Universiteit Leiden
The Netherlands

A comparison of methods for the construction of conditional independence networks

Rodenburg, F.

Citation
Rodenburg, F. (2017). *A comparison of methods for the construction of conditional independence networks*.

Version:	Not Applicable (or Unknown)
License:	License to inclusion and publication of a Bachelor or Master thesis in the Leiden University Student Repository
Downloaded from:	https://hdl.handle.net/1887/3596279

Note: To cite this publication please use the final published version (if applicable).

Mathematical Institute

Thesis

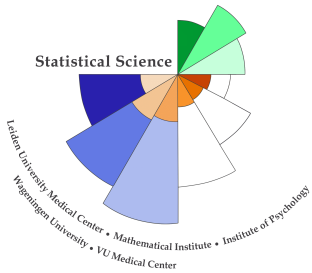
Statistical Science for the Life and Behavioural Sciences

A Comparison of Methods for the Construction of Conditional Independence Networks

Author:
Frans Rodenburg
Student number 1015788

Supervisor:
Dr. B.J.A. Mertens
Medical Statistics and Bioinformatics
Leiden University Medical Center

2016-2017



Contents

1 Introduction **2**
- 1.1 Conditional Independence Networks . . . 2
- 1.2 Regularization . . . 3
- 1.3 Other Extentions of Graphical Lasso and Ridge . . . 6
- 1.4 Bayesian Structure Learning . . . 7

2 A Brief Introduction to Graph Theory **10**
- 2.1 Centrality Measures . . . 10
- 2.2 Node Influence Metrics . . . 11
- 2.3 Community Detection . . . 13

3 Methods **15**
- 3.1 Data . . . 15
- 3.2 Software and Hardware . . . 15
- 3.3 Adaptation of the Expected Force . . . 15
- 3.4 Algorithm for the Weighted Influence of Nodes . . . 16

4 Results (Regularization) **18**
- 4.1 Determining the Support of λ . . . 18
- 4.2 Visualising the Support of λ with Regularization Paths . . . 19
- 4.3 Choosing the Optimal λ by Cross-Validation . . . 21
- 4.4 Alternatives to Cross-Validation . . . 24

5 Results (Bayesian Structure Learning) **26**
- 5.1 Prior Specification . . . 26
- 5.2 Assessing Convergence . . . 26
- 5.3 Choosing a Graph . . . 27

6 Visualisation **30**
- 6.1 `igraph` . . . 30
- 6.2 Centrality Measures . . . 34
- 6.3 Node Influence . . . 34
- 6.4 Community Detection . . . 36
- 6.5 Hierarchical Clustering . . . 37
- 6.6 Hiveplot . . . 37

7 A Closer Look at the Differences in Graphs **41**
- 7.1 Quantifying the Differences between Graphs . . . 41
- 7.2 Identified Peptides . . . 41
- 7.3 Qualifying the Differences between Graphs . . . 44

8 Discussion **45**
- 8.1 Constructing a Graph . . . 45
- 8.2 Inference in a Graph using WExF . . . 46
- 8.3 Conclusion . . . 46

9 Appendix **48**
- 9.1 GGM vs GCGM . . . 48
- 9.2 WExF: Consistency at Different Levels of Sparsity . . . 48

10 References **50**

1 Introduction

In the field of proteomics, an attempt is made to observe all proteins of an organism at once, rather than studying a preselected set of variables. While this obviates omitted-variable bias, the resulting high dimensionality of the data calls for the use of different methods than in conventional research. To this end, various techniques have been developed. In this thesis, we will focus on the use of sparse undirected graphs. Such graphs seek to visualize the most important partial correlations between the measured variables, albeit without providing a causal relationship.

While high dimensionality poses the main challenge of proteomics data, other problems arise as well when variables are highly correlated or when the number of measured variables exceeds the sample size (i.e. $n < p$). High correlation is especially challenging, since the nature of the correlation can be both biological or artificial. For example, proteins measured by mass spectrometry are fragmented, yielding multiple (correlated) peaks of the same protein. Since identification of all measured peptides is both laborious and not always possible, it is difficult to judge which resulting network is the correct one.

In this study, our objective is to evaluate the performance of existing methods to estimate and validate sparse undirected graphs, and to address their usefulness for proteomics mass spectrometry data, with $n < p$. To this end, we will compare regularization methods and a Bayesian method for the construction of sparse undirected graphs. Subsequently, we will try to express the relative importance of vertices in resulting graphs.

For ease of understanding, we consider analogies with regression analysis before addressing the graphical case, since most of these methods originate in regression analysis. We touch upon a variety of subjects, including inverse covariance matrix estimation, graph theory and node influence metrics, to address the question of how to summarize complex relationships in proteomics.

1.1 Conditional Independence Networks

Undirected graphs can be constructed by using a sparse estimate of the inverse covariance matrix of the data. Let $\boldsymbol{\Sigma} = \mathbb{E}\Big[\big(\boldsymbol{Y} - \mathbb{E}[\boldsymbol{Y}]\big)\big(\boldsymbol{Y} - \mathbb{E}[\boldsymbol{Y}]\big)^T\Big]$ be the true covariance matrix and $\boldsymbol{S} = \frac{1}{n}\boldsymbol{Y}^T\boldsymbol{Y}$ the maximum likelihood estimate (MLE). We then wish to estimate $\boldsymbol{\Sigma}^{-1} = \boldsymbol{\Theta}$, the precision matrix, using $\boldsymbol{S}$. It turns out that the entries of the precision matrix have an interpretation in terms of partial correlation (Lauritzen 1996). That is, the partial correlation of proteins i and j given the associations with all other proteins is equal to minus the standardized precision estimate:

$$r_{ij} = -\frac{\theta_{ij}}{\sqrt{\theta_{ii}\theta_{jj}}} \tag{1}$$

with $\theta_{ij} \in \boldsymbol{\Theta}$. From (1), it is clear that when an entry of the precision matrix $\theta_{ij} = 0$, proteins i and j are conditionally independent. This implies that if $\hat{\boldsymbol{\Theta}}$ is sufficiently sparse, we can represent the remaining non-zero entries as connections between nodes, which in our case correspond to protein peptides. A simple example of such a graph is shown in (24).

Sparsity is important, because we want to visualize the relationships which explain the data best, without cluttering the graph. However, recent literature has focussed primarily on finding a sparse estimate (see Friedman, Hastie, and Tibshirani 2008; Krämer, Schäfer, and Boulesteix 2009; Bien and Tibshirani 2011; Mazumder and Hastie 2012b; Mazumder and Hastie 2012a; Sun and Li 2012), without regard for the estimate closest to the true $\boldsymbol{\Theta}$. We take on a different approach and allow $\hat{\boldsymbol{\Theta}}$ to be as dense as the methods for estimating $\boldsymbol{\Theta}$ suggest it should be, while summarizing the most important nodes in the resulting graph.

1.2 Regularization

Let $\{\boldsymbol{y}_1 \dots \boldsymbol{y}_n\}$ be n observations of p variables, following a multivariate normal distribution $\boldsymbol{Y} \sim \mathcal{N}_p(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ with $\boldsymbol{\mu} = \mu_1, \dots, \mu_p$ and $\boldsymbol{\Sigma}$ a $p \times p$ covariance matrix. If we assume $\boldsymbol{\mu}$ to be known and equal to zero, the MLE of the variance-covariance matrix $\boldsymbol{\Sigma}$ is equal to $\boldsymbol{S} = \frac{1}{n}\boldsymbol{Y}^T\boldsymbol{Y}$. The likelihood of $\boldsymbol{\Theta}$ would then be:

$$\mathcal{L}(\boldsymbol{\Theta}|\boldsymbol{S}) = |\boldsymbol{\Theta}| - \exp\left\{tr(\boldsymbol{S}\boldsymbol{\Theta})\right\} \tag{2}$$

Here, $|.|$ denotes the determinant and tr the trace. From (2), we can formulate the log-likelihood as follows:

$$\ell(\boldsymbol{\Theta}|\boldsymbol{S}) = ln(|\boldsymbol{\Theta}|) - tr(\boldsymbol{S}\boldsymbol{\Theta}) \tag{3}$$

If $n > p$ and $\boldsymbol{S} \succeq 0$, (3) is maximized for $\hat{\boldsymbol{\Theta}} = \boldsymbol{S}^{-1}$. However, when $n < p$, $\boldsymbol{S}$ is no longer positive semi-definite, singular and therefore uninvertible. As a consequence, (3) becomes an ill-posed problem and $\hat{\boldsymbol{\Theta}}_{MLE}$ is not defined. A common way to estimate $\boldsymbol{\Theta}$ when $n < p$ is by adding a penalty to the MLE. To this end, we will consider two regularization methods: the ℓ_1 and ℓ_2 norms (graphical lasso and ridge, respectively).

1.2.1 Lasso (ℓ_1-norm)

The Least Absolute Shrinkage and Selection Operator, commonly just referred to as lasso, is a constriction on the MLE that results in both parameter shrinkage and selection at the same time. In regression analysis, lasso puts a restriction on the ℓ_1-norm of the parameters c.f. (5). In **figure 1** (left), the constrained parameter space of the lasso is visualized for the two-dimensional case. Here, we see that the boundaries of the lasso-contstrained parameter space are diamond shaped. As a result, the contours of equiprobability of the least squares solution can enter the constrained parameter space through a corner. When this occurs, one of the β_j is set to exactly zero, which gives the lasso its parameter selection ability. As the number of parameters p increases, so does the chance of coefficients to be set to zero.

In regression analysis, the ordinary least squares (OLS) estimate is obtained by minimizing the sum of squared residuals:

$$||\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}||_2^2 \tag{4}$$

Where $||.||_2$ denotes the ℓ_2 or Euclidean norm. When p is large and variables are correlated, lasso regression adds the ℓ_1-penalty to the OLS, multiplied by a positive scalar λ to control the size of the penalty:

$$||\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}||_2^2 + ||\lambda\boldsymbol{\beta}||_1 \tag{5}$$

Where $||.||_1$ denotes the ℓ_1 or 'taxicab' norm. Lasso regression can be extended fairly easily to the problem of covariance matrix inversion in (3), by subtracting the (matrix) ℓ_1-norm of $\boldsymbol{\Theta}$:

$$\ell(\boldsymbol{\Theta}|\boldsymbol{S}) = ln(|\boldsymbol{\Theta}|) - tr(\boldsymbol{S}\boldsymbol{\Theta}) - \lambda||\boldsymbol{\Theta}||_1 \tag{6}$$

Fast algorithms for solving (6) have been developed by Mazumder and Hastie (2012a); Mazumder and Hastie (2012b) & Friedman, Hastie, and Tibshirani (2014), such as block-coordinate descent, which partitions $\boldsymbol{\Theta}$ and $\boldsymbol{W} = \hat{\boldsymbol{\Theta}}^{-1}$ into block-matrices (block diagonal structure in $\boldsymbol{S}$ is preserved in $\boldsymbol{\Theta}$). Each block can then be solved independently, saving considerable computation time.

More explicitly, the columns of $\boldsymbol{\Theta}$ are iteratively rearranged so that the target column $\binom{\boldsymbol{\theta}_{12}}{\theta_{22}}$ in (7) is the last column. $\hat{\boldsymbol{\Theta}}$ is then updated using the current estimate of $\boldsymbol{W}$. As more and more elements of $\boldsymbol{W}$ become

zero, the speed of the algorithm increases. In Mazumder and Hastie (2012a), additional simplifications were introduced which further speed up the algorithm.

$$\boldsymbol{\Theta} = \begin{pmatrix} \boldsymbol{\Theta}_{11} & \boldsymbol{\theta}_{12} \\ \boldsymbol{\theta}_{21} & \theta_{22} \end{pmatrix}, \hat{\boldsymbol{\Theta}}^{-1} = \begin{pmatrix} \boldsymbol{W}_{11} & \boldsymbol{w}_{12} \\ \boldsymbol{w}_{21} & w_{22} \end{pmatrix} \tag{7}$$

Though the computational speed and resulting sparsity of the graphical lasso are appealing at first, a problem arises from a mathematical point of view with the ℓ_1-penalty, since the lasso solution is not uniquely determined when the number of variables is greater than the number of subjects (i.e. $n < p$) and selects at most n variables. Furthermore, the lasso solution does not converge to (3) as $n \to \infty$. Another drawback of lasso is that it tends to select only one variable among a group of predictors with high pairwise correlations. In the latter case, there are alternative solutions like the group or fused lasso (see Friedman, Hastie, and Tibshirani (2010) & Danaher, Wang, and Witten (2014)). We used the R package `glasso` to compute the graphical lasso. It allows shrinkage of all elements of $\boldsymbol{\Theta}$ or only off-diagonal elements (default).

1.2.2 Ridge (ℓ_2-norm)

When sparsity is not of essence, the most commonly used method of regularization is Tikhonov regularization, which adds a regularization term to (4) for any suitable Tikhonov matrix $\boldsymbol{\Gamma}$:

$$||\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}||_2^2 + ||\boldsymbol{\Gamma}\boldsymbol{\beta}||_2^2 \tag{8}$$

Where $||.||_2$ denotes the ℓ_2 or Euclidean norm. $\boldsymbol{\Gamma}$ is a $p \times p$ matrix, often taken to be a multiple of the identity matrix $\boldsymbol{\Gamma} = \lambda \boldsymbol{I}$. When this is the case, since $||\lambda \boldsymbol{I}\boldsymbol{\beta}|| = |\lambda|||\boldsymbol{\beta}||$, (8) then simplifies to:

$$||\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}||_2^2 + \lambda||\boldsymbol{\beta}||_2^2 \tag{9}$$

Which is known as ℓ_2 regularization, or ridge regression. Note that λ is on a different scale than (5), because the penalty is quadratic. The ridge penalty has the desirable property over lasso of having a unique solution, even when $p > n$. However, as is shown in **figure 1**, the ridge penalty constrains the βs to a circular space (spherical in higher dimensions). Because of this, the chance of the contours of equiprobability of the least squares solution entering the constrained space through $\beta_j = 0$ is now zero. As such, ridge regularization cannot select variables like the lasso. In graphical models, this implies that the ridge solution is not sparse and requires an additional variable selection step. Another complication arises from this, since the block-diagonal partitioning of $\boldsymbol{S}$ and $\boldsymbol{\Theta}$ in (7) is no longer beneficial for computation speed.

Intuitively, one might assume that the ridge regression is extended to graphical models as follows:

$$\ell(\boldsymbol{\Theta}|\boldsymbol{S}) = ln(|\boldsymbol{\Theta}|) - tr(\boldsymbol{S}\boldsymbol{\Theta}) - \lambda||\boldsymbol{\Theta}||_2^2 \tag{10}$$

However, as shown in (11) and (12), the archetypal graphical ridge has a different form, due to the computational difficulty of estimating $\boldsymbol{\Theta}$ when no elements become zero. Peeters & Van Wieringen addressed this and proposed two alternative estimators, which we will discuss in the next paragraph. First, we discuss the archetypes.

Commonly, a target matrix $\boldsymbol{T}$ is introduced, c.f. (11). This is known as target shrinkage, or type I ridge estimation. A common choice for the target matrix in low-dimensional setting is $\boldsymbol{T}_{jj} = \boldsymbol{S}_{jj}$. Target shrinkage with the proposed target is identical to regularization on only the off-diagonal elements (as is also the default in the graphical lasso implementation in R: `glasso`). The choice for $\boldsymbol{T}$ can also be thought of as a prior on $\boldsymbol{\Theta}$, in which case the proposed target is a weakly informative prior.

$$\ell(\boldsymbol{\Theta}|\boldsymbol{S}) = ln(|\boldsymbol{\Theta}|) - (1-\lambda)tr(\boldsymbol{S}\boldsymbol{\Theta}) - \lambda tr(\boldsymbol{\Theta}\boldsymbol{T}) \tag{11}$$

Alternatively, type II ridge estimation can be used, which penalizes all elements of $\boldsymbol{\Theta}$ (including the diagonal). The log-likelihood for this type of estimation is:

$$\ell(\boldsymbol{\Theta}|\boldsymbol{S}) = ln(|\boldsymbol{\Theta}|) - tr(\boldsymbol{S}\boldsymbol{\Theta}) - \lambda tr(\boldsymbol{\Theta}) \tag{12}$$

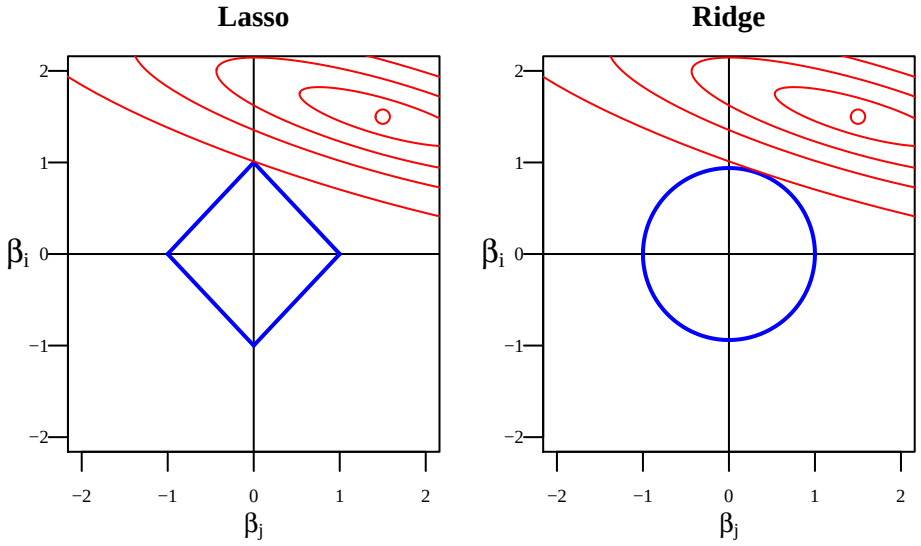


Figure 1: Visualisation of the contours of equiprobability of the least squares estimate entering the constrained parameter space of the lasso (left: $\lambda \sum |\boldsymbol{\beta}| \leq 1$) and the ridge (right: $\lambda \sum \boldsymbol{\beta}^2 \leq 1$). In the case of lasso, due to the edge(s) of the diamond-shaped constraint, some estimates will be shrunk to exactly zero, resulting in sparsity. While lasso induces sparsity, the circle-shaped constraint of ridge has probability zero to result in exactly zero for an estimate. Note that this figure depicts a 2-dimensional scenario ($p = 2$), for ease of interpretation. In higher dimensional space, the lasso will have increasingly higher chance of resulting in estimates that are exactly zero, due to the number of edges on the diamond increasing as well.

1.2.3 Alternative Ridge Estimators

Surprisingly, neither of the archetypal estimators in (11) and (12) are direct extensions of the ℓ_2-norm of $\boldsymbol{\beta}$ used in ridge regression. Van Wieringen and Peeters (2015) addressed this issue by deriving analytic expressions for two alternative ridge estimators (c.f. (13), Type I and Type II, respectively), which resemble the constraint to the ℓ_2-norm of the entries of $\boldsymbol{\beta}$ in ridge regression more closely. Their article contains proof that the alternative estimators amount to a proper ridge penalty as in (14) (Type I and Type II, respectively). Furthermore, the alternative estimators were shown to be superior to the default ridge estimators under various loss functions. Additionally, the estimators retain the property of the fast lasso algorithms (e.g. block coordinate descent in eq. 7), which speed up as the penalty increases. The alternative estimators have been implemented in the R package `rags2ridges`, which we have used extensively in this article.

$$\hat{\boldsymbol{\Theta}}^{\mathrm{I}}(\lambda) = \left(\left(\lambda \mathbf{I}_p + \frac{1}{4}(\mathbf{S} - \lambda \mathbf{T})^2 \right)^{1/2} + \frac{1}{2}(\mathbf{S} - \lambda \mathbf{T}) \right)^{-1} \tag{13a}$$

$$\hat{\boldsymbol{\Theta}}^{\mathrm{II}}(\lambda) = \left(\left(\lambda \mathbf{I}_p + \frac{1}{4}\mathbf{S}^2 \right)^{1/2} + \frac{1}{2}\mathbf{S} \right)^{-1} \tag{13b}$$

The penalties corresponding to the alternative estimators shown above are respectively:

$$\frac{1}{2}\lambda tr\Big((\mathbf{\Theta}-\mathbf{T})^T(\mathbf{\Theta}-\mathbf{T})\Big) = \frac{1}{2}\lambda||\mathbf{\Theta}-\mathbf{T}||_2^2 \tag{14a}$$

$$\frac{1}{2}\lambda tr(\mathbf{\Theta}^T\mathbf{\Theta}) = \frac{1}{2}\lambda||\mathbf{\Theta}||_2^2 \tag{14b}$$

Here, $||.||_2$ denotes the ℓ_2 or euclidean norm. The ridge penalty is multiplied by $\frac{1}{2}$ in accordance with the penalties used in the `rags2ridges` package. Its use will become clear in the Bayesian interpretation of regularization.

1.2.4 Achieving Sparsity in the Ridge Solution

A problem with ridge regression is that it will shrink parameters towards zero, but never exactly to zero, unless λ is chosen so high that all parameters become zero. This means that after choosing the strength of the penalty, an additional step is required to perform variable selection (and thereby making the precision matrix sparse). To this end, several options are available. We used the highest x absolute entries of $\hat{\mathbf{\Theta}}$, which allowed us to specify the number of edges in the graph. Conveniently, this also allows us to create sparse ridge solutions of the same level of sparsity as a given lasso solution.

Another options for sparsifying $\hat{\mathbf{\Theta}}$ is local False Discovery Rate (lFDR, Efron 2007) which is also included in `rags2ridges`. For technical details about these methods, we refer to the package documentation.

1.3 Other Extentions of Graphical Lasso and Ridge

1.3.1 Elastic Net

In regression analysis, the lasso and ridge penalty can be combined into a hybrid penalty. This is know as an elastic net and has the advantage of deriving some of the desirable proporties of both lasso and ridge. Namely, to select variables and shrink variables proportionately, respectively. Although the use of elastic net in graphical models has been proposed in Cucuringu, Puente, and Shue (2011), no software or `R` package has implemented the graphical elastic net yet at the time of writing this thesis.

1.3.2 Fused Graphical Lasso and Ridge

The fused penalties have been proposed and implemented to allow for the simultaneous estimation of two or more matrices of the same dimensions, corresponding to different classes. Yang et al. (2015) implemented fused graphical lasso for the estimation of brain networks of Alzheimer's patients. They estimated three precision matrices simulaneously, corresponding to normal controls, individuals with mild cognitive impairment (MCI) and Alzheimer's patients (AD). Their objective was to find overall similarity between brain networks and elaborate on the differences found between AD, MCI and controls.

This could prove valuable for case/control data and so we intended to also perform an analyses using two covariance matrices created from cases and controls separately. However, due to time constraints, this was left out. The `JGL` package and the `rags2ridgesFused` module of the R package `rags2ridges` are available for the estimation of fused graphical lasso and ridge, respectively.

1.4 Bayesian Structure Learning

Bayesian statistics is an alternative method of inference, where the likelihood is combined with a prior distribution function c.f. eq. (15). This prior takes into account the available knowledge or lack thereof beforehand. Furthermore, all variables are considered random variables, to model (unknown) sources of uncertainty. The merits of Bayesian statistics for graphical models in particular become clear when we consider the prior as a means of inducing sparsity. In the following paragraphs, we discuss the application of Bayesian statistics to graphical models.

$$\pi(\boldsymbol{\theta}) = p(\boldsymbol{\theta}|\boldsymbol{y}) = \frac{p(\boldsymbol{y}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{\int p(\boldsymbol{y}|\boldsymbol{\theta})p(\boldsymbol{\theta})\,\mathrm{d}\boldsymbol{\theta}} \propto p(\boldsymbol{y}|\boldsymbol{\theta})p(\boldsymbol{\theta}) \tag{15}$$

1.4.1 Bayesian Interpretation of Lasso and Ridge

If we consider the penalized likelihood functions for regression analysis in (5) and (9), one can easily show that the penalties in lasso and ridge amount to a Laplace and multivariate normal prior on $\boldsymbol{\beta}$, respectively:

$$\pi(\boldsymbol{\beta}_j|\boldsymbol{\Sigma}) = \frac{\lambda}{2\boldsymbol{\sigma}} \exp\left\{-\frac{\lambda|\boldsymbol{\beta}|_j}{\boldsymbol{\sigma}}\right\} \tag{16}$$

$$\pi(\boldsymbol{\beta}_j|\boldsymbol{\Sigma}) = \frac{\lambda}{2\boldsymbol{\sigma}} \exp\left\{-\frac{\lambda\boldsymbol{\beta}_j^2}{2\boldsymbol{\sigma}}\right\} \tag{17}$$

with $\boldsymbol{\sigma}^T\boldsymbol{\sigma} = \boldsymbol{\Sigma}$. As can be seen in **figure 2**, the laplace prior retains its variable selection from a Bayesian perspective, if the maximum a posteriori (MAP) estimate is used. If sparsity is not of essence, the ridge penalty is also justifiable from a Bayesian perspective. Namely, if we assume the means of $\boldsymbol{\beta}$ to be zero, independent of each other and independent of the error term, normally distributed and with equal variance, the ridge solution is the most probable solution given the data with a multivariate normal prior on $\boldsymbol{\beta}$ as in eq. (17), (Vogel 2002).

Type I and type II shrinkage (13a, 13b) also have a Bayesian interpretation. Namely, if we consider the target matrix $\boldsymbol{T}$ to be prior information, then type I shrinkage — a diagonal choice for $\boldsymbol{T}$ — can be considered a weakly informative prior. Contrarily, type II shrinkage yields the least informative prior $\boldsymbol{T} = \boldsymbol{0}$, which considers all partial correlations to be equally likely a priori. Although in general we wish to express ignorance through an uninformative prior, type I shrinkage can be justified in proteomics, since we expect a sparse interaction network of proteins. In other words, a model where all proteins interact with all other proteins is not considered realistic.

Deriving analytical expressions for Bayesian regularized precision matrices is not straightforward. Hao Wang proposed a prior on $\boldsymbol{\Theta}$ corresponding to the lasso penalty, but a fully Bayesian approach to graphical ridge was not available at the time of writing this article. Furthermore, no software for Bayesian graphical lasso was already available in `R`. Hence, we did not implement Bayesian graphical regularization.

1.4.2 Alternative Bayesian Approach

Despite the lack of available Bayesian graphical lasso or ridge, an alternative approach to the construction of conditional independence networks that does not involve regularization exists and is available in the `R` package `BDgraph`. Instead of shrinkage through regularization, $\boldsymbol{\Theta}$ is sampled directly from the posterior distribution. Sparsity is induced by tuning the degrees of freedom (ν in eq. 18) on the prior of $\boldsymbol{\Theta}$, shifting from the least informative prior to a weakly informative prior.

Commonly, the G-Wishart distribution is used as a prior, which is conjugate for the precision matrix and yields a postive semi-definite posterior. There is also an intuitive explanation, namely: the inverse gamma

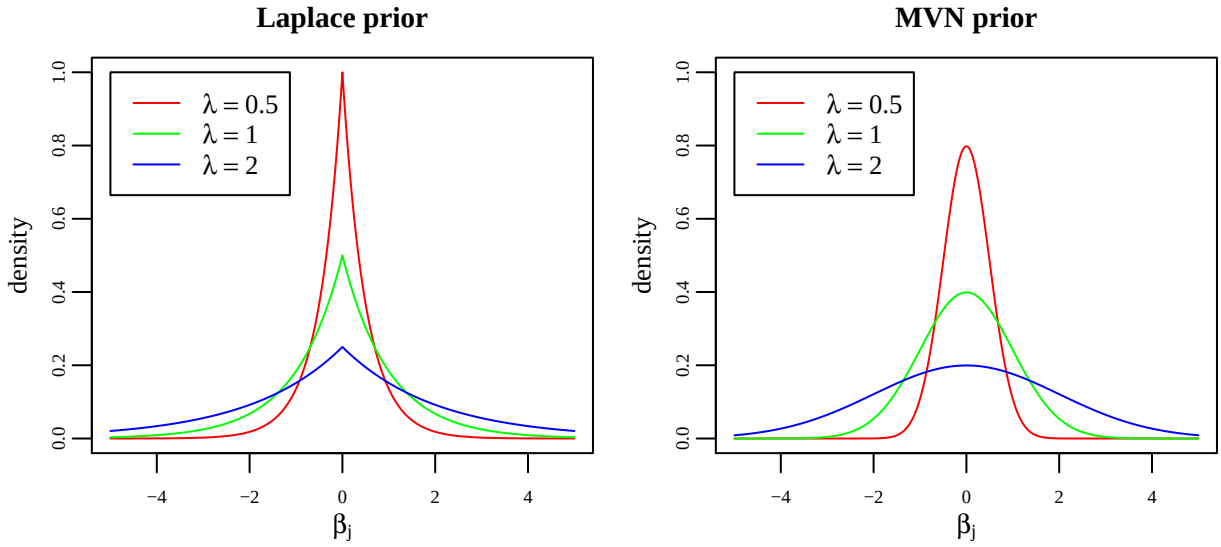


Figure 2: Visualisation of the densities of the Laplace and (multivariate) normal priors for a single β_j, yielding lasso and ridge shrinkage, respectively. Although both priors are diffuse, the Laplace distribution has a singularity at its center, allowing β_j in (16) to become exactly zero if the maximum a posteriori (MAP) estimate is used.

distribution is conjugate to a normal likelihood with unknown variance, and the univariate form of the inverse-Wishart distribution (i.e. $p = 1$) can be rewritten as the inverse-gamma distribution. As such, if we are interested in the inverse of variance, the Wishart distribution is conjugate to the posterior of $\boldsymbol{\Theta}$.

$$\pi(\boldsymbol{\Theta}) \propto |\boldsymbol{\Theta}|^{(\nu-p-1)/2} \exp\Big\{ -\frac{1}{2} tr(\boldsymbol{B\Theta}) \Big\} \tag{18}$$

Where $\nu > p - 1$ are the degrees op freedom and $\boldsymbol{B}$ is a matrix of parameters (the scale matrix for $\boldsymbol{\Theta}$), commonly taken to be the identity matrix $\boldsymbol{I}_p$. Note that in literature, the Wishart and inverse-Wishart distribution are used interchangeably, but they are not the same. For example, the least informative prior on $\boldsymbol{\Theta}$, as suggested recently by Kuismin and Silanpää (2016) would be obtained when ν is small, and when the degrees of freedom ν increase, $\boldsymbol{\Theta}$ would concentrate around the scaling matrix $\boldsymbol{B}$. This is true for the inverse-Wishart distribution, which is conjugate to the covariance matrix $\boldsymbol{\Sigma}$, but for the Wishart distribution (conjugate prior for $\boldsymbol{\Theta}$), it is the other way around, as we will see in the section **results (Bayesian structure learning)**, **table 1**.

Consider the likelihood of the data:

$$p(\boldsymbol{Y}|\boldsymbol{\Theta}) \propto |\boldsymbol{\Theta}|^{n/2} \exp\Big\{ -\frac{1}{2} tr(\boldsymbol{S\Theta}) \Big\} \tag{19}$$

With n the number of observations, $\boldsymbol{S}$ the MLE for the covariance matrix and $\boldsymbol{\Theta}$ the precision matrix to be estimated. We can combine the prior in (18) with the likelihood in (19) to obtain the posterior for $\boldsymbol{\Theta}$, proportional to some normalizing factor as shown in (15):

$$\pi(\boldsymbol{\Theta}) \propto |\boldsymbol{\Theta}|^{(n+\nu-p-1)/2} \exp\Big\{ -\frac{1}{2} tr\big((\boldsymbol{B}+\boldsymbol{S})\boldsymbol{\Theta}\big) \Big\} \tag{20}$$

1.4.3 Birth-Death MCMC

Mohammadi and Wit (2015) proposed a new Monte Carlo Markov Chain (MCMC) sampler that uses continous time birth and death rates of edges to arrive at a stationary distribution for $\pi(\boldsymbol{\Theta})$ in (20). They derived proof for convergence to such a stationary distribution based on proposed birth-death processes by Preston (1976). Their sampler is included in the `BDgraph` package and since it differs so fundamentally from the other techniques, we included a short description. For technical details, we refer to Mohammadi and Wit (2015) and Mohammadi and Wit (2016).

Specifically, their sampler calculates the independent birth rates of each edge, the independent death rates of each edge and also, the 'waiting time' at each graph. Let $\beta(\boldsymbol{\Theta})$ and $\delta(\boldsymbol{\Theta})$ denote the overall birth and death rates, respectively and let $\beta_e(\boldsymbol{\Theta})$ and $\delta_e(\boldsymbol{\Theta})$ denote the birth and death rate of an individual edge. The birth and death rates are then defined as independent Poisson processes as follows:

$$\beta(\boldsymbol{\Theta}) = \sum_{e \notin \boldsymbol{E}} \beta_e(\boldsymbol{\Theta}) \tag{21a}$$

$$\delta(\boldsymbol{\Theta}) = \sum_{e \in \boldsymbol{E}} \delta_e(\boldsymbol{\Theta}) \tag{21b}$$

With the waiting time $w(\boldsymbol{\Theta})$ at each graph defined as:

$$w(\boldsymbol{\Theta}) = \frac{1}{\beta(\boldsymbol{\Theta}) + \delta(\boldsymbol{\Theta})} \tag{22}$$

The birth-death MCMC (BDMCMC) sampler then calculates birth and death rates using (21), calculates the waiting time using (22), simulates the type of jump (birth or death) and finally, samples from the G-Wishart proportional to the waiting time, to update $\hat{\boldsymbol{\Theta}}$.

Note that since the algorithm converges to a stationary state with posterior probabilities of edge inclusion, slightly different choices for the degrees of freedom ν on the prior in (18) shrink the elements of $\hat{\boldsymbol{\Theta}}$ proportionally. However, the choice for inducing a certain level of sparsity in $\hat{\boldsymbol{\Theta}}$ is simple, since the graph is updated discretely in each jump. We can therefore keep track of the size of the graph in terms of the edges $|\boldsymbol{E}(\boldsymbol{G})|$. Then, we sparsify $\hat{\boldsymbol{\Theta}}$ to the MAP estimate of graph size, following the burn-in period. This will be shown in the **results (Bayesian structure learning)** section.

2 A Brief Introduction to Graph Theory

A branch of combinatorics, graph theory seeks to describe and summarize pairwise relations between finite discrete objects, in a graph. A graph $\boldsymbol{G} \equiv (\boldsymbol{V}, \boldsymbol{E})$ consists of a set $\boldsymbol{V}(\boldsymbol{G})$ of vertices (or nodes) and $\boldsymbol{E}(\boldsymbol{G})$ edges (or connections). Said graph is connected if every vertex is reachable by traveling across the edges. Furthermore, an undirected graph is said to be strongly connected if every vertex has edges connecting it to every other vertex. Finally, a graph is simple if and only if it contains no self-edges (loops) or multiple edges between vertices. An example of a simple, undirected graph, constructed from the adjacency matrix $\boldsymbol{A}(\boldsymbol{\Theta})$ of a hypothetical precision matrix $\boldsymbol{\Theta}$, is shown below.

$$\boldsymbol{A}(\boldsymbol{\Theta}) = \begin{cases} 0 & \text{if } \theta = 0 \\ 1 & \text{if } \theta \neq 0 \end{cases} \tag{23}$$

$$\boldsymbol{A}(\boldsymbol{\Theta}) = \begin{Bmatrix} a_{ii} & a_{ij} & a_{ik} & a_{il} \\ a_{ji} & a_{jj} & a_{jk} & a_{jl} \\ a_{ki} & a_{kj} & a_{kk} & a_{kl} \\ a_{li} & a_{lj} & a_{lk} & a_{ll} \end{Bmatrix} = \begin{Bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \end{Bmatrix} \implies \boldsymbol{E}(\boldsymbol{G}) = \begin{pmatrix} v_i — v_k \\ v_j — v_k \\ v_j — v_l \\ v_k — v_l \end{pmatrix} \tag{24}$$

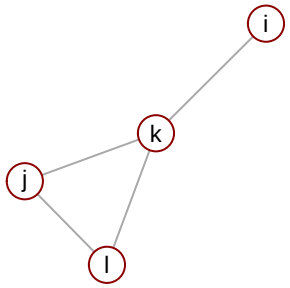


Figure 3: Example of a graph $\boldsymbol{G} \equiv (\boldsymbol{V}, \boldsymbol{E})$ constructed from a 4×4 precision matrix $\boldsymbol{\Theta}$.

Graphs constructed from precision matrices are often referred to in literature as Gaussian graphical models (GGMs). In GGMs, variables from which $\boldsymbol{S}$ is constructed are assumed to follow a multivariate normal distribution. If this assumption cannot be made, Gaussian copula graphical models (GCGM) can be used instead. When we discuss graphical models in this report, we solely refer to GGMs. Furthermore, it is implied in this report that any graph $\boldsymbol{G}$ is undirected.

2.1 Centrality Measures

Once a (possibly sparse) graph $\boldsymbol{G}$ is constructed from $\hat{\boldsymbol{\Theta}}$, the question remains of how to summarize the graph. While this may be trivial for sufficiently sparse solutions, a graph with cardinality between $0 < |\boldsymbol{V}| < 196$ nodes and $0 < |\boldsymbol{E}| < \frac{196 \cdot 195}{2}$ edges can be sparse in terms of the number of zero elements in $\hat{\boldsymbol{\Theta}}$, but still difficult to interpret. To this end, centrality measures, which we will denote by C have been developed to identify the most central nodes in graph theory. In the following chapter, we will cover the most important ones.

2.1.1 Degree Centrality

The degree centrality (sometimes simply referred to as degree or valency) is the simplest of the centrality measures, but an important one. The degree of a vertex $v \in \boldsymbol{V}(\boldsymbol{G})$ is defined as the number of edges connected to that vertex and is usually denoted as $C_{\text{degree}} = \deg(v)$. A connected graph can be obtained by removing all vertices with degree zero.

While the degree gives some indication of which vertices are central, it is not a good measure for graphs with variable partial correlations. Hence, a simple extension is the weighted degree, which is the sum of edge weights connected to a vertex.

2.1.2 Betweenness

The betweenness is defined as the number of times a vertex is crossed when traveling the shortest distance from one vertex to another in a graph $\boldsymbol{G}$. The betweenness of a vertex $v \in \boldsymbol{V}(\boldsymbol{G})$ is calculated by defining the shortest distances between each pair of vertices and then counting the number of times the given vertex is present in the shortest path. Let $\delta_{min}(v_i, v_j)$ denote the shortest path between vertex i and j. If i and j are adjacent, their shortest distance is taken to be 0, since no nodes are crossed and they do not contribute to any other nodes betweenness.

$$C_{\text{between}}(v_k) = \sum_{i \neq j \neq k \in \boldsymbol{V}(\boldsymbol{G})} \sum v_k \in \delta_{min}(v_i, v_j) \tag{25}$$

2.1.3 Closeness

The closeness of a vertex $v \in \boldsymbol{V}(\boldsymbol{G})$ is defined as the reciprocal of its farness, which is the sum of its (shortest) distances to every other node in the graph. Note that for regularization, a slightly different penalty can cause the inclusion of an otherwise disconnected node, causing all 'nearby' vertices to gain a sudden increase in closeness, with respect to the vertices farther from the newly included node. It may therefore not be a suitable centrality measure for estimated graphs, but is commonly used in predefined graphs.

$$C_{\text{close}}(v_i) = \frac{1}{\sum_{i \neq j \in \boldsymbol{V}(\boldsymbol{G})} |\delta_{min}(v_i, v_j)|} \tag{26}$$

2.1.4 Eigenvalue centrality

Eigenvalue centrality owes its name to the fact that it can be rewritten as $\boldsymbol{\lambda v} = \boldsymbol{Av}$, meaning λ is an Eigenvalue of $\boldsymbol{A}$. The Eigenvalue centrality of each vertex is proportional to the sum of the Eigenvalue centrality of each neighbouring vertex. This recursive property results in higher Eigenvalue centrality in relatively dense parts of the graph: higher values are assigned to verices surrounded by others with high values, and this Formally, the Eigenvalue centrality is defined as follows:

$$\lambda \cdot C_{\text{EV}}(v_i) = \sum_{v_j \in N_G(v_i)} C_{\text{EV}}(v_j) = \sum_{v_j \in N_G(v_i)} a_{ij} \cdot C_{\text{EV}}(v_j) \tag{27}$$

Where $N_G(v_i)$ denotes the neighbourhood of vertex v_i, $a_i j \in \boldsymbol{A}$ is the adjacency matrix of $\boldsymbol{\Theta}$ shown in (23) and $\frac{1}{\lambda}$ is a constant. An adaptation of Eigenvalue centrality is Google's PageRank (Page et al. 1998).

2.2 Node Influence Metrics

While centrality measures are mathematically attractive summaries of a given network, an important limitation is their sensitivity to subtle changes in the network and its size. While this does not pose a problem for known networks, our data yields an estimate $\hat{\boldsymbol{\Theta}}$ of the true $\boldsymbol{\Theta}$ and cannot be assumed to be without error. To determine the relative importance of nodes in an incomplete or noisy graph, physicists and epidemiologists have developed node influence metrics. We will discuss one of these metrics: the expected force of infection (ExF), and a simple extension: the (partial correlation) weighted expected force of infection (WExF).

2.2.1 Expected Force of Infection

In epidemiology, infection can be modelled in graphs, depicting those susceptible to infection as vertices and all possible transmissions of infection as edges between those vertices. Furthermore, we can define transmission paths of length t to assess the local influence of a given vertex. The goal in epidemiology is then to identify the vertices with the highest local influence, as this can predict the spreading power of a disease from any given starting point, such as an airport in a network of commercial flights (Lawyer 2016). Although its roots lie in epidemiology, ExF has been shown to effectively predict node influence in a variety of scientific areas, ranging from digital transactions to ecological processes. (Milkau and Bott 2015; Jordan et al. 2016; Pereira et al. 2015; Ellinas et al. 2015; Lawyer 2016).

What makes the ExF more appealing than its competitors (e.g. PageRank) for covariance matrix inversion is that it does not depend on the entire graph, but only on the local influence defined by transmission. Since it depends on local graph structure, it is much less sensitive to changes in the graph than alternatives like k-shell and PageRank. This is important, since our graph is estimated from the precision matrix, and not strictly defined.

As is shown in **figure 4**, we can define the force of infection by starting at an infected vertex v_i, with all other vertices susceptible. We then consider every possible cluster after $t = 2$ transmissions and sum the number of edges that can be reached from any of the vertices on such a cluster. Lawyer (2014) showed that more than two transmissions adds very little information to the ExF, while increasing the complexity of the problem from $\mathcal{O}(p^3)$ to $\mathcal{O}(p^{t+1})$. We therefore only consider $t = 2$ from here on.

Lawyer (2015) defines the ExF as follows. Starting at a specific vertex, define all possible transmission paths. Then, denote the number of outward edges from a given path: the cluster degree d_j, for transmission paths (clusters) $j = 1 \dots |J|$. Then normalise the cluster degree over the cluster degrees of all paths from the starting vertex.

$$\bar{d}_j = \frac{d_j}{\sum_{j=1}^{|J|} d_j} \tag{28}$$

Finally, we take the entropy of the normalised cluster degree to be the ExF of the starting vertex:

$$\text{ExF}(i) = -\sum_{j=1}^{|J|} \bar{d}_j \cdot \log_2(\bar{d}_j) \tag{29}$$

As is also shown in **figure 4**, not all vertices are reachable exactly once, denoted by an asterisk (*). If we calculate the cluster degree as the sum of degrees of vertices in a cluster, then this property is ignored. Furthermore, the cluster degree needs to be calculated without counting the edges within a cluster. We therefore present a more explicit definition of the ExF in the **methods** section.

2.2.2 Weighted Expected Force of Infection

Lawyer (2015) also states how the ExF naturally extends to weighted graphs, by considering the weight of an edge to be proportional to the likelihood of transmitting over that edge. We can then simply multiply the weights of each transmission path to calculate the (relative) probability for a path to occur. He then also redefines the cluster degree d_j in (28) as the sum of edge weights leading out from a cluster.

The WExF is especially interesting when $n < p$, since we become less dependent on sparsity. Namely, if we proportionally shrink $\hat{\mathbf{\Theta}}$, using graphical ridge, or BDMCMC, we would expect the relative node influence to remain unchanged, when using slightly different values of λ and ν, respectively. For the lasso, a slightly higher value of λ could mean the exclusion of an edge. Since an edge which is almost shrunk to zero by lasso already has a very small value, its contribution to the sum of weights was already minimal. Therefore, its exclusion should have minimal impact on the WExF.

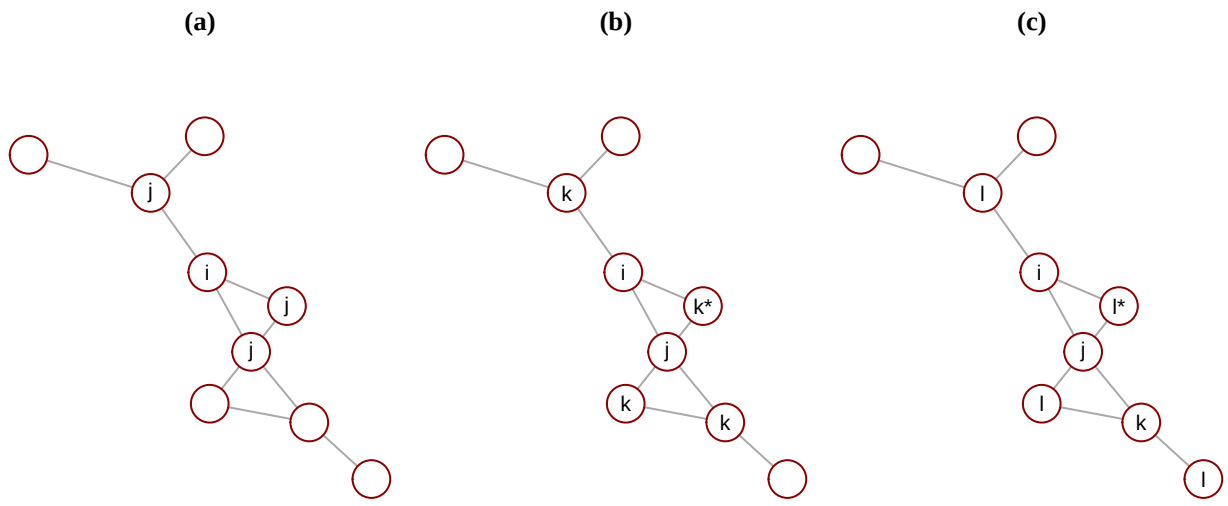


Figure 4: Example of the possible transmissions from a starting vertex (**a** → **b** → **c**). From the starting vertex v_i, the neighbourhood $N_G(v_i)$ is calculated and denoted v_j (**a**). Then, for each pair $\{v_i, v_j\}$, denote the new neighbourhoods $N_G(v_i)_{\setminus v_j}$, $N_G(v_j)_{\setminus v_i}$ as v_k (**b**). Finally after two transmissions, the neighbourhoods of every possible cluster $\{v_i, v_j, v_k\}$, are denoted as v_l (**c**). Note that some vertices are neighbours of multiple other vertices, which means they can be infected in several ways. In the figure, these are denoted by an asterisk (*).

2.3 Community Detection

In sufficiently large graphs, it can be useful to collapse strongly connected clusters to a single, new node. If no such local strong connectivity can be collapsed, it is still useful to define subsets of nodes that are more connected to each other than to nodes outside their subset. Such subsets are called communities, since their conception originates in social sciences. Various methods for the detection of communities have been proposed, some of which also allow for overlapping subsets of nodes. This could prove useful for proteomics data, which consists both of correlations between fragments of the same protein (within) and correlations between distinct proteins (between). If the within correlation is generally stronger than the between correlation, the network can be divided into communities representing the original proteins. Alternatively, clusters of proteins of the same functional group could potentially be obtained from community detection, which would also help in understanding large proteomics networks.

2.3.1 Cohesive Blocking

A connected graph is said to have connectivity or cohesion k, where k is the minimum number of unique elements in $\boldsymbol{G}$ that need to be removed to render the graph disconnected. For simplicity, we will only discuss edge cohesion, which is the number of unique edges that need to be removed to this end. For edge cohesion, the cohesion is identical to the minimum degree of (connected) vertices.

If we define proper subsets of $\boldsymbol{V}(\boldsymbol{G})$, such that these subsets have higher cohesion than the whole graph, we can create communities based on these subsets:

$$\begin{aligned} &\text{Define: } \boldsymbol{V}^* \subset \boldsymbol{V}(\boldsymbol{G}), \text{ such that:} \\ &\min(\deg(v \in \boldsymbol{V}^*)) > \min(\deg(v \in \boldsymbol{V}(\boldsymbol{G}))) \end{aligned} \tag{30}$$

The `cohesive.blocks()` function in the package `igraph` recursively identifies l-cohesive subsets as in (30) with $l > k$.

2.3.2 Hierarchical Clustering

Another method to detect communities is by defining a similarity measure for vertices and performing hierarchical clustering. The first k clusters can then be chosen as communities in the graph. Commonly, the Hamming distance (31) between the rows of the adjacency matrix in (23) is used:

$$\sum_{k=1}^{p} |a_{ik} - a_{jk}| \tag{31}$$

Where every $\boldsymbol{a}_i$ and $\boldsymbol{a}_j$, $\{i \neq j\} = 1 \dots p$ is a row of the adjacency matrix $\boldsymbol{A}$ of $\boldsymbol{\Theta}$. Altenratively, the Jaccard index (32) or the Sørensen-Dice index (33) between vertices are used:

$$\frac{|\boldsymbol{E}_i \cap \boldsymbol{E}_j|}{|\boldsymbol{E}_i \cup \boldsymbol{E}_j|} = \frac{|\boldsymbol{E}_i \cap \boldsymbol{E}_j|}{|\boldsymbol{E}_i| + |\boldsymbol{E}_j| - |\boldsymbol{E}_i \cap \boldsymbol{E}_j|} \tag{32}$$

$$\frac{2|\boldsymbol{E}_i \cap \boldsymbol{E}_j|}{|\boldsymbol{E}_i| + |\boldsymbol{E}_j|} \tag{33}$$

Where $\boldsymbol{E}_i$ and $\boldsymbol{E}_j$, $\{i \neq j\} = 1 \dots p$ are the sets of edges connected to two vertices $\{v_i, v_j\} \in \boldsymbol{V}(\boldsymbol{G})$. In the **visualisation** section, we will use the latter to attempt to decompose our resulting graph into a set of subgraphs.

3 Methods

3.1 Data

We used data from a case/control study on pancreatic cancer patients ($n = 43$) and controls ($n = 102$). Mass-spectrometry data from low ($p = 196$) and high ($p = 256$) mass protein peptides were analysed, yielding a total of $p = 452$ parameters measured in $n = 145$ individuals (Nicolardi et al. 2014). Measurements were taken in replicates of 4 and log-transformed, after which the median was taken of the replicates. For this article, we focussed on the low mass protein peptides only, since their corresponding peaks can be best discerned from one another, as will be explained in the following paragraph.

3.1.1 MALDI-FTICR

Matrix assisted laser desorption ionisiation Fourier transformed ion-cyclotron mass spectrometry (MALDI-FTICR) is an ultra high resolution mass spectrometry technique. The output of this method is a collection of peaks corresponding to different mass to charge ratios. It is especially useful for protein peptide analysis, because the large polymers correspond to such high mass to charge ratios, that conventional mass spectrometry methods fail to distinguish between the peaks formed by the numerous combinations of isotopes that can be formed in this mass to charge range. MALDI-FTICR has substantially higher resolution than conventional mass spectrometry and allows for the distinction of protein peptides, up to a certain size. Since resolution is still an issue for large peptides, we focused on the low mass range for this thesis.

3.1.2 Obtaining a Covariance Matrix

An empirical covariance matrix S was created using the centered values of the rows (the means of a person's measured peptide abundance subtracted from the measurements of individual peptides), after which the covariance matrix was standardized. Additionally, two covariance matrices were created using cases and controls separately for later analyses with the fused lasso. Both the fused lasso and ridge allow for a weighted estimation based on sample size.

3.2 Software and Hardware

All calculations were made in `R`, version 3.2.4, using the `R` Studio GUI. Packages used are listed at the reference section. We used custom functions for cross-validation, precision weighted degree and (weighted) expected force of infection. All calculations were performed on an Intel Core i7 4790k processor at 4.4GHz with 16GB of RAM.

3.3 Adaptation of the Expected Force

As mentioned in **2.2.1**, the cluster degree is ambiguous, in that it does not explicitly state which edges to include, and moreover: whether to include them once or several times. When using `igraph` to produce a graph, this is not necessarily a problem, `igraph` had a built-in function for cluster degree. However, when the input is a (possibly sparse) matrix $\boldsymbol{R}$, we have to define what this cluster degree amounts to. In this paragraph we provide an explicit definition of the cluster degree and in the next we present a possible algorithm for calculating the weighted expected force using this definition.

If we call the starting node v_i, we can denote the neighbourhood of that node as $N_G(v_i)$ and the nearest neighbours therein v_j (**figure 4a**). Then, for each pair $\{v_i, v_j\}$, we denote their new neighbourhoods, excluding the edge between them: $N_G(v_i)_{\setminus v_j}$, $N_G(v_j)_{\setminus v_i}$ and call those neighbours v_k (**figure 4b**). Finally, we denote the neighbourhoods of each set of vertices in a transmission $\{v_i, v_j, v_k\}$, namely: $N_G(v_i)_{\setminus\{v_j,v_k\}}$, $N_G(v_j)_{\setminus\{v_i,v_k\}}$ and $N_G(v_k)_{\setminus\{v_i,v_j\}}$, and call their members v_l (**figure 4c**). Note that v_l is now an element of

a multiset: If v_l is a neighbour of more than one vertex of a given transmission path $\{v_i, v_j, v_k\}$, then it can occur more than once in the set of neighbours. For simplicity, we denote the multiset of reachable vertices $N_{\text{reach}}(v_i)_c$ where $c = 1 \dots |C|$ are the possible transmission paths (clusters) of length $t = 2$.

Once we have defined all neighbours $v_l \in N_{\text{reach}}(v_i)_c$ of all possible transmission paths $\{v_i, v_j, v_k\}$, we can calculate the cluster degree for a given vertex v_i, denoted by $N_{\text{reach}}(v_i)$. We then calculate the normalised cluster degree $\bar{N}_{\text{reach}}(v_i)$ in the same manner as Lawyer (2015).

$$\bar{N}_{\text{reach}}(v_i)_c = \frac{|N_{\text{reach}}(v_i)_c|}{\sum_c |N_{\text{reach}}(v_i)_c|} \tag{34}$$

Finally, recall from the original definition that we calculate ExF as the entropy of the cluster degree, which corresponds to:

$$\text{ExF}_i = -\sum_{c=1}^{|C|} \bar{N}_{\text{reach}}(v_i)_c \cdot \log_2(\bar{N}_{\text{reach}}(v_i)_c) \tag{35}$$

3.4 Algorithm for the Weighted Influence of Nodes

We are particularly interested in the ExF, since it easily extends to weighted graphs (Lawyer 2015). Because of this, it is not necessary to induce sparsity per se, so long as the less important edges are shrunk towards zero, so that their contribution to the weighted influence is minimal.

Although sparsity is not a requisite, it speeds up calculations substantially and we therefore recommend using lFDR discussed in the section **achieving sparsity in the ridge solution** of the **regularization** section, when using ridge. For the Bayesian solution, the top absolute values can be chosen, such that the graph has a number of edges equal to the MAP estimate of the graph size (see **birth-death MCMC** in the section **Bayesian structure learning**). The lasso solution is inherently sparse.

The input of the algorithm is a $p \times p$ matrix of partial correlations $\boldsymbol{R}$, obtained by standardizing the precision matrix $\boldsymbol{\Theta}$ c.f. (1). We divide the possible transmission paths into two categories, namely: from v_i to two neighbouring vertices (first order transmission) and from v_i to a neighbour v_j to a subsequent neighbour v_k (second order transmission, **figure 4**).

Note that our implementation of the (W)ExF is currently under the restriction that all elements of $\boldsymbol{R}$ are nonnegative. This can be realised either by using the absolute values of the elements of $\boldsymbol{R}$ or by setting the negative elements to zero. We used the former, since in biological terms, a strongly repressing effect of a protein can be considered just as influential as a strongly promoting effect.

Algorithm 1: Weighted Expected Force of Infection (WExF)

input : A (possibly sparse) $p \times p$ matrix of partial correlations $\boldsymbol{R}$
output: A $p \times 1$ vector of weighted ExF of each vertex $v_i \equiv r_{ii}$
begin

define $\boldsymbol{G} \equiv (\boldsymbol{V}, \boldsymbol{E})$ from $\boldsymbol{R}$
define $N_G(v_i) \leftarrow$ which $\{r_{ij} \neq 0,\, i \neq j\}$
define $f : \{v_i, v_j\} \mapsto r_{ij}$
$r_{ij} \in \boldsymbol{R} \leftarrow |r_{ij}| \in \boldsymbol{R}\ \forall\ i, j$
for $i \leftarrow 1$ **to** p **do**
  contribution of first order transmissions
  while $\deg(v_i) \leq 1$ **do**
    $i \leftarrow i + 1$
  define $\boldsymbol{C} \equiv (\boldsymbol{c}_1\ \boldsymbol{c}_2)$ as a matrix of combinations of two neighbours $v_j \in N_G(v_i)$
  $\boldsymbol{w} \leftarrow f(v_i, \boldsymbol{c}_1)^T f(v_i, \boldsymbol{c}_2)$
  for $j \leftarrow 1$ **to** $|\boldsymbol{w}|$ **do**
    $N_{\text{reach(1)},\, ij} \leftarrow w_j \cdot \sum \left\{ f(v_i, N_G(v_i)_{\setminus\{c_{1j}, c_{2j}\}}), f(c_{1j}, N_G(c_{1j})_{\setminus\{v_i, c_{2j}\}}), f(c_{2j}, N_G(c_{2j})_{\setminus\{v_i, c_{2j}\}}) \right\}$
for $i \leftarrow 1$ **to** p **do**
  contribution of second order transmissions
  while $\deg(v_i) = 0$ **do**
    $i \leftarrow i + 1$
    $N_{\text{reach},i} \leftarrow 0$
  define $\boldsymbol{c}_1 \leftarrow \left(v_j \in N_G(v_i)\right)$
  $\boldsymbol{w}_1 \leftarrow f(v_i, \boldsymbol{c}_1)$
  for $j \leftarrow 1$ **to** $|\boldsymbol{w}_1|$ **do**
    define $\boldsymbol{c}_2 \leftarrow \left(\boldsymbol{c}_1, v_k \in N_G(\boldsymbol{c}_1)_{\setminus v_i}\right)$
    define $\boldsymbol{w} \leftarrow w_{1j} f(c_{1j}, \boldsymbol{c}_2)$
    for $k \leftarrow 1$ **to** $|\boldsymbol{w}|$ **do**
      $N_{\text{reach(2b)},\, k} \leftarrow$
      $w_k \cdot \sum \left\{ f(v_i, N_G(v_i)_{\setminus\{c_{1j}, c_{2k}\}}), f(c_{1j}, N_G(c_{1j})_{\setminus\{v_i, c_{2k}\}}), f(c_{2k}, N_G(c_{2k})_{\setminus\{v_i, c_{1j}\}}) \right\}$
    $N_{\text{reach(2a)},\, i} \leftarrow \left\{ N_{\text{reach(2a)},\, i}, N_{\text{reach(2b)},\, i} \right\}$
for $i \leftarrow 1$ **to** p **do**
  WExF of each vertex
  $N_{\text{reach},\, i} \leftarrow \left\{ N_{\text{reach(1)},\, i}, N_{\text{reach(2a)},\, i} \right\}$
  $\bar{N}_{\text{reach},\, i} \leftarrow \dfrac{N_{\text{reach},\, i}}{\sum N_{\text{reach},\, i}}$
  $\text{ExF}_i \leftarrow -\sum \bar{N}_{\text{reach},\, i} \cdot \log_2(\bar{N}_{\text{reach},\, i})$

4 Results (Regularization)

We comprehensively compare three methods for the construction of conditional independence networks: graphical lasso, graphical ridge and BDMCMC. Since the former two are closely related and involve nearly identical steps, we split the results section in two parts. First we discuss the regularization methods and then the Bayesian approach.

4.1 Determining the Support of λ

For each type of penalty, an appropriate value for λ must be chosen, such that $\hat{\boldsymbol{\Theta}}$ is non-singular. On the other hand, if λ is chosen too high, important relationships will be excluded from the final solution. In this chapter, we discuss three methods that will allow us to determine the minimum, the support and the optimum value for λ, respectively.

4.1.1 Minimum Value of λ by Condition Number

Although we could directly determine the optimum by cross-validation, this is known to generally select a value of λ that is too low, resulting in networks that are not sparse and covariance matrices that are not well-conditioned. Won et al. (2013) addressed this issue by subjecting the cross-validated value of λ to a condition number in the order of magnititude of 10^3.

The condition number of a function for a given parameter provides an indication of how much the output value of the function changes for a small change in the given parameter. In our case, this allows us to see for which value of λ in eq. (6, 11 - 13) the solution is not sensitive to small changes anymore, so that a minimum value of λ can be determined. In regularizated precision matrix estimation, the condition number is defined as the ratio between the largest and smallest Eigenvalue of $\boldsymbol{S}$ for a given amount of shrinkage (denoted $\boldsymbol{S}_\lambda$):

$$\text{cond}(\boldsymbol{S}_\lambda) = \text{abs}\left(\frac{\max(\boldsymbol{\Lambda}_{ii})}{\min(\boldsymbol{\Lambda}_{ii})}\right),\ \boldsymbol{S}_\lambda = \boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^{-1} \tag{36}$$

With the convention $\frac{\max(\boldsymbol{\Lambda}_{ii})}{0} = \infty$. Here, $\text{cond}(\boldsymbol{S}_\lambda)$ is the condition number for a covariance matrix regularized by λ. $\boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^{-1}$ is the eigenvalue decomposition of $\boldsymbol{S}_\lambda$ and the diagonal values $\boldsymbol{\Lambda}_{ii}$ the eigenvalues. We used capital Lambda for the eigenvalues to avoid confusion with the shrinkage parameter λ. The condition number of $\boldsymbol{S}$ can be seen as the reciprocal of the distance to the set of singular matrices (i.e. lower is better). In either interpretation, we can take these values to be the respective minimum values of λ for each of the penalties.

By plotting the condition number against the range of possible values for λ, we can visualize the condition number of $\boldsymbol{S}_\lambda$ against the range of the regularization parameter λ. This way, we can roughly determine the minimal value of λ, which lies beyond the 'elbow' observed in **figure 5**, because the condition number (and therefore the relative change in the solution) stabilizes beyond that point.

The curves of the condition numbers (**figure 5**) show a clear elbow, from which we drew a reference line at $\text{cond}(\boldsymbol{S}_\lambda) \leq 1.8 \cdot 10^3$, since it is in the right order of magnitude and beyond the elbow for each graph. For comparability between estimators, We took the closest integer below this threshold to be the minimal value of λ. This way, we determined the minimum values of λ to be e^{-2} and e^{-4} for the alternative type I and type II ridge estimators, respectively. The closest integer value for the archetypal I ridge estimator would be $\lambda = e^0$, but the archetypal I ridge is bound to $\lambda \in (0, 1)$, so instead we used $e^{-\frac{1}{2}}$. Finally, the archetypal II ridge estimator has its minimum around e^{-2}. The lasso requires the largest value for λ, of roughly e^0.

From **figure 5**, it is immediately clear that the alternative estimators generally yield a more well-conditioned $\boldsymbol{S}$ than the other penalties, even at very small values of λ. However, this should come as no surprise, since they resemble true ridge penalties more closely than the archetypal ridge penalties, as was shown in van

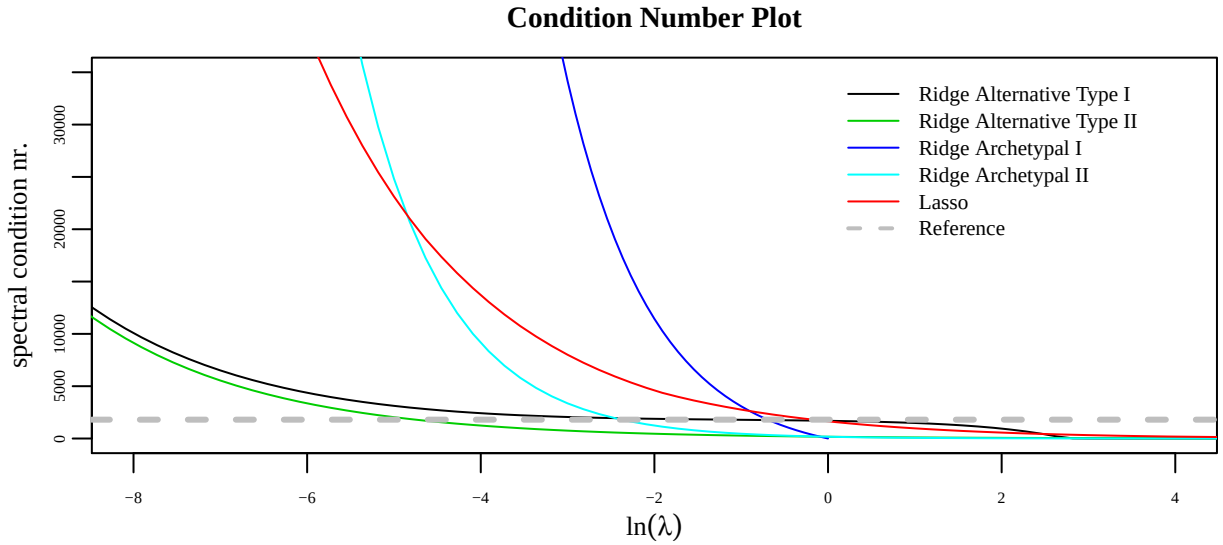


Figure 5: Condition number plots of the different regularization methods (ridge and lasso). By choosing λ sufficiently large, the condition number in (36) of $\boldsymbol{S}$ will be small enough to result in a stable estimate of $\boldsymbol{\Sigma}^{-1}$. As a rule of thumb, any value of λ beyond the elbow of the curve is sufficiently large, which we indicated by the grey dashed line (reference).

Wieringen & Peeters (2015) and summarized in (14). Interestingly however, the alternative estimators also appear to outperform the lasso penalty in terms of matrix conditioning. This could either be due to the fact that the lasso penalty is on a different scale than the ridge penalty (absolute values and squared values, respectively), or because the ridge has more an appealing interpretation than the lasso, which was conceived due to its selective proporties and not because of some inherently logical attribute.

4.2 Visualising the Support of λ with Regularization Paths

To further substantiate our choice for potential λs in ridge and lasso, we will illustrate the shrinkage of parameters as λ increases. Ideally, some parameters are shrunk to zero quickly, while others shrink more slowly, indicating the importance of some partial correlations between proteins over others.

Figures 6-8 show the shrinkage of individual elements of $\hat{\boldsymbol{\Theta}}$ as a function of λ (here on the log-scale). Three important observations can be made: First, the sum of elements of $\hat{\boldsymbol{\Theta}}$ may shrink when λ increases, but individual elements can increase proportionally to the others. This means that the relative size of precision estimates is dependent on λ, which means the choice for λ is nontrivial, even when we are not interested in the absolute size of the precision estimates. Second, it would appear that there are more negative than positive precision elements. This may be the due to the fact that positive correlations exist biologically, but are also introduced in the protein peptide fragmentation, while an inverse relationship between two fragments seems less intuitive. This would result in more positive than negative partial correlations and remember from (1) that negative precision elements correspond to positive partial correlations. Third, we see that beyond the previously established minima for λ, the estimates of the precision elements change rapidly with small changes in λ, fortifying the assertion that no value below these minima should be chosen.

If we compare the alternative type I estimator (**figure 6**, left) to the other ridge estimators, we see that the former shows greater difference in shrinkage of small and large elements. Conversely, the other estimators show a more constant shrinkage of all elements. It would appear that the alternative type I estimator has a stronger discriminative ability than the other ridge estimators. Theoretically, the ridge should produce more consistent results than the lasso, since the former has a unique solution, even when $n < p$. As in the

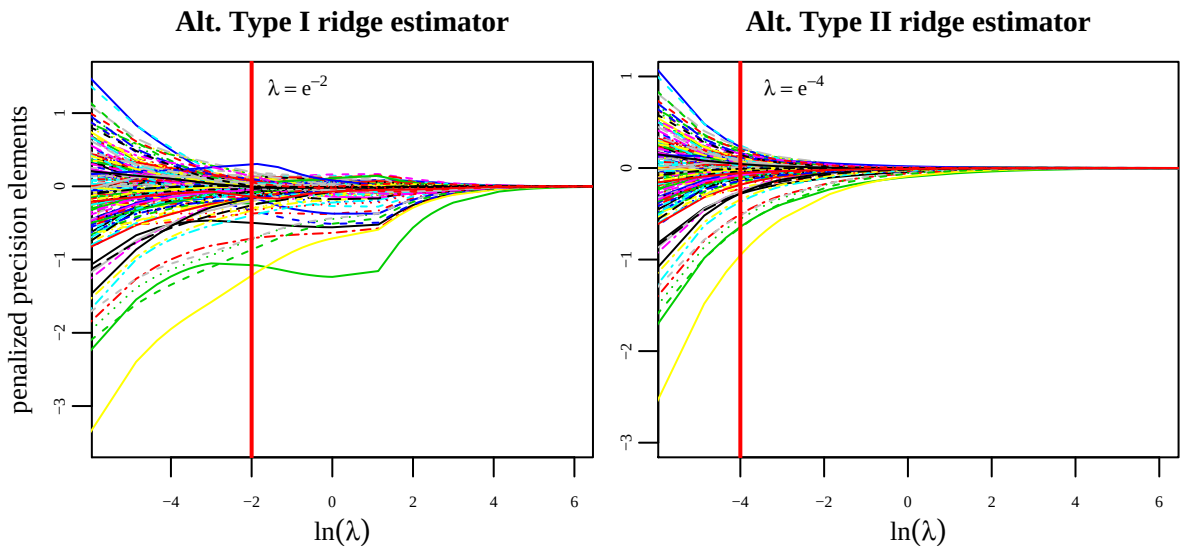


Figure 6: Visualisation of the shrinkage of individual elements of $\hat{\boldsymbol{\Theta}}$ as λ increases, using the alternative ridge estimators.

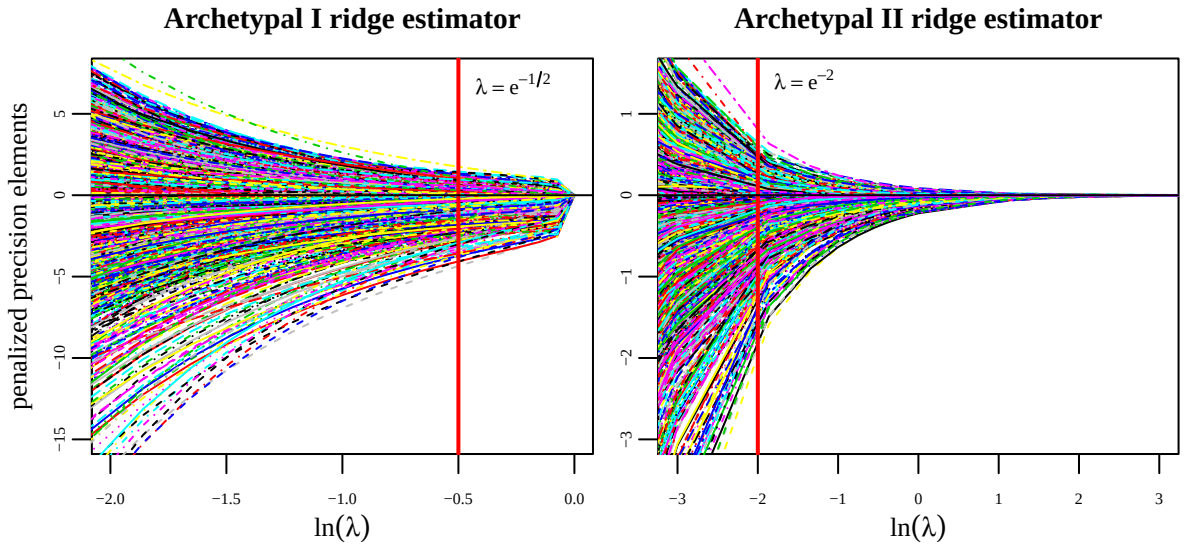


Figure 7: Visualisation of the shrinkage of individual elements of $\hat{\boldsymbol{\Theta}}$ as λ increases, using the archetypal ridge penalties. Note that the penalty of the archetypal I ridge estimator is bound to $\lambda \in (0, 1)$, as can be seen in (11).

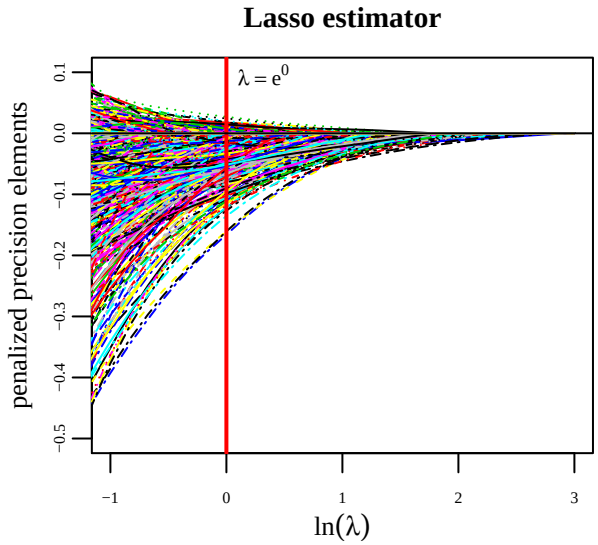


Figure 8: Visualisation of the shrinkage of individual elements of $\hat{\boldsymbol{\Theta}}$ as λ increases, using the lasso penalty.

regression case, it appears that the ridge allows for some elements to increase as others are shrunk, while the zero-elements of the lasso will always remain zero as λ increases.

4.3 Choosing the Optimal λ by Cross-Validation

With **figure 5** we determined the minimal value of λ for each estimator and **figures 6-8** show the support of λ for these estimators. Using these minima and ranges, we will attempt to obtain a cross-validated value for each of the estimators, using leave-one-out cross-validation (LOOCV). Asymptotically, this is equivalent to minimizing the Kullbach-Leibler divergence between the distribution of the data given the regularized precision matrix $\boldsymbol{Y} \sim \mathcal{N}_p(\mathbf{0}, \hat{\boldsymbol{\Theta}}^{-1})$ and the true distribution of the data $\boldsymbol{Y} \sim \mathcal{N}_p(\mathbf{0}, \boldsymbol{\Theta}^{-1})$ (Vujačić, Abbruzzo, and Wit 2015). Surprisingly, no cross-validation functions were available for the graphical lasso at the time of writing this thesis, so we implemented our own.

4.3.1 LOOCV

The package used for the estimation of $\boldsymbol{\Theta}$ using the ridge penalty (`rags2ridges`) includes built-in functions for LOOCV, using a grid search or a root-finding algorithm for the optimal value of λ in eq. (11 - 13), by leaving out one observation at a time. The cross-validated log-likelihood is then given by:

$$\ell(\boldsymbol{\Theta}|\boldsymbol{S})_{\text{LOOCV}} = \sum_{i=1}^{n} ln(|\hat{\boldsymbol{\Theta}}_{-i}|) - tr(\boldsymbol{S}_i\hat{\boldsymbol{\Theta}}_{-i}) \tag{37}$$

Where $\boldsymbol{S}_i$ denotes the sample covariance matrix of the observation left out in the calculation of $\hat{\boldsymbol{\Theta}}_{-i}$. Although (37) is the actual LOOCV log-likelihood, the negative log-likelihood is used instead in calculations due to the fact that optimizing in `R` minimizes the result of a function, rather than maximizing. Note that the `rags2ridges` package contains a computationally more efficient approximate LOOCV that minimizes the Kullbach-Leibler divergence as proposed in (Vujačić, Abbruzzo, and Wit 2015). However, for comparability with our own implementation of LOOCV for the graphical lasso, we decided to use exact LOOCV instead.

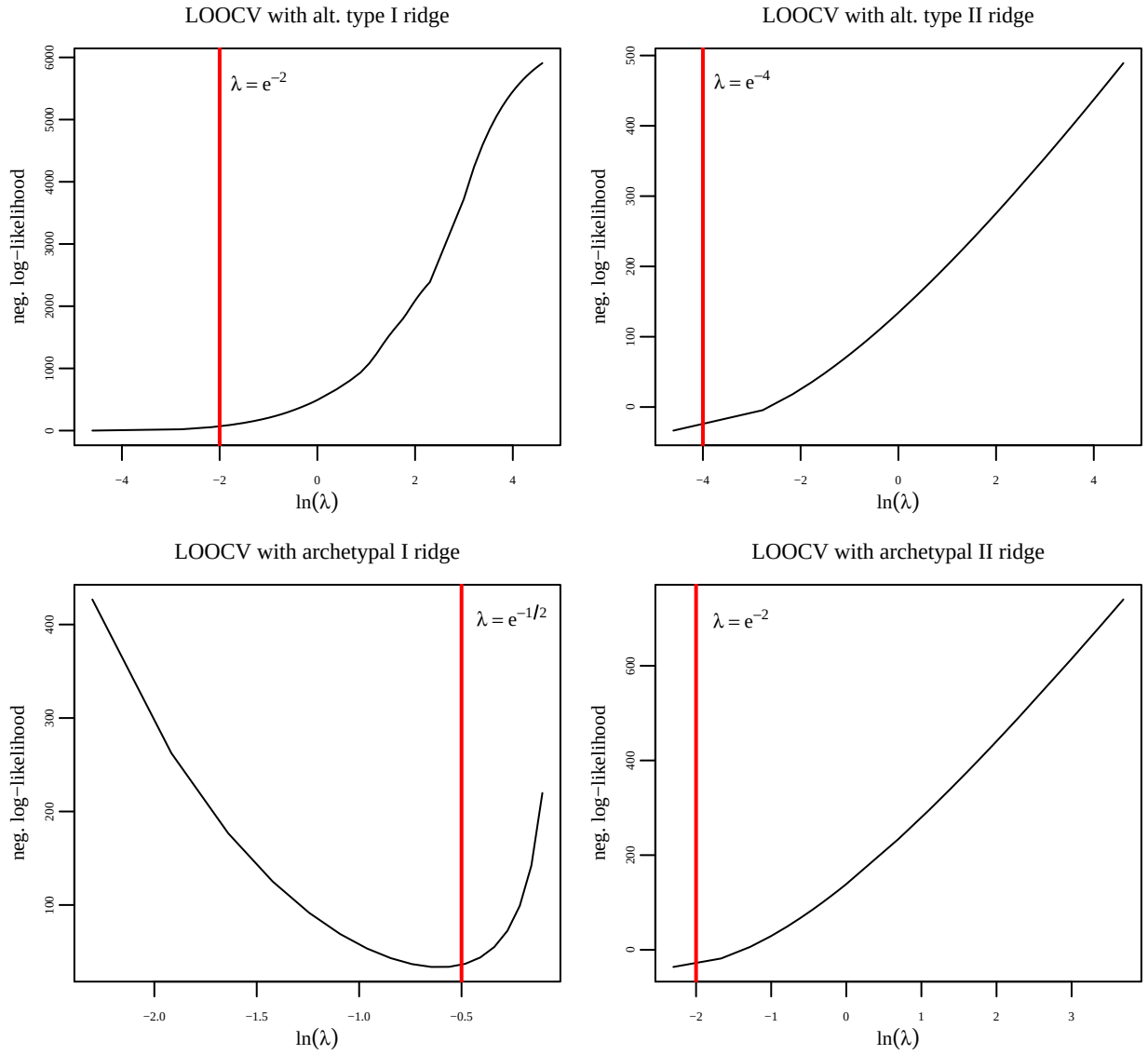


Figure 9: LOOCV negative log-likelihood over a range of values for λ, of all four ridge estimators, using the built-in function of `rags2ridges`. Although this is a concave problem, the optimum lies beyond the previously determined mimimum (red line). Furthermore, the optimum falls outside the range of the plots (except for archetypal I shrinkage), due to the fact that below the plotted values, the penalty is too low for $\boldsymbol{S}$ to be inverted and no likelihood can be calculated. Note that the archetypal I ridge estimator is on a different scale (see eq. (11), $\lambda \in (0, 1)$).

In **figure 9**, we see that LOOCV suggests an extremely small optimal value of λ for each of the estimators. The minimum even lies below the range of calculated values in three out of four penalties, due to the fact that below the plotted values, the penalty is too low for $\boldsymbol{S}$ to be inverted and no likelihood can be calculated. The only exception is the archetypal I ridge estimator, which has a minimized negative log-likelihood for $\lambda \approx e^{-0.6}$. However, this value for λ is still below the previously determined minimum of $e^{-\frac{1}{2}}$. Furthermore, we would prefer to use the alternative estimators as discussed in the introduction section, because they amount to a proper ridge penalty.

Surprisingly, even though we could not determine the optimal value with the built-in function, LOOCV suggests the use of a penalty as low as possible, which would correspond to a graph with maximal complexity. We proceed to compare LOOCV with its generalization: k-fold CV.

4.3.2 k-fold CV

To compare the preferred ridge estimator (alternative type I) to the lasso, we implemented k-fold CV and its special case LOOCV in our own function. This function uses `rags2ridges` and `glasso` to estimate $\boldsymbol{\Theta}$ with the ridge and lasso penalty, respectively.

LOOCV can be seen as a special case of k-fold CV, where each subsample has size $n = 1$. To assess whether the obtained results from LOOCV were influenced by the use of a single sample covariance matrix as in eq. (37), we also performed k-fold cross-validation with different sample sizes ($n = 2$, $n = 3$ & $n = 15$, corresponding to $k = 72$, $k = 48$ and the popular 10-fold CV). The equation for k-fold CV is given by:

$$\ell(\boldsymbol{\Theta}|\boldsymbol{S})_{k\text{-fold CV}} = \sum_{k=1}^{K} ln(|\hat{\boldsymbol{\Theta}}_{-k}|) - tr(\boldsymbol{S}_k \hat{\boldsymbol{\Theta}}_{-k}) \tag{38}$$

Where each k is a disjoint subset of equal size.

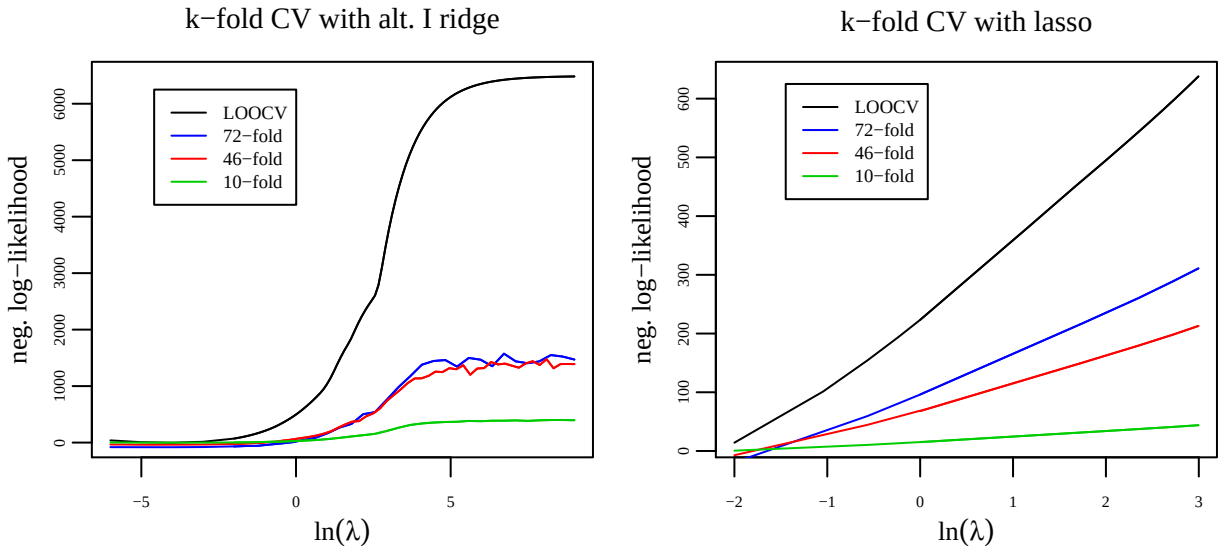


Figure 10: Cross-validated likelihood for λ for the alternative type I ridge estimator and the lasso estimator, using our own implementation of k-fold CV and its special case LOOCV. The red lines show the previously determined minimum values of λ (see **figure 3**).

In **figure 10**, our own implementation of LOOCV is shown for the alternative Type I ridge estimator and the lasso estimator. Since we implemented this function only to perform cross-validation for the lasso and to assess whether LOOCV was working correctly, we did not include the other three ridge estimators. For both

the ridge and lasso solution, LOOCV suggests a model with a minimal value for λ, resulting in a saturated graph.

It would appear that either cross-validation underperforms when $n < p$, or the true $\boldsymbol{\Theta}$ is not sparse. Before we continue to compare the results, we will consider two alternatives to cross-validation.

4.4 Alternatives to Cross-Validation

Numerous alternatives to leave-one-out, or k-fold cross-validation have been proposed to help deal with the poor performance of cross-validation techniques when $n < p$. The most commonly used alternatives include: Extended Bayesian Information Criterium (EBIC) and Stability Approach to Regularization Selection (StARS). Both of these methods base their selection on the resulting (sparse) solutions instead of the likelihood. As such, it is difficult to apply to the ridge solution, which does not induce sparsity by itself. We discuss EBIC and StARS since they are frequently used in literature, but conclude they are not suitable for our objective.

4.4.1 EBIC

The extended Bayesian information criterium (EBIC) for graphical models as proposed by Foygel and Drton (2010), is given by the following equation:

$$\text{EBIC}_\gamma = -2\ell(\boldsymbol{\Theta}|\boldsymbol{S}) + |\boldsymbol{E}|ln(n) + 4|\boldsymbol{E}|\gamma ln(p) \tag{39}$$

Where $\ell(\boldsymbol{\Theta}|\boldsymbol{S})$ is equal to the log-likelihood defined in (3) and $|\boldsymbol{E}|$ is the cardinality of $\boldsymbol{E}(\boldsymbol{G})$, constructed from a standardized precision matrix $\hat{\boldsymbol{\Theta}}$. $0 < \gamma < 1$ is a parameter to increase or decrease the extent to which EBIC scales with sparsity. When $\gamma = 0$, EBIC is identical to the ordinary Bayesian information criterion.

It is easy to see that EBIC was intended for use in graphical lasso, since it depends in great part on the number of edges in the resulting graph, which is constant for graphical ridge. Although we can reduce the number of edges in a ridge solution using the aforementioned lFDR (see **achieving sparsity in the ridge solution**), this in turn depends on a chosen threshold for the lFDR. Furthermore, both for lasso and ridge, EBIC requires choosing some value of $0 < \gamma < 1$.

We are trying to select the best value for λ, such that the resulting graph is the closest to the true underlying biological interactions. If we choose λ based on EBIC, we are ultimately shifting the choice to another, somewhat arbitrarily set parameter.

4.4.2 StARS

Introduced by Liu et al. (2010), StARS aims to use the least amount of regularization, such that the graph resulting from $\hat{\boldsymbol{\Theta}}$ is sparse, while being replicable under random subsampling. More specifically, StARS divides the original data $\boldsymbol{Y}$ into an arbitrarily large number of subsets (k) of equal size (b). Then, for each of the subsamples, a graph is constructed from the regularized precision matrix with the same value for λ. The instability is then defined as the number of times an edge is included in one of a pair of graphs, but not in the other. The total instability is then calculated for each pair of graphs and averaged over all edges. Finally, a threshold β is defined and λ is chosen such that total instability for a given value of $\frac{1}{\lambda}$ is smaller than β.

As with the EBIC, the inclusion/exclusion of edges is not a convenient way to compare graphical lasso to ridge. Furthermore, it would seem that the problem of choosing λ has shifted to the problem of choosing β. Ironically, the authors even mention this in the original paper, proposing a default value of 0.05 for β.

4.4.3 Concluding Remarks on the Alternatives to Cross-Validation

While ad hoc modifications of the graphical ridge solution could be used to implement alternatives to CV such as EBIC and StARS, we would have to choose at least two new thresholds (that of the ad hoc modification and that of the proposed technique). We conclude that this would only complicate the interpretation of the new optimal value of λ and proceed to use the cross-validated values, respecting the mininum values determined by the condition number in (36). Before comparing the resulting networks, we discuss the results from the Bayesian approach to graph construction.

5 Results (Bayesian Structure Learning)

In this chapter we discuss the results of the graphs constructed by sampling from the G-Wishart distribution shown in eq. (18 - 20), using the BDMCMC sampler shown in eq. (21, 22). We used the `R` package `BDgraph`, which has implemented the BDMCMC sampler, generally using 10^5 iterations per chain. While this takes considerable computation time (upwards of 48h single-threaded), remember that there are $\frac{196*195}{2} = 19110$ possible unique edges, which all need to be explored in the posterior. Fortunately, the package allows for multiple birth and death events to be calculated simultaneously. Following some trial and error, we set the argument to `multi.update=8`, which sped up calculations.

5.1 Prior Specification

We know from theory that for a given prior and likelihood cf. (18, 19), the BDMCMC algorithm should converge to a stationary state (Mohammadi and Wit 2015). In its current version (2.27), the package only allows for the specification of the degrees of freedom ν on the prior in (18) and the starting point: an empty graph, a full graph or a specified starting point. Note that contrasting to what Kuismin and Sillanpaa (2016) report, a higher degree of freedom yields a more diffuse prior on $\boldsymbol{\Theta}$. We believe the contrast arises from a misinterpretation of Hsu, Sinay, and Hsu (2012), who correctly describes the properties of the inverse-Wishart distribution, not the Wishart distribution. We tried several degrees of freedom for the prior of arbitrarily large size. Interestingly, both very diffuse and very narrow priors performed poorly in terms of convergence and priors with $n \leq \nu \leq p$ generally did well (**table 1**). A narrow prior can be seen as a means to induce sparsity in the posterior, as we will see in the posterior graph size. Consequently, a very weakly informative diffuse prior can be seen as the closest we have to a non-informative prior. Mohammadi and Wit (2015) stated that they intend to include more options for prior specification in future updates of the package `BDgraph`.

5.2 Assessing Convergence

To assess convergence of the algorithm, the traceplot of the number of edges $e \in \boldsymbol{E}(\boldsymbol{G})$ (or half the number of non-zero off-diagonal elements of $\hat{\boldsymbol{\Theta}}$) can be inspected. An example is shown in **figure 11** (right), where the posterior graph size stabilizes within the first few thousand iterations following the burn-in period. The plot of posterior probability of graph size also shows what appears to be central tendency around a mean size of approximately 2000-2100 elements. Using these diagnostics, the results of various prior degrees of freedom ν are summarized in **table 1**.

	MAP graph size	prior d.f.	starting point	converged	iterations
1	3422	2	empty start	no	1×10^5
2	**2065**	**145**	empty start	yes	1×10^5
3	**2330**	**196**	empty start	yes	1×10^5
4	4025	452	empty start	no	1×10^5

Table 1: BDMCMC graphs constructed with 4 different values of the degree of freedom on the Wishart prior, corresponding to the minimum ($\nu = 2$), the number of observations ($\nu = n = 145$), the number of parameters ($\nu = p = 196$) and the number of parameters in the complete dataset of Nicolardi et al. (2013) ($\nu = 452$). All graphs were estimated with 10^5 iterations, taking approximately 50h to complete each.

Likewise, **table 2** shows the convergence and posterior graph size of the BDMCMC sampler, using different starting points. We assessed convergence, starting from an empty graph ($\boldsymbol{E}(\boldsymbol{G}) = \emptyset$, $\boldsymbol{\Theta} = \boldsymbol{I}_p$), a full graph ($e = 1 \; \forall \; e \in \boldsymbol{E}(\boldsymbol{G})$, $\boldsymbol{\Theta} = \boldsymbol{1}_{p \times p}$), a lasso solution ($\boldsymbol{\Theta} = \hat{\boldsymbol{\Theta}}_{\text{lasso}}$, as obtained in (6), $\lambda = e^0$) and a ridge solution, sparsified by lFDR ($\boldsymbol{\Theta} = \hat{\boldsymbol{\Theta}}_{\text{ridge}}$, as obtained in (13a), $\lambda = e^{-2}$).

	graph size	prior d.f.	starting point	converged	iterations
1	3020	145	full start	no	1×10^5
2	**2065**	**145**	empty start	yes	1×10^5
3	**2066**	**145**	lasso start	yes	1×10^5
4	**2110**	**145**	ridge start	yes	1×10^5
5	**2330**	**196**	empty start	yes	1×10^5
6	**2371**	**196**	lasso start	yes	1×10^5
7	2268	196	lasso start	no	5×10^4
8	2352	196	ridge start	no	5×10^4

Table 2: BDMCMC graphs constructed from different starting points, corresponding to an empty graph, a full graph, a lasso solution at $\lambda = e^0$ and a (sparsified) ridge solution at $\lambda = e^{-1}$. From the table, it is clear that the BDMCMC sampler reaches its stationary state fastest from a sparse starting point. Due to time constraints, we were not able to rerun the chains in rows 7 and 8 with 10^5 iterations.

Finally, **table 3** displays the behaviour of the BDMCMC sampler under random subsampling of $\boldsymbol{Y}$. To understand the extent to which the result depends on the number of observations in our data, we defined proper subsets $\boldsymbol{Y}^* \subset \boldsymbol{Y}$ of ranging from $n = 5$ to $n = 100$. We then ran the BDMCMC algorithm with a lasso solution as starting point and a constant prior with $\nu = 196$ degrees of freedom.

	sample	graph size	prior d.f.	starting point	converged	iterations
1	n = 5	12296	196	lasso solution	no	1×10^5
2	n = 10	9180	196	lasso solution	no	1×10^5
3	n = 25	5612	196	lasso solution	no	1×10^5
4	n = 50	4190	196	lasso solution	no	1×10^5
5	n = 100	2787	196	lasso solution	no	1×10^5
6	n = 145 = N	**2371**	**196**	lasso solution	yes	1×10^5

Table 3: BDMCMC graphs constructed with random subsamples of different sizes to assess the effect on estimated graph size. All graphs were estimated with 10^5 iterations.

5.3 Choosing a Graph

Though it is technically possible to cross-validate the included edges $e \in \boldsymbol{E}(\boldsymbol{G})$ by partitioning the data, this would take extravagant computation time, since **table 3** showed us that (large) subsamples of the data converge poorly, even after 10^5 iterations. We will therefore use one of the converged chains, based on what we believe to be a good choice for prior. It would be preferable to have the degree of freedom depend on the number of variables (p) and not the number observations (n), since the true underlying structure is independent of sample size. Hence, for our final comparison of regularization and Bayesian methods, we will use a graph constructed with $\nu = 196$ degrees of freedom on the Wishart prior. Note that while the starting point of the algorithm can affect convergence significantly, the eventual result is almost identical, provided the algorithm converges at all.

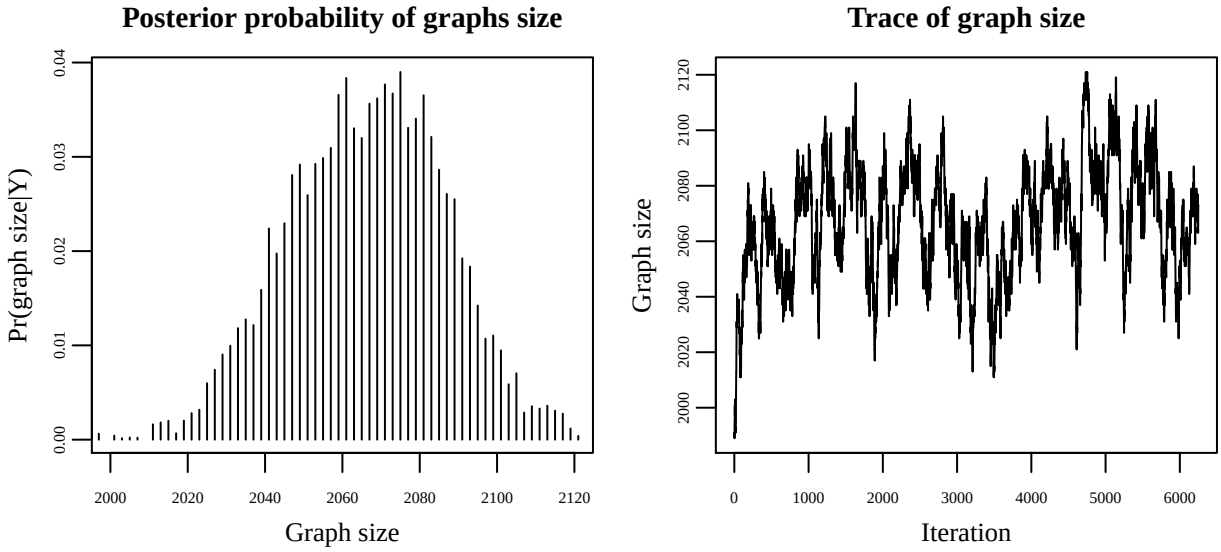


Figure 11: Example diagnostic plots of the BDMCMC chain, using the lasso solution with $\lambda = e^0$ as a starting point and $\nu = 145$ degrees of freedom on the Wishart prior. On the left, the graph size is depicted as a function of the posterior probability of a given graph. On the right, a part of the traceplot for the graph size is shown, following a $5 \cdot 10^4$ iteration burn-in period. Although there appears to be slight autocorrelation judging from the traceplot (i.e. local trends can be observed instead of apparent white noise), the mean graph size stabilizes quickly after the burn-in period.

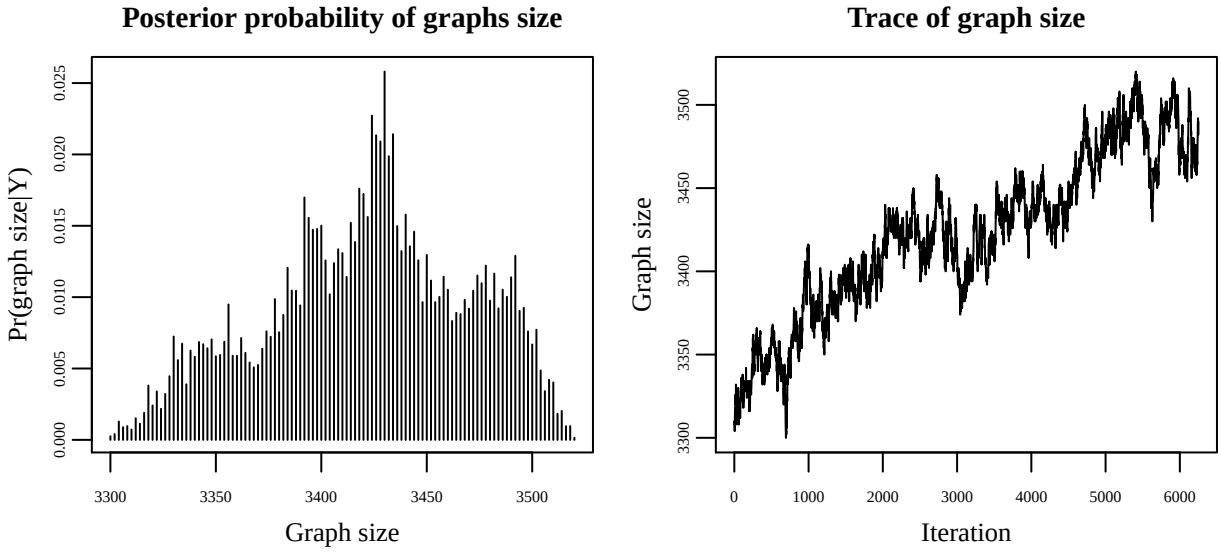


Figure 12: Example of diagnostic plots of an unconverged BDMCMC chain, using an empty graph as a starting point and $\nu = 452$ degrees of freedom on the Wishart prior. On the left, the graph size is depicted as a function of the posterior probability of a given graph. On the right, a part of the traceplot for the graph size is shown, following a $5 \cdot 10^4$ iteration burn-in period. Here we see that when the chain does not converge as there is still a trend towards higher graph size, and there is less central tendency for the posterior graph size.

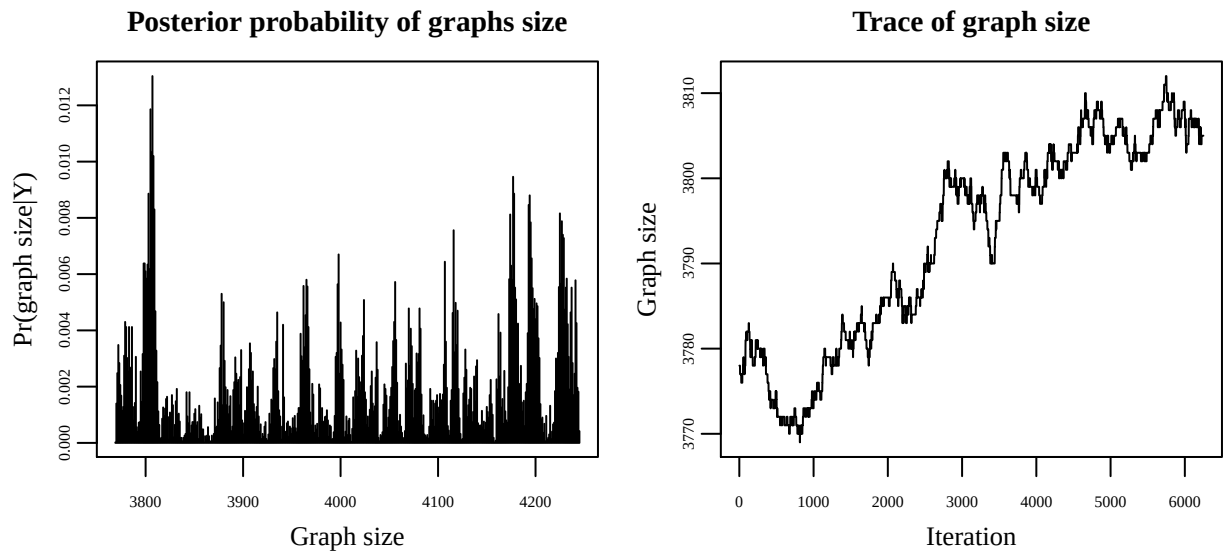


Figure 13: Example of diagnostic plots of an unconverged BDMCMC chain, using an empty graph as a starting point and $\nu = 2$ degrees of freedom on the Wishart prior. On the left, the graph size is depicted as a function of the posterior probability of a given graph. On the right, a part of the traceplot for the graph size is shown, following a $5 \cdot 10^4$ iteration burn-in period. Here we see that when the chain does not converge as there is still a clear trend towards higher graph size, and there is no central tendency for the posterior graph size.

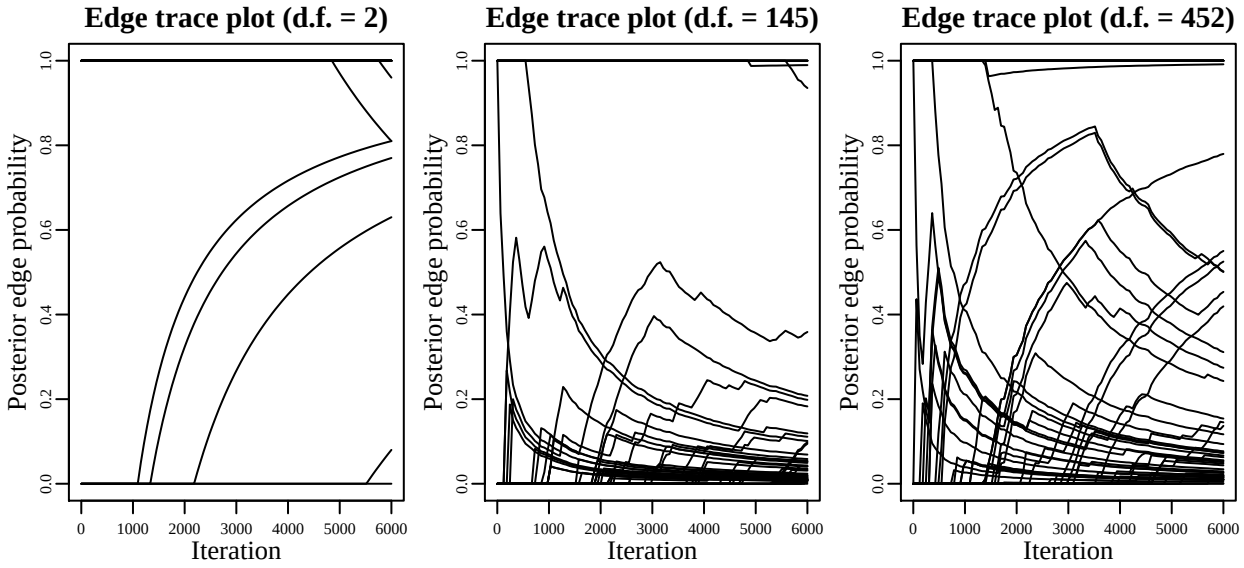


Figure 14: Coda plot of the converged (middle) and unconverged (left, right) BDMCMC chains in **figures 11 - 13**. In these plots we see how the birth and death rates of individual edges affect their posterior probability. Note that for clarity, only 100 randomly chosen edges are shown. If an edge is consistently included in the graph over multiple jumps, its probability goes towards 1. Conversely, if an edge is consistently not in the graph, its probability goes towards 0. The behaviour of the BDMCMC algorithm of the unconverged chain (right) is a reflection of the degrees of freedom on the prior ($\nu = 452$), which is too high for the algorithm to efficiently explore the posterior. Likewise, the unconverged chain on the left has degrees of freedom $\nu = 2$, resulting in very few births and death occurring at all.

6 Visualisation

Proper visualisation is a vital part of statistics, as it translates our complex methods into easy to digest summaries. Especially in the field of covariance matrix inversion, we believe that this aspect of statistics is often overlooked. Once a graph of a certain size is constructed from $\hat{\boldsymbol{\Theta}}$ using either of the three methods, visualisation can prove challenging. One way to resolve the vertex placement issue is by simply using the graphical lasso with a high penalty, so that few edges remain. However, higher regularization does not necessarily remove the least important edges in the lasso solution, as variables are not shrunk proportionally (figure **figure 8**). Alternatively, an arbitrary number of highest absolute values of the partial correlations can be displayed, but this may fail to display local strong connectivity of influential vertices. In this chapter, we describe the available methods for summarizing and decomposing large, dense graphs, when the true graph structure is preferred over sparsity and propose a new method, based on hierarchical clustering.

6.1 `igraph`

The `igraph` package contains a number of different algorithms for distributing vertices randomly (**figure 15a**), circular (**figure 15b**) or force based (**figure 15c-f**) on a 2D plane. For small graphs, a self-avoiding random distribution of vertices can be sufficient, but for larger graphs, edges tend to cross too often. A circular distribution of vertices has the appealing feature of reflecting the level of sparsity well, but can it can be difficult to inspect a single vertex and edges between adjacent vertices are practically invisible. Force based algorithms, like Fruchterman-Reingold iteratively search for placement of vertices that minimize the number of crossing edges, while avoiding large differences in edge length. Unfortunately, even algorithms specifically designed for the distribution of nodes in a way that least obscures relationships fail to produce comprehensible graphs when $|\boldsymbol{V}|$ and $|\boldsymbol{E}|$ are sufficiently large (**figure 15**). **Figures 16 - 18**, show the graphs of lasso, ridge and BDMCMC, respectively, using the Fruchterman-Reingold algorithm to illustrate the difficulty of visualisation.

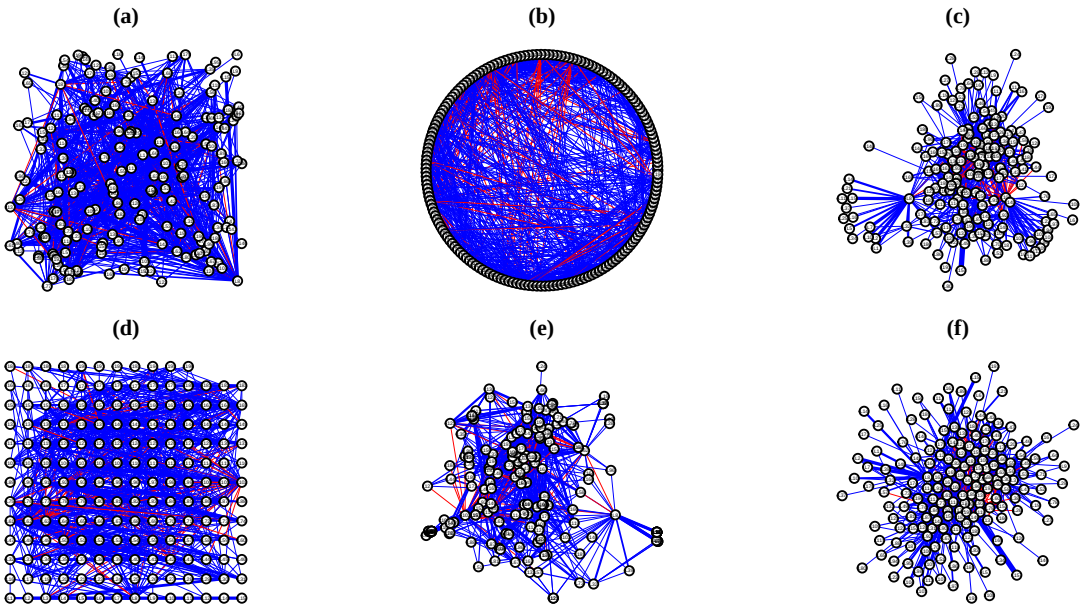


Figure 15: Six different algorithms for distributing vertices over a 2D plane, included in the `igraph` package. **(a)**: random layout. **(b)**: circular layout. **(c)**: Fruchterman-Reingold algorithm. **(d)**: Reingold-Tilford algorithm. **(e)**: Large Graph Layout algorithm. **(f)**: graphopt algorithm. Edge line width is proportional to partial correlation and red lines represent negative correlation. In this example, graphical lasso with $\lambda = e^0$ was used.

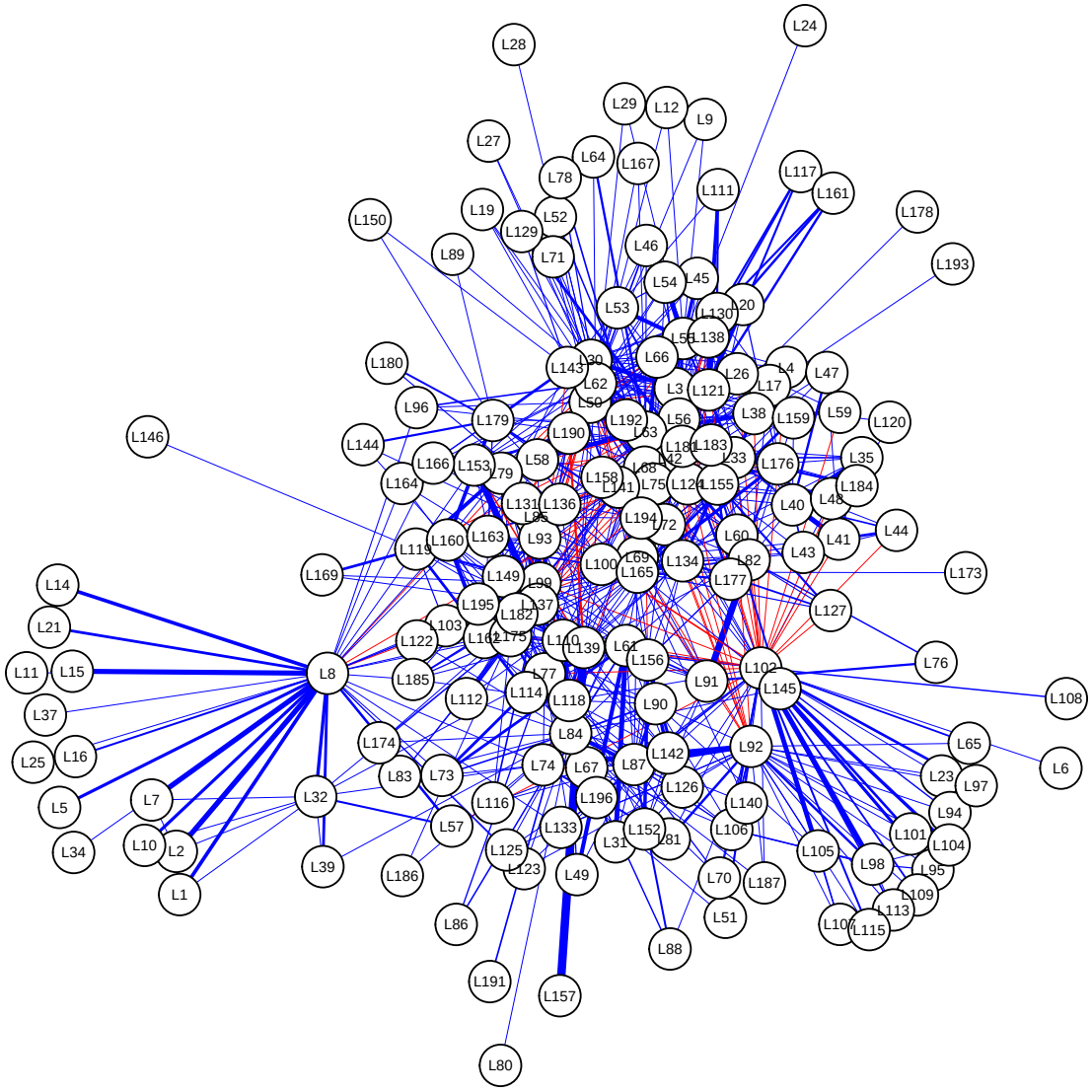


Figure 16: Graphical lasso with $\lambda = \exp^0$ visualized using the Fruchterman-Reingold algorithm included in the `igraph` package. Blue and red lines represent positive and negative partial correlations, respectively. Furthermore, the line width of edges is proportional to the partial correlation. Even though the lasso induces sparsity on its own, the graph is still too large to represent using conventional methods.

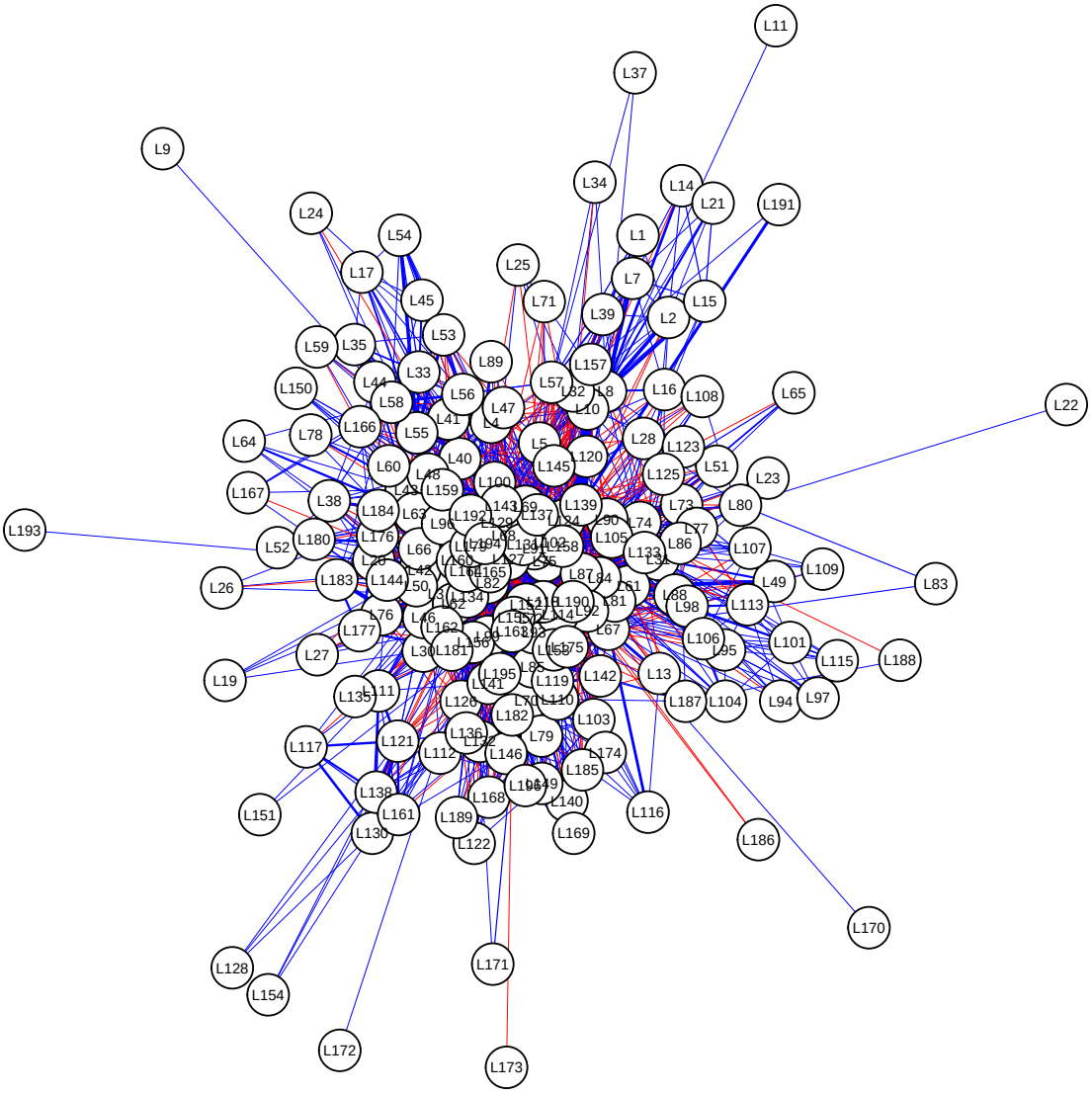


Figure 17: Graphical ridge with $\lambda = \exp^{-2}$ visualized using the Fruchterman-Reingold algorithm included in the `igraph` package. Blue and red lines represent positive and negative partial correlations, respectively. Furthermore, the line width of edges is proportional to the partial correlation. Even after using the lFDR to induce sparsity in the ridge solution, the graph is still too large to represent using conventional methods.

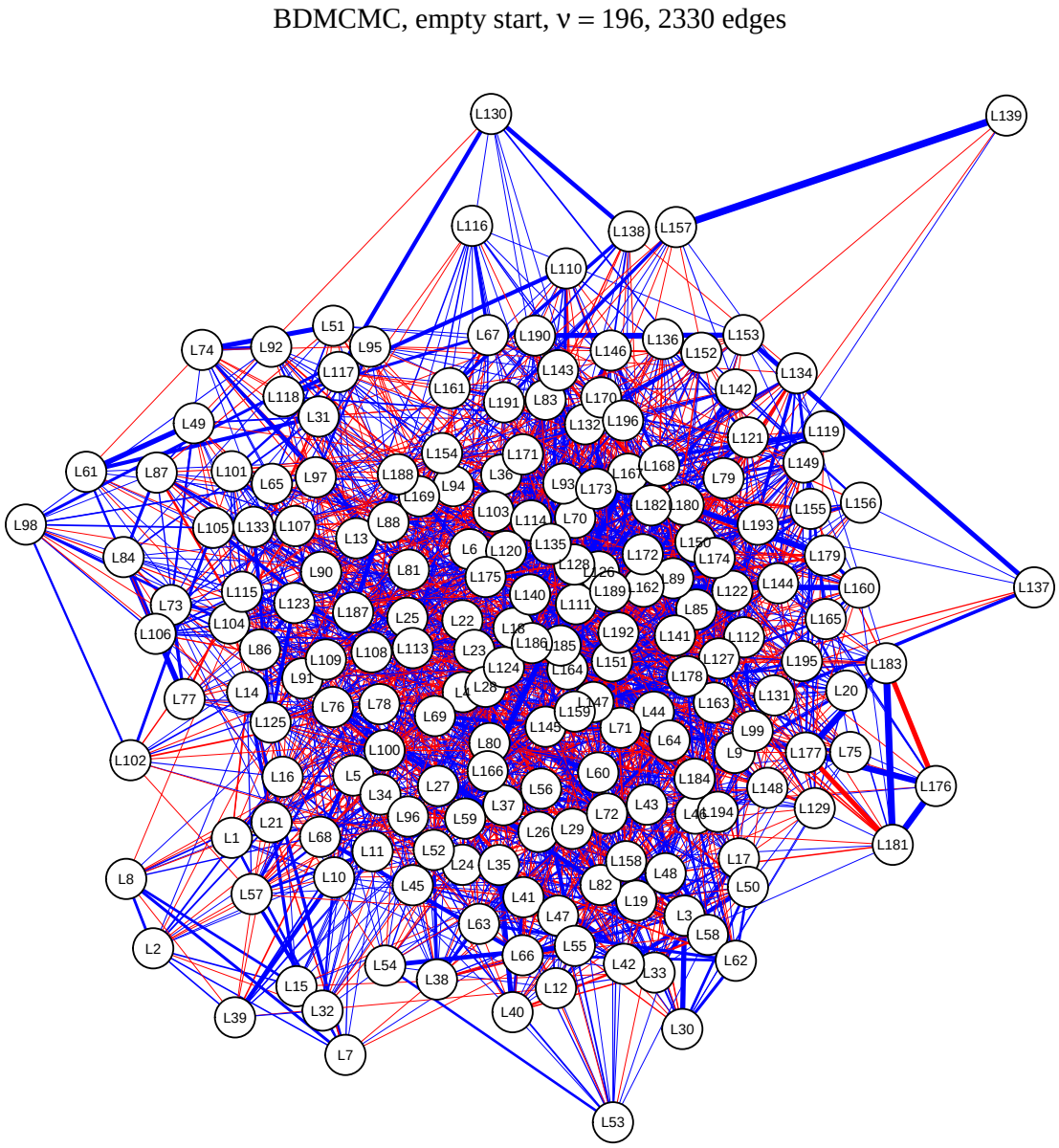


Figure 18: BDMCMC, using the estimate of $\mathbf{\Theta}$, thresholded to the MAP graph size. A lasso solution was chosen as starting point and $\nu = 196$ prior degrees of freedom. The resulting graph is visualized using the Fruchterman-Reingold algorithm included in the `igraph` package. Blue and red lines represent positive and negative partial correlations, respectively. Furthermore, the line width of edges is proportional to the partial correlation. The iterative nature of the procedure is reflected in the graph as many vertices with high degree and low edge weights.

6.2 Centrality Measures

In **figure 19**, the betweenness, closeness and Eigenvalue centrality described in eq. (25 - 27) are shown. The highest scoring values are labelled in each of the graphs and a summary is shown in **table 4**, revealing large differences both between the construction methods (lasso, ridge, Bayesian) and between the centrality measures. Although the lasso is notorious for somewhat randomly selecting one among a number of correlated variables, we did not expect large inconsistencies between the ridge and Bayesian solutions. We will therefore assess relative importance of vertices using the alternative to centrality measures: node influence metrics.

	lasso			**ridge**			**BDMCMC**		
	betw.	**close.**	**EVc**	**betw.**	**close.**	**EVc**	**betw.**	**close.**	**EVc**
1	L102	L102	L102	L158	L158	L155	L128	L6	L177
2	L8	L8	L84	L139	L92	L102	L6	L128	L183
3	L165	L142	L62	L92	L102	L69	L26	L186	L181
4	L142	L162	L137	L67	L139	L165	L188	L26	L176
5	L162	L119	L92	L102	L129	L158	L36	L178	L135
6	L92	L175	L87	L69	L137	L137	L178	L188	L179
7	L119	L59	L77	L155	L87	L194	L27	L36	L162
8	L118	L118	L165	L156	L67	L134	L186	L18	L180
9	L62	L93	L175	L137	L69	L143	L94	L27	L143
10	L84	L23	L50	L194	L82	L75	L147	L154	L174

Table 4: Highest scoring vertices for different centrality measures and for each of the three estimation methods. Abbreviations: betweenness (betw.), closeness (close.), Eigenvalue centrality (EVc).

6.3 Node Influence

Since centrality measures displayed large inconstistencies both among each other and between the graphs, we will now assess the performance of the weighted expected force of infection (WExF), presented in the methods section (**algorithm 1**). **Figure 20** shows the WExF of each vertex in the lasso, ridge and Bayesian solution. A summary of the highest ranking vertices in terms of WExF is shown below in **table 5**. Interestingly, the BDMCMC graph show no consistency with the results found in the regularized solutions. Many of the central vertices in **table 4** are also identified as relatively important. Note that the overall WExF increases with graph denseness, as vertices can reach each other more easily.

	lasso	**ridge**	**BDMCMC**
1	L62 (9.06)	L158 (11.13)	L22 (10.08)
2	L102 (9.04)	L155 (10.94)	L71 (9.79)
3	L175 (8.7)	L102 (10.93)	L151 (9.77)
4	L165 (8.42)	L194 (10.74)	L186 (9.73)
5	L92 (8.34)	L92 (10.46)	L135 (9.72)
6	L137 (8.33)	L69 (10.27)	L128 (9.65)
7	L99 (8.14)	L143 (10.22)	L18 (9.57)
8	L75 (8.09)	L93 (10.19)	L28 (9.56)
9	L63 (8.08)	L164 (10.17)	L4 (9.55)
10	L84 (7.95)	L165 (10.17)	L189 (9.55)

Table 5: Highest scoring vertices for the weighted expected force of infection (WExF) and for each of the three estimation methods. Contrary to centrality measures, WExF is a measure of relative node influence and thus representable for the importance.

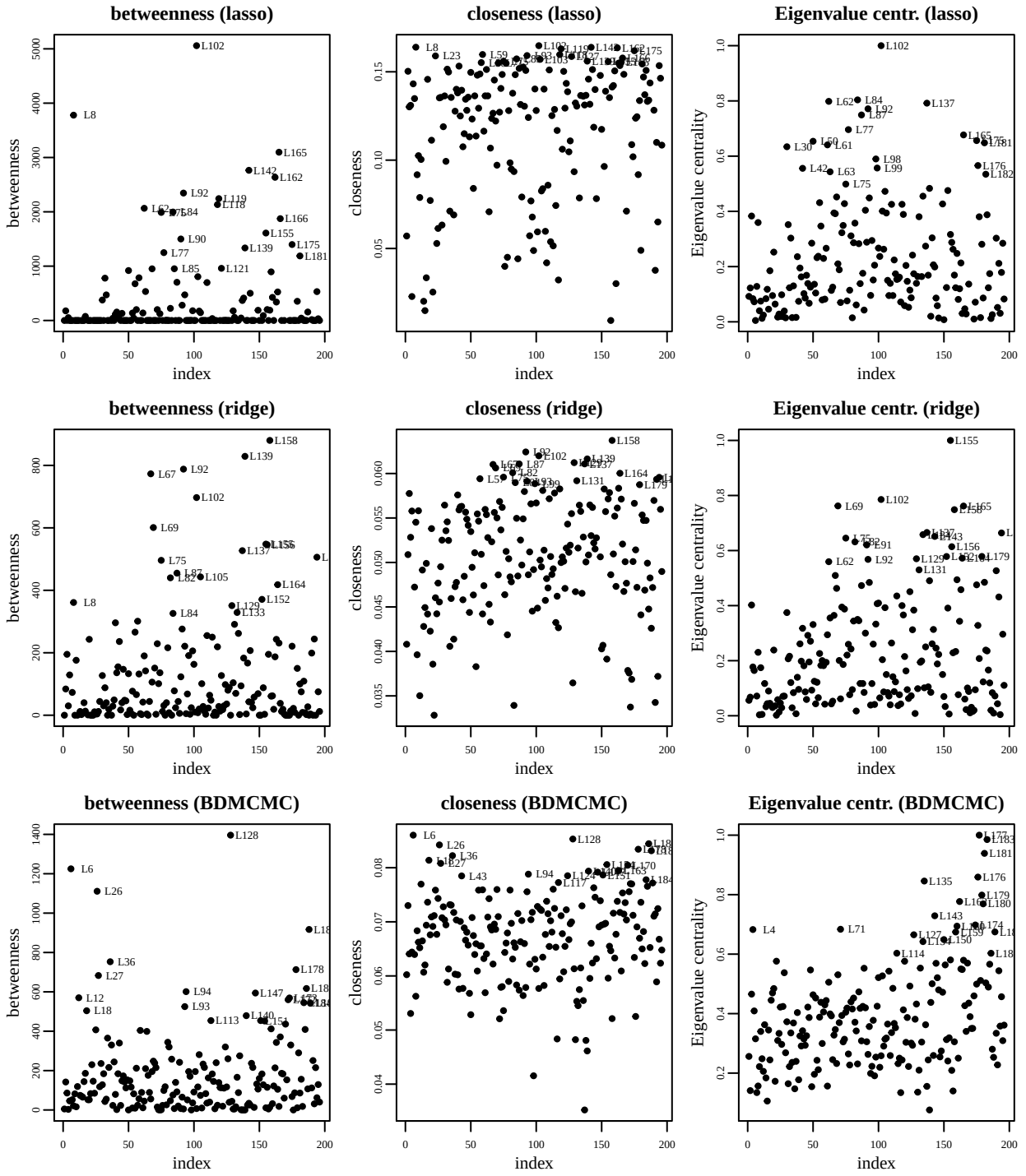


Figure 19: Centrality measures of all three resulting graphs (lasso, ridge, Bayesian). The most central points in the graph according to either of the three methods described in eq. (25 - 27) do not appear to be consistent over different graphs. This lack of consistency centrality suggest that the graphs are very uncertain. As a result, centrality measures poorly reflect the influence of nodes.

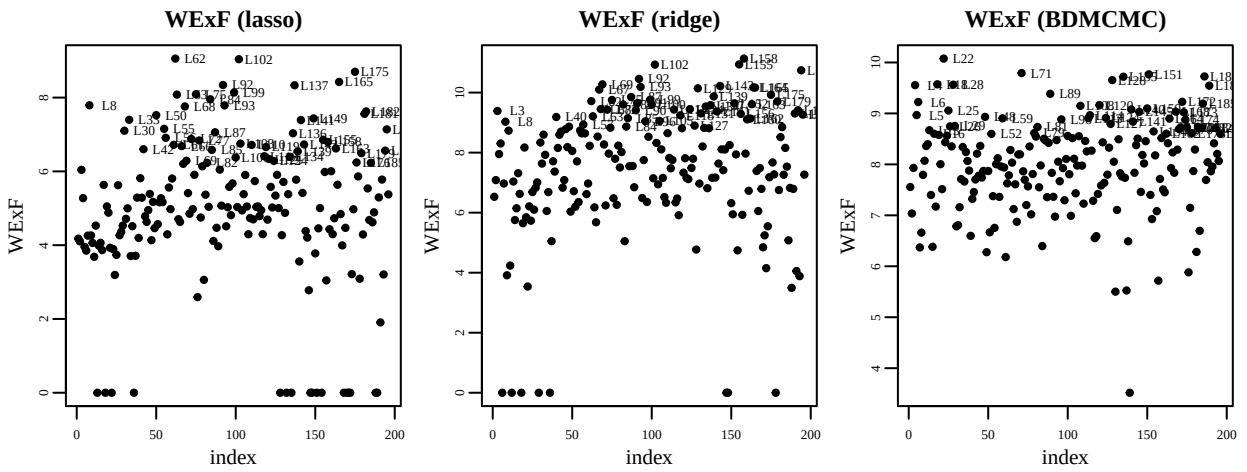


Figure 20: WExF of all three resulting graphs (lasso, ridge, Bayesian). There appears to be greater consistency in node influence between the regularized graphs than as determined by centrality measures. Both regularized graphs still display no similarity to the Bayesian approach.

6.4 Community Detection

Figure 21 shows the estimated communities from the three graphs in **figures 16-18**. Due to the relative denseness of the graphs, no disjoint communities could be detected. However, there appears to be hierarchical structure in the connectivity of the graphs, as the estimated communities are all nested in each other.

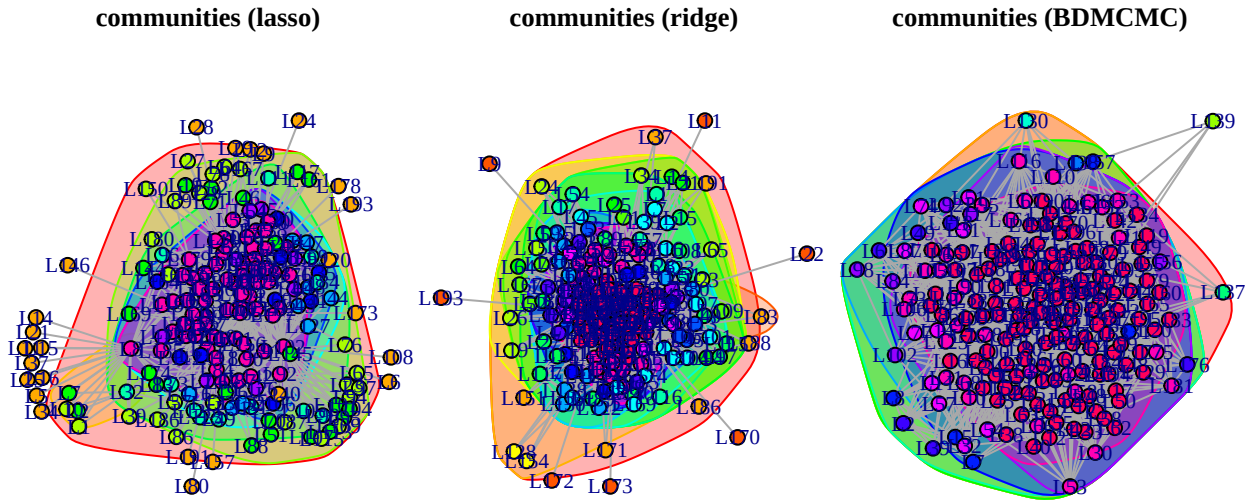


Figure 21: Community detection of each of the three graphs. There appears to be a strong hierarchical structure in the graph, with local high cohesion only present nested in other clusters.

6.5 Hierarchical Clustering

Figure 22 shows the hierarchical clustering of nodes, using the Sørensen-Dice index of similarity between nodes in eq. (33). Complete linkage was used to create the dendrogram, but as shown in the appendix, Ward's minimum variance criterium also performed reasonably well. (include in appendix, Ward 1963) Especially for the ridge and Bayesian solution, distinct hierarchical clusters are clearly visible. In the following step, we will decompose the graphs into their respective hierarchical clusters.

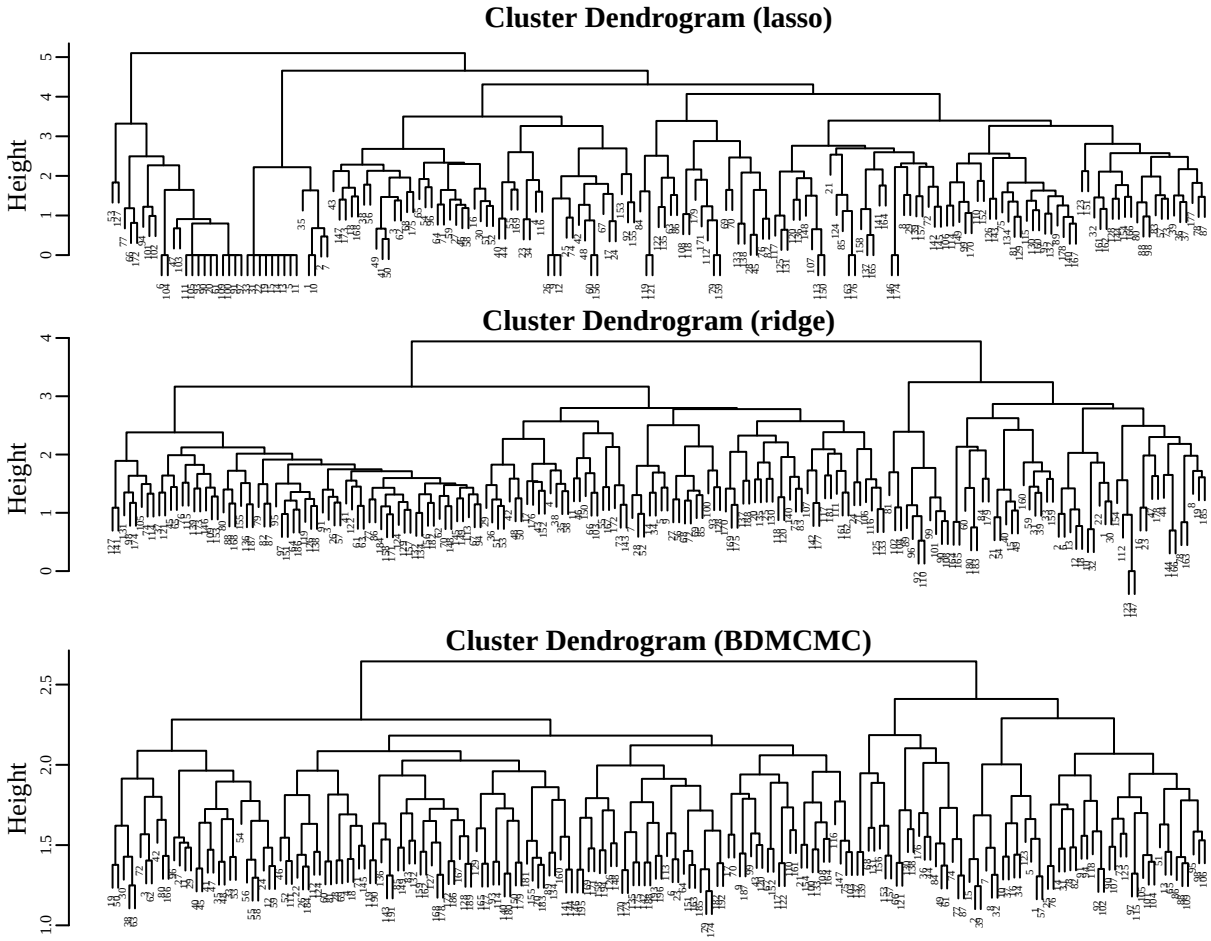


Figure 22: Hierarchical clustering of vertices based on the Sørensen-Dice index of similarity (33), using complete linkage. We cut the dendrograms such that 6 clusters were defined, which we can then use to decompose the graphs in **figures 16 - 18**.

6.6 Hiveplot

Our final visualisation step uses a relatively new technique to visualize large graphs called Hiveplot. [reference] An appealing feature of these plots is that we can define any number of axes and assign vertices to those axes based on any property, such as hierarchical clustering. The plot then assigns the vertices on an axis such that the vertices with the highest degree are the most outward, which yields a relatively clear plot of the graph, compared to the conventional force based methods in **figure 15c-f**. A popular example of the use of Hiveplot is by Yan et al. (2010), who compared the gene regulatory network of *E. coli* to the call graph of a Linux operating system kernel.

To decompose the graphs in **figures 16 - 18**, we defined 6 axes for each graph, corresponding to their respective hierarchical clusters in **figure 22**. For visualisation, we made the node size and color proportional to the WExF in **algorithm 1** and defined edge transparancy based on the partial correlations in $\boldsymbol{R}$, obtained from $\hat{\boldsymbol{\Theta}}$ (1). The results are shown in **figures 23 - 25**.

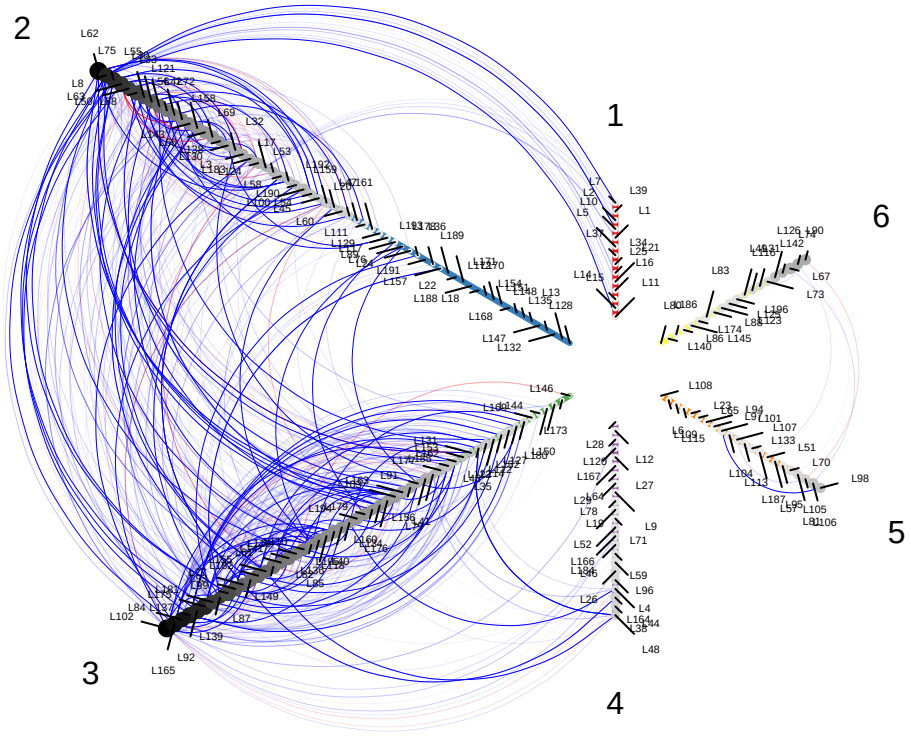


Figure 23: Hiveplot of the graphical lasso with $\lambda = e^0$. Position on the axes is proportional to the degree of the vertex, such that vertices with more edges are always on the outside on the graph. Weaker partial correlations are more transparent, such that emphasis lies on the important relationships.

The hiveplots are much clearer than the Fruchterman-Reingold plots in **figures 16 - 18**. In fact, the vertices were assigned a label at alternating distance, such that each vertex can be found in the hiveplots, without the overlapping vertices and edges in the Fruchterman-Reingold plots. Another interesting observation in each of **figures 23-25**, is how vertices on the first axes are substantially more influential than on the latter axes. Finally, compared to ridge and BDMCMC, the Hiveplot of the graphical lasso (**figure 21**), appears to display that the lasso favors shrinkage of negative partial correlations (positive elements of $\boldsymbol{\Theta}$). Note however, that the minimum amount of regularization for lasso is much higher than for ridge, resulting in fewer edges overall, leading to this artificial observation.

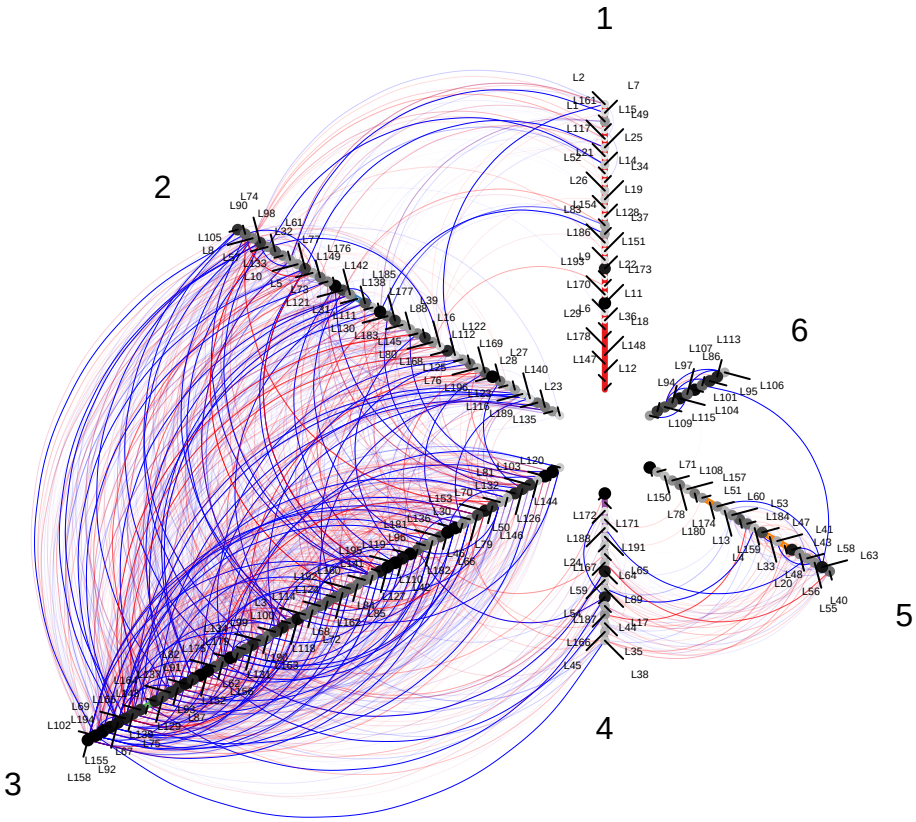


Figure 24: Hiveplot of the graphical ridge with $\lambda = e^{-2}$. Position on the axes is proportional to the degree of the vertex, such that vertices with more edges are always on the outside on the graph. Weaker partial correlations are more transparent, such that emphasis lies on the important relationships.

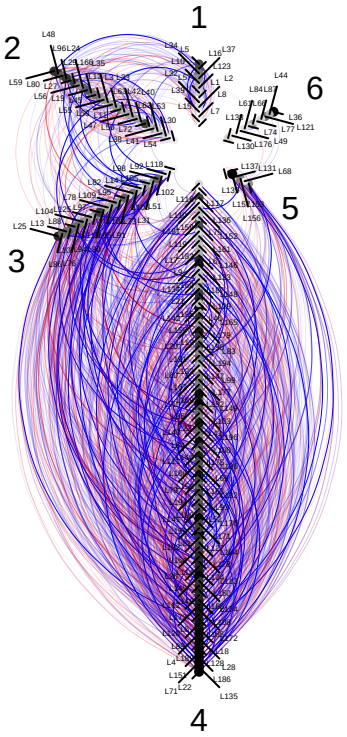


Figure 25: Hiveplot of the BDMCMC chain with $\nu = 196$. Position on the axes is proportional to the degree of the vertex, such that vertices with more edges are always on the outside on the graph. Weaker partial correlations are more transparent, such that emphasis lies on the important relationships.

7 A Closer Look at the Differences in Graphs

In this chapter we will attempt to quantify the differences in terms of rank correlation of the WExF of nodes between graphs. Then, we will make a comparison of the mass to charge ratios (m/z) of peaks in our MALDI-FTICR data and compare them to a list of proposed sequences, to get an indication of the possible proteins involved in the graphs. In doing so, we hope to shed some light on the actual differences in results between lasso, ridge and BDMCMC for graph construction.

7.1 Quantifying the Differences between Graphs

The WExF helps us identify influential vertices and we showed that it should in principle be more consistent between methods than measures of centrality. We will now plot the rank of vertices in terms of WExF against each other and compute the (Spearman) rank correlation to give some numeric indication of the size of differences between graphs. Of course, other methods to compare networks exist, like the Hamming distance in (31). However, most of such measures come from set theory, merely comparing inclusion of edges (as if the graph were unweighted). By comparing the rank in terms of influence of vertices, we take into account the information of the strengths of the edges.

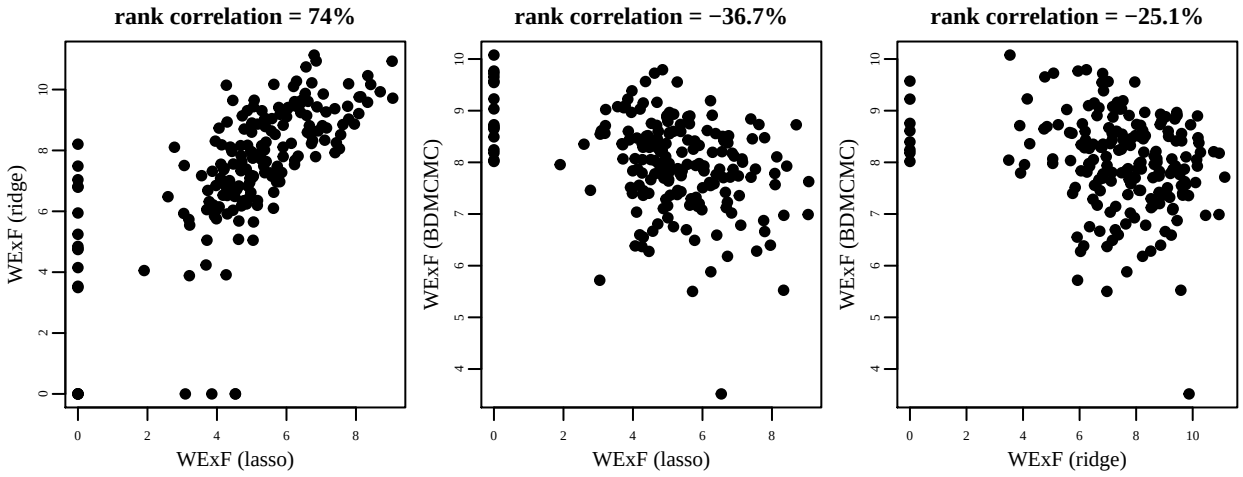


Figure 26: Here we attempt to quantify the difference between the three graphs by calculating the Spearman rank correlation of vertices' WExF in each of the three methods used for estimating $\hat{\mathbf{\Theta}}$.

In **figure 26**, we confirm that the regularized solutions are much more alike than the BDMCMC algorithm, which even correlates negatively in terms of WExF rank. From the figure, this would appear to be the case due to a number of vertices which have near 0 influence in the regularized solutions and, contrastingly, have very high influence in the BDMCMC graph. In the appendix, we give an example of the rank correlation between graphs of different sizes using the same method (by means of thresholding using lFDR), which was generally a lot higher ($> 80\%$).

7.2 Identified Peptides

With the help of the proteomics group at the LUMC, a select number of vertices were identified from the original MALDI-FTICR data by comparing them to a list of proposed sequences from Shen (2010). Due to the aforementioned difficulty of uniquely identifying protein peptides from mass spectrometry data (see the **methods** section), not all vertices could be identified. The results of the identification are shown in **table 6**.

Proposed protein	Proposed sequence	Exp. m/z	Obs. m/z	ppm diff.	Variable
Complement C3 precursor	SSKITHRIHWESASLLR	2021.10	2021.10	20.20	L102
Complement C3 precursor	SSKITHRIHWESASLL	1865.00	1865.00	19.70	L87
Complement C3 precursor	SKITHRIHWESASLL	1778.00	1778.00	19.10	L84
Complement C3 precursor	ITHRIHWESASLLR	1718.90	1719.00	19.30	L81
Complement C3 precursor	THRIHWESASLLR	1605.90	1605.90	18.80	L74
Complement C3 precursor	SKITHRIHWESAS	1551.80	1551.80	17.90	L70
Complement C3 precursor	HRIHWESASLLR	1504.80	1504.80	21.60	L65
Complement C3 precursor	IHWESASLLR	1211.70	1211.70	17.60	L31
Complement C3 precursor	TGLQEVEVKAAVYHHFISDGVRK	2583.40	2583.40	13.70	L136
Complement C3 precursor	QGTPVAQMTEDAVDAERLK	2059.00	2059.10	44.70	L107
Complement C3 precursor	RIPIEDGSGEVVLSR	1626.90	1626.90	-2.10	L76
Complement C4-A	LLLFSPSVVHLGVPLSVGVQLQDVPR	2769.60	2769.50	-29.50	L156
ITIH4	PGVLSSRQLGLPGPPDVPDHAAYHPF	2724.40	2724.40	22.00	L149
ITIH4	SRQLGLPGPPDVPDHAAYHPFR	2427.20	2427.30	15.50	L126
ITIH4	SRQLGLPGPPDVPDHAAYHPF	2271.10	2271.20	19.80	L119
ITIH4	RQLGLPGPPDVPDHAAYHPF	2184.10	2184.10	20.80	L116
ITIH4	QLGLPGPPDVPDHAAYHPF	2028.00	2028.00	19.90	L103
ITIH4	PPDVPDHAAYHPFR	1618.80	1618.70	-50.40	L75
ITIH4	PGPPDVPDHAAYHPF	1616.70	1616.70	-37.40	L75
ITIH4	GPPDVPDHAAYHPF	1519.70	1519.70	10.40	L66
ITIH4	VPDHAAYHPFR	1309.60	1309.60	-50.20	L46
ITIH4	NVHSGSTFFKYYLQGAKIPKPEASFSPR	3156.60	3156.70	24.60	L175
ITIH4	NVHSAGAAGSRMNFRPGVLSS	2115.10	2115.10	21.30	L112
ITIH4	EKNGIDIYSLTVDSR	1709.90	1709.90	-3.40	L80
ITIH4	YLQGAKIPKPEASFSPR	1889.00	1889.00	1.10	L90
ITIH2	SILQMSLDHHIVTPLTSLVIENEAGDER	3117.60	3117.50	-31.60	L173
Fibrinogen alpha chain	SSSYSKQFTSSTSYNRGDSTFESKSY	2931.30	2931.40	24.60	L165
Fibrinogen alpha chain	SSSYSKQFTSSTSYNRGDSTFESKS	2768.20	2768.30	23.60	L155
Fibrinogen alpha chain	SSSYSKQFTSSTSYNRGDSTFESK	2681.20	2681.30	21.60	L145
Fibrinogen alpha chain	SETESRGSESGIFTNTKESSSHHPGIAEFPSRG	3505.60	3505.70	25.50	L194
Fibrinogen alpha chain	SETESRGSESGIFTNTKESSSHHPGIAEFPSR	3448.60	3448.70	19.50	L193
Fibrinogen alpha chain	SYKMADEAGSEADHEGTHSTKRGHAKSRPV	3239.50	3239.60	26.40	L179
Fibrinogen alpha chain	REYHTEKLVTSKGDKELR	2189.20	2189.10	-14.80	L117
Fibrinogen alpha chain	ADSGEGDFLAEGGGVR	1536.70	1536.70	18.60	L68
Fibrinogen beta chain	GHRPLDKKREEAPSLRPAPPPISGGGY	2882.50	2882.60	23.90	L163
APOA4	AQDTQEKLNHQLEGLTFQMKKNAEELKA	3242.60	3242.60	-12.40	L179
APOA4	PYAQDTQEKLNHQLEGLTFQMK	2619.30	2619.20	-24.80	L140
APOA4	ARLLPHANEVSQKIGDNLRELQQ	2629.40	2629.40	-5.20	L141

APOA4	ARLLPHANEVSQKIGDNL	1975.10	1975.10	32.80	L94
APOA4	GLQKSLAELGGHLDQQVEEFR	2354.20	2354.20	-2.70	L121
APOA4	YADEFKVKIDQTVEELR	2083.10	2083.00	-25.80	L110
APOA4	RVEPYGENFNK	1352.70	1352.70	0.30	L50
Pigment epithelium-derived factor	ISSPDIHGTYKELLDTVTAPQKNLK	2768.50	2768.50	9.80	L156
Pigment epithelium-derived factor	YDLISSPDIHGTYKELLDTVTAPQKNLK	3159.70	3159.70	19.50	L175
Pigment epithelium-derived factor	IVFEKKLR	1032.70	1032.60	-32.20	L6
Pigment epithelium-derived factor	IVFEKKLR	1032.70	1032.70	31.00	L5
Pigment epithelium-derived factor	VLTGNPRLDLQEINNWVQAQMKGK	2752.50	2752.50	21.90	L152
Alpha-2-HS-glycoprotein	HTFMGVVSLGSPSGEVSHPR	2081.00	2081.00	-6.60	L110
Alpha-2-HS-glycoprotein	PSGEVSHPRKT	1194.60	1194.50	-62.50	L27
Leucine-rich alpha-2-glycoprotein	TLDLGENQLETLPPDLLR	2037.10	2037.10	26.60	L104
Leucine-rich alpha-2-glycoprotein	ALGHLDLSGNRL	1265.70	1265.60	-56.00	L42
Histidine-rich glycoprotein	KQANKALEKYKEENDDFASFR	2531.30	2531.30	30.60	L131
IGFBP-3	SAGSVESPSVSSTHR	1487.70	1487.70	-29.90	L63
IGFBP-3	VSDPKFHPLHSKIIIIKKGHAKDSQRY	3142.80	3142.70	-25.50	L174
Vitronectin	TSAGTRQPQFISR	1448.80	1448.70	-56.80	L60
Vitronectin	GVPGQVDAAMAGRIYISGMAPRPSLAKKQRF	3272.80	3272.70	-8.30	L182
Prothrombin	TATSEYQTFFNPR	1561.70	1561.80	17.60	L72
Prothrombin	SLEDKTERELLESYIDGR	2153.10	2153.10	21.10	L114
Antithrombin-III	ANRPFLVFIREVPLNTIIFMGR	2603.50	2603.40	-35.60	L137
Gelsolin	AQPVQVAEGSEPDGFWEALGGK	2272.10	2272.20	41.10	L119
Carboxypeptidase B2	SKSKDHEELSLVASEAVR	1985.00	1985.00	-1.60	L96
Ceruloplasmin	LYKKALYLQYTDETF	1896.00	1896.10	47.20	L91
Calmodulin	VFDKDGNGYISAAELR	1754.90	1754.90	-6.10	L83
Alpha-2-antiplasmin	NPNPSAPRELKEQQ	1607.80	1607.90	51.80	L74
Kininogen-1	RHDWGHEKQR	1348.70	1348.70	55.40	L49

Table 6: Proposed aminoacid sequences and proteins and ppm difference from the observed peaks.

7.3 Qualifying the Differences between Graphs

It is important to keep in mind that the identifications in **table 6** are merely proposed sequences and would require tandem mass spectrometry for confirmation. It is therefore difficult to address the biological implications of the networks at this stage. However, if we assume the proposed sequences to be correct, we can compare the differences between lasso, ridge and BDMCMC in terms of proteins. In other words, are the graphs comparable in terms of most influential proteins? For example, it might be the case that although lasso selects one among a number of correlated variables, that the differences with ridge are merely different fragments of the same protein. We have made an attempt at such a comparison below.

#	**lasso**	**ridge**	**BDMCMC**
1	L62	L158	L22
2	Complement C3 precursor	Fibrinogen alpha chain	L71
3	ITIH4	Complement C3 precursor	L151
4	Fibrinogen alpha chain	Fibrinogen alpha chain	L186
5	L92	L92	L135
6	Antithrombin-III	L69	L128
7	L99	L143	L18
8	ITIH4	L93	L28
9	IGFBP-3	L164	L4
10	Complement C3 precursor	Fibrinogen alpha chain	L189
11	L8	L129	L89
12	L93	L67	L172
13	Fibrinogen alpha chain	ITIH4	Pigment epithelium-derived factor
14	Vitronectin	L139	L185
15	L181	Complement C3 precursor	L120
16	APOA4	L99	L108
17	ITIH4	ITIH4	L150
18	L33	L62	APOA4
19	APOA4	Fibrinogen alpha chain	L25
20	L55	Ceruloplasmin	L168

Table 7: Highest ranking vertices for the weighted expected force of infection (WExF) and for each of the three estimation methods. Here we replaced the variable names of identified nodes with their proposed proteins, where possible.

From **table 7**, we again see that the regularized solutions are more alike. Complement C3 precursor and Fibrinogen alpha chain occur at least twice in either method's influential vertices. A more interesting observation is that although most proposed proteins are similarly influential in the lasso and ridge graph, some proteins are influential in the lasso graph, but not in the ridge graph (e.g. APOA4).

8 Discussion

In chapters **4** and **5**, we showed the results from the three methods for constructing conditional independence networks: graphical lasso, graphical ridge and BDMCMC. In all of these methods, the resulting networks were not sufficiently sparse for interpretation. In the regularized methods, we addressed the choice of LOOCV over other methods, that seek to sparsify a graph beyond what might oridnarily be possible given $\frac{n}{p}$ ratio. Since variable selection is not the primary goal — we wish to eventually understand the relationships between proteins — we opted for LOOCV. Interestingly, the Bayesian method produced similarly dense graphs, suggesting there is too much uncertainty given the sample size to create a sparse graph that is still reasonably correct. Hence, in chapter **6**, we attempted to address the issue of visualizing a relatively dense graph. Finally, in chapter **7** we tried to concretize the differences of the resulting graphs. Now we will discuss our findings.

8.1 Constructing a Graph

Graphical Lasso vs. Ridge

In chapter **1**, we reviewed the popular graphical lasso (`glasso`), developed by Hastie, Friedman and Tibshirani. It is extensively used for its sparsity inducing property to estimate graphs, or as a preceding step for PCA (Zou, Hastie, and Tibshirani 2006). However, the graphical lasso not only shares sparsity with its regression counterpart, but also inconsistency (Rolfs and Rajaratnam 2013). In light of this, attempts have been made to induce sparsity in a precision matrix regularized by the inherently more consistent ridge penalty (`rags2ridges`, Peeters, Bilgrau, and Wieringen 2015). In particular, local False Discovery Rate has been implemented to modify $\hat{\mathbf{\Theta}}$, such that it is sufficiently sparse to produce a graph. However, with either regularization technique, the problem of choosing the amount of regularization (λ) still remains, for which numerous procedures and criteria have been developed (Ha and Sun 2014; Rina Foygel 2010; Han Liu and Wasserman 2010). We covered the important methods and concluded that the familiar LOOCV is still superior when interest lies in the true graph structure and not sparseness. However, we did note that LOOCV tends to select very small values for λ, and subjected the LOOCV optimum to a minimum value corresponding to the matrix conditioning in (36).

Among the possible types of ridge penalties, we decided to use the alternative type I shrinkage, which corresponds to graphical lasso without penalizing the diagonal elements of the precision matrix. Both of these are amount to a prior on $\mathbf{\Theta}$, assuming the off-diagonal elements to be zero (i.e. a sparse matrix). Furthermore, this penalty was superior to both lasso and the other archetypal ridge penalties, in terms of matrix conditioning in **figure 5**. Using the LOOCV levels of regularization subject to matrix conditioning, the graphical lasso and ridge were performed, following a comprehensive analysis of centrality and node influence. Although there was overlap in the influential vertices of the lasso solution and the ridge solution, there were still nontrivial differences in the solutions (e.g. the proposed sequences for APOA4 in lasso but not in ridge, **table 7**). We therefore wish to express our preference for the graphical ridge, which — like its regression counterpart — has a unique solution, better consistency when $n < p$ and an appealing Bayesian interpretation (section **1.4.1**). However, conclusive evidence for better performance of either method would require a prediction setting, which we have not attempted due to time considerations.

BDMCMC

A completely different approach to graph construction by directly sampling from the Wishart posterior of $\mathbf{\Theta}$ was also evaluated: BDMCMC (18 - 22). Although we encourage a Bayesian perspective and the use of Bayesian approaches, it is unfortunate that some trial-and-error was required to obtain working values for the degrees of freedom on the prior ν, especially since the BDMCMC chains generally required at least 10^5 iterations to reach convergence for a mere 196×196 covariance matrix. This is a large drawback for high-dimensional datasets, such as in the field of genomics, where the number of genes is exceeds a thousand. Nevertheless, we compared the BDMCMC to the regularization techniques.

Interestingly, the estimated posterior graph size was comparable to that of the lFDR thresholded ridge. However, we found substantial differences between BDMCMC and the regularized solutions in terms of graph structure. Both the centrality and node influence of BDMCMC showed almost no similarity to the regularized solutions. Due to time constraints, we did not comprehensively compare the regularized solutions to a Gaussian copula graphical model (GCGM) contructed with BDMCMC, but a GCGM is included in the appendix, and was found to be quite similar to the GGM using BDMCMC (in terms of influential vertices). Furthermore, the authors of the `bdgraph` package noted that for practical use, a more advanced approach, incorporating Dirichlet variable selection would be preferred.

8.2 Inference in a Graph using WExF

To cope with the large resulting graphs, we implemented a recent development in node influence metrics: the WExF, to identify the most influential nodes in an otherwise incomprehensibly large graph. Our implementation of the WExF correctly assigns weights to negative partial correlations in an -omics context, where negative correlation are both frequent and important (contrary to a network of infection). The node influence has already proved to be useful in many areas and we found it to be a nice addition to the methods of inference in conditional independence networks. The most appealing difference with measures of centrality, is that the influence depends only on local graph structure and is thus less sensitive to small changes in a network.

8.3 Conclusion

8.3.1 Graphical Lasso vs Ridge

Although simulation studies have been performed on the graphical lasso and ridge, few authors go into detail about the differences in graph structure and node influence between these two methods. While the graphical lasso is very popular, it is far from consistent with results from a ridge solution. Let this thesis therefore serve as a cautionary note to those interested in true graph structure (and thus true relationships between variables), that a sparse solution selected by lasso need not be a good solution in terms of included edges.

Future research on a dataset where each variable's origin (e.g. protein peptide sequence) can be determined with relative certainty could further elucidate the differences in regularized solutions. Although a prediction setting could pick a 'winner' in terms of classification, it would also be interesting to find out the cost in terms of erroneous interpretation is when using the convenient graphical lasso.

8.3.2 BDMCMC vs Regularization

In our analysis, it would appear the (current) BDMCMC sampler does not compete with the regularization techniques for $n < p$ scenerios, since it displayed little to no similarity in terms of centrality or influence. Furthermore, many of the abundant (proposed) proteins, showed little to no influence in the graph.

8.3.3 Biological Interpretation

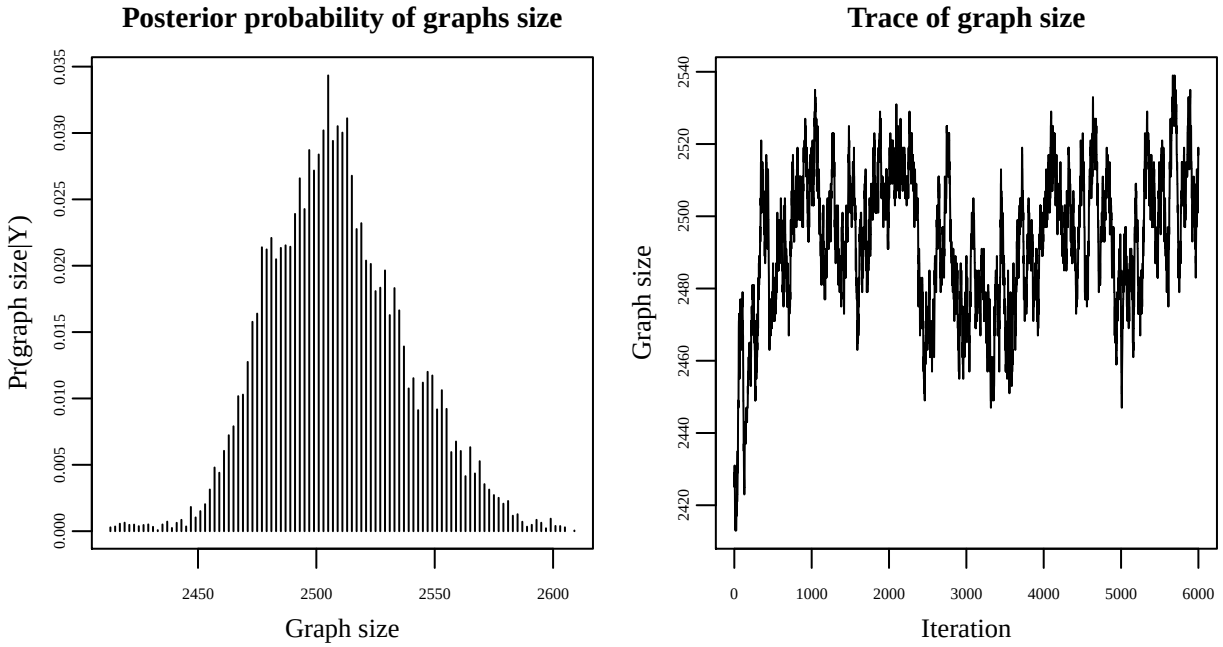
Table 6 showed us a selection of proposed protein peptides from the MALDI-FTICR data, suggesting the complement system of the innate immune response (C3 precursor, C4 precursor) could be very influential in a network of protein interactions, constructed from a covariance matrix of cases and controls. However, conclusive evidence about the biological implications of these particular networks are difficult, since only 65 out of 196 variables could be identified. Moreover, these identifications are merely proposed sequences. Confirming the sequences would require tandem mass spectrometry, such as that performed in Shen (2010). With that in mind, further research could elucidate the role of the complement system in pancreatic cancer

specifically, which has already been implied in a number of cancers. (e.g. Pio, Ajona, and Lambris 2013; Pio, Corrales, and Lambris 2014; Ostrand-Rosenberg 2008; Rutkowski et al. 2010)

9 Appendix

9.1 GGM vs GCGM

Here we show how relaxing the assumption of normality with a Gaussian Copula Graphical Model (GCGM) has minimal effect on the resulting graph, using the BDMCMC method. In the figure below we confirm convergence of the BDMCMC algorithm.



9.2 WExF: Consistency at Different Levels of Sparsity

Continuing from the GCGM example, we now show that the WExF is relatively insensitive to small changes in the graph. In particular, we show insensitivity to thresholding a dense graph, such that the smallest partial correlations are set to zero.

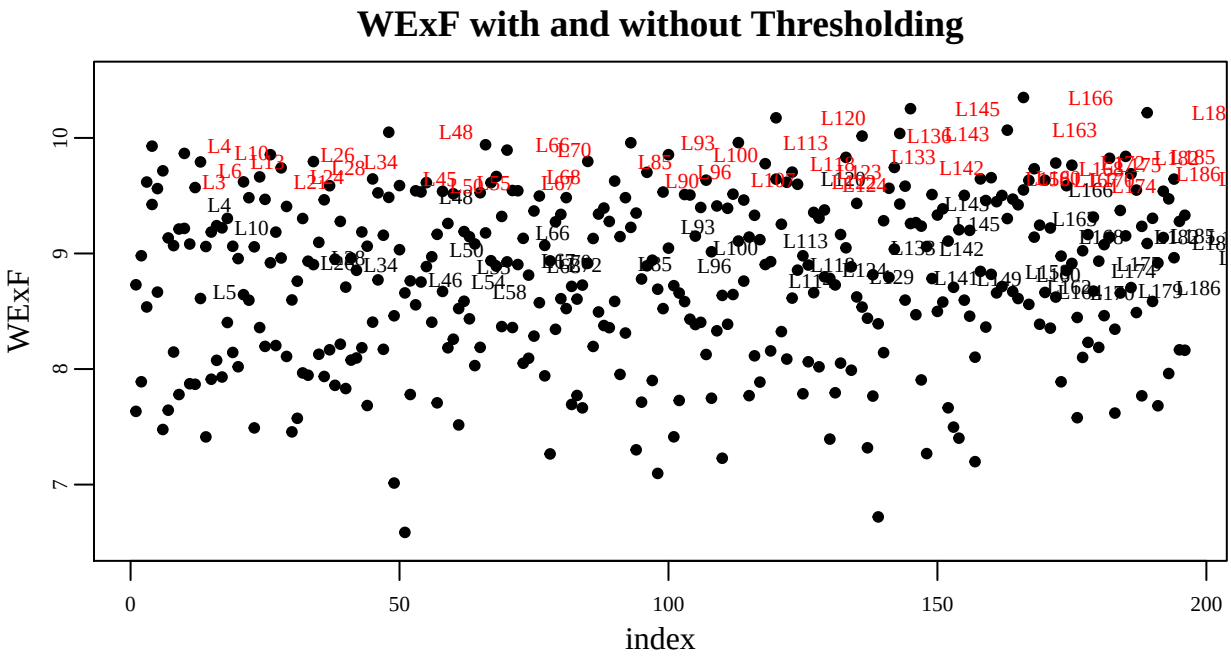


Figure 27: WExF of Bayesian GCGM. In black, the WExF of the thresholded $\hat{\Theta}$ is shown. In red, the WExF of vertices is shown without thresholding. Note how the WExF is fairly consistent, even after thresholding. Excluding the lowest partial correlations appears to reduce the WExF porportionally among nodes.

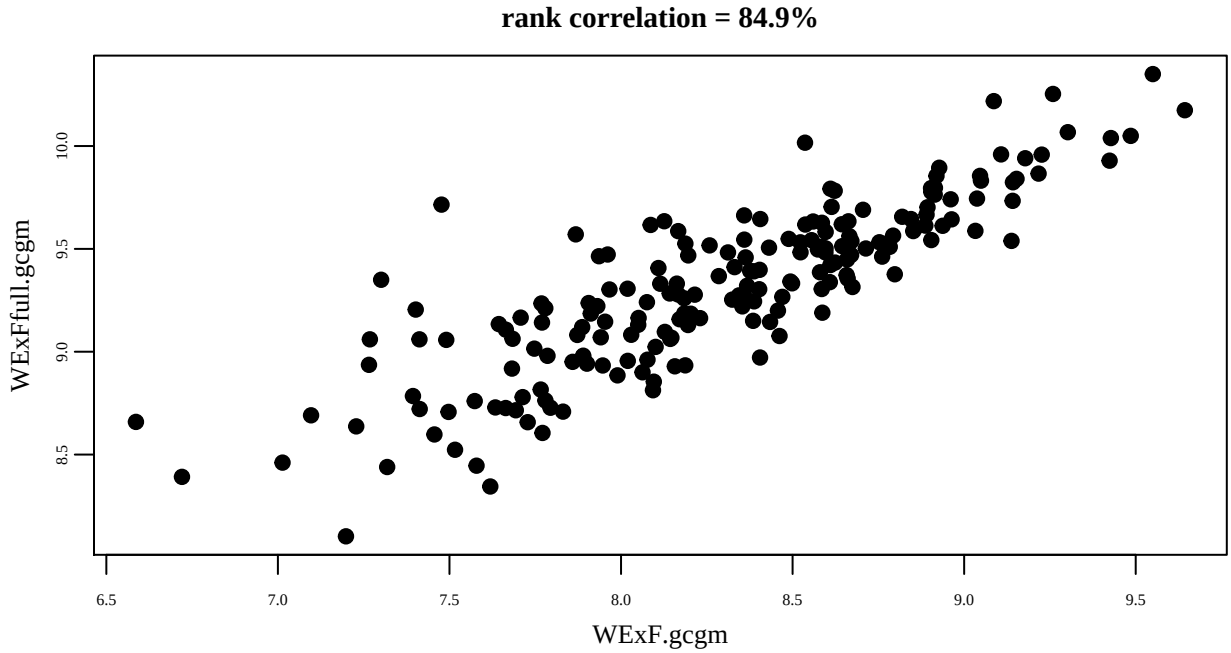


Figure 28: Here we confirm the WExF is consistent after thresholding by calculating the rank correlation between the thresholded and unthresholded $\hat{\Theta}$.

10 References

Bien, J., and R. J. Tibshirani. 2011. "Sparse estimation of a covariance matrix." *Biometrika* 98 (4): 807–20.

Cucuringu, Mihai, Jesus Puente, and David Shue. 2011. "Model Selection in Undirected Graphical Models with the Elastic Net."

Danaher, P., P. Wang, and D. M. Witten. 2014. "The joint graphical lasso for inverse covariance estimation across multiple classes." *J R Stat Soc Series B Stat Methodol* 76 (2): 373–97.

Efron, Bradley. 2007. "Size, Power and False Discovery Rates." *Ann. Statist.* 35 (4). The Institute of Mathematical Statistics: 1351–77. doi:10.1214/009053606000001460.

Ellinas, C., N. Allan, C. Durugbo, and A. Johansson. 2015. "How Robust Is Your Project? From Local Failures to Global Catastrophes: A Complex Networks Approach to Project Systemic Risk." *PLoS ONE* 10 (11): e0142469.

Friedman, J., T. Hastie, and R. Tibshirani. 2008. "Sparse inverse covariance estimation with the graphical lasso." *Biostatistics* 9 (3): 432–41.

Friedman, Jerome, Trevor Hastie, and Rob Tibshirani. 2014. *Glasso: Graphical Lasso- Estimation of Gaussian Graphical Models*. https://CRAN.R-project.org/package=glasso.

Friedman, Jerome, Trevor Hastie, and Robert Tibshirani. 2010. "Applications of the Lasso and Grouped Lasso to the Estimation of Sparse Graphical Models." In.

Ha, Min Jin, and Wei Sun. 2014. "Partial Correlation Matrix Estimation Using Ridge Penalty Followed by Thresholding and Re-Estimation." *Biometrics* 70 (3): 762–70. doi:10.1111/biom.12186.

Han Liu, Kathryn Roeder, and Larry Wasserman. 2010. "Stability Approach to Regularization Selection (Stars) for High Dimensional Graphical Models."

Hsu, Chih-Wen, Marick S. Sinay, and John S. J. Hsu. 2012. "Bayesian Estimation of a Covariance Matrix with Flexible Prior Specification." *Annals of the Institute of Statistical Mathematics* 64 (2): 319–42. doi:10.1007/s10463-010-0314-5.

Jordan, Lyndon Alexander, Sean M. Maguire, Hans A. Hofmann, and Masanori Kohda. 2016. "The Social and Ecological Costs of an over-Extended Phenotype." *Proceedings of the Royal Society of London B: Biological Sciences* 283 (1822). The Royal Society. doi:10.1098/rspb.2015.2359.

Krämer, Nicole, Juliane Schäfer, and Anne-Laure Boulesteix. 2009. "Regularized Estimation of Large-Scale Gene Association Networks Using Graphical Gaussian Models." *BMC Bioinformatics* 10 (1): 1–24. doi:10.1186/1471-2105-10-384.

Kuismin, M., and M. J. Sillanpaa. 2016. "Use of Wishart Prior and Simple Extensions for Sparse Precision Matrix Estimation." *PLoS ONE* 11 (2): e0148171.

Lauritzen, Steffen L. 1996. *Graphical Models*. Vol. 17. Clarendon Press.

Lawyer, G. 2015. "Understanding the influence of all nodes in a network." *Sci Rep* 5: 8665.

Lawyer, Glenn. 2016. "Measuring the Potential of Individual Airports for Pandemic Spread over the World Airline Network." *BMC Infectious Diseases* 16 (1): 1–10. doi:10.1186/s12879-016-1350-4.

Mazumder, Rahul, and Trevor Hastie. 2012a. "The Graphical Lasso: New Insights and Alternatives." *Electron. J. Statist.* 6. The Institute of Mathematical Statistics; the Bernoulli Society: 2125–49. doi:10.1214/12-EJS740.

———. 2012b. "Exact Covariance Thresholding into Connected Components for Large-Scale Graphical Lasso." *J. Mach. Learn. Res.* 13 (March). JMLR.org: 781–94. http://dl.acm.org/citation.cfm?id=2188385.2188412.

Milkau, Udo, and Jürgen Bott. 2015. "Digitalisation in Payments: From Interoperability to Centralised

Models?" *Journal of Payments Strategy & Systems* 9 (3): 321–40. http://www.ingentaconnect.com/content/hsp/jpss/2015/00000009/00000003/art00008.

Mohammadi, A., and E. C. Wit. 2015. "Bayesian Structure Learning in Sparse Gaussian Graphical Models." *Bayesian Anal.* 10 (1). International Society for Bayesian Analysis: 109–38. doi:10.1214/14-BA889.

Mohammadi, Abdolreza, and Ernst Wit. 2016. *BDgraph: Bayesian Graph Selection Based on Birth-Death Mcmc Approach*. https://CRAN.R-project.org/package=BDgraph.

Nicolardi, Simone, Berit Velstra, Bart J. Mertens, Bert Bonsing, Wilma E. Mesker, Rob A.E.M. Tollenaar, André M. Deelder, and Yuri E.M. van der Burgt. 2014. "Ultrahigh Resolution Profiles Lead to More Detailed Serum Peptidome Signatures of Pancreatic Cancer." *Translational Proteomics* 2: 39–51. doi:http://dx.doi.org/10.1016/j.trprot.2013.12.003.

Ostrand-Rosenberg, S. 2008. "Cancer and complement." *Nat. Biotechnol.* 26 (12): 1348–9.

Page, Lawrence, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1998. "The PageRank Citation Ranking: Bringing Order to the Web." Stanford Digital Library Technologies Project. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.31.1768.

Peeters, C.F.W., A.E. Bilgrau, and W.N. van Wieringen. 2015. *Rags2ridges: Ridge Estimation of Precision Matrices from High-Dimensional Data*.

Pereira, V. H., M. C. Gama, F. A. Sousa, T. G. Lewis, C. A. Gobatto, and F. B. Manchado-Gobatto. 2015. "Complex network models reveal correlations among network metrics, exercise intensity and role of body changes in the fatigue process." *Sci Rep* 5: 10489.

Pio, Ruben, Daniel Ajona, and John D. Lambris. 2013. "Complement Inhibition in Cancer Therapy." *Seminars in Immunology* 25 (1): 54–64. doi:http://dx.doi.org/10.1016/j.smim.2013.04.001.

Pio, Ruben, Leticia Corrales, and John D. Lambris. 2014. "The Role of Complement in Tumor Growth." In *Tumor Microenvironment and Cellular Stress: Signaling, Metabolism, Imaging, and Therapeutic Targets*, edited by Constantinos Koumenis, Ester Hammond, and Amato Giaccia, 229–62. New York, NY: Springer New York. doi:10.1007/978-1-4614-5915-6_11.

Preston, C.J. 1976. "Special Birth-and-Death Processes." *Bulletin of the International Statistical Institute* 46: 371–91.

Rina Foygel, Mathias Drton. 2010. "Extended Bayesian Information Criteria for Gaussian Graphical Models."

Rolfs, Benjamin T., and Bala Rajaratnam. 2013. "A Note on the Lack of Symmetry in the Graphical Lasso." *Computational Statistics & Data Analysis* 57 (1): 429–34. doi:http://dx.doi.org/10.1016/j.csda.2012.07.013.

Rutkowski, Martin J., Michael E. Sughrue, Ari J. Kane, Steven A. Mills, and Andrew T. Parsa. 2010. "Cancer and the Complement Cascade." *Molecular Cancer Research* 8 (11). American Association for Cancer Research: 1453–65. doi:10.1158/1541-7786.MCR-10-0225.

Shen, Nikola AND Liu, Yufeng AND Tolić. 2010. "Blood Peptidome-Degradome Profile of Breast Cancer." *PLOS ONE* 5 (10). Public Library of Science: 1–13. doi:10.1371/journal.pone.0013133.

Sun, H., and H. Li. 2012. "Robust Gaussian graphical modeling via l1 penalization." *Biometrics* 68 (4): 1197–1206.

Vogel, Curtis R. 2002. *Computational Methods for Inverse Problems*. Philadelphia, PA, USA: Society for Industrial; Applied Mathematics.

Vujačić, Ivan, Antonino Abbruzzo, and Ernst Wit. 2015. "A Computationally Fast Alternative to Cross-Validation in Penalized Gaussian Graphical Models." *Journal of Statistical Computation and Simulation* 85 (18): 3628–40. doi:10.1080/00949655.2014.992020.

Ward, Jr., J. H. 1963. "Hierarchical Grouping to Optimize an Objective Function." *Journal of the American*

Statistical Association 58: 236–44.

Won, J. H., J. Lim, S. J. Kim, and B. Rajaratnam. 2013. "Condition Number Regularized Covariance Estimation." *J R Stat Soc Series B Stat Methodol* 75 (3): 427–50.

Yan, K. K., G. Fang, N. Bhardwaj, R. P. Alexander, and M. Gerstein. 2010. "Comparing genomes to computer operating systems in terms of the topology and evolution of their regulatory control networks." *Proc. Natl. Acad. Sci. U.S.A.* 107 (20): 9186–91.

Yang, Sen, Zhaosong Lu, Xiaotong Shen, Peter Wonka, and Jieping Ye. 2015. "Fused Multiple Graphical Lasso." *SIAM J. Optim.* 25 (2). Society for Industrial & Applied Mathematics (SIAM): 916–43. doi:10.1137/130936397.

Zou, Hui, Trevor Hastie, and Robert Tibshirani. 2006. "Sparse Principal Component Analysis." *Journal of Computational and Graphical Statistics* 15 (2): 265–86. doi:10.1198/106186006X113430.