technische universität
dortmund

Statistical modeling of protein-protein interaction networks

Dissertation by
Yessica Yulieth Fermin Ruiz

submitted to the Faculty of Statistics,
TU Dortmund University, Germany

in fulfillment of the requirements for the degree of
Doktor der Naturwissenschaft

Submitted on 9th November 2018
Oral examination held on 17th December 2018

Supervisor 1: Prof. Dr. Katja Ickstadt

Supervisor 2: Prof. Dr. Jörg Rahnenführer

Abstract

Statistical modeling of protein-protein interaction networks

Understanding how proteins bind to each other in a cell is the key in molecular biology to determine how experts can repair anomalies in cells. The major challenge in the prediction of protein-protein interactions is the cell-to-cell heterogeneity within a sample, due to genetic and epigenetic variabilities. Most studies about protein-protein interaction carry out their analysis without awareness of the underlying heterogeneity. This situation can lead to the identification of invalid interactions. As part of the solution to this problem, we proposed in this thesis two aspects of analysis, one for snapshot data, where different samples of ten proteins were taken by toponome imaging and another for the analysis of time correlated data that guarantees a better approximation to the prediction of protein-protein interactions. The latter represents an advance in the analysis of data with high temporal resolution, such as that obtained through the quantification technique known as multicolor live cell imaging. The thesis here presented is divided into two parts: The first part called "**Revealing relationships among proteins involved in assembling focal adhesions**" consists of the development of a methodology based on frequentist methods, such as machine learning and meta-analysis, for the prediction of protein-protein interaction on six different toponome imaging datasets. This methodology presents an advance in the analysis of highly heterogeneous snapshot data. Our aim here focused on the formulation of a single model capable of identifying the relationship among different samples by summing is common results over them concerning their random variation. This methodology leads to a set of common models over the six datasets hierarchized by their predictive power, where the researcher can choose the model according to its accuracy in the prediction or according to its parsimony. The developing of this part is in Chapters 1-7 — this part published in Harizanova et al. (2016).

The second part is called "**Modelling of temporal networks with a nonparametric**

mixture of dynamic Bayesian networks". The content of this part contemplates the advance of a Bayesian methodology regarding temporal networks that successfully enables to identify subpopulations in heterogeneous cell populations as well as at the same time reconstructing the protein interaction network associated with each subpopulation. This method extends the nonparametric Bayesian networks (NPBNs) (Ickstadt et al., 2011) for the analysis of time-correlated data by using Gaussian dynamic Bayesian Networks (GDBNs). We evaluate our model based on the variation of specific parameters such as the underlying number of subpopulations, network density, intra-subpopulation variability among others. On the other hand, a comparative analysis with existing clustering methods such as NPBNs and hierarchical agglomerative clustering (Hclust), shows that the inclusion of temporal correlations in the classification of multivariate time series is relevant for an improvement in the classification. The classic Hclust method using the dynamic time warping distances (T-Hclust) was found to be similar in precision to our Bayesian method here proposed. On the other hand, a comparative analysis with the GDBNs shows the lack of adjustment of the GDBNs to reconstruct temporal networks in heterogeneous cell populations through a single model, while our method, as well as the joint use of the T-Hclust classifications with the GDBNs (T-Hclust+), show a high adequacy in the prediction of temporal networks in a mixture. The developing of this part is in Chapters 8-16.

This dissertation is dedicated to my love,

Jesus Christ

Acknowledgment

A very special thank to my advisor Prof. Dr. Katja Ickstadt for her confidence on my capabilities, for providing me the opportunity to work on this project, and her guidance and support along the performance of this research.

Many thanks to Dr. Eli Zamir from Max Planck Institute of Molecular Physiology, Department of Systemic Cell Biology, and at present from Max Planck Institute for Medical Research. His collaboration in this project and his support was fundamental for the culmination of this dissertation.

Thank you to Prof. Dr. Jörg Rahnenführer, for his collaboration in the evaluation of this dissertation and his support all the time.

Thank you to my colleagues Dr. Claudia Köllmann, Dr. Leo Geepert, Dr. Sabrina Siebert, Dr. Jacob Wieczorek, Dr. Anita Thieler, M.Sc. Jonathan Rathjens. Their support went beyond the exchange of academic insight, by easing my adaptation process to the University of Dortmund and making the work environment-friendly. They additional supported me to overcome the linguistic barriers related to German being my second language, which I highly appreciate.

Many thanks to the DAAD for giving me the opportunity of coursing my doctoral studies throughout the scholarship given during the first three years of my studies.

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Dissertation selbständig verfasst und dabei keine anderen als die angegebenen Hilfsmittel benutzt habe. Sämtliche Stellen der Dissertation, die im Wortlaut oder dem Sinn nach Publikationen oder Vorträgen anderer Autoren entnommen sind, habe ich als solche kenntlich gemacht. Die Dissertation wurde bisher weder gesamt noch in Teilen einer anderen Prüfungsbehörde vorgelegt und auch noch nicht veröffentlicht. Ich habe bisher kein Promotionsverfahren erfolglos beendet und keine Aberkennung eines bereits erworbenen Doktorgrades vorliegen.

Yessica Yulieth Fermin Ruiz

Ort, Datum

Contents

List of Tables

List of Figures

Introduction

The proteins are formed by units called amino acids, these are still unknown. Many claims that the number of essential amino acids is 22, but the verification of such quantity is unknown. The lack of knowledge about amino acids is an example of how protein synthesis could be misunderstood, which affects the environment as well as the human race since with this lack of knowledge many of us have acquired food habits that are not suitable for our molecular structure and are not apt to be degraded by nature. Many chemicals or drugs do not come from a real molecular structure, which when discarded by the body goes to the environment, which can be degraded if and only if the chemical compound that represents the drug is compatible with those of the environment. Otherwise, the new molecules become part of the environment thus contaminating nature and forming new additives not suitable for humans. In this work, we will undertake a search to analyze data related to protein synthesis.

Part I

Revealing relationships among proteins involved in assembling focal adhesions

1. Introduction

The space between the cells is filled in a multi-cellular organism by a nonliving material called the extracellular matrix. The roles of this matrix vary from tissue to tissue and include providing structural, biochemical support to the cell, regulating thereby the cell's dynamic behavior, such as its morphology, migration, proliferation, survival, and differentiation (Gumbiner, 1996; Hynes and Lander, 1992). The binding process between a cell and the extracellular matrix is mediated by large multi-protein structures within the cell known as cell-matrix adhesion sites (Zamir and Geiger, 2001) or in a short form as focal adhesions, a term employed in this thesis. Figure 1.1 contains a schematic representation of this process, where the focal adhesions consist of receptor proteins called integrins and a group of interacting proteins that link the receptor proteins to the actin filaments. This is a familiar picture of the process, but actually, the group of interacting proteins can contain over 100 different proteins (Hanein and Horwitz, 2012; Geiger et al., 2009).

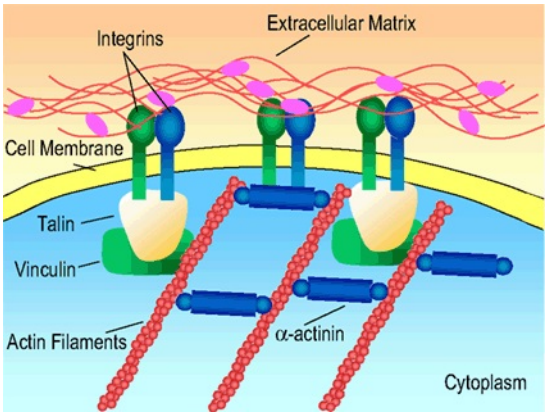


Figure 1.1: Exemplification of the cell-matrix adhesion process. Figure adapted from http://www.mun.ca/biology.

The understanding of how proteins interact with each other in a cell can be influenced to repair anomalies in them. The knowledge of the formation of multi-protein structures is guided by the quantitation of the states of the associated proteins to predict the relationship between them. Unfortunately, in spite of significant advances in proteomic and methods, a large number of components and the diversity among these components within a single cell pose a fundamental challenge for monitoring the full molecular content involved in a specific cellular process. Therefore, thus far there is no way to observe the whole system at once (i.e., in the same cell) but only a small part of it. This limitation, by itself, could have been overcome by looking at different parts of the system in different cells and building step by step a model of the whole system (Harizanova et al., 2016; Wieczorek et al., 2015). However, focal adhesions are quite complex structures, often varying in size, shape, distribution, dynamics, and molecular compositions which make the integration of information highly challenging.

In this work, we focused on the analysis of a part of these multi-protein structures formed by ten proteins known as α-actinin, F-actin, FAK, FAK-pY397, Hic-5, paxillin, paxillin-pY118, VASP, vinculin, and zyxin. We use a measurement technique known as high-throughput cyclic immunofluorescence to quantify levels of these proteins in thousands of individual focal adhesions by using a measurement technique known as high-throughput cyclic immunofluorescence. The measurement procedure was carried out by the research group of systems biology of cell-matrix adhesion at the Max Planck Institute of Molecular Physiology, and the results were six independent datasets. Our aims are:

1. To quantify, independently for each dataset, how much a combination of input proteins informs us about the level of another, target, protein across the focal adhesions.
2. To find common results among the different datasets.
3. To integrate the common findings over the different datasets for interpretation and practical use of the results.

For the first aim, we screened systematically for high-order correlations using artificial neural

networks (ANNs) and Random Forests (RFs). In particular, ANNs and RFs are two machine learning techniques which are currently valuable tools for variable interactions modeling and are especially useful in analyzing large datasets and identifying non-linear relationships (Zeng and Wang, 2010). ANNs are potent and practical tools to solve complex dependencies which are difficult with other traditional statistical methods. Additionally, ANNs have the advantages that a deep learning neural network takes into consideration the inner links among features links and by applying multiple hidden layers on the model, we would be able to capture the confounding relationships among the elements and combine the items into robust features each time we apply the additional layer. In studies comparing machine learning methods (Liu et al., 2013; Were et al., 2015; Rodriguez-Galiano et al., 2015) have been found that ANNs yield the highest accuracy rate in almost all cases, while RFs have shown to perform favorably in cases of heterogeneous data. It may be because RFs use subsets of training sets with bagging and subsets of features which can help reduce the effect of heterogeneity in the data. Therefore we consider convenient the application of both methods to find the most suitable results. Nevertheless, before the application of these two methods, to reduce the cell-to-cell variability, we propose a Box-Cox transformation (Box and Cox, 1964) of the measurements with a posterior standardization and removal of the outliers from the obtained standard scores by using Tukey's approach (Tukey, 1977). To address the second aim, we propose first to identify the suitable input protein combinations separately within each dataset by using determination coefficient and hypothesis tests about the linear relation between observed and predicted values from the best results between ANNs and RFs. Afterward, the common results over the six data subsets are identified, based on the suitable input proteins combinations in each dataset. Finally, for the third aim, we merge common findings using a model of random effects meta-analysis (Schulze et al., 2002).

Some of the results presented here are already published in Harizanova et al. (2016).

This part is divided into five chapters. Chapter 2 includes details about the six datasets. Chapter 3 the description of the methods employed. Chapter 4 the data analysis methodology. Chapter 5 the results and Chapter 6 the conclusions.

2. Description of datasets

In this chapter, we will present in detail the datasets used for the achievement of this proposal.

2.1 Description of datasets

The datasets, here employed, contain the intensity levels of ten proteins, α-actinin, F-actin, FAK, FAK-pY397, Hic-5, paxillin, paxillin-pY118, VASP, vinculin, and zyxin, whose levels were quantified in thousands of individual focal adhesions from 257 live cells using a method known as high-throughput cyclic immunofluorescence. This procedure is carried through successive cycles of immunolabeling, imaging, and bleaching, enabling the labeling of many components to use the same fluorophore (YFP-paxillin), which is a component of a molecule that makes it fluoresce. Figure 2.1 contains a description of this procedure, and we give more details in Harizanova et al. (2016). We computed the levels of a given component in a given focal adhesion as the sum of the intensity in its corresponding pixels in the corresponding image divided by the number of these pixels.

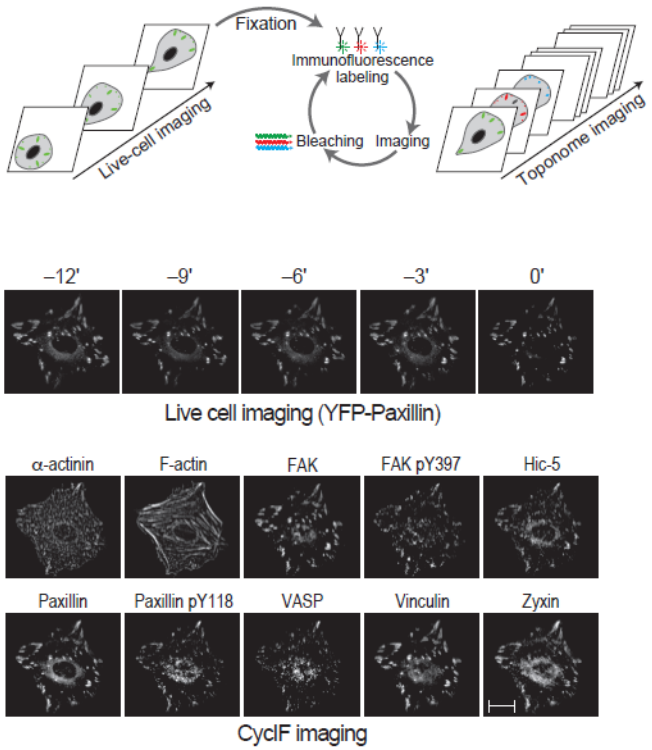


Figure 2.1: High-throughput cyclic immunofluorescence imaging of cell-matrix adhesion sites. Imaging procedure and an example of the images obtained for a cell. Scale bar, 10 μm. Figure adapted from Harizanova et al. (2016).

As a result of the employment of the procedure described above, we obtained six different datasets by using three repeats of four cycles with two permuted labeling (see Table 2.1). We denote the datasets as R1O1, R1O2, R2O1, R2O2, R3O1, R3O2. Table 2.1 contains the number of cells and focal adhesions per dataset.

Table 2.1: Labeling orders of the components in the CycIF imaging cycles. Table adapted from Harizanova et al. (2016).

Order	Cycle 1	Cycle 2	Cycle 3	Cycle 4
1	Vinculin	Zyxin	Paxillin	F-actin
	FAK	VASP	Paxillin-pY118	Hic-5
	FAK-pY397			α-Actinin
2	F-actin	Paxillin	Zyxin	Vinculin
	Hic-5	Paxillin-pY118	VASP	FAK
	α-Actinin			FAK-pY397

Table 2.2: Number of cells and focal adhesions per dataset.

	R1O1	R1O2	R2O1	R2O2	R3O1	R3O2
No. of cells	38	36	49	51	46	37
No. of FAs	353074	352226	497612	483875	435290	299534

2.2 Exploratory data analysis

To explore the behavior of the levels of the ten proteins and to compare these among the datasets, we employed frequency histograms where the different datasets were overlapped in the same graph to facilitate comparison among them (see Figure 2.2). In these histograms, we can not only see that the distributions of the levels of the proteins are not approximately symmetrical but also it is evident that the lack of asymmetry can be due to outliers, in this comparison we observed that lack of asymmetry could be causing discrepancies between samples.

Additionally, we support the above conclusions with Bar plots depicted in Figure 2.3. Here it can be appreciated that effectively the levels of proteins are asymmetrically distributed and directly affected by a large number of outliers. Besides, we can see that there are discrepancies among the central values on the six subsets. Nonetheless, there is no reason to assume that this is due to a systematic variation. Note that there is not a particular pattern indicating that the different cycles and labelings of the measurement technique are causing the discrepancies among the different datasets. On the contrary, these discrepancies seem to be due to random variations,

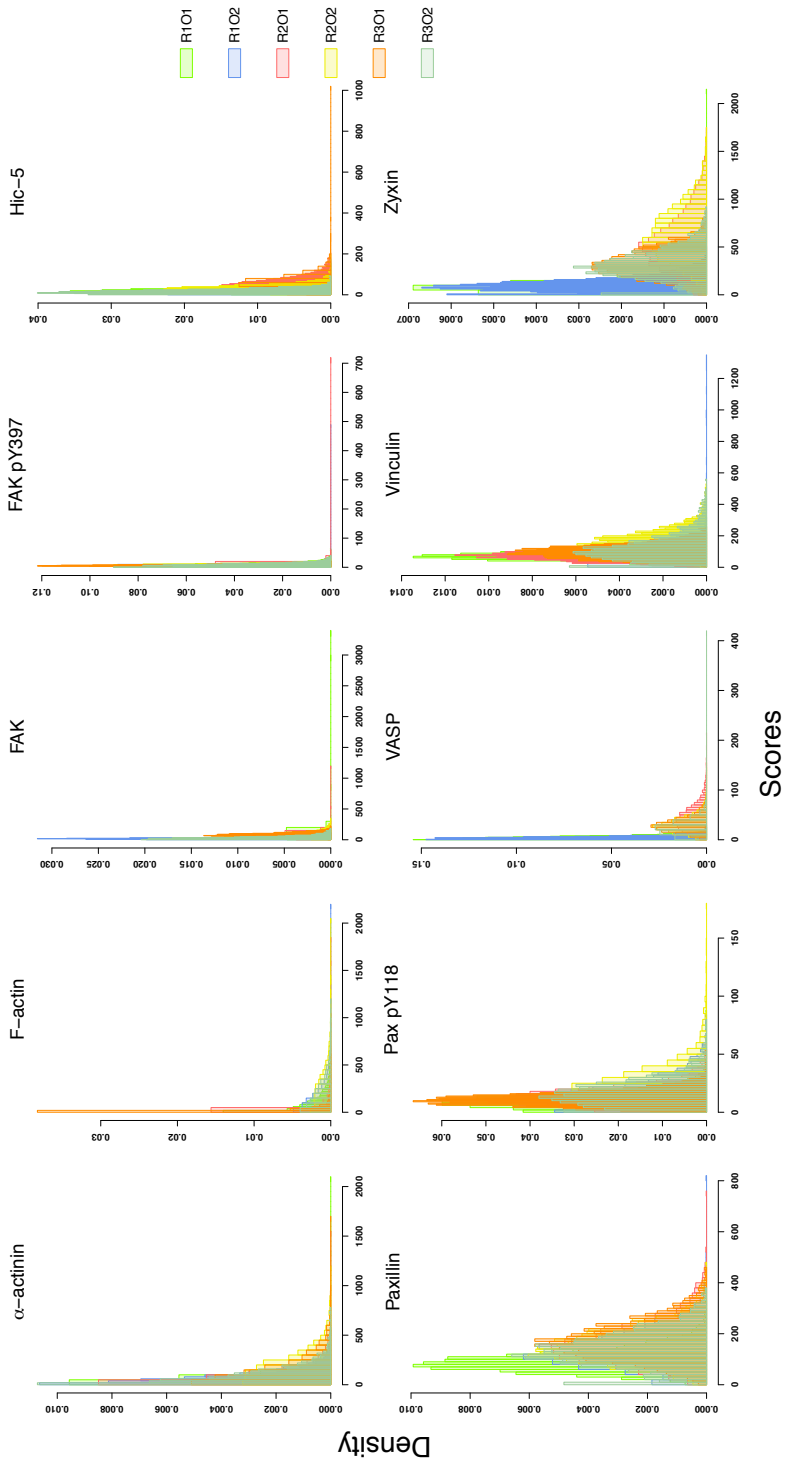


Figure 2.2: Histogram of frequencies for the levels of the ten proteins over the six datasets.

a product of factors not considered. To reduce the number of extreme values and to make the distributions of protein levels more symmetric, we propose to preprocess the data subsets by employing a Box-Cox transformation (Box and Cox, 1964) of the measurements, followed by a normalization of the transformed scores and finally reducing the number of extreme values by using Tukey's approach (Tukey, 1977). We give more details about this procedure in Section 4.1.

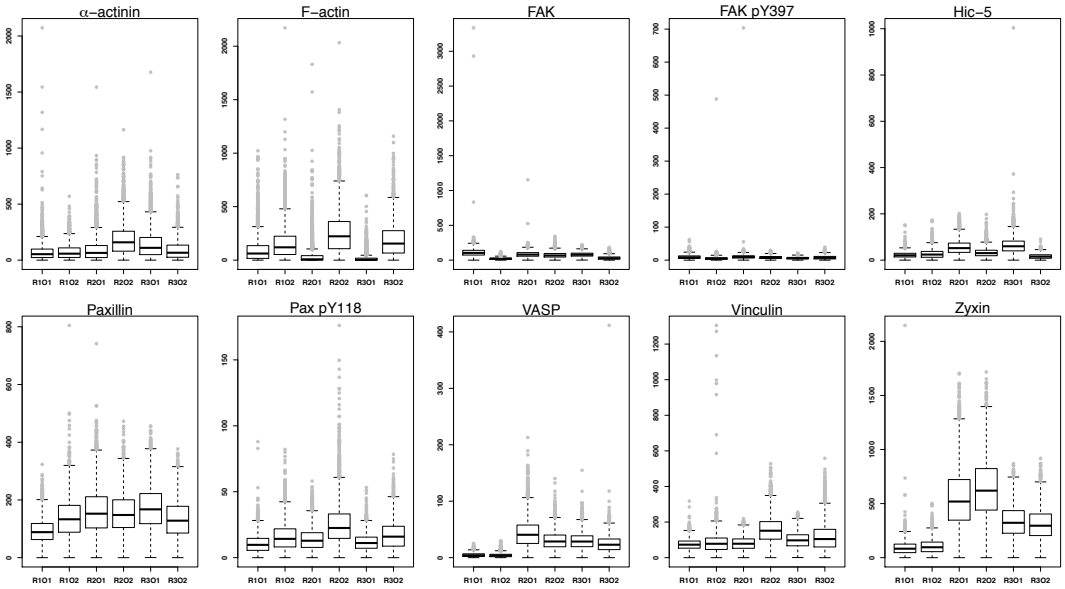


Figure 2.3: Bar plots for the levels of the ten proteins over the six datasets.

3. Methods

3.1 Box-Cox transformation

The Box-Cox transformation (Box and Cox, 1964) attempts to transform continuous measurements into values approximately normally distributed by scaling the values using a parametric family of transformations indexed by an unknown parameter λ which defines a particular transformation. When the random variable X to be transformed is garantizada to be posittive and diffrente from zero, the Box-Cox transformation is given by:

$$X^{(\lambda)} = \begin{cases} (X^{\lambda} - 1)/\lambda & \lambda \neq 0 \\ \log(X) & \lambda = 0 \end{cases} \tag{3.1}$$

In case that X does not have positiveness, this must be achieve by using a constant term, a, such that $X + a > 0$ and the corresponding transformation will lead to:

$$X^{(\lambda)} = \begin{cases} [(X + a)^{\lambda} - 1]/\lambda & \lambda \neq 0 \\ \log(X + a) & \lambda = 0 \end{cases} \tag{3.2}$$

The implementation of the Box-Cox transformation attempts for finding the λ value that

maximizes the log-likelihood function:

$$\mathrm{L}_{max}(\lambda) = -\frac{1}{2} N \log \widehat{\sigma}^2_{(X^\lambda)} + (\lambda - 1) \sum_{i=1}^{N} \log x_i \tag{3.3}$$

for the transformations 3.1 and

$$\mathrm{L}_{max}(\lambda) = -\frac{1}{2} N \log \widehat{\sigma}^2_{[(X+a)^\lambda]} + (\lambda - 1) \sum_{i=1}^{N} \log(x_i + a) \tag{3.4}$$

for the transformations 3.2. Here N denotes the number of realization of the variable X and $\widehat{\sigma}^2$ is the sample variance of the transformed values.

3.2 Artificial neural networks

Artificial neural networks (ANNs) can be seen as two-step regression or classification model. In general they are represented by network diagrams as the one in Figure 3.1, where each unit represents a neuron, and the connections (edges) represent synapses. This network applies only to regression; the respective network for classification has more than one output. Our main aim in this work is to regress variables instead of classifying them. Therefore, we restrict our attention to the definition of ANNs for regression. For details and description of ANNs for classification see e.g. Hastie et al. (2009).

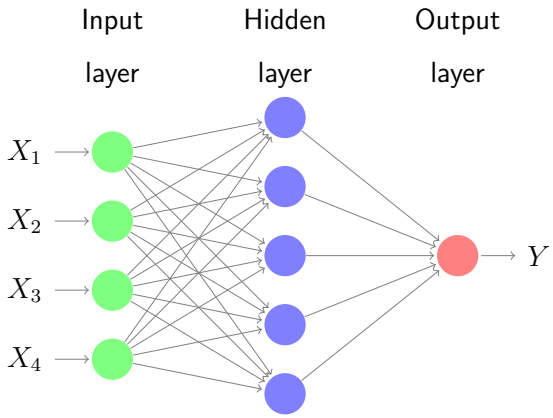


Figure 3.1: Structure of an feedforward ANN with four input variables and one hidden layer.

Let $\mathbf{X} = (X_1, \ldots, X_n)^\top$ be a vector of n random variables and Y a target variable to be explained (or predicted) by $\mathbf{X}$. In ANNs, the input variables are combined to create a new vector $\mathbf{Z} = (Z_1, \ldots, Z_M)^\top$, which can be used to explained Y in a linear functional form as follows:

$$Y = \beta_0 + \boldsymbol{\beta}^T \mathbf{Z} + \varepsilon, \tag{3.5}$$

where the coefficients β_0, $\boldsymbol{\beta}$ and the error term ε have the same meaning as in linear regression models.

The vector $\mathbf{Z}$ is created by using a scalar-to-scalar function known as *activation function* on a linear combination of the input variables in $\mathbf{X}$, that is:

$$Z_m = \phi(\alpha_{0m} + \boldsymbol{\alpha}_m^\top \mathbf{X}), \;\; m = 1, \ldots, M. \tag{3.6}$$

The flexibility of ANNs to fit a wide variety of functions derives from the freedom to choose different values for the coefficients $\{\alpha_{0m}, \boldsymbol{\alpha}_m; m = 1, \ldots, M\}$, $\{\beta_{0k}, \boldsymbol{\beta}\}$, the activation function ϕ and any parameters which they might contain. Note that if ϕ is the identity function, then the complete Eq. 3.5 collapses to the conventional linear regression model on $\mathbf{X}$. The activation function in this work is the one known as sigmoid activation function, given by:

$$\phi(w) = \frac{1}{1 + exp\{-w\}}. \tag{3.7}$$

Other alternatives can be found for example in Hastie et al. (2009) and Banyasz (2014).

ANN structures like Figure 3.1 are sometimes depicted with an additional *bias* unit feeding into every unit in the output and hidden layers. Contemplating the constant "1" as an additional input component, this bias unit captures the intercepts α_{0m} and β_0 in model 3.5. The units in the middle of the network structure are called hidden units because they represent the derived features Z_m whose values are not directly observed. In general there may be more than one hidden layer. Hastie et al. (2009) mentions that it is better to have too many hidden units than too few. With too few hidden units, the model might not have enough flexibility to capture the nonlinearities in the data; with too many hidden units, the extra coefficients can be shrunk toward zero if appropriate regularization is used. Typically the number of hidden units is somewhere in the range of 5 to 100, with the number increasing with the number of inputs and number of training cases.

For the estimation of the coefficients involved in Eq. 3.5. ANNs use the least square method, which is aimed to minimize the sum-of-squared errors

$$R(\boldsymbol{\theta}) = \sum_{i=1}^{N} (y_i - \hat{y}_i)^2, \tag{3.8}$$

where $\boldsymbol{\theta}$ represents the set of parameters $\{\alpha_{0m}, \boldsymbol{\alpha}_m; m = 1, \ldots, M\}$ $(M * (n + 1)$ coefficients) and $\{\beta_{0k}, \boldsymbol{\beta}\}$ (M+1 coefficients); and $\hat{y}_i$ the predicted values of y_i obtained by using Eq. 3.5.

The most popular approaches to minimizing $R(\boldsymbol{\theta})$ are the algorithms known as gradient descent backpropagation, Gauss-Newton (GN) and Levenberg-Marquardt (LM). A gradient descent backpropagation algorithm is a supervised learning technique that attempts to minimize the error of the network by moving down the gradient of the error curve as stated. The huge advantage of backpropagation is that unlike other techniques, it works independently of

theoretical assumption (Tadeusiewicz et al., 2014). In the backpropagation algorithm, each hidden unit passes and receives information only to and from units that share a connection. Thus it can be implemented efficiently on a parallel platform. However, since this algorithm approaches the minimum in a linear (first order) manner its speed of convergence is normally low, thus does not always possess adequate convergence properties. On the other hand, the GN algorithm has quadratic convergence properties that make it very fast. Nonetheless, its convergence depends, highly, on the choice of the initial weight values. The prediction of an appropriate set of initial values is not always possible in real-world applications, which makes the GN method unworkable for many applications. A better alternative is given by the LM algorithm, which combines the positive characteristic of the GN and gradient descent algorithms. Thus, the LM algorithm overcomes the limitations of these approaches and is more appropriated in practice. More details of theses algorithms can be found, e.g. in Sun and Yuan (2006) and Press et al. (1992, Chapter 15.5).

3.3 Random forests for regression

Random forests (RFs) are an ensemble learning approach proposed by Breiman (2001) for building a predictor by using a set of regression trees.

RFs are the consequence of an extension of bagging with decision trees introduced by Breiman (1996) by adding the idea of using random selection of variables introduced independently by Ho (1998) and Amit and Y (1997) in order to enhance prediction accuracy of bagging. The essential idea of bagging is to average many noisy but approximately unbiased models, hence reducing the prediction variance without affecting the prediction bias. Trees are ideal candidates for bagging, since they can capture complex interaction structures in the data and have low relatively bias if grown sufficiently deep (Hastie et al., 2009). The trees generated in bagging are identically distributed (i.d.), hence the expectation of an average of B such trees is the same as the expectation of any one of them. This means the bias of bagged trees is the same as that of

the individual trees, and the only hope of improvement is through variance reduction. Since the trees are simply i.d. (identically distributed, but not necessarily independent) with positive pairwise correlation ρ, the variance of the average is given by:

$$\rho\sigma^2 + \frac{1-\rho}{B}\sigma^2, \tag{3.9}$$

where ρ is the sampling correlation between any pair of trees used in the averaging and σ^2 is the sampling variance of any single randomly drawn tree.

As B increases, the second term in Eq. 3.9 disappears, but the first remains, and hence the size of the correlation of pairs of bagged trees limits the benefits of averaging. The idea in random forests is to improve the variance reduction of bagging by reducing the correlation between the trees, without increasing the variance too much. This is achieved in the tree-growing process through random selection of the input variables.

In random forests, each tree is grown specifically as follows:

1. A bootstrap training data set, $\mathbf{D}_b$, is drawn, with replacement, from the original training data set.
2. Grow the tree on $\mathbf{D}_b$ selecting, at each node of the tree, $m \leq n$ inputs variables at random from the inputs subset $\mathbf{X} = (X_1, \ldots, X_n)^\top$ and using the best variable (split-point) from those m.
3. Each tree is grown to maximum length, with no pruning.

After B such trees, $\{T(\mathbf{x}; \boldsymbol{\theta}_b)\}_1^B$, are grown, the random forest (regression) predictor is computed by:

$$\hat{f}_{rf}^B(\mathbf{x}) = \frac{1}{B}\sum_{b=1}^{B} T(\mathbf{x}; \boldsymbol{\theta}_b), \tag{3.10}$$

where $\boldsymbol{\theta}_b$ is the set of parameters associated to the b-th random forest tree in terms of split variables, cutpoints at each node, and terminal-node values. Hastie et al. (2009) shows how the

correlation between pairs of trees decreases as m (number of input variables selected) decreases: pairs of tree predictions at $\mathbf{x}$ for different training sets are likely to be less similar if they do not use the same splitting variables. Hence, intuitively from Eq. 3.9, reducing m will reduce the variance of the average. In addition, Breiman (2001) recomends using as default value for m $n/3$ and as the minimum node size five.

An important feature of random forests is its use of out-of-bag (oob) samples, that is, for an specific bootstrap sample $\mathbf{D}$ with observations $d_i = (\mathbf{x}_i, y_i)$, construct its random forest predictor by averaging only those trees corresponding to bootstrap samples in which z_i did not appear. Call this the oob predictor. Then the oob error is the error of the oob predictor on $\mathbf{D}$, which is obtained as the mean squared error (MSE) in a linear model. Random forests also use the oob samples to construct a variable-importance scoring function to measure the prediction strength of each input variable. To measure the importance of the i-th input variable, the values of that variable are permuted in the oob samples and the oob error is again computed on this perturbed data set. The decrease in accuracy as a result of this permuting is averaged over all trees, and is used as an importance score for the i-th variable in the random forest. The randomization effectively voids the effect of a variable, much like setting a coefficient to zero in a linear model. This does not measure the effect on prediction were this variable not available, because if the model was refitted without the variable, other variables could be used as surrogates.

3.4 Random-effects model of meta-analysis

Meta-analysis is a widely used statistical procedure for merging and contrasting findings from independent but related studies in order to identify similarities or discrepancies among them, or other issues that may come to light in the context of multiple studies.

Meta-analysis basically is done by identifying a common statistical measure, called effect size, that is shared between studies and afterward computing a weighted average of this measure.

The associated weights are related to the study's precision: Studies with relatively a good precision are corresponding with high weights, while studies with relatively poor precision with low weights.

There are two models common used in a meta-analysis, one knows as fixed-effect model (FE model) and another as random-effects model (RE model). In the FE model, the inverse of the estimates' variance is commonly used as the weights. Thus, by using the inverse of the variance of the effect size as weights, more accurate estimates contribute more to the final average than less accurate estimates. The FE model assumes that all the studies in the analysis share the same true effect size. However, in many systematic reviews this assumption is mostly implausible.

Studies may differ for example, in terms of experimental unit characteristics or methods employed, among other reasons, which may lead to different values for the true effect size. We might not information about covariances actually related to the size of the effect. Nevertheless, logic dictates that such factors do exist and will lead to significance variations in the effect size estimation. A way to consider this variation in the analysis is by using a RE model. In contrast to the FE model, the RE model allows that the study-specific estimators of the effect size may possess different expected values. Let us consider k independent studies and let $\theta_1, \ldots, \theta_k$ be the (one dimensional) parameter of interest, where θ_k denotes the effect size in the k-th study. The parameter $\theta_i,\ i = 1, \ldots, k$ may represent, for example in analyzes with continuous variables, the true mean, the standardized difference of means or the normalized correlation coefficients. In each study an estimate of the effect size is available, and all study-specific estimators $\widehat{\theta}_i,\ i = 1, \ldots, k$, are stochastically independent. Thus, assuming that the specific estimator are at least approximately normally distributed and unbiased or at least consistent, it holds approximately:

$$\widehat{\theta}_i|\theta_i, \sigma^2(\widehat{\theta}_i) \sim \mathsf{N}(\theta_i, \sigma^2(\widehat{\theta}_i)),\ \ i = 1, \ldots, k, \tag{3.11}$$

where $\sigma^2(\widehat{\theta}_i)$ denotes the variance of the estimator in the i-th study. And for each study-specific mean θ_i we assume that it is drawn from some superpopulation of effects with mean θ and variance τ^2, that is,

$$\theta_i|\theta,\tau^2 \sim N(\theta,\tau^2). \tag{3.12}$$

The parameters θ and τ^2 are referred as hyperparameters, θ represents the average effect size and τ^2 the between-study variation. Given the hyperparameters, the marginal distribution of the estimators $\widehat{\theta}_i$ is given by (Schulze et al., 2002):

Definition 3.4.1 (Random-effects model)

$$\widehat{\theta}_i \sim N(\theta,\tau^2+\sigma^2(\widehat{\theta}_i)),\ \ i=1,\ldots,k, \tag{3.13}$$

If the between-study variance τ^2 is equal to zero then the RE model 3.13 reduces to the FE model.

In RE model 3.13, the best linear unbiased estimator of the average effect size θ is given by

$$\widetilde{\theta}_{RE} = \frac{\sum_{i=1}^{k} w_i\widehat{\theta}_i}{w},\ \ w=\sum_{i=1}^{k} w_i, \tag{3.14}$$

where $w_i = [\tau^2+\sigma^2(\widehat{\theta}_i)]^{-1}$ is the inverse of the variance of the estimator $\widehat{\theta}_i$. The estimator $\widetilde{\theta}_{RE}$ is also the maximum likelihood estimator (MLE) of θ if the variance components are known and the normal distribution in 3.13 exactly holds.

The estimator $\widetilde{\theta}_{RE}$ in 3.14 possesses furthermore the following distributional property:

$$\widetilde{\theta}_{RE} \sim N(\theta,\frac{1}{w}). \tag{3.15}$$

This property can be used for confidence interval construction for $\widetilde{\theta}_{RE}$, as well as for the performance of conventional significance tests associated to the parameter of interest.

Mostly in practice, the involved variances τ^2 and $\sigma^2(\widehat{\theta}_i),\ i = 1, \ldots, k$, are hardly ever known. So they have to be replaced by appropriate estimators. Normally, an estimator of $\sigma^2(\widehat{\theta}_i)$, say $\widehat{\sigma}^2(\widehat{\theta}_i)$, is available in each study. These estimators $\widehat{\sigma}^2(\widehat{\theta}_i),\ \ i = 1, \ldots, k$, are assumed to be stochastically independent and at least nearly unbiased for $\sigma^2(\widehat{\theta}_i)$. One method for estimating τ^2 is the method of moments (or the DerSimonian and Laird) method (Schulze et al., 2002). The method of moments estimator of τ^2 is defined as follows:

$$\widehat{\tau}^2 = \frac{Q - (k-1)}{\upsilon - \sum_{i=1}^{k} \upsilon_i^2/\upsilon},\ \ \upsilon = \sum_{i=1}^{k} \upsilon_i,\ \ \upsilon_i = [\sigma^2(\widehat{\theta}_i)]^{-1},\ \ i = 1, \ldots, k. \tag{3.16}$$

Since the estimator depends on the unknown variances $\sigma^2(\widehat{\theta}_i)$, for a practical application the estimator of τ^2 used is given by (Schulze et al., 2002):

$$\widehat{\tau}^2 = \frac{Q - (k-1)}{\widehat{\upsilon} - \sum_{i=1}^{k} \widehat{\upsilon}_i^2/\widehat{\upsilon}}, \tag{3.17}$$

where Q is computed by

$$Q = \sum_{i=1}^{k} \widehat{\upsilon}_i \widehat{\theta}_i^2 - \frac{(\sum_{i=1}^{k} \widehat{\upsilon}_i \widehat{\theta}_i)^2}{\widehat{\upsilon}}. \tag{3.18}$$

Thus, a feasible estimator of the average effect size θ in the RE model is given by:

$$\widetilde{\theta}_{RE} = \frac{\sum_{i=1}^{k} \widehat{w}_i \widehat{\theta}_i}{\widehat{w}},\ \ \widehat{w} = \sum_{i=1}^{k} \widehat{w}_i,\ \ \widehat{w}_i = [\widehat{\tau}^2 + \widehat{\sigma}^2(\widehat{\theta}_i)]^{-1},\ \ i = 1, \ldots, k. \tag{3.19}$$

4. Data analysis methodology

In this section, we will describe the methodological procedure used for the data analysis designed to achieve the aims described at the beginning of this study.

4.1 Data preprocessing

It is well known that both parametric and non-parametric methods benefit from distributions of approximately Gaussian. Thus, to obtain a more symmetric distribution of the data presented in Figure 2.2, we performed the Box-Cox transformation defined in Eq. 3.2. We computed the optimal transformation parameter λ as the mean of individually determined transformation parameters for each connected component within a single dataset using the maximum likelihood approach (Box and Cox, 1964). Once we transformed the data, we proceeded to normalize the data by subtracting the mean and dividing it by the standard deviation. Afterward, we removed the outliers on the obtained standard scores by using Tukey's approach (Tukey, 1977), which considers a standardized observation z an outlier if:

$$z \notin [\mathtt{Q}_1 - 1.5 \times \mathtt{IRQ}; \mathtt{Q}_3 + 1.5 \times \mathtt{IRQ}], \tag{4.1}$$

where $\mathtt{Q}_1$ is the lower quartile, $\mathtt{Q}_3$ the upper quartile, and $\mathtt{IRQ} = \mathtt{Q}_3 - \mathtt{Q}_1$ the interquartile range.

We performed this analysis in Matlab version 7.10 (R2010a)(MATLAB, 2010) using the function "boxcox".

4.2 Aim 1: Finding different combinations of input proteins suitable to predict another target protein

In order to test if the information about the levels of a given subset of proteins (input proteins) is sufficient for predicting the levels of another given protein (target protein), we proceeded to infer high-order statistical relations between the levels of proteins in focal adhesions by using artificial neural networks (ANNs) and Random Forests (RFs) defined in Chapter 3.

ANNs and RFs are currently powerful and useful tools to analyze large datasets and to identify high-order interactions (Zeng and Wang, 2010). They have the advantage that they do not make any assumptions for the functions we try to approximate, and they can approximate any linear and non-linear function although unlike ANNs, RFs method has the advantage that it is computationally efficient and thus can operate quickly over large datasets.

ANNs and RFs were performed on each of the six datasets independently, to allow the evaluation of reproducibility. This analysis consists of three steps. First, we organized the input and target data for training and evaluating the models. A given component was considered as a target protein, while all possible combinations of the remaining nine elements were used to generate 511 separate inputs. Second, for each input-target combination, independent sessions of training and testing were performed, each session using the components levels of randomly sampled 40% and 60% focal adhesions, respectively. In the third step, the performance was evaluated based on the root mean squared error (Eq. RMSE) between the target (y_i) and predicted ($\widehat{y}_i$) output after training using the testing data. The RMSE is given by

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(\widehat{y}_i - y_i)^2}{n}}. \tag{4.2}$$

The type of neural network used was a multilayer perceptron, consisting of an input layer, five hidden layers and an output layer with the number of neurons in each layer equaling the

number of input components. To train the network, we employed the Levenberg-Marquardt back-propagation algorithm with the sigmoid activation function available in Neural Networks Toolbox of Matlab version 7.10 (R2010a)(MATLAB, 2010). On the other hand, RFs analysis was performed using the randomForest package (Liaw and Wiener, 2002) in R version 3.0.1 (R Core Team, 2013). For the training, we selected 500 trees to grow and all components in the respective combination of input components as a candidate at each node.

Lastly, we compared the findings with ANNs and RFs through the RMSE values to continue the following analysis with the results of the method that shows greater accuracy.

4.3 Aim 2: Identifying common findings among different datasets

In cases of modeling biological systems, the idea of making an inference from a single (best) model cannot be reasonable, since it is well-known that most biological data are inherently noisy due to the underlying nature of the process. Therefore, analysis of different datasets obtained even under the same experimental conditions may lead us to mixed results.

In this study, for specific research purposes, it is expected that the six datasets will lead to similar results. Therefore we aim here to seek common results on the different datasets rather than highlighting the discrepancies among them. Nonetheless, the selection of a single model in each dataset would still be too strict here to assess the reproducibility of the results over the six datasets. If we rank the resulting models separately for each dataset using, e.g. the root mean squared error (RMSE) values, we may have that a model associated with a particular combination of input components for a given dataset has here the highest RMSE, while in another dataset it has the second highest value. However, both RMSEs may not be significantly different and the accuracy of the model with the second highest value may be in a range of precision that could still be considered appropriate for an explanation of the target component levels. Thus, instead of proceeding as commonly done, selecting the best model, we continued to identify a set of input protein combinations suitable to explain a target protein.

The variety of goodness-of-fit measures available for nonlinear models such as ANNs and RFs is quite broad, among them we could name the coefficient of determination between the target and predicted values, the RMSE, the mean absolute error (MAE), and the Akaike information criterion (AIC). However, the selection of appropriate models by comparing the precision between models with measures such as RMSE, MAE or AIC is not possible as the comparison between, e.g. two RMSE values is not intuitively interpretable. How do we know how higher or how to lower an RMSE is concerning the highest RMSE?. Unlike these measures, the comparison of models using the coefficient of determination is intuitively interpretable because this measure takes values that range from 0 to 1. However, this measure could lead to the selection of models with poor prognosis, since it is clear that with these measures, models approaching the target protein above its value or below this, even significant differences, could be selected. One way to avoid this is to apply a hypothesis test related to the linear relationship between the target (y) and predicted $(\widehat{y})$ values. Note that if we were in the case of a perfect or nearly perfect prediction, we expressed the degree of agreement between the target and predicted values in terms of a linear regression with intercept 0 and coefficient 1. In other words, we would expect $y \approx \widehat{y}$. Thus, by adding a hypothesis test, we overcome the weaknesses of the correlation measure. However, it is important to note that this technique is quite flexible, keep in mind that we are selecting models that fall in the region of non-rejection, which is quite broad, that can be seen as choosing a parameter value within a 95 % confidence interval, so that a large number of models are likely to be selected as suitable models. However, along with the use of the coefficient of determination as a measure of goodness of fit and considering the number of input proteins the user of the deriving results could have the possibility to choose between a group of input proteins that best explain the target protein or a parsimonious number of input proteins with similar accuracy results.

Once we have found in each dataset the respective set of input protein combination that explains suitably a target protein, we proceed to the selection of common input combinations among the six datasets.

4.4 Aim 3: Merging common findings

Because of variability within and among datasets, the common combinations of potential input proteins over the different datasets do not have the same hierarchy of precision in each of them. That is, a model with a particular combination of input proteins can have the highest determination coefficient in dataset R1O1 and dataset R1O2, the third highest value, which makes difficult the selection of one common result. A way to address this limitation if by proceeding a meta-analysis (see section 3.4) by using the random-effects model (RE model). This model allows differences among effect size estimates over different studies, and this can be interpreted as considering the variation among data sets. The definition of the effect size should be related to the accuracy of the prediction of the target protein. Thus, we employed as effect size the normalized correlation coefficient of the observed and predicted values of a target protein. The normalization of the correlation coefficient, also known as Fisher's z-transformation, is given by:

$$Z_r = 0.5 \times \ln\left(\frac{1+r}{1-r}\right). \tag{4.3}$$

This transformation is required for the RE model to fulfill the normal distribution assumption on the effect sizes estimator. The average final result of this analysis is afterward transformed in the original form of a correlation coefficient and used in terms of its squared form, a determination coefficient, for a reasonable comparison among the potential combinations of input proteins.

5. Results

This section contains the most relevant results of the analyses proposed above. We conducted these analysis by using Matlab version 7.10 (R2010a)(MATLAB, 2010) and R version 3.0.1 (R Core Team, 2013).

5.1 Frequency distribution of the transformed levels of proteins

Figure 5.1 shows the frequency distribution of the Box-Cox transformed and normalized protein levels over the six datasets following the methodology described in section 4.1.

In Figure 5.1, we can appreciate that normal distribution approximate the distribution of the observations. Thus suggesting that these are better in transformation than in raw.Only in a few exceptions such as for protein Vasp, where a small deviation of normality can be observed in datasets R1O1 and R2O2, and for Hic-5 in dataset R2O2. Comparing distributions we do not find errors in transformed measures.

Comparison of the six datasets using Figure 5.1 does not appear to be straightforward. We additionally employed Box-Whisker-plots (see Figure 5.2). On them, there does not also seem to be a significant difference among the distributions of the transformed levels of the proteins.

Figure 5.1: Histograms of Box-Cox transformed and normalized protein levels. Datasets are labeled by colors. Normal distribution is here found to be more approximated to the frequentist distribution of the data for each sample distribution labeled by colors describe on the left side of the graphic.

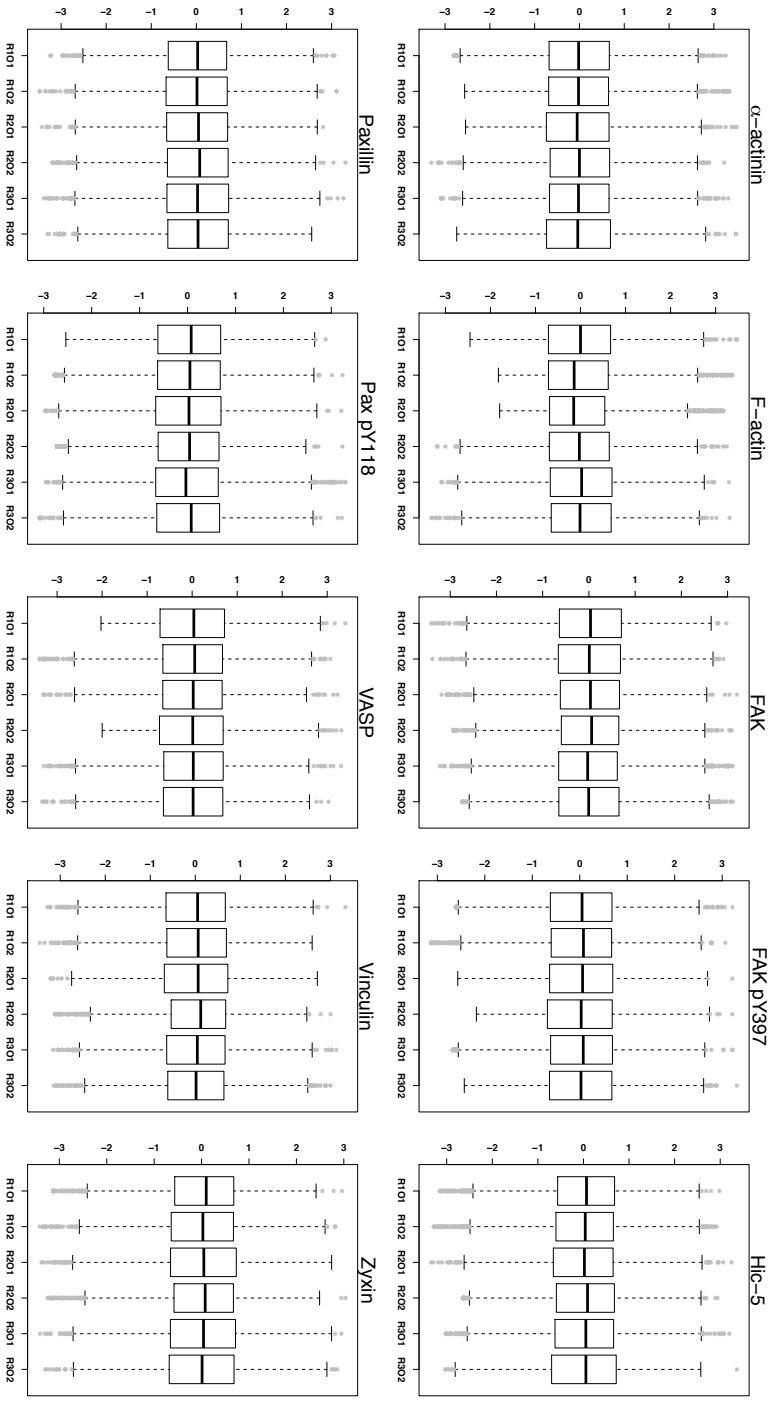


Figure 5.2: Box-Whisker-plots for protein levels over the six datasets.

5.2 Assessment of linearity in the correlation between proteins

In biological terms the evidence of linear relationship between the proteins on focal adhesions is directly related to a protein stoichiometry uniform among focal adhesions. Figure 5.3 contains scatterplots and Pearson correlation coefficients of the pairwise protein combinations.

The scatterplots, in Figure 5.3, show that the pair-wise relations between the levels of proteins can be approximated to be either linearly correlated or uncorrelated. Through the Pearson correlation coefficients, the levels of the analyzed proteins were found to be positively correlated with each other with different strengths. This indicates that the instrumental variation is smaller than the biological variation in protein levels, as correlations would otherwise be masked. The correlations between the levels of FAK and FAK-pY397, as well as between paxillin and paxillin-pY118, in focal adhesions are moderate, indicating that the relative extent of FAK and paxillin phosphorylation is heterogeneous among focal adhesions. The extent of stoichiometry conservation between each two proteins across focal adhesion does not correlate with the extent of their physical association in the cytosol (Hoffmann et al., 2014), see also Figure 5.4. This indicates that the compositional stoichiometry of focal adhesions is actively and locally shaped by them, rather than being a passive reflection of the building blocks design in the cytosolic pool (Harizanova et al., 2016). Strikingly, the strongest correlation is between the densities of zyxin and paxillin, although these proteins do not interact directly nor physically in the cytosol and are located in distal vertical layers across focal adhesions (Harizanova et al., 2016). Zyxin and paxillin appear to form hubs of order within focal adhesions, since proteins that are vertically adjacent to them (FAK, vinculin and VASP) exhibit relatively more conserved stoichiometric ratios among them but not with the distally located α-actinin and F-actin (Harizanova et al., 2016).

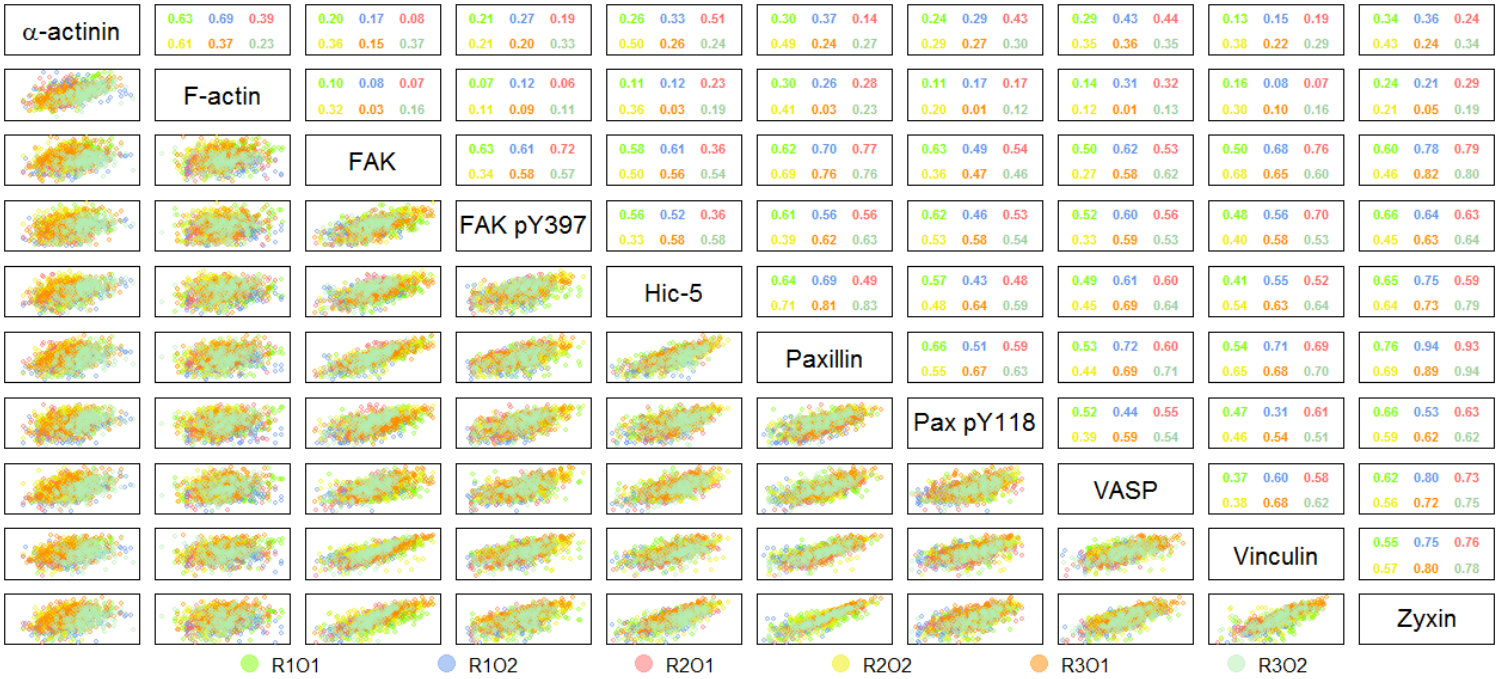


Figure 5.3: Scatterplots and Pearson correlation coefficients of Box-Cox transformed and normalized protein levels for each dataset. For the visualization, the scatter plots were depicted by using random samples of 10% of the total of focal adhesions per datasets.

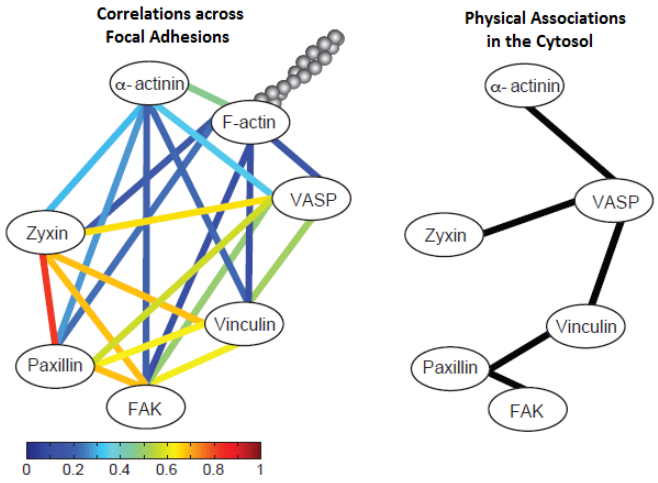


Figure 5.4: Superposition of the correlations with the reported vertical positions of the proteins across focal adhesions (Hoffmann et al. (2014)) (left) and a comparison with their reported physical associations in the cytosol (right).

Although in the previous section a significant difference between the distributions of transformed levels of proteins did not become evident, here a comparison of Pearson correlation coefficients of the pairwise protein combinations over the six datasets seems to indicate otherwise (see Figure 5.3). However, there is no reason to assume that the discrepancies between the samples are due to the different cycles and labelings used in the employed measuring technique, but rather to an underlying variation of the phenomenon under study, either due to a variation between cells, focal adhesion or to factors not considered here.

5.3 Aim 1: Finding different combinations of input proteins suitable to predict another target protein

Artificial neural network (ANNs) and random forests (RFs) were applied as described in Section 4.2 for testing and quantifying how well the levels of a subset of proteins (called here input proteins) enable the prediction of the level of another protein (termed the target protein)

in focal adhesions. To evaluate the agreement between the results of both methods we employed scatterplots of the resulting RMSE's from ANNs and RFs labeled by dataset (Figure 5.5) and plots of the mean of ANN's RMSE and RF's RMSE versus the differences between both RMSE's labeled by number of input proteins (Figure 5.6). Both figures indicate that the RMSE's obtained from the two methods are strongly correlated, with marginally better predictions by artificial neural networks. Additionally in Figure 5.6 we appreciate that both analysis methods indicate that a joint consideration of multiple proteins improves the predictability of each target protein. Note also in this figure that the accuracy of RF-adjusted models increases with the number of input proteins considered, whereas ANNs tend to be a more parsimonious method. Because of the results observed here, we continue the analyses using the outputs from ANNs. The analysis of neuronal network was made by Dr. Jana Harizanova for the anaysneurnal networks results from in Matlab version 7.10 (R2010a).

5.4 Aim 2: Identifying common findings among different datasets

The common results over the six datasets were found by selecting input protein combinations suitables to explain a target protein. To this end we employed hypothesis tests where the null hypothesis corresponds to a linear regression with intercept equal 1 and constant 0, this methodology is described in Section 4.4. The hypothesis tests were conducted by setting the significance level at 5%.

The number of combinations of input proteins suitables to explain a target protein found in each dataset are shown in Figure 5.7. Note that in almost all the cases these numbers of input combinations are over 200 which is a big quantity. This is because two things: (1) most of the input proteins are correlated to each other, which leads to different combinations among them to predict the target protein in similar degree of accuracy, and (2) the hypothesis tests conducted in this form are very flexible because they tend to select many input protein

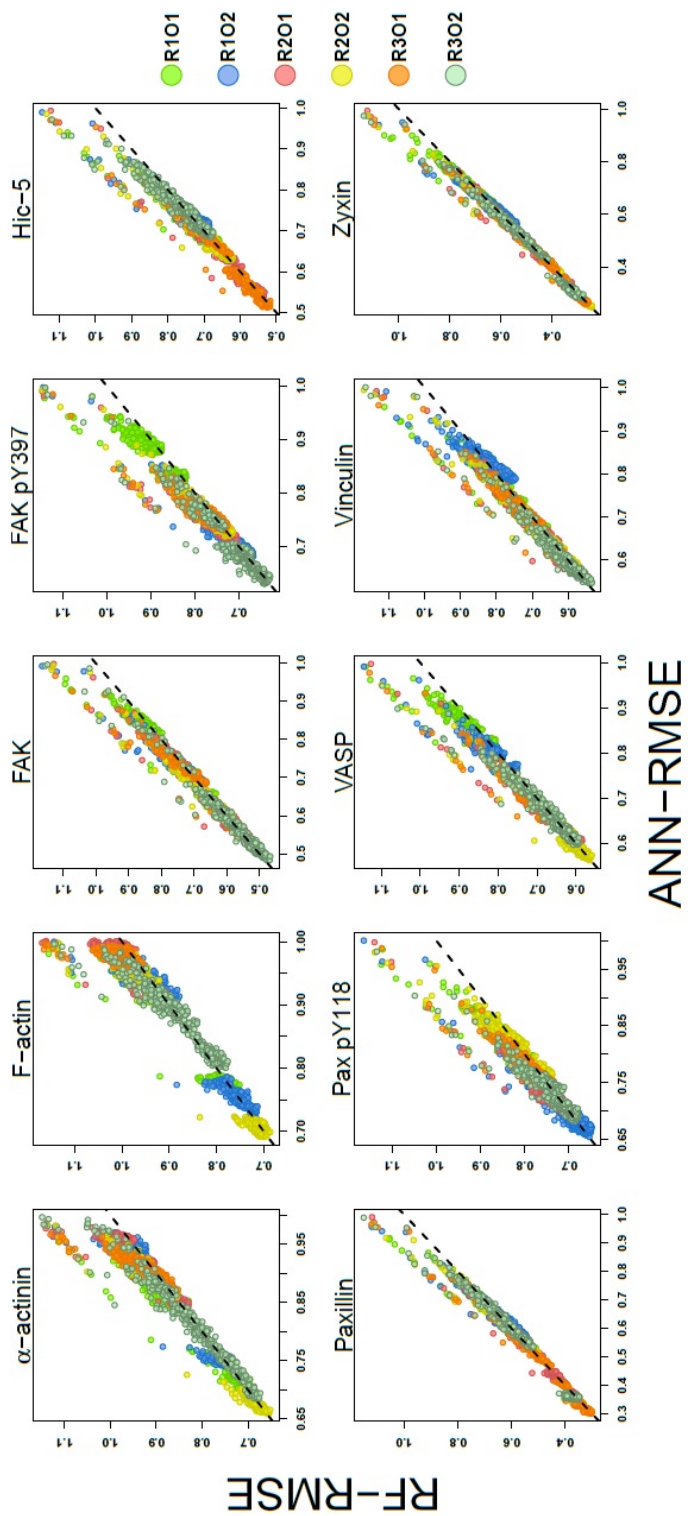


Figure 5.5: Correlation between the results of ANNs and RFs. Comparison through the root mean squared error (RMSE) of the observed values with the predicted values. All possible combinations of input and target proteins are plotted and datasets are labeled by colors.

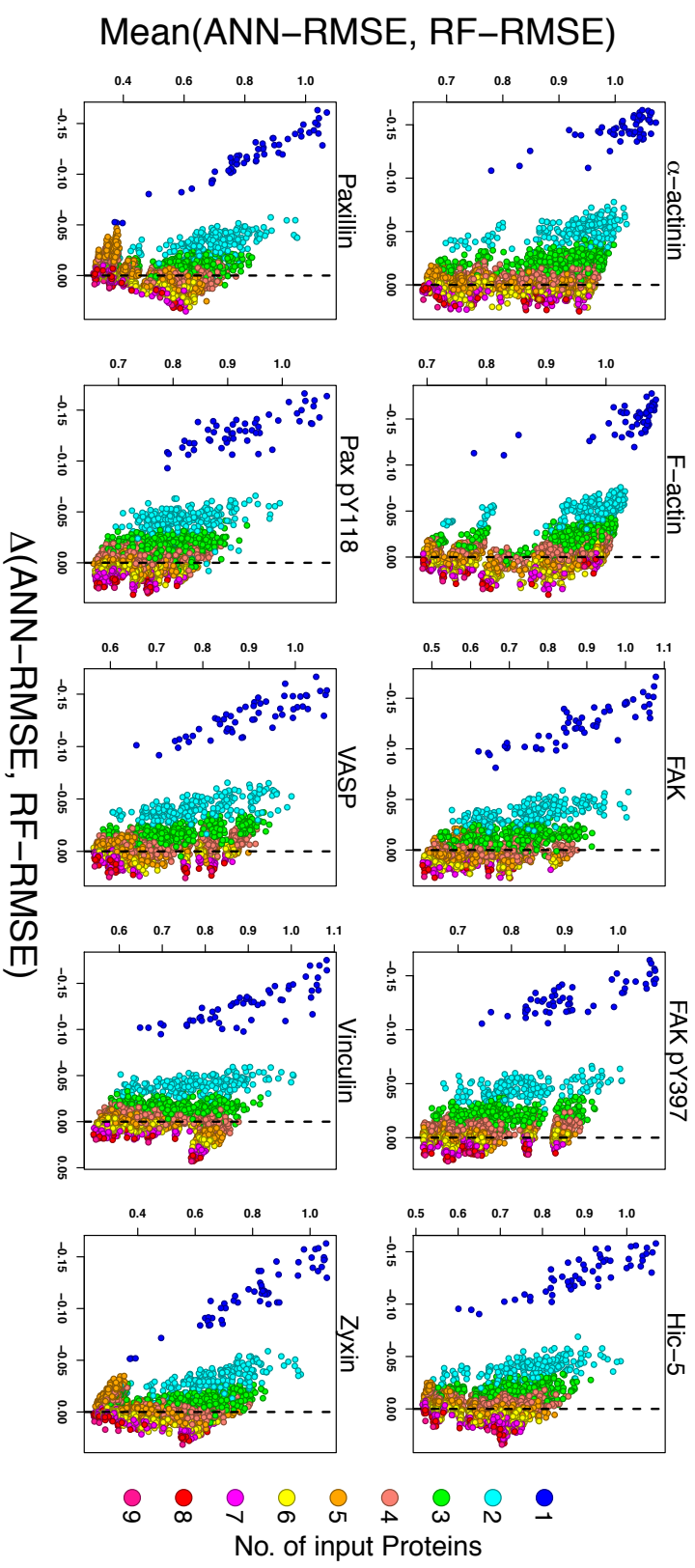


Figure 5.6: Plot of the mean of the root mean squared error of artificial neuronal network (ANN-RMSE) and random forests (RF-RMSE) against between the differences from the RMSE of both methods labeled by the number of input proteins. When a mean of RMSE's is lower than a difference then it can be said that the method is better than the other.

combinations. However, that can be seen here as an advantage instead of a disadvantage, since without this property it would not be possible to find common results over the different datasets. Moreover, we use together with this technique the coefficient of determination, which makes the comparison among the candidate input protein combinations intuitively interpretable. This allows the researcher to choose a combination of input proteins that best explains the target protein or a more parsimonious input combination.

Through a systematic search over the candidate input protein combinations in the six datasets, we selected the common ones among datasets. The number of common input protein combinations found on the six datasets can be observed in Figure 5.8. Note that the number of common combinations among datasets is relatively small compared to the number of input protein combinations found in each dataset. Again, this is indicative of how datasets differ from each other and that without a flexible method as the one used above we could not be able to find common results among the different datasets.

5.5 Aim 3: Merging common findings

Because of variability within and among datasets, the common combinations of input proteins over the six datasets do not have the same hierarchy of accuracy in each dataset. That is, a model with a particular combination of input proteins can have the highest determination coefficient in dataset R1O1 and the third highest one in dataset R1O2. Therefore in order to establish a general hierarchy among candidate models over the six datasets, we merged the common findings by performing meta-analysis on the six datasets using a random-effects model (RE model) on the Pearson correlation coefficients as defined in section 3.4. The final results of this analysis are transformed in terms of determination coefficient by squaring the RE correlation coefficients and Figure 5.9 illustrates the confidence intervals resulting of this analysis.

Systematic screen of all combinations of input and target proteins revealed paxillin and zyxin levels to be significantly better predicted upon integration of multiple input proteins (Figure 5.9).

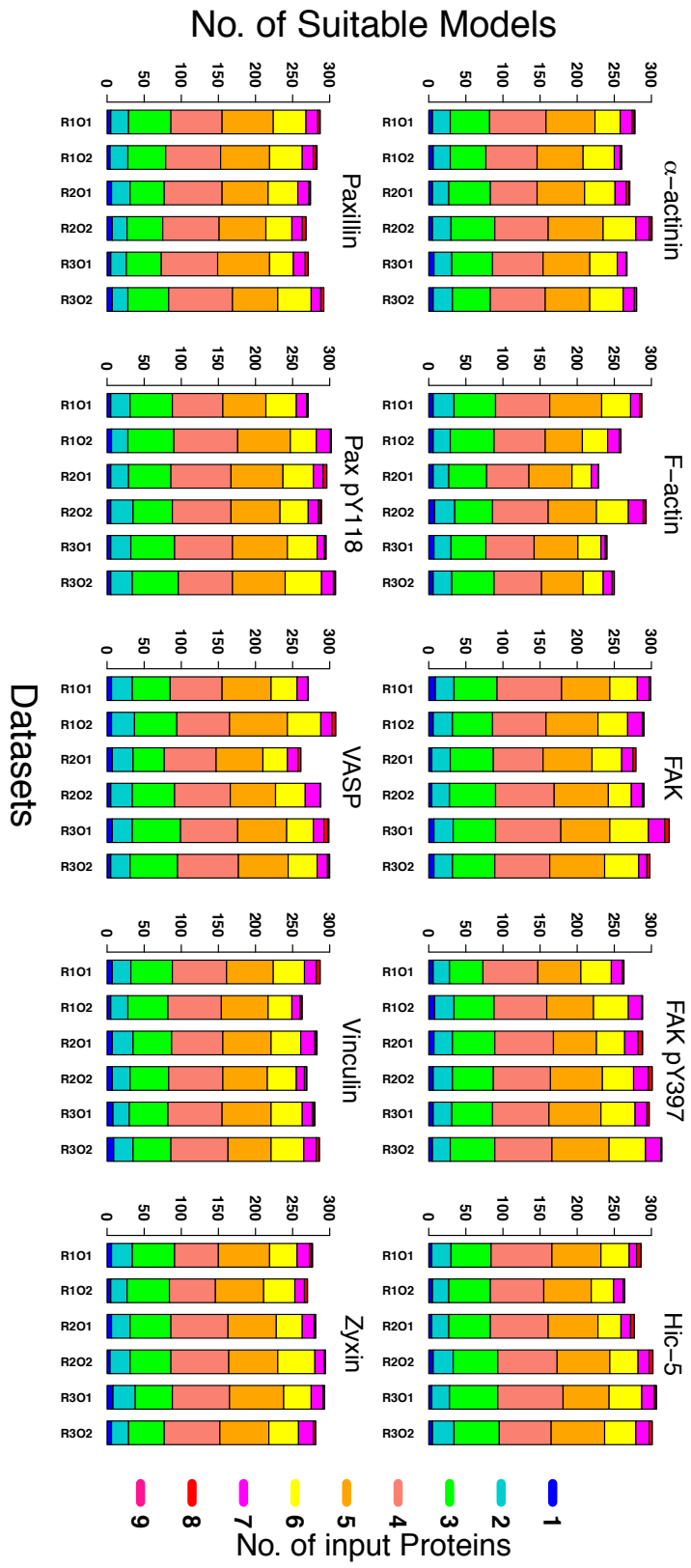


Figure 5.7: Bar plots indicating for each dataset the number of input protein combinations suitable to explain a target protein.

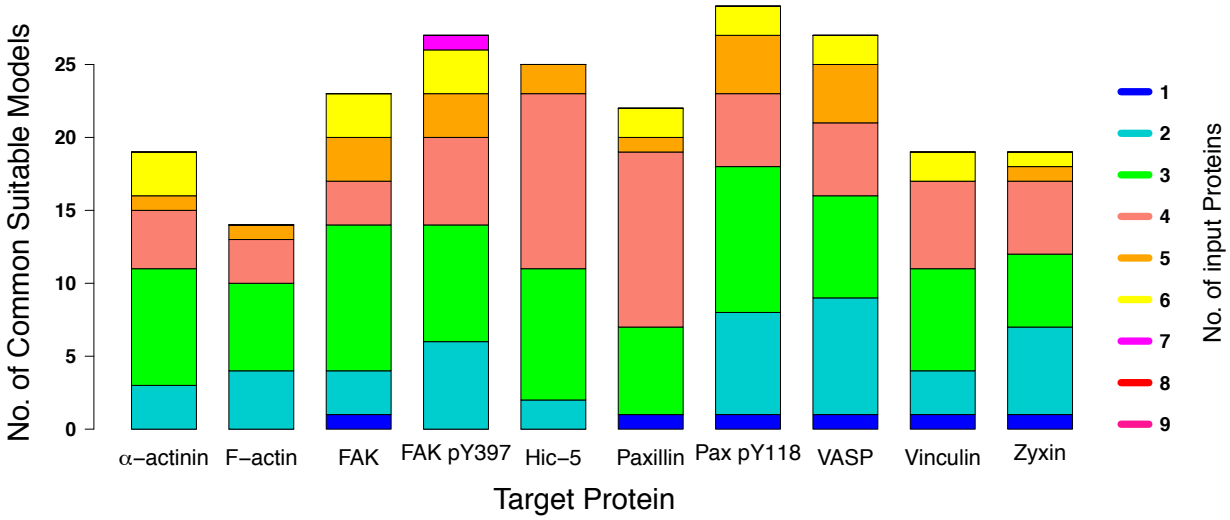


Figure 5.8: Bar plot illustrating the number of common input protein combinations among the six datasets.

Of note, this is not a side effect of the strong pairwise correlation between paxillin and zyxin (Figure 5.3) but rather indicates a statistically significant incremental contribution of each of the input proteins to the prediction of paxillin or zyxin levels. This suggests that paxillin and zyxin levels in focal adhesions are fine-tuned by integrating the levels of other multiple proteins, thus averaging-out stochastic fluctuations. It is also interesting to observe that proteins as paxillin and zyxin do not require of more than six components to be predicted with a high degree of precision, even note that in case of zyxing the best accuracy is achieved with only three input proteins.

In appendix Section A we provide to the readers a list of input protein combinations for each target protein. This list helps to select the set of input proteins that best predicts the target protein or the most parsimonious set of input proteins.

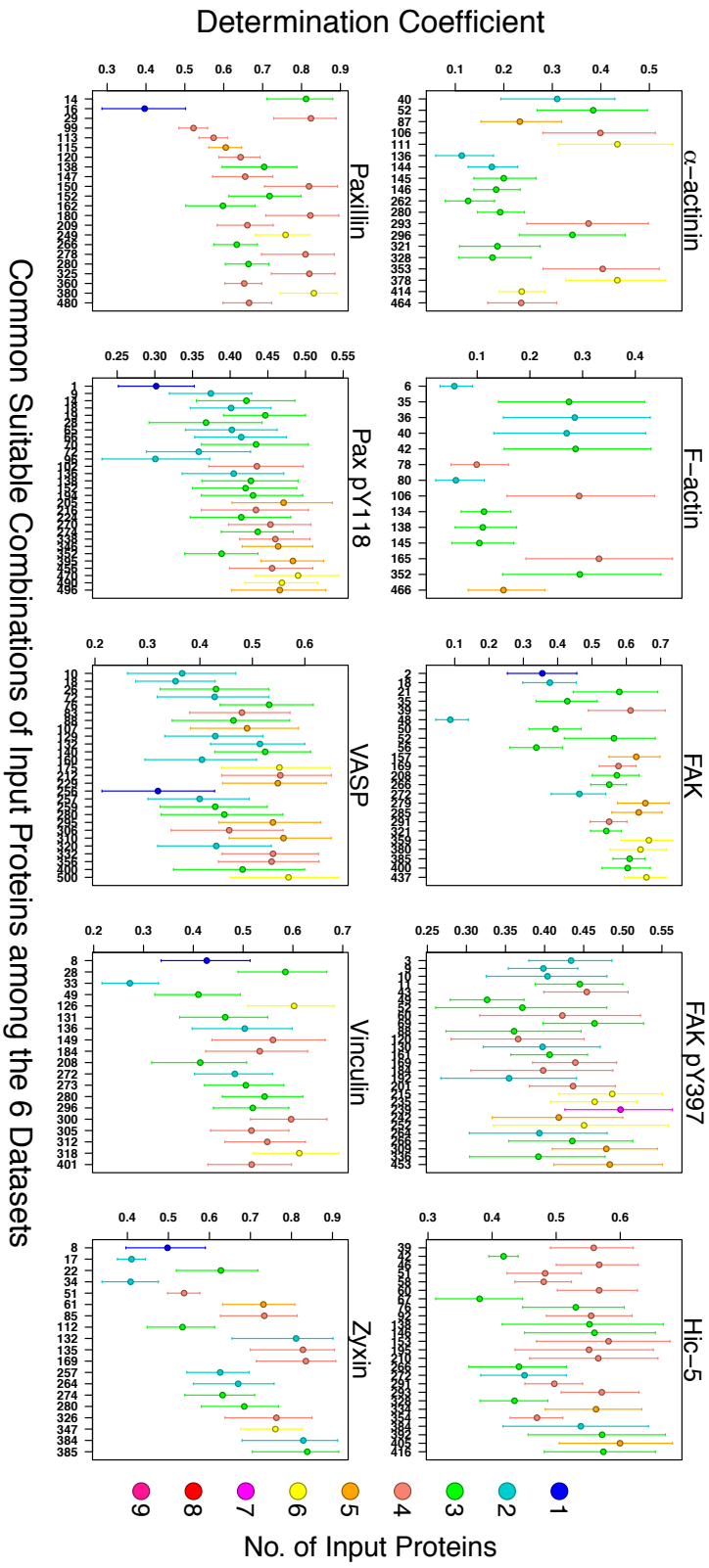


Figure 5.9: Confidence intervals for the coefficient of determination resulting of a meta-analysis using random-effects models on common findings among the six datasets for each protein considered here.

6. Conclusions

Although it is already accepted or assumed by the scientific community of this field that all proteins involved in the formation of focal adhesions interact with each other, we intend here to see how the levels of one group of these proteins affect the levels of another target protein, in an explanatory sense. It can provide information on how the existence of a protein or a group of proteins is essential for the presence of another target protein. This study based on the analysis of six datasets, where each one led to different results. Discrepancies that according to the results are assumed to be from variation between cells, differences between focal adhesions or due to influential factors no observables, rather than from a systemic variation. The discrepancy between the individual analysis of these datasets leads to a problem in the integration of results. Here we proposed a random effects model of meta-analysis for a combination of the results from these datasets.

It is assumed that in the presence of non-variation the results between different datasets should be the same. To find common results among datasets we carried out different types of procedures such as the use of the determination coefficient along with a hypothesis test, and the use of random-effects models (RE models) to merge the common results among the different datasets. The method of RE models gave us the possibility to introduce in the merging of the outputs the variation among datasets and within each dataset. Our procedure showed to be very flexible in the sense that conducted us to select more than one combination of input proteins for a target protein. However, it was also commented here the importance of this property to find common results among the six datasets. As a final result, we provide the readers with a

list of input protein combinations for each target protein so they can decide which aspect of the results is of greater importance. That is if they consider more appropriate for their research to select the input protein combined with the highest accuracy in predicting or to select a more parsimonious input combination.

On the other hand, we found that paxillin and zyxin are proteins that are well explained by groups of proteins also considered here. These, in turn, belong in the majority of the cases to the group of input proteins that best describe the rest of the proteins. Now, it is essential to keep in mind that we can observe only a part of a complex system of interacting proteins that could contain over 100 proteins. Thus, those proteins that do not seem to be well explained by the rest of the proteins considered can be well defined by other proteins in the system that are not found. For example, note the fact that one protein that interacts with another, it does not require anything additional, but other related components present. Suppose paxillin and zyxin were not present in the analysis, then many of the other elements would not be there. I believe that such type of system should be considered as a whole, in the sense that only the system could be understood if we can observe all the components involved at once instead of observing independently different groups of them. When we watch a part of the system, we notice that the stoichiometries of proteins brooks in the FAs, which could lead to differences among FAs. It must be of concern to biologist because incorrect understandings of this kind of process could lead to the development of drugs or treatments for a particular disease that may lead to the development of new diseases or body reactions still unknown. It is clear that technical resources are not enough at this time to observe and understand this system as a whole. However, I am confident that the scientific community in this field could do more in the development of new measurement techniques or in counteracting the shortcomings of current technologies.

Regarding the statistical methods here employed, it is clear that both approaches, random forests and artificial neural networks, are comparable in accuracy. Then, it depends on what the user is looking for: A good prediction or interpretable results. In this work, the aim was fixed to have a prediction as good as we can, that is why we used the outputs of ANNs. The

techniques we used here to merge different datasets has worked well in the sense that most of the combination of input proteins that are common and significant among the different data sets were here detected. However, more that one protein input combination is here selected, because of the high correlation between all the associated proteins.

7. References

1. Y. Amit and D. Geman, Shape quantization and recognition with randomized trees. Neural Computation, 9:1545–1588, 1997.

2. C. Banyasz. Adaptive Systems in Control and Signal Processing 1995. IFAC Postprint Volume. Elsevier Science, 2014. ISBN 9781483296890.

3. G. E. P. Box and D. R. Cox. An analysis of transformations. Journal of the Royal Statistical Society. Series B (Methodological, pages 211–252, 1964.

4. L. Breiman. Bagging predictors. Machine Learning, 24(2):123–140, 1996.

5. L. Breiman. Random forests. Machine Learning, 45(1):5–32, 2001.

6. B. Geiger, J. P. Spatz, and A. D. Bershadsky. Environmental sensing through focal adhesions. Nature Reviews Molecular Cell Biology, 10(1):21–33, 2009. doi: 10.1038/nrm2593.

7. B. M. Gumbiner. Cell adhesion: the molecular basis of tissue architecture and morphogenesis. Cell, 84(3):345–357, 1996.

8. D. Hanein and R. Horwitz. The structure of cell-matrix adhesions: the new frontier. Current Opinion in Cell Biology, 24(1):134–140, 2012. doi: 10.1016/j.ceb.2011.12.001.

9. J. Harizanova, Y. Fermin, R. S. Malik-Sheriff, J. Wieczorek, K. Ickstadt, H. E. Grecco, and E. Zamir. Highly multiplexed imaging uncovers changes in compositional noise within assembling focal adhesions. PLoS ONE, 11(8), 2016. doi: 10.1371/journal.pone.0160591.

10. T. Hastie, R. Tibshirani, and J. Friedman. The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition. Springer Series in Statistics. Springer New York, 2009. ISBN 9780387848587.

11. T. K. Ho. The random subspace method for constructing decision forests. IEEE Trans. Pattern Anal. Mach. Intell., 20(8):832–844, 1998.

12. J.-E. Hoffmann, Y. Fermin, R. L. Stricker, K. Ickstadt, and E. Zamir. Symmetric exchange of multi-protein building blocks between stationary focal adhesions and the cytosol. eLife, 3:e02257, jun 2014. ISSN 2050-084X. doi: 10.7554/eLife.02257. URL https://dx.doi.org/10.7554/eLife.02257.

13. R. O. Hynes and A. D. Lander. Contact and adhesive specificities in the associations, migrations, and targeting of cells and axons. Cell, 68(2):303–322, 1992.

14. A. Liaw and M. Wiener. Classification and regression by randomforest. R News, 2:18–22, 2002.

15. M. Liu, M. Wang, J. Wang, and D. Li. Comparison of random forest, support vector machine and back propagation neural network for electronic tongue data classification: Application to the recognition of orange beverage and chinese vinegar. Sensors and Actuators B: Chemical, 177:970–980, 2013.

16. MATLAB. version 7.10.0 (R2010a). The MathWorks Inc., Natick, Massachusetts, 2010.

17. W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. Numerical Recipes in C (2Nd Ed.): The Art of Scientific Computing. Cambridge University Press, New York, NY, USA, 1992. ISBN 0-521-43108-5.

18. R Core Team. R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria, 2013. URL http://www.R-project.org/.

19. V. Rodriguez-Galiano, M. Sanchez-Castillo, M. Chica-Olmo, and M. Chica-Rivas. Machine learning predictive models for mineral prospectivity: An evaluation of neural networks,

random forest, regression trees and support vector machines. Ore Geology Reviews, 71:804–818, 2015.

20. R. Schulze, H. Holling, and D. Boehning. Meta-Analysis. Hogrefe Publishing, 2002. ISBN 9781616762667.

21. W. Sun and Y. Yuan. Optimization Theory and Methods: Nonlinear Programming. Springer Optimization and Its Applications. Springer US, 2006. ISBN 9780387249766.

22. R. Tadeusiewicz, R. Chaki, and N. Chaki. Exploring Neural Networks with C#. Taylor & Francis, 2014. ISBN 9781482233407. J. W. Tukey. Exploratory Data Analysis. Addison Wesley, 1977.

23. K. Were, D. T. Bui, Ø. B. Dick, and B. R. Singh. A comparative assessment of support vector regression, artificial neural networks, and random forests for predicting and mapping soil organic carbon stocks across an afromontane landscape. Ecological Indicators, 52:394–403, 2015.

24. J. Wieczorek, R. S. Malik-Sheriff, Y. Fermin, H. E. Grecco, E. Zamir, and K. Ickstadt. Uncovering distinct protein-network topologies in heterogeneous cell populations. BMC Systems Biology, 9:24, 2015. doi: 10.1186/s12918-015-0170-2. URL http://dx.doi.org/10.1186/s12918-015-0170-2.

25. E. Zamir and B. Geiger. Molecular complexity and dynamics of cell–matrix adhesions. Journal of Cell Science, 114 (20):3583–3590, 2001.

26. Z. Zeng and J. Wang. Advances in Neural Network Research and Applications. Springer Publishing Company, Incorporated, 1st edition, 2010. ISBN 3642129897, 9783642129896.

Part II

Modelling of temporal networks with a nonparametric mixture of dynamic Bayesian networks

8. Introduction

Many of the cellular processes are carried out by macromolecular assemblies, known as cell-matrix adhesion sites, or also known as focal adhesions. Focal adhesions are formed through the interactions of proteins. Proteins are interacting continuously with the environment of the cells and in between. Many of this interaction are randomly produced, and therefore the protein interaction network is still unknown. In the reconstruction of cellular protein interaction networks, methods such as machine learning methods (Qi et al., 2009; Harizanova et al., 2016; Ganapathiraju et al., 2016), differential equations (Kauffman, 2009; Chen et al., 1999) and Bayesian networks (BNs) (Liang et al., 1998; Imoto et al., 2002; Friedman et al., 1999) have been applied. Among the machine learning methods, artificial neural networks (ANNs) is the most effective for discovering protein interactions (Harizanova et al., 2016). However, unlike artificial neural networks, differential equations and BNs allow capturing in more detail the regulatory relationships in the network, as for example the direction of regulation. A disadvantage of differential equations is the requirement of prior knowledge of kinetics parameters associated to the interactions between proteins, while BNs employ a probabilistic mechanism for the identification of protein-protein interaction, which as ANNs require only the quantification of protein levels in a molecular sample. BNs are graphical probabilistic models in which directed acyclic graphs are used to represent the relationships between proteins (Pearl, 1988) graphically. This compact, easy-to-interpret solution makes BNs a preferred model in practice.

The major challenge in the reconstruction of protein-protein interaction networks is the cell-to-cell heterogeneity within a sample, due to genetic and epigenetic variabilities. This heterogeneity

hampers the usage of parametric models like those mentioned in the previous paragraph. Such models will lead to a degradation of the actual network due to lack of adjustment. To address this, recent studies (Hasenauer et al., 2014; Wieczorek et al., 2015) have suggested the identification of cellular subpopulations through the use of mixture models for a next prediction of the protein-protein interaction network. Hasenauer et al. (2014) use mixtures of ordinary differential equations, while in Wieczorek et al. (2015) a nonparametric Bayesian mixture of Gaussian BNs called nonparametric Bayesian networks (NPBNs) was employed. Both works showed the success of using mixtures models in conjunction with the recognition of the protein-protein interaction for the classification of observations in heterogeneous cell populations. The limitation of both methods is that they were developed on snapshot data of a dynamic process, what makes them fail to capture temporal information and modeling of cyclic networks. Thus, these methods can be applied only on the type of single-cell multiparametric measurements currently available such as multicolor flow-cytometry, multiplexed mass cytometry, and toponome imaging. In recent years, there has been an essential transition from imaging the static distributions of molecules as a snapshot in time in fixed material to live-cell imaging of the dynamics of molecules in cells. There have been significant advances in so-called "super-resolution" imaging methods that have overcome the resolution limits imposed by the diffraction of light in optical systems (Ball et al., 2012). By using high spatiotemporal resolution, imaging provides an opportunity for a better understanding of biological function through the study of cellular dynamics (Monya, 2010). The analysis of these data requires the modeling of temporal relationships. Thus, we address this piece of work to the extension of the NPBNs by using Gaussian dynamic Bayesian networks (GDBNs), which are better suited for characterizing time series expression data than the static version GBNs. The use of GDBNs has been quite extensive in the reconstruction of gene regulatory networks structures (see, e.g. Murphy and Mian (1999); Dojer et al. (2006); Kim et al. (2003); Grzegorczyk et al. (2008)). However, any of them can deal with cell-to-cell variability. This could lead in cases of heterogeneity to poorly specified interaction models. Here, through a nonparametric Bayesian mixture of dynamic Bayesian networks, we build a method that allows the successful

reconstruction of protein interaction networks in heterogeneous cell populations. This approach is here called "nonparametric mixture of dynamic Bayesian networks" (NPMDBNs).

This work is described in 7 Chapters. Chapter 9 contains the methods involved in the development of NPMDBNs. In Chapter 10 we defined the NPMDBNs. In Chapters 11 and 12 the algorithms and procedures involved in the implementation of NPMDBNs are described. Chapter 13 contains the evaluation methodology. In Chapters 14 and 15 the results of the evaluation and the conclusions are respectively presented.

9. Methods

9.1 Basic concepts about graphs

This section briefly introduces the main concepts and notations of graph theory used in the next sections. For more definitions see e.g. Gould (1988) and Cowell et al. (1999).

When a collection of objects $\mathbf{V} = \{V_1, \ldots, V_n\}$ that are related to each other by edges is graphically represented, being E_{ij} the edge joining the elements V_i and V_j of $\mathbf{V}$, we are implicitly defining a graph, where $\mathbf{V} = \{V_1, \ldots, V_n\}$ are the nodes and $\mathbf{E}$ the set of edges that form it. More formally speaking by:

Definition 9.1.1 (Graph)

A graph is represented as a pair $G = (\mathbf{V}, \mathbf{E})$ *where* $\mathbf{V} = \{V_1, \ldots, V_n\}$ *is the finite set of nodes or vertices and* $\mathbf{E} \subseteq \mathbf{V} \times \mathbf{V}$ *is the set of edges, that form it, the set of ordered pairs of distinct elements of* $\mathbf{V}$ *that are related.*

Depending on the relation and order between the nodes of G, one can speak of two types of edges: directed edges and undirected edges. Thus, directed edges are used when $E_{ij} \in \mathbf{E}$ but $E_{ji} \notin \mathbf{E}$, and it is denoted as $V_i \rightarrow V_j$, so that V_i is connected with V_j and not vice-versa. While undirected edges, which notation is $V_i - V_j$, occur when $E_{ij} \in \mathbf{E}$ and $E_{ji} \in \mathbf{E}$, being both nodes V_i and V_j connected. The type of edge can determine the graph, so if all edges of G are directed (undirected) then G is said to be directed (undirected) and when G has directed and undirected edges, G is said mixed graph. Associated with a directed G there is always an undirected version of G that is obtained by replacing the directed edges of the graph by

undirected edges and the new G is called skeleton. Figure 9.1 shows a graph with two types of edges.

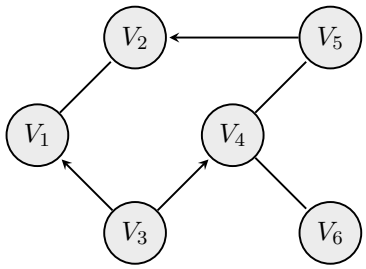


Figure 9.1: Mixed graph with six nodes.

Another important term to be here defined is parents and children:

Definition 9.1.2 (Parents and children)
If $G = (\mathbf{V}, \mathbf{E})$ is a directed graph, then a node V_i is a parent of node V_j if $E_{ij} \in \mathbf{E}$. Moreover, if V_i is parent of V_j, then V_j is said to be child of V_i. Thus, for any node $V_j \in \mathbf{V}$ of a given graph G the set $\mathbf{Pa}_j = \{V_i \in \mathbf{V} | V_i \to V_j\}$ defines the set of parents of V_j.

From the next section, Bayesian networks, the term directed acyclic graph will be often used, here the meaning:

Definition 9.1.3 (Directed acyclic graph)
A graph $G = (\mathbf{V}, \mathbf{E})$ is called a directed acyclic graph (DAG) if it is a directed graph where for any node $V \in \mathbf{V}$ there are no cycles. That is, there is no way that an edge starts and ends at the same node.

9.2 Bayesian networks (BNs)

In this section, we introduce a brief definition of Bayesian networks (BNs). For additional material see e.g. Pearl (1988), Heckerman and Geiger (1995) and Koski and Noble (2009).

BNs are the most widely used technique of graphs related to probability theory. They are

fairly flexible models and easy to interpret. A BN consists of a qualitative and a quantitative part. The qualitative part is specified by a directed acyclic graph DAG $G = (\mathbf{X}, \mathbf{E})$, where the nodes in G represent a set of random variables, $\mathbf{X} = \{X_1, \ldots, X_n\}$, and the directed edges $\mathbf{E}$ between nodes represent conditional dependencies. The directed edges in G, imply several (in-)dependence constraints, which are described by the local directed Markov condition:

Definition 9.2.1 (Local directed Markov condition)

Let $\mathbf{X} = \{X_1, \ldots, X_n\}$ be a set of random variables and G a BN graph structure over $\mathbf{X}$. Then G satisfies the local directed Markov condition if for every $X_i \in \mathbf{X}$, variable X_i is stochastically independent of its non-descendants given its parents in G.

Pearl (1988) provides a graphical condition called d-separation that can be used to identify the entire set of (in-)dependence constraints that are implied by BN structures. Figure 9.2 shows a simple example of a BN graph with four variables, $\mathbf{X} = \{X_1, X_2, X_3, X_4\}$. The intuitive meaning of the BN structure in Figure 9.2 is that X_1 and X_3 are marginally dependent, $p_{\{X_1,X_3\}|G}(x_1, x_3|g^*) \neq p_{X_1}(x_1)p_{X_3}(x_3)$, but once we know node X_2, node X_3 becomes independent of node X_1, $p_{X_3|\{X_1,X_2\},G}(x_3|x_1, x_2, g^*) = p_{X_3|X_2}(x_3|x_2)$. Another statement implied by this DAG is that, the nodes X_2 and X_4 are marginally independent, $p_{\{X_2,X_4\}|G}(x_2, x_4|g^*) = p_{X_2}(x_2)p_{X_4}(x_4)$, but because of the nodes X_2 and X_4 have the common descendant X_3 they become dependent when conditional on X_3, $p_{\{X_2,X_4\}|X_3,G}(x_2, x_4|x_3, g^*) \neq p_{X_2|X_3}(x_2|x_3)p_{X_4|X_3}(x_4|x_3)$.

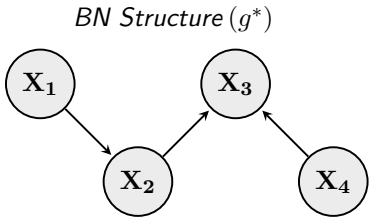


Figure 9.2: Directed acyclic graph (DAG) with four variables.

The quantitative part in a BN represents the joint probability distribution over the variables,

indexed by a parameter set $\boldsymbol{\theta}$ associated to the BN graph. Thus, the joint probability distribution can be obtained as the product of the conditional distributions defined by the conditional dependencies represented in G. The graph in Figure 9.2 lead, for example, to the following joint probability distribution:

$$p_{\mathbf{X}|\boldsymbol{\Theta},G}(\mathbf{x}|\boldsymbol{\theta}, g^*) = p_{X_1|\boldsymbol{\Theta}_1}(x_1|\boldsymbol{\theta}_1)p_{X_2|X_1,\boldsymbol{\Theta}_2}(x_2|x_1,\boldsymbol{\theta}_2)p_{X_3|\{X_2,X_4\},\boldsymbol{\Theta}_3}(x_3|x_2,x_4,\boldsymbol{\theta}_3)p_{X_4|\boldsymbol{\Theta}_4}(x_4|\boldsymbol{\theta}_4).$$

A general definition is given as follows:

$$p_{\mathbf{X}|\boldsymbol{\Theta},G}(\mathbf{x}|\boldsymbol{\theta}, g) = \prod_{i=1}^{n} p_{X_i|\mathcal{U}_i,\boldsymbol{\Theta}_i}(x_i|\mathbf{u}_i,\boldsymbol{\theta}_i). \tag{9.1}$$

where the $\boldsymbol{\Theta}_i$'s are the node-specific subvectors, which specify the local conditional distributions and $\mathcal{U}_i = \{X_j \in \mathbf{X} | X_j \longrightarrow X_i\}$ is the parent set of X_i.

In practice, often the conditional probability distributions are assumed to be a member of some specific parametric family of distributions, where the choice depends on the variable and the parent set. In the BNs literature we can find two dominant examples: (1) When all variables are discrete, the joint distribution is become an unrestricted discrete distribution and the conditional probability distributions are independent multinomials for each variable and each parent configuration. (2) When all variables are continuous, the joint distribution is defined as a multivariate Gaussian and the conditional probability distributions are independent Gaussians with mean structured as a linear regressions of each variable on its respective parent set, see Definition 9.2.2.

Definition 9.2.2 (Gaussian Bayesian networks)

A Gaussian Bayesian network is a Bayesian network where the conditional distributions, $p_{X_i|\mathcal{U}_i,\mathbf{\Theta}_i}(x_i|\mathbf{u}_i,\boldsymbol{\theta}_i)$*, are given by normal distributions of the form:*

$$p_{X_i|\mathcal{U}_i,\mathbf{\Theta}_i}(x_i|\mathbf{u}_i,\boldsymbol{\theta}_i) \sim \mathrm{N}(x_i|\mu_i + \sum_{\mathcal{J}_i} \beta_{i,j}(x_j - \mu_j), \sigma_i^2), \tag{9.2}$$

where $\mathcal{J}_i = \{j \in \{1,\ldots,n\}|X_j \in \mathcal{U}_i\}$*,* $\boldsymbol{\theta}_i = (\mu_i, \sigma_i^2, \beta_i)$*, the* μ_i *are the unconditional means of* X_i*,* $\boldsymbol{\beta}_i = (\beta_{i,j})_{j\in\mathcal{J}_{U_i}}$ *is a set of real coefficients determining the influence of each* $X_j \in \mathcal{U}_i$ *on* X_i *and* σ_i^2 *is the conditional variance of* X_i *given its parents,* $\mathcal{U}_i$*.*

One of the most relevant issues to deal with BNs is that for a set of variables $\mathbf{X}$, more than one DAG may be selected to represent exactly the same set of conditional independence relationships. For example, for a given set $\mathbf{X} = \{X_1, X_2, X_3\}$, the three network structures, $G1$: $X_1 \rightarrow X_2 \rightarrow X_3$, $G2$: $X_1 \leftarrow X_2 \rightarrow X_3$ and $G3$: $X_1 \leftarrow X_2 \leftarrow X_3$ imply the same statements: X_1 and X_3 are conditional independent given X_2. These graphs are in practice called Markov equivalent and the set of them Markov equivalence class. Verma and Pearl (1991) define a way to characterize similar classes more precisely: *Two DAGs are equivalent if and only if they have the same skeleton and the same v-structures*. A v-structure in a DAG G is an ordered triplet of nodes, (X_1, X_2, X_3) , such that G contains the directed edges $X_1 \rightarrow X_2$ and $X_2 \leftarrow X_3$, and the nodes X_1 and X_3 are not adjacent in G. Thus, the Markov equivalence classes can be uniquely represented by a completed partially directed acyclic graphs (CPDAGs) (Grzegorczyk et al., 2008). A CPDAG is defined as a graph with the same skeleton as each DAG in its equivalence class and its edges consisting of a directed edge for every compelled edge and undirected edge for every reversible edge. An edge is compelled if its reversal changes the set of v-structures. Chickering (2002), cited in Grzegorczyk et al. (2008), provides an algorithm that converts a DAG to its corresponding (unique) CPDAG.

9.2.1 Gaussian score for learning BNs

The learning of a BN is usually performed in two separate steps. First, the graph structure G is learned from the data, and second, the parameter distribution set $\mathbf{\Theta}$ is estimated conditional on G. In practice, it is more common to have an interest only in the graph structure than in the parameter set. Thus, this section will be concerned only to the learning graph procedure.

One of the most popular methods for learning BN structures from data is the score-based approach. Let $\mathbf{D}$ be a data set with m independent realizations of the random vector $\mathbf{X} = (X_1, \ldots, X_n)^\top$. The aim of this approach is to identify one or more BN structures that best describe the set of observed data $\mathbf{D}$ according to some scoring criterion. Assuming a BN structure g, the score is defined as follows:

$$S(g; \mathbf{D}) = p_{G|\mathbf{X}}\left(g|\mathbf{D}\right). \tag{9.3}$$

The score quantifies how much g is supported by the observed data $\mathbf{D}$. An score-based method attempts to optimize Eq. 9.3, that is, to find the BN graph with the highest posterior probability.

By using Bayes' theorem the posterior distribution of g given the observed data $\mathbf{D}$ can be obtained as follows:

$$p_{G|\mathbf{X}}\left(g|\mathbf{D}\right) = \frac{p_G\left(g\right) \mathcal{L}_{\mathbf{X}|G}\left(g|\mathbf{D}\right)}{\sum_{g^{(\tau)} \in \mathcal{G}} p_G(g^{(\tau)}) p_{\mathbf{X}|G}(\mathbf{D}|g^{(\tau)})}, \tag{9.4}$$

where $\mathcal{G}$ denotes the set of all valid DAG's. The denominator can be seen as a normalization constant to ensure that $\sum_{g^{(\tau)} \in \mathcal{G}} p_{G|\mathbf{X}}\left(g^{(\tau)}|\mathbf{D}\right) = 1$. Thus, instead of employing Eq. 9.4 we can use:

$$p_{G|\mathbf{X}}\left(g|\mathbf{D}\right) \propto p_G\left(g\right) \mathcal{L}_{\mathbf{X}|G}\left(g|\mathbf{D}\right), \tag{9.5}$$

where $p_G\left(g\right)$ is known as prior distribution and can be used to incorporate prior knowledge about the relation of the variables involved. This kind of information may be available from previous

studies or other external sources. Castelo and Siebes (2000) cited in Grzegorczyk et al. (2008) provide several possible ways to define $p_G(g)$ from prior information. In the absence of genuine prior knowledge a uniform distribution may be assumed in order to give the same chance to all candidate graphs. On the other hand, the likelihood $\mathcal{L}_{\mathbf{X}|G}(g|\mathbf{D})$ is known as the marginal (or integrated) likelihood of G , where the parameters $\boldsymbol{\Theta}$ are integrated out, that is,

$$\begin{aligned}\mathcal{L}_{\mathbf{X}|G}(g|\mathbf{D}) &= \int \mathcal{L}_{\mathbf{X}|\boldsymbol{\Theta},G}(\boldsymbol{\theta}, g|\mathbf{D})\, p_{\boldsymbol{\Theta}|\boldsymbol{\Phi},G}(\boldsymbol{\theta}|\boldsymbol{\phi}, g)\, d\boldsymbol{\theta} \\ &= p_{\mathbf{X}|\boldsymbol{\Phi},G}(\mathbf{D}|\boldsymbol{\phi}, g),\end{aligned} \tag{9.6}$$

where $p_{\boldsymbol{\Theta}|\boldsymbol{\Phi},G}$ is the distribution of the unknown parameter vector which is specified conditional on G and from Eq. 9.1, $\mathcal{L}_{\mathbf{X}|\boldsymbol{\Theta},G}(\boldsymbol{\theta}, g|\mathbf{D})$ is given by:

$$\begin{aligned}\mathcal{L}_{\mathbf{X}|\boldsymbol{\Theta},G}(\boldsymbol{\theta}, g|\mathbf{D}) &= \prod_{i=1}^{n}\prod_{\ell=1}^{m} p_{X_i/\mathcal{U}_i,\boldsymbol{\Theta}_i}(X_i = d_{i\ell}|\mathcal{U}_i = \mathbf{d}_{\mathcal{J}_{\mathcal{U}_i}\ell}, \boldsymbol{\theta}_i) \\ &= \prod_{i=1}^{n}\prod_{\ell=1}^{m} p_{X_i/\mathcal{U}_i,\boldsymbol{\Theta}_i}(\mathbf{d}_{\{i,\mathcal{J}_{\mathcal{U}_i}\}\ell}|\boldsymbol{\theta}_i),\end{aligned} \tag{9.7}$$

where $\mathcal{J}_{\mathcal{U}_i} = \{j \in \{1,\ldots,n\}|X_j \in \mathcal{U}_i\}$, the $\boldsymbol{\Theta}_i$'s are the node-specific subvectors, which specify the local conditional distributions in the factorization, $d_{i\ell}$ denotes the ℓ-th realization of the node X_i and $\mathbf{d}_{\mathcal{J}_{\mathcal{U}_i}\ell}$ is a vector containing the ℓ-th realization of the nodes $\mathcal{U}_i$.

An important score, to be mentioned here, is the one defined by Heckerman and Geiger (1995) for GBNs. Here, the marginal likelihood in Eq. 9.3 is defined first in term of the quotient of two more simple functions:

$$\mathcal{L}_{\mathbf{X}|G}(g|\mathbf{D}) = \prod_{i=1}^{n} \frac{p_{\{X_i,\mathcal{U}_i\}|\boldsymbol{\Phi}\{X_i,\mathcal{U}_i\},G_{\{X_i,\mathcal{U}_i\}}}(\mathbf{d}_{\{i,\mathcal{J}_{\mathcal{U}_i}\}.}|\boldsymbol{\phi}_{\{X_i,\mathcal{U}_i\}}, g_{\{X_i,\mathcal{U}_i\}})}{p_{\mathcal{U}_i|\boldsymbol{\Phi}\mathcal{U}_i,G_{\mathcal{U}_i}}(\mathbf{d}_{\mathcal{J}_{\mathcal{U}_i}.}|\boldsymbol{\phi}_{\mathcal{U}_i}, g_{\mathcal{U}_i})}, \tag{9.8}$$

where the numerator and denominator are likelihoods from Gaussian distributions of the set of variables $\{X_i, \mathcal{U}_i\}$ and $\mathcal{U}_i$ respectively, where the associated parameters are integrated out. The

corresponding integrations were made by Heckerman and Geiger (1995) and the results is as follows:

$$p_{\mathcal{S}|\mathbf{\Phi}_\mathcal{S},G_\mathcal{S}}(\mathbf{d}_{\mathcal{J}_\mathcal{S}\cdot}|\boldsymbol{\phi}_\mathcal{S}, g_\mathcal{S}) = \pi^{-\frac{n_s \cdot m}{2}} \cdot \left(\frac{\nu}{\nu+m}\right)^{-\frac{n_s}{2}} \frac{c(n_s, \alpha+m)}{c(n_s,\alpha)} |\mathbf{T_0}|^{\frac{\alpha}{2}} |\mathbf{T_m}(\mathbf{d}_{\mathcal{J}_\mathcal{S}\cdot})|^{-\frac{\alpha+m}{2}}. \tag{9.9}$$

This result corresponds to a function known as the kernel of a multivariate t-distribution with α degrees of freedom. $\mathcal{S}$ is a subset of $\mathbf{X}$, which corresponds to the variables involved in the numerator or denominator. n_s is the cardinality of this subset. $g_\mathcal{S}$ is a DAG representing the conditional correlations between the variables in $\mathcal{S}$. $\mathbf{d}_{\mathcal{J}_\mathcal{S}}$ is a matrix containing all the m realizations of the variables in $\mathcal{S}$. ν, $\mathbf{T_0}$ and $\mathbf{T_m}$ are the hyperparameters associated to the joint probability distribution parameters, where ν represents the degrees of freedom of a Wishart distribution, while $\mathbf{T_0}$ and $\mathbf{T_m}$ are precision matrices of size n_s-by-n_s. The procedure of learning BNs required of initial values for ν and $\mathbf{T_0}$, while $\mathbf{T_m}$ can be obtained as follows:

$$\begin{aligned} \mathbf{T_m}(\mathbf{d}_{\mathcal{J}_\mathcal{S}\cdot}) &= \mathbf{T_0} + \mathbf{S}_m + \frac{\nu \cdot m}{\nu + m} \left(\boldsymbol{\mu}_0 - \overline{\mathbf{d}}_{\mathcal{J}_\mathcal{S}\cdot}\right)\left(\boldsymbol{\mu}_0 - \overline{\mathbf{d}}_{\mathcal{J}_\mathcal{S}\cdot}\right)', \\ \mathbf{S}_m &= \sum_{\ell=1}^{m} \left(\mathbf{d}_{\mathcal{J}_\mathcal{S}\ell} - \overline{\mathbf{d}}_{\mathcal{J}_\mathcal{S}\cdot}\right)\left(\mathbf{d}_{\mathcal{J}_\mathcal{S}\ell} - \overline{\mathbf{d}}_{\mathcal{J}_\mathcal{S}\cdot}\right)' \end{aligned} \tag{9.10}$$

$$\overline{\mathbf{d}}_{\mathcal{J}_\mathcal{S}\cdot} = \frac{1}{m} \sum_{\ell=1}^{m} \mathbf{d}_{\mathcal{J}_\mathcal{S}\ell}. \tag{9.11}$$

On the other hand the constants $c(n_s, \alpha)$, $c(n_s, \alpha+m)$ of the distribution 9.9 can be computed by:

$$c(n_s, \alpha) = \prod_{i=1}^{n_s} \Gamma\left(\frac{\alpha+1-i}{2}\right). \tag{9.12}$$

For more details about the distribution 9.9 and parameters see Heckerman and Geiger (1995).

9.3 Nonparametric Bayesian networks (NPBNs)

For the use of Gaussian Bayesian networks (GBNs), two essential assumptions must be fulfilled for the variables in a study: (1) The distribution of the variables can be approximated by a Gaussian distribution; (2) variables are linear correlated. In practice both assumptions are often violated. A way to overcome this limitation of the GBNs is by using the non-parametric version of them developed by Ickstadt et al. (2011) . This method has been called nonparametric Bayesian networks (NPBNs) and consists of a combination of a nonparametric Bayesian mixture model and of GBNs. The use of such a strategy leads to a classification of the data according to similarities concerning the conditionals dependencies between the variables, thereby a better estimation of the graph structure can be produced. The NPBNs can also be seen as a way to approximate joint probability distributions in cases that the underlying distributions of the variables are no Gaussian.

The DAG in NPBNs is defined as in GBNs, while the joint probability distribution over a vector of random variables, $\mathbf{X} = (X_1, \ldots, X_n)^\top$, is given by:

$$p_{\mathbf{X}|\boldsymbol{\lambda},K,\boldsymbol{\Theta},G}(\mathbf{x}|\boldsymbol{\lambda}, k, \boldsymbol{\theta}, g) = \sum_{h=1}^{k} \lambda_h \, p_{\mathbf{X}|\boldsymbol{\Theta}_h,G}(\mathbf{x}|\boldsymbol{\theta}_h, g). \tag{9.13}$$

The NPBNs depends on new parameters, such as K and $\boldsymbol{\lambda} = \{\lambda_1, \ldots, \lambda_K\}$, which are associated to the nonparametric Bayesian mixture model here employed. This parameter are defined as follows: K is an unfinite number and represent the number of mixture or the number of Gaussians to be considered for the approximation of the joint probability distribution. $\boldsymbol{\lambda}$ is a parameter that gives a weight to each group of data thus providing a likelihood of the same, as it is a measure of probability they must be positive and $\sum_{h=1}^{k} \lambda_h = 1$. Both K and λ are considered to be random variables. On the other hand, $\boldsymbol{\theta}_h = (\mu_h, \sigma_h^2, \mathbf{B}_h)$ is the set of parameters associated to the GBN defined in the h-th mixture component and g is a DAG that represents the conditional statments in a NPBN. As in GBNs, $p_{\mathbf{X}|\boldsymbol{\Theta}_h,G}(\mathbf{x}|\boldsymbol{\theta}_h, g)$ can be obtained by a factorization as the

product of the conditional Gaussian distributions of form showed in Eq. 9.2 imply by g.

The number of components K, the set of parameters $\boldsymbol{\Theta} = (\boldsymbol{\Theta}_1, \ldots, \boldsymbol{\Theta}_K)$ and the mixture weights $\boldsymbol{\lambda}$ are all regarded as unknowns. NPBNs assume the same DAG structure overall mixture components, but let the component to be different in parameters. The NPBN DAG structure can be seen as an average DAG over all components.

Let $\mathbf{D}$ be a data set with m independent realization of the set of random variables $\mathbf{X}$ and g the NPBN structure over that set. For learning NPBNs Eq. 9.13 has been rewritten in Ickstadt et al. (2011) as follows:

$$p_{\mathbf{X}|K,\boldsymbol{\Theta},\mathbf{C},G}(\mathbf{D}|k,\boldsymbol{\theta},\mathbf{c},g) = \prod_{\ell=1}^{m} p_{\mathbf{X}|\boldsymbol{\Theta}_{C_\ell},G}(\mathbf{d}_{\cdot\ell}|\boldsymbol{\theta}_{c_\ell},g) \tag{9.14}$$

where $\mathbf{C} = (C_1, \ldots, C_m)$, represent an allocation vector, that is, which allows the identification of which cluster each observation in D belong. For example, $C_\ell = h$ would mean that the ℓ-th observation in $\mathbf{D}$ belongs to the h-th mixture component. Although this expression does not depend on the weights $\boldsymbol{\lambda}$ as Eq. 9.13, here it is assumed that $p(C_\ell = h) = \lambda_h$. The integration of Eq. 9.14 over $\mathbf{C}$ would lead again to Eq. 9.13 (see Nobile (1994) cited in Ickstadt et al. (2011)). On the other hand, $p_{\mathbf{X}|\boldsymbol{\Theta}_{C_\ell},G}(\mathbf{d}_{\cdot\ell}|\boldsymbol{\theta}_{c_\ell},g)$ is the joint probability distribution of a GBN with parameters $\boldsymbol{\Theta}_{C_\ell} = (\boldsymbol{\mu}_{C_\ell}, \boldsymbol{\sigma}_{C_\ell}, \mathbf{B}_{C_\ell})$ and $\mathbf{d}_{\cdot\ell}$ is a submatrix of $\mathbf{D}$ containing the ℓ-th realization of $\mathbf{X}$.

NPBNs employ as defined in section 9.2.1 and score approach given by the posterior probability distribution of K, $\mathbf{C}$ and G:

$$S(g,\mathbf{c},k;\mathbf{D}) = p_{K,\mathbf{C},G|\mathbf{X}}(k,\mathbf{c},g|\mathbf{D}) \propto p_G(g)p_K(k)p_{\mathbf{C}|K}(\mathbf{c}|k)\mathcal{L}_{\mathbf{X}|K,\mathbf{C},G}(k,\mathbf{c},g|\mathbf{D}) \quad , \tag{9.15}$$

where $p_G(g)$ is the prior distribution of G, $p_K(k)$ the prior of the number of mixture components, $p_{\mathbf{C}|K}(\mathbf{c}|k)$ is a conditional probability of $\mathbf{C}$ given K after the integration of λ parameter, on which $\mathbf{C}$ is also dependent. $\mathcal{L}_{\mathbf{X}|K,\mathbf{C},G}(k,\mathbf{c},g|\mathbf{D})$ is the integrated likelihood, where the parameter

vector $\boldsymbol{\Theta} = (\boldsymbol{\Theta}_1, \ldots, \boldsymbol{\Theta}_K)$ are integrated out (see Equations 9.8 and 9.9). In a practical example, Ickstadt et al. (2011) proposes the use of an uniform distribution over the cardinalities of parent sets for G. For K a Poisson distribution with parameter equal to 1. For λ a symetric Dirichlet distribution with a k-dimensional parameter vector $(\delta_1, \ldots, \delta_k)$ with $\delta_h = 1 \ \forall h$, leading to

$$p(\mathbf{c}|k) = \frac{\Gamma(k)}{\Gamma(r+k)} \prod_{h=1}^{k} \Gamma(m_h + 1) = \frac{k!}{(r+k)!} \prod_{h=1}^{k} m_h!, \tag{9.16}$$

where m_h is the number of observations allocated by $\mathbf{c}$ to component h.

For learning NPBNs, Ickstadt et al. (2011) provides an MCMC algorithm. Upon request, the coding of this algorithm is provided by the authors in the language of Matlab.

9.4 Dynamic Bayesian networks (DBNs)

Here a definition of dynamic Bayesian networks (DBNs) is given. For details or additional material see e.g. Ghahramani (1997) and Grzegorczyk and Husmeier (2009).

Dynamic Bayesian Networks (DBNs) are a generalization of BNs to model temporal correlations between interacting random variables. Different from BNs, in a DBN structure the edges between nodes flow forward in time, that is, an edge pointing from X_i to X_j indicates that the realization of X_j at time point t is conditionally dependent on the realization of X_i at time point $t-1$. Here it is also allowed dependencies on the same variable, e.g., an edge pointing from X_i to X_i, which is known as a feedback loop. In the study of biochemical networks, a feedback loop can be intuitively interpreted as self-regulation or self-inhibition.

Figure 9.3 shows an example of a DBN of order 1 with three variables, $\mathbf{X} = \{X_1, X_2, X_3\}$. The DBN structure is represented by the state-space directed graph g^*, shown in the left panel. In this example, node X_2 has a recurrent feedback loop and nodes X_1 and X_2 act as a regulator of node X_3. On the other hand, the right panel of Figure 9.3 shows the same graph

unfolded in time, where it can be seen that the acyclicity constraint is guaranteed to be satisfied. That means that a DBN structure also satisfies the local directed Markov condition defined in Definition 9.2.1. Thereby, the joint probability distribution over $\mathbf{X}_{(t)}$ can also be here factorized as the product of conditional probability distribution as in BNs (see Eq. 9.1).

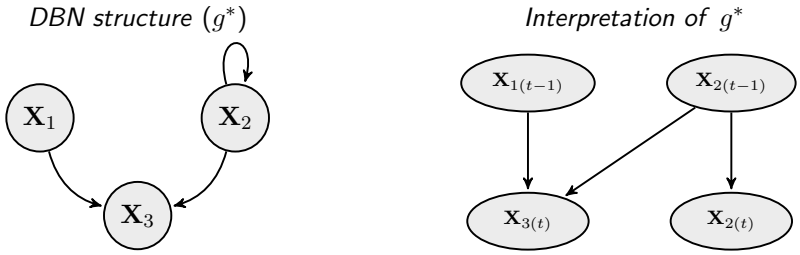


Figure 9.3: A DBN of order 1 with three variables. The left panel shows the state space graph of the DBN and the right panel shows the same graph unfolded in time.

Let $\mathbf{X} = (X_1, \ldots, X_n)^\top$ be a random vector with n variables time correlated, the joint probability distribution given by a DBN structure is given by:

$$p_{\mathbf{X}_{(t)}|\boldsymbol{\Theta},G}(\mathbf{x}_{(t)}|\boldsymbol{\theta}, g) = \prod_{i=1}^{n} p_{X_{i(t)}|\mathcal{U}_{i(t-1)}\boldsymbol{\Theta}_i}(x_{i(t)}|\mathbf{u}_{i(t-1)}, \boldsymbol{\theta}_i) \tag{9.17}$$

where $\boldsymbol{\Theta}$ is the total parameter vector, composed of node-specific subvectors $\boldsymbol{\Theta}_i$, which specify the local conditional distributions in the factorization and $\mathcal{U}_{i(t-1)} = \{X_{j(t-1)} \in \mathbf{X}_{(t-1)} | X_{j(t-1)} \rightarrow X_{i(t-1)}\}$. Note that, although the variables are assumed to be time correlated, the parameters associated to the conditional distributions, $\boldsymbol{\Theta}_i$ are assumed to be time-invariant.

The DBNs are common assumed to be of order 1, but there is no fundamental reason to restrict them to represent temporal correlation of order 1. The concept can be generalized to κ^{th}-order DBNs allowing edges from $\{X_{i(t-\kappa)}, \ldots, X_{i(t-1)}\}$ to $X_{j(t)}$.

As in Bayesian networks (BNs), when all the variables are continuous is commonly assumed Gaussian distributions for the conditional probabilities associated to the joint probability distribution, with the difference that here the vector of means represent a first order autoregressive multivariate model. That is,

Definition 9.4.1 (Gaussian dynamic Bayesian networks)

A Gaussian dynamic Bayesian network is a dynamic Bayesian network where the conditional distributions, $p_{X_i|\mathcal{U}_i,\boldsymbol{\Theta}_i}(x_i|\mathbf{u}_i,\boldsymbol{\theta}_i)$*, are given by normal distributions of the form:*

$$p_{X_{i(t)}|\mathcal{U}_{i(t-1)},\boldsymbol{\Theta}_i}(x_{i(t)}|\mathbf{u}_{i(t-1)},\boldsymbol{\theta}_i) \sim \mathrm{N}(x_{i(t)}|\mu_i + \sum_{\mathcal{J}_i} \beta_{i,j}(x_{j(t-1)} - \mu_j), \sigma_i^2), \tag{9.18}$$

where $\mathcal{J}_i = \{j \in \{1,\ldots,n\}|X_{j(t-1)} \in \mathcal{U}_{i(t-1)}\}$*,* $\boldsymbol{\theta}_i = (\mu_i, \sigma_i^2, \beta_i)$*, the* μ_i *are the unconditional means of* $X_{i(t)}$*,* $\boldsymbol{\beta}_i = (\beta_{i,j})_{j\in\mathcal{J}_{U_{i(t-1)}}}$ *is a set of real coefficients determining the influence of each* $X_{j(t-1)} \in \mathcal{U}_{i(t-1)}$ *on* $X_{i(t)}$ *and* σ_i^2 *is the conditional variance of* $X_{i(t)}$ *given its parents,* $\mathcal{U}_{i(t-1)}$*.*

The learning of DBNs can be performed in the same way as explained in Section 9.2.1. However, here some different computation must be done to introduce the temporal correlation. In (Grzegorczyk and Husmeier, 2009) a way to adapt the Gaussian score of Heckerman and Geiger (1995) for GDBNs is showed, another form is here presented in the next section.

Note that the term "dynamic" means we are modeling a dynamic system, not that the network changes over time. See Dondelinger et al. (2013) for a discussion of DBNs which change their structure over time.

10. Nonparametric mixture of dynamic Bayesian networks (NPMDBNs)

In this section we define a new novel Bayesian method to identify distinct cell subpopulations in a macromolecular sample based on different temporal protein interaction networks. Our approach is an extension of the NPBNs (Ickstadt et al., 2011) in which an Ongaro-Cattaneo mixture model of Gaussian Bayesian networks (GBNs) is proposed to approximate the joint probability distribution of a group of non-Gaussian random variables. Unlike Ickstadt et al. (2011) we employ Gaussian dynamic Bayesian networks (DBNs) as mixing components for modeling temporal relationships between proteins.

The dynamic of the cellular protein interaction networks can be described by a multivariate temporal process of 1st-order Markovian dependence structure. That is, a process in which the expression levels of the involved proteins are temporal correlated, for example, two proteins, $X_{i(t)}$ and $X_{j(t-1)}$, are 1st-order temporal correlated if the levels of $X_{i(t)}$ at a time point t are correlated to the levels of X_j at a time point $t-1$. Let $\{\mathbf{X}_{(t)} = (X_{1(t)}, \ldots, X_{n(t)})^\top, t \in \mathbb{Z}\}$ denote this process. Thus, a GDBN is defined as a dynamic Bayesian network with conditional probability distributions given by normal distributions of the form:

$$p(X_{i(t)}|\mathbf{Pa}_{i(t-1)}, \boldsymbol{\beta}_i, \sigma_i^2) \sim \mathrm{N}(X_{i(t)}|\sum_{\mathcal{J}_i} \beta_{i,j} X_{j(t-1)}, \sigma_i^2), \tag{10.1}$$

where $\mathbf{Pa}_{i(t-1)}$ is the set of protein parents of $X_{i(t)}$, i.e., those proteins $X_{j(t-1)} \in \mathbf{X}_{(t)}$ that directly regulate a target protein $X_{i(t)}$; $\mathcal{J}_i = \{j \in \{1, \ldots, n\}|X_{j(t-1)} \in \mathbf{Pa}_{i(t-1)}\}$;

$\boldsymbol{\beta}_i = (\beta_{i,j})_{j \in \mathcal{J}_i}$ denotes the set of real coefficients determining the influence of each $X_{j(t-1)}$ on $X_{i(t)}$; σ_i^2 is the conditional variance of $X_{i(t)}$ given its parents, $\mathbf{Pa}_{i(t-1)}$.

Different from BNs, a DBN graph is a directed graph that allows dependencies on a same variable, e.g., an edge pointing from $X_{i(t-1)}$ to $X_{i(t)}$, which is known as feedback loop. This in the study of biochemical networks can be intuitively interpreted as self-regulation or self-inhibition.

In an interaction structure, the participating members have a multivariate distribution that can be obtained by the product of conditional probabilities of each target protein on the set of protein parents. Assuming a heterogeneous cell population, in which underlies K cellular subpopulations, we define a mixture of multivariate distribution for the set of proteins associated with a molecular structure of varied cellular origin given by:

$$p(\mathbf{X}_{(t)} = \mathbf{x}_{(t)}(t \in \mathbb{Z}) | K, \boldsymbol{\lambda}, \mathcal{G}, \mathbf{B}, \boldsymbol{\sigma}^2) = \sum_{h=1}^{K} \lambda_h \, p(\mathbf{X}_{(t)} = \mathbf{x}_{(t)}(t \in \mathbb{Z}) | \mathcal{G}_h, \mathbf{B}_{(h)}, \boldsymbol{\sigma}^2_{(h)}). \tag{10.2}$$

K and $\boldsymbol{\lambda}$ are the parameters associated with the mixture model. These are here assumed as random variables, where their distributions can be defined as geometric or Poisson for K and as Dirichlet distribution for $\boldsymbol{\lambda}$. As in Wieczorek et al. (2015), we assume a unique and different correlation structure for each mixture component. $\mathcal{G}$ denotes the set of network structures associated with each component, $\mathbf{B}$ the set of regression coefficients associated with the Gaussian joint probability distribution and $\boldsymbol{\sigma}^2$ the respective variances. For each component h, $\mathbf{B}_{(h)} = \{\boldsymbol{\beta}_{i(h)}\}_{i=1}^{n}$ and $\boldsymbol{\sigma}^2_{(h)} = \{\sigma^2_{i(h)}\}_{i=1}^{n}$.

By Markovian conditional independence condition Pearl (1988) the joint probability distribution associated with a mixture component can be factorized conditional on $\mathcal{G}_h$ as follows:

$$p(\mathbf{X}_{(t)} = \mathbf{x}_{(t)}(t \in \mathbb{Z}) | \mathcal{G}_h, \mathbf{B}_{(h)}, \boldsymbol{\sigma}^2_{(h)}) = \prod_{i=1}^{n} p(X_{i(t)} | \mathbf{Pa}^{(h)}_{i(t-1)}, \boldsymbol{\beta}_{i(h)}, \sigma^2_{i(h)}), \tag{10.3}$$

where the conditional probability distributions are given by Eq. 10.1.

The classification of observations in different components is achieved by using an indicator variable, commonly called allocation vector. This vector allows the assignment of the observations to their respective component. In a sample of a temporal multivariate process $\{\mathbf{X}_{(t)}\}$, $\mathbf{D} = \{\mathtt{D}^{(1)}, \ldots, \mathtt{D}^{(r)}\}$ with r independent observations, let $\mathbf{c} = (c_1, \ldots, c_r)$ be the allocation vector, where $c_s = h$ implies that the s-th sample, $\mathtt{D}^{(s)}$, is coming from the h-th mixture component. In Eq. 10.2, the independence of the component implies that the observation in them are among component also independent, that is, c_s, $s = 1, \ldots, r$ are independent over the K components, with $p(c_s = h) = \lambda_h$ so that $p(\mathbf{c}) = \prod_{h=1}^{K} \lambda_h^{r_h}$, where r_h is the number of samples allocated by $\mathbf{c}$ to component h. Conditioning Eq. 10.2 on $\mathbf{c}$ it becomes:

$$\begin{aligned} p(\mathbf{D}|K, \mathbf{c}, \mathcal{G}, \mathbf{B}, \boldsymbol{\sigma}^2) &= \prod_{s=1}^{r} p(D^{(s)}|\mathcal{G}_{c_s}, \mathbf{B}_{(c_s)}, \boldsymbol{\sigma}^2_{(c_s)}) \\ &= \prod_{s=1}^{r} \prod_{i=1}^{n} p(D^{(s)}_{i(t)}|D^{(s)}_{\mathbf{Pa}_i(t-1)}, \boldsymbol{\beta}_{i(c_s)}, \sigma^2_{i(c_s)}). \end{aligned} \tag{10.4}$$

where $\mathtt{D}^{(s)}$ is an n-by-m_s matrix consisting of m_s time-dependent realizations of $\{\mathbf{X}_{(t)}\}$. $D^{(s)}_{i(t)}$ and $D^{(s)}_{\mathbf{Pa}_i(t-1)}$ denote the $m_s - 1$ realizations of the target variable $X_{i(t)}$ and of its parent set $\mathbf{Pa}_{i(t-1)}$ respectively.

As all the observations within the same component belong to the same cell subpopulation, the protein interaction network and their associated parameters are also the same. Therefore, we rewrite Eq. 10.4 as follows,

$$\begin{aligned} p(\mathbf{D}|K, \mathbf{c}, \mathcal{G}, \mathbf{B}, \boldsymbol{\sigma}^2) &= \prod_{h=1}^{K} p(D^{(h)}|\mathcal{G}_h, \mathbf{B}_{(h)}, \boldsymbol{\sigma}^2_{(h)}) \\ &= \prod_{h=1}^{K} \prod_{i=1}^{n} p(D^{(h)}_{i(t)}|D^{(h)}_{\mathbf{Pa}_i(t-1)}, \boldsymbol{\beta}_{i(h)}, \sigma^2_{i(h)}), \end{aligned} \tag{10.5}$$

where $D^{(h)} = \{D^{(s)} \in \{D^{(1)}, \ldots, D^{(r)}\} : c_s = h\}$.

In this methodology we emphasize the estimation of the number of distinct subpopulations, the allocation vector and the temporal protein interaction networks associated with each component. Thus, to ease computations, we do not consider directly the Gaussian probabilities defined in Eq. 10.1 but its derivative of the integration of parameters that is given by a multivariate t. This integration is well known in Bayesian statistics since models such as linear regression models are obtained in the same way. Following thus the parametrization used in Gelman et al. (2004) we obtain that:

$$p(D^{(h)}_{i(t)}|D^{(h)}_{\mathbf{Pa}_i(t-1)}) = \prod_{S_h} \int p(D^{(s)}_{i(t)}|D^{(s)}_{\mathbf{Pa}_i(t-1)}, \boldsymbol{\beta}_{i(c_s)}, \sigma^2_{i(c_s)}) p(\boldsymbol{\beta}_{i(c_s)}, \sigma^2_{i(c_s)}) d_{\boldsymbol{\beta}_{i(c_s)}} d_{\sigma^2_{i(c_s)}}$$
$$= \left[\frac{b^a \Gamma(a + \frac{(m_s-1)}{2})\sqrt{|V^{(*)}|}}{(2\pi)^{\frac{(m_s-1)}{2}}\Gamma(a)\sqrt{|V_\beta|}}\right]^{r_h} \left[b + \frac{1}{2}\left(\mu_\beta^\top V_\beta^{-1}\mu_\beta + D^{(s)\top}_{i(t)} D^{(s)}_{i(t)} - \mu^{(*)\top} V^{(*)-1}\mu^{(*)}\right)\right]^{-r_h(a+\frac{(m_s-1)}{2})}, \tag{10.6}$$

is a multivariate t with $2a$ degrees of freedom, where $S_h = \{s \in \{1, \ldots, r\} : c_s = h\}$; r_h is the number of samples allocated by $\mathbf{c}$ to component h; μ^* and V^* are given by:

$$\mu^* = \left(V_\beta^{-1} + D^{(s)\top}_{\mathbf{Pa}_i(t-1)} D^{(s)}_{\mathbf{Pa}_i(t-1)}\right)^{-1} \left(V_\beta^{-1}\mu_\beta + D^{(s)\top}_{\mathbf{Pa}_i(t-1)} D^{(s)}_{i(t)}\right) \tag{10.7}$$

and

$$V^* = \left(V^{-1} + D^{(s)\top}_{\mathbf{Pa}_i(t-1)} D^{(s)}_{\mathbf{Pa}_i(t-1)}\right)^{-1}; \tag{10.8}$$

a, b, μ_β, V_β, V^*, μ^* are hyperparameters associated with the prior distribution $p(\boldsymbol{\beta}_{i(c_s)}, \sigma^2_{i(c_s)})$. Thus, the integrated form of Eq. 10.5 is given by:

$$p(\mathbf{D}|K, \mathbf{c}, \mathcal{G}) = \prod_{h=1}^{K} p(D^{(h)}|\mathcal{G}_h) = \prod_{h=1}^{K}\prod_{i=1}^{n} p(D^{(h)}_{i(t)}|D^{(h)}_{\mathbf{Pa}_i(t-1)}). \tag{10.9}$$

The approximation of K, $\mathbf{c}$ and $\mathcal{G}$ is given by maximizing their posterior probability distribution:

$$p(K, \mathbf{c}, \mathcal{G}|\mathbf{D}) \propto p(K)p(\mathbf{c}|K)\prod_{h=1}^{K} p(\mathcal{G}_h)p(D^{(h)}|\mathcal{G}_h). \tag{10.10}$$

For each $\mathcal{G}_h$ we use as prior distribution an uniform distribution over the cardinalities of parent sets. For K a Poisson distribution with parameter equal to 1. $p(\mathbf{c}|K)$ is obtained by integrating $p(\mathbf{c}|k, \lambda)$ with respect to the $\lambda's$, by using a symetric Dirichlet distribution with a k-dimensional parameter vector $(\delta_1, \ldots, \delta_k)$ with $\delta_h = 1\ \forall h$ as prior distribution. Thus, $p(\mathbf{c}|K)$ is a function depending only on K and is given by $\frac{k!}{(r+k)!}\prod_{h=1}^{k} r_h!$.

11. Algorithm for the learning of NPMDBNs

Estimating the number of subpopulations K, the allocation vector $\mathbf{c}$ and the temporal networks $\mathcal{G}_h$, we employ algorithms of Ickstadt et al. (2011) to generate an algorithm that samples observations from the posterior distribution Eq. 10.10. Unlike the algorithm defined in Ickstadt et al. (2011), our algorithm consists of four movements: M1 move, M2 move, ejection move, and absorption move. The **M1** and **M2** moves are two different Metropolis-Hastings structures to re-allocate some observations from one component h_1 to another one h_2. On the other hand, the **ejection** and **absortion** moves are also Metropolis-Hastings structures, where the **ejection move** proposes to increase the number of mixture components by 1, while the **absortion move** decreases the number of mixture components by 1. For each update of the allocation vector, the network structures associated with each component are generated at the same time.

The four moves are selected randomly to generate a better approximation to the true posterior distribution observations. The main algorithm is given by Algorithm 1.

The **M1** and **M2** moves are two different Metropolis-Hastings structures to re-allocate some observations from one component h_1 to another one h_2 (see Algorithms 2 and 3). The difference between the two moves is that in M1 a random number ϱ is generated and each observation in h_1 is left in h_1 with probability ϱ or changed to h_2 with probability $1-\varrho$. While in M2 the number of observations to be changed (m) is randomly selected and then m observations in component h_1 are randomly selected and changed to h_2.

Algorithm 1: Learning NPMDBNs

Input : Data set: $\mathbf{D}$; initial allocation vector: $\mathbf{c}^{(0)}$; initial values of hyperparameters: a, b, μ_β and V_β; iteration number: T; thinning: $N_{Thinning}$; Burn-In: $N_{Burn-In}$; iteration number for learning the network: T_g; initial parent sets: $\{\mathbf{Pa}_{i(t-1)}^{(0)}\}_i = 1^n$; iteration number for the generator of network structures algorithm: T_g; Burn-In for the generator of network structures algorithm: $N_{g-Burn-In}$

. **Output:** K, $\mathbf{c}$ and $\mathcal{G}$
Set $K^{(0)} = \max(\mathbf{c}^{(0)})$
By using $\{\mathbf{Pa}_{i(t-1)}^{(0)}\}_i = 1^n$ the initial network set $\mathcal{G}^{(0)}, \ldots, \mathcal{G}^{(0)}$
Compute μ^* and V^*.
Set the probabilities p_1, p_2, p_3, p_4 for choosing a move.
for $\tau = 1, \ldots, T$ **do**
    if $K^{(\tau-1)} = 1$ **then**
        Select **Ejection Move**
    else
        Generate a random number u from the uniform distribution in the interval $[0, 1]$
        if $u <= p_1$ **then**
            Select **M1 move**
        else if $u <= p_1 + p_2$ **then**
            Select **M2 move**
        else if $u <= p_1 + p_2 + p_3$ **then**
            Select **Ejection Move**
        else
            Select **Absorption move**
        end
    end
end
Estimate K as the average of $K^{(N_{Burn-In}+1)}, \ldots, K^{(T)}$ and assign it to $\bar{K}$
Postprocess the MCMC sample of the allocation vector $\mathbf{c}^{(N_{Burn-In}+1)}, \ldots, \mathbf{c}^{(T)}$ using the mcclust R package. Name this vector $\bar{\mathbf{c}}$
Use the **generator of network structures** to update the network structure set associated with $\bar{\mathbf{c}}$. Name this set $\bar{\mathcal{G}}$
Return $K = \bar{K}$, $\mathbf{c} = \bar{\mathbf{c}}$ and $\mathcal{G} = \bar{\mathcal{G}}$

Algorithm 2: M1 move

Input : Data set: $\mathbf{D}$; allocation vector: $\mathbf{c}^{(\tau-1)}$; hyperparameters values: a, b, μ_β and V_β; set of networks: $\mathcal{G}^{(\tau-1)}$; iteration number for the generator of network structures algorithm: T_g; Burn-In for the generator of network structures algorithm: $N_{g-Burn-In}$.

Output: $K^{(\tau)}$, $\mathbf{c}^{(\tau)}$ and $\mathcal{G}^{(\tau)}$.

1 Randomly select two mixture components h_1 and h_2 among the K available

2 Draw a random number ϱ from a $Beta(\lambda_{h_1}, \lambda_{h_2})$ where λ_{h_1} and λ_{h_2} are the corresponding hyperparameters of the Dirichlet prior on the mixture weights

3 Re-allocating each observation currently belonging to the h_1-th or h_2-th component to component h_1 with probability ϱ or to component h_2 with probability $1-\varrho$ gives the proposed allocation vector $\mathbf{c}^*$

4 Estimate the set of networks associated with each component defined by $\mathbf{c}^*$ by using the **generator of network structures**. Name this set $\mathcal{G}^{(*)}$

5 Compute the ratio $R = \frac{p(|\mathcal{D}_{h_1|\mathbf{c}^{(*)}}|\mathcal{G}^{(*)})\cdot p(\mathcal{D}_{h_2|\mathbf{c}^{(*)}}|\mathcal{G}^{(*)})}{p(\mathcal{D}_{h_1|\mathbf{c}^{(\tau-1)}}|\mathcal{G}^{(\tau-1)})\cdot p(\mathcal{D}_{h_2|\mathbf{c}^{(\tau-1)}}|\mathcal{G}^{(\tau-1)}|)}$ // see details of the formulation in Nobile and Fearnside (2007)

6

7 Compute the acceptance probability $A(\mathbf{c}^{(*)}|\mathbf{c}^{(\tau-1)}) = min\{R, 1\}$.

8 Sample $\mathbf{c}^{(\tau)}$ such that $\mathbf{c}^{(\tau)} = \mathbf{c}^{(*)}$ with probability $A(\mathbf{c}^{(*)}|\mathbf{c}^{(\tau-1)})$ and set $\mathcal{G}^{(\tau)} = \mathcal{G}^{(*)}$; otherwise $\mathbf{c}^{(\tau)} = \mathbf{c}^{(\tau-1)}$ and $\mathcal{G}^{(\tau)} = \mathcal{G}^{(\tau-1)}$

9 Set $K^{(\tau)} = \max(\mathbf{c}^{\tau})$

Algorithm 3: M2 move

Input : Data set: $\mathbf{D}$; allocation vector: $\mathbf{c}^{(\tau-1)}$; hyperparameters values: a, b, μ_β and V_β; set of networks: $\mathcal{G}^{(\tau-1)}$; iteration number for the generator of network structures algorithm: T_g; Burn-In for the generator of network structures algorithm: $N_{g-Burn-In}$.

Output: $K^{(\tau)}$, $\mathbf{c}^{(\tau)}$ and $\mathcal{G}^{(\tau)}$.

1 Select randomly two mixture components h_1 and h_2 among the K available

2 Draw a random number m from a discrete uniform distribution on $1, \ldots, n_{h_1}$ where n_{h_1} is the number of time series realizations allocated to the h_1-th component

3 Select randomly m realizations among the n_{h_1} in component h_1 and move them to component h_2, obtaining thus the proposed allocation vector $\mathbf{c}^*$

4 Estimate the set of networks associated with each component defined by $\mathbf{c}^*$ by using the **generator of network structures**. Name this set $\mathcal{G}^{(*)}$

5 Compute the ratio $R = \frac{p(\mathbf{c}^{(*)}|K)\cdot\prod_{h\in h_1,h_2} p(\mathcal{D}_{h|\mathbf{c}^{(*)}}|\mathcal{G}^{(*)})}{p(\mathbf{c}^{(\tau-1)}|K)\cdot\prod_{h\in h_1,h_2} p(\mathcal{D}_{h|\mathbf{c}^{(\tau-1)}}|\mathcal{G}^{(\tau-1)})} \cdot \frac{n_{h_1}}{n_{h_2}+m} \cdot \frac{n_{h_1}!\cdot n_{h_2}!}{(n_{h_1}-m)!\cdot(n_{h_2}+m)!}$ Compute the acceptance probability $A(\mathbf{c}^{(*)}|\mathbf{c}^{(\tau-1)}) = min\{R, 1\}$

6 Sample $\mathbf{c}^{(\tau)}$ such that $\mathbf{c}^{(\tau)} = \mathbf{c}^{(*)}$ with probability $A(\mathbf{c}^{(*)}|\mathbf{c}^{(\tau-1)})$ and set $\mathcal{G}^{(\tau)} = \mathcal{G}^{(*)}$; otherwise $\mathbf{c}^{(\tau)} = \mathbf{c}^{(\tau-1)}$ and $\mathcal{G}^{(\tau)} = \mathcal{G}^{(\tau-1)}$

7 Set $K^{(\tau)} = \max(\mathbf{c}^{\tau})$

The proposal distributions used in M1 and M2 move are given respectively by

$$\frac{q_{M_1}(\mathbf{c}|\mathbf{c}^{(*)})}{q_{M_1}(\mathbf{c}^{(*)}|\mathbf{c})} = \frac{p(\mathbf{c}|K)}{p(\mathbf{c}^{(*)}|K)}$$

and

$$\frac{q_{M_2}(\mathbf{c}|\mathbf{c}^{(*)})}{q_{M_2}(\mathbf{c}^{(*)}|\mathbf{c})} = \frac{n_{h_1}}{n_{h_2}+m} \cdot \frac{n_{h_1}! \cdot n_{h_2}!}{(n_{h_1}-m)! \cdot (n_{h_2}+m)!}$$

where $p(\mathbf{c}|K)$ $p(\mathbf{c}^{(*)}|K)$ are given by Eq.10.10; n_{h_1} and n_{h_2} are the number of time series realizations allocated in h_1 and h_2 respectively.

In the ejection move (Algorithm 4) some observations of a component h_1 are taken to fill a new component with label $k+1$ with a proposal distribution given by

$$\frac{q_E(\{K, \mathbf{c}\}|\{K^{(*)}, \mathbf{c}^{(*)}\})}{q_E(\{K^{(*)}, \mathbf{c}^{(*)}\}|\{K, \mathbf{c}\})} = \frac{1-p_K^E}{p_K^E} \cdot \frac{[\Gamma(a)]^2}{\Gamma(2a)} \cdot \frac{\Gamma(2a+n_{h_1})}{\Gamma(a+n_{h_1}^{(*)}) \cdot \Gamma(a+n_{(k+1)}^{(*)})}$$

where n_{h_1} is the number of observations in h_1 bevor the ejection move; $n_{h_1}^{(*)}$ and $n_{(k+1)}^{(*)}$ are respectively the number of observations in h_1 and the new componente $k+1$ after the ejection move; p_K^E is the probability to attempt an ejection move when the number total of components is K; a is the parameter corresponding to a $Beta(a,a)$ distribution, which can be selected by numerically solving the following equation:

$$\frac{\Gamma(2a)}{\Gamma(a)} \cdot \frac{\Gamma(a+n_{h_1})}{\Gamma(2a+n_{h_1})} = 0.1.$$

Algorithm 4: Ejection move

Input : Data set: $\mathbf{D}$; allocation vector: $\mathbf{c}^{(\tau-1)}$; hyperparameters values: a, b, μ_β and V_β; set of networks: $\mathcal{G}^{(\tau-1)}$; iteration number for the generator of network structures algorithm: T_g; Burn-In for the generator of network structures algorithm: $N_{g-Burn-In}$.

Output: $K^{(\tau)}$, $\mathbf{c}^{(\tau)}$ and $\mathcal{G}^{(\tau)}$.

1 Set $K^{(\tau-1)} = \max(\mathbf{c}^{(\tau-1)})$

2 Randomly select one of the K mixture components, say h_1, as the ejecting component

3 Select a parameter a of the $Beta(a, a)$ distribution by numerically solving the following equation

4 $\frac{\Gamma(2a)}{\Gamma(a)} \cdot \frac{\Gamma(a+n_{h_1})}{\Gamma(2a+n_{h_1})} = 0.1$ // Recommendation: Use lookup table

5

6 Draw ϱ from a $Beta(a, a)$ distribution

7 Re-allocate each observation currently allocated to component h_1 in the vector $\mathbf{c}$ with probability ϱ to a new component (ejected component) with label $h_2 = K + 1$, obtaining thus the proposed allocation vector $\mathbf{c}^*$

8 Estimate the set of networks associated with each component defined by $\mathbf{c}^*$ by using the **generator of network structures**. Name this set $\mathcal{G}^{(*)}$

9 Compute the ratio
$$R = \frac{p(K^{(*)}) \cdot p(\mathbf{c}^{(*)}|K^{(*)})}{p(K^{(\tau-1)})) \cdot p(\mathbf{c}^{(\tau-1)})|K^{(\tau-1)}))} \cdot \frac{p(\mathcal{D}_{h_1|\mathbf{c}^{(*)}}|\mathcal{G}^{(*)}) \cdot p(\mathcal{D}_{h_2|\mathbf{c}^{(*)}}|\mathcal{G}^{(*)})}{p(\mathcal{D}_{h_1|\mathbf{c}^{(\tau-1)}}|\mathcal{G}^{(\tau-1)})} \cdot \frac{q(\{K^{(\tau-1)}, \mathbf{c}^{(\tau-1)}\}|\{K^{(*)}, \mathbf{c}^{(*)}\})}{q(\{K^{(*)}, \mathbf{c}^{(*)}\}|\{K^{(\tau-1)}, \mathbf{c}^{(\tau-1)}\})}$$

10 Compute the acceptance probability $A(\mathbf{c}^{(*)}|\mathbf{c}^{(\tau-1)}) = min\{R, 1\}$

11 Sample $\mathbf{c}^{(\tau)}$ such that $\mathbf{c}^{(\tau)} = \mathbf{c}^{(*)}$ with probability $A(\mathbf{c}^{(*)}|\mathbf{c}^{(\tau-1)})$ and set $\mathcal{G}^{(\tau)} = \mathcal{G}^{(*)}$; otherwise $\mathbf{c}^{(\tau)} = \mathbf{c}^{(\tau-1)}$ and $\mathcal{G}^{(\tau)} = \mathcal{G}^{(\tau-1)}$

12 Set $K^{(\tau)} = \max(\mathbf{c}^{\tau})$

In the absotion move (Algorithm 5) all observations from a component h_1 are relocated into another component h_2 with a proposal distribution given by:

$$\frac{q_A(\{K, \mathbf{c}\}|\{K^{(*)}, \mathbf{c}^{(*)}\})}{q_A(\{K^{(*)}, \mathbf{c}^{(*)}\}|\{K, \mathbf{c}\})} = \frac{1 - p_K^A}{p_K^A} \cdot \frac{\Gamma(2a)}{[\Gamma(a)]^2} \cdot \frac{\Gamma(a + n_{h_1}) \cdot \Gamma(a + n_{h_2})}{\Gamma(2a + n_{h_1}^{(*)})}$$

where n_{h_1} and n_{h_1} are respectively the number of observations in h_1 and h_2 bevor the absorption move; $n_{h_1}^{(*)} = n_{h_1} + n_{h_2}$ is the number of observations in h_1 after h_1 has absorbed all observations of h_2; p_K^A is the probability to attempt an absorption move when the number total of components is K (p_K^E and p_K^A must be predefined so that $p_K^E + p_K^A = 1$); a is also here the parameter

corresponding to a $Beta(a, a)$ distribution.

Algorithm 5: Absorption move

Input : Data set: $\mathbf{D}$; allocation vector: $\mathbf{c}^{(\tau-1)}$; hyperparameters values: a, b, μ_β and V_β; set of networks: $\mathcal{G}^{(\tau-1)}$; iteration number for the generator of network structures algorithm: T_g; Burn-In for the generator of network structures algorithm: $N_{g-Burn-In}$.

Output: $K^{(\tau)}$, $\mathbf{c}^{(\tau)}$ and $\mathcal{G}^{(\tau)}$.

1 Set $K^{(\tau-1)} = \max(\mathbf{c}^{(\tau-1)})$

2 Randomly select one of the K mixture components, say h_1, as the absorbing component and another component, say h_2 with $h_2 \neq h_1$, as the disappearing component

3 Re-allocate all observations currently allocated to the disappearing component h_2 by $\mathbf{c}$ to component h_1 to obtain the new allocation vector $\mathbf{c}^{(*)}$

4 Delete the empty component h_2 to obtain the new number of components $K^{(*)} = K - 1$

5 Estimate the set of networks associated with each component defined by $\mathbf{c}^*$ by using the **generator of network structures**. Name this set $\mathcal{G}^{(*)}$

6 Compute the ratio

$$R = \frac{p(K^{(*)}) \cdot p(\mathbf{c}^{(*)}|K^{(*)})}{p(K^{(\tau-1)})) \cdot p(\mathbf{c}^{(\tau-1)})|K^{(\tau-1)}))} \cdot \frac{p(\mathcal{D}_{h_1|\mathbf{c}^{(*)}}|\mathcal{G}^{(*)})}{p(\mathcal{D}_{h_1|\mathbf{c}^{(\tau-1)}}|\mathcal{G}^{(\tau-1)}) \cdot p(\mathcal{D}_{h_2|\mathbf{c}^{(\tau-1)}}|\mathcal{G}^{(\tau-1)})} \cdot \frac{q(\{K^{(\tau-1)}, \mathbf{c}^{(\tau-1)}\}|\{K^{(*)}, \mathbf{c}^{(*)}\})}{q(\{K^{(*)}, \mathbf{c}^{(*)}\}|\{K^{(\tau-1)}, \mathbf{c}^{(\tau-1)}\})}$$

7 Compute the acceptance probability $A(\mathbf{c}^{(*)}|\mathbf{c}^{(\tau-1)}) = min\{R, 1\}$

8 Sample $\mathbf{c}^{(\tau)}$ such that $\mathbf{c}^{(\tau)} = \mathbf{c}^{(*)}$ with probability $A(\mathbf{c}^{(*)}|\mathbf{c}^{(\tau-1)})$ and set $\mathcal{G}^{(\tau)} = \mathcal{G}^{(*)}$; otherwise $\mathbf{c}^{(\tau)} = \mathbf{c}^{(\tau-1)}$ and $\mathcal{G}^{(\tau)} = \mathcal{G}^{(\tau-1)}$

9 Set $K^{(\tau)} = \max(\mathbf{c}^{\tau})$

For the generation of network structures for each component, we generated an algorithm for calculating possible edges generated by each protein target through a progression of edges given by subsets of parents generated by the same coefficient of temporal correlation between each protein pair (see Algorithm 6). This guarantees a secure convergence to the real subset of protein parents.

Algorithm 6: Generator of network structures

Input : Allocation vector: $\mathbf{c}$; hyperparameters values: a, b, μ_β and V_β; iteration number for learning the network: T_g, initial parent sets: $\{\mathbf{Pa}_{i(t-1)}^{(0)}\}_i = 1^n$; Burn-In for the generator of network structures algorithm: $N_{g-Burn-In}$

Output: $\mathcal{G}^{(*)} = \{\mathcal{G}_1^{(*)}, \ldots, \mathcal{G}_K^{(*)}\}$

Set $K = \max(\mathbf{c})$

for $\tau_g = 1, \ldots, T_g$ **do**

  for $h = 1, \ldots, K^{(0)}$ *(For each h-th component)* **do**

    // Recommendation: Use parallel computing

    for $i = 1, \ldots, n$ *(For each i-th protein)* **do**

      Compute the pairwise temporal correlations of that protein with the rest.

      Define a set of candidate parents based on correlations different from 0.2 in absolute value.

      Select a random set of proteins from the candidate parent set to define $\mathbf{Pa}_{i(t-1)}^{\tau_g}$.

      Compute the conditional probability associated with the i-th protein, $CDP_i^* = p(D_{i(t)}^{(h)} | D_{\mathbf{Pa}_{i(t-1)}^*}^{(h)})$.

      Compute the ratio

$$R = \frac{p(D_{i(t)}^{(h)} | D_{\mathbf{Pa}_{i(t-1)}^*}^{(h)})}{p(D_{i(t)}^{(h)} | D_{\mathbf{Pa}_{i(t-1)}^{\tau_g - 1}}^{(h)})} \cdot \frac{|\mathbf{Pa}_{i(t-1)}^{(\tau_g - 1)}|}{|\mathbf{Pa}_{i(t-1)}^{(*)}|},$$

      where $|\mathbf{Pa}_{i(t-1)}|$ denotes the cardinality of $\mathbf{Pa}_{i(t-1)}$.

      Compute the acceptance rate $A(\mathbf{Pa}_{i(t-1)}^* | \mathbf{Pa}_{i(t-1)}^{\tau_g - 1}) = min\{R, 1\}$.

      Sample $\mathbf{Pa}_{i(t-1)}^{(\tau_g)}$ such that $\mathbf{Pa}_{i(t-1)}^{(\tau_g)} = \mathbf{Pa}_{i(t-1)}^{(*)}$ with acceptance rate $A(\mathbf{Pa}_{i(t-1)}^* | \mathbf{Pa}_{i(t-1)}^{\tau_g - 1})$; otherwise $\mathbf{Pa}_{i(t-1)}^{(\tau_g)} = \mathbf{Pa}_{i(t-1)}^{(\tau_g - 1)}$.

    end

    With the $\mathbf{Pa}_{i(t-1)}^{(\tau_g)}$ for each protein, define the network $\mathcal{G}_h^{(\tau_g)}$.

  end

  Postprocess the MCMC sample of network structures $\mathcal{G}^{(N_{g-Burn-In}+1)}, \ldots, \mathcal{G}^{(T_g)}$. Name this $\mathcal{G}^{(*)}$

end

For the use of our initial approach values for the parameters associated with the posterior sample distribution must be defined. As initial allocation we can use a vector of 1's, allocating all the samples to one component. The hyperparameters associated to the posterior probability Eq. 10.10 can be initialized as $a = b = 1$, $\mu_\beta = \mathbf{0}$, $V_\beta = \mathtt{I}$ where $\mathtt{I}$ denotes the identity matrix. The initial parent sets as empty sets.

Moves M1, M2, ejection and absortion can be found in Nobile and Fearnside (2007) cited by

? .

12. Methodology of postprocessing for MCMC samples in NPMDBNs

12.1 Postprocessing of allocation vectors

The learning algorithm provides a sample of allocation vectors, $\mathbf{c}^{(1)}, \mathbf{c}^{(2)}, \ldots, \mathbf{c}^{(T)}$ with $\mathbf{c}^{(\tau)} = (c_1^{(\tau)}, \ldots, c_r^{(\tau)})$. Because in MCMC the sampling procedure the labels associated with the components and the number of mixture components can change randomly, here the task of summarizing this sample of allocation vectors with a single vector is complicated. This problem is known as the label switching problem. If two allocation vectors are compared, it is not clear if a particular observation has been assigned to a different component or if the label of the component has been renamed. A way to combine the allocation vectors of each iteration into a single vector is by using the clustering estimation method developed by Fritsch and Ickstadt (2009). This method is based on maximizing posterior probabilities that the observations i and j are in the same cluster. This is achieved by maximizing an index called adjusted *Rand index* that bypasses the label switching problem.

Let $\mathbf{c}^{(1)}, \mathbf{c}^{(2)}, \ldots, \mathbf{c}^{(T)}$ with $\mathbf{c}^{(\tau)} = (c_1^{(\tau)}, \ldots, c_r^{(\tau)})$ be a MCMC sample of allocation vectors. The adjusted Rand index (AR) of estimated and true clustering is given by:

$$AR(\mathbf{c}^*, \mathbf{c}) = \frac{\sum_{i<j} \mathtt{I}_{\{c_i^*=c_j^*\}} \mathtt{I}_{\{c_i=c_j\}} - \sum_{i<j} \mathtt{I}_{\{c_i^*=c_j^*\}} \sum_{i<j} \mathtt{I}_{\{c_i=c_j\}} / \binom{r}{2}}{\frac{1}{2}\left[\sum_{i<j} \mathtt{I}_{\{c_i^*=c_j^*\}} + \sum_{i<j} \mathtt{I}_{\{c_i=c_j\}}\right] - \sum_{i<j} \mathtt{I}_{\{c_i^*=c_j^*\}} \sum_{i<j} \mathtt{I}_{\{c_i=c_j\}} / \binom{r}{2}} \tag{12.1}$$

where $\mathbf{c}^*$ denotes the proposed clustering estimate and $\mathbf{c}$ the unknown true clustering. The matrix containing $\mathtt{I}_{\{c_i^*=c_j^*\}}$ is a known $0-1$ matrix, which is referred to as estimated similarity matrix. The unknown true clustering $\mathbf{c}$ has a similarity matrix containing $\mathtt{I}_{\{c_i=c_j\}}$.

This method proposes then maximize $\mathtt{E}(AR(\mathbf{c}^*,\mathbf{c})|\mathbf{D})$ over $\mathbf{c}^*$, leading to a clustering that has maximum posterior expected adjusted Rand index with the true clustering. This can be approximated from the MCMC sample by (Fritsch and Ickstadt, 2009):

$$\widehat{\mathtt{E}}(AR(\mathbf{c}^*,\mathbf{c})|\mathbf{D}) = \frac{1}{T}\sum_{\tau=1}^{T} AR(\mathbf{c}^*,\mathbf{c}^{(\tau)}) \tag{12.2}$$

whith $AR(\mathbf{c}^*,\mathbf{c}^{(\tau)})$ being computed by Eq 12.1.

An implementation of this clustering estimation method can be found in the mcclust R package (Fritsch, 2012).

12.2 Postprocessing of graph structures

In the study of biochemical networks, the available data sets are often sparse and associated with large noise, which makes the posterior probability of structures to be diffuse (Grzegorczyk et al., 2008). That is, we can find more than one graph structure with high posterior probabilities which possibly possess substantially different sets of edges. Thus a single graph is not representative of the dependence structure among the variables. The need to capture this inference uncertainty leads to the use of a model averaging approach on the group of most likely structures instead of a single one. The NPDBN learning algorithm provides for each identified subpopulation a sample of state-space directed graph $\mathcal{G}^{(1)},\ldots,\mathcal{G}^{(T)}$ stored in the form of connectivity matrices, such as $\mathbf{A}=\{a_{ij}\}_{n\times n}$ where $a_{ij}=1$ for the presence of connection between node X_i and X_j (an edge pointing from X_i to X_j) indicating that the realization of $\{X_{jt}\}$ is conditionally dependent on the realization of $\{X_{it-1}\}$, and $a_{ij}=0$ for the absence of connection between node X_i and X_j. Hence an intuitive model averaging approach of a group of network structures

is based on the edge posterior probability estimates:

$$\widehat{p_{E_{ij}|\mathbf{X}_{(t)}}}(1|\mathbf{D}) = \frac{1}{T}\sum_{\tau=1}^{T} \mathtt{I}_{E_{ij}}(\mathcal{G}^{(\tau)}), \;\; i,j = 1,\ldots,n. \tag{12.3}$$

where τ is the index of the T iteration steps in the MCMC simulation.

The resulting $\widehat{p_{E_{ij}|\mathbf{X}_{(t)}}}(1|\mathbf{D})$ ranges from 0 (i.e., strong evidence for the absence of a connection) to 1 (i.e., strong evidence for a connection between the corresponding nodes).

13. Assessment of the NPMDBNs

13.1 Simulation study

In order to evaluate the performance of the NPMDBNs, we generated synthetic time-series protein expression data from realistic protein-protein interaction networks by using the first-order multivariate vector autoregressive model (MVAR)

$$\mathbf{x_t} = \mathbf{W}\mathbf{x_{t-1}} + \boldsymbol{\varepsilon}_t \tag{13.1}$$

where

- $\mathbf{x_t} = (x_{1t}, x_{2t}, \ldots, x_{nt})^\top$ denotes expressions of n number of proteins at time t;
- $\mathbf{W}$ is a n-by-n time-invariant matrix of coefficients (weights), where the coefficients represent the strength of interaction between all pair of proteins;
- $\boldsymbol{\varepsilon}_t = (\varepsilon_{1t}, \varepsilon_{2t}, \ldots, \varepsilon_{nt})^\top$ is additive Gaussian noise to the protein expression at time t with vector mean $\boldsymbol{\mu} = (\mu_1, \mu_2, \ldots, \mu_n)$ and standard deviation $\boldsymbol{\sigma} = (\sigma_1, \sigma_2, \ldots, \sigma_n)$.

Although there are other models to generate synthetic time-series data even from a realistic biological system as for instance the mitogen-activated protein kinase (MAPK) pathway (Sasagawa et al., 2005) in such model the parameters associated to the network structure can not be easily changed, while by using a MVAR model the interest parameters can be easily varied and controlled. To generate a realistic situation of mixtures of cellular subpopulations, we generate mixtures of observations from two and up to four subpopulations. For this purpose, we

randomly generate in each simulation a structure network that represents the interaction of the proteins for each subpopulation, varying it by at least one edge, either reversing the direction of one of these, eliminating or adding. For the generation of realistic protein interaction networks, we set in the simulation the number of nodes and the proportion of possible connections in the network (network density) based on real protein-protein network structures discussed in Zamir and Bastiaens (2008). Under this information, the function "random.graph.game" in the R package igraph can give us network structures close to these values.

There are some parameters that could have an impact on a decrease in the effectiveness of the model in identifying an adequate stratification of the data and, consequently, in reconstructing the temporal network associated with each component. We vary the levels of parameters such as number of subpopulations, sample proportion per subpopulation, number of proteins, network density and intra-subpopulation variability to identify the effect of these on our algorithm. Table 13.1 contains in the columns the list of the interest parameters and in the rows their respective values used in the simulations. Note that the intra-subpopulation variability is defined through the combination of the variation in protein levels within each subpopulation and the length of the time series. The values 0.1, 0.5 and 0.8 in the intra subpopulation variability represent the fractional deviation fd from the expresion level mean ($\sigma = \mu \cdot fd$), while 20 and 100 represent the length of the time series, 20 for short time series and 100 for long time series. A more detailed definition of these parameters are found in Appendice B.

	Mixture parameters		Network parameters		Intra-subpopulation variability
	No. of subpop.	Sample proportion	Network density	No. of proteins	Noise level & Time series length
No. of subpop.	2 4	2	2	2	2
Sample proportion	1:1	1:1 1:3 1:7	1:1	1:1	1:1
Network density	0.3	0.3	0.3 0.6	0.3	0.3
No. of proteins	5	5	5	3 5 10	5
Noise level & Time series length	0.1 & 100	0.1 & 100	0.1 & 100	0.1 & 100	0.1 & 20 0.5 & 20 0.8 & 20 0.1 & 100 0.5 & 100 0.8 & 100

Table 13.1: List of parameters involved in the simulation study and the employed values. Rows contain the parameter values set as a fixed value or varied (grey) in the simulations. Each combination of parameter values per column corresponds to a different data set. Definition of these parameters can be found in Appendix B.

The algorithm that generates these simulations is given in Algorithm 7. In this algorithm a realization of a subpopulation corresponds to a multivariate time series (MTS). In order to evaluate the stability of our approach, we generated 100 different data sets. Each dataset containing a total of 400 samples from Eq. 13.1 distributed in two or four subpopulations according to the proportion sample defined in the Table 13.1 . Each repetition of the simulation was done in parallel computing to streamline the procedure. The convergence of this algorithm is guaranteed under the run of 100.000 time data where only the last ones were used as observations of the true multivariate model.

Algorithm 7: Simulation of protein expression levels

Input : No. of subpopulations, sample size per subpopulation, number of proteins (np), network density (ND), degree of stochastic variance (fd), time series length (nts)

Output: D // Mixture data of multivariate time series realizations

for *each supopulation* **do**
  Simulate a directed graph $\mathcal{G}$ with np nodes and ND network density // The function ''random.graph.game'' in the R package igraph was here employed
  Transform $\mathcal{G}$ in an adjacency (connectivity) matrix A
  Build the coefficient matrix $\mathbf{W}$ of the MVAR model generating weights randomly from uniform distribution on the intervals [0.5,1.5] and [-1.5,-0.5] to be assigned to all the connection ($a_{ij} = 1$) in the connectivity matrix A
  Generate initial values for n proteins ($\mathbf{x}_0 = (x_{1,0}, \ldots, x_{n,0})^\top$) from a uniform distribution on the interval [0.1,1]
  for *each realisation of the subpopulation* **do**
    for *each time point t* **do**
      // to the total a fraction of points is considered to be used as Burn-In
      Generate the additive Gaussian noise to the protein expression at time t ($\boldsymbol{\varepsilon}_t$) from a normal distribution with mean vector $\mathbf{x}_0$ and standard deviation $\mathbf{sd} = (x_{1,0} * fd, \ldots, x_{n,0} * fd)$
      Sample $\mathbf{x}_t$ from the MVAR model $\mathbf{x}_t = \mathbf{W}\mathbf{x}_{t-1} + \boldsymbol{\varepsilon}_t$
    end
    Remove the first time point realizations (Burn-in) in order to delete influence from initial values
  end
  Make D a set with the mixture of all realizations
  Return D
end

The methodology described here was implemented in Matlab 8.1 (R2013a) along with R functions necessary.

13.2 Comparison analyses with existing methods

Our NPMDBNs method is here compared with two classification alternatives, the classical frequentist method hierarchical agglomerative clustering (Hclust) and the nonparametric Bayesian method NPBNs defined in Section 9.3. The Hclust treat each object as a singleton cluster and then successively agglomerate the closest pair of clusters until all clusters (the whole data set) are into a single cluster. The Hclust is performed in R software using the function

"hclust" in "cluster" package. The Hclust is applied here with two measures of similarity, one for time-uncorrelated data, using the most classic of all, the Euclidean distance between the same point of time and another that includes the temporal correlations, the dynamic time warping (DTW), where the local distance matrix was set to a matrix containing the Euclidean distance between each pair of MTS. The DTW algorithm is performed using the function "dtw" in dtw package. These analyses are denoted in the following chapters as E-Hclust and T-Hclust respectively. On the other hand, the NPBNs were implemented as in Wieczorek et al. (2015) using a specific time point for the implementation of the algorithm. The implementation in Matlab was provided by Wieczorek et al. (2015). For the specification of the NPBNs, hyperparameters associated with the sample distribution must be defined, we set these parameters as recommended by Ickstadt et al. (2011) and Wieczorek et al. (2015). In the analyzes with E-Hclust and NPBNs, three-time points from the time-series protein expression datasets were selected. The methods were applied to each point separately, and we used for the comparison the best result of the three analyzes.

13.3 Estimation of the number of subpopulations in a mixture

Unlike the classical methods of clustering, in an analysis with our NPMDBNs, the estimation of the number of components is made directly by our algorithm (see Section 11). A reasonable estimate of this parameter depends to a large extent on how well the model manages to assimilate the variance of the data in recognition of it. Thus, we evaluated the adequacy of the NPMDBNs for the estimation of this number by analyzing data sets with mixtures of two and four subpopulations for different degrees of variation.

In a comparative analysis with the existing clustering methods NPBNs and Hclust we use a nonparametric measure called average silhouette width (ASW) (Rousseeuw, 1987). The ASW provides a measure of how appropriately the data has been clustered. This measure is given as follows: Let $\mathbf{D} = \{\mathbf{D}^{(1)}, \ldots, \mathbf{D}^{(r)}\}$ be a collection of multivariate time series (MTS), where

$\mathbf{D}^{(\ell)}$ represents the ℓ-th MTS, containing m_ℓ observations of the multivariate temporal process $\mathbf{X}_{(t)} = \{X_{1(t)}, \ldots, X_{n(t)}\}$. Given a particular clustering result $\{\Pi_1, \ldots, \Pi_k\}$, the silhouette width score is calculated as follows:

$$sil(\mathbf{D}^{(\ell)}) = \frac{b_\ell - a_\ell}{\max\{a_\ell, b_\ell\}},$$

where a_ℓ is the average dissimilarity between $\mathbf{D}^{(\ell)}$ and all other MTS within the same cluster,

$$a_\ell = \frac{1}{|\Pi(\mathbf{D}^{(\ell)})|} \sum_{\mathbf{D}^{(j)} \in \Pi(\mathbf{D}^{(\ell)})} d(\mathbf{D}^{(j)}, \mathbf{D}^{(\ell)}),$$

and b_ℓ is the lowest average dissimilarity between $\mathbf{D}^{(\ell)}$ and the MTS in the remaining clusters,

$$b_\ell = \min_{\Pi \neq \Pi(\mathbf{D}^{(\ell)})} \frac{1}{|\Pi|} \sum_{\mathbf{D}^{(j)} \in \Pi} d(\mathbf{D}^{(j)}, \mathbf{D}^{(\ell)}).$$

As dissimilarity measure we employed the Euclidean distance. The $sil(\mathbf{D}^{(\ell)})$ lies between -1 and 1. a_ℓ is a measure of how well $\mathbf{D}^{(\ell)}$ is assigned to its cluster, the smaller the value, the better the assignment. The cluster with the lowest b_ℓ is the next best fit cluster for point $\mathbf{D}^{(\ell)}$. Hence an $sil(\mathbf{D}^{(\ell)})$ close to 1 means that $\mathbf{D}^{(\ell)}$ is appropriately clustered. If $sil(\mathbf{D}^{(\ell)})$ is close to -1 means that it would be more appropriate to cluster $\mathbf{D}^{(\ell)}$ in its neighboring cluster. An $sil(\mathbf{D}^{(\ell)})$ near zero means that $\mathbf{D}^{(\ell)}$ is on the border of two natural clusters. The ASW is finally computed by averaging all $sil(\mathbf{D}^{(\ell)})$ values, $ASW = \sum_{\ell=1}^{r} sil(\mathbf{D}^{(\ell)})$. The use of this measure requires the fixation of the number of clusters in NPBNs and NPMDBNs. Thus, the steps corresponding to the update of the number of components in the corresponding algorithms were eliminated. The cluster number was here set from 2 to 10 clusters.

13.4 Classification of observations to the mixture subclasses

This analysis begins with the evaluation of the effect of parameters associated with the clustering of observations on the precision of our algorithm. Parameters such as the number of subpopulations and sample proportion per subpopulations were here considered. To evaluate the adequacy of our NPMDBNs in classifying observations we use a measure of the percentage of correctly allocated observations (*pco*) defined based on a comparison of the estimated allocation vector with the right vector. The (*pco*) is here given by: $pco = \sum_{h=1}^{K} \frac{1}{r_h} \sum_{i=1}^{r_h} I(c_i^{(h)}) \cdot 100$ where r_h is the number of samples allocated by $\mathbf{c}$ to component h. By comparison with the simulation setting, observations originating from the same true component are considered as allocated correctly. Thus, the indicator function $I(c_i^{(h)})$ is set to 1, while the remaining observations in that component are considered as wrongly allocated, $I(c_i^{(h)}) = 0$.

A comparative analysis with existing clustering methods such as NPBNs and Hclust is here also carried out as a function of the intra-subpopulation variability.

13.5 Reconstruction of temporal networks in a mixture

Our algorithm is evaluated regarding to the parameters networks precision such as the number of nodes, network density, intra-subpopulation variability. We employed the well-known receiver operating characteristic curves (ROC curves) to assess the adequacy of the network in each identified subpopulation. This curve synthesizes how well a directed edge (standing for a causal effect of one protein on another one) is determined by our approach by plotting the correct positive rate against the false positive rate at various threshold settings. The right positive rate and the false positive rate are defined by comparing estimated networks with the exact network structures. A measure of precision can be given by the area under the ROC curve, where an area of 1 represents a perfect estimation, while an area of .5 represents an entirely random, hence worthless estimation.

For the comparative analysis with other methods, the classifications obtained by E-Hclust and T-Hclust were also here used. We carry out independent analyzes in each identified subpopulation applying GDBNs to determine the temporal correlation network structure associated with each subpopulation. We denote here these analyzes as E-Hclust+ and T-Hclust+ respectively. The individual application of the GDBNs was done through the implementation of the algorithm **generator of network structures** described in Section 11. The hyperparameters associated with a GDBN are defined as in our model (see Section 13.6).

A drawback is presented in the evaluation of the adequacy of the networks provided by the NPBNs, since these are undirected acyclic graphs. Here, we compare the estimated networks with the undirected graph associated with the right networks of each subpopulation.

13.6 Implementation of NPMDBN

The learning of a NPMDBN is carried out by the implementation of the algorithm described in Section 11 using Matlab 8.1 (R2013a). For the use of our approach initial values for the parameters associated to the sample posterior distribution must be defined. As initial allocation we can use a vector of 1's (allocating all the samples to one component). The hyperparameters associated to the posterior probability Eq. 10.10 can be initialized as $a = b = 1$, $\mu_\beta = \mathbf{0}$, $V_\beta = \mathtt{I}$ where $\mathtt{I}$ denotes the identity matrix. The initial parent sets as empty sets and the iteration number as $10 \cdot 10^6$ with a thinning of 100 and a burn in of $5 \cdot 10^3$. For the main algorithm and the learning of the network structures the same iteration number was employed.

14. Results

The adequacy of NPMDBNs is here evaluated. The evaluation was carried out following the methodology described in Chapter 13, which has been implemented using the programmes R and Matlab 8.1 (R2013a). To give a more accurate evaluation process, we simulated 100 different cases of mixtures of two and four subpopulations as described in Section 13.1. Boxplots were here employed to summarize the results for the 100 simulated data sets.

14.1 Estimation of the number of subpopulations in a mixture

Figure 14.1 contains a summary of the estimated average number of subpopulations for simulations with mixtures of two (Figure 14.1 (a)) and four (Figure 14.1 (b)) subpopulations for different levels of inter-cellular variability. As this variability increases, an elevation in the estimation error is observed. However, for most of the simulations, the average subpopulation number lied close to two and four. Thus, we show that our approach enables properly to identify the exact number of subpopulations.

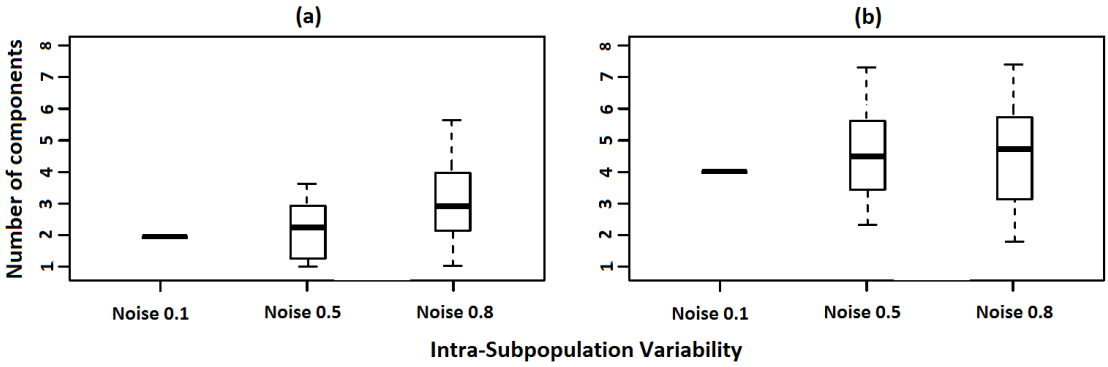


Figure 14.1: Boxplots of the average number of components. The numbers of subpopulations were obtained through an estimated average through the corresponding MCMC sample. The simulations used here correspond to simulations with 2 (a) and 4 (b) subpopulations for different levels of variation in long time series. The level of network density here employed was of 0.3, the number of proteins of 5 and the sample ratio of 1: 1. The boxplot indicates the median (line within the box), the 0.25 and 0.75 quartiles (box), margined by the largest and smallest data points which are still within the interval of 1.5 times the interquartile range from the box (whiskers). The outliers (dots) obtained from pooled values were left out.

A comparative analysis of NPMDBNs with hierarchical agglomerative clustering (Hclust) and nonparametric Bayesian method NPBNs is illustrated in Figure 14.2. The Hclust was applied using two measures of similarity, the Euclidean (E-Hclust) and the dynamic time warping (DTW) (T-Hclust). In the analyzes with E-Hclust and NPBNs, three-time points from the time-series protein expression datasets were selected for this end. They were applied to each point separately, and the best result of the three analyzes was here employed. For this analysis, we use the nonparametric measure average silhouette width (ASW). All methods compared here are similar in efficiency concerning the identification of the exact number of subpopulations in the sample.

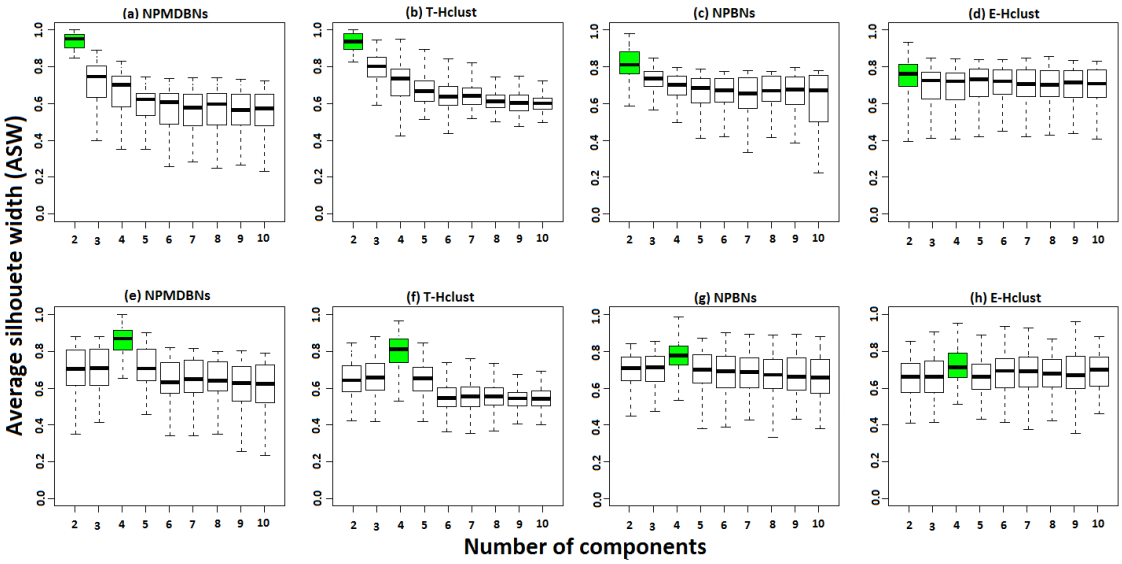


Figure 14.2: Determination of the number of underlying subpopulations. Comparison of NPMDBN with different existing clustering methods. The plots show the results of the average silhouette width (ASW) for the four clustering methods using the different number of components. Plots (a-d) contain the analysis of simulations with a mixture of 2 subpopulations and (e-h) the analysis with 4 subpopulations. In the several simulations, we set the intra-subpopulation variability to 0.5 & 100, network density to 0.3, No. of proteins to 5, and sample proportion to 1:1. ASW values are incomparable between different clustering approaches, but comparable between different parameters of the same method. The largest ASW value indicates the number of distinct subpopulations identified by the respective method. The boxplot indicates the median (line within the box), the 0.25 and 0.75 quartiles (box), margined by the largest and smallest data points which are still within the interval of 1.5 times the interquartile range from the box (whiskers). The outliers (dots) obtained from pooled values were left out.

14.2 Classification of observations to the mixture subclasses

Some parameters such as the number of subpopulations, sample proportion per subpopulations and intra-subpopulation variability can affect the accuracy of our approach to classify

observations. We evaluate our method for different values of them. These parameters depend on how well these are defined in the simulation since an increase in the number of subpopulations is directly related to a low probability of occurrence. Trying to simulate a higher number of different subpopulations, the differences among them decreases proportionally. Thus, the precision in the classification of the samples decreases as the number of subpopulations increases (Figure 14.3 (a)). On the other hand, a slight change in precision due to a variation in sample proportion is observed. However, there does not seem to be a direct relationship of the influence of this parameter on the adequacy of the model (Figure 14.3 (b)).

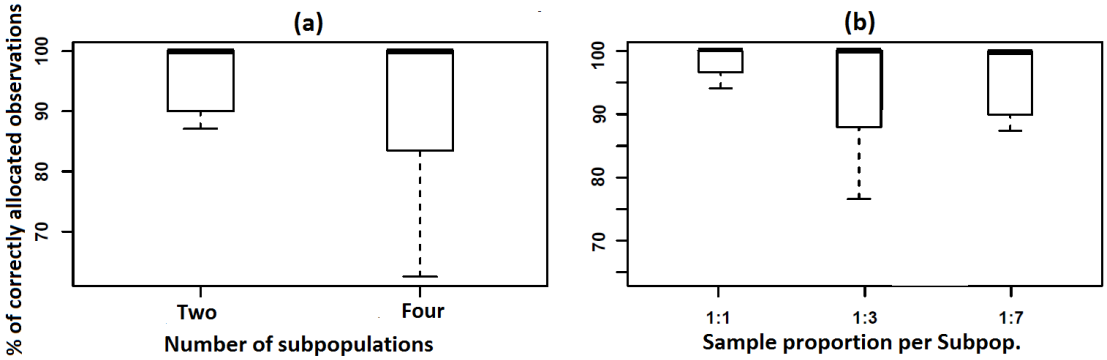


Figure 14.3: Boxplots of the percentages of correctly allocated observations (*pco*) in function of the number of subpopulations (a) and sample proportion per subpopulation (b). The simulations here employed are defined in the columns of Table 13.1 for the respective parameters. The boxplot indicates the median (line within the box), the 0.25 and 0.75 quartiles (box), margined by the largest and smallest data points which are still within the interval of 1.5 times the interquartile range from the box (whiskers). The outliers (dots) obtained from pooled values were left out.

Our method is here evaluated regarding how it manages the classification in the presence of high levels of intra-subpopulation variability (see Figure 14.4). To this end, we make a comparison with the static version NPBNs, and the classical frequentist method Hclust . We use for the Hclust, two similarity measures, the Euclidean (E-Hclust) and the dynamic time

warping (DTW) (T-Hclust). As NPBNs and E-Hclust are not allowed for time correlated data, we implement this method using a specific time point for the implementation of the algorithm. Three-time points from the time-series protein expression datasets were selected for this end. All three results showed very similar results, we use the most accurate of the three. For all methods, the *pco* is directly affected by the term variation and slightly by the length of the size of the time series. A gradual increase is observed for the variation in its different levels. Here, methods like NPBNs and E-Hclust present the most significant error in the classification, this due to the lack of inclusion of temporal correlations. While both methods T-Hclust and NPMDNs are similar in precision. This result shows how the addition of temporal correlations can bring significant improvements in the classification of time series data. However, it is important to note that although NPBNs do not allow the consideration of temporal correlations, this classifies the samples relatively well.

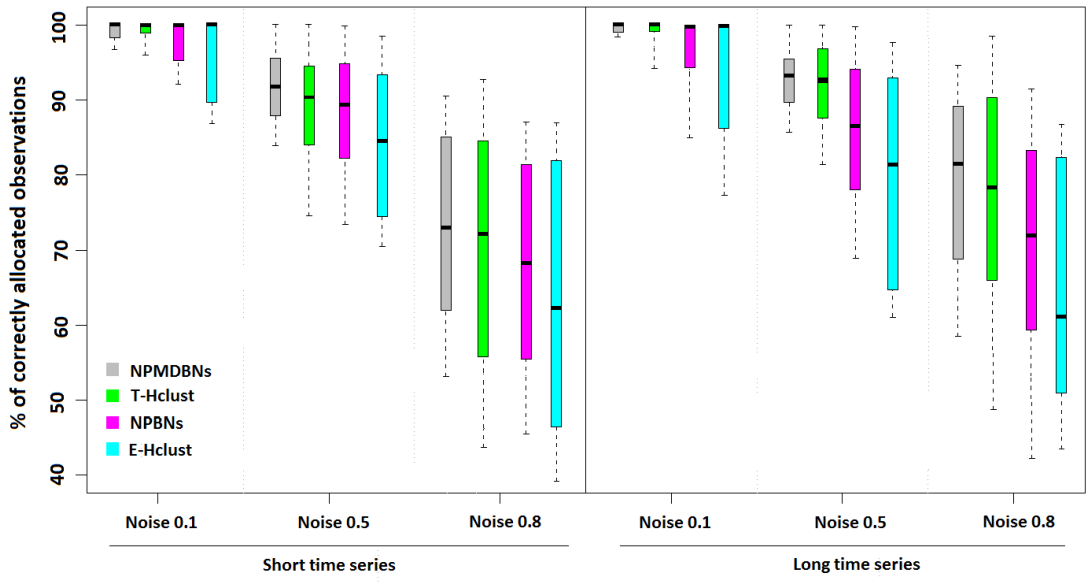


Figure 14.4: Classification of observations. Comparison of analysis between NPMDBN and different clustering methods. This diagram contains the analysis of simulations with a mixture of 2 subpopulations, setting the sample proportion per subpopulation to 1:1, No. of proteins to 5, network density to 0.3. The boxplot indicates the median (line within the box), the 0.25 and 0.75 quartiles (box), margined by the largest and smallest data points which are still within the interval of 1.5 times the interquartile range from the box (whiskers). The outliers (dots) obtained from pooled values were left out.

14.3 Reconstruction of temporal networks in a mixture

This section contemplates the evaluation of the adequacy of the network structures estimated with NPMDBN in terms of temporal conditional dependencies. To address this, we use receiver operating characteristic curves (ROC curves). A ROC curve synthesizes how well a directed edge (standing for a causal effect of one protein on another one) is determined by our approach by plotting the correct positive rate against the false positive rate at various threshold settings. The correct positive rate and the false positive rate are defined by comparing estimated networks with right network structures (networks used in the simulations). A measure of precision can

be given by the area under the ROC curve, where an area of 1 represents a perfect estimation, while an area of .5 represents an entirely random, hence worthless estimation.

Some parameters such as the number of nodes, network density, and intra-subpopulation variability can affect the accuracy of our approach to reconstruct the right temporal networks. We evaluate our method for different values of them. The levels of these terms were varied based on what is found in the literature about protein networks (Zamir and Bastiaens, 2008) (see more details in Table 13.1). Figure 14.5 presents the results of these analyses. While the number of nodes is not a direct problem (Figure 14.5 (b)), the number of associated edges (Figure 14.5 (a)) can result in an increase in the mismatch of the model. Typically, an increase in the level of density leads to a requirement for more programming to contemplate the number of networks necessary for this approach. We use a methodology that considers a large number of networks for the procedure. Despite our efforts to maintain a constant precision, the network density affects the adequacy of the algorithm (Figure 14.5 (a)) slightly. On the other hand, in the intra-subpopulation variability evaluation (Figure 14.5 (c)) we found that the variability degrades the algorithm in terms of the amount of error increase.

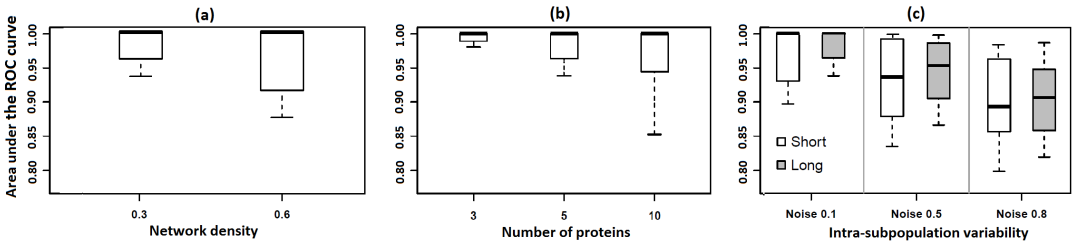


Figure 14.5: Boxplots of the area under the ROC curve (AUROC) in function of network density (a), number of proteins (b) and intra-subpopulation variability (c). The simulations here employed are defined in the columns of Table 13.1 for the respective parameters. Boxplot is the description of the variability in the procedure for network reconstruction in terms of the area under the ROC curve (AUROC). The AUROC is used to compare how good the method can reconstruct the network structure of different species. An area of 1 represents a perfect estimation, while an area of .5 represents a worthless estimation. The boxplot indicates the median (line within the box), the 0.25 and 0.75 quartiles (box), margined by the largest and smallest data points which are still within the interval of 1.5 times the interquartile range from the box (whiskers). The outliers (dots) obtained from pooled values were left out.

In Figure 14.4 it was observed that the T-Hclust analysis showed similar results to our approximation. Therefore, this classification result was used in conjunction with the individual application of GDBNs for the reconstruction of the temporal networks in each subclass. We call here this methodology T-Hclust+. We compare this method with our approach, as well as with the NPBNs that do not include temporal correlations and with the GDBNs as a single model for the mixed data set (i.e., without a previous stratification of the data). Figure 14.6 contains the results of this comparative analysis for the 100 different simulated data sets. These results show that the NPMDBNs and the T-Hclust+ are quite similar in precision. However, significant differences are observed when comparing these results with those obtained by the GDBNs without a previous classification of the observations and the NPBNs without the inclusion of temporal correlations. In general, all methods that include identification of subpopulations lead to nearer networks to the right networks. The use of GDBNs without a previous stratification of the data

point to misspecified interaction models that do not resemble any of the true networks. On the other hand, the lack of adjustment presented in NPBNs is mainly due to the non-incorporation of associated edges to the same node known as a feedback loop. In this case, NPBNs tend to seek to compensate for this variation in another edge, thus moving away from the estimated structure of the correct structure. This is what leads to a decrease in accuracy.

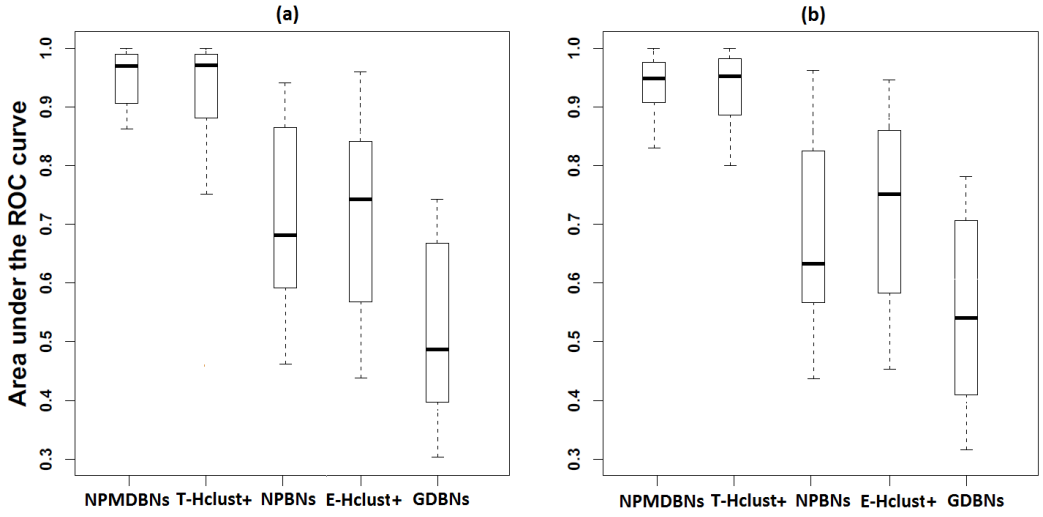


Figure 14.6: Reconstruction of temporal correlation networks by using NPMDBNs. Comparative analysis of NPMDBNs with T-Hclust+, NPBNs, E-Hclust+ and GDBNs. Simulations with a mixture of 2 subpopulations were used in this analysis, setting the sample proportion per subpopulation to 1:1, No. of proteins to 5, network density to 0.3, noise level to 0.5 and long time series (size of 100). Boxplots are represented in terms of the area under the ROC curve (AUROC). Diagram (a) contains the AUROC corresponding to the comparison of the estimated network with the true network corresponding to the subpopulation 1. Diagram (b) the same but for the subpopulation 2. The AUROC is used to compare how good the method can reconstruct the network structure of different species. An area of 1 represents a perfect estimation, while an area of .5 represents a worthless estimation. The boxplot indicates the median (line within the box), the 0.25 and 0.75 quartiles (box), margined by the largest and smallest data points which are still within the interval of 1.5 times the interquartile range from the box (whiskers). The outliers (dots) obtained from pooled values were left out.

15. Conclusions

In this thesis, a novel Bayesian method called the nonparametric mixture of dynamic Bayesian networks (NPMDBNs) was developed. This method enables the reconstruction of temporal protein interaction networks in heterogeneous cell populations with a high degree of precision by identifying cell subpopulations in the sample. Thus, this method allows adapting the high resolution of the information coming from new protein quantification tools as well as the cell to cell variability.

The adequacy of this method is reflected in its comparison with methods that do not include temporal correlations, such as NPBNs and E-Hclust, and methods without unraveling of observations such as GDBNs. We found the NPBNs to be similar in precision as long as they are associated with non-temporal effect networks in their optimization. The lack of adjustment presented in NPBNs is mainly due to the non-incorporation of associated edges to the same node known as a feedback loop. In this case, NPBNs tend to seek to compensate for this variation in another side, thus moving away from the estimated structure of the right structure. However, in general, NPBNs are suitable for both static data and dynamic data. On the other hand, in comparison with the T-Hclust+ analysis, our approach did not show a significant improvement in precision. Thus, both methods can be equally employed for similar purposes. The advantage of our algorithm is that in a single step the observations can be classified into cellular subpopulations, and the temporal interaction networks associated with the subclasses can be reconstructed. While the T-Hclust+ is reached in three steps: (1) the optimal number of subpopulations must be identified by for example using the ASW measure for different number of

clusters, (2) the Hclust method is applied for classification of observations, and (3) GDBNs are employed for the reconstruction of temporal networks in each identified cluster. Furthermore, our approach, as a Bayesian method, enables the incorporation of relevant prior information and assumptions that can give an improvement to the results. A disadvantage concerning T-Hclust+ is the complexity of the computation as well as the time required to achieve convergence. The application of our algorithm to a dataset similar to the one here simulated may take an average of 20 hours, while with the T-Hclust+ no more than 3 hours. Another important result to highlight is the one related to the comparison of GDBNs with NPMDBNs, NPBNs and the T-Hclust+. In general, all methods that used a previous identification of subpopulations lead to nearer networks to the right networks, while the use of GDBNs without a previous stratification of the data point to a misspecified interaction model.

The convergence of our algorithm is guaranteed by the convergence of the methods involved in their development. Furthermore, the success of this method in classifying observations in their respective subpopulations can also be considered here as proof of the respective convergence.

The most relevant part of this work led to the conclusion that the inclusion of temporal information is significant for a realistic reconstruction of protein-protein interaction networks.

The methodology here proposed gives an advance in the classification of multivariate time series as well as in the reconstruction of temporal interaction networks in heterogeneous data. For computational reasons, a parameterization of the conditional distributions reflecting the causal relationship between proteins may not be suitable here. Therefore, the associated parameters are here destined not to be considered. Typically, in practice, these parameters are unknown but can be approximated through Bayesian linear regressions once the network structure is known.

The method here presented can reveal how proteins in quantitative terms are related to each other by using a Gaussian probability model. However, it could be of interest to extend our methodology to understand the protein-protein interactions in terms of activation and inhibition of each protein of the molecular structure by using a binomial probability model.

Mathematically we could also redefine our model in quantum terms to facilitate the learning of protein interaction.

16. References

1. G. Ball, R. Parton, R. Hamilton, and I. Davis. A cell biologist's guide to high resolution imaging. *Methods Enzymol*, 504:29–55, 2012. doi: 10.1016/B978-0-12-391857-4.00002-1.

2. R. Castelo and A. Siebes. Priors on network structures. biasing the search for Bayesian networks. *International Journal of Approximate Reasoning*, 24(1):39 – 57, 2000.

3. T. Chen, H. He, and G. Church. Modeling gene expression with differential equations. *Pacific Symposium Biocomputing*, 4:29–40, 1999.

4. D. M. Chickering. Learning equivalence classes of Bayesian-network structures. *J. Mach. Learn. Res.*, 2:445–498, 2002. ISSN 1532-4435.

5. R. Cowell, P. Dawid, S. Lauritzen, and D. Spiegelhalter. *Probabilistic Networks and Expert Systems: Exact Computational Methods for Bayesian Networks*. Statistics for Engineering and Information Science Series. Springer, 1999. ISBN 0387987673.

6. M. Dehmer and F. Emmert-Streib. *Quantitative Graph Theory: Mathematical Foundations and Applications*. Discrete Mathematics and Its Applications. CRC Press, 2014. ISBN 9781466584518.

7. N. Dojer, A. Gambin, A. Mizera, B. Wilczyński, and J. Tiuryn. Applying dynamic bayesian networks to perturbed gene expression data. *BMC Bioinformatics*, 7(1):249, May 2006. ISSN 1471-2105. doi: 10.1186/1471-2105-7-249. URL https://doi.org/10.1186/ 1471-2105-7-249.

8. F. Dondelinger, S. L'ebre, and D. Husmeier. Non-homogeneous dynamic Bayesian networks with Bayesian regularization for inferring gene regulatory networks with gradually time-varying structure. *Machine Learning*, 90(2):191–230, 2013.

9. N. Friedman, M. Goldszmidt, and A. Wyner. Data analysis with bayesian networks: A bootstrap approach. *Proc Fifteenth Conf on Uncertainty in Artificial Intelligence (UAI)*, 1999.

10. A. Fritsch and K. Ickstadt. Improved criteria for clustering based on the posterior similarity matrix. *Bayesian Analysis*, 4:367–392, 2009. doi: 10.1214/09-BA414.

11. M. Ganapathiraju, M. Thahir, A. Handen, S. Sarkar, R. Sweet, V. Nimgaonkar, C. Loscher, E. Bauer, and S. Chaparala. Schizophrenia interactome with 504 novel protein-protein interactions. *NPJ Schizophrenia*, 2:16012, 2016. doi: 10.1038/npjschz.2016.12.

12. A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin. *Bayesian Data Analysis*. Chapman and Hall/CRC, Boca Raton, FL., 2nd ed. edition, 2004.

13. Z. Ghahramani. Learning dynamic Bayesian networks. *Lecture Notes In Computer Science*, 1387:168–197, 1997.

14. R. Gould. *Graph theory*. Benjamin/Cummings Pub. Co., 1988. ISBN 0805360301.

15. M. Grzegorczyk. An introduction to gaussian Bayesian networks. In Q. Yan, editor, Systems Biology in Drug Discovery and Development: Methods and Protocols, volume 662 of *Methods in Molecular Biology*, pages 121–147. Humana Press, Germany, 2010. ISBN 9781607617990.

16. M. Grzegorczyk and D. Husmeier. Non-stationary continuous dynamic Bayesian networks. In Y. Bengio, D. Schuurmans, J. Lafferty, C. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems* 22, pages 682–690. 2009.

17. M. Grzegorczyk, D. Husmeier, and A. Werhli. Reverse engineering gene regulatory networks with various machine learning methods. In F. Emmert-Streib and M. Dehmer,

editors, *Analysis of Microarray Data: A Network-Based Approach*, pages 101–142. Wiley-VCH, Weinheim, Germany, 2008.

18. B. M. Gumbiner. Cell adhesion: the molecular basis of tissue architecture and morphogenesis. Cell, 84(3):345–357, 1996.

19. J. Harizanova, Y. Fermin, R. S. Malik-Sheriff, J. Wieczorek, K. Ickstadt, H. E. Grecco, and E. Zamir. Highly multiplexed imaging uncovers changes in compositional noise within assembling focal adhesions. PLoS ONE, 11(8), 2016. doi: 10.1371/journal.pone.0160591.

20. J. Hasenauer, C. Hasenauer, T. Hucho, and F. J. Theis. ODE constrained mixture modelling: a method for unraveling subpopulation structures and dynamics. PLoS Comput Biol, 10(7): e1003686, Jul 2014. doi: 10.1371/journal.pcbi.1003686.

21. D. Heckerman and D. Geiger. Learning Bayesian networks: A unification for discrete and gaussian domains. In Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence, UAI'95, pages 274–284, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc. ISBN 1-55860-385-9.

22. R. O. Hynes and A. D. Lander. Contact and adhesive specificities in the associations, migrations, and targeting of cells and axons. Cell, 68(2):303–322, 1992.

23. K. Ickstadt, B. Bornkamp, M. Grzegorczyk, J. Wieczorek, M. R. Sheriff, H. E. Grecco, and E. Zamir. Nonparametric Bayesian networks (with discussion). In J. M. Bernardo, M. J. Bayarri, J. O. Berger, A. P. Dawid, D. Heckerman, A. F. M. Smith, and M. West, editors, Bayesian Statistics 9. Oxford University Press, Oxford, United Kingdom, 2011. doi: 10.1093/acprof:oso/9780199694587.003.0010.

24. S. Imoto, T. Goto, and S. Miyano. Estimation of genetic networks and functional structures between genes by using bayesian networks and nonparametric regression. Pacific Symposium on Biocomputing, 7:175–186, 2002.

25. S. Kauffman. Metabolic stability and epigenesis in randomly constructed genetic nets. J

Theor Biol, 22:437–467, 2009. doi: 10.1016/0022-5193(69)90015-0.

26. S. Y. Kim, S. Imoto, and S. Miyano. Inferring gene networks from time series microarray data using dynamic bayesian networks. Briefings in Bioinformatics, 4(3):228-235, 2003. doi: 10.1093/bib/4.3.228. URL +http://dx.doi.org/10.1093/bib/4.3.228.

27. T. Koski and J. Noble. Bayesian Networks: An Introduction. Wiley Series in Probability and Statistics. Wiley, 2009. ISBN 9780470684030. URL http://books.google.de/books?id= 0zuM_zQ72hcC.

28. S. Liang, S. Fuhrman, and R. Somogyi. Reveal, a general reverse engineering algorithm for inference of genetic network architectures. Pacific Symposium Biocomputing, 3:18–29, 1998.

29. A. Liaw and M. Wiener. Classification and regression by randomforest. R News, 2:18–22, 2002.

30. MATLAB. version 7.10.0 (R2010a). The MathWorks Inc., Natick, Massachusetts, 2010.

31. B. Monya. Cellular imaging: Taking a long, hard look. Nature, 466 (26):1137–1140, 2010. doi: 10.1038/4661137a.

32. A. Nobile. Bayesian analysis of finite mixture distributions. Ph.D. dissertation, Department of Statistics, Carnegie Mellon University, Pittsburgh, September 1994. Available at http://www.stats.gla.ac.uk/∼agostino.

33. J. Pearl. Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. II. Series. Morgan Kaufmann, 1988. ISBN 0387987673.

34. Y. Qi, H. Dhiman, N. Bhola, I. Budyak, S. Kar, D. Man, A. Dutta, K. Tirupula, B. Carr, J. Grandis, Z. Bar-Joseph, and J. Klein-Seetharaman. Systematic prediction of human membrane receptor interactions. Proteomics, 9 (23):5243–55, 2009. doi: 10.1002/pmic.200900259.

35. R Core Team. R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria, 2013. URL http://www.R-project.org/.

36. P. J. Rousseeuw. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. Journal of Computational and Applied Mathematics, 20:53 – 65, 1987. ISSN 0377-0427. doi: 10.1016/0377-0427(87)90125-7. URL http://www.sciencedirect.com/science/article/pii/0377042787901257.

37. S. Sasagawa, Y.-i. Ozaki, K. Fujita, and S. Kuroda. Prediction and validation of the distinct dynamics of transient and sustained Erk activation. Nat Cell Biol, 7(4):365–73, Apr 2005. doi: 10.1038/ncb1233.

38. T. Verma and J. Pearl. Uncertainty in artificial intelligence 6. chapter Equivalence and synthesis of causal models, pages 225 – 268. Elsevier Science Publishers, Cambridge, MA, 1991.

39. J. Wieczorek, R. S. Malik-Sheriff, Y. Fermin, H. E. Grecco, E. Zamir, and K. Ickstadt. Uncovering distinct protein-network topologies in heterogeneous cell populations. BMC Systems Biology, 9:24, 2015. doi: 10.1186/s12918-015-0170-2. URL http://dx.doi.org/10.1186/s12918-015-0170-2.

40. E. Zamir and P. I. Bastiaens. Reverse engineering intracellular biochemical networks. Nature chemical biology, 4(11):643–647, Nov. 2008. ISSN 1552-4469. doi: 10.1038/nchembio1108-643. URL http://dx.doi.org/10.1038/nchembio1108-643.

41. E. Zamir and B. Geiger. Molecular complexity and dynamics of cell–matrix adhesions. Journal of Cell Science, 114 (20):3583–3590, 2001.

Part III

Supplementary material

A. Details of potential predictor proteins

Table A.1: Input protein combinations to explain α-actinin

No. of Inputs	Input Proteins	Determination Coeff.
6	Hic-5, F-actin, FAK, Zyxin, FAK pY397, Pax pY118	0.434
6	Vinculin, Hic-5, F-actin, VASP, FAK, FAK pY397	0.434
4	Vinculin, Hic-5, F-actin, Pax pY118	0.404
4	Hic-5, F-actin, FAK, FAK pY397	0.399
3	F-actin, VASP, Zyxin	0.384
4	Vinculin, F-actin, Zyxin, Pax pY118	0.375
3	Vinculin, F-actin, FAK	0.341
2	F-actin, FAK	0.310
6	Vinculin, Paxillin, VASP, FAK, Zyxin, FAK pY397	0.237
4	Vinculin, Paxillin, Hic-5, VASP	0.236
5	Hic-5, VASP, Zyxin, FAK pY397, Pax pY118	0.233
3	Paxillin, VASP, Pax pY118	0.200
3	Vinculin, VASP, FAK	0.193
3	Vinculin, Hic-5, Pax pY118	0.187
3	Paxillin, VASP, FAK pY397	0.184
3	Vinculin, Hic-5, FAK	0.177
2	Paxillin, VASP	0.175
3	Vinculin, Zyxin, FAK pY397	0.126
2	Paxillin, FAK	0.113

Table A.2: Input protein combinations to explain F-actin

No. of Inputs	Input Proteins	Determination Coeff.
4	Paxillin, alpha-actinin, Zyxin, Pax pY118	0.331
3	Vinculin, Hic-5, alpha-actinin	0.295
4	Hic-5, alpha-actinin, FAK, FAK pY397	0.293
3	alpha-actinin, FAK, FAK pY397	0.287
2	alpha-actinin, Zyxin	0.285
3	alpha-actinin, FAK pY397, Pax pY118	0.274
2	alpha-actinin, FAK	0.270
5	Vinculin, Paxillin, Hic-5, VASP, FAK pY397	0.150
3	Paxillin, Zyxin, FAK pY397	0.113
3	Paxillin, FAK, FAK pY397	0.111
3	Paxillin, VASP, Pax pY118	0.104
4	Hic-5, FAK, Zyxin, FAK pY397	0.099
2	Hic-5, VASP	0.060
2	Zyxin, FAK pY397	0.057

Table A.3: Input protein combinations to explain FAK

No. of Inputs	Input Proteins	Determination Coeff.
6	Vinculin, Hic-5, F-actin, Zyxin, FAK pY397, Pax pY118	0.666
6	Vinculin, Paxillin, F-actin, alpha-actinin, Zyxin, Pax pY118	0.659
5	Vinculin, alpha-actinin, Zyxin, FAK pY397, Pax pY118	0.656
6	Vinculin, Hic-5, F-actin, alpha-actinin, VASP, Zyxin	0.641
5	Vinculin, alpha-actinin, VASP, Zyxin, Pax pY118	0.636
5	Paxillin, alpha-actinin, VASP, Zyxin, Pax pY118	0.629
4	F-actin, Zyxin, FAK pY397, Pax pY118	0.612
3	Vinculin, Paxillin, Pax pY118	0.610
3	Vinculin, Paxillin, alpha-actinin	0.604
3	alpha-actinin, Zyxin, Pax pY118	0.580
4	Paxillin, F-actin, VASP, Pax pY118	0.577
3	Paxillin, Hic-5, alpha-actinin	0.573
3	F-actin, alpha-actinin, Zyxin	0.564
3	Vinculin, VASP, FAK pY397	0.550
4	Vinculin, F-actin, FAK pY397, Pax pY118	0.550
3	Vinculin, Hic-5, Pax pY118	0.542
2	Vinculin, alpha-actinin	0.463
3	F-actin, FAK pY397, Pax pY118	0.428
3	F-actin, alpha-actinin, FAK pY397	0.394
2	alpha-actinin, FAK pY397	0.377
1	FAK pY397	0.355
3	F-actin, alpha-actinin, VASP	0.338
2	F-actin, alpha-actinin	0.087

Table A.4: Input protein combinations to explain FAK pY397

No. of Inputs	Input Proteins	Determination Coeff.
7	Paxillin, Hic-5, F-actin, VASP, FAK, Zyxin, Pax pY118	0.497
6	Paxillin, Hic-5, alpha-actinin, FAK, Zyxin, Pax pY118	0.487
5	Vinculin, Paxillin, Hic-5, FAK, Pax pY118	0.483
5	Vinculin, F-actin, alpha-actinin, FAK, Pax pY118	0.479
3	Hic-5, FAK, Pax pY118	0.464
6	Paxillin, Hic-5, F-actin, VASP, Zyxin, Pax pY118	0.464
4	F-actin, VASP, Zyxin, Pax pY118	0.454
6	Paxillin, Hic-5, F-actin, alpha-actinin, VASP, FAK	0.451
3	VASP, Zyxin, Pax pY118	0.445
4	Paxillin, F-actin, VASP, Pax pY118	0.439
4	Paxillin, Hic-5, VASP, Pax pY118	0.436
3	Vinculin, VASP, Zyxin	0.435
2	Zyxin, Pax pY118	0.434
4	F-actin, alpha-actinin, VASP, FAK	0.423
5	Paxillin, Hic-5, F-actin, alpha-actinin, Zyxin	0.418
3	Paxillin, F-actin, Pax pY118	0.406
2	VASP, Zyxin	0.404
2	VASP, Pax pY118	0.398
4	Paxillin, F-actin, alpha-actinin, VASP	0.398
2	Paxillin, Zyxin	0.397
2	Vinculin, VASP	0.393
3	Vinculin, Hic-5, alpha-actinin	0.392
3	F-actin, alpha-actinin, FAK	0.372
4	Hic-5, F-actin, alpha-actinin, VASP	0.366
3	Hic-5, alpha-actinin, VASP	0.361
2	Paxillin, Hic-5	0.355
3	F-actin, alpha-actinin, Pax pY118	0.326

Table A.5: Input protein combinations to explain Hic-5

No. of Inputs	Input Proteins	Determination Coeff.
5	Vinculin, Paxillin, VASP, Zyxin, Pax pY118	0.600
4	Paxillin, VASP, FAK, Pax pY118	0.582
3	Vinculin, Paxillin, alpha-actinin	0.574
3	Vinculin, Paxillin, FAK	0.572
4	Vinculin, alpha-actinin, Zyxin, Pax pY118	0.571
4	alpha-actinin, FAK, Zyxin, FAK pY397	0.567
4	alpha-actinin, VASP, FAK, Zyxin	0.567
4	Paxillin, F-actin, VASP, FAK pY397	0.566
5	Vinculin, F-actin, FAK, Zyxin, FAK pY397	0.562
3	Paxillin, VASP, FAK pY397	0.560
4	alpha-actinin, Zyxin, FAK pY397, Pax pY118	0.559
4	F-actin, VASP, FAK, Zyxin	0.555
3	Paxillin, FAK, FAK pY397	0.552
4	Paxillin, F-actin, FAK pY397, Pax pY118	0.551
2	Vinculin, Paxillin	0.538
3	F-actin, FAK, Zyxin	0.531
4	Vinculin, alpha-actinin, FAK pY397, Pax pY118	0.497
4	alpha-actinin, VASP, FAK pY397, Pax pY118	0.483
4	alpha-actinin, VASP, FAK, FAK pY397	0.481
4	Vinculin, F-actin, alpha-actinin, FAK pY397	0.470
2	Vinculin, VASP	0.451
3	Vinculin, FAK, FAK pY397	0.442
3	Vinculin, F-actin, FAK	0.435
3	alpha-actinin, FAK, FAK pY397	0.418
3	F-actin, FAK pY397, Pax pY118	0.381

Table A.6: Input protein combinations to explain paxillin

No. of Inputs	Input Proteins	Determination Coeff.
6	Vinculin, F-actin, alpha-actinin, VASP, FAK, Zyxin	0.832
4	VASP, FAK, Zyxin, Pax pY118	0.824
4	Hic-5, alpha-actinin, VASP, Zyxin	0.823
4	Vinculin, F-actin, Zyxin, Pax pY118	0.820
4	Hic-5, VASP, Zyxin, FAK pY397	0.819
3	FAK, Zyxin, FAK pY397	0.812
4	Vinculin, VASP, Zyxin, FAK pY397	0.810
6	Hic-5, F-actin, alpha-actinin, VASP, FAK, Pax pY118	0.759
3	Hic-5, VASP, FAK	0.718
3	Hic-5, FAK, FAK pY397	0.704
4	Vinculin, Hic-5, F-actin, alpha-actinin	0.665
3	Vinculin, VASP, FAK	0.664
4	Hic-5, F-actin, VASP, Pax pY118	0.661
4	Hic-5, VASP, FAK pY397, Pax pY118	0.655
4	Vinculin, F-actin, alpha-actinin, FAK	0.653
4	F-actin, alpha-actinin, VASP, FAK	0.643
3	Vinculin, FAK, FAK pY397	0.633
5	F-actin, alpha-actinin, VASP, FAK pY397, Pax pY118	0.605
3	Hic-5, alpha-actinin, FAK pY397	0.598
4	F-actin, alpha-actinin, VASP, Pax pY118	0.574
4	F-actin, alpha-actinin, FAK pY397, Pax pY118	0.522
1	VASP	0.397

Table A.7: Input protein combinations to explain paxillin pY118

No. of Inputs	Input Proteins	Determination Coeff.
6	Vinculin, Paxillin, Hic-5, alpha-actinin, FAK, Zyxin	0.490
5	Vinculin, Paxillin, VASP, Zyxin, FAK pY397	0.483
5	Paxillin, Hic-5, VASP, FAK, FAK pY397	0.471
6	Vinculin, Paxillin, Hic-5, F-actin, VASP, Zyxin	0.469
5	Vinculin, Paxillin, Hic-5, F-actin, alpha-actinin	0.466
5	Vinculin, Hic-5, alpha-actinin, VASP, Zyxin	0.464
4	Vinculin, Hic-5, alpha-actinin, Zyxin	0.460
4	Vinculin, Paxillin, Hic-5, VASP	0.456
4	Vinculin, VASP, FAK, Zyxin	0.454
3	alpha-actinin, Zyxin, FAK pY397	0.447
3	Vinculin, alpha-actinin, Zyxin	0.437
4	Hic-5, F-actin, FAK, Zyxin	0.435
3	Hic-5, FAK, Zyxin	0.434
4	Paxillin, Hic-5, alpha-actinin, VASP	0.434
3	Paxillin, Hic-5, Zyxin	0.430
3	Paxillin, VASP, Zyxin	0.428
3	VASP, FAK, Zyxin	0.422
3	Paxillin, alpha-actinin, VASP	0.420
2	Hic-5, Zyxin	0.415
3	Paxillin, Hic-5, F-actin	0.415
2	Paxillin, VASP	0.405
2	Hic-5, FAK pY397	0.402
2	alpha-actinin, Zyxin	0.401
3	Vinculin, Hic-5, F-actin	0.389
2	VASP, FAK pY397	0.374
3	alpha-actinin, VASP, FAK	0.368
2	Hic-5, VASP	0.358
1	FAK pY397	0.302
2	Hic-5, F-actin	0.301

Table A.8: Input protein combinations to explain VASP

No. of Inputs	Input Proteins	Determination Coeff.
6	Vinculin, Paxillin, Hic-5, F-actin, alpha-actinin, Zyxin	0.569
5	Vinculin, F-actin, alpha-actinin, Zyxin, FAK pY397	0.559
4	Paxillin, Hic-5, alpha-actinin, Zyxin	0.553
6	Paxillin, F-actin, FAK, Zyxin, FAK pY397, Pax pY118	0.552
5	Paxillin, Hic-5, F-actin, Zyxin, Pax pY118	0.548
5	Vinculin, F-actin, Zyxin, FAK pY397, Pax pY118	0.539
4	Vinculin, Hic-5, FAK, Zyxin	0.539
4	Vinculin, Hic-5, F-actin, Zyxin	0.537
3	Hic-5, FAK, Zyxin	0.532
3	Paxillin, FAK, Zyxin	0.524
2	Paxillin, Zyxin	0.514
5	Hic-5, F-actin, FAK, FAK pY397, Pax pY118	0.490
3	Vinculin, Paxillin, alpha-actinin	0.481
4	Hic-5, alpha-actinin, FAK pY397, Pax pY118	0.480
3	Hic-5, alpha-actinin, FAK	0.464
4	Vinculin, F-actin, alpha-actinin, FAK pY397	0.456
3	Vinculin, alpha-actinin, FAK	0.447
3	alpha-actinin, FAK, FAK pY397	0.431
2	Vinculin, Hic-5	0.431
2	Paxillin, Pax pY118	0.429
3	Vinculin, FAK pY397, Pax pY118	0.429
2	Hic-5, FAK	0.428
2	Paxillin, F-actin	0.404
2	Vinculin, Pax pY118	0.399
2	FAK, FAK pY397	0.366
2	alpha-actinin, FAK pY397	0.354
1	Vinculin	0.320

Table A.9: Input protein combinations to explain vinculin

No. of Inputs	Input Proteins	Determination Coeff.
6	Paxillin, alpha-actinin, VASP, FAK, Zyxin, FAK pY397	0.613
6	F-actin, alpha-actinin, VASP, FAK, Zyxin, FAK pY397	0.602
4	Paxillin, alpha-actinin, FAK, Zyxin	0.596
3	VASP, FAK, Zyxin	0.585
4	Hic-5, VASP, Zyxin, Pax pY118	0.560
4	Paxillin, alpha-actinin, VASP, FAK	0.549
3	Paxillin, VASP, FAK	0.543
4	Hic-5, alpha-actinin, VASP, FAK	0.533
3	Paxillin, alpha-actinin, FAK	0.519
4	Paxillin, alpha-actinin, VASP, Pax pY118	0.517
4	Paxillin, Hic-5, VASP, Pax pY118	0.517
3	Paxillin, VASP, Pax pY118	0.505
2	Hic-5, FAK	0.503
2	Paxillin, VASP	0.483
3	Hic-5, FAK pY397, Pax pY118	0.464
1	FAK	0.427
3	Hic-5, F-actin, VASP	0.414
3	alpha-actinin, VASP, Pax pY118	0.410
2	alpha-actinin, Pax pY118	0.273

Table A.10: Input protein combinations to explain zyxin

No. of Inputs	Input Proteins	Determination Coeff.
3	Vinculin, Paxillin, Pax pY118	0.839
4	Paxillin, F-actin, VASP, Pax pY118	0.835
2	Vinculin, Paxillin	0.829
4	Paxillin, FAK, FAK pY397, Pax pY118	0.828
2	Paxillin, FAK	0.811
4	Vinculin, Hic-5, FAK, FAK pY397	0.763
6	Vinculin, Hic-5, alpha-actinin, VASP, FAK pY397, Pax pY118	0.761
4	Hic-5, alpha-actinin, FAK, Pax pY118	0.734
5	F-actin, alpha-actinin, VASP, FAK, Pax pY118	0.732
3	Vinculin, alpha-actinin, VASP	0.685
2	Vinculin, VASP	0.670
3	Vinculin, alpha-actinin, FAK pY397	0.632
3	alpha-actinin, FAK, FAK pY397	0.628
2	Vinculin, Pax pY118	0.626
4	F-actin, alpha-actinin, FAK pY397, Pax pY118	0.538
3	Hic-5, F-actin, alpha-actinin	0.534
1	VASP	0.498
2	alpha-actinin, Pax pY118	0.410
2	F-actin, FAK pY397	0.408

B. Simulation parameters–definition

- **Number of subpopulations** : we varied this parameter in order to evaluate the algorithm for situations on data sets with mixtures of more than two populations. When we work with a number of 10 nodes or less the possibility to identify more than two subpopulations, statistically different, decrease. Thus for this parameter we could expect a decrease in accuracy when the number of subpopulations is increased in the data. For the evaluation of this parameter we conduct simulations for generating datasets with mixtures of 2 and 4 subpopulations

- **Sample proportion per subpopulation**: varying this parameter we generated realistic situations in which the relative abundance of the different subpopulations is unbalanced. Here we aimed to assess whether the algorithm perform equally well when the relative abundance of the subpopulations is highly unbalanced. The evaluation of this parameter in our analysis was carried out under the generation of data sets with a mixture of 2 subpopulations, onsidering the following sample proportion 1:1, 1:3 and 1:7.

- **Network density**: The network density represents the proportion of possible connections (relationships) in the network that are actually present. The value ranges from 0 to 1, with the lower limit corresponding to networks with no connections and the upper limit representing networks with all possible connections. The closer the value is to 1, the more dense is the network and the more cohesive are the nodes in the network. Based on density networks of small network that are commonly studying in biological systems we aimed to evaluate the model for relatively low density networks (0.3) and relatively high

(0.6) in order to identify whether an increase or decrease in this parameter would lead to a better or worse results.

- **Number of nodes (proteins)**: increasing the number of nodes we provide more complex network scenarios, since with the increased number of nodes, the number of parent configuration for each node grows exponentially. Thus computationally the search for a graph to represent the relationship between the nodes involved becomes more difficult, thus consequently leading to less accurate results in identifying subpopulations. What we aimed here is to know how much less accurate are the results with the increase of the number of nodes. In our analysis we generated networks with 3, 5 and 10 nodes considering the fact that current measurement techniques in biological systems can measure up to 10 proteins simultaneously.

- **Intra-subpopulation variability**: The intra-subpopulation variability is defined through the variation in protein expression levels within each subpopulation and the length of the series, since in classification of time series plays an important roll how long the time series is when it comes to recognizing temporal patterns in the presence of noise. The intra-subpopulation variability en general affects the network system itself, thus its propagation through the system generates asymmetric high-order patterns shaped by the topology of the network. Because our algorithm identify distinct subpopulations based on the differences between their temporal network structures, we evaluated the effect of intra-subpopulation variability on the accuracy of our algorithm. To introduce intra-subpopulation cell-to-cell variability, for each run of the simulation we sampled the protein expression levels around the respective initial values for a given set of fractional deviation from the mean ($\sigma = \mu \cdot fd$, where $fd \in \{0.1,\ 0.5,\ 0.8\}$), where fd represents the degree of stochastic variance in the protein expression levels. Two time series length were considered, $n = 20$ to represent short time series and $n = 100$ for long time series.

C. Similarity measures between two graphs

En esta seccion nosotros desarrollamos medidas de similaridad entre dos estructuras graphicas. The similarity measures here presented are based on the edge betweenness centrality (EBC) and the weights matrix $\mathbf{W}$ of the first-order multivariate vector autoregressive model (MVAR) described in section 13.1. The EBC is traditionally used to determine which edges in a network are most important, while the weights matrix $\mathbf{W}$ contains the coefficients which represent the strength of interaction between all the pair of proteins.

Definition C.0.1 (Edge betweenness centrality (EBC))

The EBC is defined as the number of the shortest paths that go through an edge in a graph. For a graph $G = (\mathbf{V}, \mathbf{E})$ where $\mathbf{V} = \{V_1, \ldots, V_n\}$ is the finite set of nodes and $\mathbf{E} \subseteq \mathbf{V} \times \mathbf{V}$ the set of edges, the EBC of edge e is computed by

$$EBC(e) = \sum_{\substack{V_i, V_j \in \mathbf{V} \\ i \neq j}} \frac{N_{ij}(e)}{N_{ij}}, \tag{C.1}$$

where $N_{ij}(e)$ denotes the total number of shortest path between V_i and V_j that pass through edge e and N_{ij} denotes the total number of shortest paths between V_i and V_j. An edge with high EBC score plays an important role in the network and its removal may affect the communication between many pairs of nodes. For more details about EBC and other topological centrality measures see Dehmer and Emmert-Streib (2014).

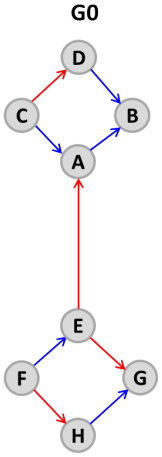
Each edge in a network can be associated with an EBC value. An edge with a high EBC score represents a bridge-like connector between two parts of the network, and its removal may affect the communication between many pairs of genes (nodes) through the shortest paths between them.

Definition C.0.2 (Weights matrix $\mathbf{A}$)

The weights Matrix $\mathbf{W} = (w_{ij})_{n\times n}$ is the n-by-n time-invariant matrix of coefficients of the first-order multivariate vector autoregressive model (MVAR) to be employed in the simulation study, where the coefficients represent the strength of interaction between all the pair of proteins. After depicting a network topology $G = (\mathbf{V}, \mathbf{E})$, a connectivity matrix $\mathbf{A} = (a_{ij})_{n\times n}$ is generated from G where $a_{ij} = 1$ for the presence of connection between protein i and j, and $a_{ij} = 0$ for the absence. Then, the weight matrix $\mathbf{W}$ is obtained by assigning weights randomly to all the connection (where $a_{ij} = 1$) in the connectivity matrix $\mathbf{A}$. These weights are assigned by getting the values from uniform distribution on the intervals $[-1.5, -0.5] \cup [0.5, 1.5]$. Two intervals are chosen to maintain the amount of negative and positive weights nearly equal.

Definition C.0.3 (Sign matrix $\mathbf{SW}$ related to $\mathbf{W}$)

A sign matrix $\mathbf{SW} = (s_{ij})_{n\times n}$ is a matrix of 0s, 1s, and -1s, where $s_{ij} = 0$ if $w_{ij} = 0$, $s_{ij} = 1$ if $w_{ij} > 0$ and $s_{ij} = -1$ if $w_{ij} < 0$.




```
> print(g0, e=TRUE, v=TRUE)
IGRAPH DN-- 8 9 --
+ attr: name (v/c)
+ edges (vertex names):
[1] A->B C->A C->D D->B E->A E->G F->E F->H H->G
> Weights_Matrix
      A    B C     D    E F     G     H
A  0.00 0.79 0  0.00 0.00 0  0.00  0.00
B  0.00 0.00 0  0.00 0.00 0  0.00  0.00
C  0.74 0.00 0 -0.54 0.00 0  0.00  0.00
D  0.00 0.51 0  0.00 0.00 0  0.00  0.00
E -0.52 0.00 0  0.00 0.00 0 -0.65  0.00
F  0.00 0.00 0  0.00 0.64 0  0.00 -0.35
G  0.00 0.00 0  0.00 0.00 0  0.00  0.00
H  0.00 0.00 0  0.00 0.00 0  0.72  0.00
> SignW_Matrix
   A B C  D E F  G  H
A  0 1 0  0 0 0  0  0
B  0 0 0  0 0 0  0  0
C  1 0 0 -1 0 0  0  0
D  0 1 0  0 0 0  0  0
E -1 0 0  0 0 0 -1  0
F  0 0 0  0 1 0  0 -1
G  0 0 0  0 0 0  0  0
H  0 0 0  0 0 0  1  0
> Edge_Betweenness_Centrality_Scores
    A   B C   D   E F   G   H
A 0.0 3.5 0 0.0 0.0 0 0.0 0.0
B 0.0 0.0 0 0.0 0.0 0 0.0 0.0
C 1.5 0.0 0 1.5 0.0 0 0.0 0.0
D 0.0 1.5 0 0.0 0.0 0 0.0 0.0
E 4.0 0.0 0 0.0 0.0 0 1.5 0.0
F 0.0 0.0 0 0.0 3.5 0 0.0 1.5
G 0.0 0.0 0 0.0 0.0 0 0.0 0.0
H 0.0 0.0 0 0.0 0.0 0 1.5 0.0
```

Figure C.1: Example of the EBC scores matrix, weights matrix and the sign matrix for a particular graph.

Since the EBC measures describe structural property of the edges in a network, then we could calculate the difference between two topologies using common measures of distance as a function of EBC scores, the $\mathbf{W}$ and $\mathbf{SW}$ matrices. We employed here the Manhattan distance and thereby we define three different similarity measures:

Let $\mathbf{Q_p} = (q_{pij})_{n\times n}$, $\mathbf{W_p} = (w_{pij})_{n\times n}$, $\mathbf{SW_p} = (s_{pij})_{n\times n}$, the EBC scores matrix, the weights matrix and the sign matrix beloging to the network $p = 1, 2$, respectively.

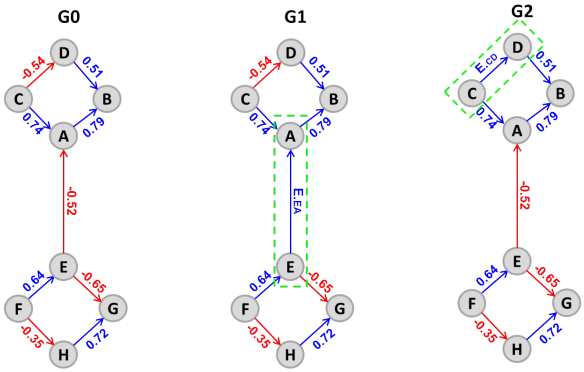
$$D_{BC} = \sum_{\substack{V_i,V_j\in\mathbf{V}\\ i\neq j}} |q_{1ij} - q_{2ij}|, \tag{C.2}$$

$$D_{BCSW} = \sum_{\substack{V_i,V_j\in\mathbf{V}\\ i\neq j}} |q_{1ij} \times s_{1ij} - q_{2ij} \times s_{2ij}| \tag{C.3}$$

$$D_{BCW} = \sum_{\substack{V_i,V_j\in\mathbf{V}\\ i\neq j}} |q_{1ij} \times w_{1ij} - q_{2ij} \times w_{2ij}| \tag{C.4}$$

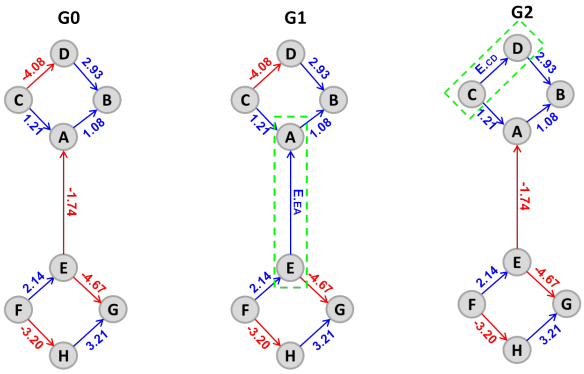
C.1 Examples using the three similarity measures

C.1.1 Example 1. Low weights



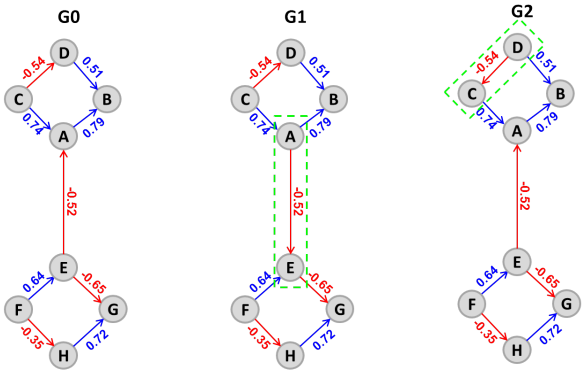
Diff	D_{BC}	D_{BCSW}	D_{BCW}
G0 vs. G1 (EEA=0.01)	0.000	8.000	2.120
G0 vs. G1 (EEA=0.80)	0.000	8.000	5.280
G0 vs. G2 (ECD=0.01)	0.000	3.000	0.825
G0 vs. G2 (ECD=0.80)	0.000	3.000	2.010

C.1.2 Example 2. High weights



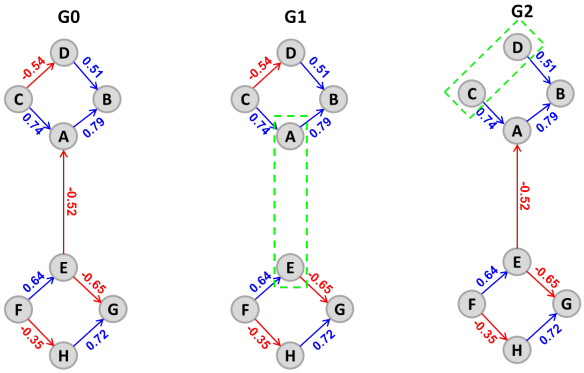
Diff	D_{BC}	D_{BCSW}	D_{BCW}
G0 vs. G1 (EEA=1)	0.000	8.000	10.960
G0 vs. G1 (EEA=5)	0.000	8.000	26.960
G0 vs. G2 (ECD=1)	0.000	3.000	7.620
G0 vs. G2 (ECD=5)	0.000	3.000	13.620

C.1.3 Example 3. Changing the direction of an edge



Diff	D_{BC}	D_{BCSW}	D_{BCW}
G0 vs. G1	16.000	22.000	14.065
G0 vs. G2	5.000	5.000	3.110

C.1.4 Example 4. Deleting an edge



Diff	D_{BC}	D_{BCSW}	D_{BCW}
G0 vs. G1	8.000	8.000	4.940
G0 vs. G2	3.000	3.000	1.830