Project Results

Isaac Ray, Renat Sergazinov, Gözde Sert

Table of contents

1 Background and Introduction

Undirected graphical models enable to model multivariate distributions. Suppose we observe a p-dimensional sample $x = (x_1, ..., x_p)$ from a multivariate Gaussian distribution with a non-singular covariance matrix. Then the conditional independence structure of the distribution can be represented with a graph G. The graph $G = (V, E)$ is characterized by a node set $V = (1, ..., p)$ corresponding to the p variables, and an edge set E such that $(i, j) \in E$ if and only if x_i and x_j are conditionally dependent given all other variable. The goal is to estimate the underlying graph G from given n i.d.d. observations $x_1, ..., x_p$. Several methods developed under this assumption however, in practice, the observations may not be identically distributed. In this paper they suppose the variability in the graph structure across observations depending on additional covariate information.

1.1 Current Literature and Previous Methods

There are several approaches to model heterogeneous graphs. Here we mention some of them.

- Without using covariate information: These methods depend on the criteria of first splitting the data into homogeneous groups and sharing information withing groups
- Adding the covariates into the mean structure of Gaussian graphical models as multiple linear regressions such that the mean is a continuous function of covariates. This approaches is studied from a Bayesian perspective and a frequentist perspective. For this approach still uses the homogeneous graph structure for all observations which we do not want.
- Modeling the underlying covariance matrix as a function of the covariates. The main difficulty of this approach is to enforce sparsity in the precision matrix while being positive definite, as the sparsity in the covariance matrix does not normally carry to the precision matrix through matrix inversion.

1.2 Proposed formulation

Let $X \in \mathbb{R}^{n \times p}$ stand for the data matrix corresponding to n individuals on p variables. We denote the rows $X_i \in \mathbb{R}^p$ corresponding the observation for individual i and the columns $x_j \in \mathbb{R}^n$. The main goal of this paper is to learn the graph structure G from a collection of p-variate independent samples X_i, as a function of some extraneous covariates z_i corresponding to the samples. The only assumption on the dependence structure is that the graph parameters vary smoothly with respect to the covariates, that is, if z_i and z_j are similar, then the graph structure corresponding to X_i and X_j will be similar. In this method, a weighted pseudo-likelihood (W-PL) function to obtain a posterior distribution for the graph structure for a fixed individual, with the weights defined as a function of the covariates.

1.3 A weighted pseudo-likelihood (W-PL) approach

First we begin with introducing the pseudo-likelihood approach. Suppose there are n individuals, indexed $i = 1, .., n$. Let the i-th observation in the data set X be denoted as $X_i = (x_{i,1}, ..., x_{i,p})$, which corresponds to the i-th individual. Let $x_{i,-j} \in \mathbb{R}^{p-1}$ denote the vector of the i-th observation including all variables except $x_{i,j}$. This approach tries to model the conditional distribution of each of the x_j's given all other variables, denoted by $X_{-j} \in \mathbb{R}^{n\times(p-1)}$. Let the $p-1$-dimensional vector β_j indicate the regression effect on X_{-j} on x_j. Then the conditional likelihood of x_j denoted by $L(j)$ can be written as

$$L(j) = p(x_j|X_{-j}, \beta_j) \sim \prod_{i=1}^{n} \exp\left\{-(x_{i,j} - x_{i,-j}^T\beta_j)^2/2\sigma^2\right\}, \tag{1}$$

with a possibly sparse coefficient vector β_j. Then for a fixed graph G the pseudo-likelihood can be calculated as

$$L(G) = \prod_{j=1}^{n} L(j) = \prod_{j=1}^{n} p(x_j|X_{-j}, \beta_j). \tag{2}$$

In this paper different from the previous methods, they define a weighted version of this conditional likelihood for each individual. They assume that the underlying graph structure is a function of extraneous covariates z. Thus, we allow the coefficient vector β_j's to be different for different individuals,depending on the extraneous covariates. $\beta_j^l \in \mathbb{R}^{p-1}$ denotes the coefficient vector corresponding to the regression of the variable x_j on the remaining variables for individual l. Let z_i denote the covariate vector associated with the i-th individual and define $\mathbf{z} = (\mathbf{z}_1, ..., \mathbf{z}_n)$. Next, relative to the covariate z, we assign weights $w(z, \mathbf{z}_i) = \phi_\tau(\|z - \mathbf{z}_l\|)$ to every individual where ϕ_τ is the Gaussian density with mean 0 and variance τ^2. When $z = z_l$ corresponds to the l-th individual in the study, we use the notation $w_l(\mathbf{z}_i) = w(\mathbf{z}_l, \mathbf{z}_i)$ to denote the weight associated with the i-th individual.

Proposed conditional working model: for $i = 1, \dots, n$,

$$x_{ij}|x_{i,-j}, \mathbf{z}, z \sim N(x_{i,-j}^T\beta_j(z), \sigma^2/w(z, \mathbf{z}_i))$$

Next let $W(z, \mathbf{z})$ denote the diagonal matrix $\mathrm{Diag}(w(z, \mathbf{z}_1), \dots, w(z, \mathbf{z}_n))$. The weighted conditional distribution function can be calculated as

$$p^w(x_j|X_{-j}, \beta_j(\mathbf{z}), \mathbf{z}, z) = \left(\prod_{i=1}^{n} \sqrt{\frac{w(z, \mathbf{z}_i)}{2\pi\sigma_*^2}}\right) \exp\left\{-\frac{(x_j - X_{-j}\beta_j(z))^T W(z, \mathbf{z})(x_j - X_{-j}\beta_j(z))}{2\sigma_*^2}\right\} \tag{3}$$

Then using the previous pseudo-likelihood for the graph G, we now give the weighted pseudo-likelihood for the graph $G(z)$ corresponding to a covariate value z,

$$L^w(G(z)) = \prod_{j=1}^{n} p^w(x_j|\beta_j(\mathbf{z}), X_{-j}, \mathbf{z}, z)$$

Next, we put a prior distribution for the coefficient parameters corresponding to the regression problem described before. Fix an observation $l \in \{1, \dots, n\}$ and a variable $j \in \{1, \dots, p\}$. Then a spike-and-slab prior on the parameter β_j^l. So for $k \in \{1, \dots, p\}$, $\beta_{j,k}^l$ is assumed to come from a zero-mean Gaussian density with variance component $\sigma^2\sigma_\beta^2$ with probability π and equals to zero with probability $1 - \pi$. Let $\gamma_{j,k}^l$ be the indicator of nonzero $\beta_{j,k}^l$ and we denote it as $\gamma_{j,k}^l = 1\{\beta_{j,k}^l \neq 0\}$ which can be treated as Bernoulli random variables with a common probability of success π. Then we define $\gamma_j^l = (\gamma_{j,1}^l, \dots, \gamma_{j,p}^l)$ and $\Gamma^l = \{\gamma_{j,k}^l, j = 1, \dots, p\}$. Then prior distribution for $(\beta_{j,k}^l, \gamma_{j,k}^l)$ can be written as

$$p_0(\beta_{j,k}^l, \gamma_{j,k}^l) = \prod_{k=1, k\neq j}^{n} \delta_0(\beta_{j,k}^l)^{1-\gamma_{j,k}^l} N(0, \beta_{j,k}^l; 0, \sigma^2\sigma_\beta^2) \prod_{k=1, k\neq j}^{n} \pi^{\gamma_{j,k}^l}(1-\pi)^{\gamma_{j,k}^l}.$$

Then the posterior distribution for $(\beta_{j,k}^l, \gamma_{j,k}^l)$ can be calculated as

$$p(\beta_{j,k}^l, \gamma_{j,k}^l | X) \propto \exp\left\{ -\frac{1}{2\sigma^2} \sum_{i=1}^{n} \left(x_{ij} - \sum_{k=1, k\neq j}^{p} x_{ik}\beta_{j,k}^l \right)^2 w_l(\mathbf{z}_i) \right\} p_0(\beta_{j,k}^l, \gamma_{j,k}^l).$$

1.4 Block Mean Field Variational Inference

Variational inference is one of the popular ways to approximate the posterior distribution. In this section, first, we give a brief introduction to it. Then we will connect it with our problem.

Suppose we have a parameter of interest θ with intractable posterior distribution $p(\theta)$, an observed data vector y, and the variational tractable family of densities $q(\theta)$. Then we want to find the best approximating density q^* in a tractable family of densities Γ with respect to KL-divergence:

$$q^*(\theta) = \arg\min_{q \in \Gamma} \mathrm{KL}(q \| p(\theta | y)).$$

Instead of solving the above problem we work on the evidence-lower bound(ELBO):

$$ELBO = \int q(\theta) \log(p(y, \theta)/q(\theta)) \, d\theta$$

and maximize it. For our problem the parameter of interest $\theta = (\beta_j^l, \gamma_j^l)$. We apply the block mean-field approach for the variational approximation. Let $\phi_{j,k}^l = (\alpha_{j,k}^l, \mu_{j,k}^l, (s_{j,k}^l)^2)$ be free parameters corresponding to the individuals. Then we have

$$q_k(\beta_j^l, \gamma_j^l; \phi_j^l) = N(\beta_{j,k}^l; \mu_{j,k}^l, (s_{j,k}^l)^2)^{\gamma_{j,k}^l} \delta_0(\beta_{j,k}^l)^{1-\gamma_{j,k}^l} (\alpha_{j,k}^l)^{\gamma_{j,k}^l} (1 - \alpha_{j,k}^l)^{1-\gamma_{j,k}^l}.$$

Then we have the following variational parameter updates using the batch-wise updating algorithm:

$$(s_{j,k}^l)^2 = \frac{\sigma^2}{1/\sigma_\beta^2 + \sum_{i=1}^n x_{ik}^2 w_l(\mathbf{z}_i)}$$

$$\mu_{j,k}^l = \frac{(s_{j,k}^l)^2}{\sigma^2} \sum_{i=1}^n \{w_l(\mathbf{z}_i) x_{ik}\}$$

$$\mu_{j,k}^l = \frac{(s_{j,k}^l)^2}{\sigma^2} \sum_{i=1}^n \left\{ w_l(\mathbf{z}_i) x_{ik} \left(x_{ij} - \sum_{m \neq j,k} x_{im} \mu_{j,m}^l \alpha_{j,m}^l \right) \right\}$$

$$\text{logit}(\alpha_{j,k}^l) = \text{logit}(\pi) + \frac{(\mu_{j,k}^l)^2}{2(s_{j,k}^l)^2} + \log\left(\frac{s_{j,k}^l}{\sigma\sigma_\beta}\right).$$

2 Problem

Despite this model demonstrating superior sensitivity to true dependence relations than competing methods, it suffers from lower specificity. Compared with competing methods from the `mgm` (Haslbeck and Waldorp 2020) and `JGL` (Danaher 2018)packages, the specificity gets substantially worse as the number of features increases. In the case of analyzing gene expression data, this could lead to worse outcomes than a less sensitive and more specific model since the cost of carrying out experiments which show a lack of a predicted relation may be very expensive. Further, validating a true but relatively weak relationship may not be desirable considering the cost. Ideally we want to **increase the specificity of the model without substantially hurting the model's sensitivity and speed**.

3 Attempted Solutions

We'll look at a variety of different modifications to the algorithm in order of least to most extensive changes necessary, and use the `covdepGE` package (Helwig et al. 2022) in order to generate data for simulation studies. For some of these solutions, the package's functions will be used without modification and changes will occur outside of the inference algorithm. Otherwise, any changes to the functions will be explicitly noted.

In general, we expect that the underlying cause of the specificity issue is due to having a common prior inclusion probability π across every spike-and-slab regression being performed despite the varying values of Z (and potentially X). We may expect that for certain values of Z, we have a different belief about whether variables in X are related. We'll approach this from 2 angles; first by trying to modify our variables and covariates in such a way as to make a common π a more appropriate choice, and then by modifying the algorithm to allow for

multiple π values to be specified either a priori or as a function $\pi(X, Z)$ through something like clustering.

3.1 Feature Scaling Changes

The first approach will be to use a different or additional approach to feature scaling on X and/or Z in order to try and make a single prior inclusion probability more appropriate.

3.1.1 Existing Feature Scaling through Normalization

Currently, the default behavior in the `covdepGE` function is to perform a column-wise Z-score Normalization on Z and a column-wise 0 centering on X. For brevity, we'll denote this procedure by "normalization". The baseline performance under this scheme is given below. All experiments were run under 4 different setups each having different values for p and n, and data was simulated using the `generateData` function. To assess sensitivity and specificity, we'll examine the number of false positives per sample and the number of false negatives per sample across 100 replications of each simulation setup. So, in all cases lower numbers are desirable. First, we'll look at the baseline performance of the existing function with no changes to the default behavior.

p	n	Baseline FPs (sd)	Baseline FNs (sd)
5	90	0.34 (0.54)	0.99 (0.53)
15	90	0.82 (0.94)	1.41 (0.69)
25	150	1.29 (0.98)	0.91 (0.38)
50	150	4.36 (2.22)	1.19 (0.48)

Table 1: False Positives and False Negatives per sample - Normalized Z, Centered X

3.1.2 Max-Min Feature Scaling

We try max-min scaling that puts all values in the range $[0, 1]$ by subtracting the minimum value and dividing by the range. We experiment with doing separately and together for X and Z. As we can from Tables 2-3, the max-min scaling on Z alone results in worse performance. Further, whenever max-min scaling is applied to X, the algorithm fails to converge, resulting in no predicted edges. This is likely due to the fact that max-min scaling alters the distribution shape of X. Therefore, we do not pursue this approach further. We explore the robustness of the algorithm to the misspecification (non-normality of the data X) in Section 3.3.

p	n	Baseline FPs (sd)	MM on Z	MM on X	MM on Z, X
5	90	0.34 (0.54)	0.81 (0.63)	0 (0)	0 (0)
15	90	0.82 (0.94)	4.08 (1.44)	0 (0)	0 (0)
25	150	1.29 (0.98)	11.89 (2.5)	0 (0)	0 (0)
50	150	4.36 (2.22)	19.86 (3.59)	0 (0)	0 (0)

Table 2: False **positives** per sample - Max Min Scaling only; (sd)

p	n	Baseline FNs (sd)	MM on Z	MM on X	MM on Z, X
5	90	0.99 (0.53)	1.14 (0.49)	4.67 (0)	4.67 (0)
15	90	1.41 (0.69)	2.35 (0.78)	4.67 (0)	4.67 (0)
25	150	0.91 (0.38)	1.71 (0.7)	4.67 (0)	4.67 (0)
50	150	1.19 (0.48)	2.42 (0.71)	4.67 (0)	4.67 (0)

Table 3: False **negatives** per sample - Max Min Scaling only; (sd)

3.1.3 Max-Min + Normalization Feature Scaling

To reduce the distribution-shape mismatch, we also apply normalization after min-max scaling. We try this separately for X and Z. The results are shown in Tables 4-5. From the results, we can see that the false positive rate drops by 15% on average. However, the false negative rate increases by 5% on average. This is likely due to the fact that normalization reduces the variance of the data, which in turn reduces the power of the algorithm to detect the true edges. Overall, we consider this method to be viable for specific applications but not optimal.

p	n	Baseline FPs (sd)	MM/N Z	MM/N X	MM/N Z, X
5	90	0.34 (0.54)	0.24 (0.47)	0.26 (0.47)	0.24 (0.44)
15	90	0.82 (0.94)	0.63 (0.83)	0.61 (0.77)	0.61 (0.77)
25	150	1.29 (0.98)	1.34 (0.99)	1.34 (0.97)	1.35 (0.98)
50	150	4.36 (2.22)	4.08 (1.89)	4.07 (1.84)	4.07 (1.82)

Table 4: False **positives** per sample - Max Min Scaling + Normalization; (sd)

p	n	Baseline FNs (sd)	MM/N Z	MM/N X	MM/N Z, X
5	90	0.99 (0.53)	1.03 (0.45)	1 (0.47)	1.02 (0.46)
15	90	1.41 (0.69)	1.53 (0.65)	1.48 (0.63)	1.47 (0.65)
25	150	0.91 (0.38)	0.95 (0.46)	0.92 (0.44)	0.92 (0.44)
50	150	1.19 (0.48)	1.15 (0.44)	1.12 (0.43)	1.11 (0.43)

Table 5: False **negatives** per sample - Max Min Scaling + Normalization; (sd)

3.2 Multiple Prior Inclusion Probabilities

We investigate the algorithm under multiple inclusion probability specifications. In particular, we study the case when separate inclusion probability, π^l, is specified for each observation $l = \{1, 2, ..., n\}$. We break this problem down further into the clustering and hyperparameter estimation steps. In the clustering step, we assign each observation to a cluster based on the extraneous covariates. In the hyperparameter estimation step, we select individual inclusion probabilities for each cluster. We note that the computational complexity of the algorithm scales linearly with the number of clusters. This is because the hyperparameter optimization has to be run independently for each cluster, which introduces an additional loop.

3.2.1 Oracle or Informative Prior

In the simulation settings, the exact mapping of observation into clusters is typically known. We first try this approach and report the results in Table 6. From Table 6, we see that we get a uniformly worse false positives rate. At the same time, the false negative rate seems to be similar to the baseline. We hypothesize that this happens due to the algorithm implicitly favoring reducing false negatives more than false positives. Thus, when endowed with more flexibility (more parameters), the algorithm starts to overfit the data. This is corroborated by the fact that the false negatives stay the same (or get slightly better for $n = 150, p = 50$ case), while the false positives get worse. In this sense, a single inclusion probability is more robust to overfitting.

p	n	Baseline FPs (sd)	Mean FP (sd)	Baseline FNs (sd)	Mean FN (sd)
5	90	0.34 (0.54)	0.45 (0.62)	0.99 (0.53)	1.09 (0.58)
15	90	0.82 (0.94)	1.25 (0.97)	1.41 (0.69)	1.53 (0.66)
25	150	1.29 (0.98)	3.24 (1.39)	0.91 (0.38)	0.99 (0.42)
50	150	4.36 (2.22)	9.16 (2.42)	1.19 (0.48)	1.01 (0.36)

Table 6: False positives **and** negatives per sample - Multiple PIP with Oracle Clustering

3.2.2 Prior Inclusion through Covariate Clustering

We investigate data-driven observation clustering through extraneous covariates. We start with parametric algorithms with a fixed number of clusters to test our hypothesize that there is an inherent trade-off between false negative and false positive rates. In particular, more flexible algorithms (more clusters) tend to overfit, resulting in a better false negative rate at the expense of the worse false positive rate. We test this hypothesis by running the algorithm with $k = 2, 3, 6$ clusters for each case. We report our results in Table 7, which confirms our beliefs. For $p \leq 25$, the number of clusters does not seem to affect the false negative performance; however, the false positive rate gets worse. For $p > 25$, we most clearly see

that as the number of clusters goes up, the false negative rate gets better; however, the false positive rate gets worse.

p	n	clusts	Baseline FPs (sd)	Mean FP (sd)	Baseline FNs (sd)	Mean FN (sd)
5	90	2	0.34 (0.54)	0.47 (0.66)	0.99 (0.53)	1.08 (0.61)
5	90	3	0.34 (0.54)	0.47 (0.68)	0.99 (0.53)	1.09 (0.6)
5	90	6	0.34 (0.54)	0.48 (0.64)	0.99 (0.53)	1.09 (0.61)
15	90	2	0.82 (0.94)	1.07 (0.98)	1.41 (0.69)	1.53 (0.69)
15	90	3	0.82 (0.94)	1.21 (0.99)	1.41 (0.69)	1.51 (0.67)
15	90	6	0.82 (0.94)	1.41 (1)	1.41 (0.69)	1.52 (0.67)
25	150	2	1.29 (0.98)	2.71 (1.37)	0.91 (0.38)	1 (0.42)
25	150	3	1.29 (0.98)	3.2 (1.32)	0.91 (0.38)	0.99 (0.43)
25	150	6	1.29 (0.98)	3.54 (1.36)	0.91 (0.38)	0.99 (0.42)
50	150	2	4.36 (2.22)	7.76 (2.3)	1.19 (0.48)	1.06 (0.37)
50	150	3	4.36 (2.22)	8.78 (2.43)	1.19 (0.48)	1.04 (0.37)
50	150	6	4.36 (2.22)	10.14 (2.46)	1.19 (0.48)	1 (0.37)

Table 7: False positives **and** negatives per sample - Multiple PIP with Hierarchical Clustering

3.2.3 Prior Inclusion at the Individual Level

We experimented with giving every observation its own prior inclusion probability by specifying the cluster mapping as $\{1, 2, ..., n\}$; however the simulations became computationally unfeasible due to the dimension of the parameter space being grid searched over scaling linearly with n. After multiple days of running the simulation crashed due to excessive memory consumption (>32GB). Based on the previous results with the oracle clustering and hierarchical clustering, and a much smaller example with $p = 5, n = 10$, we believe it highly unlikely that having a different prior inclusion probability for every observation would improve the false positive rate.

3.3 Robustness Experiments

In order to verify our results, there are a few more experiments we want to try running. In particular, we want to see whether we can break the Gaussian assumption of our true data-generating function. The hope is that the scaling on X will improve the model's robustness to a distribution with fatter tails such as a t distribution with low degrees of freedom. Similarly, we want to try adding a small percentage of 'contaminated' observations that are drawn from an unrelated, independent Gaussian distribution to the one we are trying to work with. We again hope that the additional scaling we do can help combat the effects of the bad data. Finally, we want to see if we can use the additional scaling of Z to account for potential non-smoothness.

3.3.1 Contamination - Gaussian

First, we'll consider the case that a proportion of our observations' true data-generating function is just noise; that is, $X_{\text{contaminated}} \sim N(0, I)$. Notably, it doesn't depend on Z at all.

p	n	Perc. Corrupt	Baseline Mean (sd)	MM Mean (sd)
5	90	5	0.42 (0.49)	0.41 (0.47)
5	90	10	1.14 (1.14)	1.13 (1.13)
5	90	25	2.04 (1.27)	2.01 (1.28)
15	90	5	0.52 (0.36)	0.52 (0.35)
15	90	10	1.47 (1.23)	1.46 (1.18)
15	90	25	2.61 (1.71)	2.64 (1.71)
25	150	5	0.8 (0.57)	0.8 (0.56)
25	150	10	1.8 (1.2)	1.8 (1.23)
25	150	25	3.38 (1.53)	3.41 (1.54)

Table 8: False **positives** per sample - Gaussian contamination in data

p	n	Perc. Corrupt	Baseline Mean (sd)	MM Mean (sd)
5	90	5	1.27 (0.64)	1.25 (0.63)
5	90	10	1.59 (0.76)	1.57 (0.75)
5	90	25	1.13 (0.4)	1.1 (0.39)
15	90	5	1.41 (0.63)	1.39 (0.62)
15	90	10	1.88 (0.71)	1.85 (0.71)
15	90	25	1.19 (0.42)	1.18 (0.42)
25	150	5	1.85 (0.76)	1.84 (0.77)
25	150	10	2.14 (0.73)	2.12 (0.74)
25	150	25	1.57 (0.61)	1.55 (0.6)

Table 9: False **negatives** per sample - Gaussian contamination in data

3.3.2 Contamination - Non-Gaussian

Next, we'll consider the case that a proportion of our observations' true data-generating function is coming from a multivariate t distribution with ν degrees of freedom (which we'll restrict to $\nu > 2$. When generating this data, we'll use the covariance matrix Σ obtained from Z to instead specify the *scale* matrix of our multivariate t distribution, with the relationship that the true covariance matrix of these t distributed variables is $\Sigma_t = \Sigma * \nu/(\nu - 2)$

p	n	t-df	Baseline Mean (sd)	MM Mean (sd)
5	90	3	0.57 (0.63)	0.54 (0.62)
5	90	6	1.28 (1.23)	1.3 (1.22)
5	90	15	2.42 (1.57)	2.36 (1.59)
15	90	3	0.4 (0.71)	0.38 (0.64)
15	90	6	1.01 (0.95)	1 (0.95)
15	90	15	2.45 (1.53)	2.5 (1.54)
25	150	3	0.23 (0.44)	0.21 (0.42)
25	150	6	0.78 (0.88)	0.79 (0.85)
25	150	15	1.67 (1.32)	1.69 (1.34)

Table 10: False **positives** per sample - Multivariate-t contamination in data

p	n	t-df	Baseline Mean (sd)	MM Mean (sd)
5	90	3	1.51 (1.19)	1.5 (1.18)
5	90	6	2.07 (1.08)	2.04 (1.06)
5	90	15	1.87 (1.12)	1.86 (1.14)
15	90	3	1.09 (0.57)	1.09 (0.56)
15	90	6	1.56 (0.68)	1.53 (0.69)
15	90	15	1.02 (0.42)	0.99 (0.41)
25	150	3	1.1 (0.58)	1.08 (0.58)
25	150	6	1.41 (0.58)	1.39 (0.57)
25	150	15	0.93 (0.43)	0.91 (0.44)

Table 11: False **negatives** per sample - Multivariate-t contamination in data

3.3.3 Non-Smooth Function of Z

Finally, we want to see what happens if the true graph structure isn't a smoothly varying function of Z. In particular, we'll consider the true interpolation of the precision between the first and third interval is given by $\max(\min(\tan(7.5 * z), 1), 0)$ instead of by $\beta_0 + \beta_1 z$.

p	n	Baseline Mean (sd)	MM Mean (sd)
5	90	0.61 (0.74)	0.6 (0.73)
15	90	0.84 (0.73)	0.79 (0.67)
25	150	1.73 (1.29)	1.72 (1.26)

Table 12: False **positives** per sample - True precision not smoothly varying with Z

p	n	Baseline Mean (sd)	MM Mean (sd)
5	90	0.75 (0.58)	0.74 (0.58)
15	90	1.09 (0.65)	1.06 (0.65)
25	150	0.6 (0.38)	0.57 (0.37)

Table 13: False **negatives** per sample - True precision not smoothly varying with Z

4 Discussion

In this project, we have explored different approaches for reducing the false positive rate in the recently proposed `covdepGE` algorithm for covariate dependence estimation. The specific modifications we have investigated are: 1) different re-scaling techniques, 2) individual inclusion probabilities for observation clusters. Based on the obtained results, we conclude that the re-scaling techniques are the best among the methods we have tested. They provide similar performance in terms of the false negatives and a slightly better false positive rate than the baseline. Somewhat surprisingly, the individual inclusion probabilities do not seem to improve the performance of the algorithm. We hypothesize that this is due to the fact that the algorithm implicitly favors improvements in the false negative rate over the false positive rate. Hence, it starts to overfit when given more flexibility, showing much better false negative performance but failing to reduce the false positive rate.

References

Danaher, Patrick. 2018. "JGL: Performs the Joint Graphical Lasso for Sparse Inverse Covariance Estimation on Multiple Classes." https://CRAN.R-project.org/package=JGL.

Haslbeck, Jonas M. B., and Lourens J. Waldorp. 2020. "{Mgm}: Estimating Time-Varying Mixed Graphical Models in High-Dimensional Data" 93. https://doi.org/10.18637/jss.v093.i08.

Helwig, Jacob, Sutanoy Dasgupta, Peng Zhao, Bani Mallick, and Debdeep Pati. 2022. "covdepGE: Covariate Dependent Graph Estimation." https://CRAN.R-project.org/package=covdepGE.