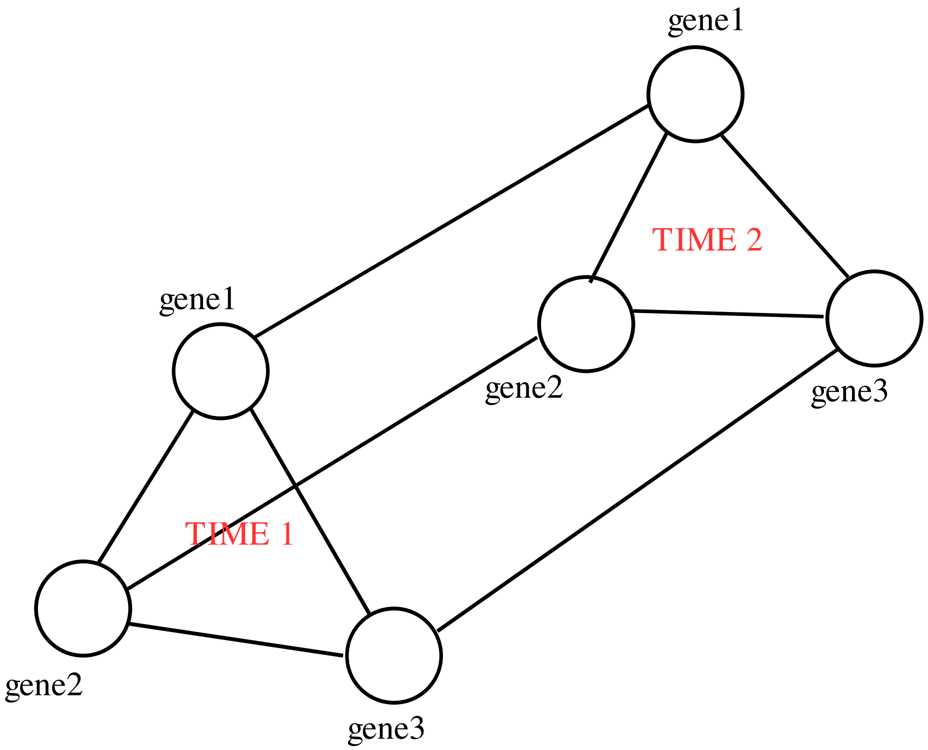


Figure 1.5. Example of a dynamic network. Three gene expression levels are collected over two time points. Genes are conditional depended within each time point but only self-self dependences are present between time point 1 and time point 2. Self-self relations are relation between the same genes at different time points.

1.2 Our work and contribution

The contribution of this thesis involves several aspects of graphical modelling. In this section motivations, we give methods and results for each of our contribution.

1. **Gaussian graphical models for structured dynamic networks (SGL).**

   Motivation. We have argued in Section 1.1 that gene expression is a temporal process, and that understanding biological dynamic networks is important to address specific experimentations issues in biology. Indeed, this is a difficult task since the number of components involved in the system is very large. One solution to such problem is using penalized Gaussian graphical models. In particular the ℓ_1-norm penalty can be imposed on the precision matrix to induce sparsity. However, the literature has focused on static graphical models which lack specific temporal constraints.

Methods. We propose a new model for estimating structured dynamic networks. We propose to use penalized Gaussian graphical models to estimate structured dynamic networks. We use the idea of coloured graphs to build several models as well as the idea of ℓ_1-norm penalization to induce sparsity on the precision matrix. The model structures can consist of specific time dynamics, i.e. known presence or absence of links and/or block equality constraints on the parameters. The main advantages of this model are that the number of parameters to be estimated is reduced, and the sparsity of the estimated graph can be inferred by increasing or decreasing a tuning parameter. This model is described in Chapter 3

Results. We implement specific sets of linear constraints in `matlab`. The code is given in Appendix A. We show that computations can be done effectively by taking advantage of an efficient solver `LogDetPPA` developed in convex optimization. We use synthetic data to illustrate that our approach is more suitable for estimating dynamic networks than graphical lasso. An analysis on a real dataset has also been provided.

2. **Copula Gaussian graphical models for dynamic networks (copulaSGL).**

   Motivation. The assumption of Gaussianity for gene expression levels can be too restrictive. One way to overcome such assumption is to use a Gaussian copula. Gaussian copulas have the main advantage of separating conditional independences from marginal structure of the data. Moreover, copulas are powerful tools to deal with complex multivariate problems, such as in graphical models for mixed random variables (Lauritzen, 1996). However, the literature has focused on static Gaussian copula.

   Methods. We propose structured non-canonical Gaussian copula graphical models for estimating dynamic networks. This is the topic of Chapter 4.

   Results. We implement specific sets of linear constraints in `matlab`. The code will be soon available in the following website:
   `http://www.math.rug.nl/stat/`.

3. **Additional model-based GGMs and GGMs with latent variables: slow changes in Dynamic networks (GL$_\Delta$).**

   Motivation. Interactions among genes tend to be different across different time-points for several reasons, such as changing in experimental conditions. However, the literature has focused on static networks, i.e. networks that are assumed to be the same through time. Static networks can only reveal overall structures but it is worth to estimate significant changes in networks. For example, if we consider a common graph, we could observe which are the interactions that appear and/or disappear from a given time point to the next one. We can also assume that changes in graphs from one time point to the next one are few because systems in biology tend to be stable.

Methods. We introduce a new model for estimating slowly changing in dynamic gene-regulatory networks which is suitable for high-dimensional dataset, e.g. time-course genetic data. Our method is based on i) penalized likelihood with ℓ_1-norm, ii) penalized differences between conditional independence elements across time points, iii) heuristic search strategy to find optimal smoothing parameters. Two tuning parameters are necessary. The first smoothing parameter regulates the sparsity of the network, and the second one penalizes changing between network at time point t and $t+1$. We called this new model sparse graphical lasso for estimating slowly changing of dynamic networks (GL_Δ). This is the topic of Section 5.1.

Results. We implement a set of linear constraints necessary to estimate sparse graphs and penalized changing in dynamic networks. These constraints are not in the linear form. For this reason, we introduce slake variables to transform our problem in a standard convex optimization problem subject to linear constraints. LogDetPPA, which is an efficient convex optimization solver, can be used to solve the optimization problem. We show that GL_Δ performs well in a simulation study. Finally, we apply the proposed model to a time-course genetic dataset "T-cell". The code will be soon available in the following website:
`http://www.math.rug.nl/stat/`.

4. **Additional model-based GGMs and GGMs with latent variables: latent variable Gaussian graphical models for dynamic networks (latentGL).**

   Motivation. The problem of estimating dynamic networks becomes even more challenging when latent or hidden variables are involved in larger systems, i.e. when some components of a network cannot be observed. This is often the case in biological systems, but it is also a feature of many real-world causal systems. State-space models have been proposed in order to study dynamic networks with latent variables. However, expectation maximization combined with Kalman filters for estimating dynamic networks with latent variables can be very unstable. We propose a penalized Gaussian graphical models to estimate dynamic networks with latent structures.

   Method. We extend the idea of Chandrasekaran *et al.* (2010) for estimating dynamic networks. This the topic of Subsection 5.3.2.

Results. We implement specific sets of linear constraints in `matlab`. The code will be soon available in the following website:
`http://www.math.rug.nl/stat/`.

We note that methodologies, as proposed in this thesis, should be considered high level microarray data analysis. The biological and computational issues that are addressed when analyzing gene expression data in general, and time-course expression data in particular can be presented using four analysis levels: experimental design, data analysis, pattern recognition and networks. A review paper on each of these four analysis levels is given by Bar-Joseph (2004). A good book on statistical techniques for analyzing microarray data was proposed by Wit and McClure (2004).

1.3 Outline of the thesis

The rest of this thesis is organized as follows. In Chapter 2 we give an overview of Graphical models, biological networks, and we introduce the regularization framework for graphical models. Chapter 3 describes model-based networks to learn undirected graphs with dynamic structures (SGL). In Chapter 4, we take advantage of copula theory to extend the use of the SGL model (copulaSGL). In the first part of Chapter 5 additional model-based Gaussian graphical models are described (GL_{Δ}). In the second part of Chapter 5 methodologies to learn undirected and directed graphs with latent variables are described. In particular, we have considered latent Gaussian graphical models and state-space models to recover networks of observed variables which take into account effects of latent variables. Motivating examples are used through this thesis to show the proposed methodologies. In particular, a real time-course microarray dataset “human T-cell data'' is studied.

Chapter 2

Graphical models for biological networks

Networks are important models to address specific questions in genomics. Dynamic gene-regulatory networks are complex objects since the number of potential components involved in the system is very large. For example, one important direction in systems biology is to discover gene regulatory networks from microarray data based on the observed mRNA levels of thousands of genes under various conditions. We shall show that one solution to such problem is the use of penalized Gaussian graphical models, which have been extensively used to estimate sparse static graphs.

In this chapter we describe basic principles of gene transcription and genetic regulatory networks. Then, a review on graphical models and several summary measures of networks are given. Finally penalized graphical models with a ℓ_1 norm penalty or the so called "graphical lasso" are described. For more details on graphical models see for example Lauritzen (1996) or Whittaker (1990). Bishop (1995) gives a good introduction on Bayesian networks and undirected graphical models. More information on gene transcription can be found in Barabási and Oltvai (2004). Newman (2010) considers biological networks with respect to their scales. Metabolic networks, protein-protein interaction networks and genetic regulatory networks are examples of microscopic scale networks while ecological networks are examples of macroscopic scale networks. We focus exclusively in the former.

2.1 Introduction to gene transcription

Proteins are essential parts of the cell that determine the cell's structure and execute nearly all its functions. The production of proteins is carried out by the *ribosomes*, but the information needed for their production is encoded in *genes* which are the segments of *DNA*. DNA contains valuable genetic information, that must be preserved. Transient *RNA* is used to carry the message from DNA to ribosomes. In all living cells, the flow of genetic information is thought to go in this way

$$\mathit{DNA} \rightarrow \mathit{RNA} \rightarrow \text{PROTEIN}.$$

This fundamental principle in biology is called the *central dogma* of molecular biology. The step from DNA to RNA consists of copying the information from genes to RNA and it is called *transcription*. The step from RNA to protein consists of decoding the information from RNA by ribosomes and it is called *translation*. Together these two processes are known as *gene expression*.

The process of transcription is carried out by special enzymes called *RNA polymerases* (RNAp). RNA polymerase binds to the promoter and then opens up the double helix of the DNA sequence immediately in front of it and slides down the gene producing the RNA molecule. The *promoter* is a region of DNA that facilitates the transaction of a particular gene and contains a sequence of nucleotides indicating the starting point for RNA synthesis. Chain elongation continues until enzyme encounters a second signal in DNA, the *terminator*, where RNAp halts and releases both the DNA chain and the newly made RNA chain. RNA which encodes information for production of a certain protein is called *messenger RNA*(mRNA).

However, to do all of this RNAp needs help from special proteins called *transcription factors*. Transcription factors bind at the promoter and form a transcription initiation complex. They position the RNAp correctly on the promoter and aid in pulling apart the two strands of DNA to allow transcription to begin and to allow RNAp to leave promoter as transcription begins. After RNAp is released from the complex it starts making RNA. Once transcription has begun, most of the transcription factors are released from the DNA so that they are available to initiate another round of transcription with a new RNAp molecule. The synthesis of the next RNA usually starts before the first RNA is completed. There maybe several polymerases moving along a single stretch of DNA and RNAs.

The main goal of gene transcription is to produce mRNA which will be translated by ribosomes to make proteins. Each mRNA can be translated several times by ribosome in order to make proteins. This is done until mRNA reaches the end of its life-span.

2.1.1 Transcription factors and gene regulatory networks

In the previous section we have mentioned special proteins called transcription factors that help RNAp to initiate transcription. These transcription factors (TFs) are called *general transcription factors* because the same ones are required for the initiation of transcription of various genes in a wide variety of different organisms. There is also a group of transcription factors that bind to special regions of a target gene called the *regulatory region* in order to regulate its transcription. This latter group of transcription factors can act either as *transcriptional activators* that stimulate transcription of the target gene or as *transcriptional repressors* that inhibit its transcription. *Enhancers* are the binding sites of the activators and *silencers* are the binding sites of the repressors. In the regulatory regions there are multiple binding sites where several TFs are able to bind. Therefore, regulation of transcription involves various interactions between several TFs. Non-cooperativity occurs when TFs are independently bound to a regulatory region. Cooperativity occurs when the affinity of the TF to a binding site depends on the amount of TFs already bound. The cooperative binding can be either positive or negative, indicating that the affinity is either increased or decreased by the binding of other TFs. Competition is also possible when two different TFs bind to one and the same site.

In case of multiple transcription factors, the regulation of gene transcription can be modulated in many ways. In some cases all the transcription factors need to bind in order for transcription to occur, and in other cases only one of them is enough. Transcription factors can form protein complexes, which serve to activate or inhibit one or more complex members. Also, some of the transcription factors are not active until they are switched "on" by addition of the phosphate group to them. This process is called *phosphorylation*. This is an important regulatory mechanism since transcription can be regulated by turning a TF "on" or "off".

Proteins that perform regulatory functions to direct the expression of a gene are in turn produced by other genes or even by itself. This gives rise to a gene regulatory network (GRN), which consists of a set of DNA, RNA, proteins and other small molecules, and is structured by mutual regulatory interactions between these components.

The network of gene regulation can be very complex, where one regulatory protein controls genes that produce other regulators that in turn control other genes. As we already mentioned, protein can either activate or repress its own synthesis, which further increases the complexity of the GRN.

Gene regulatory network models are a logical way to describe phenomena observed with transcription profiling, such as is done with the popular microarray technology. GRN models can be represented as directed or undirected graphs, where nodes are the elements of the networks DNA, RNA, proteins etc. The directed or undirected edges from one node to another represent the corresponding interaction, for example, activation, repression or translation. Being able to create gene regulatory networks from experimental data and to use them to think about their dynamics will contribute to increase our understanding of cellular functions.

2.2 Graphical models

Graphical models blend probability theory and graph theory together. They are powerful tools for analysing relationships between a large number of random variables. Their fundamental importance and universal applicability is due to a number of factors. Not only can graphs be used to represent conditional independence between random variables, but their structure is also modular so that complex networks can be described and handled by careful inspection of simple components.

In this thesis we define a *graph* as a couple $G = (V, E)$ where $V = (1, \dots, p)$ is a finite set of vertices and $E \subseteq V \times V$ is a subset of ordered pairs of distinct vertices (i, j). An edge is undirected if $(i, j) \in E$ implies $(j, i) \in E$. An undirected graph is a graph with undirected edges only. We denote the set of neighbours of a node i with $ne(i)$ that is the set of $j \in V$ such that $(i, j) \in E$ and $(j, i) \in E$. A graph is called a directed graph if it contains directed edges. An edge is directed from vertex i to j if it is possible that $(i, j) \in E$ and $(j, i) \notin E$. The set of nodes $j \in V$ such that $(i, j) \in E$ is called the set of parents and denoted by $pa(i)$. In other words, the set of parents is the set of directed links going from i to j. In graph theory i is said to be a parent of j and j is said to be a child of i.

Let $\mathbf{Y} = (Y_1, \dots, Y_p)'$ be a set of random variables, where a vertex i, $(i = 1, \dots, p)$ of the graph correspond to a random variable Y_i.

Definition 2.2.1. *A graphical model for* $\mathbf{Y}$ *is a set of probability distributions* $\mathbb{P}$ *for* $\mathbf{Y}$*, that satisfies the pairwise conditional restriction on G, but are otherwise arbitrary (Whittaker, 1990).*

A graphical model is a representation of a joint probability distribution $\mathbb{P}$ in terms of a graph and a corresponding set of functions f defined with respect to that graph. The graph encodes a set of conditional independence relations between the underlying random variables, which allow, under appropriate conditions, for the joint distribution to be decomposed in a product form (i.e. to be factorized). This decomposition implies a certain Markov property among the random variables $\mathbf{Y}$.

We can distinguish graphical models according to the type of links represented in the graph in *directed* and *undirected* graphical models.

2.2.1 Undirected graphical models

An *undirected graphical model* is also called a Markov random field. It is defined as a pair $(G, \mathbb{P})$ that specifies a probability density function f for their joint distribution $\mathbb{P}$ in the form

$$\text{(F)} \qquad f(y_1, \dots, y_p) = \frac{1}{z} \prod_{c \in C} \psi_c(\mathbf{y}_c), \tag{2.1}$$

where C is a set of cliques, i.e. complete subsets of V that are maximal, in G, $\psi_c(\mathbf{y}_c)$ is a potential function, which is a positive function of the variables $\{y_i\}_{i\in C}$, and

$$z = \sum_{\mathbf{y}} \prod_{c\in C} \psi_c(\mathbf{y}_c)$$

is a normalization factor. If the factorization (F) is possible, then it implies the global Markov property. A probability distribution $\mathbb{P}$ is said to obey the *global Markov property*, relative to G, if for any triple (A,B,S) of disjoint subsets of V such that S separates A from B in G

$$(\text{G}) \qquad\qquad \mathbf{Y}_A \perp \mathbf{Y}_B|\mathbf{Y}_S.$$

The global Markov property in turn implies the local and pairwise Markov properties. A probability distribution function is said to obey:

(L) the *local Markov property*, relative to G, if for any vertex $i \in V$

$$Y_i \perp \mathbf{Y}_{V\setminus\{cl(i)\}}|\mathbf{Y}_{bd(i)},$$

(P) the *pairwise Markov property*, relative to G, if for any pair (i, j) of non-adjacent vertices

$$Y_i \perp Y_j|\mathbf{Y}_{V\setminus\{i,j\}},$$

The boundary of i is the set of nodes such that $bd(i) = pa(i)\cup ne(i)$, and the closure of i is the set of nodes such that $cl(i) = i\cup bd(i)$. The expression $V\setminus\{i, j\}$ indicates the set of nodes V except nodes i and j. The expression $Y_i \perp Y_j|\mathbf{Y}_{V\setminus\{i,j\}}$ means that the probability distribution function can be factorized as follows:

$$f_{Y_i,Y_j|\mathbf{Y}_{V\setminus\{i,j\}}}(y_i,y_j|\mathbf{y}_{V\setminus\{i,j\}}) = f_{Y_i|\mathbf{Y}_{V\setminus\{i,j\}}}(y_i|\mathbf{y}_{V\setminus\{i,j\}})f_{Y_j|\mathbf{y}_{V\setminus\{i,j\}}}(y_j|\mathbf{y}_{V\setminus\{i,j\}}).$$

It can be shown that $(F) \Rightarrow (G) \Rightarrow (L) \Rightarrow (P)$ (Lauritzen, 1996). Moreover, Hammersley and Clifford's theorem states that:

Theorem 2.2.1 (Hammersley and Clifford)**.** *A probability distribution* $\mathbb{P}$ *with positive and continuous density* f *with respect to a product measure* μ *satisfies the pairwise Markov property with respect to an undirected graph* G *if and only if it factorizes according to* G*.*

This theorem gives the necessary and sufficient condition for $(P) \Leftrightarrow (F)$, and under this condition we have that all Markov properties are equivalent:

$$(F) \Leftrightarrow (G) \Leftrightarrow (L) \Leftrightarrow (P).$$

Undirected graphical models are useful when random variables can be analysed symmetrically. Specific undirected graphical models are distinguished by the choice of the undirected graph G and the potential functions ψ_c.

Gaussian graphical models.

A graphical model $(G,\mathbb{P})$ where $\mathbb{P}$ is a multivariate normal distribution $N(\boldsymbol{\mu},\boldsymbol{\Sigma})$ with mean $\boldsymbol{\mu}$ and variance covariance matrix $\boldsymbol{\Sigma}$ is called a Gaussian graphical model or a covariance selection model (Dempster, 1972). Let $\boldsymbol{\Theta}=\boldsymbol{\Sigma}^{-1}$ be the precision or concentration matrix then $\boldsymbol{\Theta}$ contains all conditional (in)dependence information for the Gaussian graphical model. In fact, if $\theta_{ij}=0$ then Y_i is independent of Y_j given the rest, i.e. the pairwise Markov property $Y_i \perp Y_j|\mathbf{Y}_{V\setminus\{i,j\}}$. Moreover, θ_{ij} is proportional to β_{ij}, where $\boldsymbol{\beta}_i$ is a vector of regression coefficients when $Y_i|\mathbf{Y}_{V\setminus\{i\}}$ is considered. To see that all the information on the conditional independence is contained in $\boldsymbol{\Theta}$ we need to show that given the set of $\theta_{ij}=0$ we can factorize the joint normal probability distribution $f(\mathbf{y})$ as a product of functions f which do not jointly depend to y_i and y_j when $\theta_{ij}=0$.

Consider the example in which a random variable $Y=(Y_1,Y_2,Y_3)$ is multivariate normal $N(\boldsymbol{\mu},\boldsymbol{\Theta}^{-1})$ and

$$\boldsymbol{\Theta}=\begin{pmatrix}\theta_{11} & \theta_{12} & 0\\ \theta'_{12} & \theta_{22} & \theta_{23}\\ 0 & \theta'_{23} & \theta_{33}\end{pmatrix}.$$

The multivariate normal distribution function is:

$$f(\mathbf{y})=\mathrm{const}|\boldsymbol{\Sigma}|^{-1/2}\exp\left((\mathbf{y}-\boldsymbol{\mu})'\boldsymbol{\Theta}(\mathbf{y}-\boldsymbol{\mu})\right). \tag{2.2}$$

Let $\mathbf{z}=\mathbf{y}-\boldsymbol{\mu}$, then by expanding the exponent we have:

$$\begin{aligned}\mathbf{z}'\boldsymbol{\Theta}\mathbf{z} &= z_1^2\theta_{11}+z_1z_2\theta_{21}+z_1z_3\theta_{31}+z_2z_1\theta_{12}+z_2^2\theta_{22}+z_2z_3\theta_{32}+\\ &\quad z_3z_1\theta_{13}+z_3z_2\theta_{23}+z_3^2\theta_{33}\\ &= z_1^2\theta_{11}+z_1z_2\theta_{21}+z_2z_1\theta_{12}+z_2^2\theta_{22}+z_2z_3\theta_{32}+z_3z_2\theta_{23}+z_3^2\theta_{33}\end{aligned}$$

Now we can factorize (2.2) into a product of terms which do not jointly involve Y_1 and Y_3, i.e.

$$f(\mathbf{y})=f_1f_{12}f_2f_{23}f_3,$$

which means that $Y_1 \perp Y_3|Y_2$ and shows that this conditional independence information is contained in θ_{13}. This can be generalized straightforward by to an arbitrary normal vector $\mathbf{Y}$.

Here we shall consider the relation between the conditional independence and the regression coefficient. Let's partition $\mathbf{Y}$ as $(\mathbf{Y}_1,Y_2)$ with mean vector $\boldsymbol{\mu}=(\boldsymbol{\mu}_1,\mu_2)$ and variance-covariance matrix partitioned as follow:

$$\boldsymbol{\Sigma}=\begin{pmatrix}\boldsymbol{\Sigma}_{11} & \boldsymbol{\sigma}_{12}\\ \boldsymbol{\sigma}'_{12} & \sigma_{22}\end{pmatrix},\boldsymbol{\Theta}=\begin{pmatrix}\boldsymbol{\Theta}_{11} & \boldsymbol{\theta}_{12}\\ \boldsymbol{\theta}'_{12} & \theta_{22}\end{pmatrix}.$$

From standard theory of Gaussian distribution (Tong, 1990), we can derive the conditional PDF for node 2 given the rest, i.e. $Y_2|\mathbf{Y}_1 = \mathbf{y}_1$ which is normal with mean:

$$\tilde{\mu} = \mu_2 + \boldsymbol{\sigma}_{21}\boldsymbol{\Theta}_{11}(\mathbf{y}_1 - \boldsymbol{\mu}_1),$$

and variance:

$$\tilde{\sigma} = \sigma_{22} - \boldsymbol{\sigma}_{21}\boldsymbol{\Theta}_{11}\boldsymbol{\sigma}_{12},$$

where $\boldsymbol{\sigma}_{21} = \boldsymbol{\sigma}'_{12}$. Since $\boldsymbol{\Sigma\Theta} = \boldsymbol{I}$, by definition of the inverse of a matrix, we can use partitioned inverses to get the following result:

$$\boldsymbol{\theta}_{12} = -\boldsymbol{\Sigma}_{11}^{-1}\boldsymbol{\sigma}_{12}\theta_{22} = -\boldsymbol{\beta}_2\theta_{22} \propto \boldsymbol{\beta}_2, \tag{2.3}$$

where $\boldsymbol{\beta}_2 = \boldsymbol{\Sigma}_{11}^{-1}\boldsymbol{\sigma}_{12}$ is a vector of regression coefficients that determines the conditional independence structures, and $\theta_{22} = (\sigma_{22} - \boldsymbol{\sigma}_{21}\boldsymbol{\Theta}_{11}\boldsymbol{\sigma}_{12})^{-1}$. If we consider a particular element of $\boldsymbol{\beta}_2$ for example the i-th and $\beta_{2i} = 0$, then $\theta_{2i} = 0$.

We have obtained the important results that for Gaussian graphical models:

$$(i,j) \in E \iff \theta_{ij} \neq 0 \iff \beta_{ij} \neq 0,$$

which means that conditional independence for Gaussian graphical models can be detected from the estimated precision matrix and regression coefficients.

Let us consider the problem of estimating $\boldsymbol{\Theta}$ from two different points of view: a likelihood and regression approach. Suppose that $\mathbf{Y}^{(1)}, \dots, \mathbf{Y}^{(n)}$ with $\mathbf{Y}^{(i)} \in \mathbb{R}^p$ are independent and identically distributed as a multivariate normal distribution with mean $\mathbf{0}$ and variance $\boldsymbol{\Sigma}$. The profile likelihood function is:

$$L(\bar{\mathbf{y}}, \boldsymbol{\Sigma}) = \text{const} \cdot \prod_{i=1}^{n} |\boldsymbol{\Sigma}|^{-1/2} \exp\left(-\frac{1}{2}(\mathbf{y}_i - \bar{\mathbf{y}})'\boldsymbol{\Sigma}^{-1}(\mathbf{y}_i - \bar{\mathbf{y}})\right)$$

and,

$$\begin{aligned} L(\bar{\mathbf{y}}, \boldsymbol{\Sigma}) &\propto |\boldsymbol{\Sigma}|^{-n/2} \exp\left(-\frac{1}{2}\sum_{i=1}^{n}(\mathbf{y}_i - \bar{\mathbf{y}})'\boldsymbol{\Sigma}^{-1}(\mathbf{y}_i - \bar{\mathbf{y}})\right) \\ &= |\boldsymbol{\Sigma}|^{-n/2} \exp\left(-\frac{1}{2}\sum_{i=1}^{n}\operatorname{tr}\left((\mathbf{y}_i - \bar{\mathbf{y}})(\mathbf{y}_i - \bar{\mathbf{y}})'\boldsymbol{\Sigma}^{-1}\right)\right) \\ &= |\boldsymbol{\Sigma}|^{-n/2} \exp\left(-\frac{1}{2}\operatorname{tr}\left(\sum_{i=1}^{n}(\mathbf{y}_i - \bar{\mathbf{y}})(\mathbf{y}_i - \bar{\mathbf{y}})'\boldsymbol{\Sigma}^{-1}\right)\right) \\ &= |\boldsymbol{\Sigma}|^{-n/2} \exp\left(-\frac{n}{2}\operatorname{tr}\left(\mathbf{S}\boldsymbol{\Sigma}^{-1}\right)\right) \end{aligned} \tag{2.4}$$

where

$$\mathbf{S} = \sum_{i=1}^{n} (\mathbf{y}_i - \bar{\mathbf{y}})(\mathbf{y}_i - \bar{\mathbf{y}})'/n, \tag{2.5}$$

with $\mathbf{S} \in \mathbb{R}^{p \times p}$. Our main interest is on the precision matrix $\boldsymbol{\Theta}$ so we parametrize (2.4) with respect to $\boldsymbol{\Theta}$ and consider the log-likelihood for mathematical convenience, i.e.:

$$l(\bar{\mathbf{y}}, \boldsymbol{\Theta}) \propto \frac{n}{2} \log|\boldsymbol{\Theta}| - \frac{n}{2} \mathrm{tr}(\mathbf{S}\boldsymbol{\Theta}). \tag{2.6}$$

The score function with respect to $\boldsymbol{\Theta}$ is given by:

$$\frac{\partial l(\boldsymbol{\Theta})}{\partial \boldsymbol{\Theta}} = (\boldsymbol{\Theta}^{-1} - \mathbf{S})C,$$

for some constant C. The maximum likelihood estimator is $\hat{\boldsymbol{\Theta}} = \mathbf{S}^{-1}$, provided that $\mathbf{S}$ is positive definite. On the other hand, from (2.3),

$$\hat{\boldsymbol{\beta}}_2 = s_{22} \mathbf{s}_{12}^{-1},$$

and therefore

$$\hat{\boldsymbol{\theta}}_{12} = -\hat{\boldsymbol{\beta}}_2 \hat{\theta}_{22},$$

where the diagonal elements of $\hat{\boldsymbol{\Theta}}$ can be estimated as follows:

$$\hat{\theta}_{22} = \frac{1}{s_{22} + \mathbf{s}_{21} \hat{\boldsymbol{\beta}}_2}.$$

MLE and regression perspectives are both useful in the remainder of the thesis.

2.2.2 Directed acyclic graphs or Bayesian networks

A *directed graphical model* without any cycle in the graph is also called *Bayesian network* (BN). We say that a probability distribution $\mathbb{P}$ admits a recursive factorization according to a directed acyclic graph G if $\mathbb{P}$ can be factorized as:

$$p(y_1, \ldots, y_p) = \prod_{i=1}^{p} p(y_i | pa(i)), \tag{2.7}$$

where $pa(i)$ is the set of parents for the node i. Conversely, a directed graphical model defines a joint probability density function for $\mathbb{P}$ in the form (2.7). An important implication of this factorization is that every variable Y_i is conditional independent, given its set of parents $pa(y_i)$, of all other variables Y_j that are neither parents nor descendants. This property is known as the Markov condition, and in fact it can be shown that the factorization and this condition hold in an if and only if manner. To see this, consider a moral graph of G which is given by marrying parents and deleting directions of G.

Lemma 2.2.1. *If $\mathbb{P}$ admits a recursive factorization according to the directed, acyclic graph G, it factorize according to the moral graph G^m and obeys therefore the global Markov property to G^m (Lauritzen, 1996).*

Specific directed graphical models are distinguished from each other by (i) the structures of their underlying DAGs and (ii) the form of their conditional density functions $f(y_i|pa(y_i))$.

Linear Gaussian models.

Let us consider linear Gaussian models as a simple example of a BN. Note that several widely used techniques, such as factor analysis, principal component analysis and linear dynamical systems, are examples of linear Gaussian models. Let us consider p random variables where each of these random variables conditionally on the set of parents is assumed to be Gaussian with a mean taken as a linear combination of the states of its parents nodes, i.e.

$$Y_i|\mathbf{x}_i \sim N(\alpha_i + \boldsymbol{\beta}_i'\mathbf{x}_i, \sigma_i^2),$$

where $\mathbf{x}_i = pa(y_i)$, then the logarithm of the joint PDF is given by:

$$\log p(\mathbf{y}) = \sum_{i=1}^{p} \log(p(y_i|\mathbf{x}_i)) = \text{const} - \sum_{i=1}^{p} \frac{1}{2\sigma_i^2}(y_i - \mu_i)^2,$$

where $\mu_i = \alpha_i + \boldsymbol{\beta}_i'\mathbf{x}_i$. This is a quadratic function of the components y_i, $(i = 1, \ldots, p)$, and hence the joint distribution $p(\mathbf{y})$ is multivariate Gaussian. We can estimate $\boldsymbol{\beta}_i$ and represent a BN drawing an arrow from j $(j \in pa(y_i))$ to i if the coefficient $\beta_{ij} \neq 0$. We can readily extend the linear-Gaussian model to the case in which the nodes of the graph represent multivariate Gaussian random variables. In this case, we can write the conditional distribution for node i as:

$$\mathbf{Y}_i|\mathbf{x}_i \sim N(\mathbf{B}_i\mathbf{x}_i + \boldsymbol{\alpha}_i, \boldsymbol{\Sigma}_i) \tag{2.8}$$

where $\mathbf{B}_i$ is a matrix with dimension depending of length of $\mathbf{y}_i$ and $\mathbf{x}_i$. It can be shown that the joint PDF over all variables is multivariate Gaussian.

Note that Bayesian networks do not allow cycles in the graph. This means that we cannot have feed-back effects, i.e. Y_1 effects Y_2 which effects Y_3 which in turn effects Y_1. In fact, there is no suitable joint probability distribution to model this situation (Whittaker, 1990). On the other hand, undirected graphical models do not allow for induced dependence, i.e. Y_1 effects Y_2.

2.2.3 Ising models

Another class of graphical models is the class of Ising models. In this subsection, we give a brief description of the Ising model. Moreover, we show that the number of nodes of the set vertex V, which are called bonds, can be let go to infinity. This means that graphical models can be applied both for finite set of vertex and infinite set of nodes. Moreover, we show in this subsection that a specific choice of the potential function can be the Hamiltonian function. In this case we need to derive a normalization constant which is denoted by Z in order to have a proper probability distribution function. This subsection is only a brief overview of the Ising model. The notion of infinite graph will not be used in this thesis. However, we think that it is important to know that possible generalization of finite graph exit. Note that Ising models are also known as tree models. We kept the terminology adopted in physics to describe the Ising model since it is not difficult to compare with the preview terminology given for the graph theory.

The Ising model was originally proposed in physics to study phase transitions, which occur when a small change in a parameter such as temperature or pressure causes a large-scale qualitative change in the state of a system. One purpose of the Ising model is to explain how short-range interactions between, say, molecules in a crystal give rise to long-range, correlation behaviour, and to predict in some sense the potential for a phase transition. Before describing the Ising model we need to give the notion of a lattice and a partition function. A lattice is a finite set of regularly spaced points in a space with typical examples considering dimension d=1, 2 or 3. In dimension 1 we simply have a string of points on a line, which we can enumerate from 1 to N (N will always denote the number of lattice sites, regardless of dimension). In dimension 2 we shall consider the lattice square and in dimension 3 we shall consider the lattice whose repeating units are cubes. Each line segment between lattice sites is called a bond (or link), and lattice sites are called nearest neighbours if there is a bond connecting them. Here we consider wrap around Ising model, i.e. simply add extra bounds connecting lattice sites on opposite boundary. The wrap-around Ising model has dN bonds connecting the N lattice sites.

In figure 2.1 we give an example of two dimensional lattice graph.

For each lattice site an independent variable σ_i is assigned, $i = 1, \ldots, N$. The variables σ_i take on only two variables, -1 and 1, which are, for example, possible states of the lattice site. A realization of the state for $(\sigma_1, \ldots, \sigma_N)$ is called a configuration, and the total number of possible configuration is given by 2^N. The sum of all the possible configuration is called the potential function and it is given:

$$Z(\beta, E, J, N) = \sum_{\pm 1} exp\left(-\beta H(\boldsymbol{\sigma})\right),$$

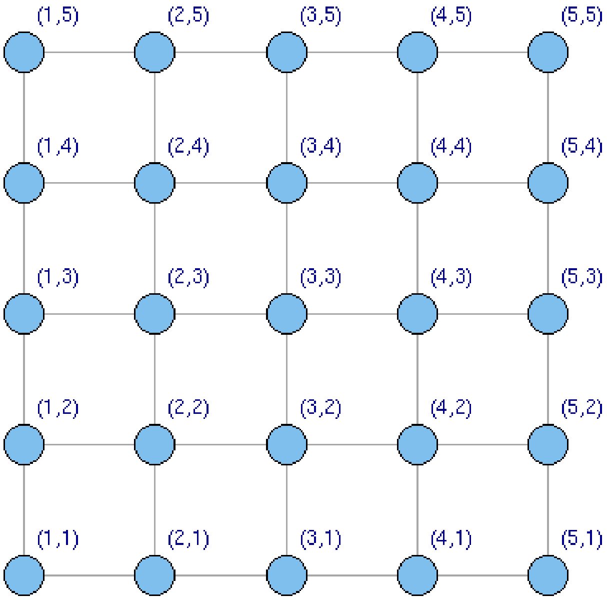


Figure 2.1. Example of a lattice graph in two dimensions.

where $H(\boldsymbol{\sigma})$ is the Hamiltonian function, i.e.

$$H(\boldsymbol{\sigma}) = -\sum_{<i,j>} E\sigma_i\sigma_j - \sum_i J\sigma_i,$$

where E and J are parameters, the first sum is over all pairs of nearest neighbours in the lattice, and the second sum is over all over the lattice sites. The strong assumption is that only nearest-neighbour interactions and interactions of the lattice sites contribute to the system. Then the probability to be in a specific configuration is given by:

$$P(\boldsymbol{\sigma}) = \frac{\exp(\beta H(\boldsymbol{\sigma}))}{Z}.$$

A small value of β tends to flatten out the distribution, making all configurations more or less equally likely, while a large value of β tends to accentuate the probabilities of the lowest energy states.

Many of the quantities one computes from the partition function turn out to depend on the logarithm of Z. This is natural, since Z, being a sum over 2^N configurations, tends to grow exponentially with the size of the lattice. This brings to define the Ising model to be:

$$F = F(\beta, E, J) = \lim_{N\to\infty}\frac{1}{N}\log Z(\beta, E, J, N),$$

where F is the termodynamic limit. The main problem in the Ising model is to find a closed-form, analytic expression for the function F. The fact that we can increase the number of lattice sites brings at the concept of graphs with infinite number of vertices, i.e. $N \to \infty$. Note that the Ising model is defined for Bernoulli random variables but we are mostly interested in continuous ones.

2.3 Regularization framework

Gene regulatory networks (GRNs) are highly complex structures. GRNs are often datasets in which the number of observations is smaller than the number of nodes since it is expensive to collect observations. In fact, it is common to study thousands of variables with only hundreds of observations. An important characteristic of GRNs is that only a small number of interactions exist between genes. Classical Gaussian graphical models cannot be used for inferring the graph of conditional independences. In a GNR, as shown by Buhl (1993), Gaussian graphical models are not identifiable in high-dimensional circumstances that is when the number of variables is much larger than the number of observations. In Gaussian graphical models, maximum likelihood estimator exists with probability one if the number of replicates is at least as large as the number of random variables (Buhl, 1993). One solution is to use penalized Gaussian graphical models. Meinshausen and Bühlmann (2006) showed that gene regulatory networks canbe deal with high-dimensional statistical inference, but sparsity is a necessary assumption.

Informally, a graph with few edges is sparse, and a graph with many edges is dense. More precisely a graph $G = (V, E)$ where $|V|$ is the number of vertices and $|E|$ is the number of links (couple of vertex or nodes) is said to be sparse if $|E| = O(|V|)$. More precisely a graph G is said to be dense if $|E| = O(|V|^2)$. These two definitions are given by Preiss (2008).

Roughly speaking, high dimensional statistical inference is possible, in the sense of leading to reasonable accuracy or asymptotic consistency, if

$$|E|\log(|V|) << n.$$

In other words, accuracy and consistency of the results depend on how one define sparsity.

Much of the methodology and techniques in high-dimensional analysis relies on the idea of penalizing the ℓ_1-norm of the precision matrix $\mathbf{\Theta}$, i.e. $\sum_{i=1}^{p}\sum_{j=1}^{p}|\theta_{i,j}| \leq \rho$, for $i > j$ where ρ is a tuning parameter that regulates the sparsity. The smaller the value of the tuning parameter ρ is the most sparse is the estimated matrix $\mathbf{\Theta}$. Such ℓ_1-penalization has become tremendously popular due to its computational attractiveness (i.e. convex function) and its statistical properties which are optimal under certain conditions; mainly, if we want to minimize a prediction error while we are choosing a model as simple as possible. It is important that model selection and parameter estimation are done contemporaneously. In fact, Breiman (1996) showed that using hypothesis testing for model selection this two steps procedure brings instability in the model. Instability means that if we slightly perturb the data our results can change considerably. Whereas, ℓ_1-penalized methodologies allow us to do model selection and estimation, simultaneously.

The most popular method to estimate a sparse precision matrix is probably the graphical lasso proposed by Friedman *et al.* (2008). Here we briefly review graphical lasso. Suppose we have n multivariate normal observations of dimension p, with mean $\mathbf{0}$ and covariance $\mathbf{\Sigma}$. Let $\mathbf{\Theta} = \mathbf{\Sigma}^{-1}$, and let $\mathbf{S}$ be the empirical covariance matrix, the problem is to maximize the log-likelihood

$$(\hat{\mathbf{\Theta}}) := \text{argmax}_{\mathbf{\Theta}}\{l(\mathbf{\Theta}) - \lambda||\mathbf{\Theta}||_1 : \mathbf{\Theta} \succeq 0\}, \tag{2.9}$$

over non-negative definite matrices, i.e $\mathbf{\Theta} \succeq 0$, where $l(\mathbf{\Theta})$ is given in (2.6). Yuan and Lin (2007) solved this problem using the interior point method for the maxdet problem, proposed by Boyd and Vandenberghe (2004). Banerjee *et al.* (2008) developed a different framework for the optimization and solved the problem by optimizing (2.9) over each row and corresponding column of $\mathbf{\Sigma}$ in a block coordinate descent fashion. Partitioning $\mathbf{\Sigma}$ and $\mathbf{S}$

$$\mathbf{\Sigma} = \begin{pmatrix} \mathbf{\Sigma}_{11} & \boldsymbol{\sigma}_{12} \\ \boldsymbol{\sigma}_{12}' & \sigma_{22} \end{pmatrix}, \mathbf{S} = \begin{pmatrix} \mathbf{S}_{11} & \mathbf{s}_{12} \\ \mathbf{s}_{12}' & s_{22} \end{pmatrix},$$

they show that the solution for $\boldsymbol{\sigma}_{12}$ satisfies

$$(\sigma_{12}) := \text{argmin}_{\mathbf{y}}\{\mathbf{y}'\mathbf{\Sigma}_{11}^{-1}\mathbf{y} : ||\mathbf{y} - \mathbf{s}_{12}||_\infty \leq \rho\}. \tag{2.10}$$

This is a box-constrained quadratic program, which they solve using an interior point procedure. Permuting the rows and columns so the target column is always the last, Banerjee *et al.* (2008) solve problem (2.10) for each column, updating their estimate of $\mathbf{\Sigma}$ after each stage. This is repeated until convergence. If this procedure is initialized with a positive definite matrix, they show that the iterates from this procedure remains positive definite and invertible, even if $p > n$. Using convex duality, Banerjee *et al.* (2008) go on to show that solving (2.10) is equivalent to solving the dual problem:

$$(\hat{\boldsymbol{\beta}}) := \text{argmin}_{\boldsymbol{\beta}}\{||\boldsymbol{\Sigma}^{1/2}\boldsymbol{\beta} - \mathbf{b}||^2 + \rho||\boldsymbol{\beta}||_1\}, \tag{2.11}$$

where $\mathbf{b} = \boldsymbol{\Sigma}_{11}^{-1/2}\mathbf{s}_{12}$; if $\boldsymbol{\beta}$ solves (2.11), then $\boldsymbol{\sigma}_{12} = \boldsymbol{\Sigma}_{11}\boldsymbol{\beta}$ solves (2.10). Expression (2.11) resembles a lasso regression (Tibshirani, 1996), and it is the basis for graphical lasso (Friedman *et al.*, 2008). Expanding the relation $\boldsymbol{\Sigma\Theta} = \mathbf{I}$ gives an expression that will be useful below:

$$\begin{pmatrix} \boldsymbol{\Sigma}_{11} & \boldsymbol{\sigma}_{12} \\ \boldsymbol{\sigma}'_{12} & \sigma_{22} \end{pmatrix} \begin{pmatrix} \boldsymbol{\Theta}_{11} & \boldsymbol{\theta}_{12} \\ \boldsymbol{\theta}_{21} & \theta_{22} \end{pmatrix} = \begin{pmatrix} \boldsymbol{I} & \mathbf{0} \\ \mathbf{0}' & 1 \end{pmatrix}. \tag{2.12}$$

Now the sub-gradient equation for maximization of the log-likelihood (2.9) is

$$\boldsymbol{\Sigma} - \mathbf{S} - \rho\boldsymbol{\Gamma} = \mathbf{0}, \tag{2.13}$$

using the fact that the derivative of log$|\boldsymbol{\Theta}|$ equals $\boldsymbol{\Theta}^{-1} = \boldsymbol{\Sigma}$, see (Boyd and Vandenberghe, 2004) for the derivation. Here $\Gamma_{ij} \in sign(\theta_{ij})$; i.e. $\Gamma_{ij} = sign(\theta_{ij})$ if $\theta_{ij} \neq 0$, else $\Gamma_{ij} \in [-1,1]$ if $\theta_{ij} = 0$. Now the upper right block of equation (2.12) is

$$\boldsymbol{\sigma}_{12} - \mathbf{s}_{12} - \rho\boldsymbol{\gamma}_{12} = \mathbf{0}. \tag{2.14}$$

On the other hand, the sub-gradient equation from (2.11) works out to be

$$\boldsymbol{\Sigma}_{11}\boldsymbol{\beta} - \mathbf{s}_{12} + \rho\mathbf{v} = \mathbf{0}, \tag{2.15}$$

where $\mathbf{v} \in sign(\boldsymbol{\beta})$ element-wise. Now suppose $(\boldsymbol{\Sigma}, \boldsymbol{\Gamma})$ solves (2.13), and hence (σ, γ) solves (2.14). Then $\boldsymbol{\beta} = \boldsymbol{\Sigma}_{11}^{-1}\mathbf{w}_{12}$ and $v = -\gamma$ solves (2.15). The equivalence of the first two terms is obvious. For the sign terms, since $\boldsymbol{\Sigma}_{11}\theta_{12} + \boldsymbol{\sigma}_{12}\theta_{22} = 0$ from (2.12), we have that $\theta_{12} = -\theta_{22}\boldsymbol{\Sigma}_{11}^{-1}\boldsymbol{\sigma}_{12}$. Since $\theta_{22} \geq 0$, it follows that $sign(\theta_{12}) = -sign(\boldsymbol{\Sigma}_{11}^{-1}\boldsymbol{\sigma}_{12}) = -sign(\boldsymbol{\beta})$. This proves the equivalence.

Problem (2.11) looks like a lasso problem, which means one wants to minimize the sum of squares subject to the sum of absolute value of the coefficient being less than a constant. In fact if $\boldsymbol{\Sigma}_{11} = \mathbf{S}_{11}$, then the solutions $\hat{\boldsymbol{\beta}}$ are easily seen to equal the lasso estimates for the p-th variable on the others, and hence related to the Meinshausen and Bühlmann (2006) proposal. As pointed out by Banerjee *et al.* (2008), $\boldsymbol{\Sigma}_{11} \neq \mathbf{S}_{11}$ in general and hence the approach proposed by Meinshausen and Bühlmann (2006) does not yield the maximum likelihood estimator.

To solve (2.11), Friedman *et al.* (2008) use $\boldsymbol{\Sigma}_{11}$ and $\mathbf{s}_{12}$, where $\boldsymbol{\Sigma}_{11}$ is our current estimate of the upper block of $\boldsymbol{\Sigma}$. Then update $\boldsymbol{\Sigma}$ and cycle through all of the variables until convergence. The algorithm is described in detail in 2.1.

Algorithm 2.1: Glasso for known zero elements in the precision matrix.

```
Require: $\mathbf{\Sigma}$.
    1. Start with $\mathbf{\Sigma} = \mathbf{S} + \rho\mathbf{I}$. The diagonal of $\mathbf{\Sigma}$ remains unchanged in what follows.
    2. For each $j = 1, 2, \ldots, p$, solve the lasso problem (2.15),
    which takes as input the inner products $\mathbf{\Sigma}_{11}$ and $\mathbf{s}_{12}$. This gives a $p-1$
    vector solution $\hat{\boldsymbol{\beta}}$. Fill in the corresponding row and column of $\mathbf{\Sigma}$ using
    $\boldsymbol{\sigma}_{12} = \mathbf{\Sigma}_{11}\hat{\boldsymbol{\beta}}$.
    3. Continue until convergence.
```

We give an example of graphical lasso for the dataset T-cell. Note that we consider gene expression levels of 58 genes at the first time point, i.e. a static network has been estimated. A range of tuning parameters between 0.002 and 0.5 is considered. To compute the optimal value for the tuning parameter we minimize the `AIC` measure, i.e. we choose the tuning parameter λ, which minimizes twice the logarithm of the likelihood minus twice the number of non-zeroes in the precision matrix. Figure 2.2 represents the estimate static graph.

This algorithm is extremely fast but it is not possible to impose symmetry constraints. Instead, one can consider the idea of Yuan and Lin (2007) which consider the convex optimization problem (2.9) with positive definite constraint on the precision matrix. In particular, Yuan and Lin (2007) use an interior point algorithm to solve the optimization problem. However, the algorithm is feasible for small-medium size problems only since it is necessary to compute the second derivative at each step. Instead, an efficient solver to solve convex optimization problem with linear constraints is `LogDetPPA` (Wang *et al.*, 2009). We will make a large use of this solver. The optimisation algorithm is described in the next Section 2.4.

2.4 Optimization Algorithm

We will take advantage of an algorithm proposed in convex optimization, which is called `LogDetPPA`, and it was proposed by Wang *et al.* (2009) to solve the general convex optimization problem in (2.16) subject to linear constraints on the precision matrix. Our main idea is to impose symmetry constraints on the precision matrix such that specific parameters will be constrained to be equal. This will bring specific structures that come natural for dynamical networks. We will explain this idea in detail in Chapter 3. By imposing symmetry constraints the number of parameters to be estimated is reduced, which is particular important in biological network estimation since the number of replicates is very small compared with the number

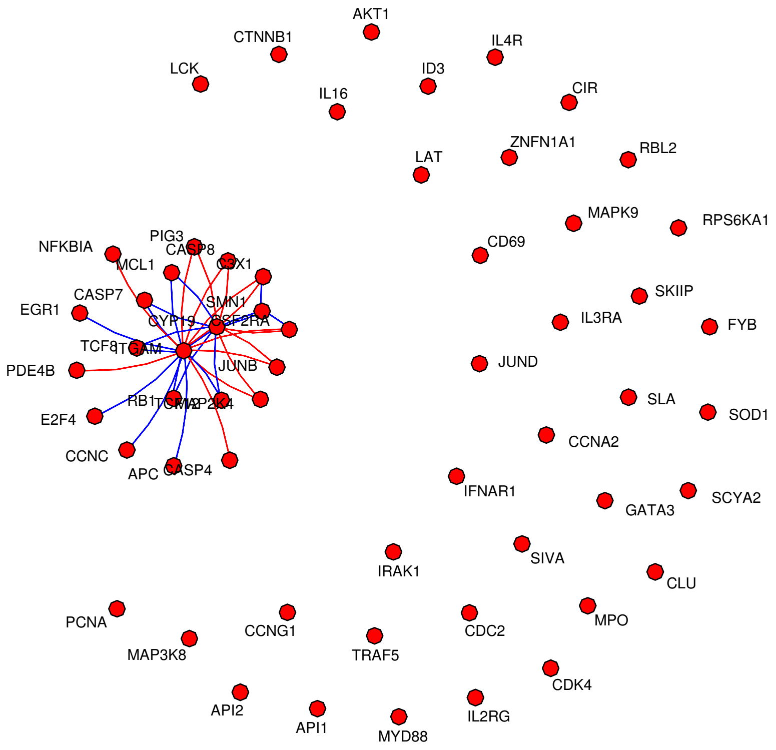


Figure 2.2. Example of a static graph estimated with graphical lasso. Gene expression levels for 58 genes at time point 1 from T-cell data set has been considered. An optimal value of the tuning parameter λ has been chose by minimizing`AIC` measure. The range of the tuning parameter was between 0.002 and 0.5. Blue links represent negative partial correlations and red links represent positive partial correlations.

of variables that are collected (genes). We will see that this is not the only advantage to use the solver `LogDetPPA`, and other model-based graphical models can be implemented. Here, we give a brief overview of the methodology behind the solver `LogDetPPA`. In particular we describe the optimization problem in its standard form. More details on `LogDetPPA` can be found in Wang *et al.* (2009). Note that Toh *et al.* (1999) and Tütüncü *et al.* (2003) proposed a more general solver which is called `SDPT3`. They describe two methods to solve conic programming problems whose constraint cone is a product of semidefinite cones, second-order cones, nonnegative orthants and Euclidean spaces; and whose objective function is the sum of linear functions and log-barrier terms associated with the constraint cones. This includes the special case of determinant maximization problems with linear matrix inequalities. However, we focus on `LogDetPPA` which can solve less general optimization problems with linear constraints but it is more efficient and sufficient for the purpose of this thesis. According to the authors (Wang *et al.*, 2009), sparse covariance selection problems with p up to 2000 and 1.8×10^6 linear constraints can be solved in a reasonable amount of time (about 26 minutes).

`LogDetPPA` employs the idea of a Newton-CG primal proximal point algorithm for solving large scale log-determinant optimization problems. This algorithms employs the essential idea of the proximal point algorithm, the Newton method and the preconditioned conjugate gradient solver. `LogDetPPA` can solve the following optimization problem:

$$(\hat{\mathbf{\Theta}}) := \underset{\mathbf{\Theta}}{\operatorname{argmin}}\{-\gamma \log|\mathbf{\Theta}| + \operatorname{tr}(\mathbf{\Theta}\mathbf{S}) : \mathbf{A}(\mathbf{\Theta}) = \mathbf{b}, \mathbf{\Theta} \succeq 0\}. \tag{2.16}$$

where $\mathbf{\Theta} \in \mathbb{R}^{p \times p}$ is a parameter non-negative definite matrix that needs to be estimated, $\mathbf{S} \in \mathbb{R}^{p \times p}$ is a known matrix, $\mathbf{b} \in \mathbb{R}^m$ is a vector of length m of real numbers, $\gamma \in \mathbb{R}^+$ is a non negative scalar, $\mathbf{A} : \mathbb{R}^{p \times p} \to \mathbb{R}^m$ is a linear map and it can be expressed as:

$$\mathbf{A}(\mathbf{\Theta}) = [\langle \mathbf{A}_1, \mathbf{\Theta} \rangle, \ldots, \langle \mathbf{A}_m, \mathbf{\Theta} \rangle], \tag{2.17}$$

where $\langle \mathbf{A}_i, \mathbf{\Theta} \rangle$, for $i = 1, \ldots, m$, is the Frobenius inner product between matrices $\mathbf{A}_i$ and $\mathbf{\Theta}$. In other words, the linear map transforms the constraints from a matrix form to a system of linear equations. This is a convex optimization problem since $\log|\mathbf{\Theta}|$ and $\operatorname{tr}(\mathbf{\Theta}\mathbf{S})$ are convex functions, with linear constraints. Moreover, we need to be sure that $\mathbf{\Theta}$ is a semi positive definite matrix and this is expressed by $\mathbf{\Theta} \succeq 0$.

The algorithm is slower than `glasso` proposed by Friedman *et al.* (2008) but it is much more flexible. We will show that many interesting model structures can be imposed by re-writing problems in the form (2.16).

2.5 Descriptive measures of networks

In this section, we initially explain some characteristics of biological networks which enable us to better understand the distinction among different types of systems. Moreover, we describe the major mathematical approaches to detect these properties. These approaches are useful to do statistical analysis of networks or graphs. In other words, we want to summarize the characteristics of the network. Then, we focus on the differences between homogeneous and non-homogeneous networks.

2.5.1 Summary measures of networks

Biological networks indicate varieties in terms of their types of connections and structure of nodes like their modularity and randomness. Hereby in order to distinguish them, we define some network measures which are the quantitative criteria describing the general pattern of the genomic connectivities such us *degree distribution, clustering coefficient, characteristic path length and diameter, existence of hubs and network robustness, flux of the reaction, existence of the hierarchical modularity*. These features can be differently indicated based on the directed and undirected networks. The degree distribution, flux mode, and the execution of the hierarchical modularity can be computed under directed networks since their calculations are found by the directions of connections. Whereas the remaining can be used for both types. In the following part we describe each measure in details and present how they can be tested for their statistical significance in a system.

Degree distribution. In an undirected graph, we call *degree* of a node k the number of nodes that are connected with k. For directed graphs we define two type of degrees, namely in-degrees and out-degrees. The in-degree for a node k is the number of parents of k, and the out-degree for a node k is the number of children of k.

The probability on the number of connections of a node k in a system is called degree distribution. If we assume that this marginal probability $p(k)$ is independent for every node in a p-dimensional system where p denotes the total number of nodes, the joint multivariate distribution of nodes $p(k_1,\ldots,k_p)$ for a system can be found from the product of their marginal densities such that

$$p(k_1,\ldots,k_p) = p(k_1)p(k_2)\ldots p(k_p).$$

However, genes are functionally connected to each other so the independence assumption becomes unrealistic. For this reason we consider the conditional independence of the genes. This conditional probability is called the *connectivity distribution*, denoted by p_k and the total number of links attached to the ith gene or node $(i = 1,\dots,p)$ is named as the *degree* or connectivity of node i, shown by k. On the other hand if the system is directed, we can compute two types of connectivity, hereby, conditional distributions: one from the number of links coming to the target gene and one from the number of links departing from the target gene. If we count the number of genes which regulate, i.e. come to, the same gene or node, this number is called the incoming connectivity, arriving connectivity or in-degree, presented by k_{in}. On the contrary, if the interest is the number of genes which are regulated, i.e. departing from the target gene, by the same node or gene, this is called the outgoing connectivity, departing connectivity or out-degree, denoted by k_{out} (Barabási and Oltvai, 2004). For every directed network we can write k in

	1	2	3	4	5	6	7	μ_K
k	0	2	3	2	0	2	3	1.71
k_{in}	0	2	0	2	0	2	0	0.85
k_{out}	0	0	3	0	0	0	3	0.85

Table 2.1. Degree for directed and undirected graph in Figure 2.3.

undirected network via $k = k_{in} + k_{out}$. In biological networks the distribution of k_{out} is generally referred by the power-law density Newman (2010) with the following expression:

$$p_k = c_0 k^{-\lambda}, \tag{2.18}$$

where λ is named as the power-law exponent and stands for the average distance between any two nodes in a system. Apart from the density expression in Equation (2.18), the truncated power-law in Equation (2.19) can be another strong alternative density for biological system.:

$$p_k = \frac{exp(-k/k_c)k^{-\lambda}}{c_1(\lambda,k_c)}, \tag{2.19}$$

where $c_1(\lambda,k_c)$ represents the normalizing factor, λ refers to the power-law exponent, and k_c denotes the cut-off parameter.

The main property of power laws that makes them interesting is their scale invariance. Given a relation $p_k = c_0 k^{-\lambda}$, scaling the argument k by a constant factor causes only a proportionate scaling of the function itself, i.e.:

$$cp_k = c(c_0 k)^{-\lambda} = c_1 p_k \propto p_k. \tag{2.20}$$

Thus, it follows that all power laws with a particular scaling exponent are equivalent up to constant factors, since each is simply a scaled version of the others. This

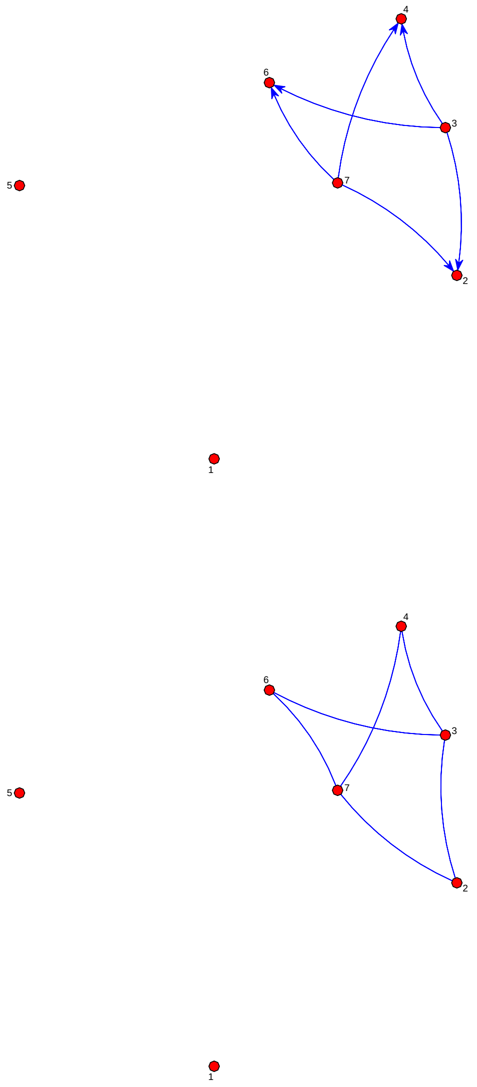


Figure 2.3. An example of directed (Top) and undirected network (Bottom). In the directed graph nodes are related by directed links while in the undirected graph nodes are related by undirected links. For directed graph in-degree (k_{in}) and out-degree (k_{out}) are given in Table 2.1 as well as degree for the undirected graph.

behaviour is what produces the linear relationship when logarithms are taken of both p_k and k, and the straight-line on the log-log plot is often called the signature of a power law. With real data, such straightness is necessary, but not a sufficient condition for the data following a power-law relation. In fact, there are many ways to generate finite amounts of data that mimic this signature behaviour, but, in their asymptotic limit, are not true power laws. Thus, accurately fitting and validating power-law models is an active area of research in statistics.

The idea of the connectivity information can be also combined by the average number of degree for both directed and undirected system. This topological measure is called the *average degree* or connectivity of the network and is computed by the μ_K expression below:

$$\mu_K = \frac{\sum_{i=1}^{p} k_i}{p}$$

where p displays the total number of nodes and k_i shows the number of links associated to the ith node.

Besides power-law and truncated power-law distributions, the generalized Pareto law, stretched exponential, geometric, and combination of these densities can also satisfy other characteristics of biological networks, which are the small-world, centrality, and lethality properties, without the scale-free feature. A *small-world network* is a type of mathematical graph in which most nodes are not neighbors of one another, but most nodes can be reached from every other by a small number of steps. There are various measures of the *centrality* of a vertex within a graph that determine the relative importance of a vertex within the graph. For a review as well as generalizations to these topics see Opsahl *et al.* (2010).

In order to detect the degree or average degree distribution of a system which enables us to distinguish different networks via their connectivities, we can define different approaches. These methods are based on the idea of the goodness of fit test in the sense that they can compare the observed number of links with the theoretical ones by Q-Q (Quantile-Quantile) plots of the data. More details about the calculations are presented in the following.

Mathematical Details. We describe two techniques to control whether the biological networks satisfy the scale-freeness. The first approach is to draw a Q-Q plot between the connectivity relative distribution p_k, which can be a power-law distribution, and the observed number of connections of the node, k. If the graph fits a straight line on the log-log scale, we can conclude the evidence of the power-law density. A similar graphical test can be done via the correlation coefficient $R^2 (0 \leq R^2 \leq 1)$ between the number of connection for each node and k on the log scale. Accordingly, the straight line, i.e. R^2 closes to 1, can be seen as proof of the power-law density, whereas, the skewness from the straight line, i.e. R^2 closes to 0, demonstrates against this evidence. As the second approach, the test can be directly conducted by estimating the power-law exponent λ in $p_k \propto k^{\lambda} (k \geq 1)$ for the dataset. The inference for λ is derived by the maximum likelihood method under the independence assumption of connectivity for node i $(i = 1, \ldots, p)$ in a large network by using the likelihood function below:

$$L(\lambda | k) = \prod_{i=1}^{p-1} k_i^{-\lambda} / \psi(\lambda), \tag{2.21}$$

where $(p-1)$ shows the maximum number of connectivity in a p-dimensional system and $\psi(\lambda)$ is called the Riemann zeta function. If we take the partial derivative of Equation (2.21) on the logarithmic scale and equate it to zero, the normal equation of λ can be found. For the power-law distribution, the solution of this equation does not have a closed form, the estimate for λ can be computed by different iterative techniques such as Newton-Raphson or Grid search (Newman, 2010). Once the estimate of λ, i.e $\hat{\lambda}$, is found, the scale-freeness is checked by the chi-square goodness of fit test power-law with exponent $\hat{\lambda}$. In the testing procedure, a chi-square statistics, χ^{*2}, is obtained from the data by computing O_k and E_k, $(k = 1, \ldots, k^*)$ that describe the observed connectivity and estimated connectivity under power-law, respectively. We can compute

$$\chi^{*2} = \sum_{k=1}^{k^*} \frac{(O_k - E_k)^2}{E_k} \sim \chi^2_{\alpha, k^*-2},$$

in which χ^2_{α, k^*-2} presents the chi-square critical value with $k^* - 2$ degrees of freedom for a given significance level α.

Implementation. For the assessment of the degree distributions, we initially define the adjacency matrix and then compute the associated degrees via degree function within the `igraph` R package. We present the coding of the in-degree distribution k_{in}

```
set1 <  graph.adjacency(obj),
degree(set1,v=V(set1), mode= ''in''),
```

where `obj` must be an adjacency matrix. An adjacency matrix is a matrix A where the generic element $a_{ij} = 1$ if an edge is present between node i and j and $a_{ij} = 0$ if an edge is not present in the graph. In our small example $\chi^{*2} = 10$ with a p-value equal to 0.0015, an acceptance region from 0.0009 to 5.0239, and $\hat{\lambda}$ is 1.75. We have used the library `powerlaw` to calculate theoretical values E_k, while λ was estimated with the function `power.law.fit`.

When we compute the exponent of the degree distribution of a system in order to detect whether it has power-law components λ, we can use the following function:

```
degree.distribution (set1, cumulative=T),
```

where cumulative stands for the calculation of the cumulative probability distribution of the degree of nodes whose graph is described via `set1`.If we plot the underlying cumulative densities versus the cumulative power-law exponent λ under $\lambda = 1,2,3,4$ and if the cumulative density lies within the boundary of the given λ, we can decide on the validity of the small-world property.

Clustering coefficient. The clustering coefficient, typically denoted by C_i, represents the inter-connectivity of a node i $(i = 1,\ldots,p)$ in a network and is computed by the total number of existing links e_i between node's neighbours over the maximum number of links between the neighbours of this node. Hereby, it lies within 0 and 1. $C_i = 0$ represents totally unconnected nodes, and $C_i = 1$ shows totally connected nodes. Since the number of total connections in a system changes with respect to the type of links, i.e. directed or undirected networks, and the appearance of the auto-regulation motifs, i.e. self-loops, the calculation of C_i varies for different choices of networks. For instance, if the network is directed and has no self-loops, C_i can be found via

$$C_i = \frac{2e_i}{k_i(k_i-1)}, \quad i = 1,\ldots,p,$$

where k_i denotes the degree, i.e. number of links, for the ith node and e_i is the total number of existing links between node's neighbours. But if the network is undirected and still has no self-loops, the denominator of C_i is changed by $k_i(k_i + 1)$. We list the expression of C_i for all the types of networks:

- directed self-loops $C_i = \frac{e_i}{k_i^2}$,
- directed no self-loops $C_i = \frac{e_i}{k_i(k_i-1)}$,
- undirected self-loops $C_i = \frac{2e_i}{k_i(k_i+1)}$,
- undirected no self-loops $C_i = \frac{2e_i}{k_i(k_i-1)}$.

In other words, clustering coefficient of a node is the ratio of the number of connections in the neighbourhood of a node and the number of connections if the neighbourhood was fully connected. Here neighbourhood of node i means the nodes that are connected to i but does not include i itself. Note that a fully connected group of k_i nodes has $k_i * (k_i - 1)/2$ connections.

As the clustering coefficients can be computed for individual nodes, we can also get a unique value for whole system by averaging these coefficients. This new statistic is called the average clustering coefficient and denoted by μ_C, and it is calculated as follows:

$$\mu_C = \frac{1}{p} \sum_{i=1}^{p} C_i.$$

Implementation. For the calculation of the clustering coefficient we use the cluster function in the `PCIT` package

```
clusteringCoefficient(adj),
```

where `adj` is an adjacent matrix. Finally in order to compute the average clustering coefficient μ_C, we can take the mean of C_i.

In Figure 2.3 we showed a directed and an undirected graph. We can compute $\mathbf{e} = \{0,0,0,0,0,0,0\}$ and $k = \{0,2,3,2,0,2,2\}$.

We give another example to clarify this concept better where nodes are represented by topics. In Figure 2.4 neighbourhood of topic 6 consists of topics 9, 12, 2 and 1. Between these topics there is only one connection, from topic 2 to topic 12. If the four topics were fully connected, that is there would be a connection from

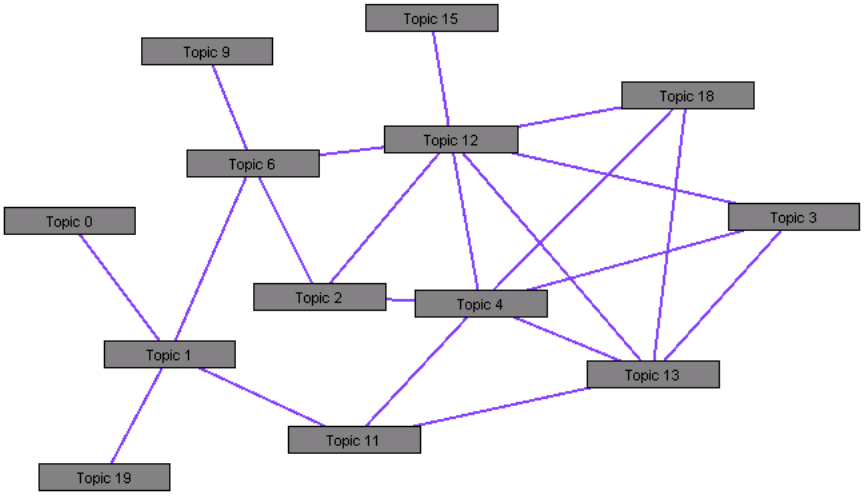


Figure 2.4. Example of clustering connected topics.

each topic to every other topic, there would be $4*3/2 = 6$ connections. Clustering coefficient of topic 6 is therefore $1/6 = 0.17$. Clustering coefficient of topic 1 is 0 because there is no connection at all between topics 0, 6, 11 and 19. Clustering coefficient of topic 3 is 1 because the neighbourhood topics 12, 4 and 13 are fully connected.

Characteristic path length and diameter. The characteristic path length or shortest path length, denoted by $\mathbf{L}$, presents the shortest distance between any two nodes. In a graphical representation, this measure refers to the minimum number of links or edges to go from one node to the other and is computed by

$$\mathbf{L} = \frac{2\sum_{i=1}^{p}\sum_{j=1}^{p} d_{ij}}{p(p-1)},$$

where d_{ij} stands for the number of links that separate node i from node j in the shortest path.

In directed networks, since the path between node i and j, i.e. d_{ij} may not be equal to the path length between node j and i, the smallest distance between these nodes in $\mathbf{L}$ is found by $\min\{d_{ij}, d_{ji}\}$. Whereas for undirected network we can accept $d_{ij} = d_{ji}$. On the other hand if we deal with the longest, rather than the shortest, path length, this measure is called the diameter, $\mathbf{D}$, and can be formulated by $\mathbf{D} = \max\{d_{ij}\}$. In a system, small $\mathbf{L}$ implies dense connections between nodes generating the hubs. If $\mathbf{L}$ is close to μ_C, the connections become sparse and the nodes have almost equal number of links. Accordingly in a biological sense, for both $\mathbf{L}$ and $\mathbf{D}$, the small values present the fast actions and the big ones refer to slower actions within intermediate stages of the system. In other words, we can analyze $\mathbf{L}$ and $\mathbf{D}$ for the interpretation of the speed of communication between nodes as well. If the system has very small $\mathbf{L}$ and $\mathbf{D}$ while the clustering coefficients are large and the power exponent of the out-degree distribution λ satisfies $\lambda > 3$, we name that the system maintains the small-world property. But the biological network typically displays shorter $\mathbf{L}$ than this, resulting in λ between 2 and 3. We call this feature as the ultra small-world property. Both small-world and ultra small-world properties also stand for the modular structure in a system which is another common feature of biological networks. The investigation of $\mathbf{L}$ and $\mathbf{D}$ values are computed by the breadth-search method (Barabási and Oltvai, 2004). Let's consider the small undirected graph given in Figure 2.3. Table 2.2 shows the matrix of the shortest path length $\mathbf{L}$ among the seven nodes present in Figure 2.3. Note that, if no path is present between node i and j then $l_{ij} = \infty$ which means one cannot reach j from i and vice versa. The average path length μ_L is 1.4.

	1	2	3	4	5	6	7
1	0	∞	∞	∞	∞	∞	∞
2	∞	0	1	2	∞	2	1
3	∞	1	0	1	∞	1	2
4	∞	2	1	0	∞	2	1
5	∞	∞	∞	∞	0	∞	∞
6	∞	2	1	2	∞	0	1
7	∞	1	2	1	∞	1	0

Table 2.2. Short path length for undirected graph in Figure 2.3. The symbol ∞ is used to indicate that there is no path between the relative nodes.

Mathematical Details. In the calculation of the diameter via the breadth-search algorithm, we initially consider each node separately such that for each transcription factor, node or gene, all of its neighbours are labelled by 1 as the distance measure. Then the nodes which have connections with these neighbours are labelled as having a distance of 2. This procedure iteratively continues until all nodes are encountered. In the computation if the same node is needed to be counted several times, the first distance is taken for that node. On the other hand if we are interested in the shortest path length, we need to consider all possible paths between any two nodes. In the detection, we perform the algorithm in two stages in the sense that first of all we apply the same algorithm for the source node as described above and then we conduct another breadth-search algorithm for the target genes. By this way, a path is constructed among nodes from the transcription factor to its target genes. Then the nodes, which are the neighbours of the transcription factor, are taken in the same layer, whereas the nodes that are the neighbours of these nodes are placed in a further layer. This process is repeated iteratively till all nodes are covered. Finally to compute the shortest path length, each node is scored with its shortest distance from the transcription factor to the target gene and other way around. We choose the minimum one as the shortest path length.

Implementation. In a network, the calculation of the characteristic path length for each node can be calculated via `shortest.paths` function within the `igraph` package by the following inputs:

```
shortest.paths(set1, mode= "all")
```

where `set1` denotes the graphical representation of the system as previously performed and mode presents the option whether the path length is investigated from a particular node or it is calculated to a particular species, or alternatively it is found without directional information. With respect to the question of interest, hereby, the mode can be equated as `mode=”in", "out"`,or `"all"`,respectively. But if the graph is undirected, the function merely computes the option `mode=”out"`as the default. On the other side for the calculation of average shortest path length μ_L and diameter **D**, we can call

```
average.path.length(set1)
diameter(set1, directed=T)
```

in order. In the calculation of the diameter, if the graph is undirected, the function considers the option of `directed=F` as the default.

Existence of hubs and network robustness. In biological networks, the number of connections for the nodes indicates a heterogeneous structure in the sense that the majority of the genes has very few links with other nodes and only a small group of genes possesses many links with others. We call these highly connected nodes as the hubs or global regulators. The small value of the shortest path length **L** can display the validity of this feature in the system. In fact the presence of hubs also enables us to observe the network robustness which represents the invariance of the network from the removal of random nodes. Because the connections between the major functional groups, i.e. modules, in the system are kept by hubs and unless we do not exclude them, the system behaves resistantly to such attacks. On the other hand if we remove them, then the system is split into isolated node clusters which causes lethal disability in certain functions. Hereby in a network, the presence of hubs, controlling the actual connectivity of the pathway, can also be remarked as the *centrality principle* and their abilities to direct the overall system are presented by the *lethality principle*. To detect the robustness in a network, we can use different approaches. For instance we can compute the characteristic path length since it indicates the connectivity in a system. If the system keeps the same path length after the removal of random nodes,then we have evidence of robustness. Moreover we can also implement the entropy measure to observe the change in the system after the random attacks. More details about these measures are given in the following Mathematical Details.

In biological networks, since the hubs have crucial roles, they are investigated extensively. From empirical studies it has been found that we can define them into two types with respect to their connected nodes, also called partners, expression profiles. These are the *date and party hubs*. The former is generally sensitive to the time and sub-cellular localization, i.e. spatial distribution, of its partners. Moreover it can work inside of the modules and its removals affect the whole network. On the contrary the latter is typically observed as insensitive to the time and can interact with many nodes simultaneously. Furthermore it is seen that these hubs are active outside of the modules and regulate the interactions between different clusters. Hereby their exclusions from the system do not lead to any lethal effect which causes to collapse whole major functions of the system. On the other hand, recent studies in yeast protein-protein interaction network report that there is no very clear difference between party and date hubs, rather, they behave more homogeneously than we may consider.

In order to classify hubs as date or party, Han *et al.* (2004) suggest to use the average Pearson correlation coefficient (PCC). In this division we calculate PCC for each hubs having five or more connections. Since the significance of PCC can change regarding the selected cut-off value such 0.8 or greater than this, the plots of PCC can alter based on the cut-off point. If there is no biological knowledge about this critical point in the average PCC, we can choose 0.5 as the natural choice to distinguish the groups. Once they are classified, we can analyze the common behaviours of these groups under distinct conditions, such as toward the change in time or their abilities of simultaneous actions. In these calculations, Han *et al.* (2004) suggest the entropy measure to capture the group differences. On the other hand, Agarwal *et al.* (2010) categorize hubs into two groups, namely, the permanent and transient group, with respect to their effects under different conditions. The permanent hubs connect the essential transcription factors or nodes to each other so that the action between vital nodes can be maintained. Therefore they can be composed of multiple functional transcription factors and are invariant to cellular states or phases. Whereas the transient hubs constitute the majority of the network and are highly sensitive to one condition and less sensitive to others. Due to their specificities of cellular phases, they can be important, in particular, for producing drug targets. For the detection of two groups, Agarwal *et al.* (2010) perform the trace-back algorithm which is used for the identification of the motifs in a system.

Mathematical Details. The investigation of the network robustness in biological systems can be implemented by the entropy via measuring the rate of information flow within the system, as suggested by Demetrius and Manke (2005). Actually the entropy is a well-known measure, especially, in physics which shows the measurements of the available energy in a system. In statistical thermodynamics, it represents the amounts of the uncertainty within the relation of unobservable particles in a network while the observable properties or system parameters such as the pressure, temperature or volume change. Hereby when the entropy increases, the uncertainty of the system raises too. Demetrius and Manke (2005) defines the following entropies in order to measure the functional and structural variability in the system after the removal of random nodes.

- Localization entropy of nodes: This entropy E_{loc} can be found as

  $$E_{loc} = -\sum_{i=1}^{n_s} L_i \log(L_i)$$

  where $L_i = T_i / \sum_{i=1}^{n_s} T_i$ and displays the fraction of the specific ith localization, apart from very large scale localization such as the cytoplasm or cell membrane. On the other hand T_i represents the frequency of the ith subcellular localization within all connected nodes, i.e. partners, of a hubs and n_s denotes the total number of different sub-cellular localizations for all connected nodes of this hubs.

- Function entropy and diversity of the module: The entropy E_{func} can be used to measure the functional diversity of the modules after removing the random nodes and is calculated by

  $$E_{func} = -\sum_{i=1}^{n_s} F_i \log(F_i)$$

  in which $F_i = T_i / \sum_{i=1}^{n_s} T_i$ and gives the fraction of the specific ith function category, except from very large scale categories such as the mitochondrial and nucleus. T_i stands for the frequency of the ith function category within all connected nodes of the selected hubs and n_s shows the total number of different function category in the module. Accordingly in order to find the diversity of this sub-network, we can take the ratio of the unique function category over the total number of function categories in that module.

Once E_{loc} and E_{func} are obtained as the estimates of the actual module parameters, their statistical significances are tested by comparing the results of random modules with the same size and the same amount of random nodes removals from the system.

Flux of the reaction. In biological networks, to understand the true description of the system, the knowledge of the strength of interactions under different biological and environmental conditions becomes important. The flux of the reaction is a measurement of the intensity or strength of the interaction in metabolic networks and refers to the amount of products per unit of time in a reaction. In the system this information can help us to find essential reactions in the activation and make prediction about them. On the other hand in genetic networks, the idea of the strength is typically controlled by the correlations. The pair of nodes or genes which has a high correlation coefficient can display the direct interaction between the proteins and can be seen as an indication of the strong strength with respect to other genes whose interactions can be indirect.

Other measures. Apart from the listed topological measurements, there are some other features which can be chosen to evaluate the networks. But due to their computational demands, they do not have common application. One of these measures is called the graphlet degree distribution. Similar to the degree distribution, it computes the distribution of the number of nodes connecting to k clusters or graphlets, rather than k connections, as the degree distribution. The correlation for profile is another measure networks which computes the correlations between degrees of its neighbour nodes. In the calculation, we find the number of connections between two nodes having degrees k_1 and k_2, i.e. $N(k_1,k_2)$, and compare it with a random network by $Nr(k_1,k_2) \pm \Delta Nr(k_1,k_2)$ generated by simulation. Here Δ displays the change in the given expression. In order to investigate the statistical significance, the method initially converts both $N(k_1,k_2)$ and $Nr(k_1,k_2)$ into the joint probability by dividing these values to k, total number of connections such that $P(k_1,k_2) = N(k_1,k_2)/k$ and $Pr(k_1,k_2) = Nr(k_1,k_2)/K$, respectively. Then a z-score is calculated by the following expression:

$$Z(k_1,k_2) = \frac{P(k_1,k_2) - Pr(k_1,k_2)}{\sigma_r(k_1,k_2)},$$

where $\sigma_r(k_1,k_2)$ is the standard deviation of $Pr(k_1,k_2)$.

2.5.2 Characteristic of different networks

Networks interpretation is usually related to the distribution of the links which can be classified as homogeneous networks and non-homogeneous networks. The networks, classified based on the distribution of the links, display major differences on their topological features. These differences help us to distinguish whether the network is random, scale-free, hierarchical, or modular. We present the topological properties of each network type in the following part.

Random network. The random network belongs to the homogeneous type of networks in the sense that every node in the system owns a similar number of interactions K whose degree may be Poisson distributed with mean $\mu_K = \lambda$. Hereby the in-degree of the ith node is described by the distribution function below:

$$p_k = \frac{\lambda^k e^{-\lambda}}{k!}$$

where λ denotes the mean number of connections per node. On the other hand, the clustering coefficient C_i, where $i = 1, \ldots, p$ and p is the total number of nodes in the system, also indicates differences in distinct networks. Accordingly the random networks, different from most of the non-homogeneous networks, have C_i which is invariant to the degree of the node and the most of C_i is approximately close to each other. This feature means that there is no highly connected nodes, resulting in no hubs and no clusters in the system. Hereby the average clustering coefficient of the system μ_C is independent on the number of connections K. Therefore, if we draw the plot of C_i as a function of k_i, we can observe a horizontal straight line on the original scale, showing that there is no inherent modularity in the construction of the network. Moreover, with respect to the average distance between any two nodes, μ_p, we can detect a proportional relation between the mean of path length L and the logarithm of p, i.e. $\mu_p \propto \log(p)$.

We present a basic form of the random network in Figure 2.5 for simplicity as well as the figure of C_i as a function of k_i for this random network. Here the average short path is 3.18 and the log of the number of node is 3.91.

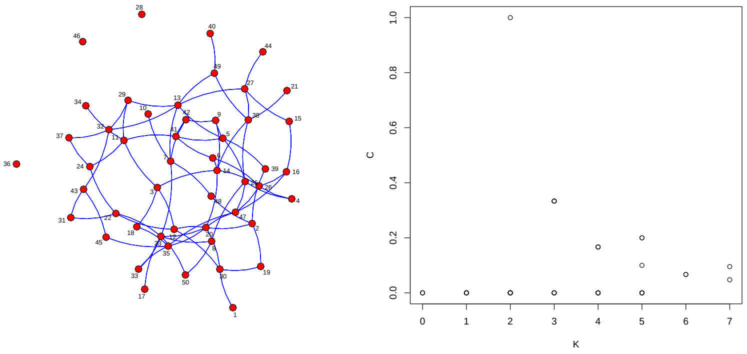


Figure 2.5. A random network is represented on the left, and the relations between clustering coefficient (y-coordinate) and degree (x-coordinate) for the random network are represented on the right.

This proportional relation implies that the random networks do not show the small-world property as the consequence of the Poisson distributed links for each node. Moreover due to this feature, the fluxes of the metabolic reactions in these systems can carry in a linear pathway under the steady-state concentrations of all metabolites and do not show any exponential behaviour such as the power-law property in the flux of the distribution for the ith reaction.

Scale-free networks. Scale-free or Barabasi-Albert networks are common network types in biology. These networks remark a non-homogeneous structure in the distribution of links, meaning that the number of connections of each node i $(i = 1, ..., p)$ can change a lot. For instance, in a very typical graph for genomic networks, most of the genes interacts with one or two other genes whereas, only some of them are bounded with many genes. Hereby, in scale-free networks, to capture the underlying variety in the number of connections for every node, the degree distribution of nodes is explained by exponential functions. The exponential function for the degree distribution of node k, which allows us to construct both highly connected and very sparse nodes with heavy tailed densities, is written as:

$$p_k = c_0 exp(\beta k),$$

where c_0 is a normalization factor and β denotes the exponential exponent. In p_k, β describes the number of the regulating genes which arrives at the same gene or node. When β increases, it is observed that many genes direct the same target genes, thereby cause less target genes and many regulators in the network. On the other hand, one can describe the density by power-law

$$p_k = c_0 k^{-\lambda},$$

even though this density is not unique in all biological networks. Indeed, from the detection of true distribution in certain empirical studies it is shown that the truncated power-law better fits the biological data without loosing other characteristics of the biological networks like centrality and lethality principles. Moreover, other possible distributions such as the generalized Pareto law, stretched exponential, geometric, and their combined densities can be alternatives of the power-law by maintaining the properties of the network except for the scale-freeness. These results show us that maybe the scale-free considered data are not in fact scale-free, but still satisfy the exponential density in the probability of the number of connections per node. On the other hand still considering that the scale-free networks have power-law distribution as the degree of each node which can be proportionally formulated by $k^{-\lambda}$ while λ represents the degree exponent of the power-law and k displays the number of links related to the nodes, we can find the possible range for λ in biological networks. Although λ can take any number from 2 to ∞ by definition, we can observe the following types of λ in biological networks:

- if $2 < \lambda < 3$, the network has highly connected nodes which indicates a shortest path length proportional to $\log(\log(p))$ where p stands for the total number of nodes in the system. Such a short distance between two nodes implies that the system owes the ultra-small world characteristics and most of the known biological networks possesses this feature.

- If $\lambda = 3$, the system has relatively less densely connected nodes in the sense that the shortest path length becomes proportional to $\log(p)/\log(\log(p))$.
- If $\lambda > 3$, the network indicates a shortest path length proportional to $\log(p)$, which describes moderately less connected nodes with respect to the networks in the previous two choices.

There are two important consequences of such dense connections. Firstly, in scale-free networks, we cannot find an average number of links per node, μ_k, in opposite to the random networks. Secondly, hubs will be present in the network. As a result if we plot a graph of the degree distribution p_k versus the number of connections k, we can obtain a linear decreasing function on the logarithmic scale. In other words, most nodes have few links and few nodes have very dense connections. Figure 2.6 shows a scale free network and its degree distribution. Note that we have approximately a continuous function whereas $k \in \mathbb{N}$. The maximum likelihood estimate for λ is 2.39, the shortest path length average is 3.72, and $log(log(p))$ is equal to 1.36.

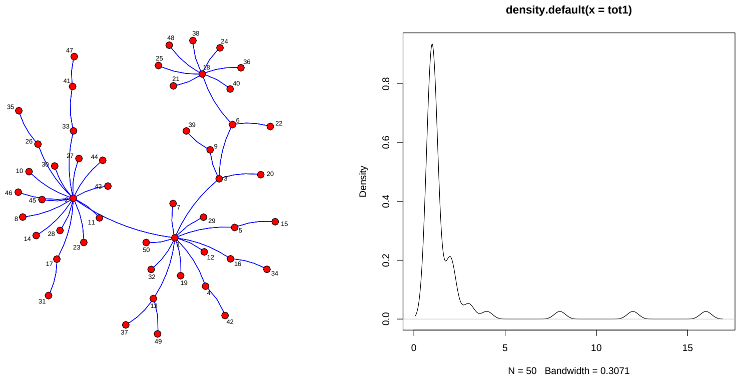


Figure 2.6. Example of scale free networks (left) and estimated degree distribution (right). The estimated degree distribution has been approximately with a continuous function.

Hierarchical and modular networks. The hierarchical and modular networks are the other two types of non-homogeneous networks classified based on the distribution of links. Thereby different from the scale-free systems, they display an inherent clustering in their constructions. If the system possesses iterative connections of clusters linked to each other, resulting in a tree structure, this type of networks can generate the hierarchical networks without scale-freeness. Whereas if the nodes are connected to each other iteratively in absence of the hierarchy as well as scale-freeness, we can observe a modular network. In both hierarchical and modular systems, since the modularity design is detected from their constructions, their average clustering coefficients μ_K are linearly proportional to the total number of connections K with the ratio of $1/K$ on the logarithmic scale, different from the scale-free networks. This feature presents that sparsely connected nodes are linked to highly clustered areas and the connections between clusters of nodes are hold by few nodes that display the presence of hubs. Hereby if we draw the plot of C_i versus k_i, we observe a straight line with slope -1 on the logarithmic scale. But similar to the scale-free systems, most of Ci's has non-homogeneous values. On the other hand for their other topological features, we can accept the characteristic of the scale-free networks as long as the hierarchical architectures of these systems are generated inherently since the high connections of the underlying sub-networks already produce scale-free systems ultimately. Figure 2.7 is drawn to present a hierarchical and a modular structure.

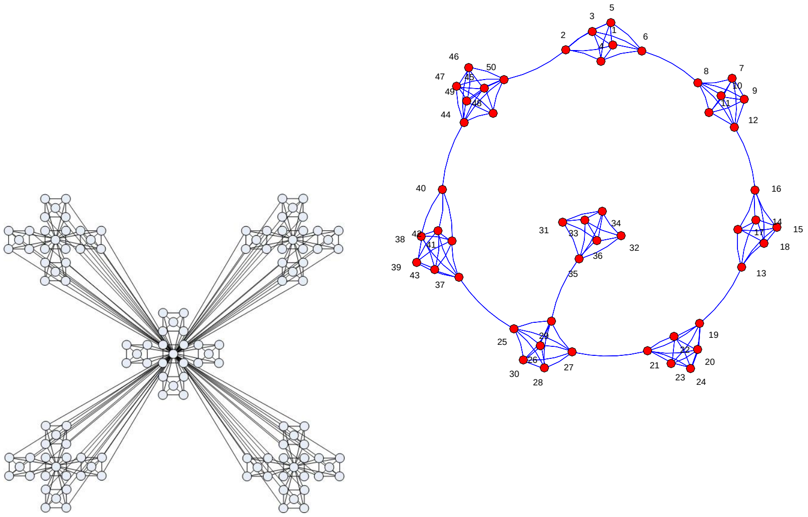


Figure 2.7. A hierarchical (left) and a modular network (right).

2.6 Summary

In this chapter, we have introduced graphical models. In particular, Gaussian graphical models, Bayesian networks and Ising models have been considered. Gaussian graphical models can be used to model conditional independence relationships when the direction of the links are not known. In other words, Gaussian graphical models do not consider causal implication. Instead, Bayesian networks can be used when the direction of the links are known a priori. This means that implication relationships maybe argued from the directed graph. Ising model has shown that graphical models can be extend when the set of links is in the graph is no finite. We have introduced the regularization framework since classical graphical models are not suitable for high-dimensional setting, i.e. the study of systems where the number of variables is very large and the number of samples is few. Accuracy and consistency of the results depend on the assumption of sparsity. The graphical lasso has been described since it is the most used in the literature. However, graphical lasso cannot incorporate any linear constraints. Whereas, we have seen that the optimization algorithm `LogDetPPA` is able to solve log determinant convex optimization problems subject to linear constraints. This algorithm will be used in the rest of this thesis. Finally, summaries statistics to describe graph characteristics are given.

Chapter 3

Factorial graphical lasso for biologycal dynamic networks

Graphical models are a powerful tool for analysing relationships between a large number of random variables. Their fundamental importance and universal applicability are due to a number of factors. Graphs can be used to represent conditional dependence between random variables. Besides, their structure is also modular which means that complex graphs can be built from many single graphs or components. This modular structure is useful to describe complex systems in a simple way. Gaussian graphical models (GGMs) are graphical models in which it is assumed that nodes of the graph are in a one-to-one correspondence with continuous random variables which follow a multivariate normal distribution. The most important property for GGMs is that the concentration matrix, i.e. the inverse of the covariance matrix, represents the conditional independence. The inverse covariance matrix can be estimated by maximizing the likelihood function over all the space of the precision matrices such that the result estimator is positive definite. The maximum likelihood estimator (MLE) for the concentration matrix is the inverse of the sample covariance matrix, and it exits when the number of observations is greater than the number of random variables.

The literature on estimating an inverse covariance matrix goes back to Dempster (1972), who advocated the estimation of a sparse dependence structure, i.e., setting some elements of the inverse covariance matrix to zero. The complexity of the covariance matrix is reduced when elements in the inverse of this matrix are fixed at zero. Moreover, it has been shown that most of the networks in biology are sparse, which means that most of the elements in the precision matrix are equal to zero. The standard approach in statistical modeling to identify zeros in the precision matrix is the backward stepwise selection method, which starts by removing the least significant edges from a fully connected graph, and continues removing edges until all remaining edges are significant according to an individual partial correlation test..

This procedure does not take into account for multiple testing i.e. many of the links will be estimated to be different from zero when they are not and vice versa. A conservative simultaneous testing procedure was proposed by Drton and Perlman (2004). However, Breiman (1996) showed that this two-steps procedure, in which parameter estimation and model selection are done separately, can lead to instability. For instability Breiman (1996) means that small changes in the dataset or small perturbations results in completely different estimated graph structures. .

The idea of Tibshirani (1996), which has been extensively and successfully applied in regression models, can be used to estimate sparse graphs, i.e. to induce zeros in the estimated inverse covariance matrix. This idea is based on the ℓ_1 norm penalty, i.e. the sum of the absolute values of the inverse of the covariance matrix has to be less or equal to a tuning parameter. The smaller the tuning parameter is, the more zero will be estimated in the precision matrix. Meinshausen and Bühlmann (2006) proposed to select edges for each node in the graph by regressing the variable on all other variables using ℓ_1 penalized regression. This method reduces to solving p separate regression problems, and does not provide an estimate of the matrix itself. Penalized maximum likelihood approaches using the ℓ_1 penalty have been considered by Yuan and Lin (2007); Banerjee *et al.* (2008); d'Aspremont *et al.* (2006); Friedman *et al.* (2008); Rothman *et al.* (2008), who have all proposed different algorithms for computing this estimator. This approach produces a sparse estimate of the inverse covariance matrix, which can then be used to estimate a graph, and has been referred to as the graphical lasso (Friedman *et al.*, 2008), or sparse permutation invariant covariance estimator (Rothman *et al.*, 2008). Theoretical properties of the ℓ_1 penalized maximum likelihood estimator in the large p scenario were derived by Rothman *et al.* (2008); Meinshausen (2008). This method has been extended by several research group. Fan and Li (2001) introduced clipped absolute deviation penalty. On the other hand, Lam and Fan (2009) extended this penalized maximum likelihood approach to general nonconvex penalties. The latter authors also established a so called "sparsistency" property of the penalized likelihood estimator, implying that it estimates true zeros correctly with probability tending to one. Alternative penalized estimators based on the pseudolikelihood instead of the likelihood were recently proposed by Peng *et al.* (2009); the latter paper also established consistency in terms of both estimation and model selection..

The complexity of the model can be reduced if one imposes some symmetry constraints on the precision matrix, i.e. the number of parameters to be estimated is reduced by intruding equality constraints. Recently, Højsgaard and Lauritzen

(2008) proposed Gaussian graphical models with symmetry. An important motivation for considering structured GGMs is that conditions for maximum likelihood estimates to exist are less restrictive than for standard GGMs. Lauritzen (1996) showed that without any restriction on the concentration matrix, the existence of the MLE for graphical Gaussian models is ensured with probability of one if the number of observations is larger than the cardinality of the largest clique in the graph. Højsgaard and Lauritzen (2008) showed that imposing restriction on the concentration matrix ensure the existence of the MLE even if the number of observations is less than the number of variables. However, Gaussian graphical models with symmetries perform parameter estimation and model selection separately, which brings, to instability of the estimation as shown by Breiman (1996)..

Penalized graphical models and GGMs with symmetries lack specific time dynamic structures, which results in a lack of a consistent interpretation of the concentration parameters across time-points. Our idea is to combine symmetry models and graphical lasso to reach the important result to estimate dynamic networks in biological systems. In particular, we propose two models that specify penalized time-dynamic structures on the precision matrix (SGL_Θ) and on the conditional correlation matrix (SGL_Ω), which is the negative scaled concentration matrix. The latter model is useful when random variables are not measured on the same scale. We structure graphical models in a way that comes naturally to time-course data and we use the idea of coloured graphs so that links in the same colour set represent equality constraints on the precision matrix.

In this chapter we focus on constrained Gaussian graphical models since our aim is to model graphs with dynamic structures for time-course genetic data. Section 3.1 decribes two motivating examples. Section 3.2 introduces notations, preliminaries and definitions. In Section 3.3 SGL_Θ and SGL_Ω are descibed. Section 3.4 addresses model selection and parameter smoothing selection, which are important issues in graphical lasso. In particular, classical approaches such as AIC, BIC are derived to do model selection, and stability selection (Meinshausen and Bühlmann, 2010) has been adapted to factorial graphical lasso models. Section 3.5 shows encouraging numerical results. A time-course microarry data set on human T-cells is studied.

3.1 Motivating examples

DNA microarray are essential tools in genetic to measure the expression levels of genes simultaneously or to genotype multiple regions of a genome. Since an array can contain tens of thousands of probes, a microarray experiment can accomplish many genetic tests in parallel. Therefore arrays have dramatically accelerated many types of investigation. In standard microarrays, the probes are synthesized and then attached via surface engineering to a solid surface by a covalent bond to a chemical matrix. The solid surface can be glass or a silicon chip, in which case they are colloquially known as an Affy chip when an Affymetrix chip is used. Other microarray platforms, such as Illumina, use microscopic beads, instead of the large solid support. Alternatively, microarrays can be constructed by the direct synthesis of oligonucleotide probes on solid surfaces. DNA arrays are different from other types of microarray only in that they either measure DNA or use DNA as part of its detection system. DNA microarrays can be used to measure changes in expression levels, to detect single nucleotide polymorphisms (SNPs), or to genotype or resequence mutant genomes. A good introduction on microarray experiments and statistical analysis for microarray is given by Wit and McClure (2004). Statistic microarray experiments are snapshots of the expression of genes in different samples. However, gene expression is a temporal process. Different proteins are required (and synthesized) for different functions and under different conditions. Even under stable conditions, due to the degradation of proteins, mRNA is transcribed continuously and new proteins are generated. This process is highly regulated. One of the most important ways in which the cell regulates gene expression is by using a feedback loop. Some of the proteins are transcrip- tion factors (TFs). These proteins regulate the expression of other genes (and possibly, their own expression) by either initiating or repressing transcription. When cells are faced with a new condition [such as starvation (Natarajan et al., 2001), infection (Nau et al., 2002) and stress (Gasch et al., 2000)], they react by activating a new expression program. In many cases, the expression program starts by activating a few TFs, which in turn activate many other genes that act in response to the new condition. Taking a snapshot of the expression profile following a new condition can reveal some of the genes that are specifically expressed under the new condition. However, in order to determine the complete set of genes that are expressed under these conditions, and to determine the interaction between these genes, it is necessary to measure a time course of expression experiments. This allows us to determine not only the stable state following a new condition, but also the pathway and networks that were activated in order to arrive at this new state.

To summarize time-course information provides valuable insight into the dynamic mechanisms underlying the biological processes being observed, and dynamic graphs can be used to visualize this information. A time-course genetic dataset T-cell is our first motivating example.

Example 1: Human T-cell microarray data. A t-cell dataset observed from a large time-course experiment to characterize the response of a human time-cell line (Jurkat) to PMA and ionomycin treatment (Beal *et al.*, 2005; Rangel *et al.*, 2004). Two experiments were conducted (tcell.34) and (tcell.10). The first data set (tcell.34) contains the temporal expression levels of 58 genes for 10 unequally spaced time points. At each time point there are 34 separate measurements. The second data set (tcell.10) stems from a related experiment which considers the same genes and identical time points (0, 2, 4, 6, 8, 18, 24, 32, 48, 72 hours). For illustrative purpose, we select a subset of 4 genes $\Gamma = \{ZFN, CGN, SIV, SCY\}$ across 2 time points $T = \{1,2\}$ and show the empirical concentration matrix $\mathbf{S}^{-1}$ for the selected subset based on 44 observations in Table 3.1. $\mathbf{S}^{-1}$ can be partitioned as follows:

$$\mathbf{S}^{-1} = \mathbf{I}_{\Gamma T} \odot \mathbf{S}^{-1} + \begin{pmatrix} \mathbf{0} & \mathbf{I}_{\Gamma\Gamma} \\ \mathbf{I}_{\Gamma\Gamma} & \mathbf{0} \end{pmatrix} \odot \mathbf{S}^{-1} \begin{pmatrix} \mathbf{D}_{\Gamma\Gamma} & \mathbf{0} \\ \mathbf{0} & \mathbf{D}_{\Gamma\Gamma} \end{pmatrix} \odot \mathbf{S}^{-1} + \begin{pmatrix} \mathbf{0} & \mathbf{D}_{\Gamma\Gamma} \\ \mathbf{D}_{\Gamma\Gamma} & \mathbf{0} \end{pmatrix} \odot \mathbf{S}^{-1},$$

where $\odot$ is the element-wise matrix multiplication, $\mathbf{I}_{\Gamma T}$, $\mathbf{I}_{\Gamma\Gamma}$ are identity matrices, and $\mathbf{D}_{\Gamma\Gamma}$ is a square matrix with one off-diagonal. Four terms are showed in the summation: The first one indicates how well the variance of $(gene_{ij})_{i \in \Gamma, j \in T}$ is predicted given the rest; the second and the fourth ones indicate self conditional independence at temporal lag 0 and 1, respectively; the third one indicates networks at lag interaction 1. In Section 3.3 we partion the concentration according to the natural partitions.

Time				*1*				*2*	
	Gene	*ZNF*	*CCN*	*SIV*	*SCY*	*ZNF*	*CCN*	*SIV*	*SCY*
1	*ZNF*	1.38	-0.05	-0.50	0.25	-0.20	-0.12	-0.01	-0.11
	CCN	-	1.58	0.02	-0.39	-0.12	-0.93	-0.02	0.07
	SIV	-	-	1.61	-0.13	-0.17	0.12	-0.78	-0.02
	SCY	-	-	-	1.29	0.05	0.25	0.49	-0.09
2	*ZNF*	-	-	-	-	1.10	-0.18	0.04	0.08
	CCN	-	-	-	-	-	1.78	0.06	0.42
	SIV	-	-	-	-	-	-	1.58	0.09
	SCY	-	-	-	-	-	-	-	1.16

Table 3.1. Maximum likelihood estimator of the precision matrix or empirical concentration matrix $\mathbf{S}^{-1}$ based on 44 replicates for 4 genes measured across 2 time points. The number of genes measured in the T-cell experiment was 58 across 10 time points but we randomly selected four genes to give an illustration of the estimated precision matrix.

Note that (Beal *et al.*, 2005; Rangel *et al.*, 2004) estimated a directed graph with latent structures which result in a summary of the 9 dynamic directed possible graphs. To compare our graph with the graph proposed by these authors we will use a specific factorial graphical lasso. However, SGL_{Θ} is much more general since the 9 graphs could be estimated.

Example 2: Educational study. A different field in which factorial graphical models can be applied to estimate dynamic graphs is in social science. Here we give an example of application for the Edu dataset. Edu dataset is a longitudinal dataset in which a set of scores to analyse teacher-student relationship were collected accros differnt time points $0,1,4,7,10$ months. The study were conducted in the Netherlands and the students were follow up from their first year of the secondary school (Opdenakker and Maulana, 2012; Opdenakker *et al.*, 2011). In particular, 20 classes of students and 24 educational scores were considered. For illustrative purpose, we consider a subset of four scores by naming the variables: Controlled (Contr), Autonomies (Auton), Influence (Infl) and Proximity (Prox). The first two scores concern student behaviours, taking values between -3 and 3. The last two scores measure teacher behaviours and take values between 0 and 5. The scale for these variables is not compatible and conditional correlations are therefore more meaningful than the concentrations. In section 3.3, we show that conditional correlations are invariant under changes of scale for individual variables.

Time		*1*				*2*			
	Subject	*Contr*	*Auton*	*Infl*	*Prox*	*Contr*	*Auton*	*Infl*	*Prox*
1	*Contr*	1.79	*0.31*	*-0.07*	*-0.00*	*0.61*	*-0.23*	*0.18*	*-0.06*
	Auton	-0.57	1.86	*0.03*	*0.22*	*-0.19*	*0.57*	*-0.07*	*-0.03*
	Infl	0.11	-0.05	1.25	*-0.04*	*0.00*	*-0.01*	*0.41*	*0.09*
	Prox	0.01	-0.37	0.05	1.55	*0.04*	*-0.03*	*-0.12*	*0.51*
2	*Contr2*	-1.09	0.35	-0.00	-0.07	1.76	*0.31*	*-0.08*	*-0.01*
	Auton	0.43	-1.09	0.02	0.05	-0.58	1.96	*0.13*	*0.21*
	Infl	-0.28	0.11	-0.53	0.18	0.12	-0.21	1.37	*0.21*
	Prox	0.10	0.06	-0.13	-0.82	0.02	-0.39	-0.32	1.68

Table 3.2. Empirical conditional correlations (upper triangular and diagonal) and conditional covariance (lower triangular) based on 20 replicates for 4 score measures measured across 2 time points. The number of educational measures in the experiment was 24 across 4 time points but we randomly selected four measures to give an illustration of the estimated precision matrix and the estimated scaled precision matrix.

The empirical conditional correlations based on 20 classes of students for 4 scores measured across 2 time points are shown in the upper triangular block (italic numbers) of the matrix in Table 3.2 while the conditional covariances are shown in the lower triangular block.

3.2 Preliminaries and notation

In this section, we introduce definitions and notations of dynamic graphs, natural partitions, coloured graphs and Gaussian graphical models for longitudinal or time-course datasets.

Let $G = (V, E)$ be a graph where $V = (v_{jt})_{j \in \Gamma, t \in T}$ is a finite set of vertices and $E \subseteq V \times V$ is a subset of ordered pairs of distinct vertices. Here, $\Gamma = \{1, \ldots, n_\Gamma\}$ is a set of nodes which we call natural vertices and $T = \{1, \ldots, n_T\}$ is an ordered set, typically describing time points. Note that the set Γ is unlabelled, but here described as an ordered set for convenience of notation.

Definition 3.2.1 (Dynamic graphs)**.** *A dynamic graph is a pair $G = (V, E)$, where $V = \{v_{ij}\}_{i \in \Gamma, j \in T}$ is a finite set of vertices and $E \subseteq V \times V$ is a set of ordered couples of elements. Γ and T are finite sets.*

The main characteristic of dynamic graphs is that the same vertices are measured accross different time points. In the T-cell example the same 56 genes were measured accross 10 time points and between time point t and $t+1$ the experimental conditions were changed. For a dynamic graph we draw a directed link if $\{(v_{ij}, v_{kl})\} \in E$ and $j < l$, i.e. $v_{ij} \rightarrow v_{kl}$. A undirected link is drawn if $\{(v_{ij}, v_{kl}), (v_{kl}, v_{ij})\} \in E$, i.e. $v_{ij} \leftrightarrow v_{kl}$.

Definition 3.2.2 (Coloured Graph)**.** *A coloured graph $\tilde{G} = (V, E, F)$ is a triplet, where $G = (V, E)$ is a graph and F is a mapping on the links, i.e.*

$$F : E \longrightarrow C,$$

where C is a finite set of colours.

In other words, coloured graphs induce partitions on the graph by F, that is different subsets of E are visualized with different coloures. Since we are interested in analysing specific subpartitions of the vertex set E. We denote the partitions induced by coloured graphs F by $E \prec F$, i.e.:

$$E \prec F = \cup_{c \in C} F^{-1} \cap E,$$

where $E \prec F$ indicates that the partition is induced on E, and E stands for the complete set of links. The mapping can be apllied to subsets of E. In particular we consider specific subsets called natural partitions S_i and N_i. Each partition represents relationships between natural vertices at some time point $t \in T$. Let $\{S_i\}_{i=0}^{n_T-1}$, and $\{N_i\}_{i=0}^{n_T-1}$ be subsets of vertices and links where S_i, N_i are natural partitions such that:

$$S_i = \{\{(v_{jt}, v_{j,t+i}), (v_{j,t+i}, v_{jt})\} | j \in \Gamma, t = 1, \dots, n_T - i\},$$

and

$$N_i = \{\{(v_{jt}, v_{k,t+i}), (v_{k,t+i}, v_{jt})\} | \forall j \neq k \in \Gamma, t = 1, \dots, n_T - i\}.$$

Each of these partitions is interpreted as follows: S_i considers the lag i interactions between the same natural vertices, and N_i is a graph at time lag i. We induce further partitions on S_i and N_i by using the idea of coloured graphs in order to give more consistent interpretations of dynamic graphs. In particular we consider the following mappings: $E \prec F_1$ indicates that all edges in the partition are coloured with the same colour; $E \prec F_T$ indicates that all edges in the partition are coloured with colours which are the same within natural vertices; $E \prec F_\Gamma$ indicates that all edges in the partition are coloured with the same colour within time points and different colours across natural vertices; $E \prec F_{\Gamma T}$ indicates that all edges in the partition are coloured differently across time points and natural vertices. This can be summarize with the following functions:

$$F_1 : E \to C, \forall v_i, v_j \in E, F_1(v_i) = F_1(v_j),$$

$$F_T : E \to C, \forall v_{it,js}, v_{ku,lv} \in E, F_T(v_{it,js}) = F_T(v_{ku,lv}), \text{ if} t = u \text{ and } s = v,$$

$$F_\Gamma : E \to C, \forall v_{it,js}, v_{ku,lv} \in E, F_\Gamma(v_{it,js}) = F_\Gamma(v_{ku,lv}), \text{if } i = k \text{ and } j = l,$$

$$F_{\Gamma T} : E \to C, \text{no restriction.}$$

where we can substitute at E a specific natural partition S_i or N_i for $i = 1, \dots, T-1$.

Let's consider coloured graphs for $S_i \prec F_i$ and $N_i \prec F_j$, where $i, j = F_1, F_T, F_\Gamma, F_{\Gamma T}$ such that further partitions are induced as follows:

$$S_i = \{S_i^m\}_{m=1}^{s_i},$$

and

$$N_i = \{N_i^m\}_{m=1}^{n_i}.$$

For example, $S_i \prec F_T$ induces the following partition of S_i:

$$S_i = \{S_i^1 \cup \ldots \cup S_i^{n_T-i}\},$$

where $S_i^t = \{\{(v_{jt}, v_{j,t+i}), (v_{j,t+i}, v_{jt})\} | j \in \Gamma, t = 1, \dots, n_T - i\}$. Edges belonging to S_i^t are coloured with the $n_T - i$ colours $C = \{C_1, \dots, C_{n_T-i}\}$, respectively. We abuse notation and let $N_i \prec 0$ and $S_i \prec 0$ in order to express that $N_i = \emptyset$ and $S_i = \emptyset$, i.e. the graph G does not contain such edges.

In Table 3.3 we show the number of colours for any combination of S_i, N_i and graph colouring. The total number of colours n_C can be calculate from Table 3.3.

Factor	S_0	N_0	S_1	N_1	S_2	N_2
F_1	1	1	1	1	1	1
F_T	n_T	n_T	$n_T - 1$	$n_T - 1$	$n_T - 2$	$n_T - 2$
F_Γ	n_Γ	$\frac{1}{2}n_\Gamma(n_\Gamma - 1)$	n_Γ	$n_\Gamma(n_\Gamma - 1)$	n_Γ	$n_\Gamma(n_\Gamma - 1)$
$F_{\Gamma T}$	$n_\Gamma n_T$	$\frac{1}{2}n_\Gamma(n_\Gamma - 1)n_T$	$n_\Gamma(n_T - 1)$	$n_\Gamma(n_\Gamma - 1)$ $\times(n_T - 1)$	$n_\Gamma(n_T - 2)$	$n_\Gamma(n_\Gamma - 1)$ $\times(n_T - 2)$

Table 3.3. Number of colours for any combination of S_i, N_i and graph colouring. S_i, N_i represent natural partitions of E, where E is a set of links. The natural partitions are sub partitions of the set E.

Figure 3.1 shows an example of "coloured graph" where vertices $(v_{ij})_{i\in\Gamma, j\in T}$ are all of the same colour and edges with the same line styles are of the same colours. The graph resembles the following model:

$$[S_0 \prec F_1, N_0 \prec F_T, S_1 \prec F_1, N_1 \prec 0], \qquad (3.1)$$

where, firstly the following natural partitions (sub-partitions) are created:

$$\begin{aligned} S_0 &= \{v_{11}, v_{12}, \ldots, v_{pT}\}, \\ S_1 &= \{(v_{11}, v_{12}), (v_{21}, v_{22}), (v_{31}, v_{32}), (v_{41}, v_{42})\}, \\ N_0 &= \{(v_{11}, v_{21}), (v_{21}, v_{41}), (v_{41}, v_{31}), (v_{31}, v_{11}), (v_{11}, v_{41}), (v_{21}, v_{31}), \\ &= (v_{12}, v_{22}), (v_{22}, v_{42}), (v_{42}, v_{32}), (v_{32}, v_{12}), (v_{12}, v_{42}), (v_{22}, v_{32})\}. \end{aligned}$$

Note that if a couple (v_{ij}, v_{kl}) is present then the couple (v_{ij}, v_{kl}) is present too. We have omitted the symmetric couples from the sets to simplify the notation. Secondly, $S_0 \prec F_1$ induces the following sub-set:

$$S_0^1 = \{v_{11}, v_{12}, \ldots, v_{pT}\},$$

so that a colour is created. Then, $N_0 \prec F_T$ brings the following two sub-partitions:

$$\begin{aligned} N_0^1 &= \{(v_{11}, v_{21}), (v_{21}, v_{41}), (v_{41}, v_{31}), (v_{31}, v_{11}), (v_{11}, v_{41}), (v_{21}, v_{31})\}, \\ N_0^2 &= \{(v_{12}, v_{22}), (v_{22}, v_{42}), (v_{42}, v_{32}), (v_{32}, v_{12}), (v_{12}, v_{42}), (v_{22}, v_{32})\}, \end{aligned}$$

so that two colours are created.

Let $\mathbf{Y} = (Y_{v_{ij}})_{v_{ij}\in V} \in \mathbb{R}^{\Gamma T}$ be a set of random variables. Each vertices $V = (v_{ij})_{i\in\Gamma, j\in T}$ is related to a random variable $Y_{v_{ij}}$.

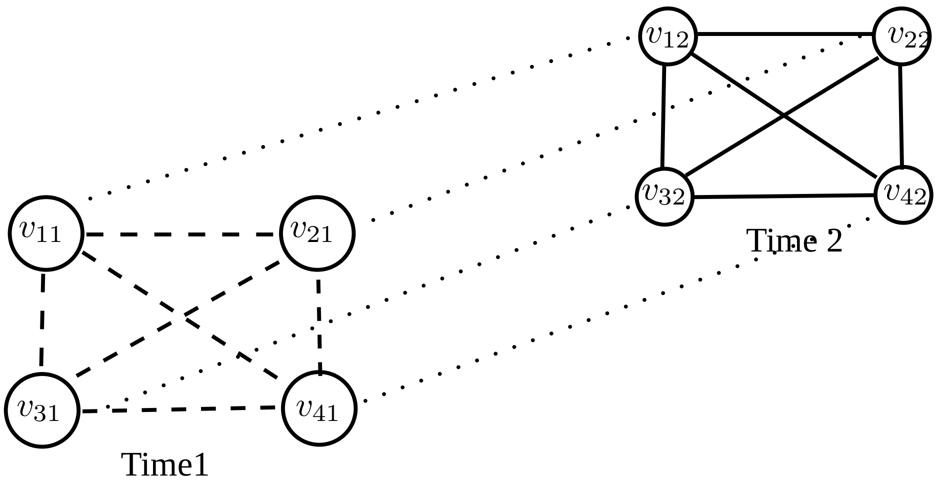


Figure 3.1. Example of coloured graph with four natural vertices and two time points. The natural partitions N_0, S_1 and N_1 are represented. N_0 is represented by discontinuous and continuous links. S_1 is represented by points which connect the same natural vertices. N_1 is assumed to be empty set so no link is represented for the networks at lag 1. Moreover, $N_0 \prec F_T$ is used to partition the natural partition N_0 such that these two natural partitions are represented by two colours. The two colours are represented in the graph by continuous and discontinuous lines, respectively. $S_1 \prec F_T$ produce one colour which is represented by dot line in the graph.

Definition 3.2.3 (Dynamic graphical models)**.** *A graphical model $M = (G, \mathbb{P})$ is a couple G and $\mathbb{P}$, where G is a dynamic graph and $\mathbb{P}$ is a probability distribution on $\mathbf{Y}$ satisfying some Markovian properties, i.e. set of conditional independence relations encoded by the (un)directed edges E.*

In order to connet the idea of coloured graphs and graphical models we give the following definition:

Definition 3.2.4 (Factorial dynamic graphical models)**.** *A factorial graphical model $M = (G, \mathbb{P}, F)$ is a triplet $(G, \mathbb{P}, F)$, where G is a dynamic graph, $\mathbb{P}$ is a probability distribution and F is a mapping on the natural partitions S_i and N_i, for $i = 1, \ldots, T-1$ and T is the total number of time points.*

Assume that $\mathbf{Y} \sim N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ follows a multivariate normal distribution then we have a factorial Gaussian graphical models where pairwise conditional independences are equivalent to zeros in the concentration matrix $\mathbf{\Theta} = \mathbf{\Sigma}^{-1}$, i.e:

$$Y_{v_{ij}} \perp Y_{v_{kl}} | \mathbf{Y}_{V \setminus \{v_{ij}, v_{kl}\}} \Leftrightarrow \theta_{\{ij,kl\}} = 0.$$

The scaled off diagonal elements $\omega_{ij,kl|V\setminus\{ij,kl\}} = -\frac{\theta_{ij,kl}}{\sqrt{\theta_{ij,ij}\theta_{kl,kl}}}$ are the negative of conditional correlation coefficients, where $i,k \in \Gamma$ and $j,l \in T$ for $i \neq j$ and $k \neq l$. For future use, we denote $\mathbf{\Omega} = (\omega_{ij,kl|V\setminus\{ij,kl\}})$ be a matrix of scaled elements of $\mathbf{\Theta}$.

For example, the factorial Graphical model with graph represented in Figure 3.1, $\mathbf{Y} = (Y_{11}, Y_{12}, \dots, Y_{42})$ vector of random variables which correspond to vertices $\{v_{11}, v_{12}, \dots, v_{42}\}$ and relations $[S_0 \prec F_1, N_0 \prec F_T, S_1 \prec F_1, N_1 \prec 0]$ given in (3.1) imply the following precision matrix:

$$\mathbf{\Theta} = \left[\begin{array}{ccc|ccc} \theta_1 & \theta_2 & \theta_2 & \theta_3 & 0 & 0 \\ \theta_2 & \theta_1 & \theta_2 & 0 & \theta_3 & 0 \\ \theta_2 & \theta_2 & \theta_1 & 0 & 0 & \theta_3 \\ \hline \theta_3 & 0 & 0 & \theta_1 & \theta_4 & \theta_4 \\ 0 & \theta_3 & 0 & \theta_4 & \theta_1 & \theta_4 \\ 0 & 0 & \theta_3 & \theta_4 & \theta_4 & \theta_1 \end{array}\right].$$

3.3 Sparse Gaussian graphical models for coloured graphs

We have described the idea of coloured graphs which allows us to create subpartitions of natural partitions. Moreover, equality constraints on conditional correlations can be imposed such that edges with the same colours imply these equality restrictions. Here, we define a set of design matrices $\mathbf{X}$ which is useful to directly connect elements of a coloured graph with concentration or conditional correlation matrix parameters.

Let's consider a coloured graph which defines partitions on E, i.e. $\{S_i^m\}$ and , $\{N_i^m\}$, then two sets of design matrices $\mathbf{X}^S = \{\mathbf{X}^{S_i^m}\}_{i=0,m=1}^{n_{\Gamma-1},s_i}$ and $\mathbf{X}^N = \{\mathbf{X}^{N_i^m}\}_{i=0,m=1}^{n_{\Gamma-1},n_i}$ where $\mathbf{X}^{S_i^m}, \mathbf{X}^{N_i^m} \in \mathbb{R}^{\Gamma T}$ and $\mathbf{X}^{S_i^m}$ can be uniquely identified such that:

$$x_{jt,gs}^{S_i^m} = \begin{cases} 1 & \text{if } (v_{jt}, v_{gs}) \in S_i^m \\ 0 & \text{otherwise} \end{cases}$$

and,

$$x_{jt,gs}^{N_i^m} = \begin{cases} 1 & \text{if } (v_{jt}, v_{gs}) \in N_i^m \\ 0 & \text{otherwise} \end{cases}$$

We re-define the set of design matrices as:

$$\mathbf{X} = \{\mathbf{X}^S, \mathbf{X}^N\} = \{\mathbf{X}_1, \dots, \mathbf{X}_{n_c}\},$$

where $\mathbf{X}^S = \sum_{i=1}^{n_S} \mathbf{X}^{S_i^m}$, $\mathbf{X}_i^N = \sum_{i=1}^{n_N} \mathbf{X}^{N_i^m}$, $n_c = n_S + n_N$ is the total number of colours, n_S is the total number of colours for the self-self partitions, and n_N is the total numebr of colours for the network partitions.

3.3.1 Modelling the concentration matrix with graph colouring.

The design matrix $\mathbf{X}$ can be used to induce the following parametrization on $\mathbf{\Theta}$, i.e.

$$\mathbf{\Theta} = \sum_{m=1}^{n_C} \mathbf{X}^{(m)} \theta_m.$$

where $\boldsymbol{\theta} = (\theta_m)_{m=1}^{n_C}$ is a vector of unknown parameters. Because $\mathbf{X}_m$ are induced by the graph the specific elements of the concentration matrix which correspond to edges with the same colours are constraint to be equal. Note that the restrictions so defined are linear in the concentration matrix.

Example 1: Human T-cell. Consider the following factorial graphical model for human T-cell microarray data (see Section 3.1 for a description):

$$[S_0 \prec F_1, N_0 \prec F_1, S_1 \prec F_1, N_1 \prec 0],$$

then the design matrices $\mathbf{X} = \{\mathbf{X}^S, \mathbf{X}^N\} = \{\mathbf{X}_1, \mathbf{X}_2, \mathbf{X}_3\}$ are:

$$\mathbf{X}^{S_0^1} = \mathbf{I}, \mathbf{X}^{S_1^1} = \begin{pmatrix} \mathbf{0} & \mathbf{I} \\ \mathbf{I} & \mathbf{0} \end{pmatrix}, \mathbf{X}^{N_0^1} = \begin{pmatrix} \mathbf{D} & \mathbf{0} \\ \mathbf{0} & \mathbf{D} \end{pmatrix}, \mathbf{X}^{N_1^1} = \begin{pmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{pmatrix}$$

where $\mathbf{D}$ is a square matrix with 1 off-diagonal and 0 on the diagonal. Note that $\mathbf{X}^{N_1^1}$ is an empty matrix since S_1^1 is an empty set which implies 0 colours. Table 3.4 shows the estimated concentration matrix based on 44 replicates for 4 genes measure across 2 time points. The model induces $n_C = 3$ colours. For the estimation procedure see Højsgaard and Lauritzen (2008). For a faster and more general estimation procedure see the algorithm in subsection 3.3.3.

3.3.2 Modelling the conditional correlation matrix with graph colouring.

It is generally important that all variables are on comparable scales if a structured model on the concentration matrix is considered so that conclusions are interpretable. In contrast, model based on the conditional correlation matrix have proprieties of invariance under rescaling as showed in Lemma 3.3.1.

Time	*1*				*2*			
Gene	*ZNF*	*CCN*	*SIV*	*SCY*	*ZNF*	*CCN*	*SIV*	*SCY*
ZNF	1.11	0.01	0.01	0.01	-0.42	0	0	0
CCN	0.01	1.11	0.01	0.01	0	-0.42	0	0
SIV	0.01	0.01	1.11	0.01	0	0	-0.42	0
SCY	0.01	0.01	0.01	1.11	0	0	0	-0.42
ZNF	-0.42	0	0	0	1.11	0.01	0.01	0.01
CCN	0	-0.42	0	0	0.01	1.11	0.01	0.01
SIV	0	0	-0.42	0	0.01	0.01	1.11	0.01
SCY	0	0	0	-0.42	0.01	0.01	0.01	1.11

Table 3.4. Estimated conditional covariance based on 44 replicates for 4 genes measured across 2 time points. The number of parameter estimated is 3, that is the number of colour in the graph is 3. Note that the following model $S_0 \prec F_1, N_0 \prec F_1, S_1 \prec F_1$ with the rest $S_i, N_i \prec 0$ has been imposed.

Lemma 3.3.1. (Invariance) *The inverse variance matrix* $\mathbf{\Theta}$ *is not invariant under rescaling* $\mathbf{Y}$ *by* $\mathbf{D}$. *Let* $\mathbf{D}$ *be a diagonal matrix with diagonal entries the scaled precision matrix* $\mathbf{\Omega}$ *is invariant.*

Proof. The inverse variance matrix $\mathbf{\Theta}^*$ of $\mathbf{Y}^* = \mathbf{DY}$ is given by

$$\mathbf{\Theta}^* = (\mathbf{D\Sigma D})^{-1} = \mathbf{D}^{-1}\mathbf{\Sigma}^{-1}\mathbf{D}^{-1} = \mathbf{D}^{-1}\mathbf{\Theta}\mathbf{D}^{-1},$$

so

$$\mathbf{\Theta} = \mathrm{var}(\mathbf{Y})^{-1} \neq \mathrm{var}(\mathbf{Y}^*)^{-1} = \mathbf{\Theta}^*,$$

so $\mathbf{\Theta}$ is not invariant under rescaling of $\mathbf{Y}$. Now, we will proof that $\mathbf{\Omega}$ is invariant under rescaling, i.e $\mathbf{\Omega} = \mathbf{\Omega}^*$, where

$$\mathbf{\Omega}^* = \mathbf{\Sigma}_0^{*-\frac{1}{2}}\mathbf{\Theta}^*\mathbf{\Sigma}_0^{*-\frac{1}{2}} = \mathbf{\Sigma}_0^{*-\frac{1}{2}}\mathbf{D}^{-1}\mathbf{\Theta}\mathbf{D}^{-1}\mathbf{\Sigma}_0^{*-\frac{1}{2}}\mathbf{\Sigma}_0 = \mathbf{\Sigma}_0^{-\frac{1}{2}}\mathbf{\Theta}\mathbf{\Sigma}_0^{-\frac{1}{2}} = \mathbf{\Omega},$$

and $\mathbf{\Sigma}_0^* = \mathbf{\Sigma}_0\mathbf{D}^{-1}$. □

A factorial graphical model with structured conditional correlation matrix is obtained by restricting elements of $\mathbf{\Omega}$ such that:

- all diagonal elements of $\mathbf{\Omega}$ (inverse partial variances) must be identical, and
- all partial correlations corresponding to edges in the same colour class must be identical.

Let $\mathbf{\Theta} = \mathbf{\Sigma}_0 \mathbf{\Omega} \mathbf{\Sigma}_0$ be the concentration matrix, where $\mathbf{\Sigma}_0 = \sum_{m=1}^{s_0} \mathbf{X}^{S_0^m} \theta^m$, and

$$\mathbf{\Omega} = \mathbf{I} + \sum_{i=1}^{n_\Gamma - 1} \sum_{m=1}^{s_i} \mathbf{X}^{S_i^m} \theta_m + \sum_{i=0}^{n_\Gamma - 1} \sum_{l=m+1}^{n_c} \mathbf{X}^{N_i^l} \theta_l .$$

A structured graphical model for the conditional correlation matrix is obtained by partitioning $\mathbf{\Omega}$, i.e.:

$$\mathbf{\Omega} = \sum_{m=1}^{n_C} \mathbf{X}^{(m)} \omega_m .$$

Elements of the conditional correlation matrix $\mathbf{\Omega}$ are related to elements of the natural partitions S_i, N_i. Note that edges with the same colours correspond to specific elements of the conditional correlation matrix which are constrained to be equal. Restrictions are non linear in the conditional correlation matrix and an iterative algorithm is considered in subsection 3.3.3 to estimate $\mathbf{\Theta}$ such that specific elements in the conditional correlation matrix $\mathbf{\Omega}$ are constrained to be equal.

Example 2: Educational study. Consider the following factorial graphical model for educational study dataset (see Section 3.1 for the description):

$$[S_0 \prec F_{\Gamma T}, N_0 \prec F_\Gamma, S_1 \prec F_1, N_1 \prec 0].$$

Table 3.4 shows the estimated conditional concentration elements (upper triangular) and conditional covariance elements (lower triangular and diagonal). Note that conditional correlation elements are constraint to be equal while concentration elements are different.

ℓ_1-norm. We have reached some important results by considering constraints on the concentration (or conditional correlation) matrix. The number of parameters to be estimated can be considerable reduced. Each sub-network can be interpreted as its corresponding natural partition. However, dynamic genetic graphs are usually sparse which means that few vertices will be connected. We have given the concepts of sparse and dense graphs in Section 2.3 when the number of nodes tends to infinity. A measure of the density of a graph in the finite case can be defined as follow. For undirected graphs, density is:

$$\rho = \frac{2|\Gamma T|}{|\Gamma T|\,(|\Gamma T| - 1)} .$$

The maximum number of edges is $\frac{1}{2}|\Gamma T|(|\Gamma T| - 1)$, so the maximal density is 1 (for complete graphs) and the minimal density is 0.

Time		*1*				*2*			
	Subject	*Contr*	*Auton*	*Infl*	*Prox*	*Contr*	*Auton*	*Infl*	*Prox*
1	*Contr*	1.32	*0.13*	*0.21*	*-0.14*	*0.34*	*0*	*0*	*0*
	Auton	-0.15	1.08	*-0.07*	*-0.07*	*0*	*0.34*	*0*	*0*
	Infl	-0.26	0.08	1.18	*-0.07*	*0*	*0*	*0.34*	*0*
	Prox	0.17	0.08	0.08	1.13	*0*	*0*	*0*	*0.34*
2	*Contr*	-0.43	0	0	0	1.22	*0.13*	*0.21*	*-0.14*
	Auton	0	-0.37	0	0	-0.15	1.12	*-0.07*	*-0.07*
	Infl	0	0	-0.40	0	-0.25	0.08	1.18	*-0.07*
	Prox	0	0	0	-0.38	0.16	0.08	0.08	1.14

Table 3.5. Estimated conditional correlations (upper triangular) and conditional covariance (lower triangular and diagonal). The number of parameter estimated is 8 for the diagonal elements (natural partions S_0), 6 for the natural partition N_0 and 4 for the natural partition S_1. The total number of estimated parameter is 16, that is the number of colours in the graph is 16. Note that the following model $S_0 \prec F_{\Gamma T}, N_0 \prec F_{\Gamma}, S_1 \prec F_1$ with the rest $S_i, N_i \prec 0$ has been imposed.

We could think to estimate a complete graph and to produce multiple hypothesis testing on the edges of the graph. However, model selection and parameter estimation would be done separately in this case and it would bring at instability of the model Breiman (1996). Alternatively, we consider ℓ_1-norm penalty on the concentration (or conditional correlation) matrix to induce sparsity. The choice of a ℓ_1-norm can be considered the unique one since other ℓ_p-norm, where p is typically in the range $[0,2]$, are not suitable in high-dimensional data analysis. The estimates being exactly zero for $p \leq 1$ only, while the optimization problem is convex for $p \geq 1$. Hence ℓ_1-norm occupies a unique position, as $k = 1$ is the only value of k for which variable selection takes place while the optimization problem is still convex and hence feasible for high-dimensional problems Banerjee *et al.* (2008).

3.3.3 Penalized likelihood for coloured graphs

Consider a coloured graph that specifies a partition of $\boldsymbol{\Theta}$, then a set of design matrices $\mathbf{X}$ is used to produce linear operators which are necessary to impose linear restrictions on $\boldsymbol{\Theta}$. Every design matrix $\mathbf{X}^{(m)}$, $m = 1, \ldots, n_c$ consists of zeroes and ones and induce $p-1$ linear constraints on $\boldsymbol{\Theta}$ when the number of 1 in $\mathbf{X}^{(m)}$ is p. We define a linear map $\mathbf{A} = (\mathbf{A}_1, \ldots, \mathbf{A}_{n_p})$ where each $\mathbf{X}^{(m)}$ induces a set of matrices $\mathbf{A}_1, \ldots, \mathbf{A}_{n_m}$ and an element in $\mathbf{A}^{(i)}$, $i = 1, \ldots, n_p$ assumes value -1, 0 or 1 as described below:

$$\mathbf{a}^{(i)}_{jt,gs} = \begin{cases} 1 & \text{if } \sum_{jg}\sum_{ts}\mathbf{X}^{(i)}_{jt,gs} \text{ before } jt,gs = p-1 \\ -1 & \text{if } \sum_{jg}\sum_{ts}\mathbf{X}^{(i)}_{jt,gs} \text{ before } jt,gs = p \\ 0 & \text{otherwise,} \end{cases}$$

where ‘before’ is meant with respect to the total row-major ordering of the matrices. Now let n_p be the total number of linear constraints, then $\mathbf{A} = \{\mathbf{A}_1, \dots, \mathbf{A}_{n_p}\}$ and the linear map is expressed as $\mathbf{A} : \mathbb{R}^{\Gamma T}_{\text{sym}} \to \mathbb{R}^{n_p}$ with:

$$\mathbf{A}(\mathbf{\Theta}) = [\langle \mathbf{A}_1, \mathbf{\Theta}\rangle, \dots, \langle \mathbf{A}_{n_p}, \mathbf{\Theta}\rangle]$$

where $\langle , \rangle$ is the Frobenius inner product.

When we derive the objective function we want to take into account the sparsity assumption, i.e. the sum of the absolute values of the precision matrix should be less than ρ. The ℓ_1-norm constraint $||\mathbf{\Theta}|| \leq \rho$ can be written as a set of linear equality constraints by introducing slack variables $\mathbf{x}^+, \mathbf{x}^- \in \mathbb{R}^k$, i.e.

$$\begin{aligned} (\hat{\mathbf{\Theta}}) \quad &:= \quad \underset{\mathbf{\Theta}}{\operatorname{argmin}}\{-l(\mathbf{\Theta}) + \lambda\mathbf{x}^+ + \lambda\mathbf{x}^-\} \qquad (3.2) \\ \text{subject to} \quad & \quad \mathbf{B}(\mathbf{\Theta}) - \mathbf{x}^+ + \mathbf{x}^- = \mathbf{0} \\ & \quad \mathbf{\Theta} \succeq 0, \mathbf{x}^+, \mathbf{x}^- \geq 0. \end{aligned}$$

where $\mathbf{B}_i$, for $i = 1, \dots k$, are symmetric matrices with just element $b_{g,t} = b_{t,g} = 1$ and the rest equal to zero, and $k = |\Gamma T|(|\Gamma T| + 1)/2$ or $k = |\Gamma T|(|\Gamma T| - 1)/2$ if diagonal elements are penalized or not, respectively.

Now we want to include the set of linear constraints $\mathbf{A}(\mathbf{\Theta})$ which are derived by imposing a specific coloured graph. To achieve sparse graph structures with a structured a priori graph one can minimize a convex log-likelihood function, i.e.:

$$\begin{aligned} (\hat{\mathbf{\Theta}}) \quad &:= \quad \underset{\mathbf{\Theta}}{\operatorname{argmin}}\{-l(\mathbf{\Theta}) + \lambda\mathbf{x}^+ + \lambda\mathbf{x}^-\} \qquad (3.3) \\ \text{subject to} \quad & \quad A(\mathbf{\Theta}) = \mathbf{0} \\ & \quad \mathbf{B}(\mathbf{\Theta}) - \mathbf{x}^+ + \mathbf{x}^- = \mathbf{0} \\ & \quad \mathbf{\Theta} \succeq 0, \mathbf{x}^+, \mathbf{x}^- \geq 0. \end{aligned}$$

We have re-written the convex optimization problem in the standard form. This is a quadratic semi-definite log-determinant programming problem which allows to impose factorial graphical models.

The non linearity of the objective function and the positive definiteness of the constraint make the optimization problem (3.3) not trivial. We use an algorithm called LogDetPPA to find a solution of (3.3). LogDetPPA employs the essential ideas of the proximal point algorithm (PPA), the Newton method and the preconditioned conjugate gradient solver (Wang *et al.*, 2009).

Note that LogDetPPA algorithm was developed into Matlab and it gives the opportunity to solve convex optimization problems with linear constraints which need to be implemented. We implemented linear constraints for factorial graphical models as described in Section 3.2. Moreover, it is possible to use function of Matlab within R. In fact, the package `R.Matlab` allows to connect Matlab and R. R is more suitable for statistical analysis and it is an open source software so source codes of packages are available. We took advantage from `R.Matlab` to create a virtual connection between R and Matlab so that we are able to solve the constraint optimization problem within R.

LogdetPPA optimization of SGL concentration model. For SGL$_\Theta$ problems, we can apply the algorithm logdetPPA after having create the linear operators $\mathbf{A}$ and $\mathbf{B}$. Then, $\boldsymbol{\Theta}$ is the matrix that minimizes the penalized log-likelihood (3.3) among the space of all symmetric $\Gamma T \times \Gamma T$ matrices for whom the non linear restriction on $\boldsymbol{\Theta}$ holds. For example we used SGL$_\Theta$ to estimate the matrix represented in Table 3.4.

LogdetPPA optimization of SGL conditional correlation model. This algorithm allows to introduce linear constraint but in case we are modelling conditional correlations the constraints are not linear. However, we overcome this problem by using an iterative algorithm which make use of logdetPPA after having found an initial guess for diag($\boldsymbol{\Theta}$). The pseudo-code is described in Algorithm 3.1. Usually, the convergence is reached after few iterations (4-10). By taking a starting point $\boldsymbol{\Sigma}_0$ the optimization problem (3.3) is a convex optimization problem, i.e., the objective function is convex on $\boldsymbol{\Theta}$, and the feasible region is convex.

3.4 Model selection and stability selection

We have seen that estimation of $\boldsymbol{\Theta}$ considering different coloured graphs given a smoothing parameter λ is possible inside a convex optimization framework. We have also proposed several factorial graphical models. In this subsection we address same issues on how to choose the 'best' coloured graph, and what should be a good compromise between a sparse and a dense graph. In particular, according to Meinshausen and Bühlmann (2010) we want to find a smoothing parameter such that the expected number of false positive links is taken under control. This aim

```
Algorithm 3.1: Calculate sparse $\mathbf{\Theta}$ with structured on $\mathbf{\Omega}$
Require: Coloured graph $S_i \prec *, N_i \prec *$ and set an initial vector $\mathbf{\Sigma}_0$.
    1. Find the linear maps $\mathbf{A}_1, \ldots, \mathbf{A}_m$.
    2. $k = 0, \ldots,$.
    3. Estimate $\mathbf{\Theta}^{(k)}$.
    4. Set $\mathbf{\Sigma}_0 = \text{diag}(\mathbf{\Theta}^{(k)})$.
    5. Replay $\mathbf{\Sigma}_0^{(k)}$ with $\mathbf{\Sigma}_0^{(k+1)}$ and estimate $\hat{\mathbf{\Theta}}^{(k+1)}$.
    6. If $||\hat{\mathbf{\Theta}}^{(k+1)} - \hat{\mathbf{\Theta}}^{(k)}||_1 < \varepsilon$ Stop
       else $k = k + 1$, go to 3
       end.
```

can be reached through stability selection. Stability selection is similar to bootstrap idea which consists of a re-sampling procedure. For example we used SGL_Ω to estimate the matrix represented in Table 3.5.

3.4.1 Model selection.

Let's assume that the smoothing parameter is fixed (point-wise control) $\lambda = \lambda_{opt}$ and two coloured graphs need to be compared:

$$[S_0 \prec F_{\Gamma T}, N_0 \prec F_\Gamma, S_1 \prec F_1, N_1 \prec 0],$$

and

$$[S_0 \prec F_{\Gamma T}, N_0 \prec F_\Gamma, S_1 \prec F_\Gamma, N_1 \prec 0].$$

An information criterion such as AIC, BIC and AICc can be used to compare different factorial graphical models. AIC typically will select more and more complex models as the sample size increases, because the maximum log-likelihood increases linearly with n while the penalty term for complexity is proportional to the number of degrees of freedom. Note that for factorial graphical models the number of degrees of freedom is approximated by the number of "free" parameters different from zero, i.e. the estimated elements different from zero which do not belong to the same partition. AICc penalizes complexity more strongly than AIC, with less chance of over-fitting the model. BIC is constructed in a manner quite similar to AIC with stronger penalty for complexity (Claeskens and Hjort, 2008).

3.4.2 Stability selection.

Usually, the choice of smoothing parameter λ is crucial as it leads to the structure of the network. In particular, if $\lambda = 0$ and there are no constraints, the maximum likelihood estimator is $\hat{\mathbf{\Theta}} = \mathbf{S}^{-1}$, if $\lambda \to \infty$, $\hat{\mathbf{\Theta}}$ is diagonal which means all random variables are independent. Generalized cross validation can be used to select the tuning parameter $\lambda = \lambda_{cv}$ but Leng *et al.* (2006) showed that given λ_{cv} the estimator of $\mathbf{\Theta}$ is not consistent in terms of variables selection. Alternatively, bootstrap (Breiman, 1999) can be considered to estimate empirical distribution of each elemtent of $\hat{\mathbf{\Theta}}$. We prefer stability selection (Meinshausen and Bühlmann, 2010) because the expected number of links falsely estimated is controlled and variables selection is consistent. Moreover, the choice of a smoothing parameter λ becomes less important Meinshausen and Bühlmann (2010). We adapt stability selection to factorial graphical models.

Let's consider a vector of smoothing parameters such that $\lambda \in \Lambda \subseteq \mathbb{R}^+$ that determines the amount of regularization.

Theorem 3.4.1. *A necessary and sufficient condition for $\theta_{ij,kl} = 0$ for all $i,k \in \Gamma$ and $j,l \in T$ is that $|S_{ij,kl}| \leq \lambda$ for all $i \neq k$ and $l \neq j$ Mazumder and Hastie (2011).*

Upper and lower bounds of λ, u_λ and l_λ respectively, are calculated such that $u_\lambda = max(|S_{ij,kl}|)$ and $l_\lambda = min(|S_{ij,kl}|)$ for $i \neq k$ $l \neq j$, where $i,k \in \Gamma$ and $j,l \in T$. Then, for all $\lambda > u_\lambda$ an empty graph is estimated while for $\lambda < l_\lambda$ a fully connected graph is estimated. We search the solution of the optimization problem (3.3) for value of λ into the range $[l_\lambda, u_\lambda]$. The "optimal" value of $\lambda = \lambda_{opt}$ can be chosen by minimizing a score which measures the goodness-of-fit.

Let's suppose a graph $\hat{G} = (V, \hat{E})$ has been inferred, where

$$\hat{E}_{\lambda_{opt}} = \{(v_i, v_j) : \hat{\theta}_{i,j} \neq 0\}$$

is the estimated edge set and,

$$E = \{(v_i, v_j) : \theta_{i,j} \neq 0\}$$

denotes the active set. Let $I = \{1, \ldots, n\}$ be the index set for sample $\mathbf{y}^{(i)}$, $i \in I$, then: The set of stable edges is indicated as:

$$\hat{E}_{stable} = \{(v_i, v_j) : \hat{\Pi}^{\lambda}_{ij} \geq \pi_{thr}\},$$

and it depends on λ_{opt} via $\hat{\Pi}^{\lambda_{opt}}_{ij}$. The tuning parameter π_{thr} indicates a threshold and controls the expected number of falsely selected links. Assume that the joint

Algorithm 3.2: Stability selection for graphical models.

Require: n.

- Draw sub-samples of size [n/2] without replacement, denoted by $I^* \subset \{1,\dots,n\}$, where $|I^*| = [n/2]$.
- Run the selection algorithm $\hat{E}_{\lambda_{opt}}(I^*)$ on I^*.
- Do these steps many times and compute the relative frequencies,

$$\hat{\Pi}_{ij}^{\lambda_{opt}} = P^*((v_i, v_j) \in \hat{E}_\lambda), \text{for } \; i, j = 1, \dots, p.$$

distribution of the random variables is exchangeable and $\hat{E}$ is a better choice than a random guessing, then it can be shown that

$$\mathbb{E}(FP) \leq \frac{1}{2\pi_{thr} - 1} \frac{q^2}{k},$$

where k is the dimension of the model (it depends on the factorial model), q is the number of selected variables (e.g. $|\hat{E}|$), and $FP = |E^c \cap \hat{E}_{stable}|$ is the number of falsely positive selected. This is a finite sample control, even if $k >> N$. Choose $\mathbb{E}(FP) \leq v$, then if $q^2 \leq vk$:

$$\pi_{thr} = (1 + q^2/vk)/2,$$

and $\pi_{thr} \in (\frac{1}{2}, 1)$ is bounded.

Figure 3.2 is taken from Meinshausen and Bühlmann (2010) and it illustrates that the choice of a tuning parameter λ is less important with stability selection than cross validation.

3.5 Application

3.5.1 Real data set example T-cell

We have seen that several coloured graphs can be imposed on a graphical model and a model selection procedure is necessary to select the "best" coloured graph. Moreover, a smoothing parameter λ that regulates the sparsity needs to be selected.

Let's consider several coloured graphs for Human T-cell dataset. Information criterion measures, for each of the top ten models, are showed in Table 3.6. We select the first coloured graph since it has the smallest AICc. This model is described

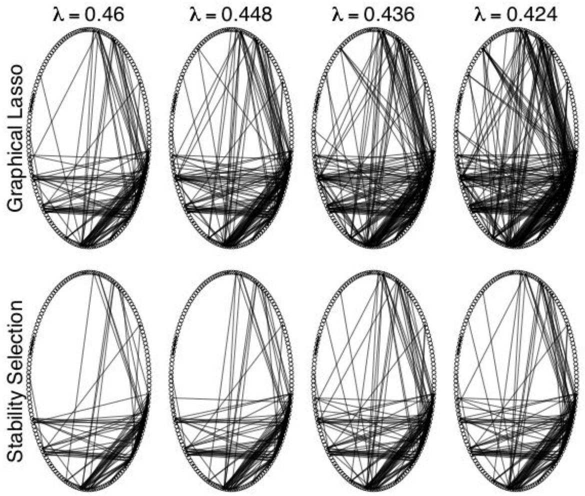


Figure 3.2. Graph selection with cross validation and stability selection procedure.

Model					AIC	AICc
$S_0 \sim F_1$	$N_0 \sim F_\Gamma$	$S_1 \sim F_T$	$N_1 \sim F_\Gamma$	$S_2 \sim F_T$	175.82	178.19
$S_0 \sim F_1$	$N_0 \sim 0$	$S_1 \sim F_\Gamma$	$N_1 \sim F_\Gamma$	$S_2 \sim 0$	175.84	178.42
$S_0 \sim F_1$	$N_0 \sim F_{\Gamma T}$	$S_1 \sim F_\Gamma$	$N_1 \sim F_{\Gamma T}$	$S_2 \sim 0$	176.32	178.60
$S_0 \sim F_1$	$N_0 \sim F_\Gamma$	$S_1 \sim F_1$	$N_1 \sim 0$	$S_2 \sim 0$	176.78	178.93
$S_0 \sim 1$	$N_0 \sim F_\Gamma$	$S_1 \sim F_\Gamma$	$N_1 \sim F_\Gamma$	$S_2 \sim 0$	177.04	179.19
$S_0 \sim 1$	$N_0 \sim F_{\Gamma T}$	$S_1 \sim F_1$	$N_1 \sim F_{\Gamma T}$	$S_2 \sim 0$	176.81	179.81

Table 3.6. Model ordering according to AICc for tcell dataset

in Figure 3.3. According to the coloured graph this scheme summarize the characteristics of the estimated graph. The networks at temporal lag 0 are constrained to be equal across the five observed time points. Moreover, the networks at temporal lag 1 are constrained to be equal across time. There are no links between time t and $t+2$ since we are assuming conditional independence between time t and $t+2$ except for self interactions, i.e. interactions between the same couple of genes. Figure 3.4 shows interactions between genes at lag 0 (left part of the figure), and it shows interactions between genes at lag 1 (right part of the figure).

3.5.2 Simulation study for SGL models

We considered a simulation study to show the performance of the proposed model. Table 3.7 shows the simulation study scheme in which four different scenarios are

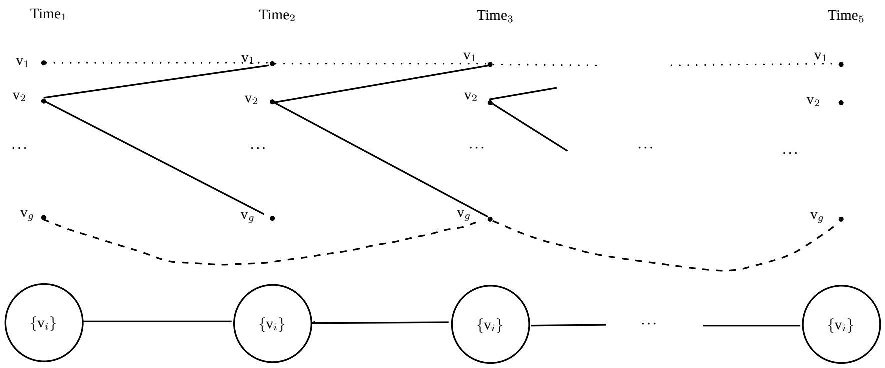


Figure 3.3. Selected model after model selection for T-cell dataset.

studied. Here for different scenarios we mean that the number of nodes, links or time points change while the structure of the networks is the same. We are mainly interested in the performance of our estimator in terms of false positive (FP), false negative (FN), true positive (TP) and true negative (TN). Let $\mathbf{\Theta}$ be the true and $\hat{\mathbf{\Theta}}$ be the estimated precision matrix. These measure summarize:

ID	g	g*	t	p	n
1	20	0	3	60	50
2	-	20	-	120	-
3	-	40	-	180	-
4	-	60	-	240	-

Table 3.7. Simulation study scheme in which four scenarios are represented. The first column is an identification number, the second one indicates the number of variables per each time point (third column). The number of independent samples are represented in the last column.

- the percentage of links that are falsely estimated, i.e. an element in $\mathbf{\Theta}$ is zero but is not zero in $\hat{\mathbf{\Theta}}$ (FP) and an element in $\mathbf{\Theta}$ is not zero but is zero in $\hat{\mathbf{\Theta}}$ (FN),

- the percentage of links that are positively estimated, i.e. an element in $\mathbf{\Theta}$ is not zero and it is not zero in $\hat{\mathbf{\Theta}}$ (TP) and an element in $\mathbf{\Theta}$ is zero and it is zero in $\hat{\mathbf{\Theta}}$ (TN).

Other measure like these are the false discovery and false not discovered.

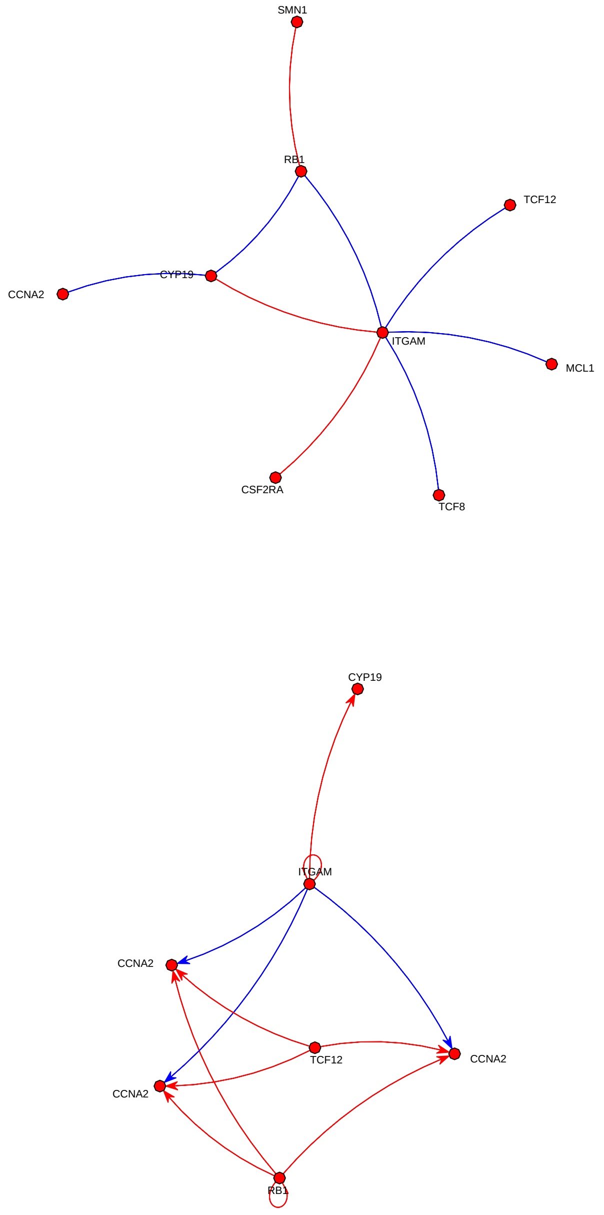


Figure 3.4. Representation of interactions between genes at temporal lag 0. Note that networks at lag 0 at time $1,2,3,\ldots,5$ are equal since we impose $N_0 \sim F_\Gamma$(top). Representation of interaction between genes at temporal lag 1. Note that networks at lag 1 between time $(1,2),(2,3),(3,4),(4,5)$ are equal since we impose $N_1 \sim F_\Gamma$(botton).

Whereas, we are not looking at the performance of our estimator in terms of distances between the ”true” parameter $\boldsymbol{\Theta}$ and the estimated ones $\hat{\boldsymbol{\Theta}}$. Another element of interest is the *sign* of the estimated conditional covariance. In fact, this element is not zero, if $-sign(\theta_{ij})$ is positive this indicates a positive dependence whereas if $-sign(\theta_{ij})$ is negative it indicates a negative dependence.

For each scenario we simulate 100 datasets from a multivariate normal distribution with $\boldsymbol{\mu}$ equal to zero and $\boldsymbol{\Sigma}$ equal to the inverse of a precision matrices $\boldsymbol{\Theta}$ where the coloured graph is simulated from the following model:

$$[S_0 \sim F_{\Gamma T}, N_0 \sim F_{\Gamma}, S_1 \sim F_1],$$

while the rest is zero. Note that we keep the true network constant but we increase the number of nodes in the graph. Random variables associated with these added nodes are independent. We keep the number of replicates and time points constants. The number of replicates is fewer than the number of random variables. Table 3.8 shows the average over 100 of measures that is $\bar{FP} = \sum_{i=1}^{100} FP_i$, $\bar{FD} = \sum_{i=1}^{100} FD_i$ and so on. Moreover we show that AICc performs better on average and other graphical lasso such us proposed by Tibshirani (1996) does not perform well in case of structured dynamic graphical models (see Table 3.9).

		$\bar{FP}$	$\bar{FN}$	$\bar{FD}$	$\bar{FnD}$
1	AICc	0.0092	0.0811	0.2000	0.0031
	BIC	0.0363	0.0139	0.4873	0.0005
	AIC	0.0698	0.0069	0.6470	0.0003
2	AICc	0.0057	0.0447	0.2899	0.0006
	BIC	0.0088	0.0321	0.3826	0.0005
	AIC	0.0437	0.0041	0.7514	0.0001
3	AICc	0.0016	0.4585	0.2730	0.0036
	BIC	0.0016	0.4585	0.2730	0.0036
	AIC	0.0288	0.1452	0.8088	0.0012
4	AICc	0.0091	0.1034	0.1680	0.0052
	BIC	0.0396	0.0517	0.4527	0.0027
	AIC	0.0670	0.0000	0.5704	0.0000

Table 3.8. The average of the proportions of how many links have been correctly estimated were calculated by the False Positive (FP), False Negative (FN), False Discovery (FD) and False not Discovery (FnD).

		$\bar{FP}$	$\bar{FN}$	$\bar{FD}$	$\bar{FnD}$
1	glasso	0.002	0.977	0.667	0.035
	SGL	0.006	0	0.137	0
2	glasso	0.001	0.964	0.620	0.013
	SGL	0.005	0	0.263	0
3	glasso	0.001	0.988	0.906	0.008
	SGL	0.001	0.458	0.273	0.004

Table 3.9. Average performs for neighbourhood selection model, graphical lasso and structured graphical lasso.

3.6 Summary

As more and more large datasets become available, the need for efficient tools to analyse such data has become imperative. In this chapter, we have considered sparse dynamic Gaussian graphical models with ℓ_1-norm penalty. This type of modelling offers a straightforward interpretation, i.e. the edges of the graph defining the partial conditional correlations among the nodes. In particular, under the sparsity assumption, a large part of the precision matrix can be filled with zeros a priori. Based on the consideration of dynamic and model-oriented definitions, we are able to reduce the number of parameters to be estimated. We have shown that SGL_Θ proved to be powerful on both simulated and real data analysis.

Chapter 4

Copula Gaussian graphical models

Most of the research efforts in the graphical models literature have been focused on multivariate normal models or on log-linear models; see, for example, the monograph of Lauritzen (1996). These models relate to datasets that contain exclusively continuous or categorical variables. CG distributions (Lauritzen, 1996) constitute the basis of a class of graphical models for mixed variables, but they impose an overly restrictive assumption; i.e. the conditional distribution of the continuous variables given the discrete variables must be multivariate normal. As such, the three main classes of graphical models are too restrictive to be widely applicable.

Since multivariate datasets typically contain variables of many types, our goal is to consider approaches to graphical model determination that are broad enough to be applicable to any study that involves a mixture of binary, ordinal, count and continuous variables. Moreover, we want to develop graphical models for estimating dynamic networks given some a priori structures.

Copulas (Nelsen, 2006) provide the theoretical framework in which multivariate associations can be modelled separately from the univariate distributions of the observed variables. The seminal work of Sklar (1959) that formally introduced the notion of copula provides the theoretical framework in which a joint probability distribution can be represented by its univariate marginal distribution and a copula. As a result multivariate association which is fully described by a copula function can be modelled separately from the univariate marginal distributions.

Although many types of two dimensional copulas exist; see, for example, the monographs by Joe (1997) and Nelsen (2006), their extension to multivariate copulas has been limited. Recently, new and extension of families of multivariate copulas have been proposed, see for example, Fischer *et al.* (2009) for a detailed discussion. Examples of multivariate copulas are: elliptical copulas that include Gaussian and t-copulas, multivariate Archimedean copulas (Joe, 1997), Koehler-Symanowski copulas (Palmitesta and Provasi, 2005), Liebscher copula (Liebscher, 2008), and Pair-copula decompositions (Aas *et al.*, 2009). However, the applications of these copulas to multivariate data analysis and graphical models in high dimensional setting have been limited due to the complexity of the copula densities.

In what follows we employ the Gaussian copula and further require conditional independence constraints on the inverse of its correlation matrix. The resulting models are called copula Gaussian graphical models because they only impose a multivariate normal assumption for a set of latent variables which are in a one-to-one correspondence with the set of observed variables. Gaussian copula seems a natural choice beyond the bivariate case.

Genest *et al.* (1995) develop a popular semiparametric estimation or rank based estimation in which the association among the variables are represented with a parametric copula but the marginals are treated as nuisance parameters and estimated non parametrically. The resulting semiparametric estimators are well behaved for continuous data but fail for discrete data for which the distribution of the rank depends on the univariate marginal distributions, making them somewhat inappropriate for the analysis of mixed continuous and discrete data (Hoff, 2007). Hoff (2007) proposed the extended rank likelihood which is a type of marginal likelihood that does not depend on the marginal distribution of the observed variables. Under the extended rank likelihood approach the ranks are free of the nuisance parameters (or marginal likelihood distributions) of the discrete data. This makes the extended rank likelihood approach more focused on the determination of graphical models (or multivariate association) and avoids the difficult problem of modelling marginal distribution (Dobra and Lenkoski, 2011).

The extended rank likelihood was implemented for studying association among mixed variables under a Bayesian framework by Hoff (2007) and further studied in the graphical model setting by Dobra and Lenkoski (2011) who used Bayesian model averaging approach to estimate the graph under the assumption of copula Gaussian density. Since the marginal are treated as nuisance parameters, the parameter of interest is the the correlation matrix or its inverse the precision matrix. Ambroise *et al.* (2009) raised their concern on the challenging task involved in Bayesian framework to construct prior distribution on the precision matrix.

In this chapter, we propose Gaussian copula modelling to estimate conditional (in)dependence structures among continuous and discrete random variables of type: binary, ordinal and count. Various approaches were proposed to study association among discrete variables that include latent models and rank based methods; see, for example Hoff (2007) for a brief review. Under the latent models approach, discrete ordinal data are considered as a realization of a continuous vector of latent variables usually assumed to follow a multivariate normal distribution. The assumption of normality is too restrictive and it can be relaxed via copulas. A copula based multivariate distribution has the advantage to allow arbitrary or unspecified univariate marginal distributions. In the graphical model setting, Dobra and Lenkoski (2011) refereed to Gaussian copula graphical models as an extension of Gaussian graphical models for mixed variables under the assumption of Gaussian copula density for the joint distribution of the entire set of random variables.

In section 4.1 we illustrate the static copula Gaussian model. In section 4.2 we propose coloured graphs to estimate dynamic networks where random variables corresponding to the nodes of the networks can be continuous, binary, ordinal, counts or a mix of the former types. We use the Gaussian copula density and we take advantage from the efficient solver LogDetPPA developed in convex optimization (Wang *et al.*, 2009).

4.1 Introduction to copula

In this section we briefly introduce copula modelling for multivariate data.

What are copulas? From one point of view, copulas are functions that join or 'couple' multivariate distribution functions to their one dimensional marginal distribution functions. Alternatively, copulas are multivariate distribution functions whose one-dimensional margins are uniform on the interval [0,1].

Copulas are of interest for two main reasons: firstly, as a way of studying scale-free measures of dependence; and secondly, as a starting point for constructing families of bivariate distributions, sometimes with a view to simulation.

Sklar (1959) in the theorem describes the functions that 'join together' one-dimensional distribution functions to form multivariate distribution functions. It seems that the earliest paper explicitly relating copulas to the study of dependence among random variables was written by Schweizer and Wolff (1981). In that paper, Schweizer and Wolff discussed and modified Renyi's (1959) criteria for measures of dependence between pairs of random variables, presented the basic invariance properties of copulas under strictly monotone transformations of random variables, and introduced the measure of dependence now known as Schweizer and Wolff's. In their words, since

" . . . under almost surely increasing transformations of (the random variables), the copula is invariant while the margins may be changed at will, it follows that it is precisely the copula which captures those properties of the joint distribution which are invariant under almost surely strictly increasing transformations. Hence the study of rank statistics insofar as it is the study of properties invariant under such transformations may be characterized as the study of copulas and copula-invariant properties".

Suppose we have p random variables $Y_1,\dots,Y_p$ resulting a $p \times 1$ vector of variables $(Y_1,\dots,Y_p)^T$. Denote the cumulative distribution function for these variables by

$$H(y_1,\dots,y_p) = P(Y_1 \leq y_1,\dots,Y_1 \leq y_p),$$

and marginal distributions by $F_j(y_j) = P(Y_j \leq y_j),\ \ j = 1,\dots,p$. Then, according to Sklar's theorem there exists a function $C : [0,1]^p \to R$ called copula such that the joint distribution is given by

$$H(y_1,\dots,y_p) = C(F_1(y_1),\dots,F_p(y_p)),$$

or the copula is given by

$$C(u_1,\dots,u_p) = H(F_1^{-1}(u_1),\dots,F_p^{-1}(u_p)), \quad (u_1,\dots,u_p) \in [0,1]^p,$$

where $F_j^{-1}(u_j) = \inf\left\{y_j : F_j(y_j) \geq u_j\right\}$ is the quantile function of Y_j.

The probability density function corresponding to the joint distribution H can be written in the following way:

$$h(y_1,\dots,y_p) = c(F_1(y_1),\dots,F_p(y_p)) \prod_{j=1}^{p} f_j(y_j),$$

where $c(F_1(y_1),\dots,F_p(y_p))$ is the copula density and $f_j(y_j)$ is the jth marginal probability density function.

We consider a parametric family of copulas denoted by $c_{\mathbf{\Sigma}}(u_1,\dots,u_p)$, where $\mathbf{\Sigma}$ is a dependence parameter, and its inverse $\mathbf{\Theta}$ is a conditional (in)dependence parameter.

Suppose we observe a sample of n replicates of $\mathbf{Y} = (\mathbf{Y}^{(1)},\dots,\mathbf{Y}^{(n)})'$ where $\mathbf{Y}^{(i)} = (Y_1,\dots,Y_p)$. The Gaussian copula is constructed by projection of a multivariate normal distribution on $\mathbb{R}^p$ by means of the probability integral transform to the unit cube $[0,1]^p$. For a given correlation matrix $\mathbf{\Sigma} \in \mathbb{R}^{p \times p}$ the Gaussian copula can be written as

$$C_{\mathbf{\Sigma}}(\mathbf{u}) = \mathbf{\Phi}_{\mathbf{\Sigma}}(\phi^{-1}(u_1),\dots,\phi^{-1}(u_p)),$$

where ϕ^{-1} is the inverse CDF of a standard normal, $\mathbf{\Phi}_{\mathbf{\Sigma}}$ is the joint CDF of a multivariate normal distribution, and $\mathbf{u} = (u_1,\dots,u_p) = (F_1(y_1),\dots,F_p(y_p))$. The density can be written as:

$$\begin{aligned} c_{\boldsymbol{\Sigma}}(\mathbf{u}) &= |\boldsymbol{\Sigma}|^{-1}\exp\left(-\frac{1}{2}\left(\phi^{-1}(u_1),\ldots,\phi^{-1}(u_p)\right)'\left(\boldsymbol{\Sigma}^{-1}-\mathbf{I}\right)\left(\phi^{-1}(u_1),\ldots,\phi^{-1}(u_p)\right)\right) \\ &= |\boldsymbol{\Theta}|\exp\left(-\frac{1}{2}\boldsymbol{\phi}'^{-1}(\mathbf{u})(\boldsymbol{\Theta}-\mathbf{I})\boldsymbol{\phi}^{-1}(\mathbf{u})\right) \qquad (4.1) \\ &= |\boldsymbol{\Theta}|\exp\left(-\frac{1}{2}\mathbf{z}'(\boldsymbol{\Theta}-\mathbf{I})\mathbf{z}\right). \end{aligned}$$

The log-likelihood for a sample of n independent and identically distributed replicates is given by

$$\ell(\boldsymbol{\Theta} \mid \mathbf{Y}) = \sum_{i=1}^{n} \log c(F_1(y_{1i}),\ldots,F_p(y_{pi}) \mid \mathbf{Y}) + \sum_{i=1}^{n}\sum_{j=1}^{p} f_j(y_{ji}).$$

Now we could consider some parametric form for the unknown functions $F_j(y_j)$ but we would rather focus on the conditional independence structures. Using the semiparametric estimation proposed in Genest *et al.* (1995), one can proceed first by estimating the marginal distributions nonparametrically, for example using the rescaled empirical distribution:

$$\hat{F}_j(y_j) = \frac{1}{n+1}\sum_{i=1}^{n} \mathscr{I}\{Y_{ji} \leq y_j\},$$

where $j = 1,\ldots,p$, and then estimate the dependence parameter by maximizing the profile Gaussian copula log-likelihood which is given by

$$\ell_p(\boldsymbol{\Theta},\hat{F}_1,\ldots,\hat{F}_p \mid \mathbf{Y}) = \sum_{i=1}^{n} \log c(\hat{F}_1(y_{1i}),\ldots,\hat{F}_p(y_{pi})).$$

Note that we have omitted the term $\sum_{i=1}^{n}\sum_{j=1}^{p} f_j(y_{ji})$ since it does not involve the parameter $\boldsymbol{\Theta}$ with respect to the profile likelihood needs to be maximized.

Under the condition that the univariate marginals $F_1(Y_1),\ldots,F_p(Y_p)$ are continuous, Genest *et al.* (1995) showed that the resulting estimator of the parameter $\boldsymbol{\Theta}$ is consistent and asymptotically normal. However, Hoff (2007) argued that imposing such continuity condition on the marginal distributions calls to question the appropriateness of this approach for discrete data. To remedy this, he introduced the extended rank likelihood approach for mixed type of data which is discussed in subsection 4.1.2.

4.1.1 Gaussian copula graphical models

Let us recall Gaussian graphical models before moving to copula Gaussian graphical models.

Graphical models are efficient tools for studying statistical models through a compact representation of the joint probability distribution of the underlying random variables. Consider an undirected graph $G = (V,E)$, where V corresponds to the set of nodes or vertices of the graph G with p elements and $E \subset V \times V$ corresponds to the set of ordered pairs of distinct nodes called the edges of G. The nodes of the graph represent the random variables $Y_1,\dots,Y_p$. Let the random vector $\mathbf{Y} = (Y_1,\dots,Y_p)^T$ be Gaussian with a positive definite covariance matrix $\mathbf{\Sigma}$ of dimension $p \times p$. Without loss of generality, we assume $\mathbf{Y}$ follows a p-dimensional multivariate normal distribution with mean zero and covariance matrix $\mathbf{\Sigma}$, $N(\mathbf{0},\mathbf{\Sigma})$. A graphical model $G = (V,E)$ for $\mathbf{Y} \sim N(\mathbf{0},\mathbf{\Sigma})$ is called a Gaussian graphical model.

On the graph G, the edges represent conditional dependence among the random variables. Absence of an edge between any pairs of entries Y_{v_i} and Y_{v_j} corresponds with the conditional independence of these two random variables Y_{v_i} and Y_{v_j} given the remaining variables $\mathbf{Y}_{V\setminus\{v_i,v_j\}}$ where the index $V\setminus\{v_i,v_j\}$ refers to variables other than those indexed by Y_{v_1} and Y_{v_2}. Such conditional independence is usually denoted by

$$Y_{v_i} \perp Y_{v_j} \mid \mathbf{Y}_{V\setminus\{v_i,v_j\}}.$$

Consider the precision matrix also known as the concentration matrix which is the inverse of the covariance matrix, $\mathbf{\Theta} = \mathbf{\Sigma}^{-1}$. Each entry of the precision matrix $\boldsymbol{\theta}_{i,j}$, $i \neq j$ is related to the partial correlation coefficient $\rho_{\{i,j\}|V\setminus\{v_iv_j\}}$ between variables Y_{v_i} and Y_{v_j} by

$$\rho_{\{i,j\}|V\setminus\{v_iv_j\}} = -\frac{\theta_{i,j}}{\sqrt{\theta_{i,i}\theta_{j,j}}},$$

and then it holds that

$$Y_{v_i} \perp Y_{v_j} \mid \mathbf{Y}_{V\setminus v_i,v_j} \Leftrightarrow \theta_{i,j} = 0.$$

In practice, we encounter both discrete and continuous variables that may not be Gaussian. Thus, the assumption of multivariate normal distribution would be too restrictive. To relax the normality requirement, we use the copula framework to construct multivariate distributions for given marginals. For computational convenience, we consider Gaussian copula, however, other copulas can be used in a similar way.

Firstly, we assume that all the random variables are continuous and we show that the conditional independence are encoded in $\mathbf{\Theta}$ while in the next subsection we consider mixed random variables.

The Gaussian copula with parameter $\mathbf{\Sigma}$ of dimension $p \times p$ having $p(p-1)/2$ parameters is given by (4.1). We note that under the Gaussian copula the correlation matrix Σ is the matrix of correlation coefficients among the transformed variables $\phi^{-1}(F_j(y_j)), j = 1, \dots, p$ which represent the maximum pairwise correlation of Y_js, $j = 1, \dots, p$. However, if the univariate marginal distributions are normal, then entries of the correlation matrix represent pairwise correlation coefficients of the variables. We now define the precision matrix as the inverse of the correlation matrix $\mathbf{\Theta} = \mathbf{\Sigma}^{-1}$ to represent conditional (in)dependence among the transformed variables $\Phi^{-1}(F_j(y_j))$s and hence the observed variables y_js if the observed variables are continuous. This is a result of the invariance property of the conditional independence relations over equivalent probability measures as shown by Van Putten and Van Schuppen (1985) in Theorem 3.6.

4.1.2 Gaussian copula for mixed variables

We now focus on graphical modeling for observed variables Y of mixed (continuous, binary, ordinal or count) types. Suppose the j-th variable Y_j has univariate distribution F_j with its pseudo-inverse F_j^{-1}. A Gaussian copula model discussed above can also be constructed by introducing a vector of latent variables $Z \sim N(0_p, \mathbf{\Theta})$ that are related to the observed variables Y as $Y_j = F_j^{-1}(\Phi(Z_j^*))$, $j = 1, \dots, p$, where Z_j^* is a value from some defined interval on Z_j. This could be achieved by specifying a mapping of the discrete values of Y_j into some defined intervals expressed through some thresholds on the continuous latent variable Z_j.

The main assumption to be made on the treatment of mixed variables is that reconstructing the graphical structure implied by the discrete data relies on the conditional dependence induced by the precision matrix of the latent variables. Thus, inference about the precision matrix involves the unobservable latent variable Z. Hoff (2007) shows that though the Zs are not observable the observed Y_js do provide a limited amount of information about them. Since the F_js are nondecreasing, observing $Y_1 < Y_2$ implies that $Z_1 < Z_2$. More generally, observing the ordered data $Y = (Y_1, \dots, Y_n)'$ tells us that $Z = (Z_1, \dots, Z_n)'$ must lie in the set

$$D(Y_1, \dots, Y_n) = \{Z_1, \dots, Z_n \in \mathbb{R}^{n \times p} : L_{ji}(Z_1, \dots, Z_n) < z_{(ji)} < U_{ji}(Z_1, \dots, Z_n)\}, \tag{4.2}$$

where $L_{ji}(Z_1, \dots, Z_n) = \max\{z_{(jk)} : y_{(jk)} < y_{(ji)}\}$ and $U_{ji}(Z_1, \dots, Z_n) = \min\{z_{(jk)} : y_{(ji)} < y_{(jk)}\}$.

To determine the intervals (L, U) in (4.2) in a form convenient for further analysis we consider the typical relationship between Y_j and Z_j which is expressed through some thresholds $\tau_j = \left(\tau_{j0}, \tau_{j1}, \dots, \tau_{jw_j}\right)$ with $-\infty = \tau_{j0} < \tau_{j1} < \dots < \tau_{jw_j}$. Let the observed values of Y_j be ranked in increasing order as $\left\{c_{j1} < \dots < c_{jw_j}\right\}$

so that y_j is the set:

$$y_{ji} = \sum_{r=1}^{w_j} c_{jr} \times I\left\{\tau_{j,r-1} < z_{ji} \leq \tau_{jr}\right\}.$$

It follows that the mapping of the discrete values of Y_j into some defined intervals and the corresponding values to be taken by the latent variable Z_j explicitly given by

$$\begin{aligned}
&z_{ji} \in \left(-\infty, \Phi^{-1}(\hat{F}_j(c_{j1}))\right) \text{ if } y_{ji} = c_{j1} \\
&z_{ji} \in \left(\Phi^{-1}(\hat{F}_j(c_{j1})), \Phi^{-1}(\hat{F}_j(c_{j2}))\right) \text{ if } y_{ji} = c_{j2} \\
&\vdots \\
&z_{ji} \in \left(\Phi^{-1}(\hat{F}_j(c_{jw_{j-1}})), \infty\right) \text{ if } y_{ji} = c_{jw_j}.
\end{aligned}$$

The collection of these intervals is the set $\mathbf{D} = D(Y_1, \ldots, Y_n)$ in (4.2). Finally we take the occurrence of event $Z \in \mathbf{D} = D(Y_1, \ldots, Y_n)$ as our data to infer about the precision matrix of the Gaussian copula separately from the marginal distributions. Such inference approach is referred to as extended rank likelihood by Hoff (2007) and in the graphical modeling setting copula Gaussian graphical modeling by Dobra and Lenkoski (2011).

ℓ_1 penalized EM estimation. Now we consider the implementation of the Expectation Maximization (EM) algorithm Dempster *et al.* (1977) jointly with and without the ℓ_1 penalized likelihood approach. EM algorithm is a popular approach to maximum likelihood estimation. Green (1990) studied convergence properties of the EM algorithm for penalized likelihood. Following the argument of Dempster *et al.* (1977) and Green (1990), for our data setting, the EM algorithm complete data representation involves to consider the observed discrete data $\mathbf{Y}$ as a statistic calculated from the unobserved vector $\mathbf{Z}$ which is assumed to follow a multivariate normal distribution, satisfying:

$$P(\mathbf{Y} \mid \mathbf{\Theta}, F_1, \ldots, F_p) = \int_{Z \in D} \phi_p(z \mid \Theta, F_1, \ldots, F_p) dz.$$

Under the Gaussian copula where we consider $F_1, \ldots, F_p$ as nuisance parameters, the likelihood function using 4.1 is

$$\begin{aligned}
L(\mathbf{\Theta}) &= \int_{Z \in D} \Phi(z \mid \mathbf{\Theta}, F_1, \ldots, F_p) dz \\
&= \int_{Z \in D} \Phi(z \mid \mathbf{\Theta}) dz. \qquad (4.3)
\end{aligned}$$

Then, for large sample sizes the precision matrix $\mathbf{\Theta}$ is estimated by maximizing the log-likelihood $\log L(\mathbf{\Theta})$ as a function of $\mathbf{\Theta}$. For high dimensional data, we add an the ℓ_1-norm penalty to encourage sparsity in the precision matrix. That is, the ℓ_1 penalized log-likelihood takes the form

$$\log L_{\text{pen}}(\mathbf{\Theta}) = \log L(\mathbf{\Theta}) - \lambda \|\mathbf{\Theta}\|_1 , \tag{4.4}$$

where the scalar parameter $\lambda > 0$ controls the size of the penalty.

However, due to the complexity of maximizing the log-likelihood $\log L(\Theta)$ in (4.3) and the penalized log-likelihood (4.4) we use EM algorithm that alternate iteratively between the E-step computing conditional expectation, $Q(\cdot|\cdot)$ defined as

$$Q(\mathbf{\Theta} \mid \mathbf{\Theta}^{(m)}), \tag{4.5}$$

and it is defined on

$$Q(\mathbf{\Theta} \mid \mathbf{\Theta}^{(m)}) - \lambda \|\mathbf{\Theta}\|_1 , \tag{4.6}$$

for the ℓ_1 penalized likelihood ((4.4)), where $Q(\mathbf{\Theta} \mid \mathbf{\Theta}^{(m)}) = E\left[\log L_c(Z \mid \mathbf{\Theta}) \mid z \in \mathbf{D}\right]$ is the conditional expectation of the complete data log-likelihood given observed data and $\mathbf{\Theta}^{(m)}$ is an estimate of $\mathbf{\Theta}$ from the previous step of the algorithm. The complete data log-likelihood is $\log L_c(Z \mid \mathbf{\Theta}) = \prod_{i=1}^{n} \phi_p(Z_i \mid \mathbf{\Theta}, F_1, \ldots, F_p)$. In the M-step we want to maximize the quantity (4.5) or the penalized conditional expectation 4.6 over $\mathbf{\Theta}$, i.e:

$$(\hat{\Theta}^{(m+1)}) := \text{argmax}_{\Theta} Q(\mathbf{\Theta} \mid \mathbf{\Theta}^{(m)}), \tag{4.7}$$

or for the penalized case

$$(\hat{\Theta}^{(m+1)}) := \text{argmax}_{\Theta} Q(\mathbf{\Theta} \mid \mathbf{\Theta}^{(m)}) - \lambda \|\mathbf{\Theta}\|_1 , \tag{4.8}$$

Remark 4.1.1. *We note that the EM algorithm based on (*(4.5)*) is the maximum likelihood approach to that of the Bayesian inference in Hoff (2007).*

For the EM algorithm we have the following computational strategy:
E-step: This step involves computing conditional expectation of the complete data log-likelihood given the observed data. Given the complete data log-likelihood

$$\log L_c(Z \mid \mathbf{\Theta}, F_1, \ldots, F_p) = \log L_c(Z \mid \mathbf{\Theta}),$$

that focuses only on the parameter of interest $\mathbf{\Theta}$, then the conditional expectation given the data $z \in \mathbf{D}$ is obtained as follows:

$$\begin{aligned} Q(\mathbf{\Theta} \mid \mathbf{\Theta}^{(m)}) &= \mathbb{E}\left[\log L_c\left(Z \mid \mathbf{\Theta}) \mid z \in \mathbf{D}, \mathbf{\Theta}^{(m)}\right)\right] \\ &= \mathbb{E}\left[\sum_{i=1}^{n} \log \phi_p(Z_i \mid \mathbf{\Theta}) \mid z_i \in \mathbf{D}, \mathbf{\Theta}^{(m)}\right]. \end{aligned}$$

Under Gaussian copula with density given in 4.1, it follows that:

$$
\begin{aligned}
Q(\boldsymbol{\Theta} \mid \boldsymbol{\Theta}^{(m)}) &= \mathbb{E}\left[\sum_{i=1}^{n}\left(\frac{1}{2}\log|\boldsymbol{\Theta}| - \frac{1}{2}Z_i'\boldsymbol{\Theta}Z_i + \frac{1}{2}Z_i'Z_i\right) \mid z_i \in \mathbf{D}, \boldsymbol{\Theta}^{(m)}\right] \\
&= \frac{n}{2}\left\{\log|\boldsymbol{\Theta}| - \frac{1}{n}\sum_{i=1}^{n}\mathrm{tr}\left(\boldsymbol{\Theta}\mathbb{E}\left[Z_iZ_i' \mid z_i \in \mathbf{D}, \boldsymbol{\Theta}^{(m)}\right]\right)\right. \\
&\qquad \left. + \frac{1}{n}\sum_{i=1}^{n}\mathrm{tr}\left(\mathbb{E}\left[Z_iZ_i' \mid z_i \in \mathbf{D}, \boldsymbol{\Theta}^{(m)}\right]\right)\right\} \\
&= \frac{n}{2}\left\{\log|\boldsymbol{\Theta}| - \mathrm{tr}\left(\boldsymbol{\Theta}\bar{R}\right) + \mathrm{tr}\left(\bar{\mathbf{R}}\right)\right\}, \qquad (4.9)
\end{aligned}
$$

where tr stands for a trace of a matrix and

$$
\bar{R} = \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}\left[Z_iZ_i' \mid z_i \in \mathbf{D}, \boldsymbol{\Theta}^{(m)}\right]. \qquad (4.10)
$$

Note that in (4.9) we need to calculate $\mathbb{E}\left[Z_iZ_i' \mid z_i \in \mathbf{D}, \boldsymbol{\Theta}^{(m)}\right]$, where the expectation is defined on the truncated multivariate normal distribution for Z_i given $z_i \in \mathbf{D}$ for subject i. The truncated multivariate normal density is given by

$$
\phi_p(z \mid z \in \mathbf{D}, \boldsymbol{\Theta}^{(m)}) = \frac{\phi_p(z \mid \boldsymbol{\Theta}^{(m)})}{P(z \in \mathbf{D})}. \qquad (4.11)
$$

For singly truncated multivariate normal distribution, moment generating functions and explicit expressions for moments were derived by using the correlation matrix by Tallis (1961). Extension to doubly truncated multivariate normal is considered by Wilhelm and Manjunath (2010) who provided an algorithm to compute mean and covariance for the truncated normal random vector using the moment formulas given below and also based on MCMC Gibbs sampling. We present here the expressions for the first two moments from Wilhelm and Manjunath (2010) in terms of the correlation matrix $\left(\boldsymbol{\Theta}^{(m)}\right)^{-1} = ((\rho_{kl}^{(m)}))$, $k, l = 1, \ldots, p$, that corresponds to the precision matrix $\boldsymbol{\Theta}^{(m)}$. Let the lower and upper truncation points of Z be

$(L,U) = \{(L_k,U_k),\ \ k = 1,\dots,p\} \in \mathbf{D}$. The first and second moments are:

$$\begin{aligned}
\mathbb{E}\left[Z_s \mid (L,U) \in \mathbf{D}, \boldsymbol{\Theta}^{(m)}\right] &= \sum_{k=1}^{p} \rho_{sk}^{(m)} \left[G_k(L_k) - G_k(U_k)\right], \quad s = 1,\dots,p, \\
\mathbb{E}\left[Z_s Z_t \mid (L,U) \in \mathbf{D}, \boldsymbol{\Theta}^{(m)}\right] &= \sum_{k=1}^{p} \rho_{sk}^{(m)} \rho_{tk}^{(m)} \left[L_k G_k(L_k) - U_k G_k(U_k)\right] \\
&\quad + \sum_{k=1}^{p} \rho_{tk}^{(m)} \sum_{l \neq k} \left(\rho_{tl}^{(m)} - \rho_{kl}^{(m)} \rho_{tk}^{(m)}\right) \\
&\quad\quad \times \{[G_{kl}(L_k,L_l) - G_{kl}(L_k,U_l)] \\
&\quad\quad\ - [G_{kl}(U_k,L_l) - G_{kl}(U_k,L_l)]\}
\end{aligned}$$

where $s,t = 1,\dots,p$ and

$$\begin{aligned}
G_k(z_k) &= \int_{L_1}^{U_1} \cdots \int_{L_{k-1}}^{U_{k-1}} \int_{L_{k+1}}^{U_{k+1}} \cdots \int_{L_p}^{U_p} \phi_p(z_k, z_{-k} \mid \Theta^{(m)}) dz_{-k} \\
G_{kl}(z_k,z_l) &= \int_{L_1}^{U_1} \cdots \int_{L_{k-1}}^{U_{k-1}} \int_{L_{k+1}}^{U_{k+1}} \cdots \int_{L_{l-1}}^{U_{l-1}} \int_{L_{l+1}}^{U_{l+1}} \\
&\quad \cdots \int_{L_p}^{U_p} \phi_p(z_k, z_l, z_{-k,-l} \mid \Theta^{(m)}) dz_{-k,-l},
\end{aligned}$$

with Z_{-k} stands for all variables except z_k and and $z_{-k,-l}$ for all except z_k and z_l.
M-step: Update the parameter estimate using the likelihood in (4.9):

$$\left(\hat{\boldsymbol{\Theta}}^{(m+1)}\right) := \text{argmax}_{\boldsymbol{\Theta}} \left\{Q(\boldsymbol{\Theta} \mid \boldsymbol{\Theta}^{(m)})\right\} \tag{4.12}$$

or the ℓ_1 penalized likelihood in (4.9):

$$\left(\hat{\boldsymbol{\Theta}}^{(m+1)}\right) := \text{argmax}_{\boldsymbol{\Theta}} \left\{Q(\boldsymbol{\Theta} \mid \boldsymbol{\Theta}^{(m)}) - \lambda \left\|\boldsymbol{\Theta}\right\|_1\right\} \tag{4.13}$$

Substituting (4.9) into (4.12) or (4.13) and ignoring constants with respect to $\boldsymbol{\Theta}$, it follows for the unpenalized likelihood from (4.12) that

$$\left(\hat{\boldsymbol{\Theta}}^{(m+1)}\right) := \text{argmax}_{\boldsymbol{\Theta}} \left\{\log|\boldsymbol{\Theta}| - \text{tr}(\boldsymbol{\Theta}\bar{R})\right\}. \tag{4.14}$$

and for the ℓ_1 penalized likelihood from (4.13)

$$\left(\hat{\boldsymbol{\Theta}}^{(m+1)}\right) := \text{argmax}_{\boldsymbol{\Theta}} \left\{\log|\boldsymbol{\Theta}| - \text{tr}(\boldsymbol{\Theta}\bar{R}) - \lambda \left\|\boldsymbol{\Theta}\right\|_1\right\}. \tag{4.15}$$

Remark 4.1.2. *Computation of $\bar{R}$ from the E-step can be done using the package tmvtnorm in R developed by Wilhelm and Manjunath (2010). We note that use of the Gibbs sampling option leads to fast computational time specially when p is very large.*

Remark 4.1.3. *The maximization problem in* (4.14) *and* (4.15) *can be implemented using the package glasso in R developed by Friedman* et al. *(2008), where for the unpenalized estimation in* (4.14) *we set the penalty parameter* $\lambda = 0$ *and for the* ℓ_1 *penalized maximization we determine* λ *based on the formula given in Banerjee* et al. *(2008) but adopted to scaled latent variables,* $\lambda = \frac{t_{\alpha/p^2}}{\sqrt{n-2+t^2_{\alpha/p^2}}}$*, where* t_{α/p^2} *denotes the* $(100-\alpha/p^2)\%$ *point of the Student's t-distribution for n-2 degrees of freedom. However, one can proceed with the choice of* λ *using information criteria based methods like AIC and BIC, or by using cross-validation that we have not formally addressed in this thesis, or by using stability selection that we have addressed in Chapter 4.3.2.*

4.1.3 Examples

In this sub-section we apply copula based EM estimation on two multivariate datasets where the analysis can be done with and without ℓ_1 penalized likelihood.

4.1.4 General Social Survey

Using labour force data from the General Social Survey (GSS) that is available from http://webapp.icpsr.umich.edu/GSSS/, Hoff (2007) studied the dependencies among seven relevant variables of interest from 1002 males in the U.S. labor force. These variables include the income, education and number of children of the survey respondent, as well as similar variables for the respondent's parents. Age of the survey respondent is additionally included, as it is typically strongly related to income and number of children. The measurement scales for these variables are as follows:

INC: income of the respondent in 1000s of dollars, binned into 21 ordered categories.

DEG: highest degree ever obtained, ordered as None, HS, Associates, Bachelors, Graduate.

CHILD: number of children ever had by the respondent.

PINC: financial status of respondent's parents when respondent was 16 (on a 5-point scale).

PDEG: maximum of mother's and father's highest degree, ordered in five categories.

PCHILD: number of siblings of the respondent plus one.

AGE: age of the respondent in years.

There is a certain amount of missing data with higher rates for INC and PINC. It is suggested that the missing values in these variables can be reasonably considered as missing at random. The heterogeneity in the marginal distributions of the seven observed variables makes the study of their joint distribution very difficult. However, the copula-based inference that consider the marginal distributions as nuisance parameters have been applied successfully on this dataset using the Bayesian framework (Hoff, 2007). We analysed this data using the Gaussian copula-based EM proposed in this work.

We present the partial correlation and partial regression coefficient estimates of the seven variables induced by the multivariate normal latent variables in Table 4.1. Elements above the main diagonal shows partial correlation coefficients and elements below the main diagonal represent partial regression coefficients. To assess conditional independence between the variables in this dataset we consider tests based on partial correlations and also partial regression coefficients. Those estimates significant at 5% level taking into account multiple comparison are indicated by asterisks(*). We remark that the relationships among income, degree and fertility seem to hold across generations. Though, this conclusion is in agreement with Bayesian inference in Hoff (2007), our findings slightly differ as we found a significant positive relationship between INC and PINC in the presence of intergenerational relationships of DEG, PDEG, CHILD and PCHILD. In addition, there is no significant relationship between DEG and PCHILD.

The conditional independence tests help to determine the presence or absence of edges or links for graphical visualization. Figure 4.1 displays the links among the six variables implicitly conditioning on age of respondent, see Hoff (2007).

4.1.5 CHIP-seq count data

CHIP-Seq (Chromatin Immunoprecipitation followed by sequencing) is used to analyze protein interactions with DNA. It combines chromatin immunoprecipitation (CHIP) with massively parallel DNA sequencing to identify the cistrome of DNA-associated proteins. It can be used to precisely map global binding sites for any protein of interest. Kasowski *et al.* (2010) compared protein occupation of DNA regions between ten human individuals: 9 females and 1 male by CHIP-Seq to map

	INC	DEG	CHILD	PINC	PDEG	PCHILD	AGE
INC	-	0.4516*	0.2146*	0.0988*	0.0191	-0.0028	0.1875*
DEG	0.4398*	-	-0.1009*	-0.0487	0.3733*	-0.0848	0.0484
CHILD	0.2075*	-0.1002*	-	-0.0442	-0.0748	0.1489*	0.4959*
PINC	0.1085*	-0.0560	-0.0512	-	0.3449*	-0.0970*	-0.0278
PDEG	0.0186	0.3740*	-0.0792	0.3004*	-	-0.1356*	-0.1578*
PCHILD	-0.0033	-0.1023	0.1809*	-0.1017*	-0.1633*	-	-0.0304
AGE	0.1834*	0.0486	0.5013*	-0.0242	-0.1581*	-0.0253	-

Table 4.1. Estimated partial correlations (elements above the main diagonal) and partial regression coefficients (elements below the main diagonal) in the GSS data.

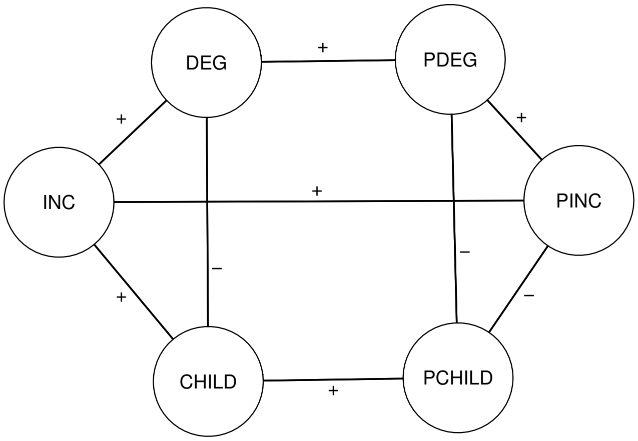


Figure 4.1. Conditional dependence reduced graph for the GSS data.

nuclear factor kB (NFkB) and RNA polymerase II (polII) binding sites. They compiled a list of binding regions for polymerase II and FNkB, and counted, for each sample, the number of reads that mapped onto each predetermined binding region. The aim of their study was to investigate how much the regions occupation differed between individuals. Significant differences in binding were observed. Moreover, they suggested that adjacent binding sites and binding regions may influence one another, perhaps through cooperative binding or interactions with other proteins.

Our aim is to assess how binding regions influence each other through graphical modelling using count data from CHIP-seq. We consider a small portion of the 19011 binding regions for polymerase II. In particular, to illustrate our approach we consider the 555 binding regions labelled as ”chrX”. In practice in many CHIP-seq datasets only few replications are provided. This dataset includes 10 individuals with varied number of repeated measurements. To increase the sample size we considered all $n = 39$ replications ignoring the correlations among the repeated measures. To obtain a sparse graphs we used the Gaussian copula EM with ℓ_1 penalized likelihood estimation. The resulting graph that provide the main links among 36 binding sites of ”chrX” for $\lambda = 0.93$ are displayed in Figure 4.2.

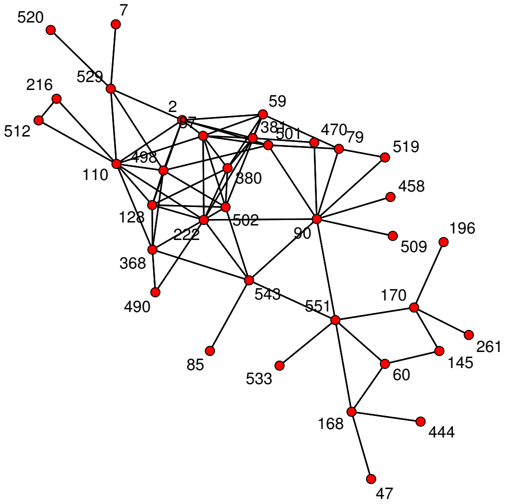


Figure 4.2. Conditional dependence graph for the CHIP-seq data.

4.2 Coloured graphs for estimating dynamic networks

In this section we address the problem of estimating dynamic networks when random variables are continuous but the multivariate normality cannot be assumed or mixed (binary, ordinal and counts). In order to take into account the dynamic of the networks we consider coloured graphs which are graphs where vertices and nodes in the same partitions have the same colours. Coloured graphs allow to define several model-based graphical models. In the case of Gaussian graphical models we have seen that specific "dynamic constraints" can be imposed on the precision matrix by assuming a specific coloured graph. Now we apply the same idea to the Gaussian coloured graph. Firstly, we recall some definition about coloured graphs and natural partitions. Secondly, we connect Gaussian copula graphical models with coloured graphs. This results in an extension of the model proposed in Chapter 2.

Definition 4.2.1 (Coloured Graph)**.** *A coloured graph $\tilde{G} = (V,E,F)$ is a triplet, where $G = (V,E)$ is a graph and F is a mapping on the links, i.e.*

$$F : E \longrightarrow C,$$

where C is a finite set of colours.

Suppose we have g genes $(Y_1,\dots,Y_g)$ observed at time t resulting in a $gt \times 1$ vector of variables $\mathbf{Y} = (Y_{11},\dots,Y_{g1},\dots,Y_{1t},\dots,Y_{gt})$ and denote the CDF:

$$H(y_{11},\dots,y_{g1},\dots,y_{1t},\dots,y_{gt}) = P(Y_{11} \le y_{11},\dots,Y_{gt} \le y_{gt}) \tag{4.16}$$

and marginal distributions by $F_{jk}(y_{jk}) = P(Y_{jk} \le y_{jk})$, $j = 1,\dots,g$ and $k = 1,\dots,t$. According to Sklar's theorem there exists a copula C such that the joint distribution is given by

$$H(y_{11},\dots,y_{g1},\dots,y_{1t},\dots,y_{gt}) = C(F_{11}(y_{11},\dots,F_{g1}(y_{g1}),\dots,F_{1t}(y_{1t}),\dots,F_{gt}(y_{gt})),$$

and the density function corresponding to the joint distribution H can be written in the following way:

$$h(y_{11},\dots,y_{gt}) = c(F_{11}(y_{11}),\dots,F_{gt}(y_{gt}))\prod_{j=1}^{p}\prod_{k=1}^{t} f_{jk}(y_{jk}) \tag{4.17}$$

where $c(F_{11}(y_{11}),...,F_{gt}(y_{gt}))$ is the copula density and $f_{jk}(y_{jk})$ is the jk-th marginal pdf.

We have seen that we can consider Gaussian copula graphical models with association parameters $\mathbf{\Sigma}$ which is a correlation matrix parametric. However, this would imply some constraints on $\mathbf{\Sigma}$ when we consider to solve the optimization problem. In particular we should impose constraints on the diagonal elements which must be ones and on the off-diagonal elements which must be in the range $[-1,1]$. However, we are more interested in $\mathbf{\Theta}$ and to impose such constraints on $\mathbf{\Sigma} = \mathbf{\Theta}^{-1}$ would be a difficult task. Instead, we assume that $\mathbf{Y}$ follows a non-canonical Gaussian copula distribution denoted by $C(u_{11},\dots,u_{gt};\mathbf{\Theta})$ where we define the canonical Gaussian copula as:

Definition 4.2.2 (Non-canonical Gaussian copula)**.** *The non canonical Gaussian copula with matrix $\mathbf{\Theta}$ is given by*

$$C(u_{11},\dots,u_{gt}|\mathbf{\Sigma}) = \mathbf{\Phi}_{gt}(\Phi_1^{-11}(u_{11}),\dots,\Phi_{gt}^{-1}(u_{gt})|\mathbf{\Sigma}), \tag{4.18}$$

where ϕ_i is the CDF of normal distribution with mean 0 and with variance σ_{ii}, $\mathbf{\Phi}_{gt}$ is the CDF of a multivariate normal distribution $N(\mathbf{0},\mathbf{\Sigma})$ with diagonal elements σ_{ii}, and the precision matrix $\mathbf{\Theta} = \mathbf{\Sigma}^{-1}$ has diagonal equal one.

Here $\mathbf{\Theta}$ is refereed as dependence or association parameter of dimension $gt \times gt$ and it has $\mathbf{\Theta}$ has $gt(gt-1)/2$ parameters since we impose a constraint on the diagonal of $\mathbf{\Theta}$, i.e. the diagonal elements must be one. Suppose we observe a sample of n replicates of $(Y_{11},...,Y_{gt})$. The Gaussian copula with matrix $\mathbf{\Sigma}$ of dimension $gt \times gt$ is given by:

$$C(u_{11},\dots,u_{gt}|\mathbf{\Sigma}) = \mathbf{\Phi}_{gt}(\Phi_{11}^{-1}(u_{11}),\dots,\Phi_{gt}^{-1}(u_{gt})|\mathbf{\Sigma}), \tag{4.19}$$

where ϕ_i is the CDF of normal distribution with mean 0 and with variance σ_{ii}, and $\mathbf{\Phi}$ is the CDF of multivariate normal distribution $N(\mathbf{0},\mathbf{\Sigma})$.

There is a crucial difference between this approach and the previous one. In fact, this function cannot be defined to be a Gaussian copula function since its arguments ϕ_{ij} are normal density with variance range 0 and σ_{ii}. However, we need still to impose a constraint on the precision matrix to avoid identifiability problems. This is easier than to impose constraints on $\mathbf{\Sigma}$.

Note that we have defined the precision matrix as the inverse of the covariance matrix $\mathbf{\Theta} = \mathbf{\Sigma}^{-1}$. The precision matrix has the characteristic to be scale free since the diagonal elements are constraint to be one. It represents conditional (in)dependence among the transformed variables and hence the observed variables. This is because of the invariance property of conditional independence relation over equivalent probability measures as shown by Van Putten and Van Schuppen (1985).

The corresponding copula-based distribution function is

$$H(y_{11},\dots,y_{gt}|\mathbf{\Sigma},F_{11},\dots,F_{gt}) = \mathbf{\Phi}_{gt}(\Phi_{11}^{-1}(F_{11}(y_{11})),\dots,\Phi_{gt}^{-1}(F_{gt}(y_{gt}))). \tag{4.20}$$

Differentiating the Gaussian copula $C(\cdot|\mathbf{\Theta})$ with respect to $\mathbf{u} = (u_{11}, \ldots, u_{gt})$ yields the non-canonical Gaussian copula density function given by,

$$\begin{aligned} c(u_{11}, \ldots, u_{gt}|\mathbf{\Theta}) &= \frac{\boldsymbol{\phi}_{gt}(\Phi_{11}^{-1}(u_{11}), \ldots, \Phi_{gt}^{-1}(u_{gt})|\mathbf{\Theta})}{\boldsymbol{\phi}_{gt}(\Phi_{11}^{-1}(u_{11}), \ldots, \Phi_{gt}^{-1}(u_{gt})} \\ &= |\mathbf{\Theta}|^{1/2} \exp\left(\frac{1}{2}\boldsymbol{\phi}_{gt}^{-1}(\mathbf{u})'\mathbf{\Theta}\boldsymbol{\phi}_{gt}^{-1}(\mathbf{u})\right) \exp\left(\frac{1}{2}\boldsymbol{\phi}_{gt}^{-1}(\mathbf{u})'\boldsymbol{\phi}_{gt}^{-1}(\mathbf{u})\right) \\ &= |\mathbf{\Theta}|^{1/2} \exp\left(\frac{1}{2}\boldsymbol{\phi}_{gt}^{-1}(\mathbf{u})'(\mathbf{\Theta} - \mathbf{I})\boldsymbol{\phi}_{gt}^{-1}(\mathbf{u})\right). \qquad (4.21) \end{aligned}$$

For convenience, we denote $\mathbf{z} = \boldsymbol{\phi}_{gt}^{-1}(\mathbf{u}) = (\phi_{11}^{-1}(u_{11}), \ldots, \phi_{gt}^{-1}(u_{gt}))$, and so the alternative Gaussian copula density is given by

$$\boldsymbol{\phi}_{GC}(\mathbf{z}) = |\mathbf{\Theta}|^{1/2} \exp\left(\frac{1}{2}\mathbf{z}'(\mathbf{\Theta} - \mathbf{I})\mathbf{z}\right). \qquad (4.22)$$

Now we consider the expectation maximization steps for ℓ_1-penalized likelihood approach for non-canonical copula Gaussian graphical models. This allows us to deal with dynamic graphical modelling for mixed random variables $\mathbf{Y}$. Under the Gaussian copula where we consider $F_{11}, \ldots, F_{gt}$ as nuisance parameters, the likelihood function in the expectation step is:

$$\begin{aligned} Q(\mathbf{\Theta}|\mathbf{\Theta}^{(m)}) &= \mathbb{E}\left[\sum_{i=1}^{n} \frac{1}{2}\left(\log|\mathbf{\Theta}| - \frac{1}{2}\mathbf{Z}_i'\mathbf{\Theta}\mathbf{Z}_i + \frac{1}{2}\mathbf{Z}_i'\mathbf{Z}_i | \mathbf{z}_i \in \mathbf{D}, \mathbf{\Theta}^{(m)}\right)\right] \\ &= \frac{n}{2}\log|\mathbf{\Theta}| - \frac{n}{2}\sum_{i=1}^{n}\left(\mathrm{tr}(\mathbf{\Theta}\mathbb{E}\left[\mathbf{Z}_i\mathbf{Z}_i'|\mathbf{z}_i \in \mathbf{D}, \mathbf{\Theta}^{(m)}\right]\right) \\ &= \frac{n}{2}\left(\log|\mathbf{\Theta}| - \mathrm{tr}(\mathbf{\Theta}\bar{\mathbf{R}})\right), \qquad (4.23) \end{aligned}$$

where $\mathbf{Z} \sim N(\mathbf{0}, \mathbf{\Theta}^{-1})$ is the continuous latent variable, $\mathbf{D}$ is the range of value lower and upper bounds for $\mathbf{Y}$ the observed variable, tr stands for a trace of a matrix and $\bar{\mathbf{R}} = \frac{1}{n}\sum_{i=1}^{n} \mathbb{E}\left[\mathbf{Z}_i\mathbf{Z}_i' \mid z_i \in \mathbf{D}, \mathbf{\Theta}^{(m)}\right]$. The expectation can be computed with an MCMC Gibbs sampling procedure (Wilhelm and Manjunath, 2010).

Now we can impose specific constraints on the precision matrix, as we have seen in Chapter 2, and we will use the idea of natural partitions, i.e. given a graph $G = (V, E)$ we can partition V and E such that:

$$S_i = \{\{(v_{jt}, v_{j,t+i}), (v_{j,t+i}, v_{jt})\} | j \in \Gamma, t = 1, \ldots, n_T - i\},$$

and

$$N_i = \{\{(v_{jt}, v_{k,t+i}), (v_{k,t+i}, v_{jt})\} | \forall j \neq k \in \Gamma, t = 1, \ldots, n_T - i\}.$$

where $\{S_i\}_{i=0}^{n_T-1}$, and $\{N_i\}_{i=0}^{n_T-1}$ are subsets of vertices V and links E. Each of these partitions is interpreted as follows: S_i considers the lag i interactions between the same natural vertices, and N_i is a graph at time lag i. As we show in Chapter 2 specific maps can be imposed on the natural partitions, i.e. $S_i, N_i \prec F_j$, where $i = 1, \ldots, T-1$ and $j = 1, T, \Gamma$ or ΓT. Then, two sets of design matrices $\mathbf{X}^S = \{\mathbf{X}^{S_i^m}\}_{i=0,m=1}^{n_{\Gamma-1}, s_i}$ and $\mathbf{X}^N = \{\mathbf{X}^{N_i^m}\}_{i=0,m=1}^{n_{\Gamma-1}, n_i}$ where $\mathbf{X}^{S_i^m}, \mathbf{X}^{N_i^m} \in \mathbb{R}^{\Gamma T}$ and $\mathbf{X}^{S_i^m}$ can be uniquely identified such that:

$$x_{jt,gs}^{S_i^m} = \begin{cases} 1 & \text{if } (v_{jt}, v_{gs}) \in S_i^m \\ 0 & \text{otherwise} \end{cases}$$

and,

$$x_{jt,gs}^{N_i^m} = \begin{cases} 1 & \text{if } (v_{jt}, v_{gs}) \in N_i^m \\ 0 & \text{otherwise} \end{cases}$$

We re-define the set of design matrices has:

$$\mathbf{X} = \{\mathbf{X}^S, \mathbf{X}^N\} = \{\mathbf{X}_1, \ldots, \mathbf{X}_{n_c}\},$$

where $\mathbf{X}^S = \cup \mathbf{X}^{S_i^m}$, $\mathbf{X}^N = \cup \mathbf{X}^{N_i^m}$, and n_c is the total number of colours.

Consider a coloured graph that specifies a partition of $\mathbf{\Theta}$, then a set of design matrices $\mathbf{X}$ is used to produce linear operators which are necessary to impose linear restrictions on $\mathbf{\Theta}$. Every design matrix $\mathbf{X}^{(m)}$, $m = 1, \ldots, n_c$ consists of zeroes and ones and induce $p-1$ linear constraints on $\mathbf{\Theta}$ when the number of 1 in $\mathbf{X}^{(m)}$ is p. We define a linear map $\mathbf{A} = (\mathbf{A}_1, \ldots, \mathbf{A}_{n_p})$ where each $\mathbf{X}^{(m)}$ induces a set of matrices $\mathbf{A}_1, \ldots, \mathbf{A}_{n_m}$ and an element in $\mathbf{A}^{(i)}$, $i = 1, \ldots, n_p$ assumes value -1, 0 or 1 as described below:

$$\mathbf{a}_{jt,gs}^{(i)} = \begin{cases} 1 & \text{if } \sum_{jg} \sum_{ts} \mathbf{X}_{jt,gs}^{(i)} \text{ before } jt, gs = p-1 \\ -1 & \text{if } \sum_{jg} \sum_{ts} \mathbf{X}_{jt,gs}^{(i)} \text{ before } jt, gs = p \\ 0 & \text{otherwise,} \end{cases}$$

where 'before' is meant with respect to the total row-major ordering of the matrices. Now let n_p be the total number of linear constraints, then $\mathbf{A} = \{\mathbf{A}_1, \ldots, \mathbf{A}_{n_p}\}$ and the linear map is expressed as $\mathbf{A} : \mathbb{R}_{\text{sym}}^{\Gamma T} \to \mathbb{R}^{n_p}$ with:

$$\mathbf{A}(\mathbf{\Theta}) = [\langle \mathbf{A}_1, \mathbf{\Theta} \rangle, \ldots, \langle \mathbf{A}_{n_p}, \mathbf{\Theta} \rangle]$$

where $\langle , \rangle$ is the Frobenius inner product.

When we derive the objective function we want to take into account the sparsity assumption as well as the linear constraints deriving by imposing a specific coloured graph. To achieve sparse graph structures one can minimize a penalized log-likelihood function, i.e.:

$$
\begin{aligned}
(\hat{\boldsymbol{\Theta}}) \quad &:= \quad \underset{\boldsymbol{\Theta}}{\operatorname{argmin}}\{-\frac{n}{2} - \log|\boldsymbol{\Theta}| + \operatorname{tr}(\boldsymbol{\Theta}\bar{\mathbf{R}}) + \lambda\mathbf{x}^{+} + \lambda\mathbf{x}^{-}\} \qquad (4.24) \\
\text{subject to} \quad & A(\boldsymbol{\Theta}) = \mathbf{0} \\
& \mathbf{B}(\boldsymbol{\Theta}) - \mathbf{x}^{+} + \mathbf{x}^{-} = \mathbf{0} \\
& \boldsymbol{\Theta} \succeq 0, \mathbf{x}^{+}, \mathbf{x}^{-} \geq 0.
\end{aligned}
$$

where **A** and **B** and $\mathbf{x}^{+}$, $\mathbf{x}^{-}$ are derived in Section 3.3.3. Here the objective function is represented by the conditional expectation function $Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^{(m)})$ from the E-step and we want to maximize this function over $\boldsymbol{\Theta}$. Note that the diagonal elements of $\boldsymbol{\Theta}$ are constraints to be one.

4.3 Application

4.3.1 Simulation study for copula model

Example 1. Consider a dynamic graph with $p = 10$ variables and $t = 2$ time points. Simulate $\mathbf{z}^{(i)} \sim N(\mathbf{0}, \boldsymbol{\Theta}^{-1})$, where $\boldsymbol{\Theta}$ is the precision matrix with diagonal one, $i = 1, \ldots, 100$ is the number of independent replicates. We transform **z** such that:

$$F_i^{-1}(\Phi_i(\mathbf{z}_i)),$$

where F_i is the CDF of an exponential distribution with parameter $\lambda = 2$. The pseudo-inverse of F is given by

$$\mathbf{z}_i = \frac{\log(1 - \mathbf{x}_i)}{-\lambda}$$

where $i = 1, \ldots, 20$, $\mathbf{x}_i = \Phi_i(\sigma_i)(z_i)$ is the standard normal CDF with mean zero and variance σ_{ii}.

We can use copula theory, i.e. we consider the inverse transformation $\Phi_i^{-1}(\hat{F}_i(\mathbf{y_i}))$ to find a new variable $\mathbf{z}_i^{*}$ which is marginally Gaussian. Here $\hat{F}_i$ is a scaled empirical CDF. Marginal densities for the first variable of the simulated data **z**, the exponential data **y**, and copula transformed data $\mathbf{z}^{*}$ are shown in Figure 4.3. We can see that exponential data are highly skewed distributed.

Now we are ready to apply a structured Gaussian graphical model for the exponential data and a structured copula Gaussian graphical model for the transformed data. We consider the following model:

$$S_0 \prec F_1, N_0 \prec F_{\Gamma}, S_1 \prec 0,$$

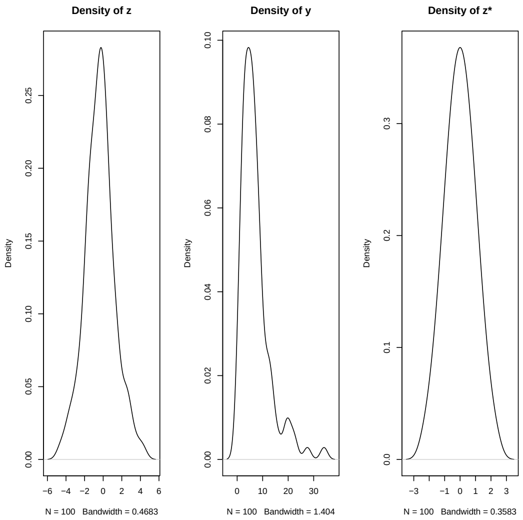


Figure 4.3. Smooth density estimates based on 100 replicates for latent $\mathbf{z}_1$, observed $\mathbf{y}_1$, and the copula back-transformed $\mathbf{z}_i^*$.

which includes the true dynamic networks where the data **z** were generated from. Figure 4.4 shows the true network while the recovered graphs for the exponential data and the copula transformed data are shown in Figure 4.5. Indeed, we can recover a much closer structure to the true one by considering the copula transformation. An empty graph was estimated from the data **y** which are non Gaussian distributed and graph with almost the same structure as the true graph was estimated by considering the copula transformation. Note that the nodes involved in the right graph of Figure 4.5 are the same involved in the true graph Figure 4.4. Note that we showed the networks at the first time point in Figure 4.5 and 4.4 since at the second time point the structures are identical due to the coloured graph constraints.

4.3.2 Real data application T-cell

In chapter , we applied several structured Gaussian graphical models for the data set T-cell which is a time-course dataset where expression levels of 58 genes were collected across 10 time points. A better description of this data set is given in Rangel *et al.* (2004). Here we apply structured copula Graphical model for estimating the structure of the dynamic networks. Figure shows the approximate marginal densities for the original data T-cell where 47 genes were considered.

Since the application is mainly meant to be an illustration we consider the same

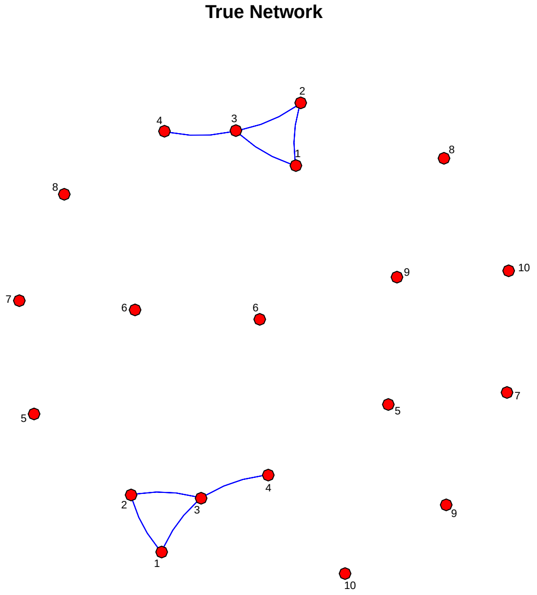


Figure 4.4. True networks.

model as in Section 3.5, i.e.:

$$[S_0 \sim F_1, N_0 \sim F_\Gamma, S_1 \sim F_T, N_1 \sim F_\Gamma, S_2 \sim F_T],$$

that implies that the networks at temporal lag 0 are constrained to be equal across the five observed time points. Moreover, the networks at temporal lag 1 are constrained to be equal across time, no links are presents between time t and time $t+2$ except for the self-self interactions, i.e. interactions between the same couple of genes. The recovered network structures after copula transformation are completely different from the recovered network structures from the non transformed data (see Section 3.5, Figure 5.6). This is due the non normality of the data which is shown in the approximate marginal densities in Figure 4.6 and 4.7. We use the non-canonical Gaussian copula graphical models proposed in Section 4.2 so data are transformed and the assumption of Gaussianity holds. The non-normality introduces spurious relationships which are reduced in our last analysis.

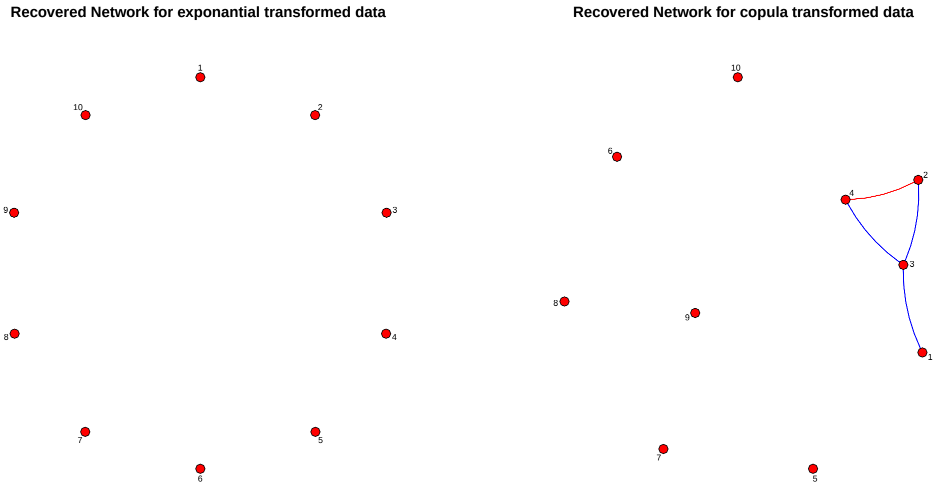

Recovered Network for exponantial transformed data
1
10
2
9
3
8
4
7
5
6
Recovered Network for copula transformed data
10
6
4
2
3
8
9
1
7
5

Figure 4.5. Recovered network structure at time one ignoring non-normaility (left) and implementing the copula method (rigth).

4.4 Summary

Large datasets often involve not continuous variables. Moreover, biological data are often continuous but non Gaussian. In order to avoid spurious relationship in the recovered or estimated graphs and to deal with mixed random variables we have introduced Gaussian copula graphical models. First of all we have seen that it is possible to estimate graphs for non normal but still continuous random variables. In order to solve the optimization problem when mixed random variables are considered we have used the Expectation Maximization algorithm. Then we have considered dynamic graphs. In particular, we have seen that it is possible to impose specific structures for estimating the graph in case we assume a non-canonical Gaussian copula graphical models. The non-canonical distribution it is necessary to consider constraints on the precision matrix instead that in the inverse of this matrix which in the Gaussian copula corresponds to a correlation matrix while in the non-canonical Gaussian copula is a covariance matrix. Finally, we have shown that these models can be applied for a real dataset analysis.

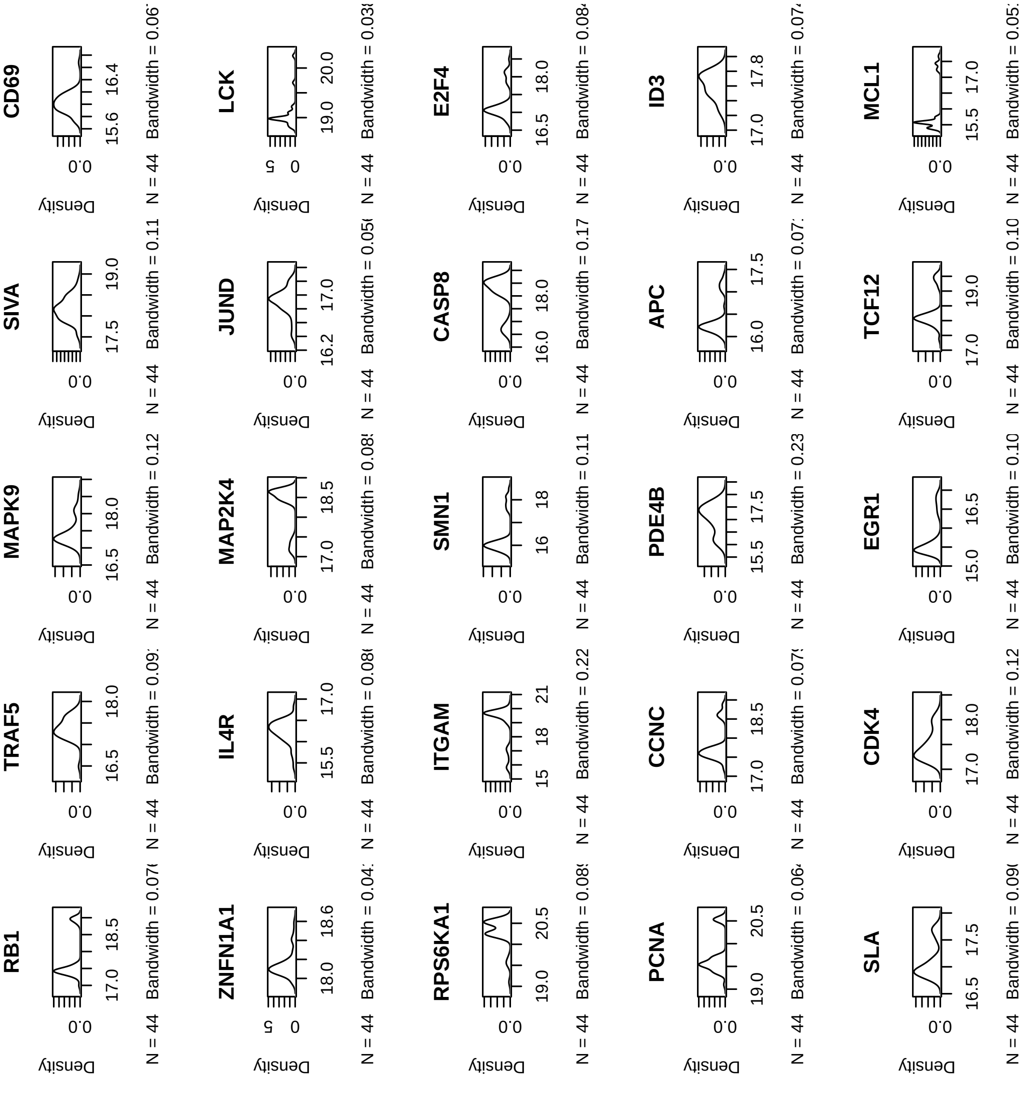


Figure 4.6. Approximate marginal density for the first 25 genes at time point one.

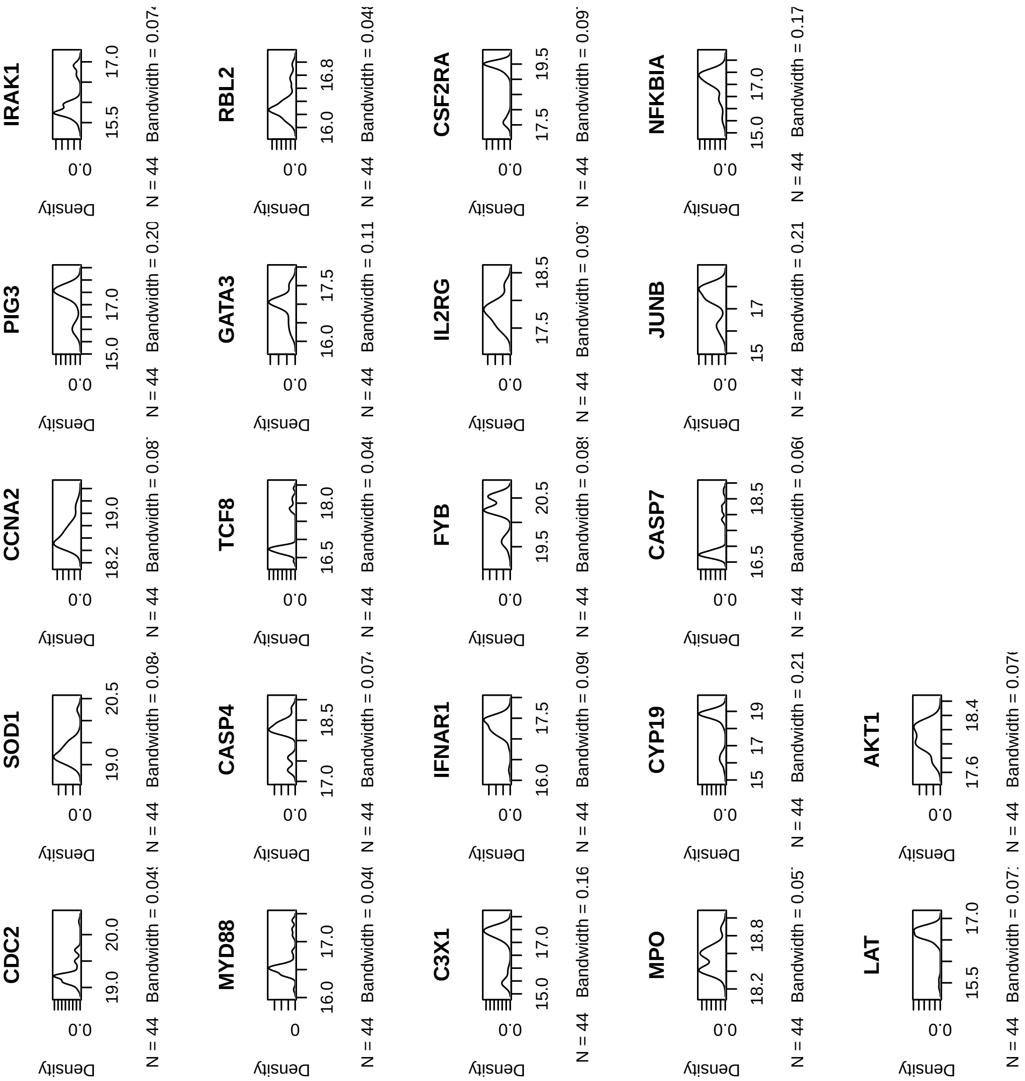


Figure 4.7. Approximate marginal density for genes 26 to 47 at time point one.

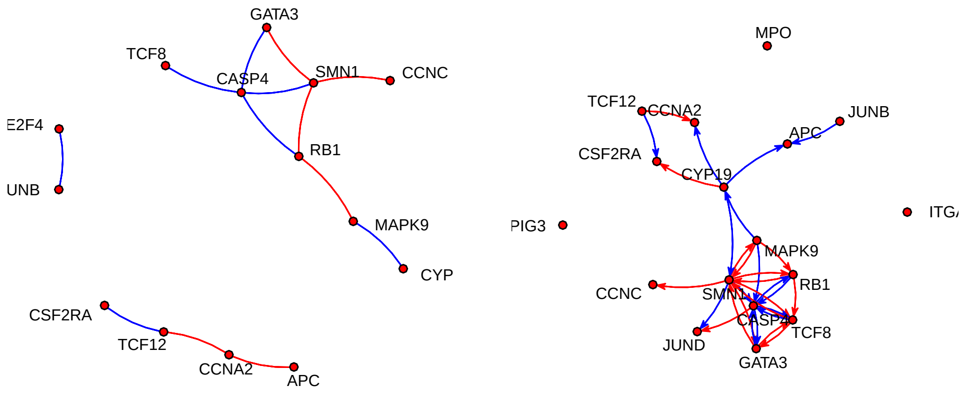


Figure 4.8. Recovered network structures after copula transformation. Representation of interactions between genes at temporal lag 0. Note that networks at lag 0 at time $1,2,3,\dots,5$ are equal since we impose $N_0 \sim F_\Gamma$ (left). Representation of interaction between genes at temporal lag 1. Note that networks at lag 1 between time $(1,2),(2,3),(3,4),(4,5)$ are equal since we impose $N_1 \sim F_\Gamma$ (right).

Chapter 5

Additional topics for dynamic network modelling

In this chapter we consider further characteristics of networks that can be modelled with Gaussian graphical models. In particular, we focus on modelling evolution of dynamic networks, scale free networks and partially unobserved networks.

Modelling of evolution of dynamic networks is about deciding which links are statistically different from a network at a given time point t to $t+i$, where t denotes temporal points in which random variables have been observed and i denotes some temporal lag in which we assume the changes happened. This is slightly different approach than the one we proposed in Section 5.1. Our aim is to develop an estimator for such Gaussian graphical models which is appropriate for data from several graphical models that share the same variables and some of the dependence structure. In this setting, estimating a single graphical model would mask the underlying heterogeneity, while estimating separate models for each time point does not take advantage of the common structure. We called this model sparse Gaussian graphical models for estimating evolution of networks (GL_{Δ}).

Scale-free networks are common network structures in biology. The scale free property indicates that we can reach a node in the networks from another in few steps. A node which is connected with many others is called hub node. For some experiment the scale-free assumption can be seen as a normal a priori assumption to estimate graphs. We propose a methodology to estimate these networks. We called this model graphical lasso for scale-free networks (GL_{sf}).

Partially unobservable networks are networks in which some nodes cannot be observed. This can happen for many reasons, for example, concepts cannot be observed in social science, or protein interaction and metabolic interaction cannot be directly observed when we consider microarray experiments in which the expression levels of some genes are collected. Relations between nodes can be the effect of spurious relations and our aim is to propose methods in which the effect of latent variable is separated from the effect of the observed variable so that the recovered networks contain as few spurious relationship as possible.

5.1 Sparse Gaussian graphical models for detecting evolution of networks

Gaussian graphical models explore dependence relationships between random variables, through the estimation of the corresponding inverse covariance matrices.

Suppose we have data from several categories that share the same variables but differ in their dependence structure, with some edges common across all categories and other edges unique to each category. For example, consider education data set (see Section3.1) in which different schools are considered. Or suppose we have data from several time points that share the same variables but differ in their dependence structure. For example, time-course t-cell dataset (see Section 3.1) in which several time points were consider and the experimental conditions are supposed to change from time point t to time $t+1$. In such cases, a common structure could hold at two different points but some pathways will be changed due to the different combinations. Investigating such changes and discovering common structure is an interesting challenge and it is useful in real application. However, the focus so far in the literature has been on estimating a single Gaussian graphical model Banerjee *et al.* (2008), Meinshausen and Bühlmann (2006), d'Aspremont *et al.* (2006) but in many applications it is more realistic to fit a collection of such models, due to the 'heterogeneity' of the data involved.

To accomplish this joint estimation, Guo *et al.* (2011) have proposed a method that links the estimation of separate graphical models through a hierarchical penalty. Its main advantage is the ability to discover a common structure and jointly estimate common links across graphs, which leads to improvements compared to fitting separate models, since it borrows information from other related graphs.

Copula Gaussian graphical models can be used to extend to graphical models with count, ordinal or binary data as well as mixed random variables (see Chapter 4). Höfling and Tibshirani (2009) and Ravikumar *et al.* (2010) proposed similar models to estimate the graph when categorical variables are considered.

In this Section, we propose a model to estimate dynamic graphs. The main idea is to impose the ℓ_1-norm to penalize changes in the networks through time or for different categories. Since we are mainly interested in time-course genetic data we focus on changes in time. Given a longitudinal graph $G=(V,E)$ the edge set E can be partitioned into natural partitions S_s, N_s, where S_s and N_s are interpretable as self-self interactions at lag s and networks interactions at lag s. Each of this subset can be further partitioned and we indicate with S_s^t and N_s^t these new subpartitions. S_s^t is the self-self term at lag s and time t and N_s^t is the network at lag s and time t. Consider a Gaussian graphical model $M = (G, \mathbb{P})$ where $\mathbb{P}$ is a multivariate normal distribution parametrized by $\mathbf{\Sigma}^{-1} = \mathbf{\Theta}$, and we consider the following decomposition of the precision matrix $\mathbf{\Theta}$:

$$\mathbf{\Theta} = \left[\begin{array}{cc|cc|cc|cc}
S_0^1 & N_0^1 & S_1^1 & N_1^1 & S_2^1 & N_2^1 & \dots & \dots \\
 & S_0^1 & N_1^1 & S_1^1 & N_2^1 & S_2^1 & \dots & \dots \\
\hline
 & & S_0^2 & N_0^2 & S_1^2 & N_1^2 & S_2^2 & N_2^2 \\
 & & & S_0^2 & N_1^2 & S_1^2 & N_2^2 & S_2^2 \\
\hline
 & & & & S_0^3 & N_0^3 & S_1^3 & N_1^3 \\
 & & & & & S_0^3 & N_1^3 & S_1^3 \\
\hline
 & & & & & & \ddots & \vdots \\
 & & & & & & & \ddots
\end{array}\right],$$

where S_s^t are self-self conditional correlations of the genes across time lag s and time t, and N_s^t is a genetic network with time lag s and time t. Our interest is detecting evolution of the networks, where the evolution is evaluated from the element-wise differences between N_s^t and N_s^{t+1}, i.e

$$N_s^t - N_s^{t+1}.$$

Our aim is to estimate ”significant” differences between these elements while the general structure is still sparse.

5.1.1 Maximum likelihood estimation for delta graphical lasso.

Suppose that $\mathbf{Y}^{(1)}, \dots, \mathbf{Y}^{(n)}$ with $\mathbf{Y}^{(i)} \in \mathbb{R}^{gt}$, where g is the number of random variables per each time point and t is the number of time points, are independent and identically distributed as a multivariate normal distribution with mean $\mathbf{0}$ and variance $\mathbf{\Sigma}$. We have seen in Chapter 2 that to reach sparse structures one can minimize a penalized log-likelihood (3.3). Moreover, we proposed copula Gaussian graphical models (see Chapter 4) to deal with mixed random variables. In the latter case the objective function to be minimized was given in (4.24). Let us indicate these optimization problems with the following objective function:

$$(\hat{\mathbf{\Theta}}) \quad := \quad \underset{\mathbf{\Theta}}{\operatorname{argmin}}\{-\log|\mathbf{\Theta}| + \operatorname{tr}(\mathbf{S\Theta}) + \lambda_1 \mathbf{x}^+ + \lambda_1 \mathbf{x}^-\} \tag{5.1}$$

$$\text{subject to} \qquad \mathbf{B}(\mathbf{\Theta}) - \mathbf{x}^+ + \mathbf{x}^- = \mathbf{0}$$
$$\mathbf{\Theta} \succ 0, \mathbf{x}^+, \mathbf{x}^- \geq \mathbf{0}.$$

where $\mathbf{B}$ indicates the usual ℓ_1 constraint, i.e. $||\mathbf{\Theta}||_1 \leq \rho_1$. Note that $\mathbf{x}^+$ and $\mathbf{x}^-$ are slack variables in $\mathbb{R}^m$ where $m = gt(gt-1)/2$. Here λ_1 is a smoothing parameter which regulates the sparsity in the precision matrix $\mathbf{\Theta}$ and $\mathbf{S}$ is given in (2.5) or (4.10) for sparse Gaussian graphical models or sparse copula Gaussian graphical models, respectively.

Now we want to penalize the difference between networks with lag s at time k and the same networks at time $k+1$, i.e.

$$||\Delta\mathbf{\Theta}||_1 = \sum_{s=0}^{t-1}\sum_{k=0}^{t-1} ||N_s^k - N_s^{k+1}||_1 = \sum_{s=0}^{t-1}\sum_{k=0}^{t-1}\sum_{i,j} |\theta_{(i,k),(j,k+s)} - \theta_{(i,k+1),(j,k+1)}| \leq \rho 2 \tag{5.2}$$

We want to take advantage from LogDetPPA so we need to write the linear map $\mathbf{A}$ such that the system of linear equations is included in the optimization problem (5.1). Since the inequality constraints needs to be converted in equality constraints we need to introduce another vector of slack variables in the optimization problem such that:

$$\sum_{s=0}^{t-1}\sum_{k=0}^{t-1}\sum_{i,j} |\theta_{(i,k),(j,k+s)} - \theta_{(i,k+1),(j,k+1)}| - y_k^+ + y_k^- = 0 \tag{5.3}$$

where $k = 1,\dots,K$, and $\mathbf{y}^+, \mathbf{y}^- \geq 0$. The optimization problem (5.1) is now written as:

$$\begin{aligned} (\hat{\mathbf{\Theta}}) \quad &:= \quad \underset{\mathbf{\Theta}}{\operatorname{argmin}}\{-\log|\mathbf{\Theta}| + \operatorname{tr}(\mathbf{S\Theta}) + \lambda_1\mathbf{x}^+ + \lambda_1\mathbf{x}^- + \lambda_2\mathbf{y}^+ + \lambda_2\mathbf{y}^-\} \\ \text{subject to} \quad & \quad \mathbf{B}(\mathbf{\Theta}) - \mathbf{x}^+ + \mathbf{x}^- = \mathbf{0} \\ & \quad \mathbf{A}(\mathbf{\Theta}) - \mathbf{y}^+ + \mathbf{y}^- = \mathbf{0} \\ & \quad \mathbf{\Theta} \succ 0, \mathbf{x}^+, \mathbf{x}^-, \mathbf{y}^+, \mathbf{y}^- \geq \mathbf{0}. \end{aligned} \tag{5.4}$$

The optimization problem (5.1) subject to (5.2) is a convex optimization problem which we have rewritten in a standard from.

It should be notice that both λ_1 and λ_2 are non-negative smoothing parameters that need to be selected. We consider a grid of values (λ_1, λ_2) and minimize information criterion scores such as AIC, AICc, and BIC. Then we use stability selection to select a more stable graph, i.e. we re-sample and select the graph such that an edge is present more than α times in the selection procedure (see Section 3.4 for further details or Meinshausen and Bühlmann (2010)).

Example: T-cell We apply Δ_{gl} to T-cell dataset where 4 genes and 2 time points were considered to show a small example of real data. Table 5.1 shows the estimated precision matrix. Here, we fixed tuning parameters $\lambda_1 = 0.01$ and $\lambda_2 = 0.1$.

Let's consider differences between elements of network at lag 0 at time 1 and at time 2, then Table 5.1 shows that ”significant” differences were estimated between ZNF-CCN and ZNF-SIV. While an edge was absent between ZNF and CCN at time 1 the same was present at lag 2. Opposite is the situation for ZNF-SIV.

Time		*1*				*2*			
	Gene	*ZNF*	*CCN*	*SIV*	*SCY*	*ZNF*	*CCN*	*SIV*	*SCY*
1	*ZNF*	1.24	0	-0.26	0.18	-0.22	-0.11	-0.11	-0.07
	CCN	-	1.49	0	-0.17	-0.18	-0.84	0.06	0.12
	SIV	-	-	1.44	0	-0.15	0.08	-0.69	-0.01
	SCY	-	-	-	1.19	0.02	0.13	0.41	-0.10
2	*ZNF*	-	-	-	-	1.07	-0.02	0	0.12
	CCN	-	-	-	-	-	1.55	0	0.24
	SIV	-	-	-	-	-	-	1.52	0
	SCY	-	-	-	-	-	-	-	1.08

Table 5.1. Conditional covariance $\hat{\boldsymbol{\Theta}}$ based on 44 replicates for 4 genes measured across 2 time points. $\lambda_1 = 0.01$ and $\lambda_2 = 0.1$.

5.1.2 Simulation study for delta graphical lasso model

We considered a simulation study to show the performance of the proposed model. Table 5.2 shows the simulation study scheme in which four different scenarios are studied. Here for different scenarios we mean that the number of nodes, links or time points change while the structure of the networks is the same.

ID	g	g*	t	p	n
1	20	0	3	60	50
2	-	20	-	120	-
3	-	40	-	180	-
4	-	60	-	240	-

Table 5.2. Simulation study scheme in which four scenarios are represented. The first column is an identification number, the second one indicates the number of variables per each time point (third column). The number of independent samples are represented in the last column.

For each scenario we simulate 100 datasets from a multivariate normal distribution with $\boldsymbol{\mu}$ equal to zero and $\boldsymbol{\Sigma}$ equal to the inverse of a precision matrix $\boldsymbol{\Theta}$. The structure of the graph changes across time. In fact, we want to consider graph with similar structures across some time points. Let us consider a graph with $gt \times gt$ nodes and m connections and let's say that these g nodes are observed at t time points. In order to build our matrix $\boldsymbol{\Theta}$ we start to build N_0^1, i.e. the network at lag 0 and time point 1. We can refer to this network as the starting point network. Then for N_0^2 we assume that few changes happened so that N_0^1 and N_0^2 are simular networks. For example we allow n_1 edges to be birth and n_2 edges to be death. We repeat this procedure for $t-1$ times and then we put the $N_{0,i}$ sub-matrices into the matrix $\boldsymbol{\Theta}$. Note that we assume networks in which $y_{i,k}$ and $y_{j,k+1}$ are independent so that N_i^l with $l \geq 2$ are filled with zeroes. We increase the number of nodes in the graph from scenarios 1 to 4. Random variables associated with these added nodes are independent. We keep the number of replicates and time points constants. The number of replicates is fewer than the number of random variables.

We take advantage of the R package `simone` to simulate networks with few changing points. The function `coNetworks` gives the opportunity to create such structures with $n = n_1 + n_2$ links different from a given structure. Note that we have implemented the constraints and used `R.Matlab` to connect Matlab and R. Table 5.3 shows the average of false positive, false negative and false discovery after 100 simulation were run. These results show that the model is reliable and it can be used for real applications when small changes in different time points are

		$\bar{FP}$	$\bar{FN}$	$\bar{FD}$	$\bar{FnD}$
1	AICc	0.0092	0.0811	0.2000	0.0031
	BIC	0.0363	0.0139	0.4873	0.0005
	AIC	0.0698	0.0069	0.6470	0.0003
2	AICc	0.0057	0.0447	0.2899	0.0006
	BIC	0.0088	0.0321	0.3826	0.0005
	AIC	0.0437	0.0041	0.7514	0.0001
3	AICc	0.0016	0.4585	0.2730	0.0036
	BIC	0.0016	0.4585	0.2730	0.0036
	AIC	0.0288	0.1452	0.8088	0.0012
4	AICc	0.0091	0.1034	0.1680	0.0052
	BIC	0.0396	0.0517	0.4527	0.0027
	AIC	0.0670	0.0000	0.5704	0.0000

Table 5.3. The average of the proportions of how many links have been correctly estimated were calculated by the False Positive (FP), False Negative (FN), False Discovery (FD) and False not Discovery (FnD).

present. Indeed, the assumption given in Section 2.3 should still be satisfied.

5.1.3 Application of difference graphical lasso to Tcell

In this subsection we consider human T-cell dataset and apply Δ_{gl}. We assume that genes $Y_{s,t}$ and $Y_{s,t+2}$ are conditional independent given the rest. This means that the edge set for networks at lag 2, i.e. N_2, is an empty set. We define the following Δ_{gl} model:

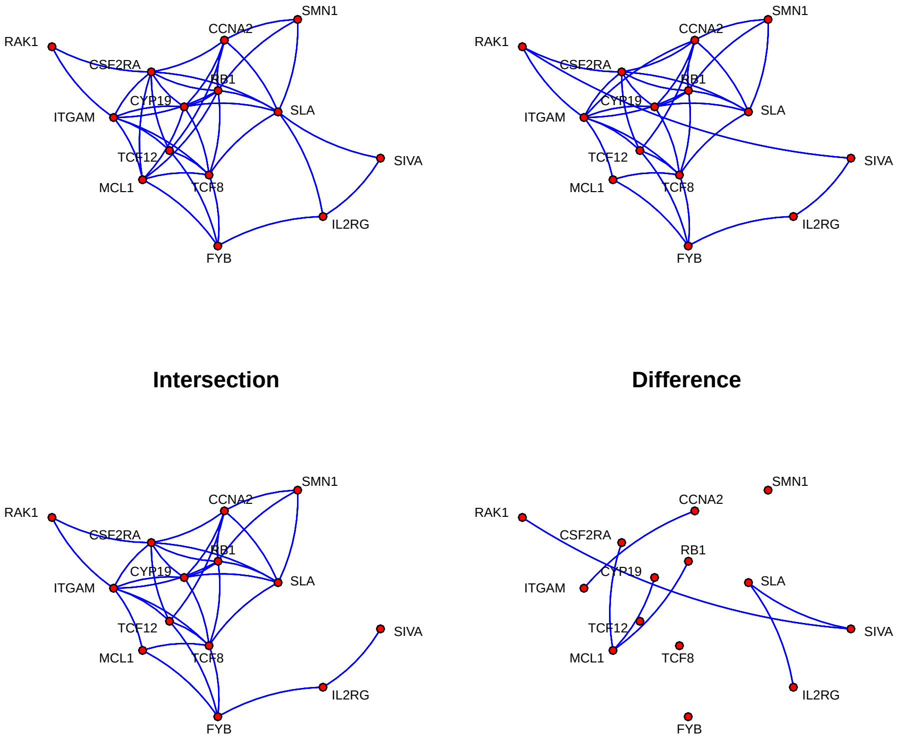
$$[N_1 \prec 1, N_2 \prec 0],$$

where we allow the following constraints $N_i \prec 0, 1$ or 2, i.e elements in N_i are all zero, penalized as described in formula (5.2), or 2 no constraint is imposed. Figures 5.1, 5.2, 5.3 are obtained from the estimation procedure where two graphs (upper-left, upper right), intersection (bottom-left) and difference (bottom-right) between time 1 and time 2, time 2 and time 3, and time 3 and time 4 are represented.

Once a graph has been estimated several questions on how to analyse time-evolution networks might be posed. In what way do new entities enter a network? Does the network retain certain graph properties as it grows and evolves? Does the graph undergo a phase transition, in which its behaviour suddenly changes? In answering these questions it is of interest to have a diagnostic tool for tracking graph properties and noting anomalies and graph characteristics of interest. We take advantage of ADAGE (McGlohon and Faloutsos, 2007), a software package that analyse: number of edges over time, number of nodes over time, densification law, eigenvalues over increasing nodes, size of largest connected component vs. nodes, number of connected components vs. nodes and time, comparative sizes of connected components over time. The densification law states that number of edges vs. number of nodes should follow a power law. The eigenvalues of a graph are of interest, as they might indicate a phase transition. The sizes and numbers of connected components are also indicative of phase transitions. One would expect the last item to follow power laws with the same slope in each time step.

5.2 Sparse Gaussian graphical models for scale-free networks

Scale-free networks are quite common structure in networks biology. These networks are assumed to own prior internal structures of connectivity which drive the inference method. Ambroise et al. (2009) provided a method that looks for sparse solutions, but also for an internal structure of the network that drives the inference. Indeed, biological networks and particularly gene regulation networks are known not only to be sparse, but also organized, so that nodes belong to different classes



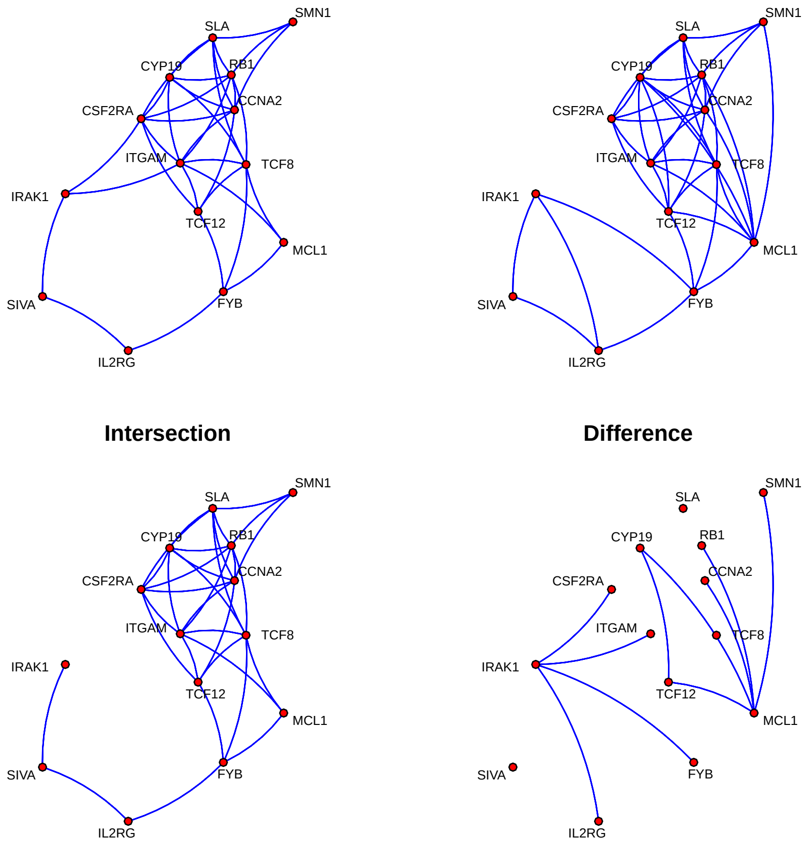
Intersection

Difference

Figure 5.1. Graph, intersection and difference between time 1 and time 2

of connectivity. Thus, they suggested a criterion that takes this into account. The internal structure considered relies on affiliation networks. That is, genes are clustered into groups that share the same connectivity patterns. This can be seen as the analogous to the group-LASSO (Yuan and Lin, 2007) applied to a graphical context.

Opgen-Rhein and Strimmer (2007), L'ebre (2009), Shimamura et al. (2009) assumed a first-order vector auto-regressive (VAR1) model for the time course data generation, they provided inference methods handling high-dimensional settings. In particular, Opgen-Rhein and Strimmer suggested a shrinkage estimate while L'ebre performed statistical tests on limited-order partial correlations to select sig-

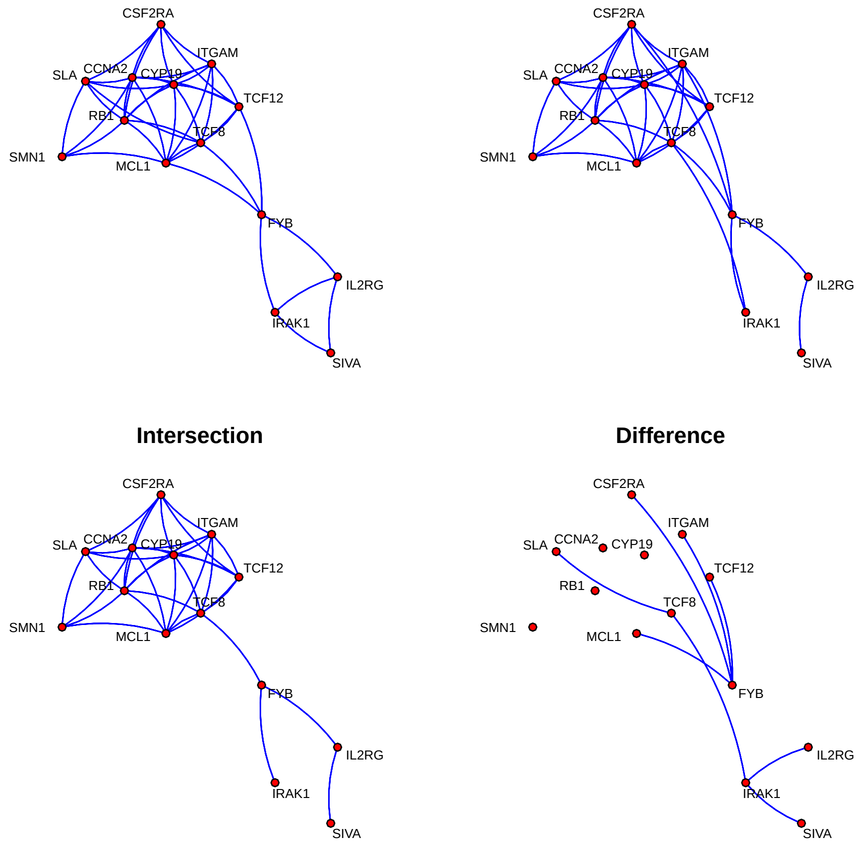


Intersection

Difference

Figure 5.2. Graph, intersection and difference between time 2 and time 3

nificant edges. In a recent work, Shimamura et al. (2009) proposed to deal with this VAR1 setup by combining ideas from two major developments of the LASSO to define the Recursive elastic-net. As an elastic-net (Zou and T. 2005), this method adds an ℓ_2 penalty to the original ℓ_1 regularization, thus encouraging the simultaneous selection of highly correlated covariates on top of the automatic selection process due to the ℓ_1 norm. As in the adaptive-LASSO (Zou 2006), weights are corrected on the basis of the former estimate so as to adapt the regularization parameter to the relative importance of coefficients. Note that, in this context, we are no longer looking for an estimate of the inverse of the covariance matrix but of the parameters of the VAR1 model, which leads to a directed graph. In this section, we


Intersection

Difference

Figure 5.3. Graph, intersection and difference between time 3 and time 4

consider scale-free networks which we have discussed in Subsection 2.5.2. Here we want to recall that the degree of a scale free networks for each node should follow a power law distribution. We want to build constraints such that the number of connections among a node and the rest are as few as possible, i.e.

$$\sum_{g=1}^{G} \#\{\Theta_{gj} \neq 0 \mid j \in G\}^{-\lambda_2} \leq \rho_2. \tag{5.5}$$

5.3 Partially unobserved networks

Suppose we have a sample of a subset of a collection of random variables. No additional information is provided about the number of latent variables, nor of the relationship between the latent and the observed variables. Is it possible to discover the number of hidden components, and estimate a graphical model over the entire collection of variables? In this section, we address this question to different prospectives which brings two classes of graphical models: state space models, and Gaussian graphical models with latent variables.

First of all we consider linear-Gaussian state space models (SSMs) to estimate model interaction parameters. Expectation maximization algorithm (EM) proposed by Dempster *et al.* (1977) combined with the Kalman smoothing algorithm (Beal *et al.*, 2005; Ghahramani and Hinton, 1996) are necessary to have maximum likelihood estimates. SSMs and Kalman filtering have been widely used in modelling dynamic Bayesian networks. Model selection, which involves determining a suitable dimension of the hidden states, is an additional challenge when hidden variables are present into the system. Beal *et al.* (2005) approached the problem of deciding on a suitable dimension of the hidden states through cross validation. They increase the dimension of the hidden states and monitor the predictive likelihood using the test data. One major drawback of this approach is that it is very slow and not suitable for high-dimensional frameworks.

Alternatively, we propose dynamic Gaussian graphical models to estimate genetic networks with latent variables. Our aim is to estimate the set of conditional independence parameters for a Gaussian graphical model while we are taking into account for effects of hidden components. We take advantage of the model proposed by Chandrasekaran *et al.* (2010) and extend this model for dynamic networks.

5.3.1 State space models for latent variables

Linear Gaussian state space models are a class of dynamic Bayesian networks that relate observations measurements to hidden variables. Let $M = (G, \mathbb{P})$ be a dynamic Bayesian network, where G is a directed acyclic graph and $\mathbb{P}$ is a multivariate probability distribution. Consider a vector $\mathbf{y} = (\mathbf{y}_1, \dots, \mathbf{y}_T)$ of observations, where $\mathbf{y}_t$ is a p-dimensional vector, and let $\mathbf{x} = (\mathbf{x}_1, \dots, \mathbf{x}_T)$ be a vector of hidden variables. Assume that the evolution of the hidden variables $\mathbf{x}_t$ $t = 1, \dots, T$ follows a first-order Markov process plus a Gaussian noise $\mathbf{w}_t$, then Holmes (2010)

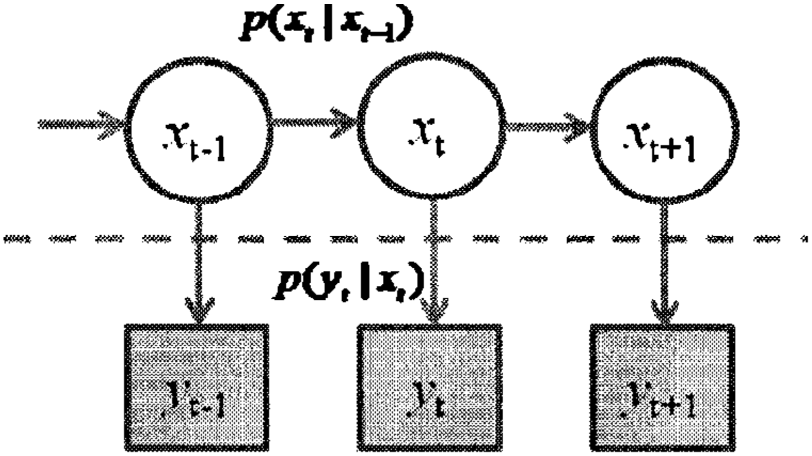


Figure 5.4. Graphical representetion for a state space model

consider the following multivariate autoregressive state space model (MARSS),

$$\begin{cases} \mathbf{x}_t = \mathbf{B}\mathbf{x}_{t-1} + \mathbf{u} + \mathbf{w}_t, & \text{where } \mathbf{w}_t \sim N(\mathbf{0}, \mathbf{Q}), \\ \mathbf{y}_t = \mathbf{Z}\mathbf{x}_t + \mathbf{a} + \mathbf{v}_t, & \text{where } \mathbf{v}_t \sim (\mathbf{0}, \mathbf{R}), \\ \mathbf{x}_\mathbf{0} \sim N(\mathbf{0}, \mathbf{V}_\mathbf{0}), & \end{cases} \tag{5.6}$$

where $\mathbf{Z}$ is a $(p \times k)$ transition matrix from state t to state $t+1$ and models the influence of the gene expression values from previous time steps on the hidden states, and $\mathbf{B}$ describes the temporal development of the regulators or the evolution of the transcription factors from previous time step $t-1$ to the current time step t and is of dimension $(k \times k)$. It provides key information on the influences of the hidden regulators on each other. The state dynamics in the model equation (5.6) assumes that hidden states $\mathbf{x}_t \in \mathbb{R}^k$ follow a genetic expression with some stochastic aspects and external inputs $\mathbf{u}$ with i.i.d. process noise $\mathbf{w}_t \sim N(\mathbf{0}, \mathbf{Q})$. The dynamic observations $\mathbf{y}_t \in \mathbf{R}^p$ also defined via the model equation (5.6) are linear combination of the hidden state and the external inputs $\mathbf{a}$ with i.i.d. Gaussian measurement noise $\mathbf{v}_t \sim N(\mathbf{0}, \mathbf{R})$. Note that hidden variables $\mathbf{x}_t$ are not directly accessible but they rather are related to the observed data vector $\mathbf{y}_t$. For time-course genetic data y_{ij} could represent expression level of gene i at time j and $\mathbf{x}_t$ the quantities of RNA produced by genes at time t.

Figure 5.4 shows a graphical representation for a state space model. In gene regulation, equation (5.6) implies that the observed gene expression equation $\mathbf{y}_t$ is a linear function of the protein transcription factor $\mathbf{x}_t$ which itself describe a linear function of the previous ones. The model describes two fundamental stages in gene regulation which is in conformity with the central dogma, i.e. DNA does not code for protein directly but rather acts through two stages, namely transcription and translation. The model acts as a feed-back loop in the following way. DNA is the molecular storehouse of genetic information, and mRNA is transcribed from DNA by enzymes called RNA polymerases and it is generally further processed by other enzymes. RNA having moved outside the nucleus, attaches to ribosome and is translated to proteins.

Ghahramani and Hinton (1996) derive EM algorithm for unconstrained MARSS model. This EM algorithm was originally derived by Shumway and Stoffer (1982). They extend the derivation to the case of a constrained MARSS model where one may impose to be equal shared elements in the parameter matrices. The algorithm consists of an expectation step (E-step), which computes the expected values of the hidden states using the Kalman filter/smoother, combined with a maximization step (M-step), which computes the maximum-likelihood estimates of the parameters given the data and the expected values of the hidden states.

Likelihood for SSM for latent variables

Let's consider model (5.6), the joint log-likelihood of the data and hidden states is

$$\begin{aligned} l(\boldsymbol{\psi};\mathbf{y}_1^T,\mathbf{x}_1^T) &= -\sum_{t=1}^{T}\frac{1}{2}(\mathbf{y}_t-\mathbf{Z}\mathbf{x}_t-\mathbf{a})'\mathbf{R}^{-1}(\mathbf{y}_t-\mathbf{Z}\mathbf{x}_t-\mathbf{a})-\frac{T}{2}\log|\mathbf{R}| \\ &- \sum_{t=2}^{T}\frac{1}{2}(\mathbf{x}_t-\mathbf{B}\mathbf{x}_{t-1}-\mathbf{u})'\mathbf{Q}^{-1}(\mathbf{x}_t-\mathbf{B}\mathbf{x}_{t-1}-\mathbf{u})-\frac{T-1}{2}\log|\mathbf{Q}| \\ &- \frac{1}{2}\mathbf{x}_0'\mathbf{V}_0^{-1}\mathbf{x}_0-\frac{1}{2}\log|\mathbf{V}_0|-\frac{n}{2}\log 2\pi, \end{aligned} \tag{5.7}$$

where $\mathbf{y}_1^T$ is shorthand for all the data from time $t=1$ to $t=T$, and n is the number of data points. Formula (5.7) comes from the likelihood function for a multivariate normal distribution since $\mathbf{X}_t|\mathbf{x}_{t-1}$ has a multivariate normal distribution and $\mathbf{Y}_t|\mathbf{x}_t$ has a multivariate normal distribution. Here $\mathbf{X}_t$ denotes the random variable hidden states at time t and $\mathbf{x}_t$ is a realization from that random variable. We expand out the terms in the joint log-likelihood

$$\begin{aligned}
& l(\boldsymbol{\psi};\mathbf{y}_1^T,\mathbf{x}_1^T) = \\
- & \frac{1}{2}\sum_{t=1}^{T}[\mathbf{y}_t'\mathbf{R}^{-1}\mathbf{y}_t - \mathbf{y}_t'\mathbf{R}^{-1}\mathbf{Z}\mathbf{x}_t - (\mathbf{Z}\mathbf{x}_t)'\mathbf{R}^{-1}\mathbf{y}_t - \mathbf{a}'\mathbf{R}^{-1}\mathbf{y}_t \\
- & \mathbf{y}_t'\mathbf{R}^{-1}\mathbf{a} + (\mathbf{Z}\mathbf{x}_t)'\mathbf{R}^{-1}\mathbf{Z}\mathbf{x}_t + \mathbf{a}'\mathbf{R}^{-1}\mathbf{Z}\mathbf{x}_t + (\mathbf{Z}\mathbf{x}_t)'\mathbf{R}^{-1}\mathbf{a} \\
+ & a'\mathbf{R}^{-1}\mathbf{a}] - \frac{T}{2}\log|\mathbf{R}| \qquad (5.8) \\
- & \frac{1}{2}\sum_{t=2}^{T}[\mathbf{x}_t'\mathbf{Q}^{-1}\mathbf{x}_t - \mathbf{x}_t'\mathbf{Q}^{-1}\mathbf{B}\mathbf{x}_{t-1} - (\mathbf{B}\mathbf{x}_{t-1})'\mathbf{Q}^{-1}\mathbf{x}_t - \mathbf{u}'\mathbf{Q}^{-1}\mathbf{x}_t \\
- & \mathbf{x}_t'\mathbf{Q}^{-1}\mathbf{u} + (\mathbf{B}\mathbf{x}_{t-1})'\mathbf{Q}^{-1}\mathbf{B}\mathbf{x}_{t-1} + \mathbf{u}'\mathbf{Q}^{-1}\mathbf{B}\mathbf{x}_{t-1} + (Bx_{t-1})'\mathbf{Q}^{-1}\mathbf{u} \\
+ & \mathbf{u}'\mathbf{Q}^{-1}\mathbf{u}] - \frac{T-1}{2}\log|\mathbf{Q}| \\
- & \frac{1}{2}(\mathbf{x}_0'\mathbf{V}_0^{-1}\mathbf{x}_0) - \frac{1}{2}\log|\mathbf{V}_0| - \frac{n}{2}\log 2\pi
\end{aligned}$$

Joint parameter estimation via EM algorithm. Given the joint log-likelihood 5.8, one wants to maximize it with respect to $\boldsymbol{\psi}$, i.e.

$$(\hat{\boldsymbol{\psi}}) := \text{argmax}_{\psi} l(\boldsymbol{\psi};\mathbf{y}_1^T,\mathbf{x}_1^T),$$

where $\boldsymbol{\psi} = [\mathbf{Z},\mathbf{B},\mathbf{u},\mathbf{a},\mathbf{Q},\mathbf{R},\mathbf{V}_0]$.

Several approaches have been proposed to estimate the vector parameter $\boldsymbol{\psi}$ that maximizes (5.8). We take advantage of EM algorithm. Note that, if we had had the complete data $\{\mathbf{y}_1^T,\mathbf{x}_1^T\}$, MLEs would have been calculated by using multivariate normal theory. However, we do not have the complete data so we need to use an iterative method for finding MLE of $\boldsymbol{\psi}$. Observed data $\mathbf{y}_t$ are used by successively maximizing the conditional expectation of the complete data likelihood given the observed values.

Given the estimated parameter vector $\boldsymbol{\psi}$, we obtain some useful interpretation of biological system networks. For example the magnitude of the effect of proteins on RNA (part of transcription process), and also estimate networks between the proteins (transcription factors).

The E-step. The EM-algorithm for SSMs was formulated by Dempster *et al.* (1977). The algorithm requires the computation of the conditional expectation of the log-likelihood given the complete data. The likelihood function that is maximized in the M-step is the expected log-likelihood function where the expectation is taken over $(\mathbf{X}_1^T|\mathbf{y}_1^T)$, meaning the set of all possible hidden states $(\mathbf{X}_1,\ldots,\mathbf{X}_T)$

conditioned to all the data $(\mathbf{y}_1, \dots, \mathbf{y}_T)$. We denote the expected log-likelihood by $H(\boldsymbol{\psi}; \mathbf{x}_1^T, \mathbf{y}_1^T)$. Using the log-likelihood equation (5.16), the algorithm cycles iteratively between an expectation step followed by a maximization step. In the expectation step, the expected values of the hidden states conditioned all the data and to a set of parameters at iteration i, ψ, are computed using the Kalman smoother. The output from the Kalman smoother provides

$$\tilde{\mathbf{x}}_t = \mathbb{E}_{X|y}(\mathbf{X}_t | \mathbf{y}_1^T, \hat{\boldsymbol{\psi}}_i). \tag{5.9}$$

$$\tilde{V}_t = \mathrm{Var}(\mathbf{X}_t | \mathbf{y}_1^T, \hat{\boldsymbol{\psi}}_i). \tag{5.10}$$

$$\tilde{V}_{t,t-1} = \mathrm{Cov}(\mathbf{X}_t, \mathbf{X}_{t-1} | \mathbf{y}_1^T, \hat{\boldsymbol{\psi}}_i). \tag{5.11}$$

From $\tilde{\mathbf{x}}_t$, $\tilde{\mathbf{V}}_t$, and $\tilde{\mathbf{V}}_{t,t-1}$, we can compute

$$\tilde{P}_t = \mathbb{E}_{X|y}(\mathbf{X}_t \mathbf{X}_t' | \mathbf{y}_1^T, \hat{\boldsymbol{\psi}}_i) = \tilde{\mathbf{V}}_t + \tilde{\mathbf{x}}_t \tilde{\mathbf{x}}_t'. \tag{5.12}$$

$$\tilde{P}_{t,t-1} = \mathbb{E}_{X|y}(\mathbf{X}_t \mathbf{x}_{t-1}' | \mathbf{y}_1^T, \hat{\psi}_i) = \tilde{\mathbf{V}}_{t,t-1} + \tilde{\mathbf{x}}_t \tilde{\mathbf{x}}_{t-1}'. \tag{5.13}$$

The subscript on the expectation, $\mathbb{E}$, denotes that the expectation is taken over the hidden states, X, conditioned to the observed data, $\mathbf{y}$. The right sides of equations (5.10) and (5.11) arise from the computational formula for variance and covariance:

$$\mathrm{Var}(X) = \mathbb{E}(XX') - \mathbb{E}(X)\mathbb{E}(X)'. \tag{5.14}$$

$$\mathrm{Cov}(X,Y) = \mathbb{E}(XY') - \mathbb{E}(X)\mathbb{E}(Y)'. \tag{5.15}$$

The M-Step. In the maximization step, a new parameter set $\hat{\boldsymbol{\psi}}_{i+1}$ is computed by finding the parameters that maximize the expected log-likelihood function (5.16) using $\tilde{\mathbf{x}}_t$, $\tilde{P}_t$ and $\tilde{P}_{t,t-1}$ from iteration i. The equations that give the parameters for the next iteration $(i+1)$ are called the update equations. After one iteration of the expectation and maximization steps, the cycle is then repeated. New $\tilde{\mathbf{x}}_t$, $\tilde{P}_t$ and $\tilde{P}_{t,t-1}$ are computed using $\boldsymbol{\psi}^{i+1}$, and then a new set of parameters $\boldsymbol{\psi}^{i+2}$ is generated. This cycle is continued until the likelihood no more increases than a specified tolerance level. This algorithm is guaranteed to increase likelihood at each iteration (if it does not, it means there is an error in update equation). The algorithm must be started from an initial set of parameter values $\boldsymbol{\psi}^1$. The algorithm is not particularly sensitive to the initial conditions but the surface could definitely be multi-modal and have local maxima. The likelihood function that is maximized in the M-step is the expected log-likelihood function where the expectation is taken over $(\mathbf{X}_1^T | \mathbf{y}_1^T)$, meaning the set of all possible hidden states conditioned on all the data. We denote the expected log-likelihood by $H(\boldsymbol{\psi}; \mathbf{x}_1^T, \mathbf{y}_1^T)$. Using the log-likelihood equation

(5.8), $H(\boldsymbol{\psi};\mathbf{x}_1^T,\mathbf{y}_1^T)$ is:

$$\begin{aligned}
&\mathbb{E}_{X|y}l(\boldsymbol{\psi};\mathbf{y}_1^T,\mathbf{x}_1^T) = \\
-\ &\frac{1}{2}\sum_{t=1}^{T}[\mathbf{y}_t'\mathbf{R}^{-1}\mathbf{y}_t - \mathbb{E}_{X|y}[\mathbf{y}_t'\mathbf{R}^{-1}\mathbf{Z}\mathbf{x}_t] - \mathbb{E}_{X|y}[(\mathbf{Z}\mathbf{x}_t)'\mathbf{R}^{-1}\mathbf{y}_t] - \mathbf{a}'\mathbf{R}^{-1}\mathbf{y}_t \\
-\ &\mathbf{y}_t'\mathbf{R}^{-1}\mathbf{a} + \mathbb{E}_{X|y}[(\mathbf{Z}\mathbf{x}_t)'\mathbf{R}^{-1}\mathbf{Z}\mathbf{x}_t] + \mathbb{E}_{X|y}[\mathbf{a}'\mathbf{R}^{-1}\mathbf{Z}\mathbf{x}_t] + \mathbb{E}_{X|y}[(\mathbf{Z}\mathbf{x}_t)'\mathbf{R}^{-1}\mathbf{a}] \\
+\ &a'\mathbf{R}^{-1}\mathbf{a}] - \frac{T}{2}\log|\mathbf{R}| \qquad (5.16) \\
-\ &\frac{1}{2}\sum_{t=2}^{T}[\mathbb{E}_{X|y}[\mathbf{x}_t'\mathbf{Q}^{-1}\mathbf{x}_t] - \mathbb{E}_{X|y}[\mathbf{x}_t'\mathbf{Q}^{-1}\mathbf{B}\mathbf{x}_{t-1}] - \mathbb{E}_{X|y}[\mathbf{B}\mathbf{x}_{t-1})'\mathbf{Q}^{-1}\mathbf{x}_t] - \mathbb{E}_{X|y}[\mathbf{u}'\mathbf{Q}^{-1}\mathbf{x}_t] \\
-\ &\mathbb{E}_{X|y}[\mathbf{x}_t'\mathbf{Q}^{-1}\mathbf{u}] + \mathbb{E}_{X|y}[\mathbf{B}\mathbf{x}_{t-1}'\mathbf{Q}^{-1}\mathbf{B}\mathbf{x}_{t-1}] + \mathbb{E}_{X|y}[\mathbf{u}'\mathbf{Q}^{-1}\mathbf{B}\mathbf{x}_{t-1}] + \mathbb{E}_{X|y}[\mathbf{B}\mathbf{x}_{t-1}'\mathbf{Q}^{-1}\mathbf{u}] \\
+\ &\mathbf{u}'\mathbf{Q}^{-1}\mathbf{u}] - \frac{T-1}{2}\log|\mathbf{Q}| \\
-\ &\frac{1}{2}(\mathbf{x}_0'\mathbf{V}_0^{-1}\mathbf{x}_0) - \frac{1}{2}\log|\mathbf{V}_0| - \frac{n}{2}\log 2\pi.
\end{aligned}$$

We will refer to the expected log-likelihood throughout our derivation of the update equations; it could be written more concisely, but for deriving the update equations, we will keep this long form. The new parameters for the maximization step are those parameters that maximize the expected log likelihood $H(\boldsymbol{\psi};\mathbf{x}_1^T,\mathbf{y}_1^T)$. The equations for these new parameters are termed the update equations.

The unconstrained update equations In this paragraph, we show the derivation of the update equations when all elements of a parameter matrix are estimated and are all allowed to be different; these are the update equations one can see in Shumway and Stoffer (1982). If some of the values are fixed or are shared, the derivations are similar but they get more cluttered. Holmes (2010) shows the general update equations when there are fixed or shared values in the parameter matrices. The general update equations are used in the MARSS R package. To derive the update equations, we will find the parameters values that maximize $H(\boldsymbol{\psi};\mathbf{x}_1^T,\mathbf{y}_1^T)$ by partial differentiation of $H(\boldsymbol{\psi};\mathbf{x}_1^T,\mathbf{y}_1^T)$ with respect to the parameters of interest, and then solve for the parameters value that sets the partial derivatives to zero. The partial differentiation is with respect to each individual parameter element, for example each $\mathbf{u}_j$ in the vector $\mathbf{u}$. The idea is to single out those terms in equation (5.16) that involve $\mathbf{u}_j$ (say), differentiate by $\mathbf{u}_j$, set this to zero and solve for $\mathbf{u}_j$. This gives the new $\mathbf{u}_j$ that maximizes the partial derivative with respect to $\mathbf{u}_j$ of the expected log-likelihood. Matrix calculus gives us a way to jointly maximize $H(\boldsymbol{\psi};\mathbf{x}_1^T,\mathbf{y}_1^T)$ with respect to all elements in a parameter vector or matrix. Deriving the update equations is tedious. We show the update equation for **u** while the rest can be found in Holmes (2010).

The partial derivative of a scalar function with respect to some column vector **b** (which has elements $b_1, b_2, \ldots$) is

$$\frac{\partial H}{\partial \mathbf{b}} = \left(\frac{\partial H}{\partial b_1} \frac{\partial H}{\partial b_2} \cdots \frac{\partial H}{\partial b_n} \right) \tag{5.17}$$

Note that the derivative of a column vector b is a row vector. The partial derivatives of a scalar function with respect to some $n \times n$ matrix **B** is

$$\frac{\partial H}{\partial \mathbf{B}} = \begin{pmatrix} \frac{\partial H}{\partial b_{1,1}} & \frac{\partial H}{\partial b_{2,1}} & \cdots & \frac{\partial H}{\partial b_{n,1}} \\ \frac{\partial H}{\partial b_{1,2}} & \frac{\partial H}{\partial b_{2,2}} & \cdots & \frac{\partial H}{\partial b_{n,2}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial H}{\partial b_{1,n}} & \frac{\partial H}{\partial b_{2,n}} & \cdots & \frac{\partial H}{\partial b_{n,n}} \end{pmatrix}$$

Note that **Q** and **R** are symmetric matrix while **B** and **Z** may not be symmetric.

The update equation for u. Take the partial derivative of $H(\boldsymbol{\psi}; \mathbf{x}_1^T, \mathbf{y}_1^T)$ with respect to **u**, which is a $m \times 1$ column vector. All parameters other than **u** are fixed to constant values (because we are doing partial differentiation). Since the derivative of a constant is 0, terms not involving **u** will equal 0 and drop out. The subscript, $X|y$, on the expectation, $\mathbb{E}$, has been dropped to remove clutter. By taking the derivative of equation (5.16) with respect to **u**:

$$\begin{aligned} \frac{\partial H}{\partial \mathbf{u}} &= -\frac{1}{2}\sum_{t=2}^{T}(-\mathbb{E}[\partial(\mathbf{x}_t'\mathbf{Q}^{-1}\mathbf{u}/\partial\mathbf{u})] - \mathbb{E}[\partial(\mathbf{u}'\mathbf{Q}^{-1}\mathbf{x}_t)/\partial\mathbf{u}] && (5.18) \\ &+ \mathbb{E}[\partial((\mathbf{B}\mathbf{x}_{t-1})'\mathbf{Q}^{-1}\mathbf{u}/\partial\mathbf{u}] + \mathbb{E}[\partial(\mathbf{u}'\mathbf{Q}^{-1}\mathbf{B}\mathbf{x}_{t-1})/\partial\mathbf{u}] && (5.19) \\ &+ \partial(\mathbf{u}'\mathbf{Q}^{-1}\mathbf{u})/\partial\mathbf{u}. && (5.20) \end{aligned}$$

Using relations (1) and (2) in Appendix A and using $\mathbf{Q}^{-1} = (\mathbf{Q}^{-1})'$, we have

$$\begin{aligned} \partial H/\partial\mathbf{u} &= \frac{1}{2}\sum_{t=2}^{T}(-\mathbb{E}[(\mathbf{x}_t)'\mathbf{Q}^{-1}] - \mathbb{E}[(\mathbf{Q}^{-1}\mathbf{x}_t)']) && (5.21) \\ &+ \mathbb{E}[(\mathbf{B}\mathbf{x}_{t-1}'\mathbf{Q}^{-1})] + \mathbb{E}[(\mathbf{Q}^{-1}\mathbf{B}\mathbf{x}_{t-1})'] + 2\mathbf{u}'\mathbf{Q}^{-1}. && (5.22) \end{aligned}$$

Set the left side to zero (a $1 \times m$ matrix of zeros) and transpose the whole equation. $\mathbf{Q}^{-1}$ cancels out by multiplying on the left by **Q** (left since we just transposed the whole equation), giving

$$\mathbf{0} = \sum_{t=2}^{T}(\mathbb{E}[\mathbf{x}_t] - \mathbf{B}\mathbb{E}[\mathbf{x}_{t-1}] - \mathbf{u}) = \sum_{t=2}^{T}(\mathbb{E}[\mathbf{x}_{t-1}] - \mathbf{B}\mathbb{E}[\mathbf{x}_{t-1}] - (T-1)\mathbf{u}). \tag{5.23}$$

Solving for **u** and replacing the expectations with the Kalman smoother output, gives us the new **u** that maximizes H,

$$\mathbf{u}_{new} = \frac{1}{T-1}\sum_{t=2}^{T}(\tilde{x}_t - \mathbf{B}\tilde{x}_{t-1}). \tag{5.24}$$

Choice of hidden state dimension: AIC Model selection or the determination of the optimum dimension of the hidden state is a complex but important in the application of SSMs to networks re-construction. Most popular criterion for model selection include AIC and the BIC. We apply AIC method for our model selection. Given the log-likelihood function (5.16), AIC for a model with k-dimensional state vector is given by:

$$AIC(k) = 2l(\hat{\boldsymbol{\psi}}_k; \mathbf{y}_t, \mathbf{x}_t|) + 2e,$$

with e is the number of estimated parameters, and $l(\hat{\boldsymbol{\psi}}_k; \mathbf{y}_t, \mathbf{x}_t)$ the log-likelihood of the observed data. We settle on the hidden state dimension that has the minimum AIC, i.e we find k such that

$$(\hat{k}) := \text{argmin}_k\{AIC(k)\}.$$

In this case, we increase the number of hidden states and monitor the AIC.

5.3.2 Gaussian graphical models for latent variables

We have considered fixed structure and fixed number of hidden states so far. In this section we are going to address the following question: is it possible to discover the number of hidden components and learn a statistical model over the entire collection of variables?

Note that we want to recover the networks of the observed variables but still we want to take into account the uncertainty about the hidden components.

Our contribution is to extented the model proposed by Chandrasekaran *et al.* (2010) for dynamic networks by considering different structures of the precision matrix.

Let $\mathbf{Z} = (\mathbf{Y}, \mathbf{X})$ be the vector of observed $\mathbf{Y}$ and unobserved $\mathbf{X}$ random variables. We assume that $\mathbf{Z} \sim N(\mathbf{0}, \boldsymbol{\Theta}^{-1})$, where $\boldsymbol{\Sigma} = \boldsymbol{\Theta}^{-1}$, and partition these two matrices as follows:

$$\boldsymbol{\Sigma} = \begin{pmatrix} \boldsymbol{\Sigma}_{yy} & \boldsymbol{\Sigma}_{yx} \\ \boldsymbol{\Sigma}_{xy} & \boldsymbol{\Sigma}_{xx} \end{pmatrix}, \boldsymbol{\Theta} = \begin{pmatrix} \boldsymbol{\Theta}_{yy} & \boldsymbol{\Theta}_{yx} \\ \boldsymbol{\Theta}_{xy} & \boldsymbol{\Theta}_{xx} \end{pmatrix}.$$

The marginal concentration matrix $\boldsymbol{\Theta}_{yy}$ corresponding to the observed variables $\mathbf{Y}$ is given by the Schur complement with respect to the block $\boldsymbol{\Sigma}_{xx}$, i.e.:

$$\boldsymbol{\Theta}_{yy} = \boldsymbol{\Sigma}_{yy}^{-1} - \boldsymbol{\Sigma}_{yx}^{-1}\boldsymbol{\Sigma}_{xx}\boldsymbol{\Sigma}_{xy}^{-1}. \tag{5.25}$$

In what follows we denote by $\mathbf{S} = \mathbf{\Sigma}_{yy}^{-1}$ and with $\mathbf{L} = \mathbf{\Sigma}_{yx}^{-1}\mathbf{\Sigma}_{xx}\mathbf{\Sigma}_{xy}^{-1}$ the matrices that compose $\mathbf{\Theta}_{yy}$. It turns out that when we observe only $\mathbf{Y}$ we can find only $\mathbf{\Sigma}_{yy}$ as shown in Chapter 3. However, we have given some biological motivation to consider unobserved random variables when we are estimating the structure of the network. If we analyse more closely the two terms involved in equation (5.25), we can give a precise meaning at these two components. In fact, the first term $\mathbf{S}$ represents the concentration matrix of the conditional statistics of the observed variables given the latent variables. A necessary assumption is that $\mathbf{S}$ is a sparse matrix. On the other end, the second term $\mathbf{L}$ represents a summary of the effect of marginalization over the hidden variables $\mathbf{X}$. It is necessary to assume that this matrix has low rank which means that the information of the hidden state is spread out over $\mathbf{Y}$. One can think about this solution as a connection between principal component analysis and sparse graphical models. In standard graphical models one would approximate a concentration matrix by a sparse matrix in order to learn a sparse graphical model, while in principal component analysis the goal is to explain the statistical structure underlying a set of observations using a small number of latent variables. However, in this framework the latent variables are not principle components and are called hidden components.

The fundamental issue of identifiability is now ready to be discussed. We have just seen that we can interpret the terms on the right hand of equation (5.25) but the question is whether or not we are able to separate these two effects once that we have observed $\mathbf{Y}$ and assumed that the complete vector $\mathbf{Z}$ is normally distributed. It turns out that $\mathbf{\Sigma}_{yy}^{-1}$ and $\mathbf{\Sigma}_{yx}^{-1}\mathbf{\Sigma}_{xx}\mathbf{\Sigma}_{xy}^{-1}$ can be identified. See Chandrasekaran *et al.* (2010) for a more detailed discussion.

Likelihood for Gaussian graphical models with latent variables

We consider the following minimization problem:

$$\begin{aligned}(\hat{\mathbf{S}},\hat{\mathbf{L}}) \quad &:= \quad \mathrm{argmin}_{\mathbf{S},\mathbf{L}}\{-l(\mathbf{S}-\mathbf{L};\mathbf{\Theta})+\lambda_1||\mathbf{S}||_1+\lambda_2\mathrm{tr}(\mathbf{L})\} \qquad (5.26)\\ \text{s.t.} \quad &\mathbf{S}-\mathbf{L}\succ \mathbf{0}\\ &\mathbf{L}\succeq \mathbf{0},\end{aligned}$$

where $l(\mathbf{S}-\mathbf{L};\mathbf{\Theta}) = \log|\mathbf{\Theta}| - \mathrm{tr}(\mathbf{W}\mathbf{\Theta})$, $\mathbf{W} = \sum_{i=1}^{n}(\mathbf{y}_i - \bar{\mathbf{y}})(\mathbf{y}_i - \bar{\mathbf{y}})'$ is the sum of squares of the observed variables, $\hat{\mathbf{S}}$ provides an estimate of $\mathbf{\Theta}_{yy}$ while the estimator for $\mathbf{\Sigma}_{yx}^{-1}\mathbf{\Sigma}_{xx}\mathbf{\Sigma}_{xy}^{-1}$ is given by $\hat{\mathbf{L}}$. A smoothing parameter $\lambda = (\lambda_1, \lambda_2)$ need to be estimated. Here λ_1 regulates the sparsity and λ_2 regulates the low rank matrix. The model is more flexible and there is no need to use neither EM-algorithm nor Kalman-Filter which suffer some problems in real applications.

Example. Consider 5 observed variables and 1 hidden state per 2 time points, then: Figure 5.5 shows the true network, the estimated one, and the estimated network with latent variable or recovered network. If we apply a Gaussian graphical model with latent variables than we can recover a network which is closer to the true one than the one estimated with the graphical lasso. The rank is $rank(\hat{\mathbf{L}}) = 2$ which corresponds to the number of hidden states.

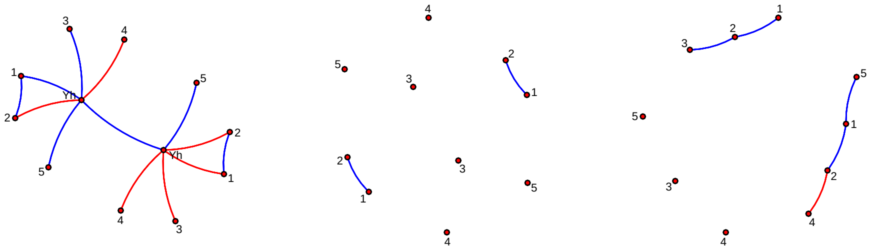


Figure 5.5. True network (left), estimated network (centre), and estimated network with latent structure (right)

5.4 Application

5.4.1 Real data application

In Chapter 3 and 4, we applied a structured Gaussian graphical models for the data set T-cell which is a time-course dataset where expression levels of 58 genes were collected across 10 time points. Here we apply Graphical graphical model for latent variable described in Section 5.3.2. We consider the same model as in Section 3.5 but in this case we take into account for latent structure. The model is

$$[S_0 \sim F_1, N_0 \sim F_\Gamma, S_1 \sim F_T, N_1 \sim F_\Gamma, S_2 \sim F_T],$$

that implies that the networks at temporal lag 0 are constrained to be equal across the five observed time points. Moreover, the networks at temporal lag 1 are constrained to be equal across time, no links are presents between time t and time

$t+2$ except for the self-self interactions, i.e. interactions between the same couple of genes. The recovered network structure is completely different from the

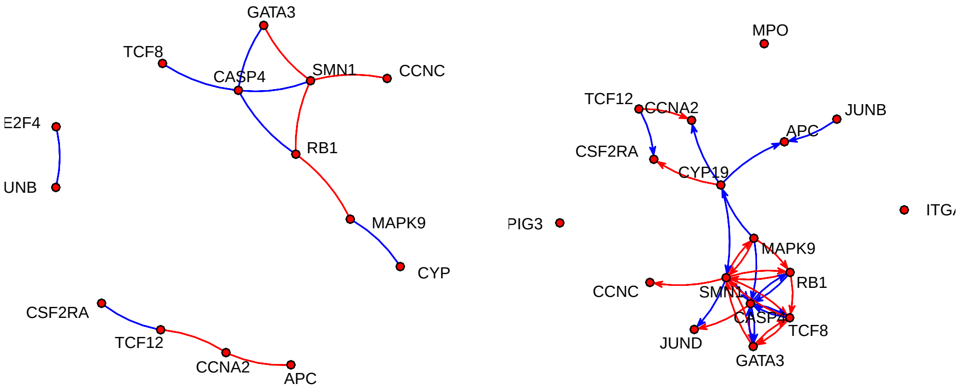


Figure 5.6. Recovered network structures for latent variables. Representation of interactions between genes at temporal lag 0. Note that networks at lag 0 at time $1,2,3,\ldots,5$ are equal since we impose $N_0 \sim F_\Gamma$ (left). Representation of interaction between genes at temporal lag 1. Note that networks at lag 1 between time $(1,2),(2,3),(3,4),(4,5)$ are equal since we impose $N_1 \sim F_\Gamma$ (right).

recovered network structure from copula and structured Gaussian graphical models. This suggest that some latent structure is present and it should be taken under consideration.

5.5 Summary

In this chapter, we have proposed methods to deal with different structured graphs. In particular we have considered a model to estimate slow change in dynamic graphs (GL_Δ). This model is very useful when small changes in the networks can be assumed. In fact, we have shown that common networks, appearance and disappearance of links, from a network at a point t to another one at time point $t+1$, can be estimated with GL_Δ model. In the second part of this chapter, we have discussed models to deal with partially unobserved graphs, i.e. graphs with latent structures. We have seen that dynamic graphical models need the Kalman-Filter algorithm to estimate the unknown quantities. This procedure makes the estimations quite unstable. A solution is to use the idea of Chandrasekaran *et al.* (2010) which we have extended for dynamic graphical lasso.

Summary

Estimating dynamic networks from data is an active research area and it is one important direction in system biology. Estimating the structure of a network is about deciding the presence or absence of relationships between random variables. Graphical models describe conditional independence relationships.

Gaussian graphical models are graphical models where it is assumed that the random variables follow a multivariate normal distribution. In this thesis, we have seen that, when Gaussian graphical models are applied in order to study large networks, they typically fail because the number of variables is much greater than the number of observations. Alternatively, penalized Gaussian graphical models have been proposed to estimate static networks in high-dimensional studies because of their statistical properties and computational tractability. However they do not consider the dynamic of the network. In order to incorporate the time dependence strucuters we have proposed several models.

In particular, we have proposed to use penalized Gaussian graphical models to estimate structured dynamic networks, for modeling slow change in dynamic networks, and to estimate particular structures such as scale-free dynamic networks in a small world setting. These models can be applied when estimating dynamic networks in high-dimensional environments.

When multivariate dynamic data are binary or ordinal random variables, transformations based on probability distribution with fixed marginals can be used to do inference. We propose the Gaussian copula for non-Gaussian graphical models to overcome the assumption of Gaussianity.

Finally, we have seen that the problem of estimating dynamic networks becomes even more challenging when latent or hidden variables are involved in larger systems. State-space models have been proposed in order to study dynamic networks with latent variables. We propose a penalized Gaussian graphical models to estimate dynamic networks with latent structures.

Samenvatting

Het bepalen van dynamische netwerken met behulp van data is een actief onderzoeksgebied, met name in de systeem biologie. Het schatten van de structuur van een netwerk heeft te maken met het bepalen van de aan- of afwezigheid van een relatie tussen de hoekpunten in de graaf. Grafische modellen definiren deze relaties via conditionele afhankelijkheid.

In Gaussiaanse grafische modellen (GGM) wordt verondersteld dat de hoekpunten een normale verdeling volgen. Dit heeft grote voordelen vanwege de computationele handelbaarheid van GGM. Standaard GGM zijn echter niet bruikbaar om grote netwerken te bestuderen, i.e. als het aantal waarnemingen minder is dan het aantal hoekpunten van de graaf. Recentelijk zijn bestrafde meest waarschijnlijke schatters voorgesteld om toch met hoog-dimensionale situaties om te kunnen gaan.

We stellen voor om bestrafde GGM te gebruiken in een aantal verschillende contexten: voor gestruktureerde dynamische modellen, voor langzaam veranderende dynamische modellen en voor modellen met een bepaalde structuur, zoals bijvoorbeeld met een kleine wereld architectuur. Elk van deze modellen kan worden toegepast in echte, hoog-dimensionale situaties waar de ontwikkeling van het netwerk een belangrijke rol speelt.

Zodra het onderliggend process op de hoekpunten binaire variabelen, ordinale variabelen, tellingen of op andere wijze niet-normale data zijn, stellen we in dit proefschrift voor om via een Gaussiaanse copula een algemeen niet-Gaussiaanse grafisch model te definiren. Deze copula transformeert de data of direct via de marginale verdelingsfunctie van de variabelen, of indirect via een latente normale variabelen. Deze aanpak is zeer successful, met name omdat het op eenvoudige wijze variabelen van verschillende typen samen kan modelleren in een grafisch model.

Het probleem van het schatten van een dynamisch network wordt nog moeilijker als een bepaald deel van de hoekpunten niet waargenomen zijn. In zulke gevallen worden typisch state-space modellen gebruikt, maar hier stellen we voor om een uitbreiding van onze bestrafde grafische model te gebruiken om het latente deel van het netwerk te schatten.

Appendix A

Structured graphical lasso

A.1 Wrapper function

```
function [X,KABI] = msglassoprecision1(S, p, t1, N, model, diag1, rho, S0, Niter)

n = p*t1;
[m] = nconstr(p, t1, diag1);

[blk, At, b, mu] = sglassoprecision(p, t1, model, n, m, diag1, S0);
[C] = funcSglasso(S, rho, m);

         OPTIONS.smoothing  = 1;
         OPTIONS.scale_data = 0;
         OPTIONS.plotyes    = 0;
         OPTIONS.tol        = 1e-6;

[~,X,~,~,~] = logdetPPA(blk,At,C,b,mu,OPTIONS);

X1 = X{1};
[KABI] = gof1(S, X1, N, p, t1, Niter, rho);

end
```

A.2 Bolck function sglassoprecision

```
function [blk, At, b, mu, h, h1, h2, h3, h4, h5, h6] =
sglassoprecision(p, t, model, n, m, diag1, S0)
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Block structure
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

blk{1,1} = ’s’; blk{1,2} = n;
blk{2,1} = ’l’; blk{2,2} = 2*m;


AA = cell(1,1);

if diag1 == 1,
    k = 1;
    for i = 1:n,
        for j = 1:n,
            if i <= j, AA{k} = sparse(i,j,1/S0(i,j),n,n);
                k = k+1;
            end

        end
    end
else

    k = 1;
    for i = 1:n,
        for j = 1:n,
            if i < j, AA{k} = sparse(i,j,1/S0(i,j),n,n);
                k = k+1;
            end
        end
    end


end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% End Block Structure
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%% Precision
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    if model(1) == 1,                   % all elements on the precision are equals

        for i = 1:n-1
            AA{k} = sparse(i,i,1,n,n);
            AA{k}(i+1,i+1) = -1;
            k = k+1;
        end

    elseif model(1) == 2,                                                   % time

        k1 = 0;

        for t1 =1:t
            for i = 1:p-1
                AA{k} = sparse(i+k1,i+k1,1,n,n);
                AA{k}(i+k1+1,i+k1+1) = -1;
                k = k + 1;
            end
            k1 = k1+p;
        end

    elseif model(1) == 3;                                                   % gene

        for j = 2:t
            for i = 1:p
                AA{k} = sparse(i+p*(j-2),i+p*(j-2),1,n,n);
                AA{k}(i+p*(j-1),i+p*(j-1)) = -1;
                k = k + 1;
            end
        end

    elseif model(1) == 4,    % all elements on theta are different, no constraints

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%% Constraint the diagonal elements to be 1 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
  elseif model(1) == 5,
        for i = 1:n,
            AA{k} = sparse(i,i, 1,n,n);
            k = k+1;
        end

    end
    h = k - m -1;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% End Precision %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Network
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    if model(2) == 0,
           u = 0;
    for t1 = 1:t,
        for i = 1:p-1,
            for j = 2:p,
                if i < j,        AA{k} = sparse(i+u,j+u,1/S0(i+u,j+u),n,n);
                    k = k +1;
                end
            end
        end
        u = u + p;
    end

    elseif model(2) == 1,
        u = 0;

        for t1 = 1:t

            for i = 1:p-1
                for j = 2:p-1
                    if i < j, AA{k}= sparse(i+u,j+u,1/S0(i+u,j+u),n,n);
                        AA{k}(i+u,j+u+1) = -1/S0(i+u,j+u+1);
                        k = k+1;
                    end

                end

            end
            for e = 1:p-2
                AA{k} = sparse(1+u,p+u, 1/S0(1+u,p+u), n,n);
                AA{k}(1+e+u,p+u) = -1/S0(1+e+u,p+u);
                k = k+1;
            end
            u = u+p;
        end

        o = 0;
        for t1 = 1:t-1,
            AA{k} = sparse(1+o,2+o,1/S0(1+o,2+o),n,n);
            AA{k}(1+p+o,2+p+o) = -1/S0(1+p+o,2+p+o);
            k = k+1;
            o = o+p;
        end

    elseif model(2) == 2,
        u = 0;
        for t1 = 1:t

            for i = 1:p-1
                for j = 2:p-1
                    if i < j, AA{k}= sparse(i+u,j+u,1/S0(i+u,j+u),n,n);
                        AA{k}(i+u,j+u+1) = -1/S0(i+u,j+u+1);
                        k = k+1;
                    end

                end

            end


            for e = 1:p-2
                AA{k} = sparse(1+u,p+u, 1/S0(1+u,p+u), n,n);
                AA{k}(1+e+u,p+u) = -1/S0(1+e+u,p+u);
                k = k+1;
            end
            u = u+p;
        end

    elseif model(2) == 3,
        o = 0;
        for t1 = 1:t-1,
            for i = 1:p
                for j = 1:p
                    if i<j,  AA{k} = sparse(i+o,j+o,1/S0(i+o,j+o),n,n);
                        AA{k}(i+p+o,j+p+o) = -1/S0(i+p+o,j+p+o);
                        k = k+1;
                    end
                end
            end
            o = o+p;
        end

    elseif model(2) == 4,

    end
    h1 = k - m - h - 1;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% End Network
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   AR1diag
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

if model(3) == 0,

    u = 0;
    u1 = p;

    for t1 = 1:t-1,

    for i = 1:p,
        for j = 1:p,
           if i == j, AA{k} = sparse(i+ u, j+u1, 1/S0(i+ u, j+u1), n, n);
               k = k +1;
           end
        end
    end
        u = u + p;
        u1 = u1 + p;
    end

elseif model(3) == 1,

    for i = 1:n-p-1,
        AA{k} = sparse(i,i+p,1/S0(i,i+p),n,n);
        AA{k}(i+1,i+p+1) = -1/S0(i+1,i+p+1);
        k = k+1;
    end

elseif model(3) == 2,                                                 % time

    k1 = 0;

    for t1 =1:t-1
        for i = 1:p-1
            AA{k} = sparse(i+k1,i+k1+p,1/S0(i+k1,i+k1+p),n,n);
            AA{k}(i+k1+1,i+k1+1+p) = -1/S0(i+k1+1,i+k1+1+p);
            k = k + 1;
        end
        k1 = k1+p;
    end

elseif model(3) == 3;                                                 % gene

    for j = 2:t-1
        for i = 1:p
            AA{k} = sparse(i+p*(j-2),i+p*(j-2)+p,1/S0(i+p*(j-2),i+p*(j-2)+p),n,n);
            AA{k}(i+p*(j-1),i+p*(j-1)+p) = -1/S0(i+p*(j-1),i+p*(j-1)+p);
            k = k + 1;
        end
    end

elseif model(3) == 4, % all elements on theta are different, no constraints
```

```
end
h2 = k - m - h - h1 - 1;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% N1
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if model(4) == 0,

    u = p;
    u1 = 0;

    for t1 = 1:t-1,
        for i = 1:p
            for j=1:p
                if i ~= j,  AA{k} = sparse(i+u1,j+u,1/S0(i+u1,j+u),n,n);
                k = k+1;
                end
            end
        end
        u = u + p;
        u1 = u1 + p;
    end


elseif model(4) == 1,

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Upper %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        u = 0;
        u1 = p;

        for t1 = 1:t-1

            for i = 1:p-1
                for j = 1:p-1
                    if i < j, AA{k}= sparse(i+u,j+u1,1/S0(i+u,j+u1),n,n);
                              AA{k}(i+u,j+u1+1) = -1/S0(i+u,j+u1+1);
                        k = k+1;
                    end
                end

            end


            for e = 1:p-2
                AA{k} = sparse(1+u,p+u1, 1/S0(1+u,p+u1), n,n);
                AA{k}(1+e+u,p+u1) = -1/S0(1+e+u,p+u1);
                k = k+1;
            end
            u = u+p;
            u1 = u1 + p;
        end

        o = 0;
        o1 = p;
        for t1 = 1:t-2,
            AA{k} = sparse(1+o,2+o1,1/S0(1+o,2+o1),n,n);
            AA{k}(1+p+o,2+p+o1) = -1/S0(1+p+o,2+p+o1);
            k = k+1;
            o = o+p;
            o1 = o1 + p;
        end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% LOWER
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

u11 = 0;
for t1 = 1:t-1,
    for i = 2:p-1,
        for j = 1:p-1,
            if i > j, AA{k} = sparse(i+u11, j+p +u11, 1/S0(i+u11, j+p +u11), n, n);
                      AA{k}(i+u11+1, j+p+1 + u11) = -1/S0(i+u11+1, j+p+1 + u11);
                      k = k+1;
            end
        end
    end
    u11 = u11 + p;
end
```

```
u22 = 0;
for t1 = 1:t-1,
    for i = 2:p-1,
        AA{k} = sparse(i+u22, p+1+u22, 1/S0(i+u22, p+1+u22),n,n);
        AA{k}(i+u22+1, p+1+u22)= -1/S0(i+u22+1, p+1+u22);
        k = k+1;
    end
u22 = u22 + p;
end

u33 = 0;
for t1 = 1:t-1,
    AA{k} = sparse(1+u33, p + u33 + 2, 1/S0(1+u33, p + u33 + 2), n, n);
    AA{k}(2+u33, 1+p+u33) = -1/S0(2+u33, 1+p+u33);
    k = k+1;
    u33 = u33 + p;
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
elseif model(4) == 2,

        u = 0;
        u1 = p;

        for t1 = 1:t-1

            for i = 1:p-1
                for j = 1:p-1
                    if i < j, AA{k}= sparse(i+u,j+u1,1/S0(i+u,j+u1),n,n);
                              AA{k}(i+u,j+u1+1) = -1/S0(i+u,j+u1+1);
                        k = k+1;
                    end

                end

            end
            for e = 1:p-2
                AA{k} = sparse(1+u,p+u1, 1/S0(1+u,p+u1), n,n);
                AA{k}(1+e+u,p+u1) = -1/S0(1+e+u,p+u1);
                k = k+1;
            end
            u = u+p;
            u1 = u1 + p;
        end


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
u11 = 0;
for t1 = 1:t-1,
    for i = 2:p-1,
        for j = 1:p-1,
            if i > j, AA{k} = sparse(i+u11, j+p +u11, 1/S0(i+u11, j+p +u11), n, n);
                      AA{k}(i+u11+1, j+p+1 + u11) = -1/S0(i+u11+1, j+p+1 + u11);
                      k = k+1;
            end
        end
    end
    u11 = u11 + p;
end

u22 = 0;
for t1 = 1:t-1,
    for i = 2:p-1,
        AA{k} = sparse(i+u22, p+1+u22, 1/S0(i+u22, p+1+u22),n,n);
        AA{k}(i+u22+1, p+1+u22)= -1/S0(i+u22+1, p+1+u22);
        k = k+1;
    end
u22 = u22 + p;
end

u33 = 0;
for t1 = 1:t-1,
    AA{k} = sparse(1+u33, p + u33 + 2, 1/S0(1+u33, p + u33 + 2), n, n);
    AA{k}(2+u33, 1+p+u33) = -1/S0(2+u33, 1+p+u33);
    k = k+1;
    u33 = u33 + p;
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
 elseif model(4) == 3,


%%%%%%%%% UPPER %%%%%%%%%%%%%%%%%%%%
          o = 0;
          o1 = p;
        for t1 = 1:t-2,
            for i = 1:p
                for j = 1:p
                    if i<j,  AA{k} = sparse(i+o,j+o1,1/S0(i+o,j+o1),n,n);
                        AA{k}(i+p+o,j+p+o1) = -1/S0(i+p+o,j+p+o1);
                        k = k+1;
                    end
                end
            end
            o = o+p;
            o1 = o1 + p;
        end

%%%%%%%%% LOWER %%%%%%%%%%%%%%%%%%%
         o2 = 0;
         o3 = p;

        for t1 = 1:t-2,
            for i = 1:p
                for j = 1:p
                    if i > j,  AA{k} = sparse(i+o2,j+o3,1/S0(i+o2,j+o3),n,n);
                               AA{k}(i+p+o2,j+p+o3) = -1/S0(i+p+o2,j+p+o3);
                        k = k+1;
                    end
                end
            end
            o2 = o2+p;
            o3 = o3 + p;
        end

elseif mode(4) == 4,

end
h3 = k - m -h - h1 - h2 - 1;
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% S2
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

if model(5) ==0,
    u = 0;
    u1 = 2*p;

    for t1 = 1:t-2,

    for i = 1:p,
        for j = 1:p,
           if i == j, AA{k} = sparse(i+ u, j+u1, 1/S0(i+ u, j+u1), n, n);
               k = k +1;
           end
        end
    end
        u = u + p;
        u1 = u1 + p;
    end

elseif  model(5) == 1,                   % all elements on the precision are equals

    for i = 1:n-2*p-1,
        AA{k} = sparse(i,i+2*p,1/S0(i,i+2*p),n,n);
        AA{k}(i+1,i+2*p+1) = -1/S0(i+1,i+2*p+1);
        k = k+1;
    end


elseif model(5) == 2,                                                   % time

    k1 = 0;

    for t1 =1:t-2
        for i = 1:p-1
            AA{k} = sparse(i+k1,i+k1+2*p,1/S0(i+k1,i+k1+2*p),n,n);
            AA{k}(i+k1+1,i+k1+1+2*p) = -1/S0(i+k1+1,i+k1+1+2*p);
            k = k + 1;
        end
        k1 = k1+p;
    end

elseif model(5) == 3;                                                    % gene

    for j = 2:t-2
        for i = 1:p
            AA{k} =
            sparse(i+p*(j-2),i+p*(j-2)+2*p,1/S0(i+p*(j-2),i+p*(j-2)+2*p),n,n);
            AA{k}(i+p*(j-1),i+p*(j-1)+2*p) = -1/S0(i+p*(j-1),i+p*(j-1)+2*p);
            k = k + 1;
        end
    end
elseif model(5) == 4;
end

h4 = k - m - h- h1 - h2 - h3 - 1;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% N2
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if model(6) == 0,

    u = 2*p;
    u1 = 0;

    for t1 = 1:t-2,
        for i = 1:p
            for j=1:p
                if i ~= j,  AA{k} = sparse(i+u1,j+u,1/S0(i+u1,j+u),n,n);
                k = k+1;
                end
            end
        end
        u = u + p;
        u1 = u1 + p;
    end


elseif model(6) == 1,
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Upper %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        u = 0;
        u1 = 2*p;

        for t1 = 1:t-2

            for i = 1:p-1
                for j = 1:p-1
                    if i < j, AA{k}= sparse(i+u,j+u1,1/S0(i+u,j+u1),n,n);
                              AA{k}(i+u,j+u1+1) = -1/S0(i+u,j+u1+1);
                        k = k+1;
                    end

                end

            end


            for e = 1:p-2
                AA{k} = sparse(1+u,p+u1, 1/S0(1+u,p+u1), n,n);
                AA{k}(1+e+u,p+u1) = -1/S0(1+e+u,p+u1);
                k = k+1;
            end
            u = u+p;
            u1 = u1 + p;
        end

        o = 0;
        o1 = 2*p;
        for t1 = 1:t-3,
            AA{k} = sparse(1+o,2+o1,1/S0(1+o,2+o1),n,n);
            AA{k}(1+p+o,2+p+o1) = -1/S0(1+p+o,2+p+o1);
            k = k+1;
            o = o+p;
            o1 = o1 + p;
        end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% LOWER
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

u11 = 0;
for t1 = 1:t-2,
    for i = 2:p-1,
        for j = 1:p-1,
            if i > j, AA{k} = sparse(i+u11, j+2*p +u11, 1/S0(i+u11, j+2*p +u11),
                       n, n);
                      AA{k}(i+u11+1, j+2*p+1 + u11) = -1/S0(i+u11+1, j+2*p+1 +
                      u11);
                      k = k+1;
            end
        end
    end
    u11 = u11 + p;
end

u22 = 0;
for t1 = 1:t-2,
    for i = 2:p-1,
        AA{k} = sparse(i+u22, 2*p+1+u22, 1/S0(i+u22, 2*p+1+u22),n,n);
        AA{k}(i+u22+1, 2*p+1+u22)= -1/S0(i+u22+1, 2*p+1+u22);
        k = k+1;
    end
u22 = u22 + p;
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
u33 = 0;
for t1 = 1:t-2,
    AA{k} = sparse(1+u33, 2*p + u33 + 2, 1/S0(1+u33, 2*p + u33 + 2), n, n);
    AA{k}(2+u33, 1+2*p+u33) = -1/S0(2+u33, 1+2*p+u33);
    k = k+1;
    u33 = u33 + p;
end


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
elseif model(6) == 2,

        u = 0;
        u1 = 2*p;

        for t1 = 1:t-2,

            for i = 1:p-1,
                for j = 1:p-1,
                    if i < j, AA{k}= sparse(i+u,j+u1,1/S0(i+u,j+u1),n,n);
                              AA{k}(i+u,j+u1+1) = -1/S0(i+u,j+u1+1);
                        k = k+1;
                    end

                end

            end
            for e = 1:p-2
                AA{k} = sparse(1+u,p+u1, 1/S0(1+u,p+u1), n,n);
                AA{k}(1+e+u,p+u1) = -1/S0(1+e+u,p+u1);
                k = k+1;
            end
            u = u+p;
            u1 = u1 + p;
        end


        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
u11 = 0;
for t1 = 1:t-2,
    for i = 2:p-1,
        for j = 1:p-1,
            if i > j, AA{k} = sparse(i+u11, j+2*p +u11, 1/S0(i+u11, j+2*p +u11),
            n, n);
                      AA{k}(i+u11+1, j+2*p+1 + u11) = -1/S0(i+u11+1, j+2*p+1 + u11);
                      k = k+1;
            end
        end
    end
    u11 = u11 + p;
end

u22 = 0;
```

```
for t1 = 1:t-2,
    for i = 2:p-1,
        AA{k} = sparse(i+u22, 2*p+1+u22, 1/S0(i+u22, 2*p+1+u22),n,n);
        AA{k}(i+u22+1, 2*p+1+u22)= -1/S0(i+u22+1, 2*p+1+u22);
        k = k+1;
    end
u22 = u22 + p;
end

u33 = 0;
for t1 = 1:t-2,
    AA{k} = sparse(1+u33, 2*p + u33 + 2, 1/S0(1+u33, 2*p + u33 + 2), n, n);
    AA{k}(2+u33, 1+2*p+u33) = -1/S0(2+u33, 1+2*p+u33);
    k = k+1;
    u33 = u33 + p;
end
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

 elseif model(6) == 3,

     %%%%%%%%% UPPER %%%%%%%%%%%%%%%%%%%%%
          o = 0;
          o1 = 2*p;
        for t1 = 1:t-3,
            for i = 1:p
                for j = 1:p
                    if i<j,  AA{k} = sparse(i+o,j+o1,1/S0(i+o,j+o1),n,n);
                        AA{k}(i+p+o,j+p+o1) = -1/S0(i+p+o,j+p+o1);
                        k = k+1;
                    end
                end
            end
            o = o+p;
            o1 = o1 + p;
        end

       %%%%%%%%% LOWER %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
         o2 = 0;
         o3 = 2*p;
        for t1 = 1:t-3,
            for i = 1:p
                for j = 1:p
                    if i > j,  AA{k} = sparse(i+o2,j+o3,1/S0(i+o2,j+o3),n,n);
                               AA{k}(i+p+o2,j+p+o3) = -1/S0(i+p+o2,j+p+o3);
                        k = k+1;
                    end
                end
            end
            o2 = o2+p;
            o3 = o3 + p;
        end

elseif mode(6) == 4,

end
h5 = k - m - h - h1 - h2 - h3 - h4 - 1;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Zero A priori constraints
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if model(7) == 0,

    u1 = p*(t-3);
    u2 = p*3;

    u = 0;

    for t1 = 1:(t-3),

        for i = 1:p,
           for j=1:u1,
                AA{k} = sparse(i+u, j+u2, 1,n,n);
                k = k+1;
            end
        end
        u1 = u1 - p;
        u = u + p;
        u2 = u2 +p;
    end
```

```
elseif model(7) == 1,

end
        h6 = k - m - h - h1 - h2 - h3 - h4 - h5 - 1;

if model(1) == 5,
b = [zeros(m,1); ones(h,1); zeros(h1,1); zeros(h2, 1); zeros(h3, 1); zeros(h4,1);
zeros(h5,1); zeros(h6, 1)];
else
b = [zeros(m,1); zeros(h,1); zeros(h1,1); zeros(h2, 1); zeros(h3, 1); zeros(h4,1);
zeros(h5,1); zeros(h6, 1)];
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

mu = [1; 10^-16];

At(1,:) = svec(blk(1,:), AA);

At{2,:} = [-speye(m, m + h + h1 + h2+ h3+ h4 + h5 + h6);
speye(m, m + h + h1 + h2+ h3 + h4 + h5 + h6)];


end
```

A.3 Function gof1

```
function [KABI] = gof1(S, X1, N, p, t1, Niter, rho)

KABI = zeros(1,6);
n = p*t1;
adj = zeros(n,n);
adj1 = zeros(n,n);

if Niter > 1,
    Rest = -X1./sqrt(diag(X1)*diag(X1)’);
    for i = 1:n
        for j = 1:n
            if i == j, Rest(i,j) = Rest(i,j)*-1;

            end
        end
    end


    for i = 1:n
        for j = 1:n

            if abs(Rest(i,j)) <= 0.000001, adj(i,j) = 0;
            else adj(i,j) = round(Rest(i,j)*10000)/100000;
            end

            if abs(X1(i,j)) <= 0.000001, adj1(i,j) = 0;
            else adj(i,j) = round(X1(i,j)*10000)/100000;
            end

        end
    end

    e1 = length(unique(diag(adj1))) + length(unique(adj)) - 1;

else

    for i = 1:n
        for j = 1:n
            if abs(X1(i,j)) <= 0.000001, adj(i,j) = 0;
            else adj(i,j) = round(X1(i,j)*100000)/100000;
            end
        end
    end

    e1 = length(unique(adj)) - 1;

end

base = (N/2)*(log(det(X1)) - (trace(S*X1)));
```

```
base1 =  (N/2)*(log(det(X1)) - (trace(S*X1))) -
  rho*sum(sum(abs(X1)));

AIC = -2*base + 2*e1;
BIC = -2*base + e1*log(N*n);

if (e1 > N || e1 == N), AICc = Inf;
else AICc = -2*base  + 2*N*n*(e1+1)/(N*n-e1-1);
end


KABI(1,1) = base;
KABI(1,2) = base1;

KABI(1,3) = BIC;
KABI(1,4) = AICc;
KABI(1,5) = e1;
KABI(1,6) = AIC;


end
```

References

Aas, K., Czado, C., Frigessi, A., and Bakken, H. (2009). Pair-copula constructions of multiple dependence. *Insurance: Mathematics and Economics*, **44**(2), 182–198.

Agarwal, S., Deane, C., Porter, M., and Jones, N. (2010). Revisiting date and party hubs: novel approaches to role assignment in protein interaction networks. *PLoS computational biology*, **6**(6), e1000817.

Ambroise, C., Chiquet, J., and Matias, C. (2009). Inferring sparse gaussian graphical models with latent structure. *Electronic Journal of Statistics*, **3**, 205–238.

Banerjee, O., El Ghaoui, L., and d'Aspremont, A. (2008). Model selection through sparse maximum likelihood estimation for multivariate gaussian or binary data. *The Journal of Machine Learning Research*, **9**, 485–516.

Bar-Joseph, Z. (2004). Analyzing time series gene expression data. *Bioinformatics*, **20**(16), 2493–2503.

Barabási, A. and Oltvai, Z. (2004). Network biology: understanding the cell's functional organization. *Nature Reviews Genetics*, **5**(2), 101–113.

Beal, M., Falciani, F., Ghahramani, Z., Rangel, C., and Wild, D. (2005). A bayesian approach to reconstructing genetic regulatory networks with hidden factors. *Bioinformatics*, **21**(3), 349–356.

Bishop, C. (1995). Neural networks for pattern recognition.

Boyd, S. and Vandenberghe, L. (2004). *Convex optimization*. Cambridge Univ Pr.

Breiman, L. (1996). Heuristics of instability and stabilization in model selection. *The Annals of Statistics*, **24**(6), 2350–2383.

Breiman, L. (1999). Prediction games and arcing algorithms. *Neural computation*, **11**(7), 1493–1517.

Buhl, S. (1993). On the existence of maximum likelihood estimators for graphical gaussian models. *Scandinavian Journal of Statistics*, pages 263–270.

Chandrasekaran, V., Parrilo, P., and Willsky, A. (2010). Latent variable graphical model selection via convex optimization. In *Communication, Control, and Computing (Allerton), 2010 48th Annual Allerton Conference on*, pages 1610–1613. IEEE.

Claeskens, G. and Hjort, N. (2008). Model selection and model averaging. *Cambridge Books*.

d'Aspremont, A., Banerjee, O., and Ghaoui, L. (2006). First-order methods for sparse covariance selection. *Arxiv preprint math/0609812*.

Demetrius, L. and Manke, T. (2005). Robustness and network evolutionan entropic principle. *Physica A: Statistical Mechanics and its Applications*, **346**(3), 682–696.

Dempster, A. (1972). Covariance selection. *Biometrics*, pages 157–175.

Dempster, A., Laird, N., and Rubin, D. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 1–38.

Dobra, A. and Lenkoski, A. (2011). Copula gaussian graphical models and their application to modeling functional disability data. *The Annals of Applied Statistics*, **5**(2A), 969–993.

Drton, M. and Perlman, M. (2004). Model selection for gaussian concentration graphs. *Biometrika*, **91**(3), 591–602.

Fan, J. and Li, R. (2001). Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, **96**(456), 1348–1360.

Fischer, M., Köck, C., Schlüter, S., and Weigert, F. (2009). An empirical analysis of multivariate copula models. *Quantitative Finance*, **9**(7), 839–854.

Friedman, J., Hastie, T., and Tibshirani, R. (2008). Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, **9**(3), 432.

Friedman, N., Linial, M., Nachman, I., and Pe'er, D. (2000). Using bayesian networks to analyze expression data. *Journal of computational biology*, **7**(3-4), 601–620.

Gasch, A., Spellman, P., Kao, C., Carmel-Harel, O., Eisen, M., Storz, G., Botstein, D., and Brown, P. (2000). Genomic expression programs in the response of yeast cells to environmental changes. *Molecular biology of the cell*, **11**(12), 4241–4257.

Genest, C., Ghoudi, K., and Rivest, L. (1995). A semiparametric estimation procedure of dependence parameters in multivariate families of distributions. *Biometrika*, **82**(3), 543–552.

Ghahramani, Z. (1998). Learning dynamic bayesian networks. *Adaptive Processing of Sequences and Data Structures*, pages 168–197.

Ghahramani, Z. and Hinton, G. (1996). Parameter estimation for linear dynamical systems. *University of Toronto technical report CRG-TR-96-2*, **6**.

Green, P. (1990). On use of the em for penalized likelihood estimation. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 443–452.

Guo, J., Levina, E., Michailidis, G., and Zhu, J. (2011). Joint estimation of multiple graphical models. *Biometrika*, **98**(1), 1.

Han, J., Bertin, N., Hao, T., Goldberg, D., Berriz, G., Zhang, L., Dupuy, D., Walhout, A., Cusick, M., Roth, F., *et al.* (2004). Evidence for dynamically organized modularity in the yeast protein–protein interaction network. *Nature*, **430**(6995), 88–93.

Hoff, P. (2007). Extending the rank likelihood for semiparametric copula estimation. *The Annals of Applied Statistics*, pages 265–283.

Höfling, H. and Tibshirani, R. (2009). Estimation of sparse binary pairwise markov networks using pseudo-likelihoods. *The Journal of Machine Learning Research*, **10**, 883–906.

Højsgaard, S. and Lauritzen, S. (2008). Graphical gaussian models with edge and vertex symmetries. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, **70**(5), 1005–1027.

Holmes, E. (2010). Derivation of the em algorithm for constrained and unconstrained multivariate autoregressive state-space (marss) models. Technical report, Technical report, Northwest Fisheries Science Center, NOAA Fisheries 2725 Montlake Blvd E., Seattle, WA 98112.

Husmeier, D. (2003). Sensitivity and specificity of inferring genetic regulatory interactions from microarray experiments with dynamic bayesian networks. *Bioinformatics*, **19**(17), 2271–2282.

Jensen, F. (1996). *An introduction to Bayesian networks*, volume 74. UCL press London.

Joe, H. (1997). *Multivariate models and dependence concepts*, volume 73. Chapman & Hall/CRC.

Kasowski, M., Grubert, F., Heffelfinger, C., Hariharan, M., Asabere, A., Waszak, S., Habegger, L., Rozowsky, J., Shi, M., Urban, A., *et al.* (2010). Variation in transcription factor binding among humans. *Science*, **328**(5975), 232.

Lam, C. and Fan, J. (2009). Sparsistency and rates of convergence in large covariance matrix estimation. *Annals of statistics*, **37**(6B), 4254.

Lauritzen, S. (1996). *Graphical models*, volume 17. Oxford University Press, USA.

Leng, C., Lin, Y., and Wahba, G. (2006). A note on the lasso and related procedures in model selection. *Statistica Sinica*, **16**(4), 1273.

Liebscher, E. (2008). Construction of asymmetric multivariate copulas. *Journal of Multivariate analysis*, **99**(10), 2234–2250.

Mazumder, R. and Hastie, T. (2011). Exact covariance thresholding into connected components for large-scale graphical lasso. *Arxiv preprint arXiv:1108.3829*.

McGlohon, M. and Faloutsos, C. (2007). *ADAGE: A software package for analyzing graph evolution*. Carnegie Mellon University, School of Computer Science, Machine Learning Dept.

Meinshausen, N. (2008). A note on the lasso for gaussian graphical model selection. *Statistics & Probability Letters*, **78**(7), 880–884.

Meinshausen, N. and Bühlmann, P. (2006). High-dimensional graphs and variable selection with the lasso. *The Annals of Statistics*, **34**(3), 1436–1462.

Meinshausen, N. and Bühlmann, P. (2010). Stability selection. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, **72**(4), 417–473.

Murphy, K. (2002). *Dynamic bayesian networks: representation, inference and learning*. Ph.D. thesis, University of California.

Natarajan, K., Meyer, M., Jackson, B., Slade, D., Roberts, C., Hinnebusch, A., and Marton, M. (2001). Transcriptional profiling shows that gcn4p is a master regulator of gene expression during amino acid starvation in yeast. *Molecular and Cellular Biology*, **21**(13), 4347.

Nau, G., Richmond, J., Schlesinger, A., Jennings, E., Lander, E., and Young, R. (2002). Human macrophage activation programs induced by bacterial pathogens. *Proceedings of the National Academy of Sciences*, **99**(3), 1503.

Neapolitan, R. (2004). *Learning bayesian networks*. Pearson Prentice Hall Upper Saddle River, NJ.

Nelsen, R. (2006). *An introduction to copulas*. Springer Verlag.

Newman, M. (2010). *Networks: an introduction*. Oxford Univ Pr.

Opdenakker, M. and Maulana, R. (2012). Changes in teachers instructional behaviour and students motivation during the first grade of secondary education: An exploration by means of multilevel growth curve modelling.

Opdenakker, M., Maulana, R., and den Brok, P. (2011). Teacher–student interpersonal relationships and academic motivation within one school year: developmental changes and linkage.

Opsahl, T., Agneessens, F., and Skvoretz, J. (2010). Node centrality in weighted networks: Generalizing degree and shortest paths. *Social Networks*, **32**(3), 245–251.

Palmitesta, P. and Provasi, C. (2005). Aggregation of dependent risks using the koehler–symanowski copula function. *Computational Economics*, **25**(1), 189–205.

Pearl, J. (2000). *Causality: models, reasoning and inference*. Cambridge Univ Press.

Peng, J., Wang, P., Zhou, N., and Zhu, J. (2009). Partial correlation estimation by joint sparse regression models. *Journal of the American Statistical Association*, **104**(486), 735–746.

Perrin, B., Ralaivola, L., Mazurie, A., Bottani, S., Mallet, J., and dAlche Buc, F. (2003). Gene networks inference using dynamic bayesian networks. *Bioinformatics*, **19**(suppl 2), ii138.

Preiss, B. (2008). *Data structures and algorithms with object-oriented design patterns in C++*. A1bazaar.

Rangel, C., Angus, J., Ghahramani, Z., Lioumi, M., Sotheran, E., Gaiba, A., Wild, D., and Falciani, F. (2004). Modeling t-cell activation using gene expression profiling and state-space models. *Bioinformatics*, **20**(9), 1361–1372.

Ravikumar, P., Wainwright, M., and Lafferty, J. (2010). High-dimensional ising model selection using 1-regularized logistic regression. *The Annals of Statistics*, **38**(3), 1287–1319.

Rothman, A., Bickel, P., Levina, E., and Zhu, J. (2008). Sparse permutation invariant covariance estimation. *Electronic Journal of Statistics*, **2**, 494–515.

Shumway, R. and Stoffer, D. (1982). An approach to time series smoothing and forecasting using the em algorithm. *Journal of time series analysis*, **3**(4), 253–264.

Sklar, A. (1959). Fonctions de répartition à n dimensions et leurs marges. *Publ. Inst. Statist. Univ. Paris*, **8**(1), 11.

Tallis, G. (1961). The moment generating function of the truncated multi-normal distribution. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 223–229.

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

Toh, K., Todd, M., and Tütüncü, R. (1999). Sdpt3a matlab software package for semidefinite programming, version 1.3. *Optimization Methods and Software*, **11**(1-4), 545–581.

Tong, Y. (1990). Multivariate normal distribution.

Tütüncü, R., Toh, K., and Todd, M. (2003). Solving semidefinite-quadratic-linear programs using sdpt3. *Mathematical programming*, **95**(2), 189–217.

Van Putten, C. and Van Schuppen, J. (1985). Invariance properties of the conditional independence relation. *The Annals of Probability*, **13**(3), 934–945.

Wang, C., Sun, D., and Toh, K. (2009). Solving log-determinant optimization problems by a newton-cg primal proximal point algorithm. *preprint*.

Whittaker, J. (1990). *Graphical models in applied multivariate statistics*, volume 16. Wiley New York.

Wilhelm, S. and Manjunath, B. (2010). tmvtnorm: A package for the truncated multivariate normal distribution. *sigma*, **2**, 2.

Wit, E. and McClure, J. (2004). *Statistics for microarrays*. Wiley Online Library.

Yuan, M. and Lin, Y. (2007). Model selection and estimation in the gaussian graphical model. *Biometrika*, **94**(1), 19–35.

Acknowledgments

I am indebted to Prof. dr. Ernst Wit which introduce me to the subject of this thesis and gave me the great opportunity to work in his group. He guided my first steps into graphical models and in particular into penalized graphical models for genetic datasets. Most of the results presented in this thesis would not have been possible without his support.

I wish to acknowledge the member of the reading committee Prof. dr. A.C.D. van Enter, Prof. dr. E.R. van den Heuvel and Prof. dr. G. Kauermann for accepting to be on my committee and reading the manuscript. Their comments were very useful to improve this thesis. I would like to thank dr. Veronica Vinciotti whom gave me many useful suggestions to improve the readability of this thesis.

I wish to express my warm and sincere thanks to my officemates and friends, whom I was extremely lucky to meet. Ivan and Nynke whom supported me in several number of ways, Lotsi whom supported me at the beginning of my visit in Groningen, Reza and Parya whom made me think in a positive way. I indebted to Fentaw whom did most of the job on static copula in Chapter 4. I am glad I met people like Javier, Peter, Hande and Younas. In particular, I am glad I met Henri whom has been a great friend.

I am obliged to the personal of the department in Groningen. They took care of all the administrative nuisances and their support was commendable. In particular, Ineke Schelhaas and Esmee Elshof helped me in many situations.

Many thanks to my advisor Prof. dr. Angelo M. Mineo, Prof dr. Marcello Chiodi and the department of statistics in Palermo to give me the great opportunity to spend long part of my PhD research in Groningen. In particular, I owe my most sincere gratitude to Prof. dr. Gianfranco Lovison which made me understand that statistics is more than formulae. I would like to thank Renato Cataldo whom I shared the passion for football. I would like to thank Rosangela Anzalone whom helped to find any article or book I need for my research.

I owe my deepest gratitude to my family whom unconditional supported me … grazie mille. I owe my loving thanks to my Maria to support me during my whole PhD study. Without her encouragement and understanding it would have been impossible for me to finish this work … Grazie Amore.

Antonino Abbruzzo
April, 2012
Gronigen.