Multiplex Regression in Security Systems

Jose Rodriguez-Acosta*
Department of Statistics, Texas A&M University
and
Sharmistha Guha
Department of Statistics, Texas A&M University
and
Samuel Gailliot
Department of Statistics, Texas A&M University
and
Adam Williams
Sandia National Laboratories

April 22, 2024

Abstract

Across a range of scientific fields, including biology, neuroimaging, and security systems engineering, there is an increasing inclination towards adopting a multilayer network approach to address significant scientific inquiries. Multilayer networks illustrate relationships among entities within various contexts, with each context being symbolized by a distinct layer. This article introduces a novel regression framework with a multilayer network as the predictor, and a continuous outcome variable. In this framework, the structure of the multilayer network is such that each layer corresponds to an undirected network. These undirected networks are typically depicted as symmetric matrices, with nodes or actors specified along both rows and columns, and the values within the matrix cells representing the association between the corresponding nodes. Existing regression methods with multilayer network predictors often do not effectively leverage information both within and across network layers. This can result in subpar inference and predictive accuracy, particularly when working with limited sample sizes. To address this constraint, our method adopts low-rank models associated with the coefficients for each layer of the network predictor, sharing parameters specific to each layer to efficiently capture the complex relationships and dependencies among nodes across multiple layers. The proposed framework identifies nodes and edges influentially related to the response within each layer of the network, as well as offers efficient posterior computation, inference, and prediction. We utilize

*The authors gratefully acknowledge *please remember to list all relevant funding sources in the unblinded version*

our approach to analyze emulator data obtained from Sandia National Labs, comprising multi-layer security networks crafted to emulate the security system network of a high-consequence facility (HCF) within the confines of permissions granted. Notably, our approach presents the *first* principled model-based statistical study of multilayer security architecture in HCFs, allowing for accurate prediction of time until detection of an external threat, as well as the statistical importance of nodes in the security networks. The empirical findings illustrate the advantages of our approach compared to prevailing high-dimensional regression methods, tensor regression techniques, and convolutional neural networks, with respect to both inference and predictive performance.

Keywords: Bayesian inference; Low-rank model; Multilayer network; Spike-and-slab prior.

1 Introduction

The growing complexities witnessed in contemporary security operations for safeguarding *high consequence facilities* stem from a variety of evolving factors in today's context. High-consequence facilities (HCFs) are defined as those whose incapacitation would have a devastating impact on national security, economic prosperity, and/or public health. However, current security frameworks are notably linear and encounter difficulties in comprehensively addressing the intricate interplay among different security layers like (1) security technology, (2) infrastructure, (3) cybersecurity, and (4) human/organizational elements. Therefore, it is imperative to develop sophisticated security models that consider these interconnections. Security scientists, especially those affiliated with the Sandia National Laboratories, are currently prioritizing fundamental interactions between various security domains to design next-generation security networks. These efforts are aimed at developing adaptable and resilient security systems that are well-suited for future needs, emphasizing *security orchestration* [Williams et al., 2020]. Security orchestration is the notion that disparate security layers can be more effective with enhanced coordination of interactions among different layers. Modern security research posits that multilayer network models are uniquely capable of capturing complex interactions between and within security layers effectively [Williams et al., 2021, Dove et al., 2023]. Nevertheless, there is a deficiency in principled statistical/data-driven understanding of the components of multilayer network security systems within the field of security research. In this article, we introduce an innovative Bayesian multi-layer network regression (BMNR) model with a scalar outcome variable and a multilayer network as a predictor in the realm of security orchestration, with its application on security orchestration laid out in Section 5. Although our work is rooted in security networks, its applicability extends beyond this domain, making it a versatile and broadly applicable approach in multi-layer network applications including neuro-imaging and genomics data.

In the existing literature on multilayer networks, they are often treated as random variables rather than being utilized as predictors within a regression framework. This literature concentrates on establishing suitable dependencies among edges and among various relationships that define the multiple layers [Gollini and Murphy, 2016, Han et al., 2015, Heaney, 2014]. These advancements have extended the applicability of exponential random graph models [Holland and Leinhardt, 1981, Frank and Strauss, 1986] and latent variable models [Nowicki and Snijders, 2001, Hoff et al., 2002, Airoldi et al., 2008], which were originally designed for single networks, to facilitate inference within multilayer frameworks. These frameworks have undergone further extensions to handle time-varying or dynamic multilayer networks without [Durante et al., 2017, Snijders et al., 2013, Hoff, 2015] or with covariates associated with the network nodes [Contisciani et al., 2020, Zhang et al., 2022, Xu et al., 2023]. There is also related literature on deep graph representation through graph neural networks (GNN), where the main neural architectures include graph convolutional networks (GCNs; Kipf and Welling [2016]), graph attention networks (GANs; Veličković et al. [2017]), graph isomorphism networks (GINs; Xu et al. [2018]), and other variants [Hamilton, 2020]. While this literature mainly focuses on a single network, there are recent and ongoing efforts to extend them to multilayer networks [Ma et al., 2019].

However, the existing methods for multilayer networks are inherently unsupervised and do not align with the inferential objectives specific to our context. Our primary focus is on generating predictive inferences regarding the outcome, estimating crucial network nodes in each layer with significant associations to the outcome, and understanding the regression impact of the network within each layer on the outcome. The proposed regression framework with a multilayer network predictor effectively addresses these inferential objectives. In our framework, the network coefficients for each layer are represented as the sum of two components: one shared across all layers and another specific to each layer. Both components utilize low-rank structures to capture how interactions between various pairs of

network nodes in each layer influence the outcome. Specifically, the low-rank structure for each component expresses each edge coefficient as a function of latent effects corresponding to the nodes connected to that particular edge. The latent effects for the nodes used to construct the first component are shared across layers, whereas those utilized for the second component are specific to each individual layer. We adopt a variable selection framework on latent effects specific to each layer to draw inference on nodes and edges in each layer significantly related to the outcome. The proposed structure accomplishes parsimony, ensures accurate predictive inference, facilitates inference on network nodes and edges related to the outcome, and provides well-calibrated uncertainties for inference and prediction. As a mode of inference, a Bayesian framework is adopted due to its ability to naturally quantify uncertainty in inference and prediction, particularly in the identification of significant nodes and edges. This is particularly significant when dealing with a moderate sample size, especially when the number of network edges far exceeds the sample size.

1.1 Related Literature

Although the literature on regression models incorporating a multilayer network predictor is sparse, some previous studies have investigated regression frameworks with a scalar outcome and object predictors. However, existing methods are unable to address all of our inferential objectives simultaneously, as outlined in this section.

In regressions involving a single network predictor, commonly used methods often involve carefully constructing a few summary statistics from the network [Bullmore and Sporns, 2009] or the transformation of the network object into a high-dimensional set of edge weights [Craddock et al., 2009, Richiardi et al., 2011]. Subsequent inference is then drawn utilizing the developments in ordinary high-dimensional regression architecture [Tibshirani, 1996, Park and Casella, 2008, Carvalho et al., 2010] or neural network (NN) models [Polson and Ročková, 2018, Dinh and Ho, 2020]. These approaches can be

extended to handle multilayer networks in a straightforward manner. However, restructuring the multilayer network predictor using these methods may not adequately capture the effects of intricate interconnections between nodes within and across different layers on the outcome, potentially compromising the accuracy and interpretability of the regression model.

Recent advancements in the study of regression models with a scalar outcome and a single network predictor have shown promising potential by leveraging the structure of the network predictor. To this end, Relión et al. [2019] introduce a frequentist penalization framework, and Guha and Rodriguez [2021, 2023] introduce Bayesian network shrinkage priors to estimate the effects of network nodes and edges on the scalar outcome, exploiting the topology of the network predictor. The approach presented in Relión et al. [2019] demonstrates favorable performance in predicting the outcome and making inferences on model parameters. However, uncertainty quantification in this approach becomes challenging, as the state-of-the-art bootstrap methods cannot guarantee consistent uncertainty estimation for the group lasso-like penalties proposed in Relión et al. [2019]. The Bayesian frameworks introduced by Guha and Rodriguez [2021, 2023] exhibit precise prediction of the outcome and robust inference on model parameters, while also providing accurate uncertainty estimates, particularly in identifying influential network nodes and edges. However, both these approaches are tailored for a single network predictor. Making further advancements to handle interactions between nodes within and across layers for a multilayer network predictor poses nontrivial modeling challenges. Addressing these complexities is crucial for the development of a comprehensive regression framework capable of effectively handling multilayer network predictors.

An alternative approach involves stacking networks from different layers to create a tensor, which is then used to construct a regression framework with the scalar outcome and the tensor predictor. This approach can take advantage of recent developments in tensor

regression, consisting of both penalized optimization [Zhou et al., 2013, Fan et al., 2019], low-rank methods [He et al., 2018, Ahmed et al., 2020] and Bayesian multiway shrinkage literature [Guhaniyogi et al., 2017, Spencer et al., 2022]. However, these approaches do not explicitly consider the symmetry constraint in the individual layers of the multilayer network. Additionally, their primary focus is on prediction and identifying significant edges or interconnections, rather than specifically detecting important nodes in each layer that impact the response.

The rest of the article proceeds as follows. Section 2 provides a detailed description of the model development and prior formulation on model coefficients. Section 3 discusses posterior computation. The simulation studies in Section 4 and the analysis of security networks data in Section 5 showcase the effectiveness of the proposed approach compared to competing methods. Finally, Section 6 offers concluding remarks and future directions. Appendix A details the full conditional distributions for the model parameters used in model estimation through Markov Chain Monte Carlo (MCMC).

2 Bayesian Multilayer Network Regression

2.1 Model Description

In this section, we describe our proposed method, referred to as the Bayesian Multilayer Network Regression (BMNR) Model. For the i-th observation, $y_i \in \mathbb{R}$ is a continuous scalar response, and $\{\mathcal{N}_i^{(l)}\}_{l=1}^L$ is a multilayer network predictor with L layers. A L-layer network is defined as a sequence of L networks: $\{\mathcal{N}_i^{(l)}\}_{l=1}^L = \{(\mathcal{A}^{(l)}, \mathcal{E}^{(l)})\}_{l=1}^L$, with the lth network consisting of nodes $\mathcal{A}^{(l)}$ and edges $\mathcal{E}^{(l)} \subset \mathcal{A}^{(l)} \times \mathcal{A}^{(l)}$. We assume that node sets are same across different layers, i.e., $\mathcal{A}^{(l)} = \mathcal{A}^{(l')} = \mathcal{A}$, for any $1 \leq l \neq l' \leq L$. The common set of nodes is denoted by $\mathcal{A} = \{\mathcal{A}_1, \ldots, \mathcal{A}_V\}$, where $|\mathcal{A}| = V$ represents the number of nodes. The network-valued feature at the l-th layer $\mathcal{N}_i^{(l)}$ is represented by a $V \times V$

adjacency matrix $\boldsymbol{X}_i^{(l)} \in \mathbb{R}^{V\times V}$, where the entry at position (v, v') signifies the strength of association between nodes $\mathcal{A}_v$ and $\mathcal{A}_{v'}$, $v, v' = 1, ..., V$. This article specifically focuses on undirected networks with no self-relationship within each layer, meaning that the adjacency matrix $\mathbf{X}_i^{(l)}$ is symmetric and the diagonal entries of this matrix are zero. In what follows, we represent the L-layer network for observation i by the sequence of adjacency matrices $\boldsymbol{X}_i = \{\boldsymbol{X}_i^{(l)}\}_{l=1}^{L}$. For each observation i $(i = 1, \ldots, N)$, the proposed high-dimensional regression model representing the relationship between the response variable y_i and the multilayer network predictor is given by,

$$y_i = \mu + \sum_{l=1}^{L} \langle \boldsymbol{X}_i^{(l)} | \boldsymbol{B}^{(l)} \rangle / 2 + \epsilon_i, \;\; \epsilon_i \overset{i.i.d.}{\sim} N(0, \tau^2), \tag{1}$$

where $\boldsymbol{B}^{(l)}$ is the coefficient matrix of dimension $V \times V$ corresponding to the lth layer $\boldsymbol{X}_i^{(l)}$, and $\langle \boldsymbol{X}_i^{(l)} | \boldsymbol{B}^{(l)} \rangle = Trace(\boldsymbol{B}^{(l)T} \boldsymbol{X}_i^{(l)})$ denotes the Frobenius inner product between the matrices $\boldsymbol{X}_i^{(l)}$ and $\boldsymbol{B}^{(l)}$. The Frobenius inner product serves as an extension of the dot product, transitioning from vector spaces to matrix spaces, and it naturally represents the inner product in the space of matrices. In a manner akin to $\boldsymbol{X}_i^{(l)}$, we presume that the coefficient matrix $\boldsymbol{B}^{(l)}$ is symmetric and has zero entries along its diagonal. Equation (1) establishes a connection between networks at multiple layers and the outcome, allowing us to infer on influential nodes at each layer in predicting the outcome. To achieve this, prior distributions on the model coefficients are constructed jointly, as elaborated in the next section.

2.2 Low-Rank Structure on Coefficients and Prior Formulation

To encompass the network predictor information within each layer and capture the relationship between the multilayer network and the scalar outcome, while maintaining flexibility, we adopt a low-order spectral representation of the network coefficients within each layer. This representation includes shared latent effects across layers. Let $\beta_{v,v'}^{(l)}$ denote the (v, v')th entry of the coefficient matrix $\boldsymbol{B}^{(l)}$, $1 \leq v < v' \leq V$, with the symmetry condition implying

$\beta^{(l)}_{v,v'} = \beta^{(l)}_{v',v}$. The low-rank representation for the coefficient is given by,

$$\beta^{(l)}_{v,v'} = \bar{\boldsymbol{u}}_v^T \boldsymbol{\Lambda} \bar{\boldsymbol{u}}_{v'} + \boldsymbol{u}_v^{(l)T} \boldsymbol{\Lambda}^{(l)} \boldsymbol{u}^{(l)}_{v'}, \quad 1 \leq v < v' \leq V, \quad l = 1, ..., L, \tag{2}$$

where $\boldsymbol{u}_v^{(l)} \in \mathbb{R}^R$ is the R-dimensional coordinate in the latent space corresponding to node $\mathcal{A}_v$ specific to the l-th layer, and $\bar{\boldsymbol{u}}_v \in \mathbb{R}^R$ is the node coordinate of $\mathcal{A}_v$ shared across all layers. Henceforth, they are referred to as *layer-specific latent effects* and *shared latent effects*, respectively. Here $\boldsymbol{\Lambda}$ and $\boldsymbol{\Lambda}^{(l)}$ are $R \times R$ diagonal matrices with the r-th diagonal entries λ_r and $\lambda_r^{(l)}$, respectively. The diagonal elements $\lambda_r, \lambda_r^{(l)} \in \{-1, 0, 1\}$'s are introduced to assess the effect of the r-th dimension of shared and layer-specific latent effects, respectively, on the network coefficients. In particular, $\lambda_r^{(l)} = 0$ implies that the r-th dimension of the l-th layer-specific latent effects are not informative for any v. A similar conclusion applies to λ_r. The assumed structure diminishes the count of estimable parameters from $LV(V-1)/2$ to $(L+1)VR + R(L+1) = R(L+1)(V+1)$, with the typical condition that $R << V$.

The structure of $\beta^{(l)}_{v,v'}$ in (2) is designed to account for the interaction between nodes $\mathcal{A}_v$ and $\mathcal{A}_{v'}$. Note that $\bar{\boldsymbol{u}}_v^T \boldsymbol{\Lambda} \bar{\boldsymbol{u}}_{v'}$ captures the interaction between the nodes $\mathcal{A}_v$ and $\mathcal{A}_{v'}$ [Hoff et al., 2002] on the outcome shared across all layers, while $\boldsymbol{u}_v^{(l)T} \boldsymbol{\Lambda}^{(l)} \boldsymbol{u}^{(l)}_{v'}$ represents the layer-specific impact of this interaction, offering flexibility in modeling the effects from different network layers. Node interactions are characterized by the dot product of shared and layer-specific node coordinates in a latent space. This arrangement allows nodes with coordinates in the same direction to positively impact the outcome, whereas nodes with coordinates in opposite directions exert a negative effect on the outcome. Importantly, $\boldsymbol{u}_v^{(l)}$ lacks identifiability due to its invariance to orthogonal transformations. Nevertheless, as elaborated in subsequent paragraphs, our inferential interest centers around $\{v : \boldsymbol{u}_v^{(l)} = \boldsymbol{0}\}$, which is an identifiable quantity.

With the specification in (2), the vth node has no effect from the lth layer on the outcome if $\boldsymbol{u}_v^{(l)} = \boldsymbol{0}$. In order to directly infer on the influential nodes from the lth layer,

a spike-and-slab [Ishwaran and Rao, 2005] mixture distribution prior is assigned on the latent effect $\boldsymbol{u}_v^{(l)}$s. More specifically, we set

$$\boldsymbol{u}_v^{(l)} \overset{iid}{\sim} \xi_v^{(l)} N(\mathbf{0}, \boldsymbol{M}^{(l)}) + (1 - \xi_v^{(l)})\delta_{\mathbf{0}}, \;\; \xi_v^{(l)} \sim Ber(\pi_l), \;\; l = 1, ..., L, \tag{3}$$

where $\delta_{\mathbf{0}}$ is the Dirac-delta function at $\mathbf{0}$ and $\boldsymbol{M}^{(l)}$ is a $R \times R$ covariance matrix. The parameter π_l corresponds to the probability of the nonzero mixture component and $\xi_v^{(l)}$ is a binary indicator set to 0 if $\boldsymbol{u}_v^{(l)} = \mathbf{0}$. Thus, the posterior distributions of $\xi_v^{(l)}$'s enable identification of nodes related to the outcome. To account for multiplicity in multiple variable selection, we assign $\pi_l \overset{iid}{\sim} Beta(a, b)$, $l = 1, ..., L$, following popular literature [Scott and Berger, 2010]. The shared latent effects for all nodes are assigned $\overline{\boldsymbol{u}}_1, \ldots, \overline{\boldsymbol{u}}_V \overset{iid}{\sim} N(\mathbf{0}, \boldsymbol{M})$, where $\boldsymbol{M}$ is a $R \times R$ covariance matrix. All covariance matrices are assigned $\boldsymbol{M}, \boldsymbol{M}^{(1)}, ..., \boldsymbol{M}^{(L)} \overset{i.i.d.}{\sim} IW(\nu, \boldsymbol{I}_R)$ where $IW(\nu, \boldsymbol{I}_R)$ denotes an Inverse-Wishart distribution with a $R \times R$ identity matrix $\boldsymbol{I}_R$ and degrees of freedom ν.

To learn which components of $\boldsymbol{u}_v^{(l)}$ and $\bar{\boldsymbol{u}}_v$ are informative, we assign a hierarchical prior. Since the choice of R is arbitrary, allowing λ_rs and $\lambda_r^{(l)}$s to be 0 protects the model from over-fitting. These variables are assigned Multinomial-Dirichlet priors given by,

$$\lambda_r^{(l)} = \delta_1 \pi_{r,1}^{(l)} + \delta_0 \pi_{r,2}^{(l)} + \delta_{-1} \pi_{r,3}^{(l)}, \;\; \lambda_r = \delta_1 \pi_{r,1} + \delta_0 \pi_{r,2} + \delta_{-1} \pi_{r,3}, \;\; l = 1, ..., L, \;\; r = 1, ..., R,$$

$$(\pi_{r,1}^{(l)}, \pi_{r,2}^{(l)}, \pi_{r,3}^{(l)}) \sim Dir(1, r^\eta, 1), \;\; (\pi_{r,1}, \pi_{r,2}, \pi_{r,3}) \sim Dir(1, r^\eta, 1), \;\; \eta > 1.$$

The hyper-parameters of the Dirichlet distribution are chosen to introduce increasing shrinkage on λ_r and $\lambda_r^{(l)}$ as r grows. Specifically, $\hat{R} = \sum_{r=1}^{R} \lambda_r$ and $\hat{R}^{(l)} = \sum_{r=1}^{R} \lambda_r^{(l)}$ represent the dimensions of $\bar{\boldsymbol{u}}_v$ and $\boldsymbol{u}_v^{(l)}$ needed for effective modeling of the data. These quantities are referred to as the *shared effective dimension* and the *layer-specific effective dimension*, respectively. The prior specification is completed by setting a non-informative prior on μ and $IG(a_\tau, b_\tau)$ prior on τ^2.

3 Posterior Computation

The summaries from the posterior distribution cannot be computed in closed form. However, all parameters have full conditional distributions available, which belong to standard families. Therefore, the posterior computation of parameters is achieved through Gibbs sampling. Details on all the full conditionals are provided in Appendix A.

3.1 Node Selection

To determine whether the vth node has influence in the lth layer for predicting the outcome, the post-burn-in S samples $\xi_{v,(1)}^{(l)}, \ldots, \xi_{v,(S)}^{(l)}$ of $\xi_v^{(l)}$ are used. The vth node $\mathcal{A}_v$ is considered influential at the lth layer if the empirically estimated probability $P(\xi_v^{(l)} = 1|Data)$, computed as $\sum_{s=1}^{S} \xi_{v,(s)}^{(l)}/S$, is greater than 0.5. Additionally, we estimate the *layer-specific effective dimensions* $\hat{R}^{(l)}$, empirically computed as $\sum_{s=1}^{S} \sum_{r=1}^{R} |\lambda_{r,(s)}^{(l)}|/S$, where $\lambda_{r,(1)}^{(l)}, \ldots, \lambda_{r,(S)}^{(l)}$ are the post-burn-in samples of $\lambda_r^{(l)}$.

3.2 Computation Time

Figure 1 illustrates the computation time per iteration of the Gibbs sampler for the proposed method, presented in minutes. The computation time exhibits a linear increase with the sample size, and it also shows a linear growth concerning the number of nodes V, where the rate of linear growth is contingent on the sample size n. All computation times are based upon model runs on a server containing two Intel Xeon E5-2697 processors with a combined 24 cores and 48 threads and 128 GB of RAM. The Gibbs sampler demonstrates rapid convergence of all parameters, with traceplots for a few parameters provided in Appendix B.

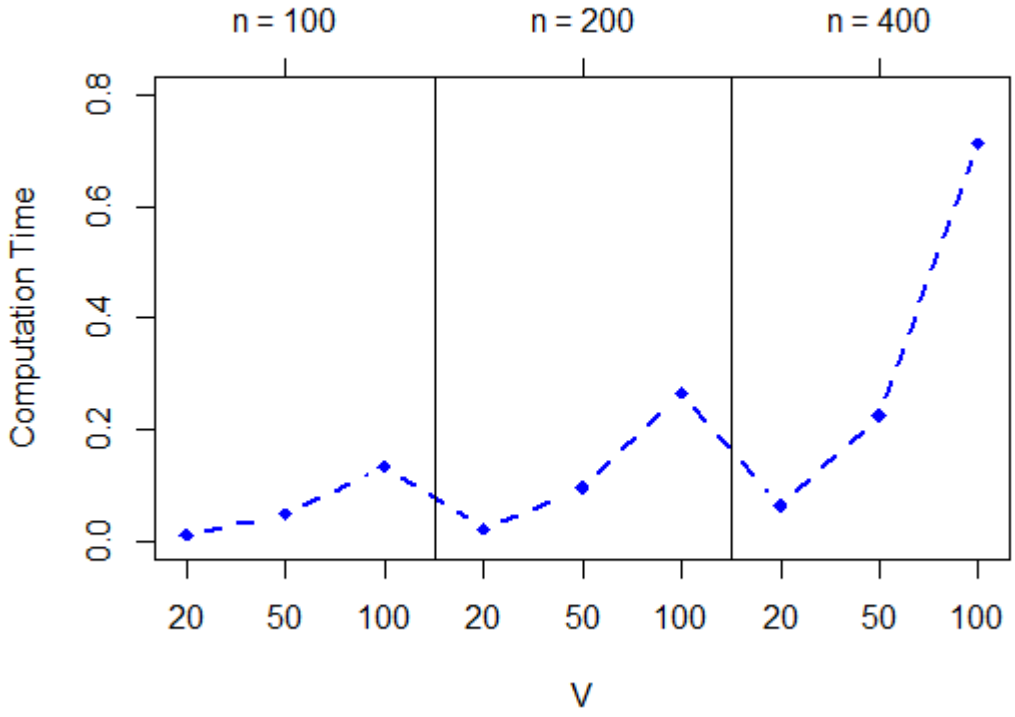


Figure 1: Computation time (in minutes) per iteration for the proposed method with respect to different choices of V and n.

4 Simulation Studies

In this section, we conduct a comprehensive comparison between the proposed Bayesian multilayer network regression (BMNR) and several competitors in various simulation settings. We evaluate both the inferential and predictive performances of BMNR in contrast to these alternative methods. Our competitors encompass penalized likelihood methods, Bayesian shrinkage priors and convolutional neural networks tailored for high-dimensional regression tasks. Additionally, we compare our method with a tensor regression approach. This comprehensive evaluation will showcase the strengths and advantages of BMNR in handling complex multilayer network data while assessing its efficacy vis-a-vis state-of-the-art alternatives. In all simulations, we fix the number of layers (L) to be equal to 4,

consistent with the number of layers in the security systems data application.

Simulated Data Generation. We simulate the multi-layer network $\boldsymbol{X}_i$, represented by symmetric adjacency matrices at L layers $\boldsymbol{X}_i^{(1)}, ..., \boldsymbol{X}_i^{(L)}$ for the ith sample, by drawing independent and identically distributed (i.i.d.) upper triangular vectors $\boldsymbol{x}_i^{(l)} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$, with all diagonal entries set to 0. The response y_i for each sample is generated following the multilayer network regression model proposed in (1), where the true error variance is τ^{*2}, the true intercept is μ^*, and the true network coefficient corresponding to $\boldsymbol{X}_i^{(l)}$ at the lth layer is denoted as $\boldsymbol{B}^{*(l)}$. In all simulations, we set $\mu^* = 0.5$ and $\tau^{*2} = 0.1$. The simulations utilize $n = 400$ samples for model fitting, reserving $n^* = 100$ samples for predictive inference.

Let π_l^* be the probability of a node at the lth layer being influential with respect to the outcome. We denote $(1 - \pi_l^*)$ as the *node sparsity* corresponding to the lth layer. To generate the true activation pattern of nodes in relation to the outcome, we simulate node-specific activation indicators $\xi_1^{*(l)}, ..., \xi_V^{*(l)} \sim \text{Ber}(\pi_l^*)$ for the lth layer, where $l = 1, ..., L$. The entries of $\boldsymbol{B}^{*(l)}$ are generated from two different scenarios.

Scenario 1: The first scenario simulates R^*-dimensional latent effects corresponding to the lth layer as $\boldsymbol{u}_v^{*(l)} \overset{ind.}{\sim} \xi_v^{*(l)} N(\boldsymbol{0}, \boldsymbol{I}) + (1 - \xi_1^{*(l)})\delta_{\boldsymbol{0}}$, for $v = 1, ..., V$. Further, the R^*-dimensional shared latent effects are drawn as $\bar{\boldsymbol{u}}_1^*, ..., \bar{\boldsymbol{u}}_V^* \overset{i.i.d.}{\sim} N(\boldsymbol{0}, \boldsymbol{I})$. The true coefficients corresponding to the (v, v')th edge at the lth layer are constructed using the low-rank formulation $\beta_{v,v'}^{*(l)} = (\bar{\boldsymbol{u}}_v^*)^T \bar{\boldsymbol{u}}_{v'}^* + (\boldsymbol{u}_v^{*(l)})^T \boldsymbol{u}_{v'}^{*(l)}$, for $v < v'$. We specify $\beta_{v,v'}^{*(l)} = \beta_{v',v}^{*(l)}$ and $\beta_{v,v}^{*(l)} = 0$ to maintain symmetry and zero diagonals in the network coefficient at each layer. In Scenario 1, we assume that the fitted and true network coefficients exhibit a similar low-rank structure, with the true rank R^* being less than the fitted rank R. In addition to evaluating the quality of inference on influential nodes, network coefficients at each layer, and predictive inference, this simulation framework enables the assessment of the quality of inference on shared effective dimension and layer-specific effective dimension.

Scenario 2: Scenario 2 allows for model misspecification by relaxing the low-rank assumption on the network coefficient $\boldsymbol{B}^{*(l)}$. Specifically, this scenario assumes $\beta^{*(l)}_{v,v'} = \bar{\beta}^{*(l)}_{v,v'} + \tilde{\beta}^{*(l)}_{v,v'}$, where $\bar{\beta}^{*(l)}_{v,v'}$ is simulated i.i.d. from N(0,1) and $\tilde{\beta}^{*(l)}_{v,v'}$ is simulated as $\tilde{\beta}^{*(l)}_{v,v'} \sim \xi^{*(l)}_{v}\xi^{*(l)}_{v'}N(0,1) + (1 - \xi^{*(l)}_{v}\xi^{*(l)}_{v'})\delta_{\mathbf{0}}$, for $v < v'$. We set $\beta^{*(l)}_{v,v'} = \beta^{*(l)}_{v',v}$ and $\beta^{*(l)}_{v,v} = 0$ to satisfy the symmetry and zero diagonals of the network coefficient at each layer, respectively. The construction of $\tilde{\beta}^{*(l)}_{v,v'}$ ensures that the edge effect connecting the nodes $\mathcal{A}_v$ and $\mathcal{A}_{v'}$ is zero if one of these nodes is not influential in predicting the outcome. Scenario 2 presents challenges due to the fitted low-rank network coefficient at each layer being employed to estimate a full-rank true network coefficient. This simulation scenario assesses the quality of inference on influential nodes, network coefficients at each layer, and predictive inference. However, it does not permit the evaluation of inference on layer-specific effective dimension, as the true network coefficient is full-rank at each layer.

For both Scenario 1 and Scenario 2, we simulate data under three distinct combinations of node sparsity at the four layers, as outlined below:

- **Combination 1.** $1 - \pi^*_1 = 0.8$, $1 - \pi^*_2 = 0.8$, $1 - \pi^*_3 = 0.7$, $1 - \pi^*_4 = 0.7$.
- **Combination 2.** $1 - \pi^*_1 = 0.4$, $1 - \pi^*_2 = 0.3$, $1 - \pi^*_3 = 0.4$, $1 - \pi^*_4 = 0.3$
- **Combination 3.** $1 - \pi^*_1 = 0.5$, $1 - \pi^*_2 = 0.8$, $1 - \pi^*_3 = 0.8$, $1 - \pi^*_4 = 0.6$.

Combinations 1-3 represent cases with high node sparsity for all layers, low node sparsity for all layers and varying node sparsity between layers, respectively. Since simulation in Scenario 1 requires specifying rank R^* for the true node-specific latent effects, we choose them to be 4,3 and 2 for Combinations 1-3, respectively. The fitted rank is set at $R = 6$. In our evaluation, we consider two different network sizes, $V = 20$ and $V = 50$, across the two scenarios and three combinations to comprehensively assess the performance of the competing methods.

Competitors. We compare the proposed BMNR with two sets of competitors. The first set of competitors treats the edges between nodes in the multilayer network predictor as a "long

vector of predictors," i.e., these competitors extract the vectorized upper triangular part $\boldsymbol{x}_i^{(l)}$ from the network at the lth layer $\boldsymbol{X}_i^{(l)}$ and perform regression of the response variable y_i on the $LV(V-1)/2$ dimensional vector $\boldsymbol{x}_i = (\boldsymbol{x}_i^{(1)T}, ..., \boldsymbol{x}_i^{(L)T})$. By adopting this approach, they overlook the relational nature of the network predictor at each layer, potentially limiting their ability to capture the effect of intricate interconnections among network nodes on the outcome. To this end, a Horseshoe prior [Carvalho et al., 2010] is employed on the regression coefficients due to its state-of-the-art empirical performance in regressions with both sparse and not-so-sparse settings. We refer to this competitor as Horseshoe, implemented through the `monomvn` [Gramacy et al., 2023] R package. A frequentist high dimensional regression competitor has also been constructed by adopting a penalized optimization framework with the Lasso penalty on the predictor coefficients [Tibshirani, 1996]. Lasso is implemented using the `glmnet` [Friedman et al., 2010] package in R, with the penalty parameter of Lasso chosen through a five-fold cross-validation technique. Additionally, a non-linear relationship between y_i and $\boldsymbol{x}_i$ is accommodated in the competitors by fitting convolutional neural network (CNN) models. CNN models are fit using the `tensorflow` [Allaire and Chollet, 2023] and `keras` [Allaire and Tang, 2023] packages in R. The inputs to the CNN models are composed of the upper-triangular entries of each network layer, stacked to create a matrix of dimension $L \times V(V-1)/2$. The models are fit with a two-dimensional convolution layer having the ReLU activation function, followed by flattening and a fully connected, linear output layer. The filter sizes, number of filters, and learning rate for the models are tuned through five-fold cross-validation. For training the CNN, 90 percent of the training samples are used to fit the model, while 10 percent are used for validation. The batch size is equal to the number of samples used to fit the model.

The second type of competitor treats the multilayer network as a tensor of dimension $V \times V \times L$ and adopts a tensor regression approach [Zhou et al., 2013] for modeling the continuous outcome and the tensor predictor constructed from the multilayer network. We

implement the tensor regression approach using the R package `TRES`. Unlike Horseshoe, Lasso and CNN, this framework accounts for the inherent association in multiple layers of the network to predict outcome, providing an alternative way to handle the multilayer network data. However, it does not account for the symmetry of the matrices at each layer of the network. Horseshoe, Lasso and Tensor Regression (TensorReg) are compared with BMNR in terms of inferential and predictive performances. In contrast, CNN is only designed to assess the comparative predictive performance with BMNR.

Metrics of Comparison. To evaluate BMNR's performance in identifying nodes significantly associated with the outcome, we present the estimated posterior probability of a node being influential in a layer, denoted as $P(\xi_v^{(l)} = 1|Data)$.

For assessing the competitors' performance in estimating the coefficient $\boldsymbol{B}^{*(l)}$, we utilize the scaled mean squared error (MSE) averaged across L layers. The MSE is defined as $\frac{1}{L}\sum_{l=1}^{L} \frac{||\widehat{\boldsymbol{B}}^{(l)} - \boldsymbol{B}^{*(l)}||^2}{||\boldsymbol{B}^{*(l)}||^2}$, where $\widehat{\boldsymbol{B}}^{(l)}$ represents a suitable point estimate of $\boldsymbol{B}^{*(l)}$. Specifically, we use the frequentist point estimate for Lasso and TensorReg, while for BMNR and Horseshoe, we use the posterior mean of $\boldsymbol{B}^{(l)}$.

To assess the quantification of uncertainty by the proposed approach, we calculate the length and coverage of posterior 95% credible intervals averaged over the edge coefficients $\beta_{v,v'}^{(l)}$ from the post burn-in MCMC samples of $\boldsymbol{B}^{(l)}$. For the frequentist competitors, we compute the coverage and length of the confidence intervals with an asymptotic coverage of 95%. For evaluating predictive power, we compute the scaled mean squared prediction error, given by $||\boldsymbol{y}^* - \boldsymbol{y}^*_{pred}||^2/||\boldsymbol{y}^*||^2$, where $\boldsymbol{y}^*$ is the true value of the response at n^* out-of-sample observations, and $\boldsymbol{y}^*_{pred}$ is the point prediction from competitors at these values. Bayesian competitors impute $\boldsymbol{y}^*_{pred}$ as the mean of the posterior predictive density. All results presented are averaged over five simulation replicates.

4.1 Simulation Results

4.1.1 Accuracy of Influential Node Identification

Table 1 displays three matrices corresponding to three different simulation combinations under *Scenario 1* for $V = 20$. Table 2 displays corresponding matrices for *Scenario 2* for $V = 20$. In each matrix, the colored and white cells in the lth column represent truly influential and truly uninfluential nodes in the lth network layer, respectively. Overlaid on these matrices are the estimated posterior probabilities $P(\xi_v^{(l)} = 1|Data)$ for all layers and nodes, $l = 1, .., 4$ and $v = 1, ..., 20$.

Table 1 corresponding to Scenario 1 illustrates highly accurate identification of truly influential nodes in scenarios characterized by low node-sparsity and varying sparsity between layers. In these instances, the estimated posterior probabilities are extremely close to either 1 or 0, correctly categorizing nodes as influential or not, with minimal uncertainty. However, in Combination 1, characterized by high node-sparsity across all layers, there are instances of both false positives and false negatives. Notably, even Combination 1 leads to excellent estimation of model coefficients and predictive inference discussed in Sections 4.1.2 and 4.1.3. The discrepancy may be attributed to the fact that high node-sparsity results in high sparsity in the edge coefficients across all layers. The abundance of zeros in the true edge coefficients creates a weak network structure, allowing our method to closely estimate the true coefficient due to the modeling architecture inducing sparsity, even when some false positives and negatives occur. In Scenario 2, the true coefficient does not exhibit a low-rank structure. Even amidst such mis-specification, the model largely distinguishes truly influential and uninfluential nodes. However, for uninfluential nodes, the probability $P(\xi_v^{(l)} = 1|Data)$ tends to fall within the range of $0.2 - 0.45$, rather than being very close to zero. The trends are largely similar for $V = 50$, though they are not presented due to space constraint.

It is important to emphasize that one of BMNR's key inferential strengths lies in its

Node	Layer 1	Layer 2	Layer 3	Layer 4
1	0.0000	0.0050	0.0000	0.0000
2	1.0000	1.0000	1.0000	1.0000
3	0.0000	0.0020	0.0000	0.0000
4	0.0000	0.0205	0.0000	0.0000
5	0.0000	0.0365	0.0000	1.0000
6	0.0500	0.2235	1.0000	0.0000
7	0.0030	0.3305	0.0000	1.0000
8	1.0000	1.0000	1.0000	1.0000
9	1.0000	1.0000	0.0000	0.0000
10	1.0000	1.0000	1.0000	1.0000
11	0.0000	1.0000	1.0000	0.0000
12	1.0000	1.0000	1.0000	1.0000
13	1.0000	1.0000	1.0000	1.0000
14	0.9465	0.8860	0.6905	0.7130
15	0.0000	1.0000	0.0000	0.0000
16	0.0000	0.9555	0.0000	1.0000
17	1.0000	1.0000	1.0000	1.0000
18	1.0000	1.0000	1.0000	1.0000
19	1.0000	1.0000	1.0000	1.0000
20	0.0000	0.0520	1.0000	0.0000

(a) Combination 1

Node	Layer 1	Layer 2	Layer 3	Layer 4
1	1.0000	1.0000	0.0000	1.0000
2	1.0000	1.0000	1.0000	0.0000
3	1.0000	0.0025	1.0000	1.0000
4	0.0000	1.0000	0.0000	0.0000
5	0.0025	0.0020	0.0000	1.0000
6	0.0000	1.0000	1.0000	1.0000
7	0.0005	1.0000	1.0000	1.0000
8	0.0085	1.0000	1.0000	0.0000
9	1.0000	1.0000	0.0005	1.0000
10	0.0005	1.0000	1.0000	1.0000
11	1.0000	0.0040	0.0000	1.0000
12	1.0000	0.0025	1.0000	1.0000
13	0.0000	1.0000	1.0000	1.0000
14	1.0000	1.0000	1.0000	0.0000
15	1.0000	1.0000	0.0000	0.0000
16	1.0000	1.0000	0.0030	1.0000
17	1.0000	0.0015	1.0000	1.0000
18	1.0000	1.0000	1.0000	0.0000
19	0.0000	0.0010	0.0000	1.0000
20	1.0000	1.0000	1.0000	1.0000

(b) Combination 2

Node	Layer 1	Layer 2	Layer 3	Layer 4
1	1.0000	0.0005	0.0000	0.0000
2	1.0000	1.0000	0.0005	0.0005
3	0.0005	0.0005	0.0000	0.0000
4	0.0000	0.0000	1.0000	1.0000
5	0.0000	0.0025	1.0000	1.0000
6	1.0000	0.0005	0.0000	0.0000
7	1.0000	0.0005	0.0000	1.0000
8	1.0000	0.0010	0.0010	0.0000
9	0.0000	1.0000	0.0005	1.0000
10	0.0000	0.0005	0.0005	0.0000
11	0.0000	0.0000	0.0000	0.0005
12	1.0000	0.0000	0.0000	1.0000
13	0.0000	0.0000	0.0000	0.0000
14	1.0000	0.0000	0.0000	0.0000
15	0.0000	1.0000	0.0000	0.0000
16	1.0000	0.0000	1.0000	1.0000
17	1.0000	0.0000	0.0000	0.0000
18	0.0000	0.0000	0.0000	1.0000
19	0.0000	1.0000	1.0000	0.0005
20	1.0000	0.0000	0.0000	1.0000

(c) Combination 3

Table 1: Node selection results for V = 20 under Combinations 1,2 and 3 of Scenario 1. In each matrix, the colored and white cells in the lth column represent truly influential and truly uninfluential nodes in the lth network layer, respectively.

ability to detect important nodes in multilayer networks while accounting for the interplay between multiple layers, all while quantifying uncertainty. This capability aligns directly with the inferential objectives of our scientific study, which involve identifying critical actors or nodes within network security systems in high-consequence facilities. In contrast, existing frequentist or Bayesian high-dimensional regression methods (e.g., Lasso, Horseshoe), tensor regression approaches (e.g., TensorReg), or neural network-based methods (e.g., CNN) do not provide node identification in the present context.

Node	Layer 1	Layer 2	Layer 3	Layer 4
1	0.3700	0.4600	0.9900	0.9000
2	0.6000	0.3300	0.3700	0.4600
3	0.3400	0.3600	0.5000	0.4000
4	0.7800	0.6000	0.7400	0.6300
5	0.9900	0.3600	0.4400	0.7400
6	0.5400	0.4100	1.0000	1.0000
7	0.3500	0.4300	0.4100	0.9400
8	0.3600	0.4200	0.3800	0.2900
9	0.3700	0.3500	0.2600	0.1100
10	0.3000	0.2800	0.2900	0.4100
11	0.8000	0.7500	0.4600	0.9500
12	0.1900	0.2600	0.3800	0.4300
13	0.4600	0.4400	0.4500	0.4300
14	0.3900	0.3900	1.0000	0.3200
15	0.1900	0.1500	0.1300	0.3600
16	0.4000	0.9700	0.4600	0.3200
17	0.3200	0.6400	0.9400	0.3800
18	0.3200	0.4300	0.3100	1.0000
19	0.4800	0.8000	0.6100	0.2200
20	0.3600	0.2800	0.4600	0.9200

(a) Combination 1

Node	Layer 1	Layer 2	Layer 3	Layer 4
1	0.9600	0.3800	0.7500	0.8600
2	0.9800	0.3400	0.7400	0.3600
3	0.9900	0.9000	0.3400	0.3900
4	0.4500	0.5800	0.5200	0.8800
5	0.9900	0.9900	0.3700	0.3200
6	0.9300	0.9500	0.8700	1.0000
7	1.0000	0.9700	1.0000	0.9600
8	0.3500	0.8200	1.0000	0.7800
9	0.9800	0.5500	0.3100	1.0000
10	0.4200	0.9600	0.9700	0.4100
11	0.7800	0.9100	0.4200	0.9700
12	0.8600	0.3300	0.9900	0.7000
13	0.4200	1.0000	0.9700	1.0000
14	0.9000	0.8200	0.4600	0.4200
15	0.3800	0.6500	0.9000	0.9600
16	0.9400	0.3800	0.3600	0.9600
17	0.2500	0.2400	0.7000	0.3500
18	0.9500	1.0000	0.7400	0.9900
19	0.2500	0.8400	0.2100	0.7300
20	0.8700	0.6100	0.3300	0.6500

(b) Combination 2

Node	Layer 1	Layer 2	Layer 3	Layer 4
1	0.3400	0.9300	0.2500	1.0000
2	0.7800	0.3600	0.2200	0.3700
3	0.9500	0.3300	0.3800	0.3600
4	0.8800	0.8100	0.3400	0.6500
5	0.9700	0.2800	0.3200	0.9300
6	0.8800	0.3900	0.6300	1.0000
7	0.9200	0.3500	0.2700	1.0000
8	0.3300	0.2400	0.3400	0.3500
9	0.3400	0.3400	0.2300	0.1900
10	0.2700	0.3500	0.4100	0.3900
11	0.9700	0.4100	0.7300	0.4000
12	0.9100	0.2900	0.4900	0.3700
13	0.8700	0.3900	0.3200	0.3000
14	0.4000	0.9900	0.3400	0.3300
15	0.9800	0.4400	0.3800	0.4400
16	0.3400	0.5500	0.3900	0.3300
17	0.3100	0.7500	0.4000	0.9400
18	0.2500	0.3100	0.2900	0.3400
19	0.2600	0.3100	0.7300	0.9800
20	0.6000	0.4100	0.7900	0.6200

(c) Combination 3

Table 2: Node selection results for V = 20 under Combinations 1,2 and 3 of Scenario 2. In each matrix, the colored and white cells in the lth column represent truly influential and truly uninfluential nodes in the lth network layer, respectively.

4.1.2 Estimation of Regression Coefficients

To evaluate the point estimation of multilayer network coefficients, Table 3 presents the scaled Mean Squared Error (MSE) for each of the competitors across the three combinations in the two scenarios. In all cases, the results consistently demonstrate superior performance of BMNR compared to its competitors. In *Scenario 1*, where the true coefficients in each layer assume a low-rank decomposition, BMNR significantly outperforms all its competitors. However, in *Scenario 2*, where model misspecification is introduced, the performance gap between BMNR and its competitors narrows, although BMNR still maintains an advantage. Furthermore, BMNR consistently exhibits better performance under Combinations 1 and 3, where there is high-node sparsity in all layers and varying node-sparsity between layers, respectively, compared to Combination 2, which exhibits low-node sparsity in all layers. In *Scenario 1*, TensorReg emerges as the second-best performer, primarily due to the imposed low-rank structure on the true coefficients which is conducive to TensorReg. However, under misspecification in *Scenario 2*, Horseshoe outperforms Lasso and TensorReg. All competitors perform better for lower dimensional cases with the number of nodes $V = 20$ as compared to $V = 50$.

BMNR offers a notable advantage over its competitors when it comes to characterizing uncertainty in parameter estimation. Figure 2 demonstrates that, for $V = 20$ in *Scenario 1*, a scenario that particularly favors BMNR, it achieves close-to-nominal coverage with significantly shorter credible intervals. Even under the more challenging *Scenario 2* with $V = 20$, BMNR consistently maintains coverage around 90%. Horseshoe, while achieving coverage around 80%, comes with slightly wider credible intervals compared to BMNR. TensorReg shows similar coverage to Horseshoe but with slightly shorter credible intervals under *Scenario 1*, though it is outperformed by Horseshoe in *Scenario 2*. In contrast, Lasso consistently underperforms in all scenarios.

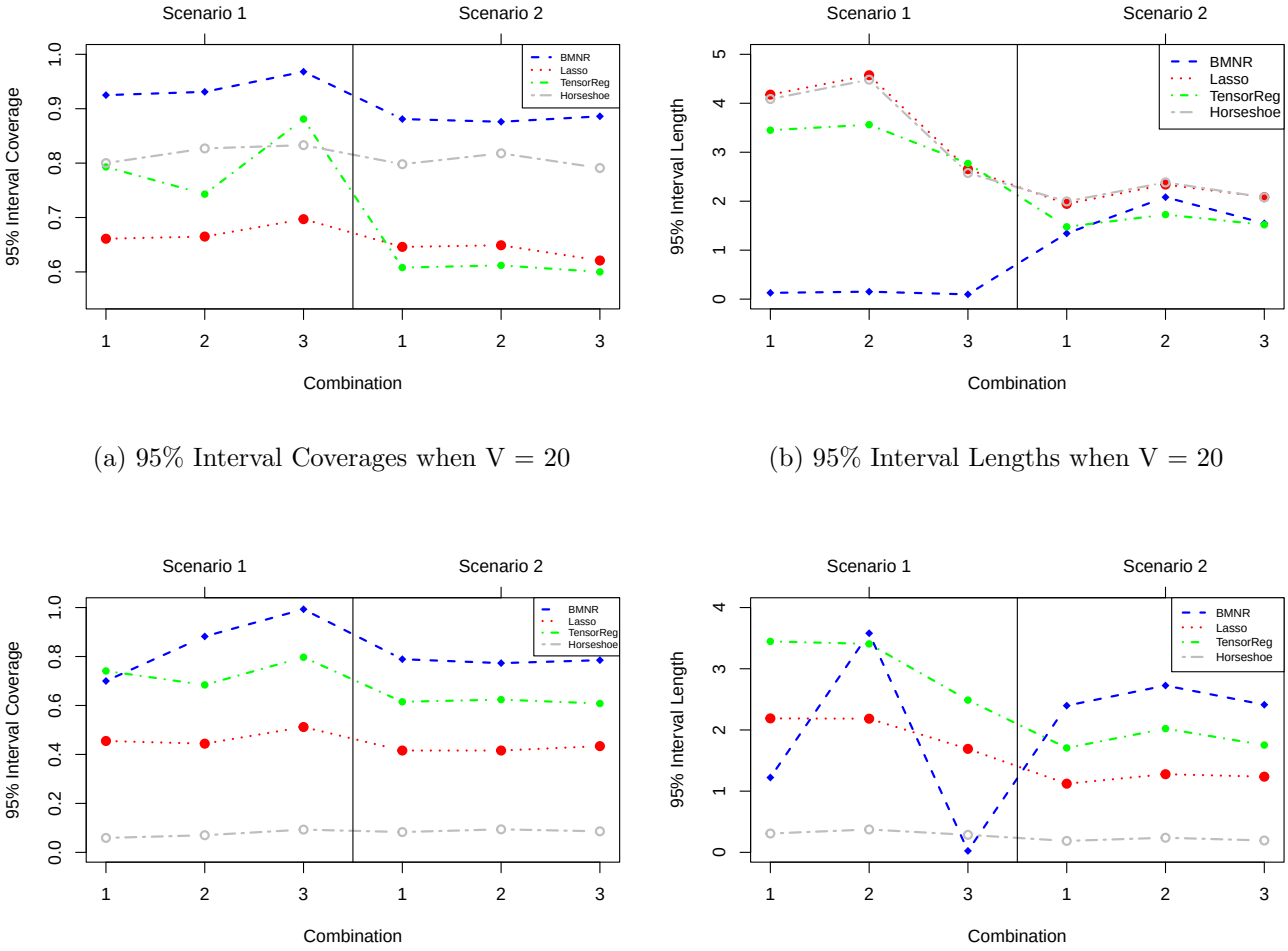
In the case of high-dimensions with $V = 50$, where both BMNR and TensorReg employ

dimension reduction, these two methods emerge as the top performers. Under *Scenario 1*, BMNR exhibits much narrower credible intervals than TensorReg for Combinations 1 and 3, but their performance is equivalent for Combination 2, which has low node-sparsity across layers. In *Scenario 2*, when neither TensorReg nor BMNR holds a clear advantage, BMNR maintains slightly higher coverage compared to TensorReg, albeit with slightly wider credible intervals.

On the other hand, both Lasso and Horseshoe are ill-suited for high-dimensional scenarios with $V = 50$, which results in approximately 8000 predictors (upon vectorization of the multilayer network) with a limited sample size of $n = 400$. Consequently, both Lasso and Horseshoe significantly underperform compared to BMNR and TensorReg in these high-dimensional settings. Overall, BMNR stands out as a robust method for delivering point estimates and characterizing uncertainty in parameter estimation, particularly in scenarios with complex network structures.

	Scenario 1					
	$V = 20$			$V = 50$		
Method	Comb. 1	Comb. 2	Comb. 3	Comb. 1	Comb. 2	Comb. 3
BMNR	**0.0006**	**0.0008**	**0.0007**	**0.1005**	**0.3748**	**0.0014**
Lasso	1.5030	1.5331	1.4064	1.0012	1.0104	1.0110
TensorReg	0.5794	0.6791	0.5344	0.7518	0.8935	0.6071
Horseshoe	0.7420	0.6547	0.6052	0.9907	0.9901	0.9960
	Scenario 2					
	$V = 20$			$V = 50$		
Method	Comb. 1	Comb. 2	Comb. 3	Comb. 1	Comb. 2	Comb. 3
BMNR	**0.2301**	**0.3965**	**0.2769**	**0.8961**	**0.9288**	**0.9026**
Lasso	1.3857	1.5426	1.6366	1.0052	1.0001	1.0372
TensorReg	0.8992	0.9338	0.9358	1.0219	1.0284	1.0228
Horseshoe	0.7086	0.6854	0.7133	0.9914	0.9911	0.9915

Table 3: Mean squared errors (MSE) in estimating multilayer network coefficients.



(a) 95% Interval Coverages when V = 20

(b) 95% Interval Lengths when V = 20

(c) 95% Interval Coverages when V = 50

(d) 95% Interval Lengths when V = 50

Figure 2: Length and coverage of 95% credible intervals for Bayesian competitors and 95% confidence intervals for frequentist competitors for three combinations under *Scenario 1* and *Scenario 2*, with $V = 20$ as well as $V = 50$.

4.1.3 Predictive Inference

In our analyses, we conducted predictive inference on the competitors using a dataset of $n^* = 100$ samples generated according to the multilayer network regression model described in (1). The predictive abilities of the competitors were evaluated based on the scaled Mean Squared Prediction Error (MSPE).

Notably, in *Scenario 1*, BMNR consistently outperforms all competitors across all combinations, demonstrating superior predictive performance for both $V = 20$ and $V = 50$. Even when faced with model misspecification in *Scenario 2*, BMNR maintains a competitive edge in predictive performance, although the performance gap between BMNR and its competitors narrows compared to *Scenario 1*, as expected. BMNR's predictive accuracy seems more pronounced when the number of predictors is set to 20 compared to when it is 50.

The data generation schemes in both *Scenario 1* and *Scenario 2* are devised such that the true coefficients in the multilayer network matrices take reasonably large positive and negative values. Additionally, a substantial number of edge coefficients are set to zero. Given the limited sample size of $n = 400$ and the large number of edges, reaching as high as 760 for $V = 20$ and 4900 for $V = 50$, all local and global parameters in the ordinary vector shrinkage priors are effectively shrunk to zero. This behavior facilitates accurate estimation of zero coefficients but leads to poor point prediction for coefficients significantly deviating from zero. Consequently, ordinary vector shrinkage priors, such as the Horseshoe, demonstrate relatively poor performance in terms of point prediction. Lasso exhibits poor performance for similar reasons.

Among the competitors, TensorReg is better equipped to handle high-dimensional matrix-valued predictors stacked at multiple layers. However, its performance is considerably compromised due to the lack of addressing the symmetry of network predictor coefficients at each layer. In contrast, CNN performs better among all competitors, as it is specifically designed for drawing targeted predictive inference. While BMNR significantly outperforms CNN for *Scenario 1*, CNN demonstrates comparable performance with BMNR under *Scenario 2* with model misspecification.

	Scenario 1					
	$V = 20$			$V = 50$		
Method	Comb. 1	Comb. 2	Comb. 3	Comb. 1	Comb. 2	Comb. 3
BMNR	**0.0006**	**0.0008**	**0.0002**	**0.1257**	**0.3286**	**0.00003**
Lasso	0.7147	0.6143	0.6298	1.0020	0.9897	0.9976
TensorReg	0.6178	0.7226	0.5931	0.8650	0.8632	0.6453
Horseshoe	0.7414	0.6513	0.6512	1.0022	0.9813	0.9815
CNN	0.4381	0.5123	0.6001	0.9305	0.9205	0.9449
	Scenario 2					
	$V = 20$			$V = 50$		
Method	Comb. 1	Comb. 2	Comb. 3	Comb. 1	Comb. 2	Comb. 3
BMNR	**0.2295**	**0.4058**	**0.2343**	0.9502	**0.9166**	0.9523
Lasso	0.7259	0.6949	0.8012	1.0028	1.0008	1.0329
TensorReg	0.9544	0.9826	0.9362	1.0574	1.0622	1.0194
Horseshoe	0.7194	0.6878	0.6900	1.0090	0.9887	1.0155
CNN	0.2934	0.4656	0.3587	**0.9490**	0.9472	**0.9466**

Table 4: Predictive performance of BMNR and its competitors are presented for Combinations 1, 2 and 3 under the two scenarios. The point prediction of the competitors are compared using scaled mean squared prediction error (MSPE) computed from $n^* = 100$ out-of-sample observations.

4.1.4 Estimation of Effective Dimensionalities

Table 5 presents the estimated shared effective dimension and layer-specific effective dimension, denoted as $\hat{R}$ for $\bar{\boldsymbol{u}}_v$ and $\hat{R}^{(l)}$ for $\boldsymbol{u}_v^{(l)}$, respectively. These estimates are compared to the true dimensions of node latent variables used in the data generation, denoted as R^*,

specifically in the context of *Scenario 1*. There is no assumption of low-rank specification in *Scenario 2* for simulating network coefficients, and therefore, such comparisons are not applicable under this scenario.

It is worth emphasizing that effective dimensionalities for latent variables are inherently challenging to identify with precision, especially for multilayer network coefficients. Although we notice occasional instances of minor over-estimation and under-estimation across certain layers, the estimated effective dimensionalities generally provide a close approximation of the true latent dimensions.

V	Combination	R^*	$\hat{R}$	$\hat{R}^{(1)}$	$\hat{R}^{(2)}$	$\hat{R}^{(3)}$	$\hat{R}^{(4)}$
20	1	4	4.0000	3.0000	3.0000	4.0000	5.0000
	2	3	4.0000	3.0150	3.3605	3.5495	5.0000
	3	2	2.0000	2.0000	2.3850	2.2615	2.7700
50	1	4	4.0000	3.0000	3.0000	5.0000	4.0000
	2	3	4.4105	2.7255	2.5950	2.8125	2.9935
	3	2	3.0000	4.0000	3.0000	2.0000	3.0000

Table 5: Estimation of the Effective Dimensions in *Scenario 1*. Here R^* is the true dimension of the latent variables, $\hat{R}$ is the estimated shared effective dimension, and $\hat{R}^{(l)}$ is the estimated lth layer-specific effective dimension.

4.1.5 Sensitivity to the Choice of R

As the analysis involves a user-defined choice of R, it is essential to conduct additional investigations to assess the sensitivity of inference to different values of R. To address this, we carried out additional model runs for each simulation scenario with R set to various values, specifically $R = 10, 15, 20$, in addition to our initially chosen value of $R = 6$. Table 6 presents the results for the MSE, coverage of the 95% CI, length of the 95% CI for estimating multi-layer network coefficients, and the MSPE for predictive inference, specifically for

the dataset associated with Combination 1 in Scenario 1, where the number of nodes is $V = 20$. These results indicate that all the comparison metrics remain sufficiently robust across different values of R. Comparable explorations into the impact of R in alternative simulation scenarios yielded similar findings.

R	MSE	Coverage of 95% CI	Length of 95% CI	MSPE
10	0.0002	0.8728	0.0813	0.0003
15	0.0006	0.9408	0.1209	0.0006
20	0.00002	0.9601	0.0316	0.0001

Table 6: MSE, Coverage of 95% credible interval, length of 95% credible interval for estimating multi-layer network coefficients and MSPE for predicting the outcome in BMNR with different choices of R.

5 Analysis of Emulator Data on Network Security Systems

In this section, we examine data originating from a simulation model of a high-consequence facility (HCF) security system, as proposed and developed by Williams et al. [2020]. The simulation model explores various events relevant to HCF security, spanning from intrusions to infrastructure failures. It is characterized as an agent-based simulation model representing a hypothetical security system.

The hypothetical Lone Pine Nuclear Power Plant (LPNPP) has been used extensively by Sandia National Laboratories for international security and physical protection training and demonstration purposes [Osborn et al., 2019]. A combination of security and modeling subject matter expertise has been used to transform the hypothetical LPNPP into a multi-layer network [Williams et al., 2020]. This network encompasses four layers, namely (i) data

security layer, (ii) power security layer, (iii) human security layer, and (iv) communication security layer. Subsequently, this network is passed into a multi-agent dynamic model that simulates a sequence of events: initially, positioning an adversary at the security system's border; next, determining the optimal path for the adversary to reach critical interior components; and finally, executing the optimal path until the adversary is either detected and assessed by the security system or successfully reaches critical interior components.

Multiple simulation runs were conducted, with two times recorded for each run. These were: T_1: the time the adversary comes into range of potential detection, and, T_2: the time the adversary is detected by a guard or sensor. The response of interest is the *Time until Detection* (TUD), which is denoted by $(T_2 - T_1)$.

The data we examine originates from a series of experimental runs designed to explore the significance of specific nodes within the multilayer network system in an ad hoc manner. The experiment involved the sequential removal of 10 nodes at each step from the network until 100 nodes were removed, spanning 10 steps. Notably, three nodes crucial to system function—the central power and communications nodes, along with one of the primary or secondary central command nodes—were deemed non-removable. This decision not only ensured reasonable simulation results, but also reflected the reality that eliminating the command centers represented by the central nodes was difficult. Following each removal step, the multilayer network underwent 256 simulations, resulting in 256 response times (TUD) obtained from the simulator.

Figure 3 illustrates the median TUD times (median computed over 256 simulations) for the experimental runs across the 10 steps. These 10 steps are repeated 32 times. Out of these 32 repetitions, 31 repetitions involved random removal of 10 nodes at each step, denoted as the *random removal* simulations. In contrast, one repetition implemented the structured removal of nodes, following the order of nodes' importance ranked by their degrees, referred to as the *structured removal*. We note that the degree of a node is the

number of connections that it has to other nodes in the network. The TUDs corresponding to random node removal are indicated by the grey lines, while the solid black line represents the TUDs for structured removal. Note that after 80 high degree nodes were removed in the structured removal, the adversary could no longer be detected by the security system. This is represented by the missing observations along the black line in Figure 3. The experiment suggests a likely association between high-degree nodes and the TUD.

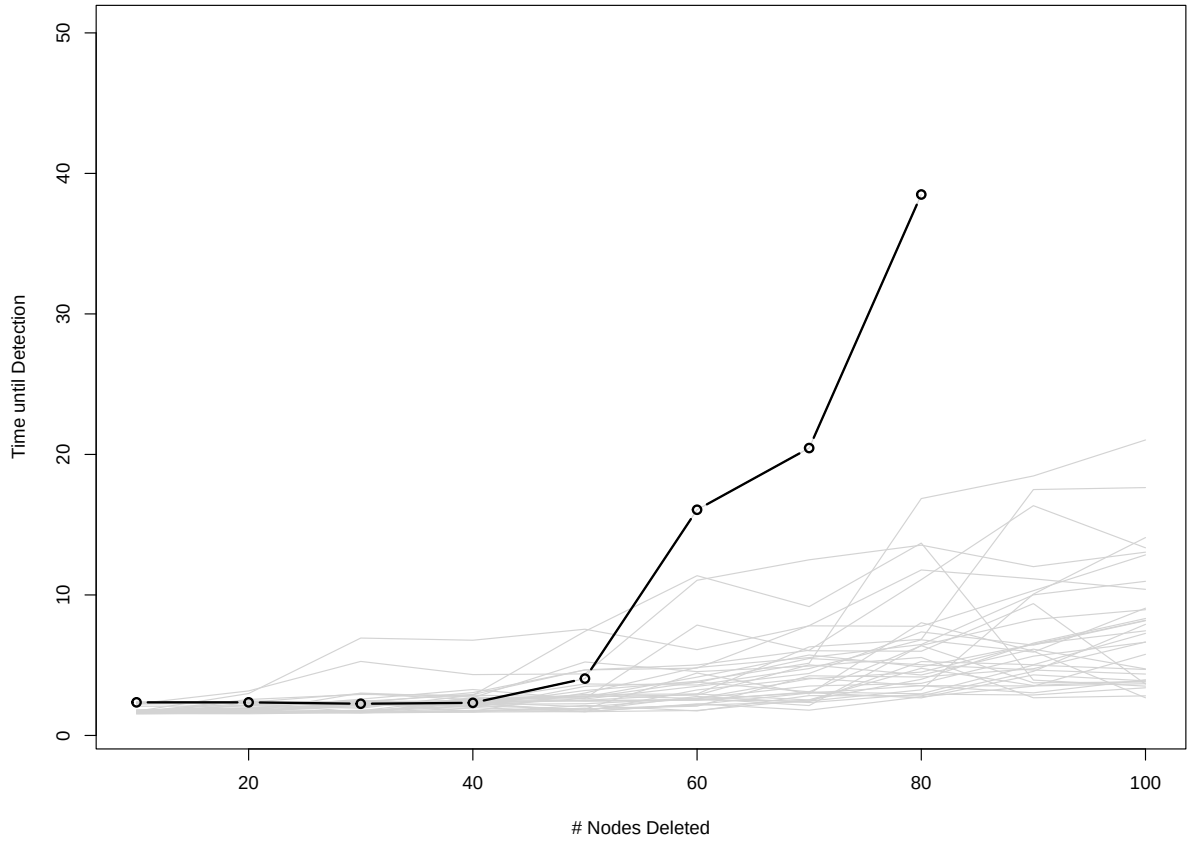


Figure 3: Time Until Detection for 32 experimental runs. The gray lines show the paths for the 31 runs where nodes were removed randomly (random removal). The black line shows the result when nodes were removed in order of importance, ranked by degree (structured removal).

The described experiment, while offering an ad-hoc insight into the relationship between

high-degree nodes and TUD, falls short of addressing two crucial objectives. First, it does not enable model-based inference on individual nodes when considering their collective impact on TUD. Second, it does not support model-based predictive inference for TUD based on the multilayer network. These represent challenging and important questions for security scientists, but they are difficult to achieve with the experiment described above, as it lacks randomness in the input network.

To address these inferential objectives, our team, in collaboration with scientists from the Sandia National Laboratories, have introduced randomness into the data. Specifically, for each sample, we constructed a very sparse Erdős-Rényi network and overlaid it on top of the original multilayer security network with 107 nodes. Importantly, due to the high sparsity of the added Erdős-Rényi network, the degree of each node in each layer is largely unaltered. Examples of sampled communication security layers constructed in this manner are shown in Figure 4. The analyzed data consisted of 431 pairs of multilayer networks constructed in this way, along with the corresponding TUDs (response) values computed from the simulator.

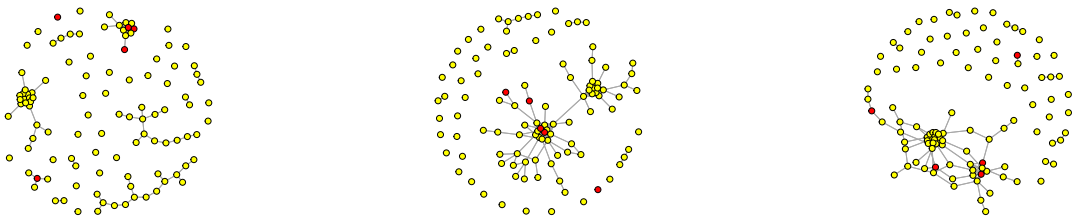


Figure 4: Examples of noise-added communication security layers.

With 431 pairs of multilayer networks and scalar-valued TUD, we partitioned the data into $n = 345$ training samples and $n^* = 86$ test samples. BMNR and its competitors are fitted on the training data and are used to draw predictive inference on the test data. Importantly, we offer identification of influential nodes using BMNR. Additionally, predictions of TUD for all competitors are obtained over $n^* = 86$ out-of-sample observations. We

conduct ten replications, fitting all competitors to ten different training-test data splits, and the results are described in the following section.

5.1 Results

Table 7 shows the performance of competing methods in terms of predicting TUD based on the multilayer network on $n^* = 86$ out-of-sample observations averaged over ten different splitting. BMNR largely outperforms all competitors in this task except CNN which shows about 10% higher MSPE compared to BMNR.

Model	BMNR	Lasso	TensorReg	Horseshoe	CNN
MSPE	**0.6769**	0.7590	0.9813	1.2118	0.7206

Table 7: Predictive performance of BMNR and its competitors in predicting TUD on the Emulator Data.

Due to security restrictions, we are unable to provide the details regarding the nodes identified as influential by BMNR. Nevertheless, we can offer a general overview of node selection. Specifically, BMNR identifies 5 nodes in the communication layer, 4 nodes in the data layer, 5 nodes in the human layer, and 7 nodes in the power layer.

A majority of the identified nodes in the data, power, and communication layers exhibit high degrees, indicating their membership in node clusters. However, the nodes identified in the human layer do not exhibit high degrees. This can be attributed to the small size of the node clusters in the human layer, which are easily influenced by added noise in the design. Our model's analysis broadly aligns with the conclusions drawn from the original simulator experiments, indicating the importance of certain high-degree nodes in designing the security architecture. It is noteworthy that our approach represents the *first* statistical exploration of the association between multilayer security networks and Time to Detection (TUD) at the Sandia National Laboratories in a principled model-based manner.

6 Conclusion and Future Work

This article introduces a novel Bayesian framework designed to address a regression problem characterized by a continuous outcome and a multilayer network-valued predictor, with each layer representing an undirected network. Our contribution encompasses modeling of network coefficients associated with different layers using low-rank structures with shared parameters across layers. This approach enables us to capture complex relationships between the outcome and various network layers by leveraging information within and across these layers. The empirical findings from our simulation studies validate the superior efficacy of our approach in situations where the regression coefficients demonstrate a multilayer network structure. Furthermore, we use our methodology to analyze emulator data of multilayer security networks from the Sandia National Laboratories. The proposed framework offers inference on multilayer network coefficients, identifies nodes within each network layer significantly related to the continuous outcome, and provides accurate predictions of the continuous outcome, all while quantifying uncertainty with precision.

Our future research will be directed towards expanding our approach to handle polychotomous outcome variables. Additionally, we aim to accommodate non-linear regression relationships between the outcome variable and the multilayer network predictor by incorporating a semiparametric regression framework. Such explorations will enhance the versatility and applicability of our method to a wider range of data scenarios and research questions.

7 Acknowledgements

This work was supported by the Laboratory Directed Research and Development program at Sandia National Laboratories, a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia LLC, a wholly owned subsidiary

of Honeywell International Inc. for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525. This paper describes objective technical results and analysis. Any subjective views or opinions that might be expressed in the paper do not necessarily represent the views of the U.S. Department of Energy or the United States Government.

Appendix A

The full conditional distributions for the model parameters are given by:

- $\tau^2 \mid - \sim IG\left(a_\tau + \frac{n}{2}, b_\tau + \frac{1}{2}\sum_{i=1}^{n}\left(y_i - \mu - \sum_{l=1}^{L}\boldsymbol{x}_i^{(l)T}\boldsymbol{\beta}^{(l)}\right)^2\right)$
- $\mu|- \sim N\left(\frac{1}{n}\underline{\mathbf{1}}_n^T(\boldsymbol{y} - \sum_{l=1}^{L}\boldsymbol{A}^{(l)}\boldsymbol{\beta}^{(l)}), \frac{\tau^2}{n}\right)$, where $\boldsymbol{A}^{(l)}$ is a $n \times V(V-1)/2$ matrix with its ith row given by $\boldsymbol{x}_i^{(l)T}$, and $\boldsymbol{y} = (y_1, ..., y_n)^T$ is a n-dimensional vector of continuous outcomes over all samples.
- $\bar{\boldsymbol{u}}_v|- \sim N(\bar{\boldsymbol{\mu}}_v, \bar{\boldsymbol{\Sigma}}_v)$, where $\bar{\boldsymbol{\Sigma}}_v = \left(\frac{1}{\tau^2}\boldsymbol{\Lambda}\bar{\boldsymbol{u}}_{-v}^T\boldsymbol{Z}_v^T\boldsymbol{Z}_v\bar{\boldsymbol{u}}_{-v}\boldsymbol{\Lambda} + \boldsymbol{M}^{-1}\right)^{-1}$, $\bar{\boldsymbol{\mu}}_v = \frac{1}{\tau^2}\bar{\boldsymbol{\Sigma}}_v\boldsymbol{\Lambda}\bar{\boldsymbol{u}}_{-v}^T\boldsymbol{Z}_v^T\tilde{\boldsymbol{y}}_v$. Here $\bar{\boldsymbol{u}}_{-v}$ is a $(V-1) \times R$ matrix given by $[\bar{\boldsymbol{u}}_1 : \cdots : \bar{\boldsymbol{u}}_{v-1} : \bar{\boldsymbol{u}}_{v+1} : \cdots : \bar{\boldsymbol{u}}_V]^T$, $\tilde{\boldsymbol{y}}_v$ is a n-dimensional vector with its ith element given by:

  $\tilde{y}_i = y_i - \mu - \sum_{k<k':k,k'\neq v}(\sum_{l=1}^{L}x_{i,(k,k')}^{(l)})\bar{\boldsymbol{u}}_k^T\boldsymbol{\Lambda}\bar{\boldsymbol{u}}_{k'} - \sum_{k<k'}\sum_{l=1}^{L}x_{i,(k,k')}^{(l)}(\boldsymbol{u}_k^{(l)})^T\boldsymbol{\Lambda}^{(l)}\boldsymbol{u}_{k'}^{(l)}$.

  Finally, $\boldsymbol{Z}_v$ is a $n \times (V-1)$ dimensional matrix with its ith row is given by $(\sum_{l=1}^{L}x_{i,(1,v)}^{(l)}, ..., \sum_{l=1}^{L}x_{i,(v-1,v)}^{(l)}, \sum_{l=1}^{L}x_{i,(v,v+1)}^{(l)}, ..., \sum_{l=1}^{L}x_{i,(v,V)}^{(l)})$.
- $\boldsymbol{u}_v^{(l)}|- \sim \xi_v^{(l)}N(\boldsymbol{\mu}_v^{(l)}, \boldsymbol{\Sigma}_v^{(l)}) + (1-\xi_v^{(l)})\delta_{\mathbf{0}}$, where $\boldsymbol{\Sigma}_v^{(l)} = \left(\frac{1}{\tau^2}\boldsymbol{\Lambda}^{(l)}\boldsymbol{u}_{-v}^{(l)T}\boldsymbol{Z}_v^{(l)T}\boldsymbol{Z}_v^{(l)}\boldsymbol{u}_{-v}^{(l)}\boldsymbol{\Lambda}^{(l)} + (\boldsymbol{M}^{(l)})^{-1}\right)^{-1}$, $\boldsymbol{\mu}_v^{(l)} = \frac{1}{\tau^2}\boldsymbol{\Sigma}_v^{(l)}\boldsymbol{\Lambda}^{(l)}\boldsymbol{u}_{-v}^{(l)T}\boldsymbol{Z}_v^{(l)T}\tilde{\boldsymbol{y}}_v^{(l)}$. Here $\boldsymbol{u}_{-v}^{(l)}$ is a $(V-1) \times R$ matrix given by $[\boldsymbol{u}_1^{(l)} : \cdots : \boldsymbol{u}_{v-1}^{(l)} : \boldsymbol{u}_{v+1}^{(l)} : \cdots : \boldsymbol{u}_V^{(l)}]^T$, $\tilde{\boldsymbol{y}}_v^{(l)}$ is a n-dimensional vector with its ith element given by

  $\tilde{y}_i^{(l)} = y_i - \mu - \sum_{k<k'}(\sum_{l=1}^{L}x_{i,(k,k')}^{(l)})\bar{\boldsymbol{u}}_k^T\boldsymbol{\Lambda}\bar{\boldsymbol{u}}_{k'} - \sum_{k<k'}\sum_{l'\neq l}x_{i,(k,k')}^{(l')}(\boldsymbol{u}_k^{(l')})^T\boldsymbol{\Lambda}^{(l')}\boldsymbol{u}_{k'}^{(l')} - \sum_{k<k':k,k'\neq v}x_{i,(k,k')}^{(l)}\boldsymbol{u}_k^{(l)T}\boldsymbol{\Lambda}^{(l)}\boldsymbol{u}_{k'}^{(l)}$. Finally, $\boldsymbol{Z}_v^{(l)}$ is a $n \times (V-1)$ dimensional matrix with its ith row is given by $(x_{i,(1,v)}^{(l)}, ..., x_{i,(v-1,v)}^{(l)}, x_{i,(v,v+1)}^{(l)}, ..., x_{i,(v,V)}^{(l)})$.

- $\xi_v^{(l)}|- \sim Ber(\tilde{\pi}_v^{(l)})$, where

$$\tilde{\pi}_v^{(l)} = \frac{\pi_l N(\tilde{\boldsymbol{y}}_v^{(l)}|\boldsymbol{0}, \boldsymbol{Z}_v^{(l)}\boldsymbol{u}_{-v}^{(l)}\boldsymbol{\Lambda}^{(l)}\boldsymbol{M}^{(l)}\boldsymbol{\Lambda}^{(l)}\boldsymbol{u}_{-v}^{(l)T}\boldsymbol{Z}_v^{(l)T} + \tau^2\boldsymbol{I}_n)}{\pi_l N(\tilde{\boldsymbol{y}}_v^{(l)}|\boldsymbol{0}, \boldsymbol{Z}_v^{(l)}\boldsymbol{u}_{-v}^{(l)}\boldsymbol{\Lambda}^{(l)}\boldsymbol{M}^{(l)}\boldsymbol{\Lambda}^{(l)}\boldsymbol{u}_{-v}^{(l)T}\boldsymbol{Z}_v^{(l)T} + \tau^2\boldsymbol{I}_n) + (1-\pi_l)N(\tilde{\boldsymbol{y}}_v^{(l)}|\boldsymbol{0}, \tau^2\boldsymbol{I}_n)}.$$

- $\pi_l|- \sim Beta(a + \sum_{v=1}^V \xi_v^{(l)}, b + V - \sum_{v=1}^V \xi_v^{(l)})$, for $l = 1, ..., L$.

- $\boldsymbol{M}|- \sim IW\left(\nu + V, \boldsymbol{I}_R + \sum_{v=1}^V \bar{\boldsymbol{u}}_v\bar{\boldsymbol{u}}_v^T\right)$

- $\boldsymbol{M}^{(l)}|- \sim IW\left(\nu + \#\{v : \xi_v^{(l)} = 1\}, \boldsymbol{I}_R + \sum_{v:\xi_v^{(l)}=1} \boldsymbol{u}_v^{(l)}\boldsymbol{u}_v^{(l)T}\right)$, for $l = 1, ..., L$.

- $(\pi_{r,1}, \pi_{r,2}, \pi_{r,3})|- \sim Dir(1 + I(\lambda_r = 1), r^\eta + I(\lambda_r = 0), 1 + I(\lambda_r = -1))$, for $r = 1, .., R$, where $I(B)$ takes value 1 when B occurs and is 0 otherwise.

- $(\pi_{r,1}^{(l)}, \pi_{r,2}^{(l)}, \pi_{r,3}^{(l)})|- \sim Dir(1 + I(\lambda_r^{(l)} = 1), r^\eta + I(\lambda_r^{(l)} = 0), 1 + I(\lambda_r^{(l)} = -1))$, for $r = 1, .., R$, $l = 1, ..., L$.

- $$\lambda_r \mid - \sim \begin{cases} 1 & w.p.\ p_{r,1} \\ 0 & w.p.\ p_{r,2} \\ -1 & w.p.\ p_{r,3} \end{cases}$$

  where

  $p_{r,1} = \dfrac{\pi_{r,1} N(\boldsymbol{y}|\mu\boldsymbol{1}_n + \sum_{l=1}^L \boldsymbol{A}^{(l)}(\boldsymbol{\beta}^{(l)})^{(\lambda_r=1)}, \tau^2 I_Q)}{S}$,
  $p_{r,2} = \dfrac{\pi_{r,2} N(\boldsymbol{y}|\mu\boldsymbol{1}_n + \sum_{l=1}^L \boldsymbol{A}^{(l)}(\boldsymbol{\beta}^{(l)})^{(\lambda_r=0)}, \tau^2 I_Q)}{S}$,
  $p_{r,3} = \dfrac{\pi_{r,3} N(\boldsymbol{y}|\mu\boldsymbol{1}_n + \sum_{l=1}^L \boldsymbol{A}^{(l)}(\boldsymbol{\beta}^{(l)})^{(\lambda_r=-1)}, \tau^2 I_Q)}{S}$, where, $Q = V(V-1)/2$,
  $S = \sum_{s\in\{0,1,-1\}} \pi_{r,s} N(\boldsymbol{y}|\mu\boldsymbol{1}_n + \sum_{l=1}^L \boldsymbol{A}^{(l)}(\boldsymbol{\beta}^{(l)})^{(\lambda_r=s)}, \tau^2 I_Q)$, with $(\boldsymbol{\beta}^{(l)})^{(\lambda_r=s)}$ is $\boldsymbol{\beta}^{(l)}$ obtained by setting $\lambda_r = s$, for $r = 1, ..., R$; $s = 0, 1, -1$; $l = 1, ..., L$.

- $$\lambda_r^{(l)} \mid - \sim \begin{cases} 1 & w.p.\ p_{r,1}^{(l)} \\ 0 & w.p.\ p_{r,2}^{(l)} \\ -1 & w.p.\ p_{r,3}^{(l)} \end{cases}$$

  where

  $p_{r,1}^{(l)} = \dfrac{\pi_{r,1}^{(l)} N(\boldsymbol{y}|\mu \mathbf{1}_n + \boldsymbol{A}^{(l)}(\boldsymbol{\beta}^{(l)})^{(\lambda_r^{(l)}=1)} + \sum_{l' \neq l} \boldsymbol{A}^{(l')}(\boldsymbol{\beta}^{(l')}), \tau^2 I_Q)}{S},$

  $p_{r,2}^{(l)} = \dfrac{\pi_{r,2}^{(l)} N(\boldsymbol{y}|\mu \mathbf{1}_n + \boldsymbol{A}^{(l)}(\boldsymbol{\beta}^{(l)})^{(\lambda_r^{(l)}=0)} + \sum_{l' \neq l} \boldsymbol{A}^{(l')}(\boldsymbol{\beta}^{(l')}), \tau^2 I_Q)}{S},$

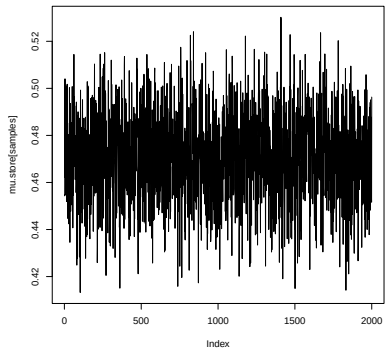
  $p_{r,3}^{(l)} = \dfrac{\pi_{r,3}^{(l)} N(\boldsymbol{y}|\mu \mathbf{1}_n + \boldsymbol{A}^{(l)}(\boldsymbol{\beta}^{(l)})^{(\lambda_r^{(l)}=-1)} + \sum_{l' \neq l} \boldsymbol{A}^{(l')}(\boldsymbol{\beta}^{(l')}), \tau^2 I_Q)}{S},$ where,

  $S = \sum_{s \in \{0,1,-1\}} \pi_{r,s}^{(l)} N(\boldsymbol{y}|\mu \mathbf{1}_n + \boldsymbol{A}^{(l)}(\boldsymbol{\beta}^{(l)})^{(\lambda_r^{(l)}=s)} + \sum_{l' \neq l} \boldsymbol{A}^{(l')} \boldsymbol{\beta}^{(l')}, \tau^2 I_Q)$

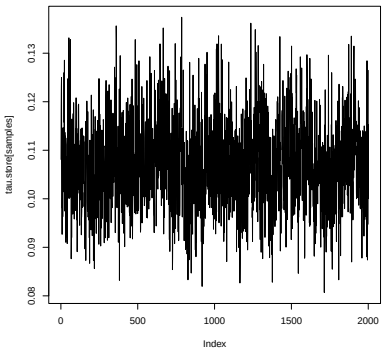
The posterior distribution is obtained by running a Markov Chain Monte Carlo (MCMC) with Gibbs sampler using the full conditional posterior distributions provided above.

Appendix B

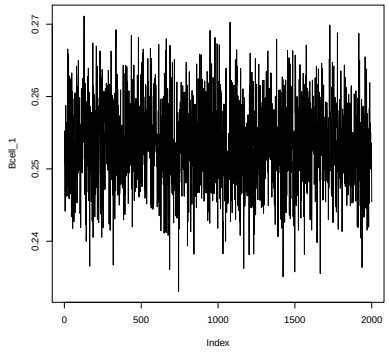
The following trace plots were produced using the simulation setting corresponding to Scenario 1, outlined in Section 4. For the simulation, the parameters were set as $\mu^* = 0.5$, $\tau^{*2} = 0.1$, $R^* = 3$, $1 - \pi_1^* = 0.8$, $1 - \pi_2^* = 0.6$, $1 - \pi_3^* = 0.7$ and $1 - \pi_4^* = 0.7$. To obtain the 2000 post burn-in samples, 15000 MCMC iterates were taken, the first 11000 were discarded as burn-in, and thinning of size 2 was done.



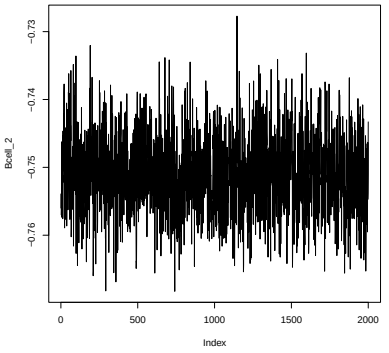
(a) Trace plot for μ



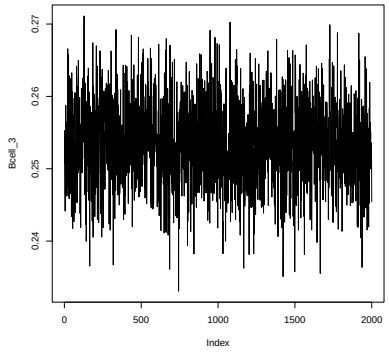
(b) Trace plot for τ^2



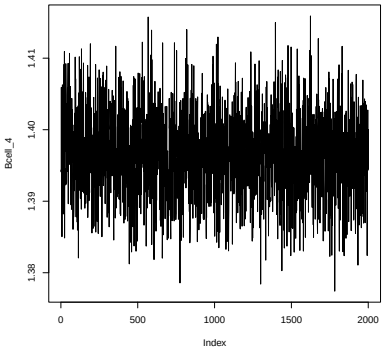
(c) Trace plot for an entry of $\boldsymbol{B}^{(1)}$



(d) Trace plot for an entry of $\boldsymbol{B}^{(2)}$



(e) Trace plot for an entry of $\boldsymbol{B}^{(3)}$



(f) Trace plot for an entry of $\boldsymbol{B}^{(4)}$

Figure 5: Trace plots for representative parameters.

References

T. Ahmed, H. Raja, and W. U. Bajwa. Tensor regression using low-rank and sparse tucker decompositions. *SIAM Journal on Mathematics of Data Science*, 2(4):944–966, 2020. doi: 10.1137/19M1299335.

E. M. Airoldi, D. M. Blei, S. E. Fienberg, and E. P. Xing. Mixed membership stochastic blockmodels. *Journal of Machine Learning Research*, 9(65):1981–2014, 2008.

J. Allaire and F. Chollet. *keras: R Interface to 'Keras'*, 2023. URL `https://tensorflow.rstudio.com/`. R package version 2.11.1.

J. Allaire and Y. Tang. *tensorflow: R Interface to 'TensorFlow'*, 2023. URL `https://github.com/rstudio/tensorflow`. R package version 2.11.0.9000.

E. Bullmore and O. Sporns. Complex brain networks: Graph theoretical analysis of structural and functional systems. *Nature Reviews Neuroscience*, 10(3):186–198, 2009. doi: 10.1038/nrn2575.

C. M. Carvalho, N. G. Polson, and J. G. Scott. The horseshoe estimator for sparse signals. *Biometrika*, 97(2):465–480, 2010. ISSN 00063444, 14643510.

M. Contisciani, E. A. Power, and C. De Bacco. Community detection with node attributes in multilayer networks. *Scientific Reports*, 10(1), 2020. doi: 10.1038/s41598-020-72626-y.

R. C. Craddock, P. E. Holtzheimer, X. P. Hu, and H. S. Mayberg. Disease state prediction from resting state functional connectivity. *Magnetic Resonance in Medicine*, 62(6): 1619–1628, 2009. doi: 10.1002/mrm.22159.

V. C. Dinh and L. S. Ho. Consistent feature selection for analytic deep neural networks. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in*

Neural Information Processing Systems, volume 33, pages 2420–2431. Curran Associates, Inc., 2020.

R. Dove, M. Winstead, H. Dunlap, M. Hause, A. Scalco, K. Willett, A. D. Williams, and B. Wilson. Democratizing systems security. In *INCOSE International Symposium*, volume 33, pages 86–97. Wiley Online Library, 2023.

D. Durante, N. Mukherjee, and R. C. Steorts. Bayesian learning of dynamic multilayer networks. *Journal of Machine Learning Research*, 18(43):1–29, 2017. URL `http://jmlr.org/papers/v18/16-391.html`.

J. Fan, W. Gong, and Z. Zhu. Generalized high-dimensional trace regression via nuclear norm regularization. *Journal of Econometrics*, 212(1):177–202, 2019. ISSN 0304-4076. doi: https://doi.org/10.1016/j.jeconom.2019.04.026. Big Data in Dynamic Predictive Econometric Modeling.

O. Frank and D. Strauss. Markov graphs. *Journal of the American Statistical Association*, 81(395):832–842, 1986. doi: 10.1080/01621459.1986.10478342.

J. Friedman, R. Tibshirani, and T. Hastie. Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1):1–22, 2010. doi: 10.18637/jss.v033.i01.

I. Gollini and T. B. Murphy. Joint modeling of multiple network views. *Journal of Computational and Graphical Statistics*, 25(1):246–265, 2016. ISSN 10618600, 15372715.

R. B. Gramacy, C. Moler, and B. A. Turlach. *monomvn: Estimation for MVN and Student-t Data with Monotone Missingness*, 2023. URL `https://CRAN.R-project.org/package=monomvn`. R package version 1.9-17.

S. Guha and A. Rodriguez. Bayesian regression with undirected network predictors with an

application to brain connectome data. *Journal of the American Statistical Association*, 116(534):581–593, 2021. doi: 10.1080/01621459.2020.1772079.

S. Guha and A. Rodriguez. High-Dimensional Bayesian Network Classification with Network Global-Local Shrinkage Priors. *Bayesian Analysis*, pages 1 – 30, 2023. doi: 10.1214/23-BA1378.

R. Guhaniyogi, S. Qamar, and D. B. Dunson. Bayesian tensor regression. *Journal of Machine Learning Research*, 18(79):1–31, 2017.

W. L. Hamilton. The graph neural network model. In *Graph Representation Learning*, pages 51–70. Springer, 2020.

Q. Han, K. S. Xu, and E. M. Airoldi. Consistent estimation of dynamic and multi-layer block models. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, page 1511–1520. JMLR.org, 2015.

L. He, K. Chen, W. Xu, J. Zhou, and F. Wang. Boosted sparse and low-rank tensor regression. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

M. T. Heaney. Multiplex networks and interest group influence reputation: An exponential random graph model. *Social Networks*, 36:66–81, 2014. ISSN 0378-8733. doi: https://doi.org/10.1016/j.socnet.2012.11.003. URL `https://www.sciencedirect.com/science/article/pii/S0378873312000664`. Special Issue on Political Networks.

P. D. Hoff. Multilinear tensor regression for longitudinal relational data. *The Annals of Applied Statistics*, 9(3), 2015. doi: 10.1214/15-aoas839.

P. D. Hoff, A. E. Raftery, and M. S. Handcock. Latent space approaches to social network analysis. *Journal of the American Statistical Association*, 97(460):1090–1098, 2002. doi: 10.1198/016214502388618906.

P. W. Holland and S. Leinhardt. An exponential family of probability distributions for directed graphs. *Journal of the American Statistical Association*, 76(373):33–50, 1981. doi: 10.1080/01621459.1981.10477598.

H. Ishwaran and J. S. Rao. Spike and slab variable selection: Frequentist and Bayesian strategies. *The Annals of Statistics*, 33(2):730 – 773, 2005. doi: 10.1214/009053604000001147.

T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

Y. Ma, S. Wang, C. C. Aggarwal, D. Yin, and J. Tang. Multi-dimensional graph convolutional networks. In *Proceedings of the 2019 siam international conference on data mining*, pages 657–665. SIAM, 2019.

K. Nowicki and T. A. B. Snijders. Estimation and prediction for stochastic blockstructures. *Journal of the American Statistical Association*, 96(455):1077–1087, 2001. doi: 10.1198/016214501753208735.

D. Osborn, M. J. Parks, R. A. Knudsen, K. Ross, C. Faucett, T. C. Haskin, P. C. Kitsos, T. G. Noel, and B. Cohn. Modeling for existing nuclear power plant security regime. Technical report, Sandia National Lab.(SNL-NM), Albuquerque, NM (United States), 2019.

T. Park and G. Casella. The bayesian lasso. *Journal of the American Statistical Association*, 103(482):681–686, 2008. doi: 10.1198/016214508000000337.

N. G. Polson and V. Ročková. Posterior concentration for sparse deep learning. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, page 938–949, Red Hook, NY, USA, 2018. Curran Associates Inc.

J. D. A. Relión, D. Kessler, E. Levina, and S. F. Taylor. Network classification with applications to brain connectomics. *The Annals of Applied Statistics*, 13(3):1648 – 1677, 2019. doi: 10.1214/19-AOAS1252.

J. Richiardi, H. Eryilmaz, S. Schwartz, P. Vuilleumier, and D. Van De Ville. Decoding brain states from fmri connectivity graphs. *NeuroImage*, 56(2):616–626, 2011. ISSN 1053-8119. doi: https://doi.org/10.1016/j.neuroimage.2010.05.081. Multivariate Decoding and Brain Reading.

J. G. Scott and J. O. Berger. Bayes and empirical-Bayes multiplicity adjustment in the variable-selection problem. *The Annals of Statistics*, 38(5):2587 – 2619, 2010. doi: 10.1214/10-AOS792. URL `https://doi.org/10.1214/10-AOS792`.

T. A. Snijders, A. Lomi, and V. J. Torló. A model for the multiplex dynamics of two-mode and one-mode networks, with an application to employment preference, friendship, and advice. *Social Networks*, 35(2):265–276, 2013. doi: 10.1016/j.socnet.2012.05.005.

D. Spencer, R. Guhaniyogi, R. Shinohara, and R. Prado. Bayesian tensor regression using the tucker decomposition for sparse spatial modeling, 2022.

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1):267–288, 1996. ISSN 00359246.

P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

A. D. Williams, G. C. Birch, S. Caskey, T. Gunda, J. Wingo, and T. Adams. A complex systems approach to develop a multilayer network model for high consequence facility

security. In *International Conference on Complex Systems*, pages 321–333. Springer, 2020.

A. D. Williams, G. C. Birch, S. A. Caskey, E. S. Fleming, T. Gunda, T. Adams, and J. Wingo. Insights for systems security engineering from multilayer network models. In *INCOSE International Symposium*, volume 31, pages 280–295. Wiley Online Library, 2021.

K. Xu, W. Hu, J. Leskovec, and S. Jegelka. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*, 2018.

S. Xu, Y. Zhen, and J. Wang. Covariate-assisted community detection in multi-layer networks. *Journal of Business & Economic Statistics*, 41(3):915–926, 2023. doi: 10.1080/07350015.2022.2085726.

X. Zhang, G. Xu, and J. Zhu. Joint latent space models for network data with high-dimensional node variables. *Biometrika*, 109(3):707–720, 2022. doi: 10.1093/biomet/asab063.

H. Zhou, L. Li, and H. Zhu. Tensor regression with applications in neuroimaging data analysis. *Journal of the American Statistical Association*, 108(502):540–552, 2013. doi: 10.1080/01621459.2013.776499. PMID: 24791032.