Markov Random Fields in Cancer Mutation Dependencies

Oscar Lindberg

Master's Thesis
May 2016

DEPARTMENT OF MATHEMATICS AND STATISTICS
UNIVERSITY OF TURKU

UNIVERSITY OF TURKU
Department of Mathematics and Statistics

LINDBERG, OSCAR: Markov Random Fields in Cancer Mutation Dependencies
Master's Thesis, 40 pages, 12 appendix pages
Statistics
May 2016

Even though a large amount of evidence would suggest that PP2A serine/threonine protein phosphatase acts as a tumour suppressor the genomics data to support this claim is limited. We fit a sparse binary Markov random field with individual sample's total mutational frequency as an additional covariate to model the dependencies between the mutations occurring in the PP2A encoding genes. We utilize the data from recent large scale cancer genomics studies, where the whole genome from a human tumour biopsy has been analysed.

Our results show a complex network of interactions between the occurrence of mutations in our twenty examined genes. According to our analysis the mutations occurring in the genes PPP2R1A, PPP2R3A, and PPP2R2B are identified as the key mutations. These genes form the core of the network of conditional dependency between the mutations in the investigated twenty genes. Additionally, we note that the mutations occurring in PPP2R4 seem to be more influential in samples with higher number of total mutations.

The mutations occurring in the set of genes suggested by our results has been shown to contribute to the transformation of human cells. We conclude that our evidence further supports the claim that PP2A acts as a tumour suppressor and restoring PP2A activity is an appealing therapeutic strategy.

Keywords: Biometry, Cancer, Graphs, Mutations, Biometria, Graafit, Mutaatiot, Syöpä

Contents

1 Introduction

1.1 Cancer Driver Mutations

Hanahan et al. (2000) present six acquired functional capabilities that cells must obtain through genetic alterations in order to become cancerous and gain selective advantage to outgrow competing cells. These traits include evading apoptosis, self-sufficiency in growth signals, insensitivity to anti-growth signals, tissue invasion and metastasis, limitless replicative potential, and sustained angiogenesis. By identifying possible mutations behind obtaining these traits our knowledge of cancer can be increased and new possible targets for cancer treatments can be discovered. Most mutations found in tumours are random background mutations, so called passenger mutations, that have no beneficial effect that would give a cell and its successors selective advantage in outgrowing other cells (Lawrence et al. 2013). One way to distinguish the possible meaningful driver mutations from the passenger mutations is to look for any non-random pattern in the distribution of different mutations in cancer samples. Conditional dependence structure between mutations can be explored for evidence on pairs of genes being mutated together less or more often than would be expected by chance.

A negative link i.e. mutual exclusion between two mutations means that they appear together less often than would be expected due to both being random background mutations. This would hint that the mutations give similar benefits for the cancer and once one of the mutations has happened the other does not offer the tumour development any more selective advantage and becomes redundant or even disadvantageous for the tumour. A positive link i.e. co-occurrence means that the two mutations are more likely to be found together in cancerous cells than separately. This gives cause to believe that they have a positive effect on the cancer cell when occurring together but separately do not affect the cell, or that appearing together the mutations have a synergistic interaction, that gives the tumour greater benefit than the effects of the mutations individually.

The data for such an analysis have become available in recent years through various large scale cancer genomic studies. In this thesis the data used in exploring dependence structure of cancer mutations is taken from the COSMIC (Catalogue Of Somatic Mutations In Cancer) database (see Forbes et al. 2014), which is to the authors knowledge the most comprehensive database of somatic mutations found in cancer cells. The current release (v76 February 2016) holds data from nearly 1,192,776 samples taken from 22,844 different papers. For the purposes of this thesis we utilize only the genome-wide data found in the database. This limits us to 18,783 samples from 162 studies, but allows a view across the breadth of cancer genome without any specific biases introduced via literature curation. New releases of the COSMIC database are made trimonthly adding data on a regular basis. This large amount of data that has a growing amount of cancer samples with their whole genome analysed makes it possible to use statistical modelling to explore possible novel cancer drivers and find additional evidence on significance of previously discovered drivers.

In recent years different methods to identify cancer drivers through modelling co-occurrence and mutual exclusion have been proposed in literature. Ciriello et al. (2012) present a method called Mutual Exclusivity Modules (MEMo) in cancer. This method uses a Human Reference Network (HRN) derived from existing pathway and interaction databases to identify gene pairs that are likely to belong into the same pathway. A graph is formed by drawing an edge between the gene pairs. Separate regions from this graph form cliques or local clusters that are likely to hold similar functions. These cliques are then assessed for mutual exclusivity between genomic alterations by an empiric p-value calculated by comparing the observed alteration frequency to an expected alteration frequency obtained by randomly permuting the set of observed mutations. Vandin et al. (2012) introduced an algorithm called De novo Driver Exclusivity (Dendrix) that searches for subsets of genes that maximize a weight function that rewards coverage while penalizing for overlap. This means that in a cancerous cell at least one mutation from the same subset is likely to be found mutated but if one of the genes in a subset is mutated then the other genes from the same subset are less likely to be mutated. Szczurek et al. (2014) proposed a statistical modelling framework for mutual exclusivity (ME) using a generative model that includes parameters representing pattern coverage, impurity, false positive rate, and false negative rate.

For this thesis our aim is to explore the mutual exclusivity and co-occurrence of mutations found in cancer cells in order to find evidence of underlying network of interactions between occurrence of mutations in the investigated genes. We focus our analysis on the genes encoding the PP2A subunits in order to distinguish which mutations occurring in this set of genes drive the cancer forward and which have occurred in our tumour samples by chance. A common way to represent the mutual exclusivity and co-occurrence of mutations in different genes is to use undirected graphs, where the nodes represent the genes and the edges represent an interaction in the connected genes being mutated. We have chosen to model these interactions using a Markov random field, which gives us an estimate for a set of parameters that can be used directly to encode an undirected graph. Using Markov random fields we also obtain the added benefit of being able to utilize methods for parameter estimation, which are not too intensive to be run on a personal computer.

1.2 Sparse Markov Random Fields

The most common and best known Markov random field models consist of continuous variables usually assumed to have a Gaussian distribution. The multivariate Gaussian distribution has at most pairwise conditional dependencies between the variables so it is straightforward to interpret it as a Markov random field i.e. a pairwise undirected graph. For detailed description of representing a multivariate Gaussian random distribution as a Markov random field see Lauritzen (1996). The information of the conditional dependencies between the variables is contained in the inverse covariance matrix. In particular, if an off-diagonal element of the inverse covariance is zero then the two corresponding variables are conditionally independent, given the other variables. If the graph structure is known, the elements corresponding with non-existent edges are constrained to zero and the estimation of the rest of the parameters is an equality-constrained convex optimization problem. However, usually estimating the graph structure by distinguishing the existing edges from non-existent is part of the problem. In recent years many authors have proposed the use of the L1-regularization to discover the graph structure from the data itself. Estimating sparse graphs by a lasso penalty applied to the inverse covariance matrix is known as graphical lasso, for proposed implementations see Meinshausen and Bühlmann (2006) and Friedman et al. (2008).

The multivariate distribution of the variables in the network to be estimated is not always Gaussian. In this thesis the data consist of binary variables. These kinds of binary Markov random field networks are also known as Ising models, especially in the statistical mechanics literature, or Boltzmann machines especially in the machine learning literature. The Ising model is named after physicist Ernst Ising, who presented a model for ferromagnetism in statistical mechanics in his dissertation published in 1925 (Ising, 1925). In this model atomic spins are presented as binary variables belonging to set $\{-1, +1\}$. Ising presented a solution for a 1-dimensional chain of such variables and then generalized it onto a 3-dimensional lattice. For the problem of solving the dependence structure of cancer mutations the Ising model suits well because the data can be structured in a 2-dimensional lattice where the columns represent the genes of interest and the rows represent samples from biopsies taken from the found tumours. The data are binary as each sample either harbors a mutation in the gene of interest or not. As most alterations in genes are random background mutations, for most pairs of genes there is no dependency between having alterations in either of the genes. Therefore, the Ising model needs to be penalized by adding sparsity to the model. This allows for distinguishing if mutations happen independently, or if there is a dependency. As the primary goal of the analysis is to find information on the underlying Markov field graph structure rather than to estimate connections from a known structure, the problem is quite different than the one Ising set out to solve with his model. However, the Ising model can be used as a base and generalized to allow for sparsity. Sparsity means adding a penalty parameter to the model that penalizes for each parameter estimated as non-zero. Due to this difference from a situation where the graph structure is known, methods often used in such situations e.g. Poisson log-linear modelling or Gibbs sampling are infeasible.

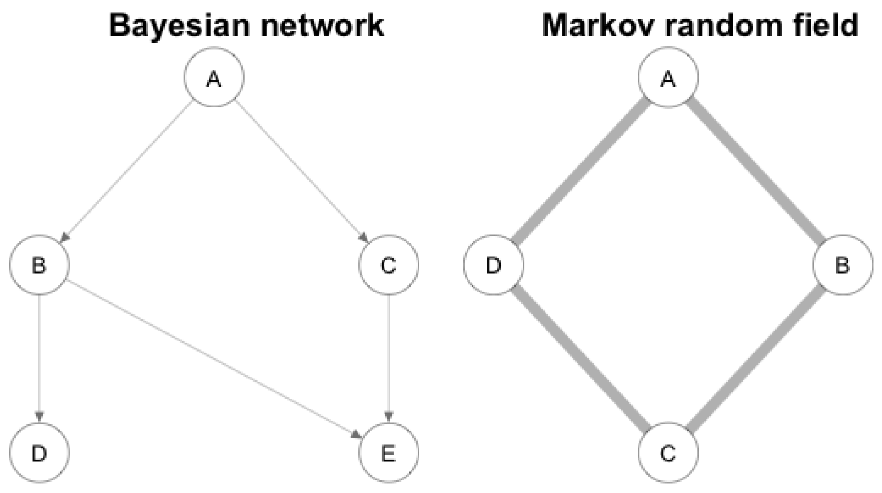
Estimating the graph structure of a pairwise Markov network is not a trivial task due to the complex nature of the likelihood function. However, multiple methods have been introduced for tackling this issue. Besag (1975) presented a way to formulate an approximation for the likelihood known as the pseudo-likelihood that can be maximized for an approximate solution. This is the basis for a method presented by Höfling & Tibshirani (2009) that starts by maximizing the pseudo-likelihood and then adjusts the the pseudo-likelihood criterion so that each additional iteration moves it closer to the exact solution. Lee et al. (2007) propose a method that maximizes the penalized log-likelihood starting with only a subset of the variables and then adding new variables into the model with the grafting procedure. This method will lead to an exact solution when using the junction tree algorithm to calculate the log-partition function of the penalized log-likelihood but the calculation becomes cumbersome if the amount of variables is not small. The authors implement a loopy belief propagation algorithm for better performance but approximate inference. Wainwright et al. (2006) described an efficient method designed especially for finding the underlying graph structure by performing separate L_1-regularized logistic regressions on each variable and then symmetrizing the estimated matrix by taking either the minimum or the maximum of the corresponding estimates.

Cheng et al. (2014) generalized the binary pairwise Markov field by adding additional covariates to the model and estimate this model with the Wainwright approach. Adding dependency towards additional covariates to the model is useful because often these *peripheral* variables are being recorded when collecting the binary data and the dependency of the binary data to these *peripheral* variables is often justified. For example, in the context of the analysis presented in this thesis the mutational frequency of each biopsy over all possible mutations can be calculated. This frequency, with a possible range between 1 mutation to roughly 20,000 mutations, is an obvious measure of the mutational tendency of a sample. This tendency can be argued to have an effect on the probability of mutations of interest so it can be seen as a useful addition to the model.

The binary pairwise MRFs are known for the computational intractability of the exact likelihood. This imposes difficulty on assessing the model fit and analysing the model consistency. The choice of L1-penalty parameter is also difficult. Hastie et al. (2009) recommend examining solution paths as a function of the penalty parameter in the multivariate Gaussian case. Same approach could also be used in the binary setting. Cheng et al. (2014) explore different options via simulation and find cross-validation to be a reasonable solution to balancing the true positive and the false positive rate. For assessing the model consistency, in other words whether the model structure stays the same as the amount of observations grows, one possible method to consider is bootstrapping (see e.g. Efron and Tibshirani 1998). Bach (2008) uses bootstrap in least-square linear regression with L1-penalty to mimic having multiple datasets from the same underlying distribution. In theory Lasso should always select all relevant variables with a strictly positive probability while irrelevant variables enter the model randomly. This means that intersecting the non-zero variables from sufficiently many datasets should eliminate irrelevant variables appearing by chance. This property could be utilized to examine the consistency of the edges in binary MRFs by evaluating the rate an edge is part of the final estimate when running multiple analyses with bootstrapped datasets from the same underlying distribution.

2 Markov Random Fields

Figure 1: Sample graphs drawn from a Bayesian network and a Markov random field



Markov random fields together with Bayesian networks are the most important subgroups of a set of models called probabilistic graphical models (see Koller & Friedman 2009 for details of probabilistic graphical models). Graphical models are named after their property to effectively represent complex distributions in a compact way as graphs. These graphs consist of two types of elements. The oval shaped vertices, or nodes, correspond to the different variables within the data, and the edges between the vertices correspond to the probabilistic interactions between the variables. Two vertices joined by an edge are called *adjacent* and multiple vertices connected by a set of edges are called *paths*.

The main difference between Bayesian networks and MRFs lies in the nature of the edges. In Bayesian networks the edges have a target and a direction while the edges in MRFs are undirected (see Figure 1). The differences in the variable independence structure and the induced factorization can be seen in Table 1. In essence, in Bayesian networks the variables form a parent-child hierarchy, where each variable is conditionally independent of the other variables, given its parent variables.

The MRFs have no hierarchy between variables, and the absence of an edge between two vertices indicates the two variables being conditionally independent,

	Bayesian networks	Markov random fields
Independences:	$(B \perp\!\!\!\perp C \mid A)$ $(E \perp\!\!\!\perp A \mid B, C)$ $(D \perp\!\!\!\perp A, C, E \mid B)$	$(A \perp\!\!\!\perp C \mid B, D)$ $(B \perp\!\!\!\perp D \mid A, C)$
Factorization:	$P(A, B, C, D, E) =$ $P(A)P(B\mid A)P(C\mid A)$ $P(D\mid B)P(E\mid B, C)$	$P(A, B, C, D) =$ $\frac{1}{\Psi}\phi_1(A, B)\phi_2(B, C)\phi_3(C, D)$ $\phi_4(D, A)$

Table 1: Independences and factorization of the Bayesian network and MRF presented in Figure 1. The functions ϕ are *clique potential* functions as defined in Equation (1).

given the other variables. If all paths connecting two subgraphs $\mathcal{G}_1, \mathcal{G}_2$ intersect vertice A then A is said to separate $\mathcal{G}_1$ and $\mathcal{G}_2$. The separators break the graph into conditionally independent pieces. This property is known as *the global Markov property.* A *clique* is a fully connected subset of the set of vertices. This means that each pair of nodes within a clique $(s, t) \in c$ also defines an edge in the graph edge set $(s, t) \in E$. A clique is *maximal* if no new vertices can be added to it still yielding a clique. For example, the MRF in Figure 1 has 4 maximal cliques: $\{A, B\}, \{B, C\}, \{C, D\}, \{A, D\}$. Any single variable is in itself a clique. However, it is a maximal clique only if it is completely isolated, in other words, it participates in no edges in the edge set.

Regarding notation, we write $x = (x_1, \ldots, x_p) \in \mathcal{X}$ for a single observation of p variables. A clique of size k can then be identified by an ordered set of indices

$$c = (i_1, ..., i_k), \quad 1 \leq i_1 < \ldots < i_k \leq p,$$

which defines the set of variables in clique c as

$$x_c = (x_{i_1}, ..., x_{i_k}) \in \mathcal{X}_c.$$

Bayesian networks are parametrized by utilizing the conditional probability distributions. However, because the interactions in Markov random fields are not directed, a different, symmetric approach needs to be taken as we need to capture the affinities between the related variables. For this, a non negative function called *clique potential* or *factor* is defined as a function (1) of values that a set of random variables $\mathcal{X}_S$ can take.

$$\phi : \mathcal{X}_S \mapsto \mathbb{R}_+ \tag{1}$$

As an example, take variables X_1 and X_2. The higher the value of the potential for some occurrence $\phi(x_1, x_2)$, the more compatible these two values are. Let us assume that X_1 and X_2 are both binary variables. Then one possible choice of the potential function is

$$\phi_1(X_1, X_2) = \begin{cases} 20, \text{if } X_1 = 0 \wedge X_2 = 0 \\ 5, \text{if } X_1 = 1 \wedge X_2 = 0 \\ 10, \text{if } X_1 = 0 \wedge X_2 = 1 \\ 40, \text{if } X_1 = 1 \wedge X_2 = 1 \end{cases} .$$

With this choice for ϕ it can be noted that it is more likely for X_1 and X_2 to share the same value, with more weight on the case that the shared value is 1. This realization of the potential function also has the property that $\phi_1(X_1 = 0, X_2 = 1) > \phi_1(X_1 = 1, X_2 = 0)$. It is more likely to have $X_1 = 0$, if the variables do not share the same value.

To define a global model for the whole graph, the local models described by the clique potentials are combined by multiplication. The potential functions are not in general density functions, in the exemplary ϕ_1 none of the entries are even in $[0, 1]$. Therefore, the result of the multiplication cannot be guaranteed to be a probability density. To obtain a legal distribution the end result needs to be normalized so that the sum of probabilities is bound to 1.

Let the joint probability function of all variables in the undirected graph be

$$f(x_1 \dots x_p) = \frac{1}{\Psi} \prod_{c \in C} \phi_c(\mathbf{x}_c), \tag{2}$$

where C is a set of cliques, $\phi_c(x_c)$ is the value of the potential function when the value of the random variables $\mathcal{X}_c$ in the clique is the observed $\mathbf{x}_c$ and Ψ is the normalizing constant. Specifically,

$$\Psi = \sum_{\mathbf{x} \in \mathcal{X}} \prod_{c \in C} \phi_c(\mathbf{x}_c). \tag{3}$$

This normalizing constant is known as the *partition function.* The name stems from the fact that Ψ is a function of the model potentials. This dependency is the key source of difficulty associated with estimating the model parameters.

The joint probability distribution of the graph is defined by factoring together the potential functions of cliques, which by definition include only the totally connected variables. The result of this is a tight connection between the factorization of the distribution and its independence properties. Let's consider a network with three variables X_1, X_2, X_3. Then,

$$P(X_1, X_2, X_3) \models (X_1 \perp\!\!\!\perp X_2 | X_3) \Leftrightarrow P(X_1, X_2, X_3) = \frac{1}{\Psi}\phi_1(X_1, X_3)\phi_2(X_2, X_3), \quad (4)$$

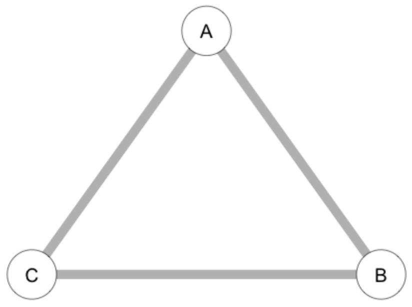
to put it otherwise, X_1 and X_2 can be considered independent given X_3, if and only if, the joint probability distribution function can be factored without a potential function for (X_1, X_2). This connection makes sense intuitively, as we only want direct interactions between the variables to be represented in the graph structure. The left side of equivalence (4) can be interpreted as a graph lacking an edge potential between X_1 and X_2 being able to fully represent the probability measure function P.

The factorization of a graph is not uniquely specified. The number of potential functions factored can be reduced by allowing functions only for maximal cliques. A fully connected graph with potentials for every pair of the variables is a single large maximal clique containing all of the variables. Associating a single potential function with this clique captures the between-variable dependence completely, but yields an amount of potentials that is exponentially large compared to the amount of the original variables ($2^p - 1$ parameters).

A more practical and often used factorization specifies only at most second-order dependence. This subclass of Markov random fields, known as pairwise Markov fields, reduces the amount of potentials by restricting the potential functions factoring graph G to node potentials: $\{\phi(X_i) : i = 1 \dots n\}$ and edge potentials: $\{\phi(X_i, X_j) : (X_i, X_j) \in G\}$. This factorization can be used to obtain joint probability distributions of exactly similar graph structures as factorization with more complex potential functions. All the edges found in graph G are now assumed to be defined by the two nodes connected by the edge, all interactions with the other nodes are assumed to be contained within the other edges leading to the two nodes. For a fully connected graph this reduces the number of potentials needed to $n + \binom{n}{2}$.

As an example take the fully connected 3-node graph in Figure 2. This graph

Figure 2: A fully connected graph with 3 nodes.



could represent the dependence structure from either of the distributions

$$
\begin{aligned}
P_{\Phi}^{(2)}(a,b,c) &= \frac{1}{\Psi}\phi(a,b)\phi(b,c)\phi(a,c), \text{ or} \\
P_{\Phi}^{(3)}(a,b,c) &= \frac{1}{\Psi}\phi(a,b,c).
\end{aligned}
$$

The pairwise model $P_{\Phi}^{(2)}(a,b,c)$ has six potentials: three for each node and three for the edges between the nodes. The second model $P_{\Phi}^{(3)}(a,b,c)$, based on the maximal clique with all three nodes, has in total seven potentials. Six of them are shared with the pairwise model, the last one being the interaction potential for all of the three variables.

For the remainder of this thesis the Markov random fields being covered are considered pairwise. Restricting the models to at most the pairs of variables in the edge set of the graph makes working with the models easier by giving the benefit of having a cost-effective number of potentials, thus giving the model minimal complexity implied by the graph structure.

3 Gaussian Markov Random Fields

If the variables represented by an undirected graph are continuous, they are most often assumed to be from a multivariate Gaussian distribution. The multivariate Gaussian distribution over random variables $X_1, \ldots, X_p$ has two possible parametrizations with distinct properties. The density function in the *moment form*, parametrized by a mean vector μ and a symmetric $p \times p$ covariance matrix, can be written in the familiar form of

$$f_x(x_1,\dots,x_p) = \frac{1}{(2\pi)^{p/2}|\Sigma|^{1/2}} \exp(-\frac{1}{2}(x-\mu)'\Sigma^{-1}(x-\mu)). \tag{5}$$

In order for the equation (5) to induce a well-defined density, the covariance matrix Σ needs to be positive definite. Because positive definite matrices can be inverted, we derive another representation for (5) by defining the distribution with the inverse of the covariance matrix $\Theta = \Sigma^{-1}$. Taking the expression in the exponent of (5)

$$\begin{aligned} -\frac{1}{2}(x-\mu)'\Sigma^{-1}(x-\mu) &= -\frac{1}{2}(x-\mu)'\Theta(x-\mu) \\ &= -\frac{1}{2}(x'\Theta x - 2x'\Theta\mu + \mu'\Theta\mu) \end{aligned}$$

The last term is a constant. We write

$$f_x(x_1,\dots,x_p) \propto \exp(-\frac{1}{2}x'\Theta x + x'\nu), \tag{6}$$

where $\nu = \Theta\mu$ is known as the potential vector and the inverted covariance matrix Θ is the Fisher information matrix. Hence, the formulation is called *information form.*

In order to express the multivariate Gaussian distribution as a Markov random field, the expression in the exponent of the information form (6) of the distribution needs to be separated into two types of terms: those involving single variables X_i (7), and those involving pairs of variables (X_i, X_j) (8).

$$\text{Terms involving single variables } X_i\text{:} \quad -\frac{1}{2}\Theta_{i,i}x_i^2 + x_i\nu_i \tag{7}$$

$$\begin{aligned} \text{Terms involving pairs of variables } (X_i, X_j)\text{:} \quad & -\frac{1}{2}(\Theta_{i,j}x_ix_j + \Theta_{j,i}x_jx_i) \qquad (8) \\ &= -\Theta_{i,j}x_ix_j \end{aligned}$$

Here, the simplification in (8) is based on the symmetry of the information matrix.

The separation of terms in (7) and (8) directly induces a pairwise Markov random field. The node potentials are derived from the potential vector and the diagonal of the information matrix, and the edge potentials from the off-diagonal elements of the information matrix. Also, due to the independence properties of multivariate Gaussian distributions zero-valued off-diagonal elements of the information matrix Θ imply conditional independence, given all of the other variables, which corresponds the lack of an edge in the MRF graph representation. Thus, any multivariate Gaussian distribution can be represented as a pairwise Markov random field.

3.1 Gaussian MRF with Known Graph Structure

Let $x_1 \dots x_n$ be a random sample from a multivariate Gaussian distribution $N_p(\mu, \Sigma)$. In order to estimate the node and the edge potentials of a saturated model with all the possible edges present, start with the empirical covariance matrix S

$$S = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})(x_i - \bar{x})', \tag{9}$$

known to be the maximum likelihood estimator of Σ, where $\bar{x}$ is the sample mean vector.

The logarithm of the multivariate Gaussian density function (5) is of the form

$$\ln(f_x(x_1, \dots, x_p)) = -p\ln(2\pi)/2 - \ln|\Sigma|/2 - (x-\mu)'\Sigma^{-1}(x-\mu)/2. \tag{10}$$

Thus, from (10) we get the log-likelihood of the n-sized sample

$$\begin{aligned}\ell(\mu, \Theta) &= -\frac{np}{2}\ln(2\pi) - \frac{n}{2}\ln|\Theta|^{-1} - \sum_{i=1}^{n}(x_i-\mu)'\Theta(x_i-\mu)\\ &= -\frac{np}{2}\ln(2\pi) - \frac{n}{2}\ln|\Theta|^{-1} - \sum_{i=1}^{n}(x_i-\bar{x})'\Theta(x_i-\bar{x}) - n(\bar{x}-\mu)'\Theta(\bar{x}-\mu).\end{aligned}$$

This can be further simplified via the cyclicity of the trace and using (9)

$$-\frac{np}{2}\ln(2\pi) - \frac{n}{2}\ln|\Theta|^{-1} - \frac{n}{2}\operatorname{Tr}(S\Theta) - \frac{n}{2}(\bar{x}-\mu)'\Theta(\bar{x}-\mu).$$

By excluding the constants, one sets

$$\ln|\Theta| - \operatorname{Tr}(S\Theta) - (\bar{x}-\mu)'\Theta(\bar{x}-\mu).$$

Finally, replacing μ with the maximum likelihood estimator $\hat{\mu} = \bar{x}$ we write the log-likelihood for Θ excluding constants as

$$\ell(\Theta) \propto \ln|\Theta| - \operatorname{Tr}(S\Theta), \tag{11}$$

which is a concave function of Θ. Setting the gradient of (11) as zero, we find the maximum likelihood estimator of Θ to be S^{-1}.

However, the model of interest in most cases is not saturated like the one proposed by (11). Non-existence of an edge between two nodes in the graph representation implies that the corresponding entry in Θ is zero. For this reason we need to constrain a subset of parameters to zero while maximizing (11).

To provide this constriction, Lagrange constants are added to the log-likelihood (11).

$$\ell_C(\Theta) = \ln|\Theta| - \mathrm{Tr}(S\Theta) - \sum_{(j,k)\notin E} \gamma_{jk}\theta_{jk}. \tag{12}$$

Setting the gradient of (12) as zero, we write the maximizing equation as

$$\Theta^{-1} - S - \Gamma = 0, \tag{13}$$

where Γ is a matrix of Lagrange parameters with non-zero entries corresponding to non-existent edges.

It can be shown that regression can be used to solve for Θ and its inverse, marked here with W. This approach is chosen because it relates with the methods in the following sections of this thesis. For the reasoning Hastie et al. (2009) are followed.

For simplicity the focus is on the last row and column. Starting with the upper right block of equation (13), we write it as

$$w_{12} - s_{12} - \gamma_{12} = 0. \tag{14}$$

Here the matrices are partitioned into two parts: the first $p-1$ rows and columns and the final pth row and column. Partitioning Θ and its inverse W in a similar fashion we get

$$\begin{bmatrix} W_{11} & w_{12} \\ w_{12}^T & w_{22} \end{bmatrix} \begin{bmatrix} \Theta_{11} & \theta_{12} \\ \theta_{12}^T & \theta_{22} \end{bmatrix} = \begin{bmatrix} I & 0 \\ 0^T & 1 \end{bmatrix}. \tag{15}$$

By using the standard formulas for partitioned inverse matrices we find that

$$\begin{aligned} w_{12} &= -W_{11}\theta_{12}/\theta_{22} \\ &= W_{11}\beta, \end{aligned} \tag{16}$$

where $\beta = -\theta_{12}/\theta_{22}$. Now substituting w_{12} in (14), we write

$$W_{11}\beta - s_{12} - \gamma_{12} = 0. \tag{17}$$

The equation (17) corresponds to $p-1$ estimating equations for the regression of X_p on the other variables, but replacing the observed mean cross-products matrix S_{11} with the current estimated covariance matrix W_{11}. We can solve (17) with subset regression. Suppose that γ_{12} has $p-q$ non-zero entries, which represent the $p-q$ edges constrained to be zero. Thus, the corresponding rows carry no information and are left out. Likewise, β is reduced to β^* by leaving out the $p-q$ zero elements. This yields a reduced system of $q \times q$ equations

$$W_{11}^*\beta^* - s_{12}^* = 0, \tag{18}$$

which is solved for an estimate of β^*

$$\hat{\beta}^* = W_{11}^{*-1} s_{12}^*. \tag{19}$$

Augmenting $\hat{\beta}^*$ with the $p-q$ zeroes we get an estimate for β. Looking at the equation (16) it seems that θ_{12} can only be estimated up to a scaling factor $1/\theta_{22}$. However, using formulas for partitioned inverse matrices on (15) we show that

$$\frac{1}{\theta_{22}} = w_{22} - w_{12}^T\beta, \tag{20}$$

and from (13) we observe that $w_{22} = s_{22}$, because the diagonal elements of Γ are zero. Thus, after solving for $\hat{\beta}$, we find $\hat{\theta}_{12}$ from

$$\hat{\theta}_{12} = -\hat{\beta}\hat{\theta}_{22}, \tag{21}$$

where

$$\frac{1}{\hat{\theta}_{22}} = s_{22} - w_{12}^T\hat{\beta}. \tag{22}$$

3.2 Gaussian MRF with Unknown Graph Structure

It is not often the case that we know the graph structure a priori. Rather, discovering the structure of the graph from the obtained data is often among the main goals of the analysis. To discover the structure of a graph we can take advantage of the parsimonious property of Lasso regression. Continuing to follow Hastie et al. (2009), we begin with the addition of L_1 penalty to the profile log-likelihood (11)

$$\ell_{\text{Lasso}}(\Theta) \propto \ln|\Theta| - \text{Tr}(S\Theta) - \lambda||\Theta||_1. \tag{23}$$

Here, $||\Theta||_1 = \sum_{i=1}^p \sum_{j=1}^p |\theta_{ij}|$ is the L_1 norm. Analogous to (13), we take the gradient of (23) and set to zero

$$\Theta^{-1} - S - \lambda\,\text{sign}^*(\Theta) = 0, \tag{24}$$

where $\text{sign}^*(\theta_{ij}) = \text{sign}(\theta_{ij})$ if $\theta_{ij} \neq 0$ and some value in the range of $[-1, 1]$ otherwise, using the sub-gradient notation due to the derivative of absolute value function being in-determined at zero.

Continuing to follow the reasoning in the previous section, we similarly to (17) represent the upper right block of (24) as

$$W_{11}\beta - s_{12} + \lambda \operatorname{sign}^*(\beta) = 0, \tag{25}$$

which can be shown to be of a similar form with the gradient of a common lasso regression. Thus, we use lasso regression to find $\hat{\Theta}$ by regressing each X_k on the other variables and updating W until convergence, then during the final cycle transforming estimated $\hat{\beta}_{(k)}$ vector using (21) and (22) for each $k \in [1, p]$. This method is known as the *graphical lasso* (Friedman et al. 2008). It has achieved popularity due to being fast and efficient, and therefore being utilizable with moderately sparse graphs with a large number of nodes.

4 Binary Markov Random Fields

In many scenarios the variables representing the nodes in our graph cannot be defined as continuous. This prevents taking advantage of the useful properties of the multivariate Gaussian distribution introduced in the previous sections, essentially providing more difficulty into the process. In the following sections the focus is in graphs having nodes with a binary set $\{0, 1\}$ of possible values, but for the most part results presented could be generalized to apply for graphs having nodes with more complex discrete sets of values.

We start with the joint probability function for a graph with p binary variables. Let $X = (X_1, \ldots, X_p)$ be a random vector, where each X_s takes values in set $\{0, 1\}$. Let a graph with p nodes each corresponding to a variable in X be denoted by $G = (V, E)$. We find the joint probability by defining a potential function (1) for each node in V and each edge in E. *The Ising model* defines the joint probability function (2) as

$$P_{\Theta}(X) = \frac{1}{\Psi(\Theta)} \exp\left(\sum_{s \in V} \theta_s x_s + \sum_{(s,t) \in E} \theta_{st} x_s x_t \right). \tag{26}$$

Because the edges in E are undirected, we set similarly to (8) that $\theta_{st} = \theta_{ts}$ for all $s > t$. Now, conditioning X_s on the remaining variables $X_{\setminus s}$ we write the conditional log-odds for X_s as

$$\ln\left(\frac{P(x_s = 1 | x_{\setminus s})}{1 - P(x_s = 1 | x_{\setminus s})} \right) = \theta_s + \sum_{s : s \neq t} \theta_{st} x_t. \tag{27}$$

Solving (27) for probability $P(x_s = 1|x_{\setminus s})$ we find that

$$P(x_s = 1|x_{\setminus s}) = \frac{1}{1 + \exp(-\theta_s - \sum_{s:s\neq t} \theta_{st} x_t)}, \tag{28}$$

which can be recognized as logistic regression.

This suggests that logistic regression can be used to estimate the model parameters. The usefulness of this notion becomes apparent when we take a look at the partition function

$$\Psi(\Theta) = \sum_{x\in\{0,1\}^p} \exp(\sum_{s\in V} \theta_s x_s + \sum_{(s,t)\in E} \theta_{st} x_s x_t). \tag{29}$$

As can be seen from (29), maximizing the joint likelihood directly is cumbersome due to requiring the summation of 2^p terms for each data point. Thus, using separate logistic regressions (28) maximizing the conditional log-likelihood of each X_s given $X_{\setminus s}$ instead of maximizing the joint likelihood is a very lucrative option.

Given input data in the form of $\{x, y\}$, where y is a binary response and x is a $p-1$ dimensional vector of covariates, logistic regression is solved by minimizing the negative log-likelihood of n observations

$$\ell(\theta; x) = -\frac{1}{n} \sum_{i=1}^{n} \ln(1 + \exp(\theta_0 + \sum_{j=1}^{p-1} \theta_j x_{ij})) - y_i(\theta_0 + \sum_{j=1}^{p-1} \theta_j x_{ij}). \tag{30}$$

Estimating the parameter matrix Θ via the connection to logistic regression (28) yields a collection of separate regression problems (30), where each variable X_s is in turn regressed onto the remaining variables sharing the same data across all p problems. The p dimensional vector of parameters corresponding to intercept terms $\theta_0^{(1)}, \ldots, \theta_0^{(p)}$ gained from the p regressions forms the diagonal of the $\hat{\Theta}$ matrix, so that the sth element of the diagonal of $\hat{\Theta}$ corresponds to the bias term of variable X_s. The off diagonal elements of $\hat{\Theta}$ are the parameters describing the size of the conditional effect of the particular predictor on the variable being regressed, so that $\hat{\theta}_{st}$ is the main effect of x_s on the conditional log-odds of x_t.

In practice, formulating the problem as separate logistic regressions simplifies solving the problem by a great deal as logistic regression is a widely used method with efficient implementations found throughout respectable statistical software packages. However, simply iterating through the p logistic regressions and stacking the estimated parameter vectors $\hat{\theta}^{(1)} \ldots \hat{\theta}^{(p)}$ into a common matrix where the intercept is placed on the diagonal, does not produce a valid estimate of $\hat{\Theta}$. The problem stems from the fact that, as the edges are presumed to be undirected, the matrix Θ should be symmetrical with respect to the main diagonal. Using separate regressions does not guarantee this symmetricity, rather as $\hat{\theta}_{st}$ and $\hat{\theta}_{ts}$ are estimated from the sth and tth regression, respectively, having either X_s or X_t as the regressed variable and the other as one of the predictors, the supposedly equal estimates will most likely differ by some degree.

To symmetrize the final estimated matrix, the corresponding initial estimates need to be combined by some function. One possibility would be to take the mean of the two initial estimates as the final estimate. Although it does combine the information from both initial estimates into the final estimates, this approach will cause problems when the problem is expanded to the estimation of the graph structure. If one of the two parameter estimates is zero and the other non-zero, the mean of the two values cannot be seen as describing the true parameter. Two more suitable approaches following Meinshausen and Bühlmann (2006) use either the minimum or the maximum of the initial estimates.

$$\text{separate-min: } \hat{\theta}_{st} = \hat{\theta}_{ts} = \hat{\theta}_{st}^{\text{init}} \mathbb{I}_{(|\hat{\theta}_{st}^{\text{init}}| < |\hat{\theta}_{ts}^{\text{init}}|)} + \hat{\theta}_{ts}^{\text{init}} \mathbb{I}_{(|\hat{\theta}_{ts}^{\text{init}}| < |\hat{\theta}_{st}^{\text{init}}|)} \tag{31}$$

$$\text{separate-max: } \hat{\theta}_{st} = \hat{\theta}_{st} = \hat{\theta}_{st}^{\text{init}} \mathbb{I}_{(|\hat{\theta}_{st}^{\text{init}}| > |\hat{\theta}_{ts}^{\text{init}}|)} + \hat{\theta}_{ts}^{\text{init}} \mathbb{I}_{(|\hat{\theta}_{ts}^{\text{init}}| > |\hat{\theta}_{st}^{\text{init}}|)} \tag{32}$$

Using the separate-min approach, if one of the initial estimates is zero then the final estimate will be zero. Thus, the separate-min is more conservative of the two approaches providing a greater amount of zero estimates compared to the separate-max approach.

4.1 Binary MRF with Unknown Graph Structure

Similarly to when the Markov random field is comprised of continuous variables, often the structure of the graph containing binary variables is not known a priori. Thus, one of the main goals of the analysis is to use the data to estimate, which of all the possible edges between the pairs of nodes are included in the edge set of the graph. That is using data to distinguish the pairs of variables that are dependent on each other from the pairs that are independent of each other, given all the remaining variables. The approach described is similar to the *graphical lasso* described in section 3.2 in the sense that L_1 regularization is used to add sparsity corresponding to non-existent edges. However, unlike in the *graphical lasso*, where the whole parameter matrix is estimated simultaneously by updating the estimated inverse of the parameter matrix by iterating through regressions on each of the nodes, completely separate logistic regressions on each of the nodes is used to obtain the initial estimates to be post-processed by either the separate-min or the separate-max approach.

Due to the elements of the diagonal of the parameter matrix Θ having the interpretation of being the bias terms for the nodes, the regularization added to the negative log-likelihood (30) should only account for the non-intercept parameters. That is, only the parameters corresponding to the edges of the graph should be susceptible to being estimated as zero due to the L_1 constraint. Regressing each variable X_s onto the remaining variables $X_{\setminus s}$, sharing the same data across the problems leads to the collection of p problems, one for each node

$$\hat{\theta}^{s,\lambda} = \underset{\theta \in \mathbb{R}^p}{\arg\min} \left\{ \ell_s(\theta_s; X) + \lambda ||\theta_{\setminus 0}||_1 \right\}, \tag{33}$$

where

$$\ell_s(\theta_s; X) = -\frac{1}{n} \sum_{i=1}^{n} \ln(1 + \exp(\theta_0 + \sum_{j \neq s} \theta_j x_j)) - x_s(\theta_0 + \sum_{j \neq s} \theta_j x_j), \tag{34}$$

and $s \in V$ i.e. the set of nodes. The estimated $\hat{\theta}_t^{s,\lambda}$ can be seen as a penalized conditional likelihood estimate for θ_{st}.

The main focus of the analysis being the estimation of the structure of the graph G, and estimating the magnitudes of individual parameters only a secondary concern, we write the estimation on the node level as

$$\hat{N}(s) = \{t \in V, \, t \neq s : \, \hat{\theta}_t^{s,\lambda} \neq 0\}. \tag{35}$$

Otherwise put, for each node s, the neighbourhood $N(s)$ of s is estimated by estimating which of the parameters associated with the other variables are non-zero.

This method presented, proposed by Wainwright et al. (2006), has been shown to consistently estimate the neighbourhood of every node in the graph simultaneously even for increasing amount of nodes p and maximum number of edges d per single node as long as the amount of observations $n = \Omega(d^3 \ln p)$, where $f(n) = \Omega(g(n))$ if $f(n) \geq Kg(n)$ for some constant $K > 0$ (see Wainwright et al. 2008). Although the magnitudes of the parameters are only a secondary concern with this method, comparisons by Höfling et al. (2009) show that the method also produces accurate estimations of the magnitudes of parameters, when compared to competing methods.

4.2 Adding Additional Covariates

In many real life applications the structure of the graph cannot be assumed to be dependent only on the random variables represented by the graph's nodes. Additional covariates with possible relations to the node variables are in many cases being collected together with the binary variables. The possible dependence of the graph structure on these additional covariates found on collected datasets motivated Cheng et al. (2014) to propose a method to study both the conditional dependency within the binary data and the effect of additional covariates.

Supposing we have additional covariate information, the data consist of n independent and identically distributed data points $\{(x^{(1)}, z^{(1)}), \ldots, (x^{(n)}, z^{(n)})\}$, where $x^{(s)} \in \{1, 0\}^p$, and $z^{(s)} \in \mathbb{R}^q$. Similarly to (33), x now represents the p binary variables belonging to the set of nodes in our graph. We use z to represent the q covariates. The data are assumed to follow the Ising distribution (26) given by

$$P_\Theta(X|Z) = \frac{1}{\Psi(\Theta)} \exp\left(\sum_{s \in V} \theta_s(z) x_s + \sum_{(s,t) \in E} \theta_{st}(z) x_s x_t\right). \tag{36}$$

Similarly to the definition (35) of the model without additional covariates, variables x_s and x_t are conditionally independent given all of the remaining node variables and covariates z if $\theta_{st}(z) = 0$.

To model $\theta_{st}(z)$, it is parametrized as a linear function of z. Specifically, we define

$$\begin{aligned} &\theta_{st}(z) = \theta_{st0} + \theta_{st}^T z, && \text{where } \theta_{st}^T = (\theta_{st1}, \ldots, \theta_{stq}) \\ &\theta_{st}(z) = \theta_{ts}(z), && \forall s > t. \end{aligned} \tag{37}$$

Expressed in terms of this expanded parametrization, the model can be written as

$$P_\Theta(X|Z) = \frac{1}{\Psi(\Theta)} \exp\left(\sum_{s=1}^{p} (\theta_{ss0} + \theta_{ss}^T z) x_s + \sum_{(t>s)} (\theta_{st0} + \theta_{st}^T z) x_s x_t\right), \tag{38}$$

and the conditional log-odds (27) for an individual node becomes

$$\ln\left(\frac{P(x_s = 1|x_{\setminus s}, z)}{1 - P(x_s = 1|x_{\setminus s}, z)}\right) = \theta_{ss0} + \theta_{ss}^T z + \sum_{t: t \neq s} (\theta_{st0} + \theta_{st}^T z) x_t. \tag{39}$$

We estimate the parameters through a similar collection of regression problems as in (33), where the additional covariates are simply included as new predictors to the regressions. The criterion for solving the regression problem for each node s is then of the form

$$\min_{\theta_s \in \mathbb{R}^{p(q+1)}} -\frac{1}{n}\sum_{i=1}^{n} \ln(1+\exp(\theta_{ss0}+\theta_{ss}^T z_i \sum_{j\neq s}\theta_j x_j)) - x_s(\theta_0 + \sum_{j\neq s}\theta_j x_j). \tag{40}$$

As can be seen from (39), the amount of parameters for each node s is increased from p to $p(q+1)$. Thus, the amount of unique parameters for a full model increases from $p(1+p)/2$ to $(q+1)p(p+1)/2$.

As using a linear parametrization (37) for $\theta_{st}(z)$ does not interfere with the conclusion that logistic regression can be used to model the conditional dependency of an individual node, we can argue that interpretation for the parameters in our model conforms with logistic regression. That is, each parameter describes the size of the conditional effect of that particular predictor. Specifically, let z_k be a continuous variable, now the corresponding parameter θ_{stk} describes the effect of z_k on the conditional log-odds of x_s when $x_t = 1$. The parameter θ_{st0} describes main effect of x_t on x_s. To determine whether an edge exists between two nodes, the whole vector of corresponding variables needs to be examined. That is, all entries in the vector $(\theta_{st0}, \theta_{st}^T)$ being zero implies that the sth and tth variables are conditionally independent given any z and the remaining node variables.

Similarly as before, sparsity is achieved by adopting the L_1 regularization as a part of the estimation. Specifically, the criterion for each individual regression (40) is expanded by adding the regularization parameter, giving the resulting criterion for each node s in the same form as (33). The difference lies only within the negative conditional log-likelihood, in which the additional covariates are added. Similarly as before, every variable except the intercept term is penalized, including all of the additional covariates.

This method lets us expand our Ising graphical model via addition of extraneous factors in the form of new covariates. Thus, we have subject-specific graphical models, where the strength of an edge varies smoothly with the values of the introduced covariates. The introduced covariates can be either continuous or categorical, but as a consequence of continuity, if all introduced covariates are continuous, the value of the covariate changing does not change the graph structure, instead it can only have an effect on the strengths of edges. However, with categorical new covariates we can have a different graph structure for each possible level of the covariates.

4.3 Value of the Regularization Parameter

The addition of the L_1 regularization enables us to have sparsity in our model by forcing some of the parameters to be estimated as zero. We are able to influence the number of zero and non-zero values in our final estimate through the choice of the value for the regularization parameter. A higher value strengthens the penalty and we obtain less non-zero estimates, and vice versa, a lower value lets us have a greater amount of non-zero values in our final estimate.

The regularization has a strong influence on our estimation, as our final estimate can range from having a non-zero estimate only for the intercept terms of each node to having $(q+1)p(p+1)/2$ non-zero terms, including the second degree interactions between nodes and between nodes and additional variables. Because the value of the regularization parameter has such a strong impact on our estimation and ultimately the conclusions we can draw from our results, the choice of the value for the parameter deserves to be one of the key points in the analysis.

Hastie et al. (2009) state that in practice it is often informative to run the analysis multiple times with different values for the regularization parameter and then to explore the set of solutions. Increasing the value of the regularization parameter and observing which edges of the graph remain while others are excluded, can provide insight on which interactions are the most meaningful in the phenomenon producing the studied data.

However, in the context of this study we advocate the view that simply exploring the different obtained solutions is not sufficient and instead we should pursue finding an optimal value for the regularization parameter. Cheng et al. (2014) compare different methods for finding the optimal value for the parameter. According to their simulations they conclude that cross-validation is the preferred option, when compared to validating the conditional likelihood on a separate dataset of the same size, using AIC, or using BIC.

There are multiple approaches for conducting a cross-validation but for our purposes we propose using a 10-fold cross-validation optimizing for the positive predictive value (41) in single mutations. Specifically, we divide our data into ten partitions of equal sizes. Each of these partitions is in turn used as a test set, while the remaining nine partitions are used as a training set which is used to estimate model parameters. Using the solution estimated from the training set we calculate the linear predictors for each node variable observation in the test set. A positive linear predictor means that, according to our model, we predict that it is more likely to encounter an event(=1) in the observation in question. We then compare predicted values of nodes with the actual values in our training set to find how the model performed with the tested regularization parameter.

The most common way to conduct cross-validation in a binary setting is trying to minimize the misclassification rate. However, for this thesis we consider a regularization parameter to be optimal when we maximize the positive predictive value while minimizing the amount of non-zero parameters. This is mainly due to the nature of our data where the number of non-events(=0) outweighs the events in our node variables. Thus, we consider a model which is able to correctly predict relatively rare events to capture the underlying phenomenon well without overfitting on sample specific attributes. Minimizing the amount of non-zero parameters translates in this scenario to finding the greatest value of λ with acceptable positive predictive value.

$$\text{PPV} = \frac{\text{number of true positives}}{\text{number of true positives} + \text{number of false positives}} \tag{41}$$

4.4 Bootstrapping Edges

It is known that the L_1 regularization or LASSO selects all the variables that should enter the model with probability tending to one exponentially fast, while selecting the other variables with strictly positive probability. In order to evaluate the results of our estimation, specifically whether a non-zero parameter estimate corresponding to an edge between two nodes in our graph is reliable, we run our estimation several times using data re-sampled with replacement from our original sample. By re-sampling our original sample and then running the estimation on these *bootstrap* samples, we estimate the probability of having a zero result on a parameter with multiple datasets from the same underlying distribution.

Let P_{ij} be the probability that $\hat{\theta}_{ij} = 0$, or not having an edge between the ith and jth node in our estimated graph. We find the bootstrap estimate of P_{ij} by taking M bootstrap samples and running our estimation on each of these samples, yielding in total M estimates for θ_{ij}: $\hat{\theta}_{ij}^{(1)} \ldots \hat{\theta}_{ij}^{(M)}$.

Taking into account bias introduced from finite sampling, $\hat{P}$ is given by

$$\hat{P}_{ij} = \frac{1}{M+1} \sum_{k=1}^{M} (\mathbb{I}_{(\hat{\theta}_{ij}^{(k)}=0)}) + 1 \tag{42}$$

We use $\hat{P}$ to evaluate which edges are unlikely to have been observed due to random chance by comparing corresponding values of $\hat{P}$ to a cut-off value of 5%. That is, we consider edges estimated as non-zero in 95% or more of the bootstrap samples to be reliable evidence of interaction between the two variables given all the other variables.

5 Modelling Graph Structure in Cancer Mutation Data

Our goal is to analyse the mutations discovered in biopsies of cancerous tumours in order to find pairs of genes, which are mutated together less frequently than expected. Finding such pairs can be a sign of the two mutations giving similar benefit for the cancer in question. As our data source we use the COSMIC database (Forbes et al. 2014), which holds data from over 18,000 tumour samples that have had their whole genomes analysed. Recent large scale cancer genomics studies have enabled researchers to utilize mathematical methods to find mutational patterns, which may lead to discovering novel driver mutations.

However, as the results gained from analysing the data from these databases are being interpreted, one needs to also consider the potential problems associated with this form of data. Only a portion of the data has been compared to genome from healthy cells of the same individual. Thus, the data can be expected to contain false positive mutations, caused by the differences in individual genotypes. The selection bias involved with the original studies used as the source for the database will also influence the pooled data. We additionally note that, while these sort of databases often contain records of either presence or absence of a genomic alteration, despite modern experimental methodologies we are not yet able to identify alterations with complete certainty. The used data can be presumed to contain both false negative and positive records due to measurement noise and uncertainty in mutation calling and interpretation. Thus, the possible evidence for mutational patterns found should be mainly considered as indicating the involved genes for further investigations to establish biological proof of involvement in cancer pathogenesis.

5.1 Data Specifications and Reformatting

The most current iteration of the COSMIC database at the time of writing is v76 (available for download from https://cancer.sanger.ac.uk/cosmic). The downloaded data is in long format, where each individual mutation is represented by a row in the data. In total we start with 4,247,063 rows of data, from which we choose to keep only observations where the whole genome is screened. We reduce the data further by dropping silent mutations, which have no effect on the amino acid sequence of the protein when the altered mRNA is translated.

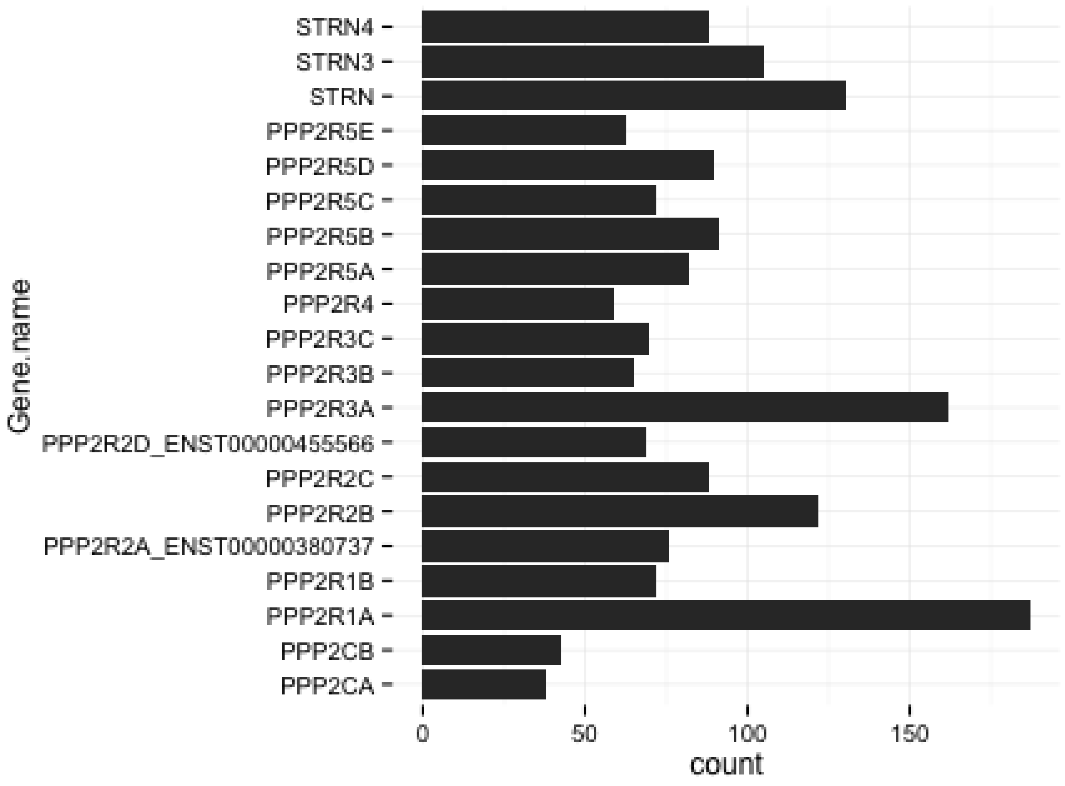
For our study, we focus on a set of genes known to code the enzyme protein phosphatase 2A (PP2A). We further narrow our set of data by selecting only the variant with the highest number of mutations for each gene, corresponding with the gene names: PPP2R1A, PPP2R3C, PPP2R2A_ENST00000380737, PPP2R5D, PPP2R5E, PPP2R3A, PPP2R5A, PPP2R2D_ENST00000455566, PPP2R3B, PPP2R4, PPP2R2B, PPP2R1B, PPP2CA, PPP2R5B, PPP2R5C, PPP2R2C, PPP2CB, STRN, STRN3, and STRN4 in the data.

After these steps, we have data with 1,772 records of mutations, with the PP2A encoding genes distributed as shown in Figure 3.

We transpose the data so that each row now represents an individual tumour from which a sample has been collected. Each one of our 20 genes of interest is represented by a column, with a binary set of possible values. For each sampled tumour, a mutation in the corresponding gene either is present (1) or not (0). We now have a data with 1,218 individual records from tumours with at least one of the 20 mutations mutated. From Figure 4 we observe that a clear majority of our sampled tumours have only one mutation of interest and the maximum amount of mutations of interest found in an individual tumour is eleven.

Hanahan et al. (2011) define genome instability and mutation as one of the enabling characteristics of cancer. Indeed, conditions that increase the mutation frequency or genome instability, such as a loss of function in one or several components of the genomic maintenance machinery, often contribute to onset and progression of various cancers. Thus, in some tumour samples we find a cumulated mutational load of several orders of magnitude greater than in other tumour samples. These *hypermutated* tumour samples can be expected to have a greater amount of mutations also in the PP2A encoding genes, as can be seen from Figure 5, where the curve represents a Poisson regression fit of number of PP2A mutations regressed on

Figure 3: Distribution of mutations in PP2A encoding genes.



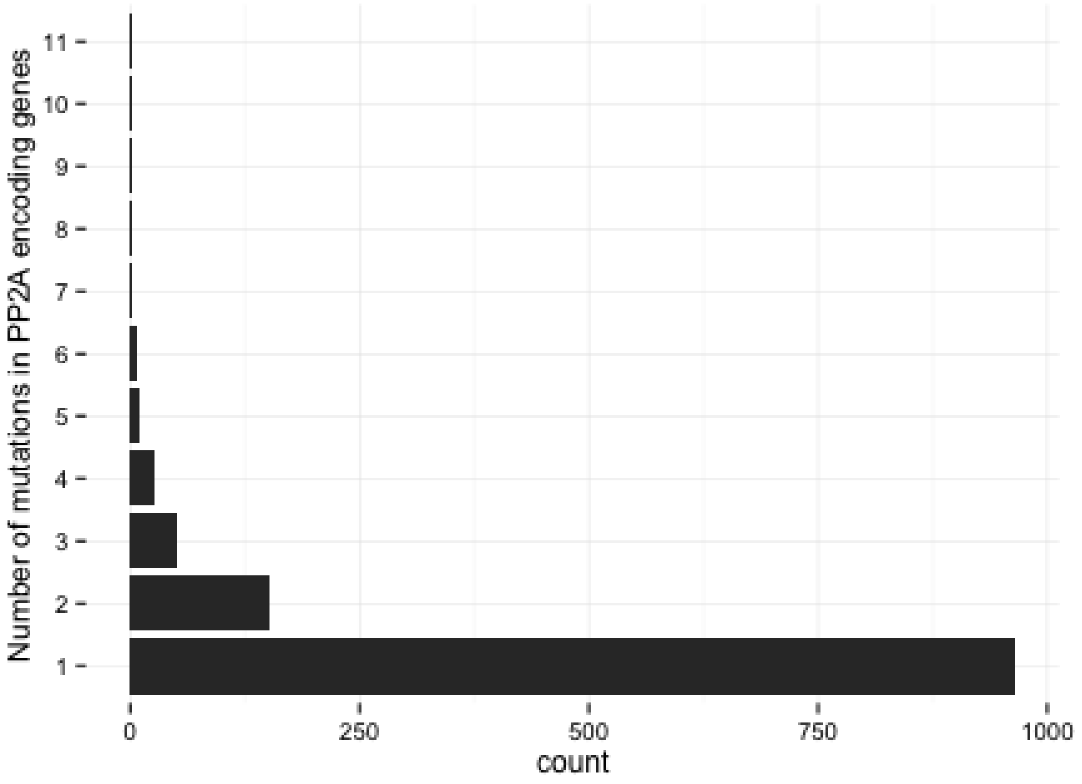
the logarithm of total number of mutations in a tumour sample.

In order to take into account the effect of the mutational load of each tumour sample in our analysis, we merge into the data a column representing the logarithm of total number of mutations for each tumour sample.

5.2 R functions used in the Analysis

We propose a binary Markov Random Field model with additional covariates as presented in section 4.2 to model the inherent dependencies between mutations in genes encoding the PP2A enzyme. Following the method of estimation by Cheng et al. (2014) also presented in section 4.2 we developed a function called BinaryMRFwith-Cov with the R programming language. The program code is listed in Appendix A. We also developed functions PredictBMRF (Appendix C) and CvBMRF (Appendix

Figure 4: Distribution of mutations in PP2A enconding genes.

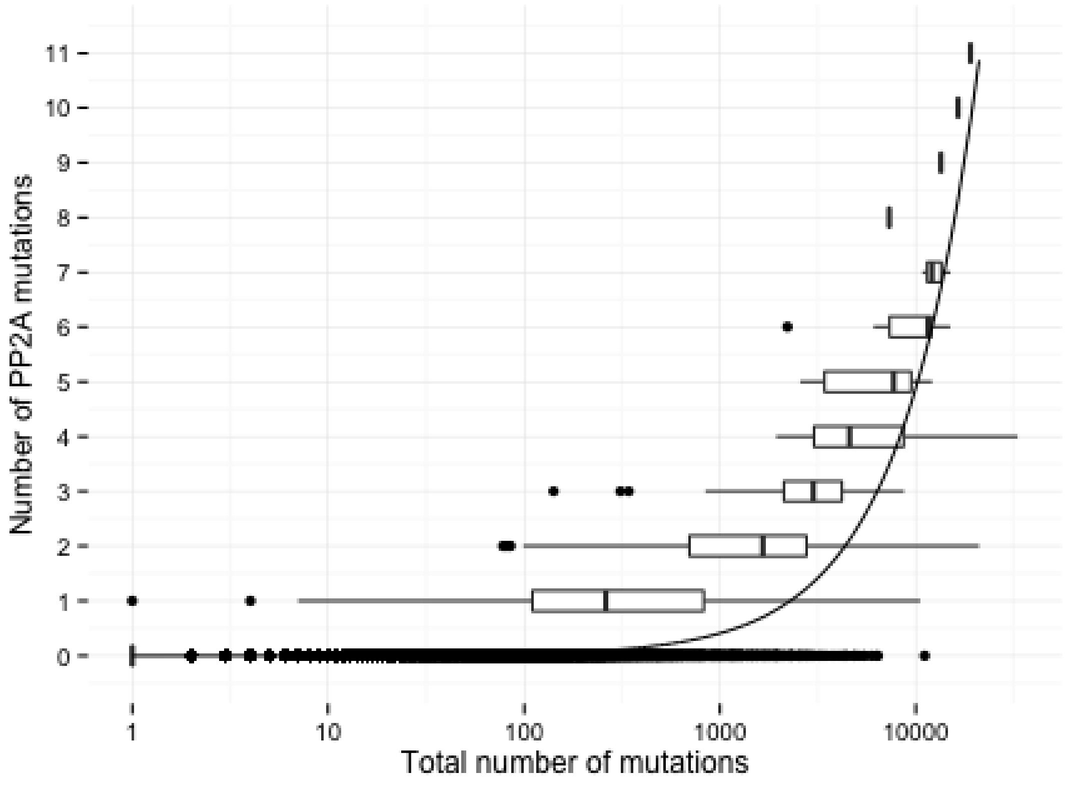


D) in order to evaluate the positive predictive value of the model with different values of the regularization parameter.

Additionally, we developed function BootstrapEdges (see Appendix B) with functionality as described in section 4.4 in order to further investigate the degree of belief we can account for the found edges by investigating the matrix ρ calculated from the bootstrap samples.

The functions are developed under R version 3.2.0 (2015-04-16) "Full of Ingredients". In order to calculate the individual L_1 regulated logistic regressions, our functions depend on the function *penalized*, from the package *penalized*. We also utilize the package *parallel* in order to save run time by calculating multiple regression estimations in parallel on systems with multiple cores. We utilize the function createFolds from the package *caret* to divide our data into folds to be used in cross-validation.

Figure 5: Mutations in PP2A encoding genes v. Total number of mutations in a tumour sample.



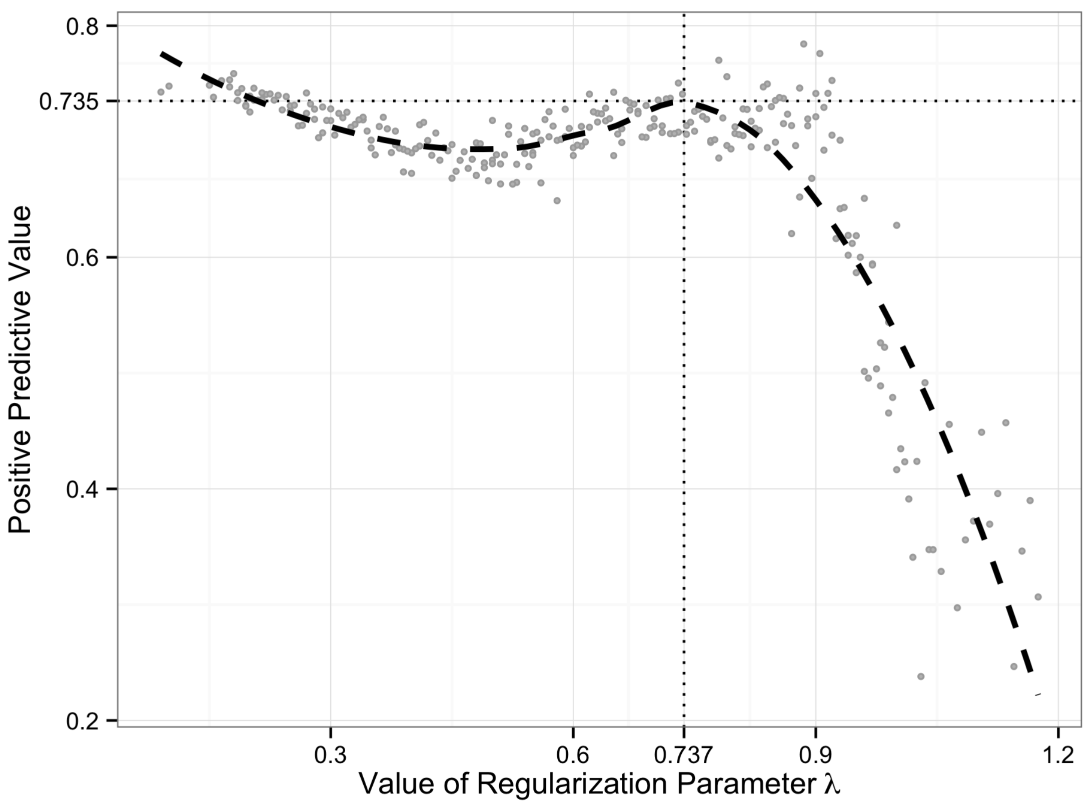
The function BinaryMRFwithCov returns a list including a matrix of numeric values named *graph*. This is the weights matrix of our MRF model which holds the final post-processed estimates of the main effect edge parameters. Multiple packages have been developed for producing visual representations of graphs given their weights matrices. For this thesis we have used the *qgraph* package to produce the visual output from our estimated weights matrices. For other plots we utilized the package *ggplot2* by Hadley Wickham (2009).

5.3 Statistical Analysis

5.3.1 Optimal Regularization Parameter

We begin by running the 10-fold cross-validation multiple times on our PP2A mutation dataset using different values of the regularization parameter λ. Our goal is to find a value, which produces a model that captures the nature of the underlying phenomenon as accurately as possible, while minimizing the number of used parameters. For each point of regularization parameter λ tested we obtain ten positive predictive values (41), one for each fold, which we average in order to find the mean positive predictive value for the tested value of the parameter. In Figure 6 we have plotted the results of multiple cross-validations performed with λ in range of $[0.1, 1.175]$. We have fitted a line with LOESS local regression to analyse the expected behaviour of PPV with different values of λ.

Figure 6: Mean Positive Predictive Values v. λ



Using our LOESS fit we find that the expected positive predictive value plateaus when the value of the λ parameter is 0.737. Thus, even if new parameters are added to the model with lower values of λ, the model does not perform better at predicting mutations. Setting λ at 0.737 our estimation will produce a model fit that be expected to be correct 73.5% of times when predicting a gene to be found mutated based on the status of the other 19 genes and the sample specific mutation rate with the minimal amount of parameters. Additionally, continuing to use cross-validation, we observe that at $\lambda = 0.737$ we obtain a negative predictive value of 94% with total accuracy of 94%.

Based on cross-validation we choose 0.737 as the value for our regularization parameter. Judging from our analysis we concur that we can obtain a solution with approximately 74% positive and 94% negative predictive values, which would suggest that the graph obtained with this value for the parameter captures the connections between occurrence of mutations in PP2A encoding genes. A lower value would introduce a greater amount of non-zero parameters, while not improving predictive performance, thus complicating the solution in vain. However, a greater value of λ would produce a fit that can be expected to fail in capturing the essential information in the data, as the expected positive predictive value decreases sharply when λ is greater than 0.737.

5.3.2 Estimation of Graph Structure and Parameters

We run our estimation on the mutation data of PP2A encoding genes using the value 0.737 as the value for the regularization parameter. We use the separate-max (32) to post-process the estimates due to superior performance when compared to the separate-min (31) according to simulations carried out by Cheng et al. (2014).

We use a binary Markov random field model with the additional covariates as described in section 4.2, where each individual gene is one of the node variables. We have added the logarithm of the total amount of mutations in each sample as an additional variable to take into account the strong interaction between the number of PP2A mutations and the total number of mutations found in a sample (Figure 5).

Our main interest lies in the parameters representing the main effect interactions between the different genes. As these represent the dependence of the odds of certain genes being mutated depending on the mutational status of other genes, while the effect of an individual sample's mutational load is taken into account, we can interpret negative parameter values as a sign of the tendency for the two genes of not being mutated together in the cancerous cells.

Table 2 holds the first set of estimated parameters: the intercept terms, and the main effect terms of the logarithm of mutational load. These parameters are represented in the model (38) by θ_{ss0} and θ_{ss}, respectively, and they determine the individual occurrence rate of the mutations.

As can be expected, the signs and the magnitudes of the intercept parameters resemble the distribution of mutations in our data as seen from Figure 3. The positive terms in θ_{ss} correspond with our notion from Figure 5 that, in a sample with higher overall mutational frequency, also multiple PP2A mutations are more likely. However, it is interesting to see that this does not seem to apply for all of the

Gene name	θ_{ss0}	θ_{ss}
PPP2CA	-2.77	-0.14
PPP2CB	-3.32	0.00
PPP2R1A	1.08	-0.11
PPP2R1B	-1.68	-0.10
PPP2R2A_ENST00000380737	-3.32	0.17
PPP2R2B	-0.24	-0.08
PPP2R2C	-4.45	0.46
PPP2R2D_ENST00000455566	-3.32	0.17
PPP2R3A	0.90	-0.14
PPP2R3B	-9.54	1.13
PPP2R3C	-3.32	0.22
PPP2R4	-2.03	0.13
PPP2R5A	-3.79	0.26
PPP2R5B	-0.83	0.00
PPP2R5C	-4.53	0.41
PPP2R5D	-0.26	-0.13
PPP2R5E	-4.88	0.37
STRN	-1.55	0.10
STRN3	-1.85	0.14
STRN4	-2.60	0.21

Table 2: Parameters of individual occurrence of mutations

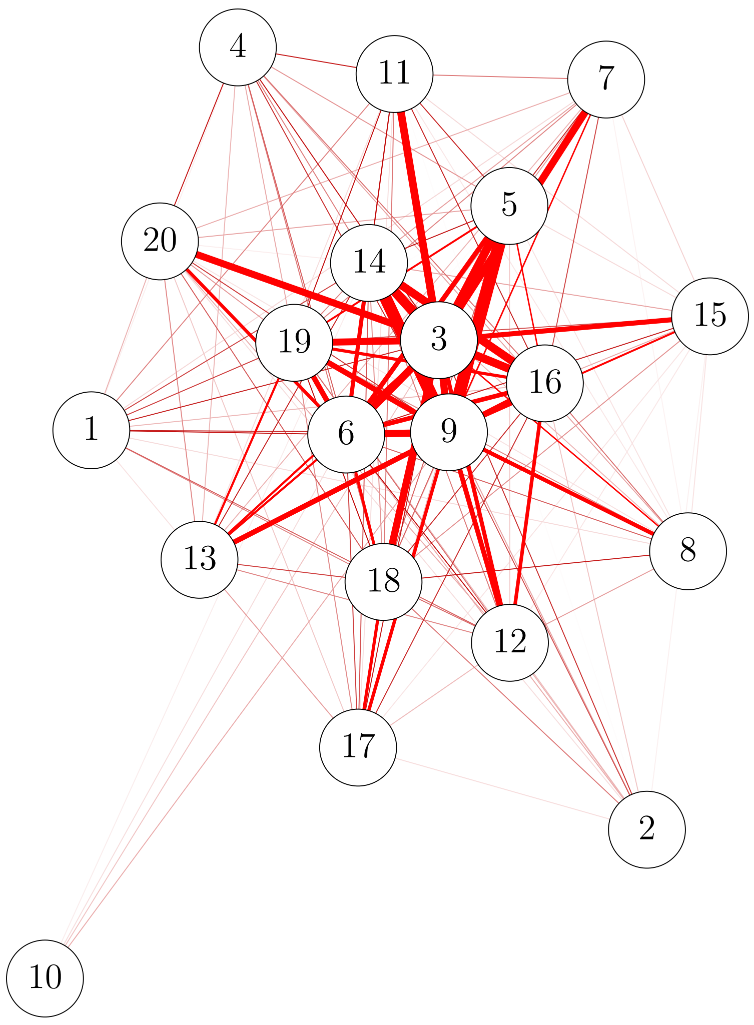
genes in our set. For example, a mutation in the gene PPP2R1A seems to be less likely in the samples with higher mutational load. Also, the mutation in PPP2CB, which is the second rarest mutation in our set, has its parameter regularized to zero. Thus, according to our model the overall mutational rate does not have an individual effect on the likeliness of a mutation in the PPP2CB gene.

In Figure 7 we have the graphical representation of the estimated interaction parameters $\hat{\theta}_{st0}$. The estimates can be found in numeric form in the matrix in Appendix E. The edges drawn between different genes correspond to the non-zero $\hat{\theta}_{st0}$ estimates. We observe that the genes form a rather complex network of interactions. PPP2R3B and PPP2CB seem to be separated from the rest of the network. Furthest from the core, PPP2R3B is neighbours with just four other genes. PPP2R1A, PPP2R3A, PPP2R2B, and PPP2R5B stand out as the most influential genes forming a hub at the centre of the graph, each of the four genes having an effect on all of the other nineteen nodes. Having estimated such a complex network of dependencies between mutations in the twenty investigated genes our model suggests that the mutations in these genes are unlikely to occur independently. Thus, we can find support for our notion that the mutations in PP2A encoding genes contribute to the fitness of a cancer cell.

In the Appendix F we have the matrix of estimates for the parameters θ_{st}, which correspond to the effect of mutational load on the network of mutations. Figure 7 corresponds to the situation, where the variable ln(Mutational load) equals zero i.e. there is only a single mutation. In order to examine how the accumulation of a greater number of mutations affects our graph's appearance, we have visualized three additional graphical representations. From the COSMIC database we calculated quartiles for the distribution of the total amount of mutations to be q1: 152, median: 682, and q3: 3055. The estimated network structure in these quartiles of the total amount of mutations is pictured in the Figure 8.

As the total amount of mutations cumulates higher, the network of interactions changes. In Figure 8a we can find that, with 152 total mutations the graph has a greater number of edges compared to Figure 7. This is due to the interactions between some genes being present in the estimates $\hat{\theta}_{st}$ while being constrained to zero in the estimation of θ_{st0} parameters. However, the form of the graph in Figure 8a is rather similar to the form in the first graph in Figure 7. The key mutations PPP2R1A, PPP2R3A, PPP2R2B, and PPP2R5B continue to form the core of the

Figure 7: Graphical representation of estimates for θ_{st0} parameters



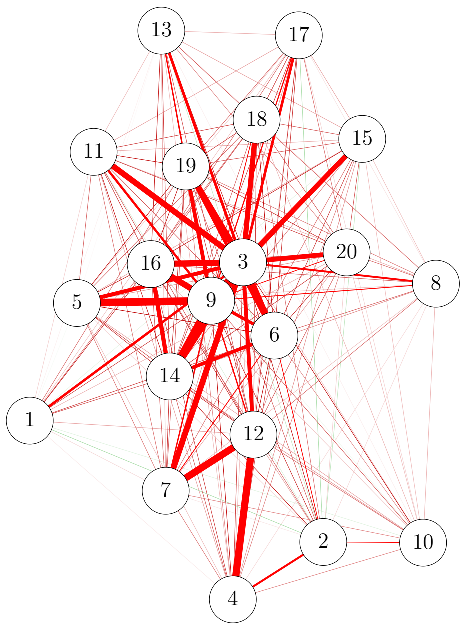
1:	PPP2CA
2:	PPP2CB
3:	PPP2R1A
4:	PPP2R1B
5:	PPP2R2A*
6:	PPP2R2B
7:	PPP2R2C
8:	PPP2R2D**
9:	PPP2R3A
10:	PPP2R3B
11:	PPP2R3C
12:	PPP2R4
13:	PPP2R5A
14:	PPP2R5B
15:	PPP2R5C
16:	PPP2R5D
17:	PPP2R5E
18:	STRN
19:	STRN3
20:	STRN4

**_ENST00000380737, **_ENST00000455566*

network. PPP2R3B has shifted from being a clear outlier towards the outer rim of the network.

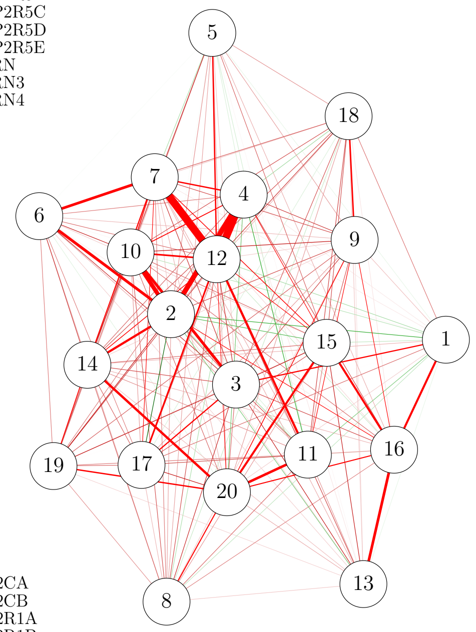
With 682 total mutations in Figure 8b, the graph has become more balanced in the relative strengths of interactions. The edges connecting (PPP2R4, PPP2R1B)

Figure 8: Interactions between mutations in PP2A encoding mutations at first quartile, median, and third quartile



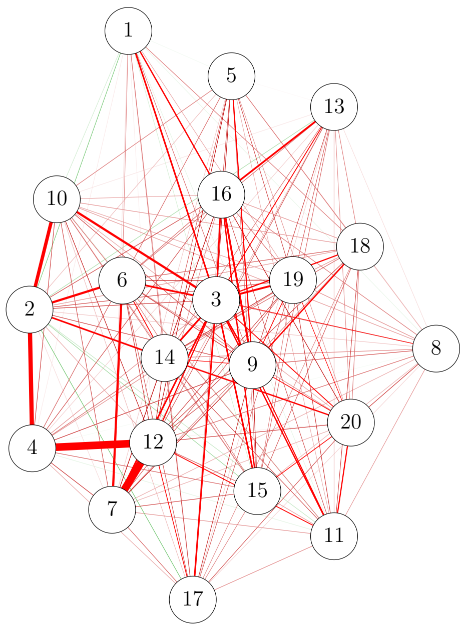
1: PPP2CA
2: PPP2CB
3: PPP2R1A
4: PPP2R1B
5: PPP2R2A*
6: PPP2R2B
7: PPP2R2C
8: PPP2R2D**
9: PPP2R3A
10: PPP2R3B
11: PPP2R3C
12: PPP2R4
13: PPP2R5A
14: PPP2R5B
15: PPP2R5C
16: PPP2R5D
17: PPP2R5E
18: STRN
19: STRN3
20: STRN4

(a) 152 mutations in total



1: PPP2CA
2: PPP2CB
3: PPP2R1A
4: PPP2R1B
5: PPP2R2A*
6: PPP2R2B
7: PPP2R2C
8: PPP2R2D**
9: PPP2R3A
10: PPP2R3B
11: PPP2R3C
12: PPP2R4
13: PPP2R5A
14: PPP2R5B
15: PPP2R5C
16: PPP2R5D
17: PPP2R5E
18: STRN
19: STRN3
20: STRN4

(c) 3055 mutations in total



1: PPP2CA
2: PPP2CB
3: PPP2R1A
4: PPP2R1B
5: PPP2R2A*
6: PPP2R2B
7: PPP2R2C
8: PPP2R2D**
9: PPP2R3A
10: PPP2R3B
11: PPP2R3C
12: PPP2R4
13: PPP2R5A
14: PPP2R5B
15: PPP2R5C
16: PPP2R5D
17: PPP2R5E
18: STRN
19: STRN3
20: STRN4

*_ENST00000380737, **_ENST00000455566*

(b) 682 mutations in total

and (PPP2R4, PPP2R2C) stand out as two stronger signals of negative bilateral effect between the probability of a mutation in the two genes. PPP2R1A continues to find its place at the core of the graph being surrounded by the previously mentioned PPP2R3A, PPP2R2B, and PPP2R5B.

Finally, with 3055 total mutations pictured in Figure 8c the interactions between mutations in the twenty investigated genes appear to be rather diluted compared to Figure 7. The previously identified key mutations PPP2R3A, PPP2R2B, and PPP2R5B no longer form the core of the network. However, the longest and most mutated gene PPP2R1A can still be found located at the centre of our network. The mutational status of the gene PPP2R4 seems to have moved towards the core, having a strong effect on the gene PPP2CB as well as the previously identified stronger interactions with PPP2R1B and PPP2R2C. It is also of interest that PPP2R3B, while identified as an outlier in the graphs with lower total mutational rates, is no longer on the outer rim of the network and holds a strong interaction with gene PPP2CB.

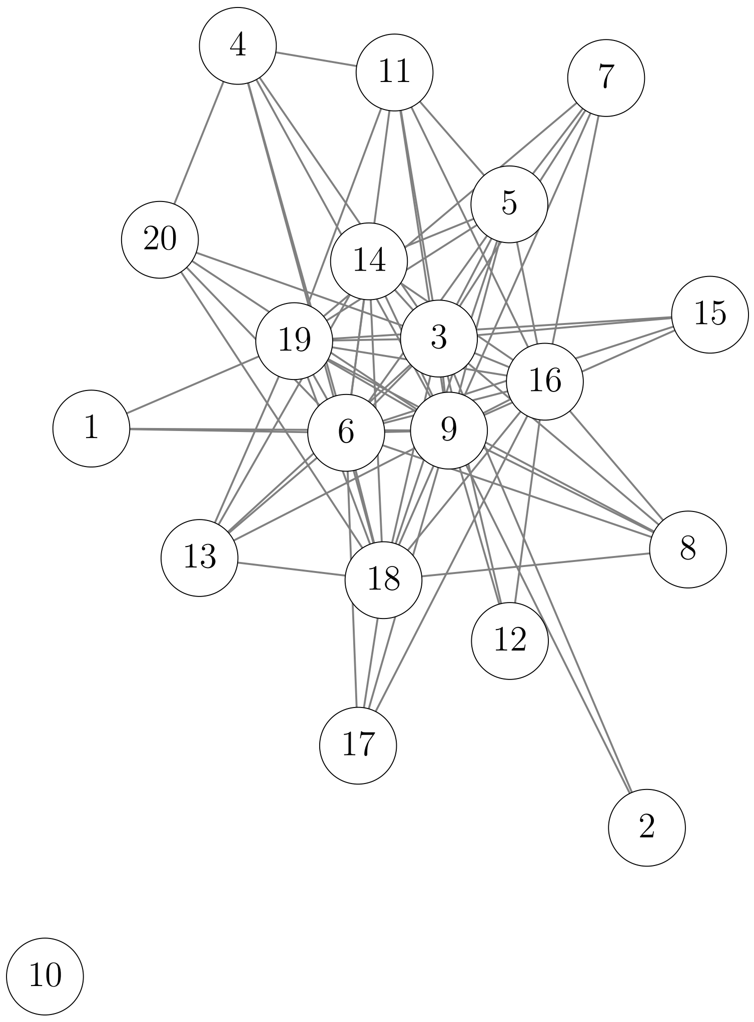
The networks at median and third quartile of total mutations include also some positive connections, which can be interpreted as a sign of co-occurrence between these mutations. That is, combinations such as (PPP2CA, PPP2R3B) or (PPP2CA, PPP2CB) are more likely to appear mutated together in samples with high total mutational rate.

5.3.3 Evaluation of the Graph Structure by Bootstrapping

In order to evaluate whether the estimates we obtained for parameters θ_{st0} presented by the edges in Figure 7 were due to random chance or actual properties of an underlying network of interactions between the mutations, we ran a bootstrapping scheme introduced in section 4.4. By running the analysis one thousand times on bootstrap samples of the original data we obtained the matrix of $\hat{P}_i j$ estimates presented in Appendix G. In Figure 9 we have drawn a graph corresponding to Figure 7 but with the original set of edges replaced by edges where $\hat{P}$ is less than 5%.

We have previously identified mutations in genes PPP2R1A, PPP2R3A, PPP2R2B, and PPP2R5B as key cancer mutations. According to our bootstrap analysis, the high number of interactions involving PPP2R1A, PPP2R3A, and PPP2R2B is reliable, each having 16, 18, and 15 interactions with $\hat{P}$ less than

Figure 9: $\hat{\theta}_{st0}$ edges with $\hat{P}$ less than 5%



1: PPP2CA
2: PPP2CB
3: PPP2R1A
4: PPP2R1B
5: PPP2R2A*
6: PPP2R2B
7: PPP2R2C
8: PPP2R2D**
9: PPP2R3A
10: PPP2R3B
11: PPP2R3C
12: PPP2R4
13: PPP2R5A
14: PPP2R5B
15: PPP2R5C
16: PPP2R5D
17: PPP2R5E
18: STRN
19: STRN3
20: STRN4

*_ENST00000380737, **_ENST00000455566*

5%, respectively. The gene PPP2R5B had only seven edges that appeared over 95% of times in the fits using bootstrapped samples.

Thus, our analysis suggests that PPP2R1A, PPP2R3A, and PPP2R2B can be considered as the genes in which occurring mutations have the most influence on the mutation of the other genes under analysis. For PPP2R5B, we cannot find evidence of having as profound effect as the nineteen edges in our $\hat{\theta}_{st0}$ estimate would suggest.

Gene PPP2R3B, which was an outlier in our graph in Figure 7, does not have an edge with $\hat{P}$ value of less than 5% with any other gene. Thus, bootstrapping supports our observation that PPP2R3B is not meaningful in the network of mutations in PP2A encoding genes.

6 Conclusions

The purpose of this study was to explore the network of interactions behind mutations in PP2A encoding genes by using a sparse binary Markov random field on a data set of mutations found in tumour biopsies. By revealing the underlying network of interactions we can find evidence of specific mutations acting as driver mutations. This notion is based on natural selection, where mutations that benefit the cancerous cells to grow are likely to help the cells outgrow competing cell populations and appear in the mutational profile of the investigated tumour. However, if multiple mutations provide similar benefits to the cancer cell, only one needs to happen for the cell to benefit. Thus, mutual exclusivity, or a certain gene being less likely to be found mutated together with another gene, can be used as evidence for both of the mutations benefiting the cancer in a similar fashion.

Through our analysis we found that there is a complex network of interactions behind mutations in genes encoding the PP2A enzyme in humans. Thus, we can say that the probability of a gene being found to be mutated in a tumour sample is not dependent only on the individual mutational tendency of each gene and the overall mutational rate of the investigated tumour sample, but in most cases also depending on the mutational status of the other nineteen PP2A encoding genes. We identified the genes PPP2R1A, PPP2R3A, and PPP2R2B as forming the core of the network of mutations. PPP2R4 was also identified as a possibly being an influential mutation in highly mutated samples.

Our aim was to provide a subset of genes in the group encoding PP2A enzyme, which could be used as a prime candidate for further biological research on defining the tumour suppressor functions of PP2A. It is of interest to note that the suppression of the four genes that stand out in our analysis have been shown to contribute to the transformation of normal human cells. For PPP2R1A see Chen et al. (2005), for PPP2R2B see Tan et al. (2010), and for PPP2R3A and PPP2R4 see Sablina et al. (2010).

Although multiple lines of evidence suggest that PP2A is a tumour suppressor serving as a regulator of cell growth, proliferation, and survival, the genomics data to support this claim is limited. This thesis provides novel genomic evidence by revealing the interdependencies of the mutations in different subunits, which suggests that the mutations occurring in PP2A genes contribute to cancer cell fitness and further highlight restoring the PP2A activity as an appealing therapeutic strategy.

References

Bach, Francis. *Bolasso: Model Consistent Lasso Estimation Through the Bootstrap.* Proceedings of the 25th International Conference on Machine Learning, 33-40, 2008.

Besag, Julian. *Statistical Analysis of Non-lattice Data.* The Statistician, Vol. 24(3), 179-195, 1975.

Chen, Wen; Arroyo, Jason D.; Timmons, Jamie C.; Possemato, Richard & Hahn, William C. *Cancer-Associated PP2A Aα Subunits Induce Functional Haploinsufficiency and Tumorigenicity.* Cancer Res., Vol. 65(18), 8183-8192 , 2005.

Cheng, Jie; Levina, Elizaveta & Wang, Pei. *A Sparse Ising Model with Covariates.* Biometrics, Vol. 70, 943–953, 2014.

Ciriello, Giovanni; Cerami, Ethan; Sander, Chris & Schultz, Nikolaus. *Mutual exclusivity analysis identifies oncogenic network modules.* Genome Res., Vol. 22, 398–406, 2012.

The COSMIC database cancer.sanger.ac.uk

Efron, Bradley & Tibshirani, Robert. *An Introduction to the Bootstrap.* Chapman & Hall, 1998.

Epskamp, Sacha; Cramer, Angelique O. J.; Waldorp, Lourens J.; Schmittmann, Verena D. & Borsboom, Denny. *qgraph: Network Visualizations of Relationships in Psychometric Data.* Journal of Statistical Software, Vol. 48(4), 1-18. `http://www.jstatsoft.org/v48/i04/`, 2012.

Forbes S.A.; Beare D.; Gunasekaran P.; Leung K.; Bindal N.; Boutselakis H.; Ding M.; Bamford S.; Cole C. & Ward S. et al. *COSMIC: exploring the world's knowledge of somatic mutations in human cancer.* Nucleic Acids Res., Vol. 43, D805-D811, 2014.

Friedman, Jerome; Hastie, Trevor & Tibshirani, Robert. *Sparse inverse covariance estimation with the graphical lasso* Biostatistics, Vol. 9, 432-441, 2008.

Goeman, Jelle; Meijer, Rosa & Chaturvedi, Nimisha. *penalized: L1 (lasso and fused lasso) and L2 (ridge) penalized estimation in GLMs and in the Cox model.* R package version 0.9-45 `http://CRAN.R-project.org/package=penalized`, 2014.

Hanahan, Douglas & Robert A. Weinberg. *The Hallmarks of Cancer.* Cell, Vol. 100, 57–70, 2000.

Hanahan, Douglas & Robert A. Weinberg. *The Hallmarks of Cancer: The Next Generation.* Cell, Vol. 144, 646–674, 2011.

Hastie, Trevor; Tibshirani, Robert & Friedman, Jerome. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* Springer, 2009.

Höfling, Holger & Tibshirani, Robert. *Estimation of Sparse Binary Pairwise Markov Networks using Pseudo-likelihoods.* Journal of Machine Learning Research, Vol. 10, 883-906, 2009.

Ising, Ernst. *Beitrag zur Theorie des Ferromagnetismus*, Z. Phys., Vol. 31, 253–258, 1925

Koller, Daphne & Friedman, Nir *Probabilistic Graphical Models: Principles and Techniques.* The MIT Press, 2009.

Kuhn, Max. Contributions from Jed Wing, Steve Weston, Andre Williams, Chris Keefer, Allan Engelhardt, Tony Cooper, Zachary Mayer, Brenton Kenkel, the R Core Team, Michael Benesty, Reynald Lescarbeau, Andrew Ziem and Luca Scrucca. *caret: Classification and Regression Training.* R package version 6.0-52 `http://CRAN.R-project.org/package=caret`, 2015.

Lauritzen, Steffen. *Graphical Models.* Oxford Science Publications, 1996.

Lawrence, Michael S.; Stojanov, Petar; Polak, Paz et al. *Mutational heterogeneity in cancer and the search for new cancer-associated genes.* Nature, Vol. 499, 214-218, 2013.

Lee, Su-In; Ganapahthi, Varun & Koller, Daphne. *Efficient Structure Learning of Markov Networks using L1-Regularization.* Advances in Neural Information Processing Systems, Vol. 19, 817–824, 2007.

Meinshausen, Nicolai & Bühlmann, Peter. *High-dimensional graphs and variable selection with the lasso.* Annals of Statistics, Vol. 34, 1436-1462, 2006.

R Core Team (2015). *R: A language and environment for statistical computing.* R Foundation for Statistical Computing, Vienna, Austria. `http://www.R-project.org/`.

Sablina, Anna A.; Hector, Melissa; Colpaert, Nathalie & Hahn, William C. *Identification of PP2A Complexes and Pathways Involved in Cell Transformation.* Cancer Res., Vol. 70(24), 10474–10484, 2010.

Szczurek, Ewa & Beerenwinkel, Niko. *Modeling Mutual Exclusivity of Cancer Mutations.* PLOS Computational Biology, Vol. 10(3), 2014.

Tan, Jing; Lee, Puay Leng; Li, Zhimei; Jiang, Xia; Lim, Yaw Chyn; Hooi, Shing Chuan & Yu, Qiang. *B55β-Associated PP2A Complex Controls PDK1-Directed Myc Signaling and Modulates Rapamycin Sensitivity in Colorectal Cancer.* Cancer Cell, Vol. 18(5), 459–471, 2010.

Vandin, Fabio; Upfal, Eli & Raphael, Benjamin J. *De novo discovery of mutated driver pathways in cancer.* Genome Res., Vol. 22, 375–385, 2012.

Wainwright, Martin J.; Ravikumar, Pradeep & Lafferty, John D. *High-Dimensional Graphical Model Selection Using l1-Regularized Logistic Regression.* Advances in Neural Information Processing Systems, 2006.

Wainwright, Marting J.; Ravikumar, Pradeep & Lafferty, John D. *High-dimensional graphical model selection using L1-regularized logistic regression. Technical report*, University of California, Berkeley, 2008.

Wickham, Hadley. *ggplot2: elegant graphics for data analysis.* Springer New York, 2009.

Appendices

A BinaryMRFwithCov

```
BinaryMRFwithCov <-
  function (data, lambda = 1, separate_min = FALSE,
            bs_run = FALSE, last_graph_param = ncol(data))  {
    #
    # Estimate sparse Markov Random Field structure given
    # a set of binary data and additional covariates
    #
    # Author: Oscar Lindberg

    # Args:
    #   data: The input data in a data.frame, where the
    #         p leftmost variables are the binary variables
    #         to be represented by nodes in the final graph
    #   lambda: A single numeric value,
    #         to be used as l1-regularization parameter
    #   separate_min: If TRUE, use the minimum of corresponding
    #         parameter estimates as the estimated
    #         value in the final symmetric post processed matrix
    #         of estimates. Else use maximum. Default is FALSE.
    #   bs_run: If TRUE, only returns the graph matrix to
    #         conserve memory when the function is ran
    #         multiple times due to bootstrapping.
    #         Else returns as described below. Default is FALSE
    #   last_graph_param: A single integer, index of last
    #         column in data which is represented by
    #         nodes in the final graph. Columns with index
    #         greater than last_graph_param are taken
    #         as additional parameters. Default is number of
    #         columns in data, corresponding to no
    #         additional covariates.
    #
    # Returns:
    #   A list containing:
    #   graph: Estimated parameter matrix of edges
    #   treshold: Estimated parameter vector of intercepts
    #   results: List of results of individual logistic
    #         regressions with LASSO penalty
    #   addit.param: Estimated parameter matrix for
    #         additional covariates effects on edges
    #   addit.p.treshold: Estimated parameter vector for
    #         additional covariates effects on intercepts
    #
    # Dependencies:
    #   Package: penalized, version 0.9-45
    #     by Jelle Goeman, Rosa Meijer and Nimisha Chaturvedi
    #   Package: parallel, version 3.2.0
    #     by R Core Team
    #

    # node variable names
    c_r_names <-
      colnames(data[,1:last_graph_param])
    # additional variable names
    c_a_names <-
      c()
    # amount of node variables,
    # each will act as response variable in regressions
    p <-
      last_graph_param
    # amount of addit. variables
    q <-
      ncol(data) - last_graph_param
```

```
# Transform data by adding new columns
# from multiplication of binary node data
# with the external variables
data1 <-
  data[,1:last_graph_param]
data_a <-
  data

# if external variables exist then do transformation
if (last_graph_param < ncol(data)) {
  # loop through the external variables
  for (i in (last_graph_param + 1):ncol(data)) {
    # multiply node variables by this external var
    data_tmp <- data1 * data[,i]
    # name new vars by external variable with suffix
    colnames(data_tmp) <-
      paste(colnames(data)[i], colnames(data1), sep = "_")
    c_a_names <- c(c_a_names, colnames(data_tmp))
    # join to common matrix with node variables
    data_a <-
      cbind(data_a, data_tmp)
  }
}

# Use function penalized from package
# penalized for separate logistic regressions with
# LASSO penalty for each parameter except intercept
Res <-
  mclapply(seq_len(p), function(i)
    penalized(
      response = data_a[,i],
      penalized = data_a[,-c(i, ncol(data) + i)],
      lambda1 = lambda, lambda2 = 0, steps = 1,
      model = "logistic",
      standardize = FALSE
    ))
Coefs <- lapply(Res, coef, "all")
addit_cov <- NULL
if (last_graph_param < ncol(data)) {
  addit_cov <-
    lapply(Coefs, function(i) {
      i[(last_graph_param + 1):length(i)]
    }) # parameters for additional covariates
  Coefs <-
    lapply(Coefs, function(i) {
      head(i, last_graph_param)
    }) # Theta without parameters for additional covariates
}

# Symmetrize coefficient matrices
Net <- matrix(0, p, p)
for (i in seq_len(p)) {
  Net[i,-i] <- Coefs[[i]][-1]
  Net[i, i] <- Coefs[[i]][1]
}

if(q > 0){
  Net2 <- lapply(1:q, function(add.param){
    mat <- matrix(0, p, p)
    for (i in seq_len(p)) {
      mat[i,-i] <-
        addit_cov[[i]][(q+(add.param-1)*p+1):(q+(add.param-1)*p+p-1)]
      mat[i, i] <-
        addit_cov[[i]][add.param]
    }
    mat
  }
  )
}
# Symmetrize matrix by separate-min or separate-max
```

```
  symmetr <- function(Net) {
    Net.upper <- Net[upper.tri(Net)]
    Net.lower <- t(Net)[upper.tri(Net)]
    if (separate_min) {
      Net.upper.new <-
        ifelse(abs(Net.upper) < abs(Net.lower),
               Net.upper, Net.lower)
    } else {
      Net.upper.new <-
        ifelse(abs(Net.upper) > abs(Net.lower),
               Net.upper, Net.lower)
    }
    Net.sym <- matrix(0, p, p)
    treshold <- diag(Net)
    Net.sym[upper.tri(Net.sym)] <- Net.upper.new
    Net.sym <- t(Net.sym)
    Net.sym[upper.tri(Net.sym)] <- Net.upper.new
    list(Net.sym, treshold)
  }
  Net.sym <- symmetr(Net)
  dimnames(Net.sym[[1]])[[1]] <-
    dimnames(Net.sym[[1]])[[2]] <- c_r_names
  if (q > 0){
    Net2.sym <- lapply(1:length(Net2),
                       function(x){
                         add.par.mat <- Net2[[x]]
                         sym <- symmetr(add.par.mat)
                         dimnames(sym[[1]])[[1]] <-
                           dimnames(sym[[1]])[[2]] <-
                           c_a_names[((x-1)*p+1):((x-1)*p+p)]
                         sym
                       }
    )
  } else {
    Net2.sym = list()
  }
  # For bootstrapping: return only graph matrix to
  # conserve memory
  # Else return a list as described in program header
  if (bs_run) {
    return(list(graph = Net.sym[[1]]))
  } else {
    return(
      list(
        graph = Net.sym[[1]],
        treshold = Net.sym[[2]],
        results = Res,
        addit.param = Net2.sym,
        data_a = data_a
      )
    )
  }
}
```

B BootstrapEdges

```
BootstrapEdges <- function(data,
                           bs_sample_n,
                           sample_seed,
                           lambda,
                           separate_min = FALSE,
                           last_graph_param) {
  #
  # Calculate matrix rho by applying BinaryMRFwithCov
  # function on bootstrapped samples of the original
  # data and calculating the proportion of zero estimates
  # for each edge parameter
  #
  # Author: Oscar Lindberg

  # Args:
  #   data: The input data in a data.frame, where the
  #         p leftmost variables are the binary variables
  #         to be represented by nodes in the final graph
  #   bs_sample_n: A single integer value, will be used as
  #         the total amount of bootstrap samples
  #   sample_seed: A single numeric value, will be used as
  #         the seed value for bs sampling
  #   lambda: A single numeric value,
  #         to be used as l1-regularization parameter
  #   separate_min: If TRUE, use the minimum of corresponding
  #         parameter estimates as the estimated
  #         value in the final symmetric post processed matrix
  #         of estimates. Else use maximum. Default is FALSE.
  #   last_graph_param: A single integer, index of last
  #         column in data which is represented by
  #         nodes in the final graph. Columns with index
  #         greater than last_graph_param are taken
  #         as additional parameters. Default is number of
  #         columns in data, corresponding to no
  #         additional covariates.
  #
  # Returns:
  #   A data.frame of same size as data containing
  #   the proportion of times the corresponding
  #   edge parameter is estimated as zero
  #
  # Dependencies:
  #   Function: BinaryMRFwithCov
  #

  bs_samp <- vector("list", bs_sample_n)
  if (is.matrix(data)) {
    data <- as.data.frame(data)
  }
  # bootstrap sample from data
  set.seed(sample_seed)
  sample_data <- function(empty) {
    sample_n(data, nrow(data), TRUE)
  }

  bs_samp <- lapply(bs_samp, sample_data)

  bs_estimates <- lapply(bs_samp,
                         BinaryMRFwithCov,
                         lambda,
                         separate_min,
                         last_graph_param,
                         bs_run = TRUE)

  bs_estimates <- lapply(bs_estimates, getElement, 'graph')

  # how many zero estimates
  count_zero <- function(x, y) {
    estimates_in_xy <-
      unlist(lapply(bs_estimates, `[`, x, y))
    zero_n <- sum(equals(estimates_in_xy, 0))
    # corrected for finite sampling
    (1 + zero_n)/(1 + bs_sample_n)
  }

  n <- nrow(bs_estimates[[1]])
  bs_matrix <- matrix(0, n, n)
  for (i in seq_len(n-1)) {
    for(j in seq(i+1, n)) {
      bs_matrix[i, j] <- count_zero(i, j)
    }
  }

  bs_matrix[lower.tri(bs_matrix)] <-
    t(bs_matrix)[lower.tri(bs_matrix)]

  rownames(bs_matrix) <-
    rownames(bs_estimates[[1]])
  colnames(bs_matrix) <-
    colnames(bs_estimates[[1]])

  return(as.data.frame(bs_matrix))
}
```

C PredictBMRF

```
PredictBMRF <- function(data,
                        estim,
                        last_graph_param = ncol(data)) {
  #
  # Calculate the linear predictor for a node variable
  # from estimated model using the other node variables
  # and additional covariates
  #
  # Author: Oscar Lindberg

  # Args:
  #   data: The input data in a data.frame, where the
  #         p leftmost variables are the binary variables
  #         to be represented by nodes in the final graph
  #   estim: A list, generated by function BinaryRMFwithCov
  #         contained the estimated solution
  #   last_graph_param: A single integer, index of last
  #         column in data which is represented by
  #         nodes in the final graph. Columns with index
  #         greater than last_graph_param are taken
  #         as additional parameters. Default is number of
  #         columns in data, corresponding to no
  #         additional covariates.
  #
  # Returns:
  #   A list containing:
  #   pred.values: Data.frame containing the predicted
  #         values for node variable observations in data
  #   lin.pred: Data.frame containing the linear predictor
  #         values for node variable observations in data

  # node variable names
  c_r_names <-
    colnames(data[,1:last_graph_param])
  # additional variable names
  c_a_names <-
    c()
  # amount of node variables,
  # each will act as a variable to be
  # predicted
  p <-
    last_graph_param
  # amount of addit. variables
  q <-
    ncol(data) - last_graph_param

  # Transform data by adding new columns
  # from multiplication of binary node data
  # with the external variables
  data1 <-
    data[,1:last_graph_param]
  data_a <-
    data

  # if external variables exist then do transformation
  if (last_graph_param < ncol(data)) {
    # loop through the external variables
    for (i in (last_graph_param + 1):ncol(data)) {
      # multiply node variables by this external var
      data_tmp <- data1 * data[,i]
      # name new vars by external variable with suffix
      colnames(data_tmp) <-
        paste(colnames(data)[i], colnames(data1), sep = "_")
      c_a_names <- c(c_a_names, colnames(data_tmp))
      # join to common matrix with node variables
      data_a <-
        cbind(data_a, data_tmp)
    }
  }

  pred_var <- function(i) {
    int <- estim$treshold[i]
    if (last_graph_param < ncol(data)){
      int.add <-
        apply(unlist(lapply(
          lapply(estim$addit.param, '[[', 2), '[', i
        )) *
          data[-c(1:last_graph_param)], 1, sum)
    }
    edg1 <- matrix(nrow = nrow(data1[,-i]),
                   ncol = ncol(data1[,-i]))
    for (m in 1:ncol(edg1)) {
      edg1[,m] <- (estim$graph[i,-i][m] * data1[,-i][m])[,1]
    }
    edg2 <- rowSums(edg1)
    if (last_graph_param < ncol(data)){
      add.edg.1 <-
        lapply(lapply(lapply(estim$addit.param,'[[', 1),'[', i,-i),
               function(x) {
                 rbind(x)[rep(1, nrow(data)),]
               })
      add.edg.2 <- lapply(1:length(add.edg.1),
                          function(k) {
                            x <- add.edg.1[[k]]
                            for (j in 1:ncol(x)) {
                              x[,j] <-
                                x[,j] *
                                data[,-c(1:last_graph_param),
                                     drop = FALSE][,k]
                            }
                            x
                          })
      add.edg.3 <-
        matrix(1, ncol = ncol(add.edg.2[[1]]),
               nrow = nrow(add.edg.2[[1]]))
      lapply(add.edg.2, function(x) {
        add.edg.3 <<- add.edg.3 * x
      })
      add.edg.4 <- add.edg.3 * data1[,-i]
    }
    if (last_graph_param < ncol(data)){
      rowSums(data.frame(int, int.add, edg2, add.edg.4))
    } else if (last_graph_param == ncol(data)){
      rowSums(data.frame(int, edg2))
    }
  }
  output <- NULL
  for (node.var in 1:length(c_r_names)) {
    output <- cbind(output, pred_var(node.var))
    colnames(output)[node.var] <- c_r_names[node.var]
  }
  list(pred.values = as.data.frame(ifelse(output <= 0, 0, 1)),
       lin.pred = as.data.frame(output))
}
```

D CvBMRF

```
CvBMRF <- function(data,
                   low.bound,
                   high.bound,
                   by,
                   separate_min = FALSE,
                   last_graph_param = ncol(data)) {
  #
  # Calculate the mean positive predictive value and mean
  # negative predictive value from 5 fold cross-validation
  # for optimising the regularization parameter
  #
  # Author: Oscar Lindberg

  # Args:
  #   data: The input data in a data.frame, where the
  #         p leftmost variables are the binary variables
  #         to be represented by nodes in the final graph
  #   low.bound: Numeric, lowest value tested for lambda
  #   high.bound: Numeric, highest value tested for lambda
  #   by: Numeric, increment of the sequence
  #   separate_min: If TRUE, use the minimum of corresponding
  #         parameter estimates as the estimated
  #         value in the final symmetric post processed matrix
  #         of estimates. Else use maximum. Default is FALSE.
  #   last_graph_param: A single integer, index of last
  #         column in data which is represented by
  #         nodes in the final graph. Columns with index
  #         greater than last_graph_param are taken
  #         as additional parameters. Default is number of
  #         columns in data, corresponding to no
  #         additional covariates.
  #
  # Returns:
  #   A list containing:
  #   mean.pos.pred: Numeric value of average PPV
  #   mean.neg.pred: Numeric value of average NPV
  #   pos.pred: Vector of PPV
  #   neg.pred: Vector of NPV
  #
  # Dependencies:
  #   Function: createFolds, from package caret, Version 6.0-52
  #     by Max Kuhn


  folds <- createFolds(data[,1], 5)

  lambda.seq <- seq(low.bound, high.bound, by)

  cv <- lapply(lambda.seq, function(l) {
    pos.pred <- c()
    neg.pred <- c()
    for (k in 1:length(folds)) {
      training <- data[-folds[[k]],]
      test <- data[folds[[k]],]

      MRFfit <-
        BinaryMRFwithCov(
          data = training,
          lambda = l,
          separate_min = separate_min,
          bs_run = FALSE,
          last_graph_param = last_graph_param)

      MRFpred <- PredictBMRF(
        data = test,
        estim = MRFfit,
        last_graph_param = last_graph_param)
      true.pos <-
        fals.pos <-
        true.neg <-
        fals.neg <-
        matrix(NA, ncol = ncol(MRFpred[[1]]),
               nrow = nrow(MRFpred[[1]]))
      for (i in 1:nrow(true.pos)) {
        for (j in 1:ncol(true.pos)) {
          true.pos[i, j] <-
            isTRUE(all.equal(MRFpred[[1]][i,j], test[i,j])) &
            isTRUE(all.equal(MRFpred[[1]][i,j], 1))
          fals.pos[i, j] <-
            !isTRUE(all.equal(MRFpred[[1]][i,j], test[i,j])) &
            isTRUE(all.equal(MRFpred[[1]][i,j], 1))
          true.neg[i, j] <-
            isTRUE(all.equal(MRFpred[[1]][i,j], test[i,j])) &
            isTRUE(all.equal(MRFpred[[1]][i,j], 0))
          fals.neg[i, j] <-
            !isTRUE(all.equal(MRFpred[[1]][i,j], test[i,j])) &
            isTRUE(all.equal(MRFpred[[1]][i,j], 0))
        }
      }
      pos.pred <- c(pos.pred,
      sum(true.pos, na.rm = TRUE) /
        (sum(true.pos, na.rm = TRUE) + sum(fals.pos, na.rm = TRUE)))
      neg.pred <- c(neg.pred,
      sum(true.neg, na.rm = TRUE) /
        (sum(true.neg, na.rm = TRUE) + sum(fals.neg, na.rm = TRUE)))
    }
    list(mean.pos.pred = mean(pos.pred, na.rm = TRUE),
         mean.neg.pred = mean(neg.pred, na.rm = TRUE),
         pos.pred = pos.pred, neg.pred = neg.pred)
  }
)
cv
}
```

E Estimated Graph Weight Matrix

	PPP2CA	PPP2CB	PPP2R1A	PPP2R1B	PPP2R2A*	PPP2R2B	PPP2R2C	PPP2R2D**	PPP2R3A	PPP2R3B	PPP2R3C	PPP2R4	PPP2R5A	PPP2R5B	PPP2R5C	PPP2R5D	PPP2R5E	STRN	STRN3	STRN4
PPP2CA	0.00	0.00	-4.59	-0.33	-0.03	-4.88	-0.82	-0.74	-4.52	0.00	-2.58	-3.28	-0.59	-3.87	0.00	-1.45	0.00	-2.45	-3.97	-1.59
PPP2CB	0.00	0.00	-4.32	0.00	0.00	-1.72	0.00	0.00	-3.61	0.00	-0.07	-1.57	-0.01	-1.91	-0.30	-1.21	-0.68	-2.82	-1.37	-0.70
PPP2R1A	-4.59	-4.32	0.00	-4.95	-8.44	-8.16	-7.96	-5.44	-7.98	-1.22	-8.21	-6.54	-5.76	-9.04	-7.01	-8.33	-4.73	-7.90	-7.83	-7.79
PPP2R1B	-0.33	0.00	-4.95	0.00	-2.21	-3.33	0.00	-1.91	-4.28	0.00	-4.78	0.00	-1.50	-2.65	0.00	-2.98	0.00	-2.74	-1.75	-4.87
PPP2R2A*	-0.03	0.00	-8.44	-2.21	0.00	-6.92	-4.59	-0.60	-10.28	0.00	-4.56	-1.08	-0.05	-4.83	-0.77	-5.39	-1.40	-3.25	-5.61	-1.84
PPP2R2B	-4.88	-1.72	-8.16	-3.33	-6.92	0.00	-2.01	-3.18	-7.89	-0.70	-3.91	-5.04	-5.69	-6.58	-4.71	-6.75	-3.97	-6.06	-7.13	-6.28
PPP2R2C	-0.82	0.00	-7.96	0.00	-4.59	-2.01	0.00	-0.27	-5.42	0.00	-2.75	0.00	-1.55	-2.02	-0.93	-3.85	0.00	-1.11	-2.56	-1.71
PPP2R2D**	-0.74	0.00	-5.44	-1.91	-0.60	-3.18	-0.27	0.00	-6.55	0.00	-0.49	-1.99	0.00	-1.49	-0.60	-3.00	0.00	-4.71	-2.41	-0.56
PPP2R3A	-4.52	-3.61	-7.98	-4.28	-10.28	-7.89	-5.42	-6.55	0.00	-1.74	-5.12	-6.86	-7.06	-10.14	-5.58	-7.44	-6.21	-3.17	-7.22	-4.09
PPP2R3B	0.00	0.00	-1.22	0.00	0.00	-0.70	0.00	0.00	-1.74	0.00	0.00	0.00	0.00	-0.37	0.00	0.00	0.00	0.00	0.00	0.00
PPP2R3C	-2.58	-0.07	-8.21	-4.78	-4.56	-3.91	-2.75	-0.49	-5.12	0.00	0.00	0.00	-0.69	-5.02	-0.79	-4.54	-0.62	-2.78	-4.98	0.00
PPP2R4	-3.28	-1.57	-6.54	0.00	-1.08	-5.04	0.00	-1.99	-6.86	0.00	0.00	0.00	-2.48	-2.75	-0.49	-6.38	-1.41	-3.06	-3.05	-1.70
PPP2R5A	-0.59	-0.01	-5.76	-1.50	-0.05	-5.69	-1.55	0.00	-7.06	0.00	-0.69	-2.48	0.00	-4.59	0.00	-0.90	-2.03	-3.91	-5.67	-2.45
PPP2R5B	-3.87	-1.91	-9.04	-2.65	-4.83	-6.58	-2.02	-1.49	-10.14	-0.37	-5.02	-2.75	-4.59	0.00	-1.90	-8.40	-3.58	-2.25	-2.50	-0.34
PPP2R5C	0.00	-0.30	-7.01	0.00	-0.77	-4.71	-0.93	-0.60	-5.58	0.00	-0.79	-0.49	0.00	-1.90	0.00	-1.77	-0.54	-2.26	-3.66	0.00
PPP2R5D	-1.45	-1.21	-8.33	-2.98	-5.39	-6.75	-3.85	-3.00	-7.44	0.00	-4.54	-6.38	-0.90	-8.40	-1.77	0.00	-4.40	-4.41	-6.13	-1.75
PPP2R5E	0.00	-0.68	-4.73	0.00	-1.40	-3.97	0.00	0.00	-6.21	0.00	-0.62	-1.41	-2.03	-3.58	-0.54	-4.40	0.00	-6.09	-2.41	-1.12
STRN	-2.45	-2.82	-7.90	-2.74	-3.25	-6.06	-1.11	-4.71	-3.17	0.00	-2.78	-3.06	-3.91	-2.25	-2.26	-4.41	-6.09	0.00	-5.10	-3.40
STRN3	-3.97	-1.37	-7.83	-1.75	-5.61	-7.13	-2.56	-2.41	-7.22	0.00	-4.98	-3.05	-5.67	-2.50	-3.66	-6.13	-2.41	-5.10	0.00	-2.16
STRN4	-1.59	-0.70	-7.79	-4.87	-1.84	-6.28	-1.71	-0.56	-4.09	0.00	0.00	-1.70	-2.45	-0.34	0.00	-1.75	-1.12	-3.40	-2.16	0.00

*_ENST00000380737, **_ENST00000455566

F Estimated Effect of Mutational Frequency Matrix

	PPP2CA	PPP2CB	PPP2R1A	PPP2R1B	PPP2R2A*	PPP2R2B	PPP2R2C	PPP2R2D**	PPP2R3A	PPP2R3B	PPP2R3C	PPP2R4	PPP2R5A	PPP2R5B	PPP2R5C	PPP2R5D	PPP2R5E	STRN	STRN3	STRN4
PPP2CA	0.00	0.17	0.31	0.03	0.02	0.60	0.08	0.14	0.51	0.06	0.43	0.48	0.09	0.42	0.05	-0.15	-0.12	0.19	0.49	0.16
PPP2CB	0.17	0.00	0.39	-0.59	-0.09	-0.18	-0.04	-0.08	0.30	-0.52	0.09	0.12	0.11	-0.11	0.18	-0.06	0.29	0.29	-0.03	0.05
PPP2R1A	0.31	0.39	0.00	0.61	1.01	0.84	0.82	0.50	0.84	-0.25	0.87	0.63	0.53	0.98	0.68	0.89	0.33	0.84	0.78	0.84
PPP2R1B	0.03	-0.59	0.61	0.00	0.30	0.31	-0.28	0.30	0.37	-0.24	0.77	-0.82	0.17	0.20	-0.21	0.22	-0.12	0.42	0.11	0.73
PPP2R2A*	0.02	-0.09	1.01	0.30	0.00	0.87	0.64	0.00	1.20	-0.18	0.60	-0.14	0.00	0.55	-0.03	0.61	0.18	0.32	0.64	0.16
PPP2R2B	0.60	-0.18	0.84	0.31	0.87	0.00	-0.13	0.33	0.93	-0.10	0.42	0.51	0.67	0.66	0.55	0.83	0.39	0.70	0.78	0.81
PPP2R2C	0.08	-0.04	0.82	-0.28	0.64	-0.13	0.00	-0.07	0.53	-0.23	0.28	-0.82	0.13	0.00	-0.10	0.45	-0.21	0.00	0.18	0.15
PPP2R2D**	0.14	-0.08	0.50	0.30	0.00	0.33	-0.07	0.00	0.77	-0.14	-0.06	0.10	-0.04	0.06	-0.01	0.27	-0.06	0.63	0.21	-0.18
PPP2R3A	0.51	0.30	0.84	0.37	1.20	0.93	0.53	0.77	0.00	0.01	0.42	0.82	0.85	1.23	0.56	0.72	0.69	0.11	0.77	0.29
PPP2R3B	0.06	-0.52	-0.25	-0.24	-0.18	-0.10	-0.23	-0.14	0.01	0.00	0.03	-0.29	-0.03	-0.14	-0.12	-0.12	-0.08	-0.14	-0.12	-0.12
PPP2R3C	0.43	0.09	0.87	0.77	0.60	0.42	0.28	-0.06	0.42	0.03	0.00	-0.35	0.00	0.54	-0.01	0.49	-0.07	0.19	0.52	-0.37
PPP2R4	0.48	0.12	0.63	-0.82	-0.14	0.51	-0.82	0.10	0.82	-0.29	-0.35	0.00	0.16	0.12	-0.15	0.84	-0.13	0.21	0.22	0.00
PPP2R5A	0.09	0.11	0.53	0.17	0.00	0.67	0.13	-0.04	0.85	-0.03	0.00	0.16	0.00	0.52	-0.19	-0.27	0.25	0.44	0.67	0.24
PPP2R5B	0.42	-0.11	0.98	0.20	0.55	0.66	0.00	0.06	1.23	-0.14	0.54	0.12	0.52	0.00	0.11	0.99	0.32	0.14	0.08	-0.32
PPP2R5C	0.05	0.18	0.68	-0.21	-0.03	0.55	-0.10	-0.01	0.56	-0.12	-0.01	-0.15	-0.19	0.11	0.00	-0.11	-0.08	0.14	0.41	-0.34
PPP2R5D	-0.15	-0.06	0.89	0.22	0.61	0.83	0.45	0.27	0.72	-0.12	0.49	0.84	-0.27	0.99	-0.11	0.00	0.57	0.53	0.74	-0.03
PPP2R5E	-0.12	0.29	0.33	-0.12	0.18	0.39	-0.21	-0.06	0.69	-0.08	-0.07	-0.13	0.25	0.32	-0.08	0.57	0.00	0.79	0.17	-0.06
STRN	0.19	0.29	0.84	0.42	0.32	0.70	0.00	0.63	0.11	-0.14	0.19	0.21	0.44	0.14	0.14	0.53	0.79	0.00	0.63	0.34
STRN3	0.49	-0.03	0.78	0.11	0.64	0.78	0.18	0.21	0.77	-0.12	0.52	0.22	0.67	0.08	0.41	0.74	0.17	0.63	0.00	0.00
STRN4	0.16	0.05	0.84	0.73	0.16	0.81	0.15	-0.18	0.29	-0.12	-0.37	0.00	0.24	-0.32	-0.34	-0.03	-0.06	0.34	0.00	0.00

*_ENST00000380737, **_ENST00000455566

G Bootstrapped $\hat{P}$ Values of Graph Weight Matrix

	PPP2CA	PPP2CB	PPP2R1A	PPP2R1B	PPP2R2A*	PPP2R2B	PPP2R2C	PPP2R2D**	PPP2R3A	PPP2R3B	PPP2R3C	PPP2R4	PPP2R5A	PPP2R5B	PPP2R5C	PPP2R5D	PPP2R5E	STRN	STRN3	STRN4
PPP2CA	0.00	0.58	0.15	0.30	0.44	0.02	0.33	0.23	0.01	0.88	0.09	0.17	0.25	0.16	0.47	0.42	0.92	0.15	0.03	0.23
PPP2CB	0.58	0.00	0.03	0.88	0.51	0.35	0.53	0.80	0.02	1.00	0.45	0.21	0.31	0.33	0.32	0.30	0.29	0.07	0.24	0.23
PPP2R1A	0.15	0.03	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.38	0.00	0.03	0.01	0.00	0.03	0.00	0.06	0.00	0.00	0.00
PPP2R1B	0.30	0.88	0.00	0.00	0.12	0.01	0.72	0.07	0.04	0.98	0.01	1.00	0.10	0.09	0.85	0.14	0.66	0.02	0.11	0.00
PPP2R2A*	0.44	0.51	0.00	0.12	0.00	0.00	0.01	0.19	0.00	1.00	0.03	0.45	0.18	0.01	0.25	0.02	0.09	0.03	0.01	0.12
PPP2R2B	0.02	0.35	0.00	0.01	0.00	0.00	0.20	0.00	0.00	0.49	0.01	0.09	0.00	0.02	0.01	0.00	0.01	0.00	0.00	0.00
PPP2R2C	0.33	0.53	0.00	0.72	0.01	0.20	0.00	0.28	0.00	0.97	0.11	1.00	0.15	0.20	0.36	0.04	0.81	0.18	0.04	0.07
PPP2R2D**	0.23	0.80	0.00	0.07	0.19	0.00	0.28	0.00	0.00	0.92	0.34	0.22	0.33	0.06	0.34	0.05	0.61	0.00	0.02	0.42
PPP2R3A	0.01	0.02	0.00	0.04	0.00	0.00	0.00	0.00	0.00	0.14	0.01	0.00	0.00	0.00	0.01	0.01	0.00	0.01	0.00	0.02
PPP2R3B	0.88	1.00	0.38	0.98	1.00	0.49	0.97	0.92	0.14	0.00	0.92	0.98	0.78	0.58	1.00	0.69	0.92	0.81	0.91	0.84
PPP2R3C	0.09	0.45	0.00	0.01	0.03	0.01	0.11	0.34	0.01	0.92	0.00	0.88	0.24	0.05	0.37	0.04	0.48	0.08	0.03	0.94
PPP2R4	0.17	0.21	0.03	1.00	0.45	0.09	1.00	0.22	0.00	0.98	0.88	0.00	0.23	0.14	0.49	0.03	0.36	0.16	0.13	0.31
PPP2R5A	0.25	0.31	0.01	0.10	0.18	0.00	0.15	0.33	0.00	0.78	0.24	0.23	0.00	0.03	0.77	0.48	0.10	0.01	0.01	0.08
PPP2R5B	0.16	0.33	0.00	0.09	0.01	0.02	0.20	0.06	0.00	0.58	0.05	0.14	0.03	0.00	0.08	0.02	0.08	0.04	0.09	0.55
PPP2R5C	0.47	0.32	0.03	0.85	0.25	0.01	0.36	0.34	0.01	1.00	0.37	0.49	0.77	0.08	0.00	0.32	0.32	0.13	0.01	0.73
PPP2R5D	0.42	0.30	0.00	0.14	0.02	0.00	0.04	0.05	0.01	0.69	0.04	0.03	0.48	0.02	0.32	0.00	0.02	0.00	0.00	0.23
PPP2R5E	0.92	0.29	0.06	0.66	0.09	0.01	0.81	0.61	0.00	0.92	0.48	0.36	0.10	0.08	0.32	0.02	0.00	0.00	0.08	0.27
STRN	0.15	0.07	0.00	0.02	0.03	0.00	0.18	0.00	0.01	0.81	0.08	0.16	0.01	0.04	0.13	0.00	0.00	0.00	0.00	0.01
STRN3	0.03	0.24	0.00	0.11	0.01	0.00	0.04	0.02	0.00	0.91	0.03	0.13	0.01	0.09	0.01	0.00	0.08	0.00	0.00	0.24
STRN4	0.23	0.23	0.00	0.00	0.12	0.00	0.07	0.42	0.02	0.84	0.94	0.31	0.08	0.55	0.73	0.23	0.27	0.01	0.24	0.00

*_ENST00000380737, **_ENST00000455566