Algorithm xxx: A Covariate-Dependent Approach to Gaussian Graphical Modeling in R

JACOB HELWIG, Department of Computer Science and Engineering, Texas A&M University, USA
SUTANOY DASGUPTA, Department of Statistics, Texas A&M University, USA
PENG ZHAO, Department of Applied Economics and Statistics, University of Delaware, USA
BANI K. MALLICK and DEBDEEP PATI, Department of Statistics, Texas A&M University, USA

Graphical models are used to capture complex multivariate relationships and have applications in diverse disciplines such as in biology, physics, and economics. Within this field, Gaussian graphical models aim to identify the pairs of variables whose dependence is maintained even after conditioning on the remaining variables in the data, known as the *conditional dependence structure* of the data. There are many existing software packages for Gaussian graphical modeling, however, they often make restrictive assumptions that reduce their flexibility for modeling data that are not identically distributed. Conversely, `covdepGE` is a R implementation of a variational weighted pseudo-likelihood algorithm for modeling the conditional dependence structure as a continuous function of an extraneous covariate. To build on the efficiency of this algorithm, `covdepGE` leverages parallelism and C++ integration with R. Additionally, `covdepGE` provides fully-automated and data-driven hyperparameter specification while maintaining flexibility for the user to decide key components of the estimation procedure. Through an extensive simulation study spanning diverse settings, `covdepGE` is demonstrated to be top of its class in recovering the ground-truth conditional dependence structure while efficiently managing computational overhead.

CCS Concepts: • **Mathematics of computing → Bayesian computation**; **Variational methods**; **Multivariate statistics**; **Statistical software**; **Probabilistic algorithms**; *Regression analysis*; *Kernel density estimators*; *Continuous functions*; *Approximation algorithms*.

Additional Key Words and Phrases: Gaussian graphical models, variational inference, structure learning, pseudo-likelihood, heterogeneous graphs

ACM Reference Format:
Jacob Helwig, Sutanoy Dasgupta, Peng Zhao, Bani K. Mallick, and Debdeep Pati. 2022. Algorithm xxx: A Covariate-Dependent Approach to Gaussian Graphical Modeling in R. 1, 1 (April 2022), 30 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

We present the R [Team 2021] package `covdepGE`, an implementation of the Gaussian graphical modeling (GGM) algorithm introduced by Dasgupta et al. [2023]. GGM seeks to capture the conditional dependence structure (CDS) of the data under the assumption that the data are normally distributed. This distributional assumption is convenient for

Authors' addresses: Jacob Helwig, jacob.a.helwig@tamu.edu, Department of Computer Science and Engineering, Texas A&M University, 435 Nagle Street, College Station, Texas, USA, 77843; Sutanoy Dasgupta, sutanoy@stat.tamu.edu, Department of Statistics, Texas A&M University, 155 Ireland Street, College Station, Texas, USA, 77843; Peng Zhao, Department of Applied Economics and Statistics, University of Delaware, 531 South College Avenue, Newark, Delaware, USA, 19716, pzhao@udel.edu; Bani K. Mallick, bmallick@stat.tamu.edu; Debdeep Pati, debdeep@stat.tamu.edu, Department of Statistics, Texas A&M University, 155 Ireland Street, College Station, Texas, USA, 77843.

inference, as the CDS is given by the sparsity structure of the precision matrix [Lauritzen 1996]. There are more than 35 existing R packages for GGM, for example Marchetti [2006]; Vinciotti et al. [2016]; Williams and Mulder [2020] to name just a few. However, as with the existing theory, such as Friedman et al. [2008]; Liu et al. [2010]; Majumdar and Michailidis [2022], many make the restrictive assumption that the precision matrix is homogeneous throughout the data. Other methods assume that the data may be partitioned into homogeneous subgroups such that all observations in a group have the same precision matrix [Danaher et al. 2014; Das et al. 2020; Gao et al. 2016; Hao et al. 2018; Peterson et al. 2015; Ren et al. 2022; Shaddox et al. 2018] or, in the case of time series data, that the entries of the precision matrix vary continuously with time [Haslbeck and Waldorp 2020; Liu et al. 2022; Qiu et al. 2016; Yang and Peng 2020]. Dasgupta et al. [2023] instead model the precision matrix as varying continuously with a potentially multivariate extraneous covariate. Intuitively, this implies that observations having similar extraneous covariate values will have similar precision matrices. While a similar framework has been proposed by Ni et al. [2019] for covariate-dependent modeling of DAGs, and recent methods for covariate-dependent GGMs have been developed concurrently with Dasgupta et al. [2023] by Ni et al. [2022] and Zhang and Li [2023], an efficient software implementation for such settings is lacking.

To facilitate information sharing through the extraneous covariate while managing computational overhead, Dasgupta et al. [2023] propose an efficient variational approximation conducted under the novel weighted pseudo-likelihood framework that we provide in the main function `covdepGE::covdepGE` visualized in Figure 1. We improve efficiency by implementing the parallelism proposed by Dasgupta et al. [2023], and further accelerate inference by executing expensive iterative computations in C++. We build on Dasgupta et al. [2023] by developing a principled, data-driven approach for hyperparameter specification that only requires the user to input data and extraneous covariates to perform inference. At the same time, we maintain users' freedom to directly specify all key aspects of modeling through an intuitive API. Finally, we offer several wrappers around `ggplot2` [Wickham 2016] for seamless visualization of resulting estimates, such as `covdepGE::matViz` and the S3 method `plot.covdepGE`.

Our contributions are as follows:

(1) A package for general covariate-dependent Gaussian graphical modeling. Currently available implementations for heterogeneous graphical modeling are limited in that they either treat the data as homogeneous within subgroups, corresponding to a discrete covariate (e.g., `JGL` [Danaher et al. 2014]), or as varying continuously with time, corresponding to a single-dimensional covariate (e.g., `mgm` [Haslbeck and Waldorp 2020]).
(2) An implementation that achieves efficiency through a variational approximation, information sharing via similarity weights, parallelism, and C++ integration with R.
(3) Three alternatives for automated hyperparameter specification and a study comparing these approaches.
(4) An experimental comparison of our package to three packages representing the most prevalent approaches for heterogeneous graphical modeling: time-varying [Haslbeck and Waldorp 2020; Yang and Peng 2020], and homogeneous subgroups [Danaher et al. 2014]. We consider diverse settings, varying the dimensionality of our data and the extraneous covariate, as well as the class of function relating the precision matrix to the covariate.

In Section 2, we overview the method of Dasgupta et al. [2023], describe how we implement it in `covdepGE`, and detail the hyperparameter specification strategies that we have included in the package. In Section 3, we perform a thorough simulation study, studying differing hyperparameter specification strategies, and comparing the performance of `covdepGE` to the R packages `JGL` [Danaher et al. 2014], `mgm` [Haslbeck and Waldorp 2020], and `loggle` [Yang and Peng 2020] in 16 diverse settings.

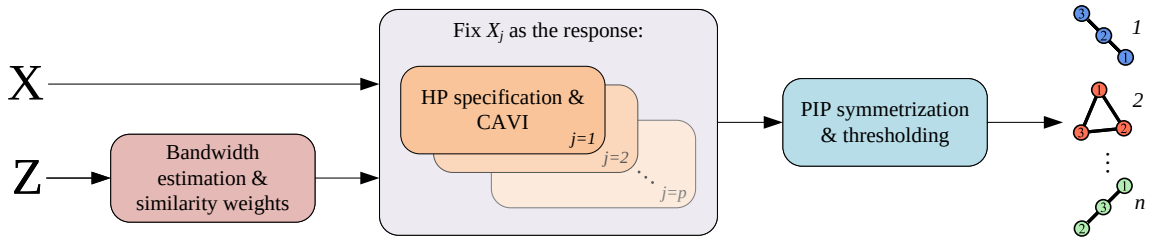

Fig. 1. Computational graph for the covdepGE::covdepGE function. We first estimate bandwidths using the extraneous covariate and then use these to calculate pair-wise similarity weights between all observations. Next, we fix each variable in the data as the response and perform variational spike-and-slab regression for each observation using coordinate ascent variational inference (CAVI), where the regressions are weighted with the observation-specific similarity weights. We select hyperparameters for these regressions using the methods detailed in Section 2.3. In the final step, we post-process the estimated posterior inclusion probabilities from these regressions to estimate the graph describing the conditional dependence structure for each observation.

2 COVARIATE-DEPENDENT GRAPH ESTIMATION

Suppose that $\mathbf{X} \in \mathbb{R}^{n \times p}$ is a data matrix and that $\mathbf{Z} \in \mathbb{R}^{n \times q}$ is a q-dimensional extraneous covariate. Further suppose that the l-th row of $\mathbf{X}$ follows a p-dimensional Gaussian distribution with mean $\mathbf{0} \in \mathbb{R}^p$ and precision matrix $\Omega(z_l) \in \mathbb{R}^{p \times p}$, where $z_l \in \mathbb{R}^q$ is the l-th row of $\mathbf{Z}$ and Ω is a continuous function mapping from the space of extraneous covariates to the space of $p \times p$ non-singular matrices. Then, for the l-th observation, the normality assumption implies that the (j,k)-th entry of $\Omega(z_l)$ is non-zero if, and only if, variable j and variable k are dependent given the remaining variables in $\mathbf{X}$ [Lauritzen 1996]. Given data satisfying these assumptions, we use `covdepGE` to estimate a graphical representation $\mathcal{G}_l$ of the sparsity structure of $\Omega(z_l)$ for each of the observations in $\mathbf{X}$ as a continuous function of $\mathbf{Z}$ by sharing information between observations with similar values of $\mathbf{Z}$. $\mathcal{G}_l$ contains an undirected edge between the node representations of the variables X_j and X_k if, and only if, X_j and X_k are conditionally dependent given the remaining variables for the l-th observation.

We present a pseudo-code representation of the overall algorithm implemented in `covdepGE` in Algorithm 1. Here, we assume that the hyperparameter specification will be fully automated for simplicity, although we note that `covdepGE` permits users to opt out of various parts of the automation as desired. In the sections that follow, we describe the algorithms in the pseudo-code subroutines `get_weights`, `symmetrize`, and `threshold` (Section 2.1), `CAVI` (Section 2.2), `get_bandwidths` (Section 2.3.1), `model_selection` (Section 2.3.2), and `generate_grid` (Section 2.3.3).

2.1 Graph Estimation via the Weighted Pseudo-Likelihood Approach

The weighted pseudo-likelihood approach proposed by Dasgupta et al. [2023] for heterogeneous graph estimation introduces similarity weights to the pseudo-likelihood approach [Atchadé 2019; Besag 1975; Meinshausen and Bühlmann 2006] in order to determine the connectivity of $\mathcal{G}_l$ as a continuous function of $\mathbf{Z}$. In the homogenous setting, we can use the pseudo-likelihood approach to estimate whether the shared CDS for all observations $\mathcal{G}$ has an edge between X_j and X_k by symmetrizing and thresholding the posterior inclusion probabilities (PIP) resulting from 2 regressions. First, X_j is fixed as the response in a multiple linear regression with the remaining variables $\mathbf{X}_{-j} := \{X_m\}_{m=1,m\neq j}^p$ as explanatory variables, giving $\{\text{PIP}_j(X_m)\}_{m=1,m\neq j}^p$, where $\text{PIP}_j(X_m)$ is the posterior probability that the regression coefficient for X_m is non-zero. Second, X_k is fixed as the response in a multiple linear regression with $\mathbf{X}_{-k}$ as explanatory variables to

Algorithm 1: Estimate the CDS of $\mathbf{X}$ as a function of $\mathbf{Z}$, selecting hyperparameters according to `hp_method`

input: $\mathbf{X} \in \mathbb{R}^{n \times p}, \mathbf{Z} \in \mathbb{R}^{n \times q}$, `hp_method` $\in$ {`grid_search`, `model_average`, `hybrid`}

```
// estimate bandwidths and get similarity weights
τ ← get_bandwidths(Z) ∈ ℝ^n;
W ← get_weights(τ) ∈ ℝ^{n×n};
// fix each of the variables in X as the response (this loop is parallelized)
final_models ← list(p);
for j = 1 to p do
    // generate the hyperparameter grid
    Θ_j ← generate_grid(data = X, response = j);
    n_Θ ← |Θ_j|;
    // fit n models for each grid point
    models ← list(n_Θ);
    for t = 1 to n_Θ do
        models_l ← list(n);
        for l = 1 to n do
            // perform variational spike-and-slab weighted with respect to the l-th
            // observation using the t-th hyperparameter grid point
            models_l[l] ← CAVI(data = X, response = j, weights = W_l, hp = Θ_{j,t});
        end
        // save n weighted models corresponding to Θ_{j,t}
        models[t] ← models_l;
    end
    // from the resulting n · n_Θ models, select the final model for each observation
    // according to hp_method
    final_models[j] ← model_selection(models, hp_method);
end
// symmetrize the PIP from the final models and threshold the symmetrized PIP into
// adjacency matrices
PIP ← symmetrize(final_models);
graphs ← threshold(PIP);
```

output: n graphs $\mathcal{G}_l \in \mathbb{R}^{p \times p}$

obtain $\{\text{PIP}_k(X_m)\}_{m=1,m\neq k}^{p}$. $\text{PIP}_j(X_k)$ and $\text{PIP}_k(X_j)$ are then symmetrized with a function mapping from $[0,1] \times [0,1]$ to $[0,1]$ determined by the argument `sym_method` (`"mean"` by default). Then, if the symmetrized PIP is greater than the `edge_threshold` argument (`0.5` by default), we estimate an undirected edge between X_j and X_k.

We extend this approach to the heterogeneous setting and model Ω as a function of $\mathbf{Z}$ by performing n weighted spike-and-slab regressions for each variable X_j fixed as the response. For the l-th regression, we calculate the similarity weight $w_{i,l}$ for observation i with respect to observation l as

$$w_{i,l} = \frac{n\hat{w}_{i,l}}{\sum_{m=1}^{n} \hat{w}_{m,l}}, \tag{1}$$

where

$$\hat{w}_{i,l} = \mathcal{N}\left(\|z_i - z_l\|; 0, \tau_l\right), \tag{2}$$

such that observations having similar values of $\mathbf{Z}$ to observation l will have larger weights. Here, $\|\cdot\|$ denotes the norm specified by the `norm` argument (L^2 by default), z_i and z_l are the values of $\mathbf{Z}$ for the i-th and l-th observations, and $\mathcal{N}$ denotes the Gaussian density function. τ_i is the bandwidth hyperparameter used to determine the similarity weight $w_{i,l}$ of observation i with respect to observation l. Then, the likelihood function for the regression with X_j fixed as the response weighted with respect to the l-th observation is given by

$$L_j^l\left(X_j|\beta_j^l, \mathbf{X}_{-j}, \mathbf{Z}, \tau, \sigma^2\right) = \prod_{i=1}^{n} \mathcal{N}\left(\mathbf{X}_{i,j}; \sum_{k=1,k\neq j}^{p} \mathbf{X}_{i,k}\beta_{j,k}^l, \frac{\sigma^2}{w_{i,l}}\right). \tag{3}$$

In Equation (1), we normalize the weights for the l-th observation to sum to n so that our prior (Equation (4)) does not dominate the likelihood (Equation (3)) in our objective function (Equation (9)). In the case where the unnormalized weights $\hat{w}_{i,l}$ are equal for all i, this ensures that the normalized weights $w_{i,l}$ will equal 1 such that Equation (3) reduces to a conventional regression likelihood function. For graph estimation, we construct $\mathcal{G}_l$ as in the homogeneous setting using the PIP from the regressions weighted with respect to observation l, denoted $\text{PIP}_j^l(\cdot)$.

Through the similarity weights, the weighted pseudo-likelihood approach allows for information sharing conditioned on $\mathbf{Z}$. As opposed to the traditional approach for Bayesian information sharing through a hierarchical modeling scheme, this approach avoids costly MCMC sampling. To add to this efficiency, we observe that since the n regressions for X_j and X_k fixed as the response are independent of each other, we may perform the regressions in parallel by setting `parallel = TRUE`. In factoring parallelism along the variables fixed as the response, we reduce the runtime of the algorithm roughly by a factor of p by registering parallel backend with `num_workers = p`.

2.2 Variational Inference

A common assumption in the GGM literature is sparsity of the precision matrix [Danaher et al. 2014; Friedman et al. 2010; Ren et al. 2022; Yang and Peng 2020], as a small number of conditional dependencies is realistic in many cases, allows for more interpretable graph estimates than a dense graph, and offers a means of dealing with ill-posedness arising in high-dimensional settings. We induce sparsity in $\mathcal{G}$ by placing a spike-and-slab prior on the regression coefficients $\beta_j^l \in \mathbb{R}^{p-1}$ [Mitchell and Beauchamp 1988]. That is, we assume *a priori* that the regression coefficients are independently drawn from a 0-mean Gaussian distribution with variance $\sigma^2\sigma_\beta^2$ with prior inclusion probability π (the slab), and drawn from a point-mass at 0 with probability $1-\pi$ (the spike). Then, for X_j fixed as the response, the prior is given by

$$p_j^0\left(\beta_j^l, \gamma_j^l|\sigma^2, \sigma_\beta^2, \pi\right) = \prod_{k=1,k\neq j}^{p} \left(\pi\mathcal{N}\left(\beta_{j,k}^l; 0, \sigma^2\sigma_\beta^2\right)\right)^{\gamma_{j,k}^l} \left((1-\pi)\,\delta_{\{0\}}\left(\beta_{j,k}^l\right)\right)^{1-\gamma_{j,k}^l}, \tag{4}$$

where $\gamma_{j,k}^l$ is a latent Bernoulli random variable with prior success probability π that takes on 1 if the k-th regression coefficient is drawn from the slab, and 0 otherwise. Thus, the posterior up to a normalizing constant for (β_j^l, γ_j^l) with X_j fixed as the response and weighted with respect to observation l is

$$\nu_j^l\left(\beta_j^l, \gamma_j^l|\mathbf{X}, \tau, \sigma^2, \sigma_\beta^2, \pi\right) \propto p_j^0\left(\beta_j^l, \gamma_j^l|\sigma^2, \sigma_\beta^2, \pi\right) L_j^l\left(X_j|\beta_j^l, \mathbf{X}_{-j}, \mathbf{Z}, \tau, \sigma^2\right). \tag{5}$$

We avoid expensive posterior sampling by approximating all posterior quantities, including $\text{PIP}_j^l(X_k)$, using a block mean-field variational approximation for spike-and-slab regression, as in Carbonetto and Stephens [2012]. Under the

variational paradigm, parameters are estimated by first specifying a variational family of densities $\mathcal{Q}$, and then selecting $q^* \in \mathcal{Q}$ that minimizes the Kullback-Leibler divergence between $\mathcal{Q}$ and the posterior ν [Jordan et al. 1999]. Formally, q^* is given by

$$q^* = \arg\min_{q \in \mathcal{Q}} \left(D_{\mathrm{KL}}(q \| \nu) \right), \tag{6}$$

where the KL divergence is defined as

$$D_{\mathrm{KL}}(q \| \nu) = \mathbb{E}_q \log \frac{q}{\nu}. \tag{7}$$

In practice, the KL divergence is not a tractable quantity, however, minimizing the divergence is equivalent to maximizing the evidence lower bound (ELBO), which is a tractable objective [Blei et al. 2017]. We perform coordinate ascent variational inference (CAVI) to optimize the ELBO over the variational parameters $\phi_j^l = \left(\mu_j^l, (s_j^l)^2, \alpha_j^l\right) \in \mathbb{R}^{p-1 \times 3}$ with respect to the spike-and-slab family $\mathcal{Q}$ defined as

$$\mathcal{Q} = \left\{ \prod_{k=1, k \neq j}^{p} \left(\alpha_{j,k}^l \mathcal{N}\left(\beta_{j,k}^l; \mu_{j,k}^l, (s_{j,k}^l)^2\right)\right)^{\gamma_{j,k}^l} \left(\left(1 - \alpha_{j,k}^l\right) \delta_{\{0\}}\left(\beta_{j,k}^l\right)\right)^{1-\gamma_{j,k}^l} : \mu_{j,k}^l \in \mathbb{R}, (s_{j,k}^l)^2 \in \mathbb{R}^+, \alpha_{j,k}^l \in [0,1] \right\}. \tag{8}$$

For each of the regression coefficients $\beta_{j,k}^l$, these parameters are the mean $\mu_{j,k}^l$ and variance $(s_{j,k}^l)^2$ of the slab and the probability $\alpha_{j,k}^l$ that the coefficient is drawn from $\mathcal{N}(\mu_{j,k}^l, (s_{j,k}^l)^2)$.

We derive the variational updates for each coordinate ascent step by taking the partial derivatives of the ELBO with respect to the variational parameters, where the ELBO is defined as

$$\mathrm{ELBO}(\phi_j^l) = \mathbb{E}_q \log p_j^0\left(\beta_j^l, \gamma_j^l | \sigma^2, \sigma_\beta^2, \pi\right) + \mathbb{E}_q \log L_j^l\left(X_j | \beta_j^l, \mathbf{X}_{-j}, \mathbf{Z}, \tau, \sigma^2\right) - \mathbb{E}_q \log q(\beta_j^l, \gamma_j^l; \phi_j^l). \tag{9}$$

Here, we denote the ELBO of the regression with X_j fixed as the response weighted with respect to observation l as $\mathrm{ELBO}\left(\phi_j^l\right)$ to emphasize that we view this objective as a function of the variational parameters. Finally, we approximate $\mathrm{PIP}_j^l(X_k)$ using $\alpha_{j,k}^l$, which we symmetrize and threshold to construct the graph estimates. In Appendix B.1, we present the full ELBO and variational updates.

As opposed to fixing variable X_j as the response and performing CAVI for the n regressions sequentially, we perform the optimizations simultaneously, using efficient matrix operations where possible. With each of the n sets of α_j^l as the rows of an $n \times (p-1)$ matrix, we end CAVI for all n regressions when the Frobenius norm of the change in the α_j^l matrix is less than the argument `alpha_tol` or the number of CAVI updates exceeds the argument `max_iter`. We further improve runtime by executing the iteration-intensive CAVI entirely with efficient C++ code.

2.3 Automated Hyperparameter Specification

`covdepGE` requires the specification of a bandwidth hyperparameter and spike-and-slab hyperparameters for each regression. Since manual hyperparameter specification can be time consuming and difficult for the user to do effectively, we provide data-driven hyperparameter specification routines within `covdepGE`. We offer the user varying levels of involvement in this procedure, ranging from fully automated to fully manual.

2.3.1 Bandwidth Hyperparameter. The bandwidth hyperparameter $\tau \in \mathbb{R}^n$ regulates the amount of information that is shared through the similarity weights. Smaller values of $\tau_l \in \mathbb{R}$ result in a local estimate of $\mathcal{G}_l$, where the estimate of $\mathcal{G}_l$ is conditioned on information restricted to observations with an extraneous covariate close to z_l. In contrast, larger values of τ_l give rise to a global estimate of $\mathcal{G}_l$. On the extremes, as $\tau_l \to 0$ for all $l \in 1, 2, \ldots, n$, graph estimates are mutually independent for all observations, and as $\tau_l \to \infty$, all observations share a common graph estimate.

We select τ using the 2-step approach for density estimation described by Abramson [1982]. Under this approach, bandwidths are initialized using Silverman's rule of thumb [Silverman 2017], and the density is subsequently refined by updating the bandwidth values. We follow this methodology, which we describe in detail in Appendix B.2, to estimate the density of $\mathbf{Z}$, and use the updated bandwidths from the second step for τ.

2.3.2 Spike-and-slab Hyperparameters. Each spike-and-slab regression requires the specification of 3 hyperparameters: π (the prior probability of inclusion), σ^2 (the prior residual variance), and σ_β^2 (the prior variance of the slab). We offer 3 methods for automated hyperparameter specification via the `hp_method` argument: `grid_search`, `model_average`, and `hybrid`. Under each approach, we generate a hyperparameter candidate grid Θ_j for each variable X_j fixed as the response by taking the Cartesian product between the arguments $\texttt{ssq}_\texttt{j}$, $\texttt{sbsq}_\texttt{j}$, and $\texttt{pip}_\texttt{j}$ (candidate values for σ^2, σ_β^2, and π). We next describe each of these approaches for the regression with X_j fixed as the response weighted with respect to observation l. For this regression, we denote the variational parameter estimates resulting from the choice of hyperparameter $\theta \in \Theta_j$ as $\phi_j^l(\theta) \in \mathbb{R}^{p-1\times 3}$ and the estimates for the final model as $\phi_j^{l^*} \in \mathbb{R}^{p-1\times 3}$. We compare the performance of each of these methods in Section 3.3.1.

In `grid_search`, we select the optimal $\theta^* \in \Theta_j$ by maximizing the total ELBO summed across all n regressions. That is,

$$\theta^* = \underset{\theta\in\Theta_j}{\arg\max} \sum_{l=1}^{n} \text{ELBO}\left(\phi_j^l(\theta)\right), \tag{10}$$

with the final estimates given by $\phi_j^{l^*} = \phi_j^l(\theta^*)$.

Alternatively, we average over Θ_j in `model_average`, where the unnormalized model averaging weights $\hat{\omega}_j^l$ are the exponentiated ELBO as proposed in Carbonetto and Stephens [2012]. For all $\tilde{\theta} \in \Theta_j$, the unnormalized model averaging weights are given by

$$\hat{\omega}_j^l(\tilde{\theta}) = \exp \text{ELBO}\left(\phi_j^l(\tilde{\theta})\right), \tag{11}$$

which are normalized to

$$\omega_j^l(\tilde{\theta}) = \frac{\hat{\omega}_j^l(\tilde{\theta})}{\sum_{\theta\in\Theta_j} \hat{\omega}_j^l(\theta)}. \tag{12}$$

Then, the final variational estimates are averaged over $\theta \in \Theta_j$ as

$$\phi_j^{l^*} = \sum_{\theta\in\Theta_j} \omega_j^l(\theta)\phi_j^l(\theta). \tag{13}$$

Finally, similar to the strategy used by the package `varbvs` [Carbonetto et al. 2017], we combine the previous approaches in `hybrid` by averaging over $\texttt{pip}_\texttt{j}$ while grid searching for $\texttt{ssq}_\texttt{j}$ and $\texttt{sbsq}_\texttt{j}$. For each $\tilde{\pi} \in \texttt{pip}_\texttt{j}$, we select $\theta_{\tilde{\pi}} = \left(\tilde{\pi}, \tilde{\sigma}^2, \tilde{\sigma}_\beta^2\right)$ as

$$\theta_{\tilde{\pi}} = \underset{\theta\in\{\tilde{\pi}\}\times\texttt{ssq}_\texttt{j}\times\texttt{sbsq}_\texttt{j}}{\arg\max} \sum_{l=1}^{n} \text{ELBO}\left(\phi_j^l(\theta)\right). \tag{14}$$

Next, we define the model averaging weights as

$$\hat{\omega}_j^l(\theta_{\tilde{\pi}}) = \exp \text{ELBO}\left(\phi_j^l(\theta_{\tilde{\pi}})\right), \tag{15}$$

$$\omega_j^l(\theta_{\tilde{\pi}}) = \frac{\hat{\omega}_j^l(\theta_{\tilde{\pi}})}{\sum_{\pi \in \texttt{pip}_\texttt{j}} \hat{\omega}_j^l(\theta_\pi)}. \tag{16}$$

Then, the final variational estimates are averaged over $\pi \in \texttt{pip}_\texttt{j}$ as

$$\phi_j^{l^*} = \sum_{\pi \in \texttt{pip}_\texttt{j}} \omega_j^l(\theta_\pi)\phi_j^l(\theta_\pi) \tag{17}$$

To improve the efficiency of the search step in `grid_search` and `hybrid` (Equations (10) and (14)), we perform a reduced CAVI for each of the hyperparameter candidates for at most `max_iter_grid` iterations. We then perform a subsequent full CAVI for at most `max_iter` iterations using the subset of Θ_j that maximized the total ELBO in the first step. By setting $\texttt{max\_iter\_grid} = \frac{1}{h}\texttt{max\_iter}$ for $h > 1$, we further reduce runtime roughly by a factor of h.

2.3.3 Hyperparameter Candidate Grids. We use a data-driven approach to generate the hyperparameter candidate grids $\texttt{ssq}_\texttt{j}$, $\texttt{sbsq}_\texttt{j}$, and $\texttt{pip}_\texttt{j}$ spaced uniformly between an upper end point and lower end point, inclusive. We calculate the upper endpoints dependent on the variable X_j fixed as the response. For $\texttt{ssq}_\texttt{j}$, we calculate the upper endpoint by scaling the sample variance of X_j by the argument `ssq_mult` (1.5 by default). To calculate the upper endpoint for $\texttt{pip}_\texttt{j}$, we use the LASSO in the `glmnet` package [Friedman et al. 2010] and regress the remaining variables $\mathbf{X}_{-j}$ on X_j to estimate the proportion of non-zero coefficients. Although the LASSO assumes that the CDS is homogeneous, it enforces sparsity while giving a rough estimate to the optimal π.

Finally, we derive a rough upper bound on the signal-to-noise ratio that depends on an upper bound for σ_β^2. For a general spike-and-slab regression of $y \in \mathbb{R}^n$ and $X \in \mathbb{R}^{n \times p}$, we first observe that the prior variance of the regression coefficients is given by $\text{Var}(\beta_k) = \pi\sigma^2\sigma_\beta^2$. Then, under the simplifying assumption that the columns of X are pairwise independent, the prior signal-to-noise ratio is given by:

$$\text{SNR} = \frac{\text{Var}(X\beta)}{\text{Var}(y - X\beta)} = \pi\sigma_\beta^2 \sum_{k=1}^{p} \text{Var}(X_k) \tag{18}$$

Any reasonable upper bound for the signal-to-noise ratio can be used for the left-hand side of Equation (18) which we parameterize with the `snr_upper` argument (25 by default). Replacing π with the upper endpoint for $\texttt{pip}_\texttt{j}$ and approximating $\text{Var}(X_k)$ with the sample variance of the k-th column of $\mathbf{X}_{-j}$ on the right-hand side induces a rough upper bound on σ_β^2, which we use as the upper endpoint for $\texttt{sbsq}_\texttt{j}$.

3 SIMULATION STUDY

We perform our simulation study on dual 14-Core (28 threads each) Intel Xeon Gold 6132 processors (@2.6GHz) with 96GB of total RAM. In each setting we consider, we perform 50 trials by first randomly generating an extraneous covariate $\mathbf{Z} \in \mathbb{R}^{n \times q}$ and then generating $\mathbf{X} \in \mathbb{R}^{n \times p}$ conditioned on $\mathbf{Z}$. We define 16 settings by considering all combinations of $q \in \{1, 2, 4\}$ and $p \in \{10, 25, 50, 100\}$, with 2 variants of the $q = 1$ settings (piece-wise linear and non-linear). In all experiments, we fix the overall sample size as $n = 225$.

We evaluate `covdepGE` on three metrics: sensitivity, specificity, and runtime. We define sensitivity as the proportion of edges correctly detected in the ground truth dependence structures $\mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_n$, not including self-loops. We

define specificity as the proportion of non-edges correctly detected. Lastly, we study the real time it takes for `covdepGE` to perform inference which, given that experiments are run on 56 threads, is $O(p)$ less than the CPU time.

We begin by comparing the hyperparameter specification strategies discussed in Section 2.3.2 in the $q = 1$, piece-wise linear settings before moving on to compare `covdepGE` to 3 baselines for heterogeneous graphical modeling in R: `JGL` [Danaher et al. 2014], `loggle` [Yang and Peng 2020], and `mgm` [Haslbeck and Waldorp 2020]. These baselines are a representative sample of the currently available packages for modeling heterogeneous graphs, as `loggle` and `mgm` assume that the precision matrices vary with time, while `JGL` assumes that the data can be partitioned into subgroups such that precision matrices are homogeneous within each of the subgroups. We note that since `JGL` and `mgm` are not implemented with parallelism, the real time we report for them is equal to their CPU time. In Appendix D.2, we report the runtime of `covdepGE` run without parallelism in the $q = 1$, piece-wise linear settings.

3.1 Baselines

Both `loggle` [Yang and Peng 2020] and the `mgm::tvmgm` function [Haslbeck and Waldorp 2020] implement regularized kernel-smoothing methods to model precision matrices as varying continuously in time. For `mgm`, we use the `mgm::bwSelect` cross-validation function to select the bandwidth hyperparameter with candidate values and settings based on a high-dimensional Gaussian example provided by the authors of the package which we detail in Appendix C.1. Similarly, we use the `loggle::loggle.cv` function to select the bandwidth, neighborhood width, and shrinkage hyperparameters for `loggle`, which we discuss in greater detail in Appendix C.2.

We note that although in the settings with $q = 1$, it is natural to frame $\mathbf{Z}$ as indexing time, `loggle` and `mgm` are not directly applicable for $q > 1$. Therefore, we sort $\mathbf{Z}$ to $\tilde{\mathbf{Z}}$ for these settings using a greedy algorithm described in Appendix C.3 such that the observation indices $T = \{1, 2, \ldots, n\}$ mapping $\mathbf{Z}$ to $\tilde{\mathbf{Z}}$ can be interpreted as time. We then apply `loggle` and `mgm` on this data, using the single-dimensional T as the time index. To demonstrate the efficacy of this approach, we evaluate `covdepGE` first using $\mathbf{Z}$ as the extraneous covariate, and then using T. We refer to the latter as `covdepGE_time`.

The `JGL` package applies the fused graphical LASSO to jointly estimate the precision matrices for each of the subgroups from a given partition of the data. To incorporate information from $\mathbf{Z}$, we apply Gaussian mixture model-based clustering with the package `Mclust` [Scrucca et al. 2016] to partition $\mathbf{X}$ into K subgroups based on $\mathbf{Z}$, selecting K by minimizing the BIC. Since `JGL` does not offer any specification routines for the shrinkage hyperparameters λ_1 and λ_2, we implement the 2-step approach proposed in Section 6 of Danaher et al. [2014] to minimize the approximate AIC over a grid of λ_1 and λ_2.

We additionally considered `HeteroGGM` [Ren et al. 2021] as a competitor. While `HeteroGGM` is also a penalized fusion-based method similar to `JGL`, it does not assume that the subgroups of the data are known *a priori*, and instead estimates subgroup membership jointly with precision matrices. However, `HeteroGGM` requires the subgroup means to be sufficiently separable to estimate membership. Since all observations in our simulation study were drawn from a $\mathbf{0}$-mean Gaussian, we found `JGL` coupled with `Mclust` to be a more appropriate choice.

3.2 Data Generation

For each trial in all settings, we begin by generating $\mathbf{Z} \in \mathbb{R}^{n \times q}$. For the piece-wise linear $q = 1$ setting, we draw 75 samples each from the uniform distribution on the intervals $(-3, -1)$, $(-1, 1)$, and $(1, 3)$. For the non-linear $q = 1$ setting, we again ensure that each of the 3 CDS shown in Figure 3 each correspond to 75 observations by drawing 37 samples each from the uniform distribution on the intervals $(-3, -2)$ and $(1, 2)$, 38 samples each from the uniform distribution on

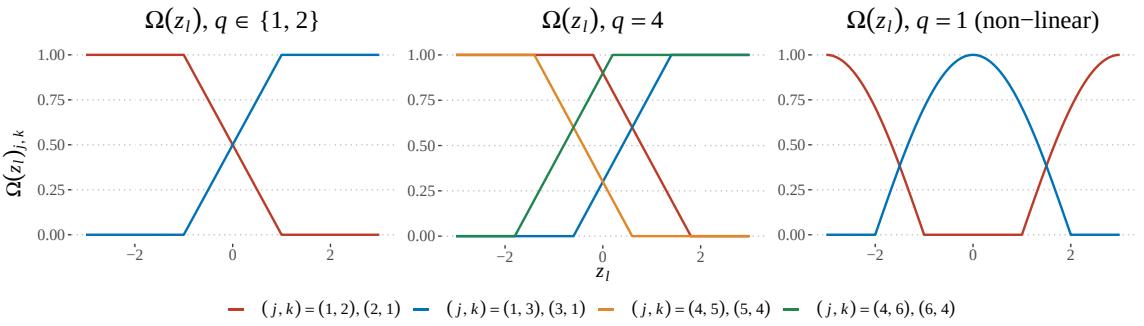

Fig. 2. $\Omega(z_l)_{j,k}$ for various settings. j, k are the indices of non-zero, covariate-dependent entries in the precision matrix which vary with the extraneous covariate z_l given in Equations (19), (22) and (23). *Left*: $q \in \{1, 2\}$, piece-wise linear. *Middle*: $q = 4$, *Right*: $q = 1$, non-linear.

the intervals $(-2, -1)$ and $(2, 3)$, and 75 samples from the uniform distribution on the interval $(-1, 1)$. For $q = 2$, we draw 25 times each from the uniform distribution on the 9 regions defined by $\{Z_1 \times Z_2 : Z_1, Z_2 \in \{[-3, -1], [-1, 1], [1, 3]\}\}$. Lastly, for $q = 4$, we draw 225 times from the uniform distribution on $[-3, 3]^4$.

We next generate $\mathbf{X} \in \mathbb{R}^{n \times p}$ with a precision matrix $\Omega(z_l) \in \mathbb{R}^{p \times p}$ that varies as a continuous function of $\mathbf{Z}$. It is important to note that while the off-diagonal non-zero entries in $\Omega(z_l)$ are adjacent in all settings for ease of visualization, this has no effect on the performance of any of the considered methods. Alternative enumerations of the variables would permute the non-zero entries but yield identical estimates to those presented here, albeit permuted.

3.2.1 $q = 1$ and $q = 2$, Piece-Wise Linear. We begin by discussing data generation for the $q \in \{1, 2\}$ settings where the relationship between the extraneous covariate and precision matrices is piece-wise linear (PWL), as here, $q = 1$ can be seen as a special case of $q = 2$. For $q = 1$, we let z_l be the extraneous covariate for the l-th observation and define $z_{l,1} = z_{l,2} = z_l \in \mathbb{R}$, while for $q = 2$, we define $(z_{l,1}, z_{l,2}) = z_l \in \mathbb{R}^2$. Then, for $q \in \{1, 2\}$, the j, k entry of the precision matrix for the l-th observation is given by

$$\Omega(z_l)_{j,k} = \begin{cases} 2 & j = k \\ 1 & (j,k) \in \{(2,3),(3,2)\} \\ \max\left(0, \min\left(1, -\frac{1}{2}\left(z_{l,1} - 1\right)\right)\right) & (j,k) \in \{(1,2),(2,1)\} \\ \max\left(0, \min\left(1, \frac{1}{2}\left(z_{l,2} + 1\right)\right)\right) & (j,k) \in \{(1,3),(3,1)\} \\ 0 & \textit{otherwise.} \end{cases} \tag{19}$$

We note that for $q = 1$, all 75 observations with $z_l \in [-3, -1]$ have identical precision matrices, as do observations with $z_l \in [1, 3]$. We further observe that for observations with $z_l \in [-1, 1]$, the entries of $\Omega(z_l)$ vary linearly with z_l, and that

$$\lim_{z_l \downarrow -1} \|\Omega(z_l) - \Omega(-1)\| = 0, \tag{20}$$

and similarly that

$$\lim_{z_l \uparrow 1} \|\Omega(z_l) - \Omega(1)\| = 0, \tag{21}$$

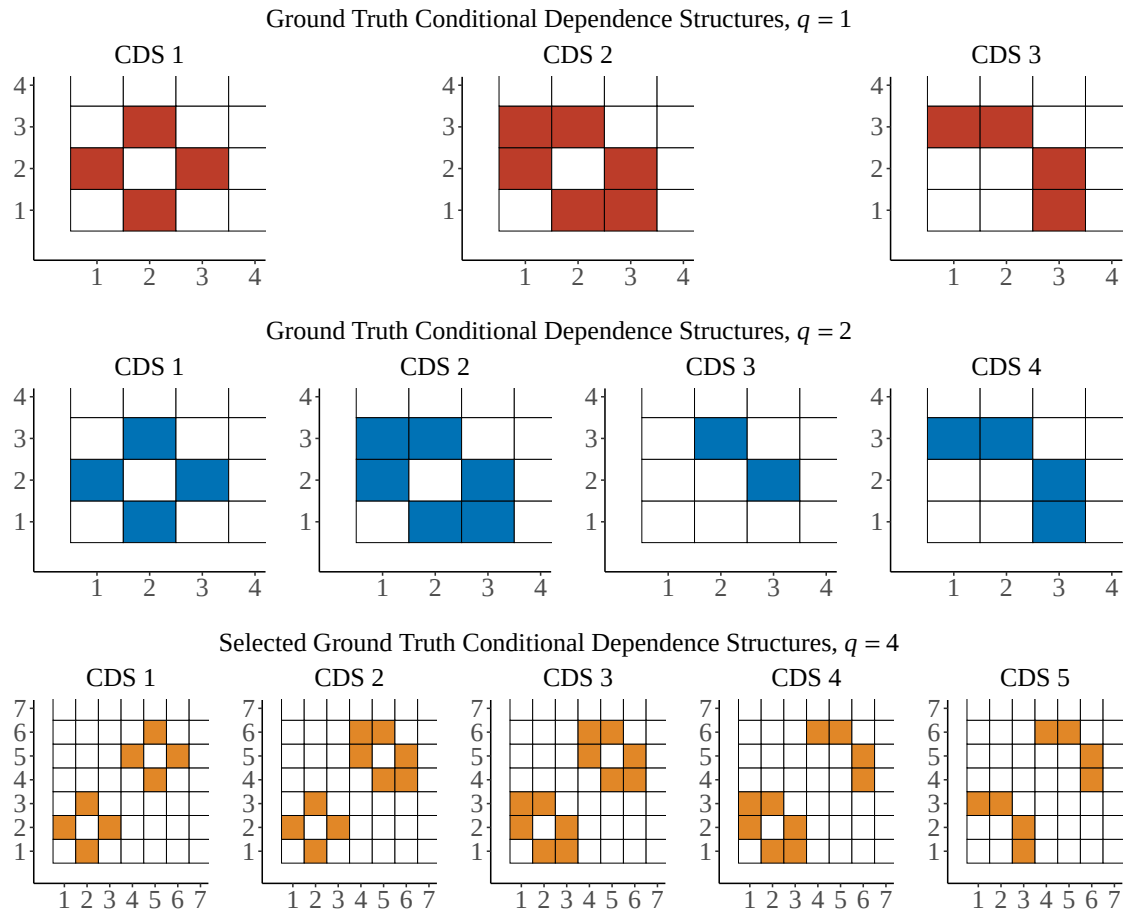

Fig. 3. Upper-left 3×3 block of the ground truth conditional dependence structure visualized using `covdepGE::matViz`. The remaining entries are all 0. *Top*: CDS for $q = 1$ settings. Piece-wise linear settings: **CDS 1** – $z_l \in [-3, -1]$, **CDS 2** – $z_l \in [-1, 1]$, **CDS 3** – $z_l \in [1, 3]$. Non-linear settings: **CDS 1** – $z_l \in [-3, -2] \cup [2, 3]$, **CDS 2** – $z_l \in [-2, -1] \cup [1, 2]$, **CDS 3** – $z_l \in [-1, 1]$. *Middle*: CDS for $q = 2$ settings. **CDS 1** – $z_l \in ([-3, -1] \times [-3, -1]) \cup ([-1, 1] \times [-3, -1])$, **CDS 2** – $z_l \in ([-3, -1] \times [-1, 1]) \cup ([-3, -1] \times [1, 3]) \cup ([-1, 1] \times [-1, 1]) \cup ([-1, 1] \times [1, 3])$, **CDS 3** – $z_l \in ([1, 3] \times [-3, -1])$, **CDS 4** – $z_l \in ([1, 3] \times [-1, 1]) \cup ([1, 3] \times [1, 3])$. *Bottom*: CDS in selected intervals for $q = 4$ settings (since there are 16 possible CDS for $q = 4$, we only visualize 5 here). **CDS 1** – $z_l \in [-3, -\frac{9}{5}]^4$, **CDS 2** – $z_l \in [-\frac{9}{5}, -\frac{3}{5}]^4$, **CDS 3** – $z_l \in [-\frac{3}{5}, \frac{3}{5}]^4$, **CDS 4** – $z_l \in [\frac{3}{5}, \frac{9}{5}]^4$, **CDS 5** – $z_l \in [\frac{9}{5}, 3]^4$.

where $\|\cdot\|$ denotes any matrix norm. That is, the precision matrix for observations with z_l in interval 2 converges to the precision matrix for the observations with extraneous covariates in interval 3 as z_l approaches interval 3, and similarly for z_l approaching interval 1, thereby ensuring the continuity of Ω. This continuity holds in all settings, as demonstrated in Figure 2 where we visualize the value of the covariate-dependent entries in $\Omega(z_l)$ as a function of z_l for all settings. We visualize the conditional dependence structures corresponding to these precision matrices for all settings in Figure 3.

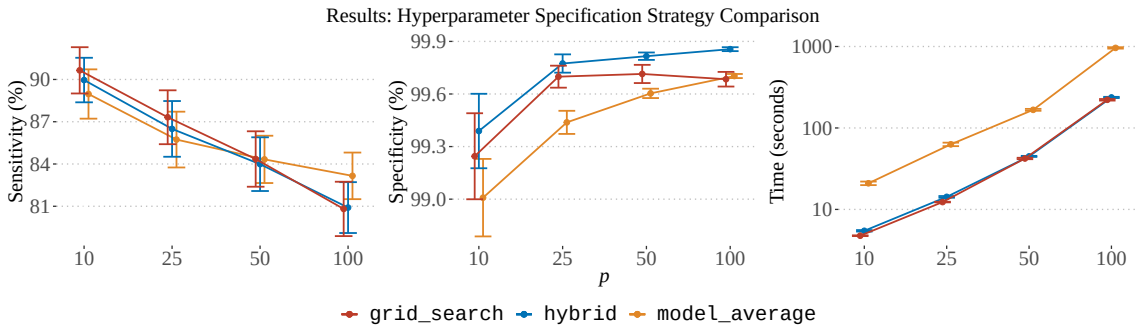

Fig. 4. Comparison of hyperparameter specification strategies in the $q = 1$, piece-wise linear settings averaged across 50 trials. Error bars are 2 times the standard error of the mean. Sensitivity and Specificity are defined as the proportion of edges and non-edges correctly recovered from the ground-truth CDS excluding self-loops. Time is the elapsed real time for the model to fit in seconds.

3.2.2 $q = 4$, *Piece-Wise Linear.* For $q = 4$, we define $(z_{l,1}, z_{l,2}, z_{l,3}, z_{l,4}) = z_l \in \mathbb{R}^4$. Then, the j, k entry of the precision matrix for the l-th observation is given by

$$\Omega(z_l)_{j,k} = \begin{cases} 2 & j = k \\ 1 & (j,k) \in \{(2,3),(3,2),(5,6),(6,5)\} \\ \max\left(0, \min\left(1, -\frac{1}{2}\left(z_{l,1} - \frac{9}{5}\right)\right)\right) & (j,k) \in \{(1,2),(2,1)\} \\ \max\left(0, \min\left(1, \frac{1}{2}\left(z_{l,2} + \frac{3}{5}\right)\right)\right) & (j,k) \in \{(1,3),(3,1)\} \\ \max\left(0, \min\left(1, -\frac{1}{2}\left(z_{l,3} - \frac{3}{5}\right)\right)\right) & (j,k) \in \{(4,5),(5,4)\} \\ \max\left(0, \min\left(1, \frac{1}{2}\left(z_{l,4} + \frac{9}{5}\right)\right)\right) & (j,k) \in \{(4,6),(6,4)\} \\ 0 & \textit{otherwise.} \end{cases} \tag{22}$$

3.2.3 $q = 1$, *Non-Linear.* We lastly discuss data generation for the $q = 1$, non-linear (NL) setting. Here we have $z_l \in \mathbb{R}$, with the j, k entry of the precision matrix for the l-th observation given by

$$\Omega(z_l)_{j,k} = \begin{cases} 2 & j = k \\ 1 & (j,k) \in \{(2,3),(3,2)\} \\ \max\left(0, \cos\left(\frac{\pi}{4}(z_l - 3)\right)\right) + \max\left(0, \cos\left(\frac{\pi}{4}(z_l + 3)\right)\right) & (j,k) \in \{(1,2),(2,1)\} \\ \max\left(0, \cos\left(\frac{\pi}{4} z_l\right)\right) & (j,k) \in \{(1,3),(3,1)\} \\ 0 & \textit{otherwise.} \end{cases} \tag{23}$$

In Equation (23), π denotes its usual numerical value, as opposed to the prior probability of inclusion as we have used previously.

3.3 Results

3.3.1 $q = 1$, *Piece-Wise Linear.* We begin by comparing the performance of different hyperparameter specification strategies in the $q = 1$, PWL settings before moving on to study the performance of `covdepGE` relative to the baselines.

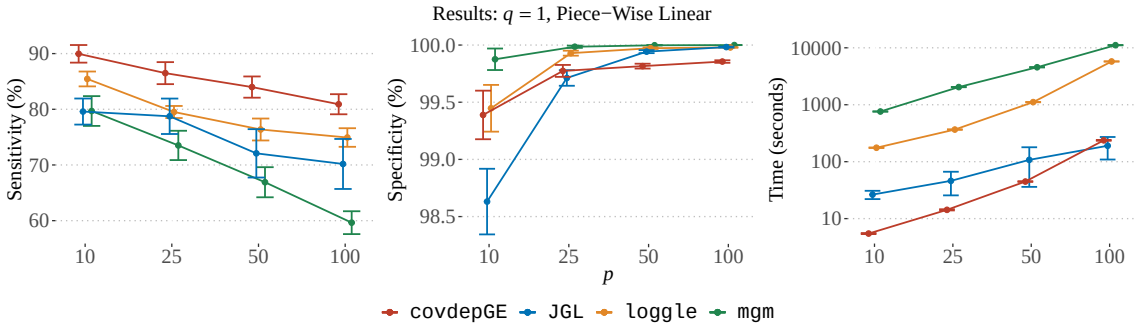

Fig. 5. Comparison of heterogeneous graphical modeling methods in the $q = 1$, PWL settings. `JGL` is the method of Danaher et al. [2014], where the subgroups of the data are chosen by applying Gaussian mixture models clustering to $\mathbf{Z}$. `loggle` is the method of Yang and Peng [2020], and `mgm` is the method of Haslbeck and Waldorp [2020].

We present results for each hyperparameter specification strategy in Figure 4, with the numerical results corresponding to the plots in all settings shown in Appendix D.1. We observe that for larger p, `model_average` detects more edges correctly but is substantially slower than `grid_search` and `hybrid`. As described in Section 2.3.2, this is because `grid_search` and `hybrid` both perform an initial abbreviated CAVI for each hyperparameter candidate $\theta \in \Theta_j$ for at most `max_iter_grid` iterations (10 by default). Based on this initial search, they select an optimal subset of Θ_j and perform a full CAVI for at most `max_iter` iterations (100 by default). By contrast, since all $\theta \in \Theta_j$ contribute to the final model produced by `model_average`, CAVI must run for the full `max_iter` iterations (or until convergence) for all $\theta \in \Theta_j$, resulting in substantially increased overhead. To assess whether performing CAVI for the full number of steps in the initial search substantially changes results, we repeat this experiment in Appendix D.3, setting `max_iter_grid` = `max_iter` for `hybrid` and `grid_search`. Unsurprisingly, the runtime for all strategies after this modification is roughly equal. We also observe negligible changes in sensitivity and specificity, suggesting the effectiveness of selecting the optimal subset of Θ_j based on an abbreviated search. We note that `hybrid` offers the best specificity while estimating the second most edges correctly in all settings except $p = 50$. For this reason, in addition to the speed-up relative to `model_average`, we select `hybrid` as the default hyperparameter specification strategy.

We next present results for `covdepGE` and the baselines in Figure 5. `covdepGE` improves the baseline sensitivity by over 4% in all settings. In Appendix D.5, we conduct a statistical analysis which reveals the statistical significance of these improvements in sensitivity. We note that although `mgm` has the lowest sensitivity, its conservative estimates allow it to excel in specificity. However, the runtime of `mgm` is more than 45 times greater than `covdepGE` in all settings. While `loggle` is second to `mgm` in terms of specificity, it has the highest baseline sensitivity. `loggle` additionally offers parallelization, however, in the $p = 100$ settings, errors due to excess memory usage are raised when using a moderate number of workers. We therefore only used 15 workers, and thus, the runtime of `loggle` is more than 20 times that of `covdepGE`.

Compared to `JGL`, `covdepGE` is twice as fast in all settings except for $p = 100$, although we observe a large amount of variability in the runtimes for `JGL`. During the shrinkage hyperparameter search in some trials, `JGL` would iterate for an abnormally long time on smaller values of λ_1. This is because smaller λ_1 result in less sparse estimates that require a greater amount of computation during optimization for operations such as matrix inversion. We remove the effects of

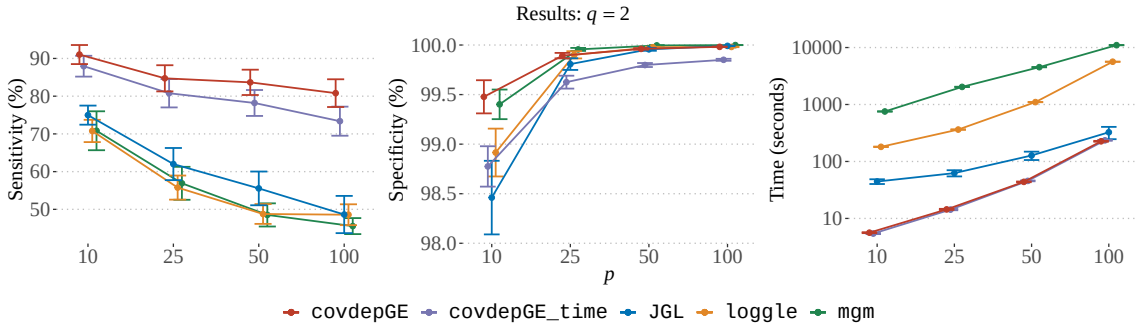

Fig. 6. Comparison of heterogeneous graphical modeling methods in the $q = 2$ settings. The time index for `loggle`, `mgm` and `covdepGE_time` is obtained by applying a greedy sorting algorithm to $\mathbf{Z}$.

outliers from our analysis in Appendix D.7 by aggregating results using median in place of mean and interquartile range in place of standard error.

Although `JGL` is faster on average than `covdepGE` in the $p = 100$ setting, `JGL` requires users to either directly specify hyperparameters *ad hoc* (a task that is often time-consuming and difficult for the user to do effectively) or to manually implement a hyperparameter specification scheme as we did. Additionally, while `covdepGE` estimates the graphs as a continuous function of $\mathbf{Z}$, `JGL` relies on a discretization of the covariate space that is independent of graph estimation via `Mclust` to incorporate $\mathbf{Z}$, and incorrectly models the precision matrix as homogeneous within each of the subgroups.

3.3.2 $q = 2$, Piece-Wise Linear. We present results for $q = 2$ in Figure 6. We observe that `covdepGE` improves the baseline sensitivity in all settings by over 15%. Unlike the $q = 1$, PWL settings, `covdepGE` has a faster runtime than `JGL` in all settings including $p = 100$. As we show in Appendix D.8, this is due to the increase in the number of subgroups estimated by `Mclust` relative to the $q = 1$, PWL settings. Because `JGL` jointly estimates the precision matrix in each of the subgroups, a greater number of subgroups increases computation.

`mgm` demonstrates the best specificity in all settings except $p = 10$. However, due to their assumption of a temporally-varying precision matrix, the application of both `mgm` and `loggle` to $q > 1$ settings was not straightforward as it was for `covdepGE`. We note that although `covdepGE_time` outperforms the baselines in sensitivity, its performance is substantially reduced compared to `covdepGE`. This highlights the importance of directly incorporating multidimensional covariates into the estimation procedure as opposed to relying on a reduction of the covariate.

3.3.3 $q = 4$, Piece-Wise Linear. We present results for the $q = 4$ settings in Figure 7. `covdepGE` has the highest sensitivity in all settings except for $p = 100$, where it is second to `covdepGE_time`. However, `covdepGE` outperforms the specificity of `covdepGE_time` in each setting.

As shown in Appendix D.8, the number of clusters estimated by `Mclust` decreases relative to the previously considered settings, although the number of unique ground truth CDS increases. As the dimensionality of the covariate $\mathbf{Z}$ increases, the variance in the distances between arbitrary covariate values z_i and z_l decreases, known as the curse of dimensionality, resulting in a fewer number of clusters. Because fewer clusters reduces the amount of computation for `JGL`, the runtime in the $p = 100$ setting is faster than that of `covdepGE`.

Similarly, due to this effect, the similarity weights shown in Equation (2) converge to uniform as q grows large, as they are computed using the distance between z_i and z_l. The result is a reduced amount of heterogeneity in the

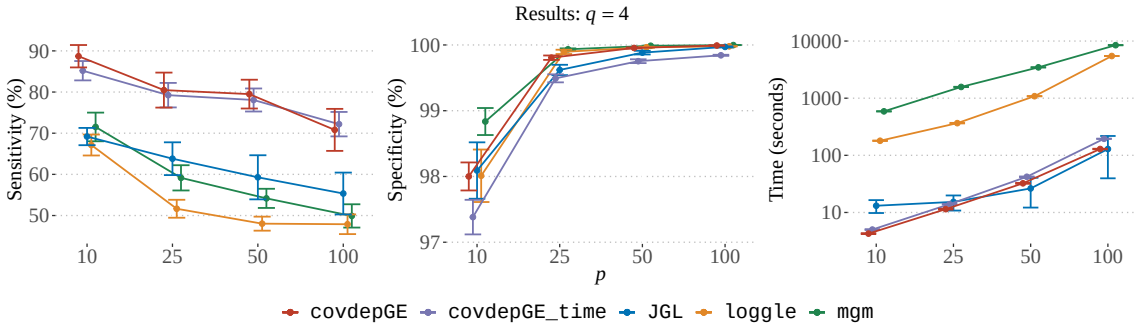

Fig. 7. Comparison of heterogeneous graphical modeling methods in the $q = 4$ settings.

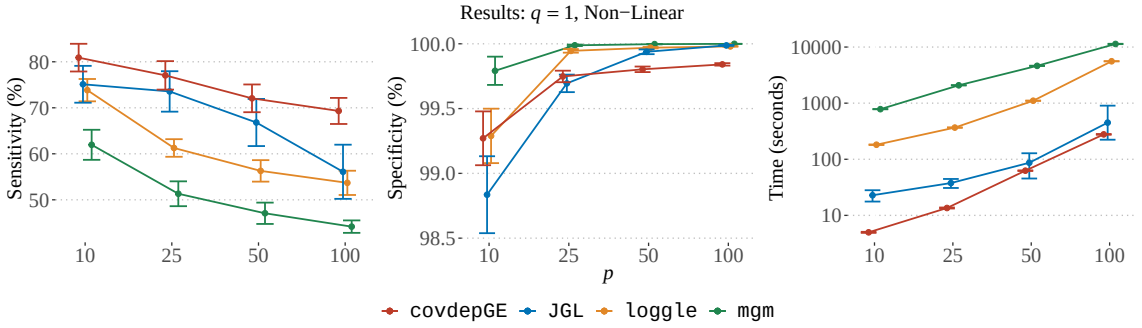

Fig. 8. Comparison of heterogeneous graphical modeling methods in the $q = 1$, non-linear settings. The lower error bar for the runtime of JGL in the $p = 100$ setting is only 1 standard error (226.58) instead of 2, as 2 standard errors below the mean (449.65) is less than 0.

estimated graphs. We analyze this in Appendix D.6, where we find that although the $q = 4$ settings have the greatest number of unique ground truth CDS, the number of unique graphs estimated by covdepGE is less than the other settings. Future work should look to increase the robustness of heterogeneous graph estimation with larger q.

3.3.4 $q = 1$, Non-Linear. We present results for $q = 1$, NL settings in Figure 8. As shown in Figures 2 and 3, these settings present a challenge unique from the $q = 1$, PWL settings previously considered, as the observations corresponding to CDS 1 have covariate values z_i which are roughly evenly divided between the intervals $[-3, -2]$ and $[2, 3]$. During graph estimation, it is therefore more difficult for information to be shared between observation i with $z_i \in [-3, -2]$ and observation l with $z_l \in [2, 3]$, despite their shared graph. Similarly, observations corresponding to CDS 2 have their covariate values divided between $[-2, -1]$ and $[1, 2]$, and thus, estimation of graphs for these observations is challenging for the same reason as CDS 1. However, since all observations corresponding to CDS 3 have covariates $z_i \in [-1, 1]$, estimation for these observations is of a similar difficulty as in the PWL settings.

In Appendix D.4, we compare the sensitivity for covdepGE broken down by CDS for both PWL and NL $q = 1$ settings. As expected, while sensitivities for CDS 3 are similar, there is a substantial drop in the NL settings in the ability detect edges correctly for CDS 1 and CDS 2. Nonetheless, as shown in Figure 8, covdepGE is still the top performer in terms of sensitivity and runtime.

4 CONCLUSION

Our package leverages parallelism and C++ code on top of the efficient algorithm proposed in Dasgupta et al. [2023] to further accelerate inference. We additionally provide fully-automated and effective hyperparameter specification. As we have demonstrated in a variety of settings through our simulation study, `covdepGE` efficiently manages computational overhead and consistently recovers the edges of the ground-truth CDS as a function of $\mathbf{Z}$ more reliably than baseline methods for heterogeneous graph estimation. In addition to applications in genetics as illustrated by the case study in Dasgupta et al. [2023], we expect `covdepGE` to find applications across diverse disciplines due to its ability to incorporate extraneous covariates in graph estimation for data that are not necessarily identically distributed. Future work should improve the scalability of the algorithm to high-dimensional settings. Later versions (>1.0.1) of `covdepGE` will be made available on CRAN[1] and GitHub[2].

REFERENCES

Ian S. Abramson. 1982. On Bandwidth Variation in Kernel Estimates-A Square Root Law. *The Annals of Statistics* 10, 4 (1982), 1217–1223. https://www.jstor.org/stable/2240724 Publisher: Institute of Mathematical Statistics.

Yves F. Atchadé. 2019. Quasi-Bayesian estimation of large Gaussian graphical models. *Journal of Multivariate Analysis* 173 (Sept. 2019), 656–671. https://doi.org/10.1016/j.jmva.2019.03.005

Julian Besag. 1975. Statistical Analysis of Non-Lattice Data. *Journal of the Royal Statistical Society. Series D (The Statistician)* 24, 3 (1975), 179–195. https://doi.org/10.2307/2987782 Publisher: [Royal Statistical Society, Wiley].

David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. 2017. Variational Inference: A Review for Statisticians. *J. Amer. Statist. Assoc.* 112, 518 (April 2017), 859–877. https://doi.org/10.1080/01621459.2017.1285773

Peter Carbonetto and Matthew Stephens. 2012. Scalable Variational Inference for Bayesian Variable Selection in Regression, and Its Accuracy in Genetic Association Studies. *Bayesian Analysis* 7, 1 (March 2012), 73–108. https://doi.org/10.1214/12-BA703

Peter Carbonetto, Xiang Zhou, and Matthew Stephens. 2017. varbvs: Fast Variable Selection for Large-scale Regression. https://doi.org/10.48550/arXiv.1709.06597

Patrick Danaher, Pei Wang, and Daniela M. Witten. 2014. The joint graphical lasso for inverse covariance estimation across multiple classes. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 76, 2 (2014), 373–397. https://doi.org/10.1111/rssb.12033

Priyam Das, Christine B Peterson, Kim-Anh Do, Rehan Akbani, and Veerabhadran Baladandayuthapani. 2020. NExUS: Bayesian simultaneous network estimation across unequal sample sizes. *Bioinformatics* 36, 3 (Feb. 2020), 798–804. https://doi.org/10.1093/bioinformatics/btz636

Sutanoy Dasgupta, Peng Zhao, Jacob Helwig, Prasenjit Ghosh, Debdeep Pati, and Bani K. Mallick. 2023. An Approximate Bayesian Approach to Covariate-dependent Graphical Modeling. https://doi.org/10.48550/arXiv.2303.08979 arXiv:2303.08979 [stat].

Jerome Friedman, Trevor Hastie, and Rob Tibshirani. 2008. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics* 9, 3 (July 2008), 432–441. https://doi.org/10.1093/biostatistics/kxm045

Jerome Friedman, Trevor Hastie, and Rob Tibshirani. 2010. Regularization Paths for Generalized Linear Models via Coordinate Descent. *Journal of statistical software* 33, 1 (2010), 1–22. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2929880/

Chen Gao, Yunzhang Zhu, Xiaotong Shen, and Wei Pan. 2016. Estimation of multiple networks in Gaussian mixture models. *Electronic journal of statistics* 10 (2016), 1133–1154. https://doi.org/10.1214/16-EJS1135

Botao Hao, Will Wei Sun, Yufeng Liu, and Guang Cheng. 2018. Simultaneous Clustering and Estimation of Heterogeneous Graphical Models. *Journal of Machine Learning Research* 18, 217 (2018), 1–58. http://jmlr.org/papers/v18/17-019.html

Jonas M. B. Haslbeck and Lourens J. Waldorp. 2020. mgm: Estimating Time-Varying Mixed Graphical Models in High-Dimensional Data. *Journal of Statistical Software* 93 (April 2020), 1–46. https://doi.org/10.18637/jss.v093.i08

Michael I. Jordan, Zoubin Ghahramani, Tommi S. Jaakkola, and Lawrence K. Saul. 1999. An Introduction to Variational Methods for Graphical Models. *Machine Learning* 37, 2 (Nov. 1999), 183–233. https://doi.org/10.1023/A:1007665907178

Steffen L. Lauritzen. 1996. *Graphical Models.* Clarendon Press.

Chunshan Liu, Daniel R. Kowal, and Marina Vannucci. 2022. Dynamic and robust Bayesian graphical models. *Statistics and Computing* 32, 6 (Nov. 2022), 105. https://doi.org/10.1007/s11222-022-10177-0

Han Liu, Xi Chen, Larry Wasserman, and John Lafferty. 2010. Graph-Valued Regression. In *Advances in Neural Information Processing Systems*, Vol. 23. Curran Associates, Inc., 1–9. https://proceedings.neurips.cc/paper/2010/hash/821fa74b50ba3f7cba1e6c53e8fa6845-Abstract.html

Subhabrata Majumdar and George Michailidis. 2022. Joint Estimation and Inference for Data Integration Problems based on Multiple Multi-layered Gaussian Graphical Models. *Journal of Machine Learning Research* 23 (2022), 1–53.

[1] `https://cran.r-project.org/package=covdepGE`
[2] `https://github.com/JacobHelwig/covdepGE/`

Giovanni M. Marchetti. 2006. Independencies Induced from a Graphical Markov Model After Marginalization and Conditioning: The R Package ggm. *Journal of Statistical Software* 15 (Feb. 2006), 1–15. https://doi.org/10.18637/jss.v015.i06

Nicolai Meinshausen and Peter Bühlmann. 2006. High-dimensional graphs and variable selection with the Lasso. *The Annals of Statistics* 34, 3 (June 2006), 1436–1462. https://doi.org/10.1214/009053606000000281

T. J. Mitchell and J. J. Beauchamp. 1988. Bayesian Variable Selection in Linear Regression. *J. Amer. Statist. Assoc.* 83, 404 (Dec. 1988), 1023–1032. https://doi.org/10.1080/01621459.1988.10478694

Yang Ni, Francesco C. Stingo, and Veerabhadran Baladandayuthapani. 2019. Bayesian Graphical Regression. *J. Amer. Statist. Assoc.* 114, 525 (Jan. 2019), 184–197. https://doi.org/10.1080/01621459.2017.1389739 Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/01621459.2017.1389739.

Yang Ni, Francesco C. Stingo, and Veerabhadran Baladandayuthapani. 2022. Bayesian covariate-dependent Gaussian graphical models with varying structure. *The Journal of Machine Learning Research* 23, 1 (Jan. 2022), 242:11049–242:11077.

Christine B. Peterson, Francesco C. Stingo, and Marina Vannucci. 2015. Bayesian Inference of Multiple Gaussian Graphical Models. *J. Amer. Statist. Assoc.* 110, 509 (March 2015), 159–174. https://doi.org/10.1080/01621459.2014.896806

Huitong Qiu, Fang Han, Han Liu, and Brian Caffo. 2016. Joint Estimation of Multiple Graphical Models from High Dimensional Time Series. *Journal of the Royal Statistical Society Series B: Statistical Methodology* 78, 2 (March 2016), 487–504. https://doi.org/10.1111/rssb.12123

Mingyang Ren, Sanguo Zhang, Qingzhao Zhang, and Shuangge Ma. 2021. HeteroGGM: an R package for Gaussian graphical model-based heterogeneity analysis. *Bioinformatics* 37, 18 (Sept. 2021), 3073–3074. https://doi.org/10.1093/bioinformatics/btab134

Mingyang Ren, Sanguo Zhang, Qingzhao Zhang, and Shuangge Ma. 2022. Gaussian graphical model-based heterogeneity analysis via penalized fusion. *Biometrics* 78, 2 (2022), 524–535. https://doi.org/10.1111/biom.13426

Luca Scrucca, Michael Fop, T. Brendan Murphy, and Adrian E. Raftery. 2016. mclust 5: Clustering, Classification and Density Estimation Using Gaussian Finite Mixture Models. *The R journal* 8, 1 (Aug. 2016), 289–317. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5096736/

Elin Shaddox, Francesco C. Stingo, Christine B. Peterson, Sean Jacobson, Charmion Cruickshank-Quinn, Katerina Kechris, Russell Bowler, and Marina Vannucci. 2018. A Bayesian Approach for Learning Gene Networks Underlying Disease Severity in COPD. *Statistics in biosciences* 10, 1 (2018), 59–85. https://doi.org/10.1007/s12561-016-9176-6

B. W. Silverman. 2017. *Density Estimation for Statistics and Data Analysis.* Routledge, New York. https://doi.org/10.1201/9781315140919

R Core Team. 2021. R: A Language and Environment for Statistical Computing. https://www.R-project.org/

Veronica Vinciotti, Luigi Augugliaro, Antonino Abbruzzo, and Ernst C. Wit. 2016. Model selection for factorial Gaussian graphical models with an application to dynamic regulatory networks. *Statistical Applications in Genetics and Molecular Biology* 15, 3 (June 2016), 193–212. https://doi.org/10.1515/sagmb-2014-0075

Hadley Wickham. 2016. *ggplot2: Elegant Graphics for Data Analysis.* Springer-Verlag New York. https://ggplot2.tidyverse.org

Donald Williams and Joris Mulder. 2020. BGGM: Bayesian Gaussian Graphical Models in R. *Journal of Open Source Software* 5, 51 (July 2020), 2111. https://doi.org/10.21105/joss.02111

Jilei Yang and Jie Peng. 2020. Estimating Time-Varying Graphical Models. *Journal of Computational and Graphical Statistics* 29, 1 (Jan. 2020), 191–202. https://doi.org/10.1080/10618600.2019.1647848 Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/10618600.2019.1647848.

Jingfei Zhang and Yi Li. 2023. High-Dimensional Gaussian Graphical Regression Models with Covariates. *J. Amer. Statist. Assoc.* 118, 543 (July 2023), 2088–2100. https://doi.org/10.1080/01621459.2022.2034632 Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/01621459.2022.2034632.

A NOTATION

We provide an overview of notation in Table 1.

Table 1. Notation. We refer to the regression with X_j fixed as the response weighted with respect to the l-th observation as the (j, l)-th regression.

Notation	Meaning
$\mathbf{X}$	Data matrix of shape $n \times p$
X_j	j-th column of $\mathbf{X}$
$\mathbf{X}_{-j}$	$\mathbf{X}$ with j-th column removed
$\mathbf{Z}$	Extraneous covariate matrix of shape $n \times q$
z_l	l-th row of $\mathbf{Z}$
Ω	Precision matrix of shape $p \times p$
τ	Bandwidth hyperparameters
$w_{i,l}$	Similarity weight for observation i with respect to observation l
β_j^l	Coefficients for (j, l)-th regression
γ_j^l	Vector of Bernoulli variables for (j, l)-th regression
p_j^0	Prior for j-th regressions
L_j^l	Likelihood function for (j, l)-th regression
ν_j^l	Posterior for (j, l)-th regression
PIP_j^l	Posterior Inclusion Probability for (j, l)-th regression
μ_j^l	Variational approximation to posterior regression coefficient means for (j, l)-th regression
$(s_j^l)^2$	Variational approximation to posterior regression coefficient variances for (j, l)-th regression
α_j^l	Variational approximation to posterior inclusion probabilities for (j, l)-th regression
ϕ_j^l	Variational parameters for (j, l)-th regression
π	Prior probability of inclusion
σ^2	Prior residual variance
σ_β^2	Prior slab variance
$\texttt{ssq}_\texttt{j}$	Prior residual variance candidates for j-th regressions
$\texttt{sbsq}_\texttt{j}$	Prior slab variance candidates for j-th regressions
$\texttt{pip}_\texttt{j}$	Prior inclusion probability candidates for j-th regressions
Θ_j	Hyperparameter candidates for j-th regressions
ω_j^l	Model averaging weights for (j, l)-th regression

B MODEL DETAILS

B.1 Full ELBO and Variational Updates

Recall from Equation (9) that for $l \in 1, 2, \ldots, n$ and $j \in 1, 2, \ldots, p$, the ELBO for the regression with X_j fixed as the response and weighted with respect to observation l is a function of the variational parameters $\phi_j^l = (\mu_j^l, (s_j^l)^2, \alpha_j^l) \in$

$\mathbb{R}^{p-1\times 3}$ and is given by

$$\text{ELBO}(\phi_j^l) = \mathbb{E}_q \log p_j^0\left(\beta_j^l, \gamma_j^l | \sigma^2, \sigma_\beta^2, \pi\right) + \mathbb{E}_q \log L_j^l\left(X_j | \beta_j^l, \mathbf{X}_{-j}, \mathbf{Z}, \tau, \sigma^2\right) - \mathbb{E}_q \log q(\beta_j^l, \gamma_j^l; \phi_j^l). \tag{24}$$

Without loss of generality, assume that $j = p$. Let $w_{i,l}$ denote the similarity weight for observation i with respect to observation l, and $x_{i,k}$ denote the i,k entry of $\mathbf{X} \in \mathbb{R}^{n\times p}$. We denote the prior probability of inclusion, which we have thus far referred to as π, as ξ, and use the usual numerical definition for π. Then, for the hyperparameters $\tau, \xi, \sigma^2, \sigma_\beta^2 \in \mathbb{R}$, each of the terms in the ELBO is given by

$$\mathbb{E}_q \log p_j^0\left(\beta_j^l, \gamma_j^l | \sigma^2, \sigma_\beta^2, \xi\right) = \sum_{k=1}^{p-1} -\frac{\alpha_{j,k}^l}{2}\log\left(2\pi\sigma^2\sigma_\beta^2\right) - \frac{\alpha_{j,k}^l\left[(\mu_{j,k}^l)^2 + (s_{j,k}^l)^2\right]}{2\sigma^2\sigma_\beta^2} + \alpha_{j,k}^l \log(\xi) + (1-\alpha_{j,k}^l)\log(1-\xi), \tag{25}$$

$$\mathbb{E}_q \log L_j^l\left(X_j | \beta_j^l, \mathbf{X}_{-j}, \mathbf{Z}, \tau, \sigma^2\right) = -\frac{n}{2}\log\left(2\pi\sigma^2\right) + \frac{1}{2}\sum_{i=1}^n \log(w_{i,l}) - \frac{1}{2\sigma^2}\sum_{i=1}^n w_{i,l}\left(\left[x_{i,p} - \sum_{k=1}^{p-1} x_{i,k}\alpha_{j,k}^l\mu_{j,k}^l\right]^2 + \sum_{k=1}^{p-1} x_{i,k}^2\left[\alpha_{j,k}^l((\mu_{j,k}^l)^2 + (s_{j,k}^l)^2) - (\alpha_{j,k}^l\mu_{j,k}^l)^2\right]\right), \tag{26}$$

$$\mathbb{E}_q \log q(\beta_j^l, \gamma_j^l; \phi_j^l) = \sum_{k=1}^{p-1} -\frac{\alpha_{j,k}^l}{2}\log(2\pi(s_{j,k}^l)^2) - \frac{\alpha_{j,k}^l}{2} + \alpha_{j,k}^l\log(\alpha_{j,k}^l) + (1-\alpha_{j,k}^l)\log(1-\alpha_{j,k}^l). \tag{27}$$

Taking the derivatives of the ELBO with respect to the k-th coordinate of the variational parameters $\mu_{j,k}^l, \alpha_{j,k}^l, (s_{j,k}^l)^2$ gives the coordinate ascent variational updates to be

$$(s_{j,k}^l)^2 = \sigma^2\left(\frac{1}{\sigma_\beta^2} + \sum_{i=1}^n x_{i,k}^2 w_{i,l}\right)^{-1}, \tag{28}$$

$$\log\left(\frac{\alpha_{j,k}^l}{1-\alpha_{j,k}^l}\right) = \log\left(\frac{\xi}{1-\xi}\right) + \frac{(\mu_{j,k}^l)^2}{2(s_{j,k}^l)^2} + \log\frac{s_{j,k}^l}{\sigma\sigma_\beta}, \tag{29}$$

$$\mu_{j,k}^l = \frac{(s_{j,k}^l)^2}{\sigma^2}\sum_{i=1}^n\left\{w_{i,l}x_{i,k}\left(x_{i,p} - \sum_{m=1}^{p-1} x_{i,m}\mu_{j,m}^l\alpha_{j,m}^l\mathbb{1}(m\neq k)\right)\right\}. \tag{30}$$

B.2 2-step Approach for Density Estimation

Here, we describe the 2-step approach for density estimation used to select the bandwidth hyperparameters τ as discussed in Section 2.3.1. Let $z_1, z_2, \ldots, z_n \in \mathbb{R}^q$, and consider the k-th coordinate of this data, $z_{j,k}$, with sample variance s_k^2. We choose the initial bandwidth estimate σ_k^2 using Silverman's rule of thumb [Silverman 2017] as

$$\sigma_k^2 = \left(0.9\min\left(s_k, \frac{IQR(z_{1,k}, z_{2,k}, \ldots, z_{n,k})}{1.35}\right)n^{-\frac{1}{5}}\right)^2. \tag{31}$$

Next, we let $\psi^0_{j,k}$ be the density of a Gaussian centered at $z_{j,k}$ with variance σ^2_k. Then, the estimate of the density for the k-th coordinate of the data is given by

$$\phi^0_k(x) = n^{-1} \sum_{j=1}^{n} \psi^0_{j,k}(x). \tag{32}$$

Repeating this for each coordinate $k = 1, 2, \ldots, q$, we will obtain a collection of q Gaussian mixture densities $\phi^0_1, \phi^0_1, \ldots, \phi^0_q$. To update this collection to a single multivariate density, first let $\mathcal{H}$ be the harmonic mean of the σ^2_k, defined as

$$\mathcal{H} = n \left(\sum_{k=1}^{q} \frac{1}{\sigma^2_k} \right)^{-1}. \tag{33}$$

Then, let Ψ_j be a multivariate Gaussian centered at z_j with variance $\tau_j \mathbb{I}^{q \times q} \in \mathbb{R}^{q \times q}$, where $\tau_j \in \mathbb{R}$ is defined as

$$\tau_j = \frac{\mathcal{H}}{\sqrt{\prod_{k=1}^{q} \phi^0_k(z_{j,k})}}. \tag{34}$$

The final KDE to the multivariate density of the data is given by

$$\Phi(x) = n^{-1} \sum_{j=1}^{n} \Psi_j(x). \tag{35}$$

However, for our purposes, we simply use τ_j as the bandwidth for the j-th observation.

C BASELINE DETAILS

C.1 `mgm` hyperparameters

As discussed in Section 3.1, we select the bandwidth hyperparameter for the `tvmgm` function following this example provided by the authors of the package:
https://github.com/jmbh/mgmDocumentation/blob/22af97a621a1a240b68d5731886d542fb588915a/examples_tvmgm.R#L31
In this example, 5-fold cross-validation is applied to time-varying Gaussian data with $n = 67$ and $p = 150$ to select the bandwidth out of the bandwidth candidates given by $0.1, 0.2, 0.3, 0.4$.

C.2 `loggle` hyperparameters

Hyperparameters for `loggle` were selected using the cross-validation functionality included in the package. For the bandwidth and shrinkage hyperparameters, we use the 5 and 11 candidate values which are defaults specified by the package, respectively. For the neighborhood width hyperparameter, we use 9 of the 12 values specified by default, removing the 3 largest values to reduce runtime.

As discussed in Section 3.3.1, we were only able to use 15 workers for parallelization due to excess memory usage. We additionally encountered prohibitively long runtimes for some trials when using `loggle` to estimate graphs for observations with times near the boundaries. We therefore set the graphs estimated for the observations with the 10 smallest and 10 largest times to the graphs estimated for the 11th smallest and largest times, respectively.

C.3 Dimensionality Reduction Sorting Algorithm

Here, we describe the sorting algorithm for reducing the dimensionality of the extraneous covariate that we discuss in Section 3.1. Let $z_1, z_2, \ldots, z_n \in \mathbb{R}^q$ which we will sort to $z_{(1)}, z_{(2)}, \ldots, z_{(n)}$. First, we let $z_{(1)} = z_1$. Then, for

Table 2. Comparison of hyperparameter specification strategies in the $q = 1$, piece-wise linear settings across 50 trials, presented as "*mean* (*standard error*)".

p	HP Method	Sensitivity(%)	Specificity(%)	Time (seconds)
10	`grid_search`	90.65(0.82)	99.24(0.12)	**4.75**(0.03)
	`hybrid`	89.96(0.79)	99.39(0.11)	5.46(0.06)
	`model_average`	88.97(0.87)	99.01(0.11)	20.93(0.53)
25	`grid_search`	87.32(0.96)	99.70(0.03)	**12.31**(0.05)
	`hybrid`	86.49(0.99)	99.77(0.03)	14.26(0.17)
	`model_average`	85.73(0.99)	99.44(0.03)	62.92(1.55)
50	`grid_search`	84.35(0.98)	99.71(0.03)	**42.20**(0.42)
	`hybrid`	83.99(0.95)	99.82(0.01)	44.77(0.33)
	`model_average`	84.33(0.84)	99.60(0.01)	166.94(2.24)
100	`grid_search`	80.81(0.96)	99.68(0.02)	**221.77**(2.44)
	`hybrid`	80.91(0.90)	99.86(0.01)	236.97(2.02)
	`model_average`	83.16(0.83)	99.70(0.01)	961.26(9.58)

$t \in \{2, 3, \ldots, n\}$, we define $\mathcal{S}^t = \{z_1, z_2, \ldots, z_n\} \setminus \{z_{(1)}, z_{(2)}, \ldots, z_{(t-1)}\}$ as the covariates that have not yet been sorted and set

$$z_{(t)} = \arg\min_{z \in \mathcal{S}^t} \|z_{(t-1)} - z\|. \tag{36}$$

This is a greedy algorithm for identifying a Hamiltonian path through the extraneous covariates. We then define the time index t_k for observation k as the position to which z_k was sorted. That is, $t_k = t$ if, and only if, $z_k = z_{(t)}$. The time index T is then given by $T = \{t_j\}_{j=1}^n$.

D EXTENDED EXPERIMENTS AND RESULTS

D.1 Numerical Results

In Tables 2 to 6, we present numerical results for Figures 4 to 8.

D.2 CPU Times

In Table 7, we present the CPU times for `covdepGE` in the $q = 1$, PWL settings and compare it to the real time with parallelism. With parallelism, 56 threads are used, meaning that up to 56 variables can be fixed as the response and regressed for all n observations at once. To obtain the CPU times, we repeat the 50 trials in the $q = 1$, PWL settings without parallelism.

D.3 Full Grid Search

As discussed in Section 3.3.1, `model_average` incurs additional overhead relative to the alternatives due its inability to perform a cursory initial sweep of Θ_j as `hybrid` and `grid_search`. To assess whether abbreviating this search has a substantial impact on performance, we repeat the experiments in Section 3.3.1, but set `max_iter_grid` = `max_iter` for `hybrid` and `grid_search` so that all methods perform CAVI for roughly the same number of iterations.

Table 3. Comparison of heterogeneous graphical modeling methods in the $q = 1$, PWL settings.

p	Method	Sensitivity(%)	Specificity(%)	Time (seconds)
10	covdepGE	**89.96**(0.79)	99.39(0.11)	**5.46**(0.06)
	JGL	79.59(1.17)	98.63(0.14)	26.41(2.19)
	loggle	85.45(0.67)	99.45(0.10)	175.72(1.05)
	mgm	79.69(1.34)	**99.88**(0.05)	760.40(3.02)
25	covdepGE	**86.49**(0.99)	99.77(0.03)	**14.26**(0.17)
	JGL	78.74(1.59)	99.71(0.03)	46.06(10.23)
	loggle	79.51(0.54)	99.93(0.01)	366.29(1.97)
	mgm	73.51(1.32)	**99.99**(0.00)	2042.37(5.93)
50	covdepGE	**83.99**(0.95)	99.82(0.01)	**44.77**(0.33)
	JGL	72.09(2.17)	99.94(0.01)	107.60(35.76)
	loggle	76.37(0.98)	99.97(0.00)	1111.65(3.51)
	mgm	66.90(1.35)	**100.00**(0.00)	4551.70(18.04)
100	covdepGE	**80.91**(0.90)	99.86(0.01)	236.97(2.02)
	JGL	70.18(2.25)	99.98(0.00)	**190.20**(40.55)
	loggle	74.93(0.83)	99.98(0.00)	5755.47(8.88)
	mgm	59.64(1.03)	**100.00**(0.00)	11112.32(36.89)

We present results for this experiment in Table 8. We observe that aside from the expected increase in runtime to be roughly equivalent with `model_average`, the effect on sensitivity and specificity relative to the results presented in Table 2 is negligible.

D.4 Sensitivity by Conditional Dependence Structure for $q = 1$ Settings

In Figure 9, we break down the sensitivity of the `covdepGE` estimates in the $q = 1$ PWL and NL settings by CDS. As discussed in Section 3.3.4, estimation of CDS 1 and CDS 2 is more challenging in the NL settings due to the domain of the covariate values for observations corresponding to these graphs being split over 2 disjoint intervals.

D.5 Sensitivity T-Test

In Tables 9 to 12, we present T-test results for testing the alternative hypothesis that the mean sensitivity of `covdepGE` is greater than that of the baseline method with the best sensitivity in each setting. We observe that in 13 of 16 settings, the mean sensitivity for `covdepGE` is significantly greater on the 0.001 level. In 15 of 16 settings, the mean for `covdepGE` is significantly greater on the 0.05 level.

D.6 Number of Unique Graphs Estimated

In Figure 10, we analyze the number of unique CDS estimated by `covdepGE` in each setting. As discussed in Section 3.3.3, despite the increased number of ground truth CDS, the number of unique graphs estimated in the $q = 4$ settings is lesser than in the other settings, which is likely due to the curse of dimensionality.

Table 4. Comparison of heterogeneous graphical modeling methods in the $q = 2$ settings.

p	Method	Sensitivity(%)	Specificity(%)	Time (seconds)
10	covdepGE	**91.03**(1.26)	**99.48**(0.08)	5.62(0.05)
	covdepGE_time	87.95(1.38)	98.78(0.10)	**5.42**(0.06)
	JGL	74.98(1.27)	98.46(0.19)	44.45(2.13)
	loggle	70.78(1.48)	98.92(0.12)	180.73(0.94)
	mgm	70.83(2.58)	99.40(0.07)	754.48(2.64)
25	covdepGE	**84.74**(1.73)	99.89(0.01)	14.41(0.16)
	covdepGE_time	80.78(1.88)	99.63(0.03)	**14.29**(0.18)
	JGL	62.00(2.13)	99.81(0.03)	62.43(3.89)
	loggle	55.77(1.60)	99.90(0.02)	363.05(1.93)
	mgm	56.90(2.18)	**99.96**(0.01)	2036.89(7.33)
50	covdepGE	**83.66**(1.68)	99.96(0.00)	**43.88**(0.31)
	covdepGE_time	78.20(1.72)	99.80(0.01)	45.60(0.36)
	JGL	55.55(2.24)	99.96(0.01)	127.38(10.66)
	loggle	48.77(1.31)	99.98(0.00)	1101.82(2.68)
	mgm	48.52(1.54)	**100.00**(0.00)	4522.41(13.59)
100	covdepGE	**80.80**(1.84)	99.98(0.00)	**225.97**(1.89)
	covdepGE_time	73.37(1.93)	99.85(0.01)	235.00(2.10)
	JGL	48.63(2.47)	99.99(0.00)	325.51(39.63)
	loggle	48.58(1.38)	99.98(0.00)	5636.29(13.33)
	mgm	45.56(1.06)	**100.00**(0.00)	11008.27(30.06)

D.7 Median/IQR-Aggregated Results

As discussed in Section 3.3.1, we observed the runtimes for JGL to be right-skewed due to some values of the hyper-parameters causing the algorithm to iterate for an abnormally long time in a subset of the trials. In Tables 13 to 16, we aggregate the results from the experiments performed in Sections 3.3.1 to 3.3.4 using median in place of mean and interquartile range in place of standard error.

D.8 Mclust Subgroup Counts

As discussed in Section 3.3.2, we observed that in the $q = 2, p = 100$ setting, JGL no longer offered a better runtime than covdepGE. We show here that this is likely a result of increased computation due to a greater number of subgroups being estimated by Mclust. Because JGL jointly estimates the precision structure in each of the subgroups, a greater number of subgroups results in longer runtimes.

In all trials, we choose the number of subgroups K by minimizing the BIC over $K \in \{2, 3, \ldots, 12\}$. In Figure 11, we visualize the optimal K in each setting. Here, we have summed across p since $\mathbf{Z}$, the variable partitioned by Mclust, is independent of p.

Table 5. Comparison of heterogeneous graphical modeling methods in the $q = 4$ settings.

p	Method	Sensitivity(%)	Specificity(%)	Time (seconds)
10	covdepGE	**88.70**(1.36)	98.00(0.11)	**4.26**(0.04)
	covdepGE_time	85.18(1.17)	97.38(0.13)	5.02(0.04)
	JGL	69.19(1.05)	98.09(0.21)	13.12(1.67)
	loggle	67.14(1.27)	98.01(0.20)	179.92(0.94)
	mgm	71.52(1.74)	**98.84**(0.10)	588.44(1.93)
25	covdepGE	**80.47**(2.13)	99.81(0.02)	**11.43**(0.10)
	covdepGE_time	79.24(1.49)	99.49(0.03)	13.99(0.11)
	JGL	63.81(1.99)	99.62(0.04)	15.32(2.26)
	loggle	51.65(1.09)	99.89(0.02)	365.84(2.02)
	mgm	59.16(1.53)	**99.93**(0.01)	1573.13(5.10)
50	covdepGE	**79.50**(1.75)	99.95(0.00)	32.37(0.29)
	covdepGE_time	78.07(1.40)	99.75(0.01)	42.13(0.27)
	JGL	59.29(2.68)	99.88(0.01)	**26.33**(7.08)
	loggle	48.03(0.85)	99.96(0.00)	1083.86(3.06)
	mgm	54.17(1.16)	**99.99**(0.00)	3465.36(8.17)
100	covdepGE	70.81(2.55)	99.99(0.00)	128.96(0.76)
	covdepGE_time	**72.20**(1.48)	99.84(0.00)	194.82(0.63)
	JGL	55.35(2.54)	99.97(0.00)	**128.91**(44.65)
	loggle	47.89(1.20)	99.98(0.00)	5470.54(8.44)
	mgm	49.91(1.42)	**100.00**(0.00)	8468.45(25.82)

Table 6. Comparison of heterogeneous graphical modeling methods in the $q = 1$, non-linear settings.

p	Method	Sensitivity(%)	Specificity(%)	Time (seconds)
10	covdepGE	**80.90**(1.50)	99.27(0.10)	**4.99**(0.05)
	JGL	75.11(2.00)	98.84(0.15)	22.85(2.61)
	loggle	73.83(1.21)	99.29(0.11)	181.66(1.07)
	mgm	61.95(1.63)	**99.79**(0.05)	781.57(2.11)
25	covdepGE	**77.05**(1.55)	99.75(0.02)	**13.54**(0.14)
	JGL	73.55(2.20)	99.69(0.03)	37.68(3.44)
	loggle	61.26(0.96)	99.95(0.01)	366.88(1.76)
	mgm	51.31(1.35)	**99.99**(0.00)	2080.77(6.64)
50	covdepGE	**72.05**(1.51)	99.80(0.01)	**62.61**(0.38)
	JGL	66.80(2.56)	99.94(0.01)	86.69(20.64)
	loggle	56.29(1.17)	99.97(0.00)	1105.18(3.68)
	mgm	47.07(1.16)	**100.00**(0.00)	4626.41(13.65)
100	covdepGE	**69.31**(1.42)	99.84(0.00)	**277.79**(2.21)
	JGL	56.09(2.94)	99.99(0.00)	449.65(226.58)
	loggle	53.68(1.32)	99.98(0.00)	5579.63(8.01)
	mgm	44.17(0.68)	**100.00**(0.00)	11348.97(40.22)

Table 7. CPU versus real time. Here we compare the runtime of covdepGE with and without parallelism in the columns Parallel Time and CPU Time, respectively. We provide the ratio of CPU time to parallel time in the column Ratio.

p	Parallel Time (seconds)	Sequential Time (seconds)	Ratio
10	5.46(0.06)	43.23(0.23)	7.95 (0.09)
25	14.26(0.17)	289.11(1.07)	20.39 (0.22)
50	44.77(0.33)	1252.96(2.92)	28.06 (0.20)
100	236.97(2.02)	5768.80(18.36)	24.41 (0.17)

Table 8. Comparison of hyperparameter specification strategies in the $q = 1$, piece-wise linear settings. Here, we set `max_iter_grid = max_iter` so that all strategies perform roughly the same number of CAVI iterations.

p	HP Method	Sensitivity(%)	Specificity(%)	Time (seconds)
10	`grid_search`	**90.67**(0.82)	99.23(0.11)	**18.61**(0.52)
	`hybrid`	89.93(0.78)	**99.36**(0.12)	18.78(0.53)
	`model_average`	88.97(0.87)	99.01(0.11)	20.93(0.53)
25	`grid_search`	**87.42**(0.96)	99.69(0.03)	**58.84**(1.53)
	`hybrid`	86.61(0.97)	**99.76**(0.03)	60.40(1.65)
	`model_average`	85.73(0.99)	99.44(0.03)	62.92(1.55)
50	`grid_search`	**84.37**(0.97)	99.75(0.01)	171.72(2.72)
	`hybrid`	83.90(0.96)	**99.81**(0.01)	172.01(2.46)
	`model_average`	84.33(0.84)	99.60(0.01)	**166.94**(2.24)
100	`grid_search`	80.51(0.95)	99.80(0.01)	981.12(14.10)
	`hybrid`	80.66(0.92)	**99.86**(0.01)	976.91(12.07)
	`model_average`	**83.16**(0.83)	99.70(0.01)	**961.26**(9.58)

Table 9. T-Test analysis for the $q = 1$, piece-wise linear settings. Here we present the results of T-Tests with the alternative hypothesis that the mean sensitivity of covdepGE is greater than the mean sensitivity of the baseline method with the best sensitivity in the $q = 1$, PWL settings.

p	covdepGE	Baseline	Baseline Name	Standard Error	T-statistic	DF	p-value
10	**89.96**(0.79)	85.45(0.67)	`loggle`	1.03	4.36	95.30	< 0.001
25	**86.49**(0.99)	79.51(0.54)	`loggle`	1.13	6.19	75.99	< 0.001
50	**83.99**(0.95)	76.37(0.98)	`loggle`	1.37	5.56	97.90	< 0.001
100	**80.91**(0.90)	74.93(0.83)	`loggle`	1.23	4.86	97.41	< 0.001

Table 10. T-Test analysis for the $q = 2$ settings. Here we present the results of T-Tests with the alternative hypothesis that the mean sensitivity of covdepGE is greater than the mean sensitivity of the baseline method with the best sensitivity in the $q = 2$ settings.

p	covdepGE	Baseline	Baseline Name	Standard Error	T-statistic	DF	p-value
10	**91.03**(1.26)	74.98(1.27)	JGL	1.79	8.98	98.00	< 0.001
25	**84.74**(1.73)	62.00(2.13)	JGL	2.75	8.28	94.12	< 0.001
50	**83.66**(1.68)	55.55(2.24)	JGL	2.80	10.05	90.90	< 0.001
100	**80.80**(1.84)	48.63(2.47)	JGL	3.08	10.45	90.61	< 0.001

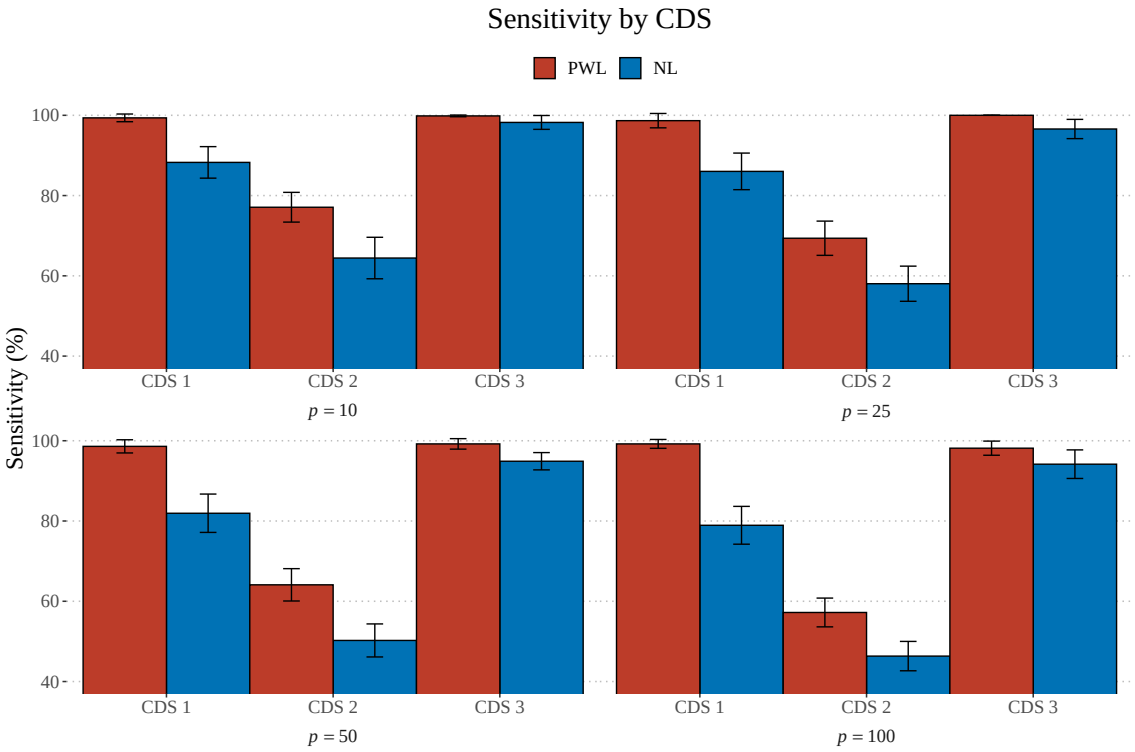

Fig. 9. Sensitivity by Conditional Dependence Structure (CDS) for $q = 1$ settings. Broken down by Piece-Wise Linear (PWL) and Non-Linear (NL). Error bars are 2 times the standard error.

Table 11. T-Test analysis for the $q = 4$ settings. Here we present the results of T-Tests with the alternative hypothesis that the mean sensitivity of covdepGE is greater than the mean sensitivity of the baseline method with the best sensitivity in the $q = 4$ settings.

p	covdepGE	Baseline	Baseline Name	Standard Error	T-statistic	DF	p-value
10	**88.70**(1.36)	71.52(1.74)	mgm	2.21	7.78	92.55	< 0.001
25	**80.47**(2.13)	63.81(1.99)	JGL	2.91	5.72	97.59	< 0.001
50	**79.50**(1.75)	59.29(2.68)	JGL	3.20	6.32	84.34	< 0.001
100	70.81(2.55)	55.35(2.54)	JGL	3.60	4.29	98.00	< 0.001

Table 12. T-Test analysis for the $q = 1$, non-linear settings. Here we present the results of T-Tests with the alternative hypothesis that the mean sensitivity of covdepGE is greater than the mean sensitivity of the baseline method with the best sensitivity in the $q = 1$, NL settings.

p	covdepGE	Baseline	Baseline Name	Standard Error	T-statistic	DF	p-value
10	**80.90**(1.50)	75.11(2.00)	JGL	2.50	2.31	90.99	0.012
25	**77.05**(1.55)	73.55(2.20)	JGL	2.69	1.30	87.96	0.099
50	**72.05**(1.51)	66.80(2.56)	JGL	2.97	1.77	79.31	0.041
100	**69.31**(1.42)	56.09(2.94)	JGL	3.27	4.05	70.53	< 0.001

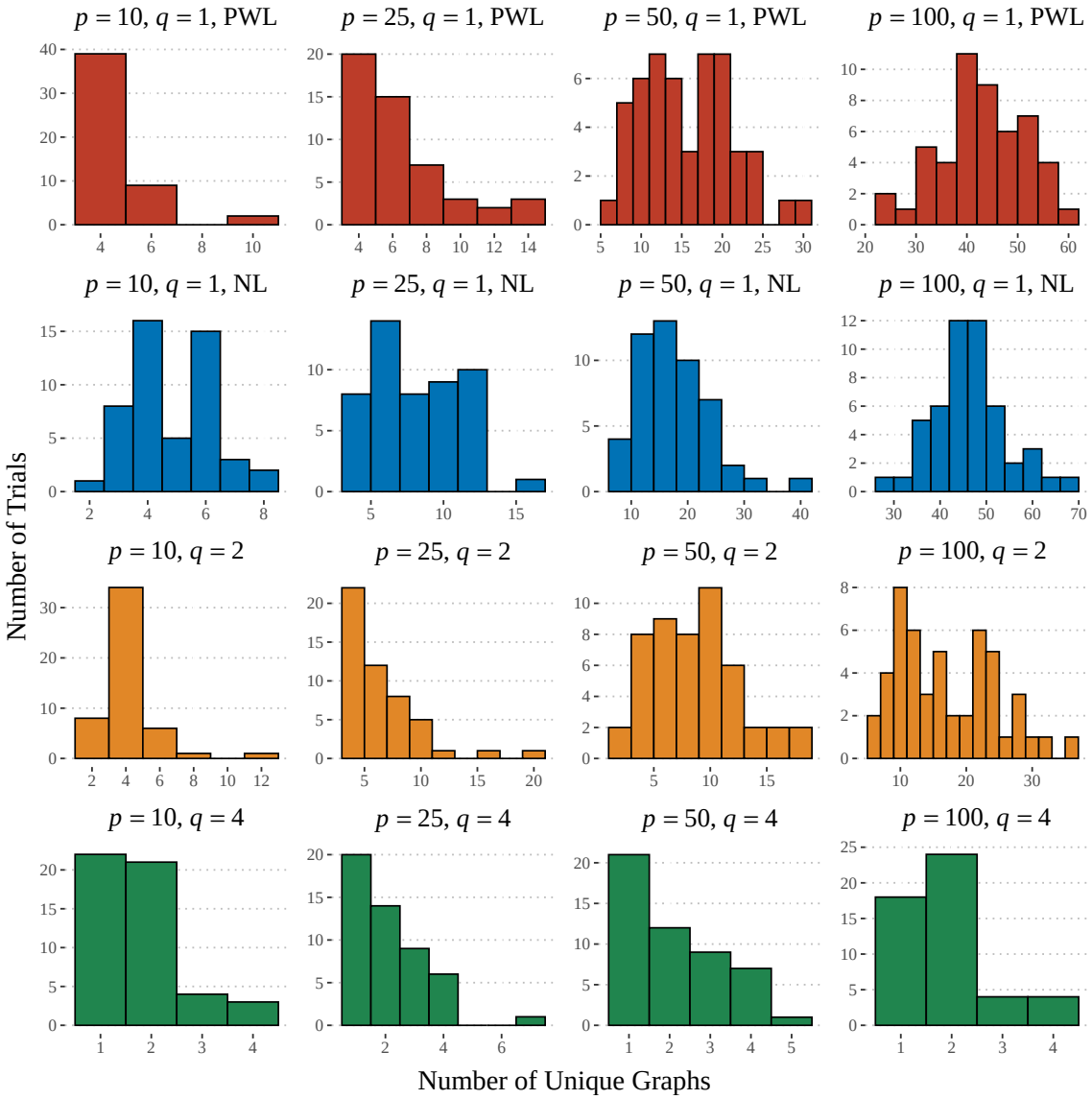

Fig. 10. Unique graphs estimated in each setting. In the $q = 1$ settings, the true number of unique ground truth graphs is 3, while in the $q = 2$ settings, it is 4. The number of unique ground truth graphs across the 200 trials in the $q = 4$ settings varies due to the sampling procedures, however, the average and standard deviation are 15.53 and 0.67, respectively.

Table 13. Comparison of heterogeneous graphical modeling methods in the $q = 1$, piece-wise linear settings, presented as "*median* (*IQR*)".

p	Method	Sensitivity(%)	Specificity(%)	Time (seconds)
10	covdepGE	**91.71**(9.10)	99.56(0.84)	**5.50**(0.54)
	JGL	85.71(13.05)	98.72(1.57)	18.86(24.68)
	loggle	86.10(7.67)	99.62(0.65)	176.10(7.82)
	mgm	78.19(6.05)	**100.00**(0.02)	759.11(15.14)
25	covdepGE	83.81(11.52)	99.82(0.26)	**14.19**(1.59)
	JGL	**84.38**(13.38)	99.77(0.36)	23.34(39.35)
	loggle	79.62(4.67)	99.96(0.12)	367.53(11.42)
	mgm	73.71(9.90)	**100.00**(0.00)	2033.92(40.81)
50	covdepGE	**82.48**(9.43)	99.81(0.09)	44.63(2.42)
	JGL	72.19(22.43)	99.95(0.06)	**36.41**(83.18)
	loggle	77.52(6.05)	99.98(0.03)	1117.10(34.46)
	mgm	68.38(14.86)	**100.00**(0.00)	4544.80(187.94)
100	covdepGE	**80.76**(7.00)	99.86(0.05)	235.50(15.63)
	JGL	71.81(22.90)	99.98(0.02)	**62.87**(212.55)
	loggle	77.05(5.90)	99.98(0.01)	5754.67(89.49)
	mgm	60.29(10.14)	**100.00**(0.00)	11038.74(336.10)

Table 14. Comparison of heterogeneous graphical modeling methods in the $q = 2$ settings, presented as "*median* (*IQR*)".

p	Method	Sensitivity(%)	Specificity(%)	Time (seconds)
10	covdepGE	**93.71**(10.48)	**99.70**(0.76)	5.59(0.46)
	covdepGE_time	90.48(11.00)	98.89(0.96)	**5.47**(0.60)
	JGL	74.95(12.19)	98.63(1.89)	43.81(8.07)
	loggle	72.29(12.57)	99.03(1.02)	180.52(8.78)
	mgm	70.29(33.10)	99.56(0.71)	753.17(18.46)
25	covdepGE	**88.10**(21.76)	99.91(0.14)	**14.38**(1.71)
	covdepGE_time	82.95(19.57)	99.68(0.29)	14.56(2.00)
	JGL	65.05(18.24)	99.86(0.25)	61.67(17.03)
	loggle	57.05(17.33)	99.94(0.13)	365.35(14.36)
	mgm	54.19(17.95)	**99.98**(0.08)	2027.88(40.53)
50	covdepGE	**84.86**(12.38)	99.96(0.03)	**43.72**(2.33)
	covdepGE_time	79.43(17.57)	99.80(0.09)	45.69(2.62)
	JGL	55.43(21.62)	99.98(0.04)	111.58(17.31)
	loggle	50.38(13.24)	99.98(0.02)	1103.93(27.33)
	mgm	42.86(9.48)	**100.00**(0.01)	4514.89(130.81)
100	covdepGE	**82.38**(17.76)	99.98(0.01)	**225.31**(20.42)
	covdepGE_time	69.62(23.52)	99.85(0.05)	236.93(22.07)
	JGL	51.52(17.57)	99.99(0.01)	297.56(59.60)
	loggle	46.67(13.62)	99.98(0.01)	5640.91(134.13)
	mgm	42.86(0.43)	**100.00**(0.00)	11003.13(288.84)

Table 15. Comparison of heterogeneous graphical modeling methods in the $q = 4$ settings, presented as "*median* (*IQR*)".

p	Method	Sensitivity(%)	Specificity(%)	Time (seconds)
10	covdepGE	**88.20**(15.30)	98.06(1.17)	**4.26**(0.30)
	covdepGE_time	85.39(13.75)	97.51(1.07)	5.02(0.37)
	JGL	69.16(6.44)	98.53(1.69)	8.40(2.29)
	loggle	66.22(11.21)	98.27(1.66)	180.68(6.64)
	mgm	70.20(18.04)	**99.09**(1.27)	587.09(15.50)
25	covdepGE	**85.21**(16.96)	99.81(0.19)	11.38(0.79)
	covdepGE_time	81.38(13.12)	99.54(0.27)	13.93(1.16)
	JGL	66.01(18.54)	99.72(0.41)	**8.70**(2.70)
	loggle	51.96(11.37)	99.93(0.08)	367.20(15.23)
	mgm	59.60(13.11)	**99.94**(0.07)	1568.32(38.59)
50	covdepGE	**81.90**(14.01)	99.96(0.04)	32.08(2.69)
	covdepGE_time	79.07(12.46)	99.77(0.13)	41.83(2.41)
	JGL	63.81(11.22)	99.90(0.13)	**9.78**(6.82)
	loggle	47.68(7.71)	99.97(0.04)	1085.58(28.69)
	mgm	53.88(10.22)	**99.99**(0.01)	3463.82(67.51)
100	covdepGE	71.55(29.52)	99.99(0.01)	128.48(7.69)
	covdepGE_time	**71.76**(17.59)	99.85(0.04)	194.73(7.17)
	JGL	59.69(24.15)	99.98(0.03)	**13.49**(91.86)
	loggle	46.05(12.20)	99.98(0.01)	5471.85(80.97)
	mgm	44.44(15.77)	**100.00**(0.00)	8429.76(268.68)

Table 16. Comparison of heterogeneous graphical modeling methods in the $q = 1$, non-linear settings, presented as "*median* (*IQR*)".

p	Method	Sensitivity(%)	Specificity(%)	Time (seconds)
10	covdepGE	79.81(12.86)	99.39(0.59)	**5.06**(0.62)
	JGL	**83.71**(14.76)	99.22(1.62)	18.00(8.61)
	loggle	74.67(11.52)	99.47(0.69)	181.50(6.74)
	mgm	63.81(11.29)	**100.00**(0.28)	779.84(9.24)
25	covdepGE	79.33(15.19)	99.78(0.21)	**13.66**(1.60)
	JGL	**80.95**(15.48)	99.76(0.30)	28.49(38.15)
	loggle	61.62(9.10)	99.96(0.06)	366.59(12.17)
	mgm	48.95(17.43)	**100.00**(0.02)	2073.21(54.71)
50	covdepGE	**71.52**(13.00)	99.82(0.10)	63.07(4.10)
	JGL	70.76(27.52)	99.97(0.09)	**33.07**(70.12)
	loggle	56.76(9.90)	99.97(0.02)	1112.05(28.70)
	mgm	42.86(2.14)	**100.00**(0.00)	4605.76(65.66)
100	covdepGE	**69.71**(11.43)	99.84(0.04)	276.46(18.02)
	JGL	57.05(28.48)	99.99(0.02)	**59.84**(88.39)
	loggle	54.29(10.19)	99.98(0.01)	5575.12(65.14)
	mgm	42.86(0.00)	**100.00**(0.00)	11325.53(253.06)

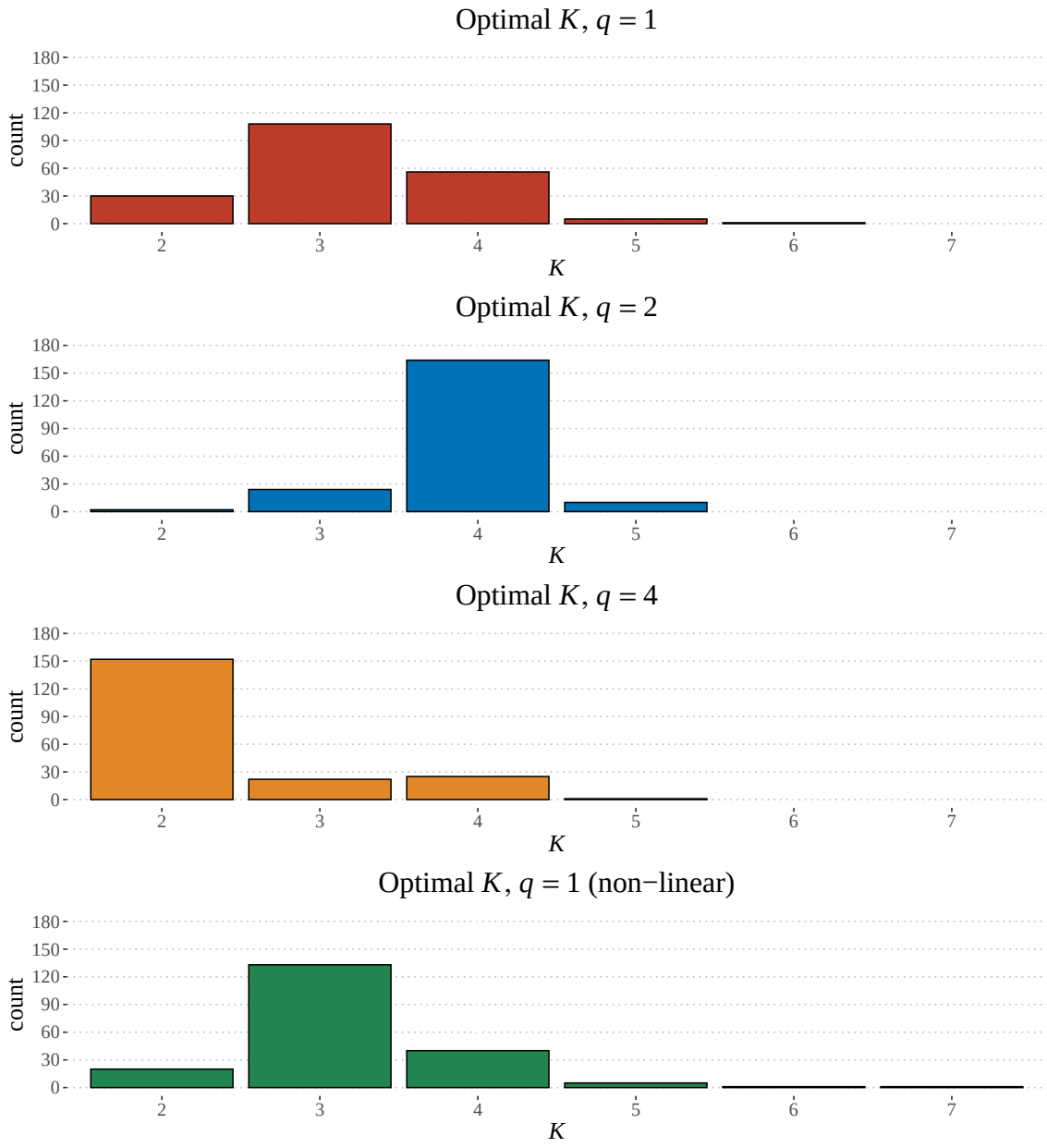

Fig. 11. Optimal number of subgroups K for partitioning the covariate $\mathbf{Z}$. Because $\mathbf{Z}$ is independent of p, we aggregated the visualized counts across trials and p. *First row*: $q = 1$, piece-wise linear settings. *Second row*: $q = 2$ settings.*Third row*: $q = 4$ settings. *Fourth row*: $q = 1$, non-linear settings.