12-2022

Learning Graphical Models of Multivariate Functional Data with Applications to Neuroimaging

Jiajing Niu
jiajinn@g.clemson.edu

Learning Graphical Models of Multivariate Functional Data with Applications to Neuroimaging

---

A Dissertation
Presented to
the Graduate School of
Clemson University

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy
Statistics

---

by
Jiajing Niu
December 2022

Accepted by:
Dr. D. Andrew Brown, Committee Chair
Dr. Christopher McMahan
Dr. Xiaoqian Sun
Dr. Yu-Bo Wang

Abstract

This dissertation investigates the functional graphical models that infer the functional connectivity based on neuroimaging data, which is noisy, high dimensional and has limited samples. The dissertation provides two recipes to infer the functional graphical model: 1) a fully Bayesian framework 2) an end-to-end deep model.

We first propose a fully Bayesian regularization scheme to estimate functional graphical models. We consider a direct Bayesian analog of the functional graphical lasso proposed by Qiao et al. (2019). We then propose a regularization strategy via the graphical horseshoe. We compare both Bayesian approaches to the frequentist functional graphical lasso, and compare the Bayesian functional graphical lasso to the functional graphical horseshoe. We applied the proposed methods with electroencephalography (EEG) data and diffusion tensor imaging (DTI) data. We find that the Bayesian methods tend to outperform the standard functional graphical lasso, and that the functional graphical horseshoe performs best overall, a procedure for which there is no direct frequentist analog.

Then we consider a deep neural network architecture to estimate functional graphical models, by combining two simple off-the-shelf algorithms: adaptive functional principal components analysis (FPCA) (Yao et al., 2021a) and convolutional graph estimator (Belilovsky et al., 2016). We train our proposed model with synthetic data which emulate the real world observations and prior knowledge. Based on synthetic data generation process, our model convert an inference problem as a supervised learning problem. Compared with other framework, our proposed deep model which offers a general recipe to infer the functional graphical model based on data-driven approach, take the raw functional dataset as input and avoid deriving sophisticated closed-form. Through simulation studies, we find that our deep functional graph model trained on synthetic data generalizes well and outperform other popular baselines marginally. In addition, we apply deep functional graphical model in the real world EEG data, and our proposed model discover meaningful brain connectivity.

Finally, we are interested in estimating casual graph with functional input. In order to process functional covariates in causal estimation, we leverage the similar strategy as our deep functional graphical model. We extend popular deep causal models to infer causal effects with functional confoundings within the potential outcomes framework. Our method is simple yet effective, where we validate our proposed architecture in variety of simulation settings. Our work offers an alternative way to do causal inference with functional data.

Dedication

This dissertation is dedicated to my husband, Sufeng, who has been a source of energy and encouragement during my challenging study and life. I am so grateful for having you in my life. This dissertation is also dedicated to my little girl, Noreen, twenty-one months, grows quicker than imagining, always give me strength to finish that I have started and work hard for the things that I aspire to achieve. And to my dog, Dollar, who is always with me.

Acknowledgments

First I would like to thank my PhD advisor Dr. D. Andrew Brown who helped me to complete my PhD journey at Clemson University. He gave me high degree of freedom to explore my research interests, and patiently polished my paper writing over and over. I also would like to thank my committee members for reviewing my thesis work. In addition, many thanks to Prisma Health for their generous funding support, and my mentor Dr. Moonseong Heo for much help and advice. Special thanks to Palmetto cluster, which provide computational resources and IT support.

Finally I would like to thank my husband, who sacrificed his plenty of time to take care of my daughter, tolerating my frustration and never giving up on me. I also would like to thank to my daughter, who gave me strength and faith to complete my PhD. Thanks to all of you.

Table of Contents

Table

Figure

Chapter 1

Introduction

In recent decades, data is explosively emerging thanks to the development of big data industries. Analyzing big data becomes critical since many of data are high-dimensional, high volumes, highly noisy and time-sensitive. Applications such as biological science, public health, social science and medical science, etc. share the common interests at processing observation measures over time, and thereby has introduced variety of machine learning and statistical modelling. A typical time series analysis that takes time series data as multivariate data has a few assumptions, such as stationary over time, evenly spaced measurements, etc. However, time series methodologies ignore the essential information of the smooth function underneath the generating process of the data. In addition, time series methodologies may erroneously assume independence among data points after removing the trends and seasonality patterns. Functional data analysis (FDA) is a branch of statistics to handle the data on a finite set of points as a function of continuous variable possibly with noise. The data may or may not be a time series. FDA consider discrete measurements from a function instead of multivariate data in time series analysis, and the whole function is viewed as as a single observation. Similarly, we could conduct statistical modelling or prediction upon the functional data.

In order to study the functional data analysis, one of the most typical FDA applications is the modern functional neuroimaging, such as functional magnetic resonance imaging (fMRI; Shappell et al., 2019), electroencephalography (EEG; Zhang et al., 1995) , magnetoencephalography (MEG; Hämäläinen et al., 1993) and positron emission tomography (PET; Bailey et al., 2005). These techniques collect the measurements of brain activity over time in the experiment, and such mea-

surements could be considered as functional data. With more developed data collection, methods for dealing with these functional data are in high demand.

Among functional data analysis for neuroimaging, one particular interesting problem in neuroscience is to study the graph structure for brain connectivity. Based on multivariate functional data collected from brain processes under different experiments, a reliable statistical tool is required to understand and estimate the brain connectivity. Specifically, our design goal can be summarized as the following:

- **Accurate Estimation**: We aim to infer the brain connectivity as close as the ground truth so that it could reflect the scientific findings.
- **Interpretability**: Our design should balance between model interpretability and accuracy, since in neuroscience, researchers sometimes not only want to know the inferred brain connectivity, but also prefer to understand *why*.
- **Model Confidence**: In many cases, we prefer to provide estimated graph with uncertainty measurements, so that the neuroimaging researcher is able to judge accordingly.
- **Causal Effects**: Beyond graph topology inference, can we conduct causal inference based on functional data?

Nevertheless, the research for graphical structure learning from multivariate functional data is underdeveloped before Zhu et al. (2016). Qiao et al. (2019) proposed a functional Gaussian graphical models which demonstrate a reasonable brain connectivity estimation, but their methods lack of measures of uncertainty about the connectivity. On the other hand, existing work uses the basis transformation to represent the functional data, which is a feature extraction that can often lose important information. Therefore, we explore two paths of functional graphical models: 1) Bayesian and 2) deep learning frameworks. Our proposed Bayesian framework is able to provide an interpretable graph structure estimation with edge confidence, whereas deep learning based approach could achieve the most accurate inference with one unified framework. Besides, we are also interested in answering causal effects with functional data since causal inference with functional data domain lack the attention from research community. For example, *are the ordinary causal models suitable for functional data for treatment effect estimation*? In the thesis, we explore a few popular causal models with functional confounders by taking advantages of deep learning framework for

causal estimation.

For the rest of introduction chapter, we introduce basic definitions and background for our works in this thesis. We first provide a brief, intuitive primer on the Gaussian graphical model and the algorithm graphical lasso. Then we review the dimension reduction method FPCA, which is a fundamental technique for infinitely dimensional functional data. To motivate our proposals, we then discuss functional (Gaussian) graphical models proposed by Qiao et al. (2019), an undirected graphical model to handle multivariate functional data. In addition, the empirical risk minimization is reviewed to present the principle behind deep functional graphical models in Section 2.3. At the end of this chapter, the structure and contribution of this dissertation is provided.

1.1 Gaussian Graphical Models and the Graphical Lasso

We first introduce the Gaussian Graphical Model (GGM) and a specific algorithm Graphical Lasso, which is used to handle multivariate data. Suppose the random vector $\boldsymbol{y} = (y_1, \cdots, y_p)^T$ follows a multivariate Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$. Then we define $\boldsymbol{\Theta} = \boldsymbol{\Sigma}^{-1}$ as the *precision matrix*, or concentration matrix. A GGM is based on an undirected graph $G = (V, E)$, where $V = \{1, \ldots, p\}$ is a non-empty set of vertices and $E \subseteq \{(i, j), i < j\}$ is a set of edges representing unordered pairs of vertices (also called nodes). Each variable y_i represents a node in the graph, and E determines the precision matrix, for $i \neq j$, $(\boldsymbol{\Theta})_{ij} \neq 0$ if and only if $(i, j) \in E$. By Theorem 2.2 in Rue and Held (2005), we thus have that E encodes a Markov property in the distribution. Letting $\mathcal{N}(i) = \{j : (i, j) \in E\}$ and adopting the convention that $\boldsymbol{y}_{\mathcal{A}} = (y_j : j \in \mathcal{A})^T$ for a set of indices $\mathcal{A}$, we have that, for any node i, $y_i | \boldsymbol{y}_{(-i)} \stackrel{d}{=} y_i | \boldsymbol{y}_{\mathcal{N}(i)}$. Extending this with $V = \{1, \ldots, p\}$, it follows that $y_u \perp\!\!\!\perp y_v | \mathbf{y}_{V \setminus \{u,v\}}$ if and only if $(\boldsymbol{\Theta})_{uv} = 0$, where $\perp\!\!\!\perp$ denotes statistical independence. This is the *pairwise Markov property.* By this property, learning the graph associated with a Gaussian graphical model is equivalent to estimating the precision matrix of the multivariate Gaussian distribution, making it a covariance estimation problem (Dempster, 1972).

Given a sample $\boldsymbol{y}_i,\ i = 1, \ldots, n$, stored in a data matrix $\boldsymbol{Y} = (\boldsymbol{y}_1 \cdots \boldsymbol{y}_n)^T$, the goal is to estimate and select non-zero elements of $\boldsymbol{\Theta}$, thereby obtaining an estimate of the undirected graph associated with the GGM. The log-likelihood of $\boldsymbol{\Theta}$ (up to an additive constant) can be written as

$$l(\boldsymbol{\Theta}) = \log \det \boldsymbol{\Theta} - \operatorname{tr}(\mathbf{S}\boldsymbol{\Theta}/n), \tag{1.1}$$

where $\mathbf{S} = \mathbf{Y}^T\mathbf{Y}$. The quantity $-l(\mathbf{\Theta})$ is a convex function of $\mathbf{\Theta}$ and the maximum likelihood estimator of $\mathbf{\Sigma}$ is $\widehat{\mathbf{\Sigma}} = \mathbf{S}/n$. This estimator enjoys nice properties such as consistency, but can be unstable when $p \approx n$. Further, even when $\widehat{\mathbf{\Sigma}}^{-1}$ exists, it can be an unsatisfactory estimator of $\mathbf{\Theta}$ due to the fact that it will generally not be sparse, even if $\widehat{\mathbf{\Sigma}}$ is sparse.

To find a more stable estimator of $\mathbf{\Theta}$ that is simultaneously sparse, Yuan and Lin (2007) proposed to solve a lasso-type regularized version of the likelihood objective function by finding

$$\widehat{\mathbf{\Theta}} = \underset{\mathbf{\Theta} \in M^+}{\arg\min} \quad \{-\log\det\mathbf{\Theta} + \operatorname{tr}(\mathbf{S}\mathbf{\Theta}/n) + \lambda\|\mathbf{\Theta}\|_1\}, \tag{1.2}$$

where M^+ is the space of $p \times p$ symmetric positive definite matrices, the norm $\|\cdot\|_1$ is the sum of the absolute values of the off-diagonal elements, and λ is a non-negative tuning parameter to control the number of zeros in the estimated precision matrix. This is a semi-definite programming problem for the precision matrix $\mathbf{\Theta}$. Yuan and Lin (2007) solved this problem with the so-called maxdet algorithm (Vandenberghe et al., 1998), while Friedman, Hastie, and Tibshirani (2008) proposed a more efficient coordinate descent algorithm for solving (1.2). This is the *graphical lasso*, an approach that has since become very popular for structure learning in GGMs.

Wang (2012) considered the fully Bayesian version of the graphical lasso by recognizing that solving (1.2) is equivalent to finding the maximum a posteriori (MAP) estimator in the following model,

$$p(\boldsymbol{y}_i \mid \mathbf{\Theta}) = N(\boldsymbol{y}_i \mid \mathbf{0},\ \mathbf{\Theta}^{-1}), \quad i = 1, \ldots, n \tag{1.3}$$

$$p(\mathbf{\Theta} \mid \lambda) \propto \prod_{i<j} DE(\theta_{ij} \mid \lambda) \prod_{i=1}^{p} Exp\left(\theta_{ii} \mid \frac{\lambda}{2}\right), \quad \mathbf{\Theta} \in M^+, \tag{1.4}$$

where $N(\cdot|\mathbf{0}, \mathbf{\Theta}^{-1})$ denotes the density of a $N(\mathbf{0}, \mathbf{\Theta}^{-1})$ distribution, and likewise for the double exponential (DE) and exponential (Exp) distributions. Using a hierarchical representation of this model (Kyung et al., 2010) and matrix partitioning techniques similar to those employed by Banerjee et al. (2008) and Friedman et al. (2008), Wang (2012) developed an efficient Gibbs sampler for exploring the full posterior distribution and thus was able to extensively compare the results of the MAP and posterior mean estimators.

1.2 Functional Principal Component Analysis

To allow a graphical model to take functional data as input, the most common technique is functional principal component analysis. For subject i, $i = 1, \ldots, n$, let the underlying, infinite dimensional function of interest be denoted $g_i(t)$, $t \in \mathcal{T}$. We assume that $g_1, \ldots, g_n$ are identically distributed and independent zero-mean functions in $L^2(\mathcal{T})$ with covariance function $\text{cov}(g_j(s), g_j(t)) =: \Sigma(s,t)$, $(s,t) \in \mathcal{T} \times \mathcal{T}$, where $\mathcal{T}$ is a compact interval on the real line. Karhunen (1946) and Loeve (1963) independently discovered the functional principal component analysis (FPCA) expansion (e.g., Bosq, 2012),

$$g_i(t) = \sum_{k=1}^{\infty} a_{ik}\phi_k(t), \tag{1.5}$$

where $\{\phi_k(t)\}_{k=1}^{\infty}$ are the orthonormal set of eigenfunctions with corresponding eigenvalues $\{\lambda_k\}_{k=1}^{\infty}$ satisfying $\Sigma(s,t) = \sum_{k=1}^{\infty} \lambda_k \phi_k(s)\phi_k(t)$, by Mercer's Theorem, and $a_{ik} = \int g_i(t)\phi_k(t)dt$ are the functional principal component (FPC) scores of g_i, uncorrelated across k with $E(a_{ik}) = 0$ and $\text{var}(a_{ik}) = \lambda_k$. By assumption, the a_{ik} are independent across i. Like ordinary principal components analysis (e.g. Jolliffe, 2002), the expansion can be truncated to obtain a finite-dimensional approximation to the infinite-dimensional process. In what follows, the proposed Bayesian functional graphical models in Chapter 2.2 can work with any basis expansion (e.g., wavelets or Fourier), but we use FPCA in simulation and application due to the mean square optimality of the truncated approximation.

Performing FPCA in practice amounts to finding the spectral decomposition of an approximation to the covariance function. When g_i, $i = 1, \ldots, n$, are observed on the same evenly spaced grid $t_1, \ldots, t_m$ independent of i, this amounts to standard singular value decomposition of the sample covariance matrix. For irregularly spaced functions and/or different numbers of observations on each function, SVD will likely provide a poor approximation to the true eigensystem associated with $\Sigma(\cdot,\cdot)$. In this case, Yao et al. (2005) proposed the PACE algorithm for performing FPCA via conditional expectation. In our applications of section 4, to apply our method of Bayesian models we use SVD for the EEG example and PACE for the diffusion MRI example, as the latter involves irregularly sampled longitudinal data.

1.3 Functional Gaussian Graphical Models

For a particular subject i, suppose we (discretely) observe p functions $g_{i1}(t), \ldots, g_{ip}(t)$ where g_{ij} is the function observed on node j. Suppose further that each function is a Gaussian process so that $(g_{i1}, \ldots, g_{ip})$ is a realization from a $p-$dimensional multivariate Gaussian process (MGP). As in typical GGMs, we associate to the MGP an undirected graph $G = (V, E)$ that represents the conditional dependence network. Here, conditional dependence of the functions g_{ij} and $g_{ij'}$ is in terms of the cross-covariance function,

$$C_{jj'}(s,t) = \operatorname{cov}\left(g_{ij}(s), g_{ij'}(t) \mid g_k(\cdot), k \neq j, j'\right), \tag{1.6}$$

assumed to be the same for $i = 1, \ldots, n$.

With the covariance function in hand, we can use FPCA and approximate each g_{ij} with the M-dimensional truncation,

$$g_{ij}^M(t) = \sum_{k=1}^{M} a_{ijk}\phi_{jk}(t), \quad M < \infty. \tag{1.7}$$

The superscript M here is used to denote the parameter is dependent on M, but we may omit the superscripts for simplicity hereafter when the content is clear. The function for subject i at node j can thus be represented with the coefficient vector $\boldsymbol{a}_{ij}^M = (a_{ij1}, \ldots, a_{ijM})^T$, so that each subject's entire functional information over all p nodes is encoded in $\boldsymbol{a}_i^M = ((\boldsymbol{a}_{i1}^M)^T, \ldots, (\boldsymbol{a}_{ip}^M)^T)^T \in \mathbb{R}^{Mp}$, where for notational simplicity we suppose the truncation level M is the same at each node. Under the Gaussian assumption and independently observed subjects, the Kahrunen-Loéve Theorem tells us that $\boldsymbol{a}_i^M \overset{\text{iid}}{\sim} N_{Mp}(\mathbf{0}, (\boldsymbol{\Theta}^M)^{-1})$. For learning the graphical model, Qiao et al. (2019, Lemma 1) show that, in the finite-rank case (in which the M-truncated approximation is exact),

$$E^M = \left\{(i,j) : \|\boldsymbol{\Theta}_{ij}^M\|_F \neq 0, (i,j) \in V^2, i \neq j\right\}, \tag{1.8}$$

where $\boldsymbol{\Theta}_{ij}^M$ is the $M \times M$ block submatrix of $\boldsymbol{\Theta}^M$ corresponding to the node pair $(i,j) \in V \times V$ and $\|\cdot\|_F$ is the Frobenius norm. Thus, structure learning in the functional graphical model is equivalent to finding the (i,j) pairs for which $\|\boldsymbol{\Theta}_{ij}^M\|_F \neq 0$.

The connection in (1.8) to the graphical lasso and the group lasso (Yuan and Lin, 2006) led

Qiao et al. (2019) to propose estimating the graph from functional data with

$$\hat{\boldsymbol{\Theta}}^M = \underset{\boldsymbol{\Theta}^M}{\arg\max} \left\{ \log\det \boldsymbol{\Theta}^M - \operatorname{tr}\left(\boldsymbol{S}^M \boldsymbol{\Theta}^M\right) - \lambda_n \sum_{i \neq j} \|\boldsymbol{\Theta}_{ij}^M\|_F \right\} \tag{1.9}$$

where $\boldsymbol{S}^M$ is the sample covariance matrix computed from estimated FPC scores $\widehat{\boldsymbol{a}}_i^M \in \mathbb{R}^{Mp}$, found via SVD or otherwise, and $\lambda_n > 0$ is a tuning parameter. As with the group lasso, blockwise sparsity is achieved as $\lambda_n \to \infty$. Qiao et al. term this approach the *functional graphical lasso* (fglasso). The edge set of the estimated graph is then $\hat{E}^M = \left\{(i,j) : \|\hat{\boldsymbol{\Theta}}_{ij}^M\|_F \neq 0, (i,j) \in V^2, i \neq j\right\}$. Rather than using identical truncated number M across $j = 1, \ldots, p$, one can select M_j separate for each j, as different functional variables may have different smoothness levels. Qiao et al. (2019) show that the fglasso enjoys model selection consistency, and provide a block coordinate descent algorithm for optimizing the objective function.

1.4 Empirical Risk Minimization Principles

Empirical Risk Minimization (ERM) is the foundation of deep model training criteria. It is a general framework applied in many methods for supervised learning such as classification and regression. We aim to select a function f among possible set of functions $\mathcal{F}$ to minimize the risk. Consider input space $\mathcal{X}$ and output space $\mathcal{Y}$, the problem of interest is to learn the function between $\mathcal{X}$ and $\mathcal{Y}$:

$$f : \mathcal{X} \to \mathcal{Y} \tag{1.10}$$

where the estimator f could be called regressor or classifier in different tasks. Assume data $\mathcal{X} \times \mathcal{Y}$ and (x_1, y_1), $\ldots$, (x_n, y_n) are sampled i.i.d. from a joint distribution $P(x, y)$. In the statistical learning, our goal is to pick a f^* so that it could minimize the average loss (risk), i.e.

$$f^* = \underset{f \in \mathcal{F}}{\arg\min}\, E_p(L(f(x), y)). \tag{1.11}$$

where loss function $L(\cdot, \cdot)$ is a continuous function which measures the performance of f. An example of loss function is the binary cross entropy loss for binary classification. The best function f^* is the

one with smallest true risk, i.e. the Bayes classifier

$$f^* = I\left[\frac{P(x,y)}{P(x)} > 0.5\right] \tag{1.12}$$

where the $P(x,y)$ is defined as the joint distribution of X and Y, and $P(x)$ is the marginal distribution over X.

In general, the joint distribution P is unknown and estimating P from data is often computationally intractable. Even though Bayes posterior is often hard to directly calculate, the data are sampled from P, and thereby we could get a reasonable approximation through empirical risk minimization

$$f = \arg\min_{f \in \mathcal{F}} \frac{1}{n} \sum_i L(f(x_i), y_i). \tag{1.13}$$

How close the empirical risk is to the (true) risk depends on following conditions. First, the more training data we could get from P, the closer we would expect empirical risk to true risk. Additionally, the complicated joint distribution P would lead to a poor approximation which requires more data. Furthermore, a large set of possible functions $\mathcal{F}$ or complex f would make the approximation challenging. In addition, the appropriate loss function L is also critical to improve the similarity between empirical risk and true risk (Vapnik, 1999).

It is straightforward to pick a simple function among a small set of possible functions to achieve better performance, but it may increase the minimum value of true risk. This is a variance-bias trade-off issue. Therefore, the typical empirical risk consists of a loss term and also a regularization term, where the loss function is used to penalize the training error and the regularization term is to penalize the function complexity.

1.5 Structure and Contribution of the Thesis

The work presented in this thesis focuses on graphical structure learning on multivariate functional data. In the first part of the thesis we develop a fully Bayesian framework of functional graphical models based on functional graphical lasso prior and horseshoe prior. On the other hand, we develop a deep learning architecture which can infer graph topology based on adaptive FPCA and data-driven framework. Finally, we provide deep learning based model for causal estimation with functional confoundings within the potential outcome framework.

In Chapter 2, we first introduce the background of our proposals, functional graphical models and deep neural networks, respectively. We then propose a fully Bayesian regularization scheme for estimating functional graphical models based on functional lasso and Horseshoe prior. We provide efficient Gibbs sampling algorithms for both of our proposed models, exploiting auxiliary variables to produce a set of easily-sampled conditional distributions. In addition, we propose a neural architecture for end-to-end functional graphical model that consist of adaptive basis layer and convolutional layer, i.e. *Deep Functional Graphical Model* (Deep FGM). We improve existing functional graphical model architectures by integrating a trainable basis that is implemented through neural networks instead of manually tuned basis transformation used in classic functional data analysis, and the covariance matrix of basis coefficients are further processed by convolutional neural networks, where we backprogagate the gradients obtained by comparing prediction graph and target graph.

In Chapter 3, we first introduce fundamental concepts of causal inference and one of the most popular frameworks: potential outcome model. We then explore a few variants of causal models which under the umbrella of potential outcome model to estimate causal effects and predict heterogeneous treatment effects. Then we dive deep into three models, S-learner, T-learner, and TARNet combined with adaptive FPCA, showing that this simple combination could allow causal inference to be applied with functional data domain.

In Chapter 4, we present numerical experiments based on proposed methods for estimating functional graphical models and causal models. To validate our hypothesis, we design a variety of experiments to compare Bayesian FGM with baselines. To validate prior assumption, we compare Bayesian fglasso and Bayesian functional graphical Horseshoe. We also compare deep model performance with the frequentest and Bayesian models. Lastly, we validate the deep causal models through a series of settings which are common in real world.

Applications to neuroimaging such as EEG and DTI data are represented in Chapter 5. Lastly, we conclude our thesis in Chapter 6, which summarizes the thesis and propose the future work.

Chapter 2

Estimation of Functional Graphical Models

In this Chapter, we discuss two methods to estimate functional graphical model: one method falls into a fully Bayesian regularization framework, and the other one leverages deep model techniques. In the Bayesian framework, we first consider a direct Bayesian analog of the functional graphical lasso proposed by Qiao et al. (2019), then extend the regularization strategy via the graphical horseshoe. In the framework of using deep model, we consider the deep graphical model proposed by Belilovsky et al. (2016). Based on limitations of Belilovsky et al. (2016), we propose to integrate adaptive basis layer proposed from Yao et al. (2021a). By combining two simple off-the-shelf algorithms, we could take functional data directly as input and embed the adaptive basis into the model and achieve end-to-end fashion of training.

This chapter is organized as follows: *preliminaries* give a brief literature reviews of functional graphical models and compare state-of-the-art methodologies for graphical models with functional inputs; *Bayesian functional graphical models* demonstrate our proposed approach, i.e. Bayesian fglasso and functional graphical horseshoe; *Deep functional graphical models* presents a neural network architecture for functional graphical model estimation.

2.1 Preliminaries

In this section, we review the background of our proposed methods, i.e. graphical models and two research lines for graph learning from multivariate data, and then we focus on introducing development of functional graphical models and the motivation for Bayesian and deep functional graphical models.

2.1.1 Graphical Models and Functional Graphical Models

Graphical models use graphs to model and draw inferences concerning conditional independence among a collection of random variables or processes, each of which is associated with a particular location (also called a node or a vertex). They have been used to study flow cytometry between cell proteins (Friedman et al., 2008), to estimate networks from gene expression data (Li et al., 2019), and to identify communicating regions from electroencephalography (EEG) data (Qiao et al., 2019). In this work we are focused on *Gaussian* graphical models, where the data follow a multivariate Gaussian distribution. In this case estimating the edge set is equivalent to identifying the nonzero elements of the precision matrix associated with the Gaussian distribution.

Broadly speaking, frequentist studies of graphical models have either involved neighborhood selection (Meinshausen et al., 2006) or the graphical lasso (Yuan and Lin, 2007; Friedman et al., 2008). The neighborhood selection method employs regression of each variable on the remaining variables with regularization, and then summarizing the neighborhoods together. Cai et al. (2011) proposed constrained L-1 minimization for sparse precision matrix, which is a Dantzig-type variant of the neighborhood approach. For extension of this research line, see Peng et al. (2009); Cai et al. (2016); Qiu et al. (2016); Sun and Zhang (2013). On the other hand, Friedman et al. (2008) proposed the graphical lasso via a Gaussian log-likelihood with the lasso regularization on the entire precision matrix. The glasso has proven to be useful and is a widely used procedure, due to the sparsity and convergence rates that have been studied (Lam and Fan, 2009) as well as associated computational techniques (Friedman et al., 2008; Zhu et al., 2014). A Bayesian version of the graphical lasso was proposed by Wang (2012), who illustrated potential differences between the posterior mean and the posterior mode that might be encountered. Li et al. (2019) extended the ideas of Wang (2012) by proposing a graphical horseshoe estimator, along with an efficient Markov chain Monte Carlo (MCMC; Gelfand and Smith, 1990) algorithm for its implementation.

To date, most of the graphical modeling literature has focused on data in which each node has an associated scalar or vector-valued response variable. However, many real world applications involve the collection of functional data at each node. In this case, we have a collection of subjects for whom a set of continuously-supported random functions are (discretely) observed, one function at each node, where the support may be time- or spatially-indexed, or both. For example, in neuroscience there is much interest in studying connectivity; e.g., in terms of connected regions of interest measured in functional magnetic resonance imaging (fMRI; Shappell et al., 2019) or communicating electrodes in electroencephalography (EEG; Zhang et al., 1995) corresponding to associated regions of neuronal activity. Alternatively, in social network analysis and marketing, it is possible to observe and record online behavior patterns among baskets of different goods for each customer over a period of time to identify related types of products. Compared to scalar or vector-valued graphical models, functional graphical models remain vastly under-explored. Qiao et al. (2019) proposed a functional version of the graphical lasso along with a block-coordinate descent algorithm for optimizing the loss function. Qiao et al. (2020) model such data using doubly functional graphical model through a nonparametric approach to smooth p covariance matrices, where the graph is functional in nature. Based on the setting of Qiao et al. (2019), Zapata et al. (2019) decomposed a functional graphical model into a sequence of standard multivariate graphical models under an assumption with partial separability for multivariate functional data. Li and Solea (2018) proposed a nonparametric functional graphical model based on conditional dependence by constructing additively nested Hilbert spaces and additive precision operator. In addition, Solea and Li (2020) relaxed the multivariate Gaussian process assumption by introducing the functional copula Gaussian graphical model. Around the same time, Zhu et al. (2016) proposed a Bayesian framework for working with functional graphical models directly in the space of infinite-dimensional random functions, essentially extending the work of Dawid and Lauritzen (1993) to function space by using hyper-inverse Wishart priors to a priori model the space of plausible, decomposable graphs. Recently, Zhang et al. (2021a) proposed a Bayesian model for functional graphical models in which independent Laplace priors are placed on reparameterized partial correlations associated with basis coefficients, inducing a so-called normal hypo-exponential shrinkage prior and allowing the graph to functionally evolve over time. They utilize basis function representations that models the within-functional correlations, then employ Bayesian regularization in the basis space assuming independence of basis coefficients across different nodes. Compared with Zhang et al. (2021a), our proposed method is

for a overall static graph with assumption that different basis coefficients could be dependent across nodes. To the best of our knowledge, Zhu et al. (2016) and Zhang et al. (2021a) are the only works on Bayesian functional graphical models.

Our work in section 2.2 is motivated by neuroimaging data that typically have low signal-to-noise ratios caused by non-neural noise arising from cardiac and respiratory processes or scanner instability, a problem that is exacerbated by the typically small numbers of subjects available from such studies. For instance, in Section 5.2 we study the effects of traumatic brain injury on connectivity of the human brain. The diffusion-weighted magnetic resonance imaging data consist of longitudinal measurements of white matter integrity within 26 regions of interest in 34 subjects, 17 of whom have been diagnosed with a traumatic brain injury (TBI). We aim to assess chronic structural connectivity differences between the TBI and non-TBI groups using the sparse, irregularly-measured longitudinal data — a goal for which few techniques currently exist. Further, quantifying model uncertainty is important. For example, Greenlaw et al. (2017) used an imaging genetics example to demonstrate dramatic differences in associations between genetic variations and brain imaging measures that might be identified when accounting for uncertainty in a model estimate versus using an optimization-based point estimate alone.

2.1.2 Deep Learning On Multivariate Functional Data

Deep neural network, which has been successfully and widely adopted in both academic and industrial, is originally inspired from biological neuron in human brain. The oldest neural network is simulated with electronic circuits by Warren McCulloch and Walter Pitts to show how neuron works in human brains in 1943 (McCulloch and Pitts, 1943). The application based on deep neural network is emerging happens after 2010 thanks to the computational power of GPU and large number of data produced from internet. The deep neural network architecture has experienced multiple evolution, such as auto-differentiable infrastructure development (Abadi et al., 2015), novel neural network architectures, novel data representation (graph neural network from Kipf and Welling (2016), language model from Devlin et al. (2018), etc.), and so on. Based on large amount of innovations and research work, modern neural network has become much deeper, and can almost convert almost any kind of the data format into dense vector, which is called "embedding" within deep neural network community, and thereby deep neural network is also named as "deep learning" (LeCun et al., 2015). During last decade, more and more types of neural network architectures

are proposed and specifically developed for different applications in variety of domains. Besides the most common multi-layer proceptron, convolutional neural networks (Krizhevsky et al., 2012) are further developed and used in computer vision, speech recognition (Abdel-Hamid et al., 2014); For sequential data with temporal dynamic behaviors like time series and natural languages, the recurrent neural networks (Rumelhart et al., 1985) are widely used, and long-short term memory (LSTM) (Hochreiter and Schmidhuber, 1997) with gate units is developed to solve the long-term dependence issue and avoid gradient vanishing problem; latest Transformers (Vaswani et al., 2017) which leverage self-attention to weight the importance of the input data has even shown great success among natural language processing (Devlin et al., 2018), computer vision (Dosovitskiy et al., 2020), and even optimal control (Chen et al., 2021); Deep models also dominate the generative models, such as variation auto-encoder (Kingma and Welling, 2013), generative adversarial network (GAN) (Goodfellow et al., 2014), and diffusion model (Reddy et al., 2021), which can generate high dimensional, complex data without any supervision.

Nevertheless, applying deep neural networks into functional data domain is emerging in the recent years. Wang et al. (2021) propose to use multi-layer perceptron to estimate the mean function of functional data and perform non-parametric regression. In the *Deep-FDA* proposed by Perdices et al. (2021), they represent functional data by embedding through autoencoder neural network for clustering task. Thind et al. (2022) propose *functional neural network* taking functional inputs and covariates through multi-layer perceptrons with functional weighting based on basis expansion. Wang and Cao (2022) utilized multi-layer perceptron to consume the multi-dimensional functional data to de-noise and recover 2D signal or 3D neuroimaging data. Yao et al. (2021a) projected functional data into embedding through adaptive basis function learned by micro-network in neural network. According to our best knowledge, estimating functional graphical models based on deep neural network gets limited attentions. Therefore, one goal of this thesis aim to investigate how to leverage deep neural network in estimating functional graphical model.

On the other hand, we review and list the state-of-art methods for graphical models with functional inputs as shown in Figure 2.1. The top panel in the Figure 2.1 illustrates the conventional workflow used in Zhu et al. (2016); Qiao et al. (2019); Niu et al. (2021). First, they project functional data into finite dimensional space through orthogonal basis functions (say of order M) to get the score A with dimension pM, where p denotes the number of nodes. Then after transforming to covariance matrix $A^T A$ with dimension $pM \times pM$, they use method, like frequentest fglasso (Qiao

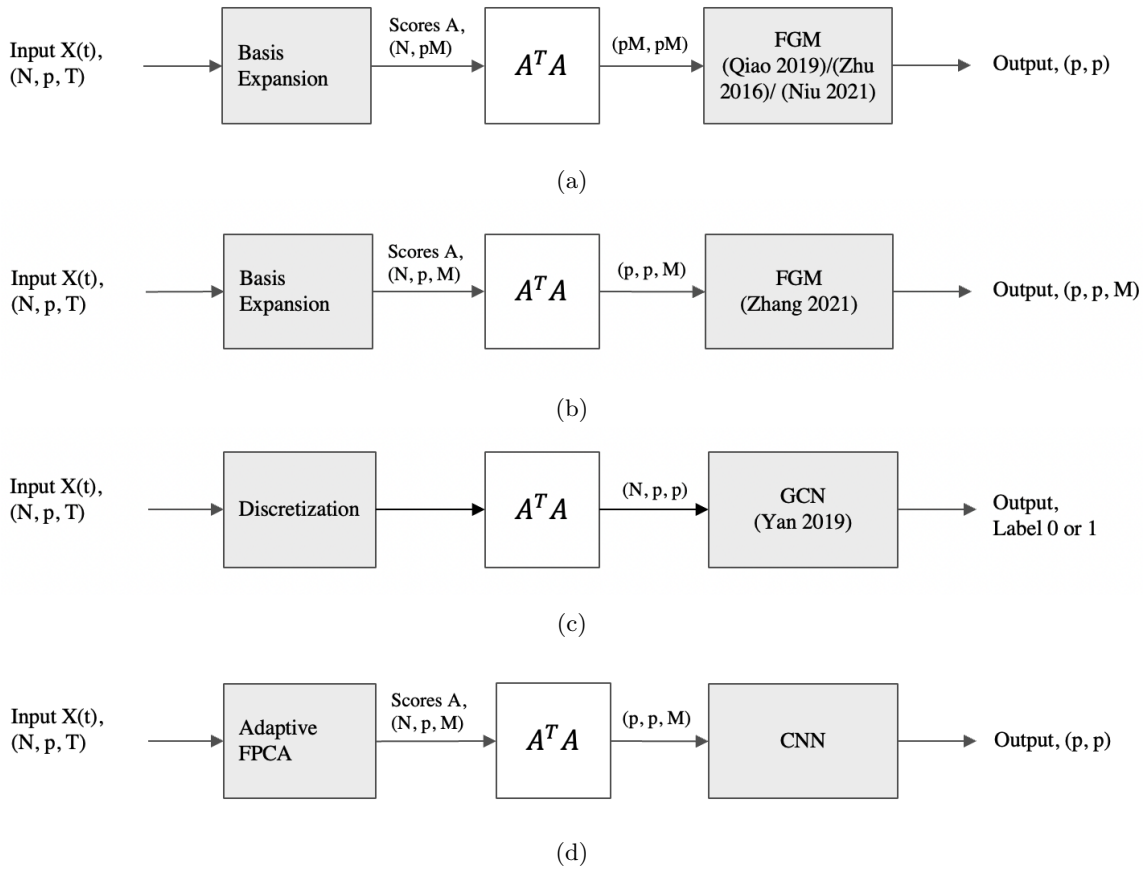

Figure 2.1: The overview of different architectures among graphical models for multivariate functional data. (a) The workflow of functional graphical models (Qiao et al., 2019; Niu et al., 2021; Zhu et al., 2016) utilizes basis expansion (orthogonal basis representation or functional principal components) combining with block coordinate descent or Bayesian regularization to infer a static graph. (b) Zhang et al. (2021b) uses the basis expansion strategy as well, then they adopts Bayesian regularization with a N-hypo mixture prior which allows the estimation of network can vary over the functional domain. The basis expansion approach in top-two frameworks can involve any general basis functions, such as functional principal component, wavelets, Fourier bases or B-splines. (c) Yan et al. (2019) treats the discretely sampled data from subjects as time series and leveraging grouping-based graph convolutional neural network for classification based on correlation matrices. (d) Our proposed neural network architecture for estimating functional graphical models consists of adaptive FPCA inspired by Yao et al. (2021a) and convolutional neural network.

et al., 2019), or Bayesian estimators (Niu et al., 2021; Zhu et al., 2016), to infer the estimated precision/adjacency matrix with dimension $p \times p$. The second top panel demonstrates the data forward pass to infer dynamic graph over a time interval (Zhang et al., 2021a). With the same shape of input, they also employ basis function representation approach to have the scores but have assumption of the independence of the bases across nodes. Similarly, then they transformed data through $A^T A$ and utilize the Bayesian regularization techniques and yield M-pair estimated precision matrices, which result in a time-varying graph. Furthermore, the third panel from the top is overview of GroupINN architecture aiming at classifying multivariate functional data (Yan et al., 2019). Instead of functional data analysis, they treat the discretely observed functional values $\{X(t_1), \ldots, X(t_T)\}$ as a time series and obtain a correlation-based matrix for each subject. They proposed a grouping-based graph convolutional neural network for a classification task.

In the setting of above state-of-the-art methods, two approaches that deal with infinite dimensional functional data are included, i.e, discretization and basis expansion. Obviously, the discretization of functions has drawbacks that it assumes the stationarity of the data over time. The frequentest FGM (Qiao et al., 2019) and Bayesian FGM (Niu et al., 2021; Zhang et al., 2021b) utilized standard dimension reduction then formulation on the (blockwise) precision matrix with regularization techniques. The popular bases are functional principal component (Li and Hsing, 2010; Yao et al., 2005; Silverman, 1996) or mathematical basis like Fourier bases, B-splines, wavelets (Cai et al., 2011; Aston et al., 2010). These basis functions and the number of bases need to be pre-determined with domain knowledge or the empirical characteristics of applications. For example, functional principal components work better for sparse and smooth functions (Baladandayuthapani et al., 2014; Aston et al., 2010); Fourier bases are suitable for functions with stationary periods; wavelets are more proper for irregular functions with spikes and discontinuities. The basis pre-selection needs domain knowledge and is not an easy task. On the other hand, separable dimension reduction task is aiming at retaining as much input information as possible and the basis function representations does not involve task-related information, which may introduce bias into graph inference. Yao et al. (2021a) embed adaptive FPCA into neural network and achieve end-to-end fashion of training for regression task. In their framework the bases is data-driven and learned specifically for their task.

2.2 Bayesian Functional Graphical Models

In this section, we propose two different regularization schemes for functional graphical models. The first approach we consider is a direct Bayesian version of the frequentist functional graphical lasso proposed by Qiao et al. (2019). We propose also a functional graphical horseshoe, due to the horseshoe's known improvements upon the lasso's tendency to over-shrink large coefficients and under-shrink small coefficients in high-dimensional problems (Wang, 2012; Li et al., 2019). Whereas most existing Bayesian approaches to covariance or precision matrix estimation assume structure such as banded covariance (e.g., Banerjee and Ghosal, 2014) or decomposable graphs (e.g., Rajaratnam et al., 2008; Xiang et al., 2015; Zhu et al., 2016), neither the Bayesian functional graphical lasso nor the functional graphical horseshoe assume any structure other than sparsity. We provide efficient Gibbs sampling algorithms for both of our proposed models, exploiting auxiliary variables to produce a set of easily-sampled conditional distributions. Through extensive simulation studies in Section 4.1, we evaluate both the classification accuracy and fidelity of the estimated coefficients. We apply our proposed Bayesian functional graphical horseshoe to two motivating data sets in Section 5.1 and 5.2, the EEG alcoholic versus control study presented by Qiao et al. (2019), and a novel study of white matter connectivity between healthy patients and those with a history of traumatic brain injury using data obtained from the Alzheimer's Disease Neuroimaging Initiative (ADNI). The code for implementing our proposed algorithms is available at `https://github.com/jjniu/BayesFGM`.

2.2.1 The Bayesian Fglasso

It is well known that frequentist optimization of objective functions may often be viewed as maximum a posteriori (MAP) estimation under a Bayesian model, provided there exists a prior density corresponding to the penalty term in the objective function. For the fglasso objective function in (1.9), the Bayesian counterpart uses a prior on the precision matrix given by

$$\pi(\mathbf{\Theta}) \propto \exp\left\{-\lambda \sum_{i \neq j} \|\mathbf{\Theta}_{ij}\|_F\right\}, \tag{2.1}$$

where $\|\cdot\|_F$ denotes the Frobenius norm and $\mathbf{\Theta}_{ij} \in \mathbb{R}^{M \times M}$ is the (i,j)th submatrix in $\mathbf{\Theta}$ associated with the conditional cross-correlation between node i and node j, $i, j = 1, \ldots, p$; $i \neq j$. Since

the precision matrix is symmetric, we need only to consider the upper off-diagonal elements for computational simplicity. As used in the Bayesian group lasso hierarchical representation (Kyung et al., 2010), we have the following identity,

$$\exp(-\lambda\|\mathbf{\Theta}_{ij}\|_F) = \int_0^\infty \left(\frac{1}{2\pi\tau_{ij}^2}\right)^{\frac{M^2}{2}} \exp\left(-\frac{\|\mathbf{\Theta}_{ij}\|_F^2}{2\tau_{ij}^2}\right) \frac{\left(\frac{\lambda^2}{2}\right)^{\frac{M^2+1}{2}} (\tau_{ij}^2)^{\frac{M^2+1}{2}-1}}{\Gamma\left(\frac{M^2+1}{2}\right)} \times \exp\left(-\frac{\lambda^2\tau_{ij}^2}{2}\right) d\tau_{ij}^2. \tag{2.2}$$

Thus, we can rewrite $\pi(\mathbf{\Theta})$ as a scale mixture of a multivariate normal distribution on the off-diagonal elements. Let $\omega_{ij} = \text{vec}(\mathbf{\Theta}_{ij}) \in \mathbb{R}^{M^2}$, for $i, j = 1, \ldots, p,\ i \neq j$. Then we can introduce the auxiliary latent parameters $\tau = (\tau_{ij})$, so the prior in (2.1) can be attained as a gamma mixture of normals, leading to the functional graphical lasso hierarchy

$$\omega_{ij}|\tau_{ij}^2 \sim N_{M^2}(\mathbf{0}, \tau_{ij}^2\mathbf{I}_{M^2}); \quad \tau_{ij}^2 \sim \text{Gamma}\left(\frac{M^2+1}{2}, \frac{\lambda^2}{2}\right). \tag{2.3}$$

We assume the basis expansion is a lossless or approximately lossless representation from the raw data $g_{ij}(t)$ to $\mathbf{a}_{ij}^M$, where isomorphic transformation ensures that any basis coefficients can be considered as transformed raw data rather than estimated parameters (Morris et al., 2011). Denote with $\hat{\mathbf{a}}_i = (\hat{\mathbf{a}}_{i1}^T, \ldots, \hat{\mathbf{a}}_{ip}^T)^T \in \mathbb{R}^{Mp}$ the estimated M-truncated functional principal component scores for the observed functions on sample i, $g_{i1}(\cdot), \ldots, g_{ip}(\cdot)$, where for simplicity we assume the truncation level M is the same across all p nodes. When $(g_{i1}(\cdot), \ldots, g_{ip}(\cdot))$ are drawn from an MGP, $\hat{\mathbf{a}}_i$ is assumed following an Mp-dimensional Gaussian distribution. Then the Bayesian fglasso model can be expressed as

$$\begin{aligned} p(\hat{\mathbf{a}}_i|\mathbf{\Theta}) &= N_{Mp}(\hat{\mathbf{a}}_i|\mathbf{0}, \mathbf{\Theta}^{-1}), \quad i = 1, \ldots, N \\ p(\mathbf{\Theta}|\lambda) &= \frac{1}{C}\prod_{\ell=1}^{Mp} Exp\left(\theta_{\ell\ell}|\frac{\lambda^2}{2}\right) \prod_{i<j} N_{M^2}(\omega_{ij}|\mathbf{0}, \tau_{ij}^2\mathbf{I}_{M^2})\text{Gamma}\left(\tau_{ij}^2|\frac{M^2+1}{2}, \frac{\lambda^2}{2}\right), \end{aligned} \tag{2.4}$$

where $\theta_{11}, \ldots, \theta_{pp}$ are the diagonal elements of $\mathbf{\Theta}$ and C is a normalizing constant.

The hierarchical representation in (2.3) facilitates the use of conditional conjugacy in deriving a block Gibbs sampler for exploring the posterior distribution. For a fixed regularization

parameter λ, the posterior distribution associated with the Bayesian fglasso model (2.4) is given by

$$
\begin{aligned}
p(\mathbf{\Theta}, \tau^2 | \mathbf{S}, \lambda) \propto & |\mathbf{\Theta}|^{\frac{n}{2}} \exp\left\{ -\operatorname{tr}(\frac{1}{2}\mathbf{S}\mathbf{\Theta}) \right\} \prod_{\ell=1}^{Mp} \frac{\lambda^2}{2} \exp\left(-\frac{\lambda^2}{2}\theta_{\ell\ell} \right) \\
& \times \prod_{i<j} \left(\frac{1}{2\pi\tau_{ij}^2} \right)^{\frac{M^2}{2}} \exp\left(-\frac{\|\mathbf{\Theta}_{ij}\|_F^2}{2\tau_{ij}^2} \right) \\
& \times \left\{ \frac{\left(\frac{\lambda^2}{2}\right)^{\frac{M^2+1}{2}} (\tau_{ij}^2)^{\frac{M^2+1}{2}-1}}{\Gamma\left(\frac{M^2+1}{2}\right)} \exp\left(-\frac{\lambda^2\tau_{ij}^2}{2} \right) \right\} 1_{\mathbf{\Theta} \in M^+},
\end{aligned}
\tag{2.5}
$$

where $\mathbf{S} = \sum_{i=1}^n \hat{\mathbf{a}}_i \hat{\mathbf{a}}_i^T$ is the sample scatter matrix of the functional principal component (fpc) scores. This representation allows us to adapt the block Gibbs sampling scheme proposed by Wang (2012).

By assumption, the off-diagonal entries of each block submatrix on the main diagonal, $\mathbf{\Theta}_{ii},\ i = 1, \ldots, p$, are all zeros. Partition the precision and sample fpc score covariance matrix as follows:

$$
\mathbf{\Theta} = \begin{bmatrix} \mathbf{\Theta}_{11} & \theta_{12} \\ \theta_{12}^T & \theta_{22} \end{bmatrix}, \ \mathbf{S} = \begin{bmatrix} \mathbf{S}_{11} & \mathbf{s}_{12} \\ \mathbf{s}_{12}^T & s_{22} \end{bmatrix} \tag{2.6}
$$

where we define

$$
\theta_{12} = \begin{bmatrix} \bar{\theta}_{12} \\ \mathbf{0} \end{bmatrix}. \tag{2.7}
$$

With $\mathbf{\Theta}$ permuted so that the last column / row corresponds to node j and score k, $\bar{\theta}_{12} = \operatorname{cov}(\hat{a}_{jk}, (\hat{\mathbf{a}}_1, \ldots, \hat{\mathbf{a}}_{j-1}, \hat{\mathbf{a}}_{j+1}, \ldots, \hat{\mathbf{a}}_p)^T) \in \mathbb{R}^{M(p-1)}$ with $\hat{\mathbf{a}}_k \in \mathbb{R}^M$ the collection of fpc scores at node k. The $\mathbf{0} \in \mathbb{R}^{M-1}$ vector follows from $\hat{a}_{jk}$ being uncorrelated with other scores at node j.

Define $\mathbf{T} = \left(\tau_{ij}^2\right)_{p\times p} \otimes \mathbf{J}_{M\times M}$ where $\tau_{ii} = 0$ for $i = 1, \ldots, p$ and $\mathbf{J}_{M\times M} = \mathbf{1}\mathbf{1}^T$ is the matrix with all ones. We similarly partition it as

$$
\mathbf{T} = \begin{bmatrix} \mathbf{T}_{11} & \mathbf{t}_{12} \\ \mathbf{t}_{12}^T & 0 \end{bmatrix}, \tag{2.8}
$$

where

$$
\mathbf{t}_{12} = \begin{bmatrix} \bar{\mathbf{t}}_{12} \\ \mathbf{0} \end{bmatrix} \tag{2.9}
$$

with $\bar{\mathbf{t}}_{12} \in \mathbb{R}^{M(p-1)}$ defined analogously to $\bar{\theta}_{12}$.

The conditional distribution of the nonzero variables in the last column (or row) of $\mathbf{\Theta}$ is

$$\begin{aligned} p(\bar{\theta}_{12}, \theta_{22} | \mathbf{\Theta}_{11}, \mathbf{T}, \mathbf{S}, \lambda) &\propto (\theta_{22} - \bar{\theta}_{12}^T \overline{\mathbf{\Theta}_{11}^{-1}} \bar{\theta}_{12})^{\frac{n}{2}} \\ &\times \exp\left\{ -\frac{1}{2} [\bar{\theta}_{12}^T \mathbf{D}_\tau^{-1} \bar{\theta}_{12} + 2\bar{\mathbf{s}}_{12}^T \bar{\theta}_{12} + (s_{22} + \lambda^2)\theta_{22}] \right\} \end{aligned} \tag{2.10}$$

where $\mathbf{D}_\tau = \text{diag}(\bar{\mathbf{t}}_{12})$ and $\overline{\mathbf{\Theta}_{11}^{-1}} \in \mathbb{R}^{M(p-1)\times M(p-1)}$ is the cross covariance matrix associated with the remaining $p-1$ nodes. We make a change of variables, $\beta = \bar{\theta}_{12}, \gamma = (\theta_{22} - \bar{\theta}_{12}^T \overline{\mathbf{\Theta}_{11}^{-1}} \bar{\theta}_{12})$, and denote $\mathbf{C} = (\mathbf{D}_\tau^{-1} + (s_{22} + \lambda^2)\overline{\mathbf{\Theta}_{11}^{-1}})^{-1}$. This implies

$$\beta, \gamma | \mathbf{\Theta}_{11}, \mathbf{T}, \mathbf{S}, \lambda \sim N_{M(p-1)}(-\mathbf{C}\bar{\mathbf{s}}_{21}, \mathbf{C}) \text{Gamma}\left(\frac{n}{2} + 1, \frac{s_{22} + \lambda^2}{2}\right). \tag{2.11}$$

All elements in the matrix $\mathbf{\Theta}$ can be sampled by sampling one row and column at a time, permuting $\mathbf{\Theta}$ after each iteration. Due to the structure of $\hat{\mathbf{a}}_i$, we first cycle through all columns corresponding to the same node, then move to next node.

After complete updating of all the off-diagonal elements, the diagonal elements of $\mathbf{\Theta}$ and the shrinkage parameters τ_{ij} need to be sampled. The full conditional distributions of $(\tau_{ij}^2)^{-1}$ are seen to be independently inverse Gaussian with mean $\sqrt{\frac{\lambda^2}{\|\mathbf{\Theta}_{ij}\|_F^2}}$ and shape λ^2. Put another way, the reparameterized model based on one particular permutation of $\mathbf{\Theta}$ under the Bayesian functional graphical lasso is

$$\begin{aligned} &\beta | \mathbf{\Theta}_{11}, \mathbf{T}, \mathbf{S}, \lambda \sim N_{M(p-1)}(-\mathbf{C}\bar{\mathbf{s}}_{21}, \mathbf{C}) \\ &\gamma | \mathbf{S}, \lambda \sim \text{Gamma}\left(\frac{n}{2} + 1, \frac{s_{22} + \lambda^2}{2}\right) \\ &\frac{1}{\tau_{ij}^2} | \mathbf{\Theta}_{ij}, \lambda \overset{indep.}{\sim} \text{Inverse Gaussian}\left(\sqrt{\frac{\lambda^2}{\|\mathbf{\Theta}_{ij}\|_F^2}}, \lambda^2\right), \quad i, j = 1, \ldots, p; \ i \neq j. \end{aligned} \tag{2.12}$$

Since $\gamma > 0$ with probability one, the positive definite constraint on $\mathbf{\Theta}$ is maintained in each iteration. The argument for the functional case is adapted from that given by Wang (2012). Suppose at the current iteration the sample $\mathbf{\Theta}^{(c)}$ is positive definite, so all its pM corresponding leading principal minors are positive. After updating the particular column and row of $\mathbf{\Theta}$ by sampling β and γ by (2.12), the new sample $\mathbf{\Theta}^{(c+1)}$ has the same leading principal minors as $\mathbf{\Theta}^{(c)}$ except the one corresponding to the updated column/row, which is of order pM. It is easy to find that

Algorithm 1 Bayesian functional graphical lasso Gibbs sampler

Input: Sum of the products matrix $\mathbf{S}$, i.e., $\mathbf{S} = \mathbf{Y}^T\mathbf{Y}$.
Output: MCMC sample of the precision matrix $\mathbf{\Theta}^{(1)}, \ldots, \mathbf{\Theta}^{(L)}$.
Initialization: Set p to be number of nodes in graph, set initial values $\mathbf{\Theta}^{(0)} = \mathbf{I}$, $\mathbf{\Sigma}^{(0)} = \mathbf{I}$, $\mathbf{T}^{(0)} = \mathbf{J}$, where $\mathbf{I}$ is $pM \times pM$ identity matrix and $\mathbf{J}$ is a $pM \times pM$ matrix with all elements equal to one
while *Convergence criteria are not met* **do**
  for $i = 1, \ldots, p$ **do**
    Partition $\mathbf{\Theta}^{(l)}$, $\mathbf{S}$ and $\mathbf{T}^{(l)}$ into $p \times p$ blocks (focus on updating ith column block of $\mathbf{\Theta}$ corresponding node i)
    for $j = 1, \ldots, M$ **do**
      1. Partition $\mathbf{\Theta}^{(l)}$, $\mathbf{S}$ and $\mathbf{T}^{(l)}$ as in (2.6) and (2.8)
      2. Draw $\gamma^{(l+1)} \sim \text{Gamma}\left(\frac{n}{2} + 1, \frac{s_{22}+\lambda}{2}\right)$
      3. Draw $\beta^{(l+1)} \sim N_{(p-1)M}(-\mathbf{C}^{(l)}\bar{\mathbf{s}}_{21}, \mathbf{C}^{(l)})$,where $\mathbf{C}^{(l)} = (\mathbf{D}_\tau^{-1\,(l)} + (s_{22} + \lambda)\overline{\mathbf{\Theta}_{11}^{-1}})^{-1\,(l)}$
      4. Update $\theta_{21}^{(l+1)} = (\beta^{(l+1)}, \mathbf{0}), \theta_{12}^{(l+1)} = \theta_{21}^{T\,(l+1)}, \theta_{22}^{(l+1)} = \gamma^{(l+1)} + \beta^{T\,(l+1)}\overline{\mathbf{\Theta}_{11}^{-1}}^{(l+1)}\beta^{(l+1)}$
    end
    Update $\mathbf{T}^{(l+1)}$ by sampling $(1/\tau_{ij}^2)^{(l+1)}|\mathbf{\Theta}^{(l+1)}, \lambda \sim$ Inverse Gaussian $\left(\sqrt{\frac{\lambda^2}{\|\mathbf{\Theta}_{ij}^{(l+1)}\|_F^2}}, \lambda^2\right)$ for $i, j = 1, \ldots, p$
  end
  Store the realization of precision matrix $\mathbf{\Theta}^{(l+1)}$ Increment $l \leftarrow l + 1$.
end

this last leading principal minor is $\det(\mathbf{\Theta}^{(c+1)}) = \gamma \det(\mathbf{\Theta}_{11}^{(c)})$, where $\det(\mathbf{\Theta}_{11}^{(c)})$ is the $(pM-1)^{th}$ leading principal minor of $\mathbf{\Theta}^{(c)}$ excluding the updated column and row. Thus $\gamma > 0$ means that $\det(\mathbf{\Theta}^{(c+1)}) > 0$ and all leading principal minors of the updated matrix are positive. To ensure each MCMC realization $\mathbf{\Theta}^{(m)} \in M^+$ for $m = 0, 1, 2, \ldots$, it is only required that the chain is initialized with $\mathbf{\Theta}^{(0)} \in M^+$. Algorithm 1 details the Bayesian fglasso Gibbs sampler.

Given the MCMC output of a sample of precision matrices, $\mathbf{\Theta}^{(1)}, \ldots, \mathbf{\Theta}^{(L)}$, several inferential procedures are possible for constructing an estimate of $\mathbf{\Theta}$. Continuous shrinkage priors do not put positive probability mass on exact zeros in the precision matrix, and Carvalho et al. (2010) argue that using (non zero) posterior means as the basis for inference is often preferable to binary thresholding due to the estimator's optimality under squared error loss. Nevertheless, it is sometimes necessary to produce a sparse estimate with exact zeros, especially in the case of graphical models. Carvalho et al. (2010) and Wang (2012) discuss some possible thresholding rules. In our case, we construct the precision matrix (and thus graph) estimate by Bayesian false discovery rate (FDR) based inference or confidence regions of $\{\hat{\mathbf{\Theta}}_{ij}\}_{i<j}$, which is discussed in Section 2.2.3.

The Bayesian fglasso proposed here assumes that the regularization parameter λ is fixed, meaning that it must be tuned and selected *a priori*. Cross-validation is computationally expensive,

especially for Bayesian models implemented via MCMC. Further, Wasserman and Roeder (2009) show that cross-validation based on the log-likelihood loss function tends to lead to overfitting and unnecessarily dense graphs. Other than cross validation, approaches such as Akaike information criterion (AIC), Bayesian information criterion (BIC), and stability selection (Meinshausen et al., 2006) have been well studied in the graphical model literature. In the functional case, though, AIC/BIC does not work well, since it is unclear how to calculate the effective degrees of freedom. Thus, selecting an appropriate hyperparameter λ ahead of time is a nontrivial task. On the other hand, in the Bayesian framework, we can (for instance) assign a gamma prior $\lambda^2 \sim \text{Gamma}(s, r)$. In this case, the full conditional for λ is

$$\lambda^2 | \mathbf{T}, \mathbf{\Theta} \sim \text{Gamma}\left(s + pM + \frac{p(p-1)(M^2+1)}{4},\ r + \frac{\sum_l \theta_{ll} + \sum_{i<j} \tau_{ij}^2}{2} \right). \tag{2.13}$$

This can in turn be incorporated into the Gibbs sampler given in Algorithm 1 as an additional sampling step. To simplify the computation in algorithm, we assume identical M across j under the assumption that the corresponding covariance operators share similar complexity structure.

In general, different functional variables may have different smoothness levels, we could assume different truncated number M_j across $j = 1, \ldots, p$. With different truncated number, $\mathbf{\Theta}_{ij}$ is a rectangle block with size $M_i \times M_j$ and $\mathbf{\Theta}$ has dimension $\sum_{j=1}^{p} M_j$. It is straightforward to modify the algorithm with nonsquare blocks. The full conditional for λ is updated as

$$\lambda^2 | \mathbf{T}, \mathbf{\Theta} \sim \text{Gamma}\left(s + \sum_j M_j + \sum_{i<j} \frac{M_i M_j - 1}{2},\ r + \frac{\sum_l \theta_{ll} + \sum_{i<j} \tau_{ij}^2}{2} \right). \tag{2.14}$$

2.2.2 The Functional Graphical Horseshoe

In the presence of sparsity, as is often the case for precision matrices associated with GGMs, it is desirable to have a shrinkage approach that yields exact or values close to zero for the true null cases while simultaneously shrinking the truly non-zero cases as small as possible so that the resulting estimates have little bias. To address this desire, Carvalho et al. (2010) proposed the horseshoe prior. The prior has high probability concentration near zero and and is heavy-tailed, properties that contribute to desired shrinkage behavior. Further, the prior can be expressed as a scale mixture of Gaussian distributions and thus is easily incorporated into a Gibbs sampler for posterior exploration. Carvalho et al. (2010) originally proposed the horseshoe for the sparse normal

means model, but it was recently extended by Li et al. (2019) to estimation of GGMs. Li et al. (2019) established that the resulting horseshoe estimates are close to be unbiased least-square estimates with high probability and, further, that the Bayesian graphical lasso tends to be further away from the least squares estimates than the graphical horseshoe (the least squared estimate does not exist when $n < p$). In this section, we propose an extension of graphical horseshoe regularization to the case of functional graphical models.

We define the functional graphical horseshoe by using horseshoe priors on each off-diagonal block of the precision matrix and exponential priors on the diagonal elements. This yields the following prior:

$$\begin{aligned}
&\theta_{\ell\ell} \sim Exp(\lambda_{\ell\ell}^2/2), \quad \ell = 1, \ldots, pM \\
&\omega_{ij} \overset{indep.}{\sim} N_{M^2}(\mathbf{0},\ \lambda_{ij}^2\tau^2\mathbf{I}), \quad i,j = 1, \ldots, p,\ i \neq j \\
&\lambda_{ij} \overset{iid}{\sim} C^+(0,1), \quad i,j = 1, \ldots, p \\
&\tau \sim C^+(0,1),
\end{aligned} \tag{2.15}$$

where $\omega_{ij} = \text{vec}(\mathbf{\Theta}_{ij})$ and $C^+(0,1)$ represents the half-Cauchy distribution with density $p(x) \propto (1+x^2)^{-1},\ x > 0$. As in other versions of the horseshoe prior, the global shrinkage parameter τ is determined by the sparsity of the entire precision matrix, whereas the local shrinkage parameters λ_{ij} preserves blocks with $\|\mathbf{\Theta}_{ij}\| \neq 0$ by allowing them to be pulled toward zero considerably less than the zero blocks. Unlike Li et al. (2019), but similar to Wang (2012), we specify an $Exp(\lambda_{\ell,\ell}^2/2)$ prior for the diagonal elements of $\mathbf{\Theta}$. This is convenient for deriving the full conditional distributions and does not affect inference since the graph is determined by the off-diagonal elements.

The full conditional distribution of $\mathbf{\Theta}$ under the assumption of multivariate Gaussian likelihood is given by

$$\begin{aligned}
p(\mathbf{\Theta}|\lambda, \Lambda, \tau, \mathbf{S}) \propto |\mathbf{\Theta}|^{\frac{n}{2}} \exp\left\{-\text{tr}(\frac{1}{2}\mathbf{S}\mathbf{\Theta})\right\} \prod_{l=1} \frac{\lambda_{ll}^2}{2} \exp\left(-\frac{\lambda_{ll}^2}{2}\theta_{ll}\right) \\
\times \prod_{i<j} N_{M^2}(\omega_{ij}|\mathbf{0}, \lambda_{ij}^2\tau^2\mathbf{I}) C^+(\lambda_{ij}|0,1) 1_{\mathbf{\Theta} \in M^+}.
\end{aligned} \tag{2.16}$$

where $\Lambda = \{\lambda_{ij}\}_{i,j=1}^p$. The standard technique for creating a straightforward Gibbs sampler with the functional graphical horseshoe is to use the realization of Makalic and Schmidt (2016) that if $x^2|a \sim$ inverse Gamma$(1/2, 1/a)$ and $a \sim$ inverse Gamma$(1/2, 1/A^2)$, then, marginally, $x \sim C^+(0, A)$. Thus, we introduce latent variables ν_{ij} and ζ to facilitate conditional conjugacy

when updating the shrinkage parameters λ_{ij} and τ.

Under the parameter-expanded hierarchical model, the full conditional distribution of the precision matrix is given by

$$\begin{aligned} p(\mathbf{\Theta}|\mathbf{S},\mathbf{\Lambda},\tau,\mathbf{V},\zeta) \propto |\mathbf{\Theta}|^{\frac{n}{2}} \exp\left\{-\operatorname{tr}(\frac{1}{2}\mathbf{S}\mathbf{\Theta})\right\} \prod_{l=1}^{Mp} \frac{\lambda_{\ell\ell}^2}{2} \exp\left(-\frac{\lambda_{\ell\ell}^2}{2}\theta_{ll}\right) \\ \times \prod_{i<j} N_{M^2}(\omega_{ij}|\mathbf{0},\lambda_{ij}^2\tau^2\mathbf{I}) \prod_{i<j} \nu_{ij}^{-\frac{1}{2}} \lambda_{ij}^{-3} \exp\left(-\frac{1}{\lambda_{ij}^2\nu_{ij}}\right) \nu_{ij}^{-\frac{3}{2}} \exp\left(-\frac{1}{\nu_{ij}}\right) \\ \times \zeta^{-\frac{1}{2}}\tau^{-3} \exp\left(-\frac{1}{\tau^2\zeta}\right) \zeta^{-\frac{3}{2}} \exp\left(-\frac{1}{\zeta}\right). \end{aligned} \tag{2.17}$$

We can use a data-augmented Gibbs sampler with the same matrix permutation as used for the Bayesian fglasso proposed in Subsection 2.2.1.

In each iteration, the rows and columns of the $Mp-$dimensional matrices $\mathbf{\Theta}$, $\mathbf{S}$, $\mathbf{\Lambda} = \{\lambda_{ij}^2\}_{p\times p} \otimes \mathbf{J}_{M\times M}$, and $\mathbf{V} = \{\nu_{ij}^2\}_{p\times p} \otimes \mathbf{J}_{M\times M}$ are partitioned the same way as in Subsection 2.2.1 to derive the full conditional distributions; i.e.,

$$\begin{aligned} \mathbf{\Theta} = \begin{bmatrix} \mathbf{\Theta}_{11} & \theta_{12} \\ \theta_{12}^T & \theta_{22} \end{bmatrix}, \ \mathbf{S} = \begin{bmatrix} \mathbf{S}_{11} & \mathbf{s}_{12} \\ \mathbf{s}_{12}^T & s_{22} \end{bmatrix}, \\ \mathbf{\Lambda} = \begin{bmatrix} \mathbf{\Lambda}_{11} & \lambda_{12} \\ \lambda_{12}^T & \lambda_{22} \end{bmatrix}, \ \mathbf{V} = \begin{bmatrix} \mathbf{V}_{11} & \nu_{12} \\ \nu_{12}^T & \nu_{22} \end{bmatrix}, \end{aligned} \tag{2.18}$$

where the blocks are arranged as before. The derivation of full conditionals for the last column θ_{12} and θ_{22} is similar to the Bayesian fglasso by changing variables. The conditional distribution of nonzero variables of the last column in $\mathbf{\Theta}$ is

$$p(\bar{\theta}_{12},\theta_{22}|-) \propto \left(\theta_{22} - \bar{\theta}_{12}^T\overline{\mathbf{\Theta}_{11}^{-1}}\bar{\theta}_{12}\right)^{\frac{n}{2}} \exp\left\{-\frac{1}{2}[\bar{\theta}_{12}^T\mathbf{D}_\tau^{-1}\bar{\theta}_{12} + 2\bar{\mathbf{s}}_{12}^T\bar{\theta}_{12} + (s_{22}+\lambda_{22}^2)\theta_{22}]\right\}, \tag{2.19}$$

where $\mathbf{D}_\tau = \tau^2\mathrm{diag}(\bar{\lambda}_{12})$. Making a change of variables by $\beta = \bar{\theta}_{12}, \gamma = (\theta_{22} - \bar{\theta}_{12}^T\overline{\mathbf{\Theta}_{11}^{-1}}\bar{\theta}_{12})$, and letting $\mathbf{C} = (\mathbf{D}_\tau^{-1} + (s_{22}+\lambda_{22}^2)\overline{\mathbf{\Theta}_{11}^{-1}})^{-1}$, then the full conditional of β,γ is

$$\beta,\gamma|- \sim N_{(p-1)M}(-\mathbf{C}\bar{\mathbf{s}}_{21},\mathbf{C})\mathrm{Gamma}\left(\frac{n}{2}+1, \frac{s_{22}+\lambda_{22}^2}{2}\right) \tag{2.20}$$

As for the Bayesian fglasso, we first cycle through all columns corresponding to the same node, then

Algorithm 2 Bayesian functional graphical horseshoe Gibbs sampler

Input: Sum of the products matrix $\mathbf{S}$, i.e., $\mathbf{S} = \mathbf{Y}^T\mathbf{Y}$.
Output: Samples of precision matrix $\hat{\mathbf{\Theta}}$.
Initialization: Set p to be number of nodes in graph, set initial values $\mathbf{\Theta} = \mathbf{I}$, $\mathbf{\Sigma} = \mathbf{I}$, $\mathbf{\Lambda} = \mathbf{J}$, $\mathbf{V} = \mathbf{J}$, where $\mathbf{I}$ is $pM \times pM$ identity matrix and $\mathbf{J}$ is a $pM \times pM$ matrix with all elements equal to one
while *Given the current* $\mathbf{\Theta} \in M^+$ *and* τ*, repeat for a large number of iterations until convergence is achieved* **do**
  for $i = 1, \ldots, p$ **do**
    Partition $\mathbf{\Theta}$, $\mathbf{S}$, $\mathbf{T}$ and $\mathbf{V}$ into $p \times p$ blocks (focus on updating ith column block of $\mathbf{\Theta}$ corresponding node i and all the other nodes)
    1. **for** $j = 1, \ldots, M$ **do**
      (1) Partition $\mathbf{\Theta}^{(l)}$, $\mathbf{S}$, $\mathbf{\Lambda}^{(l)}$ and $\mathbf{V}^{(l)}$ as (2.18)
      (2) Draw $\gamma^{(l+1)} \sim$ Gamma $\left(\frac{n}{2}+1, \frac{s_{22}+(\lambda_{22}^{(l)})^2}{2}\right)$
      (3) Draw $\beta^{(l+1)} \sim N_{M(p-1)}(-\mathbf{C}^{(l)}\bar{\mathbf{s}}_{21}, \mathbf{C}^{(l)})$,
      where $\mathbf{C}^{(l)} = (\mathbf{D}_\tau^{(l)-1} + (s_{22} + (\lambda_{22}^{(l)})^2)(e\mathbf{\Theta}_{11}^{-1})^{(l)})^{-1}$
      (4) Update $\theta_{21}^{(l+1)} = (\beta^{(l+1)}, \mathbf{0}), \theta_{12}^{(l)} = \theta_{21}^T, \theta_{22}^{(l)} = \gamma^{(l+1)} + (\beta^{(l+1)})^T(\overline{\mathbf{\Theta}_{11}^{-1}})^{(l+1)}\beta^{(l+1)}$
    end
    2. Update $\mathbf{\Lambda}^{(l+1)}$, i.e., draw sample $(\lambda_{ij}^2)^{(l+1)} \sim$ inverse Gamma $\left(\frac{M^2+1}{2}, \frac{1}{\nu_{ij}^{(l)}} + \frac{\|\mathbf{\Theta}_{ij}^{(l+1)}\|_F^2}{2\tau^{2(L)}}\right)$
    3. Update $\mathbf{V}^{(l+1)}$, i.e., draw sample $\nu_{ij}^{(l+1)} \sim$ inverse Gamma $\left(1, 1 + \frac{1}{(\lambda_{ij}^2)^{(l+1)}}\right)$
    4. Update $\tau^{(l+1)}$ and $\zeta^{(l+1)}$, i.e., $(\tau^2)^{(l+1)} \sim$ inverse Gamma $\left(\frac{M^2(p-1)p+2}{4}, \frac{1}{\zeta^{(l)}} + \sum_{i<j} \frac{\|(\mathbf{\Theta}_{ij})^{(l+1)}\|_F^2}{2(\lambda_{ij}^2)^{(l+1)}}\right)$,
    $\zeta^{(l+1)} \sim$ inverse Gamma $\left(1, 1 + \frac{1}{(\tau^2)^{(l+1)}}\right)$
  end
  Store the sample precision matrix $\mathbf{\Theta}$ Increment $l \leftarrow l+1$.
end

move to next node. After the entire $\mathbf{\Theta}$ is updated, the local and global shrinkage parameters λ_{ij} and τ need to be sampled. Through conditional conjugacy, the full conditional distributions of $\lambda_{ij}, \nu_{ij}, \tau^2$, and ζ are quickly seen to be inverse Gamma. The condition $\mathbf{\Theta} \in M^+$ is maintained during each iteration as long as the starting value is positive definite, for the same reason that positive definite constraint is satisfied in the Bayesian fglasso sampler. The full Gibbs sampler is summarized in Algorithm 2. In the case of different truncated number of principal components for each node, the algorithm is straightforward. The block $\mathbf{\Theta}_{ij}$ is rectangle and $(\lambda_{ij}^2)^{(l+1)} \sim$ inverse Gamma$(\frac{M_iM_j+1}{2}, \frac{1}{\nu_{ij}^{(l)}} + \frac{\|\mathbf{\Theta}_{ij}^{(l+1)}\|_F^2}{2\tau^{2(L)}})$ and $(\tau^2)^{(l+1)} \sim$ inverse Gamma $\left(\frac{\sum_{i<j} M_iM_j+1}{2}, \frac{1}{\zeta^{(l)}} + \sum_{i<j} \frac{\|(\mathbf{\Theta}_{ij})^{(l+1)}\|_F^2}{2(\lambda_{ij}^2)^{(l+1)}}\right)$.

2.2.3 Bayesian FDR-based Inference and Confidence Regions

Our goal is to identify significant conditional dependence between different nodes, which can subsequently be mapped into edges in the estimated graph. An intuitive way is to identify

blocks with the Frobenius norm of block at least δ, which could be a practical threshold. Another way is to identify significant blocks by multiple statistical hypothesis tests. The former way ignores the variability in the data, while the latter one only focus on statistical significance and ignores the practical significance. We consider the direct posterior probability approach; i.e., Bayesian FDR-based inference (Storey, 2003; Morris et al., 2011) to achieve the goal in a way considering both statistical and practical significance.

First, we compute the edge strength as $\|\hat{\mathbf{\Theta}}_{ij}\|_F$ by (1.8) for $i, j = 1, \ldots, p$ for each sample of the precision matrix. Then we define a threshold δ as practical significance, which could be determined by or associated with some prior knowledge such as the desired sparsity of graph. For example, the value of δ for a desired sparsity level of 95% will be higher than the one for the desired sparsity level is 90%. Further, we could compute the posterior probability of $\|\hat{\mathbf{\Theta}}_{ij}\|_F$ at least δ intensity as

$$p_{ij}^{\delta} = \Pr\{\|\hat{\mathbf{\Theta}}_{ij}\|_F > \delta\} \approx \sum_{l=1}^{L} \frac{1}{L} I\{\|\hat{\mathbf{\Theta}}_{ij}^{l}\|_F > \delta\} \tag{2.21}$$

for $i, j = 1, \ldots, p$, here L is length of MCMC output. The quantities $1 - p_{ij}^{\delta}$ can be considered as a natural "Bayesian posterior p-value" or "positive false discovery rate" analogue of p-value (also named "q-value" by Storey (2003)), defined as the Frobenius norm of a block in precision matrix with at least δ. Given a significance level α, we then identify the significant blocks by $E = \{(i, j) : p_{ij}^{\delta} > \phi_{\alpha}^{\delta}\}$, where ϕ_{α}^{δ} is a threshold on the posterior probabilities that controls the average Bayesian FDR at level α. Follow the similar strategy from Morris et al. (2011), we sort the p_{ij}^{δ} by descending order to yield $p_{(ij)}^{\delta}$. Then

$$\phi_{\alpha}^{\delta} = p_{(i^* j^*)}^{\delta}, \quad \text{where } (i^*, j^*) = \arg\max\{(i, j) : \frac{1}{B} \sum_{(ij)} (1 - p_{(ij)}^{\delta}) \leq \alpha\} \tag{2.22}$$

(i^*, j^*) is the last index for which the cumulative average of the sorted local FDRs $1 - p_{ij}^{\delta}$ is less than or equal to average Bayesian FDR of α.

Another method to identify significant blocks among blockwise precision matrix is credible regions by Mahalanobis distance (McLachlan, 1999). Let $\hat{\theta}_{ij}$ be vectorized representation of $\hat{\mathbf{\Theta}}_{ij}$. For each edge, we have $\hat{\theta}_{ij}^{(1)}, \ldots, \hat{\theta}_{ij}^{(L)}$ posterior samples, then we could have sample covariance $\Sigma_{\theta_{ij}}$.

The approximate $(1-\alpha)\times 100\%$ joint confidence region for $\hat{\theta}_{ij}$ is

$$R_{ij} = \{\hat{\theta}_{ij} | (\hat{\theta}_{ij} - \bar{\theta}_{ij})^T \Sigma^{-1}_{\hat{\theta}_{ij}} (\hat{\theta}_{ij} - \bar{\theta}_{ij}) \leq q^*\} \tag{2.23}$$

where q^* is the smallest possible q such that

$$\sum_{l=1}^{L} I\{(\hat{\theta}_{ij} - \bar{\hat{\theta}}_{ij})^T \Sigma^{-1}_{\hat{\theta}_{ij}} (\hat{\theta}_{ij} - \bar{\theta}_{ij}) \leq q\} \geq L(1-\alpha). \tag{2.24}$$

The approximate volume of confidence regions with level $(1-\alpha)\times 100\%$ (hyper-ellipse) is proportional to $\chi^2_{p'\alpha} |\Sigma_{\hat{\theta}_{ij}}|^{\frac{1}{2}}$, where p' is the dimension of $\hat{\theta}_{ij}$.

2.3 Deep Functional Graphical Models

In this section, we explore a neural network based framework to estimate functional graphical models. Conventional methods require intermediate steps to generate basis representation of functional data, for example, PCA scores. We wonder if there exist a learnable mapping between functional data and basis function, so that we could avoid manually tuned basis transformation. In addition, it is ideal to convert every components in the processing pipeline to be differentiable so that we could leverage popular deep neural network to optimize the parameters automatically, also called end-to-end differentiation. We propose an end-to-end functional graph estimation based on neural network trained with risk minimization. We present our approach in the following parts: Section 2.3.1 describe the empirical risk minimization and how we use synthetic data generation process to convert a inference problem to be a learning problem; Section 2.3.2 demonstrate how we design our neural network architecture to make entire training end-to-end differentiable, and also discuss the training objective of our method. Through simulation studies in Section 4.2, we evaluate our proposed model through area under the ROC curves (AUC). In Section 5.1, we apply our proposed deep model to EEG alcoholic vs control study presented by Qiao et al. (2019).

2.3.1 Empirical Risk Minimization and Synthetic Data

We have three challenges to leverage deep learning in functional graph estimation: 1) one constraint is that the data such as EEG and DTI is limited, whereas the neural networks can easy

get overfitting since well developed neural network are data hungry; 2) our task is an inference task that we don't have the ground truth label, and our goal is to infer the underlying structure from multivariate functional data. This conflicts with the design of modern neural network: neural networks need to be trained through auto-differentiable system (for example, TensorFlow, PyTorch, etc.) with supervision; 3) we aim to consume the functional data directly instead of using conventional dimension reduction techniques (ex: FPCA) to train the network end-to-end.

To overcome the first two issues, we leverage the graph estimation framework proposed by Belilovsky et al. (2016) and adapt it to the functional settings. They assume the mapping between covarianve matrix and graph structure is learnable. They propose to use neural network to approximate this mapping and train with synthetic data so that it can produce ground truth, where the synthetic data follows the assumption about the real data and graph structure. Therefore, we convert a inference task to be a prediction task. To input the functional data directly, we leverage the idea of Yao et al. (2021a), which is a neural network based adaptive FPCA. Therefore, our proposed architecture is demonstrated in last panel of Figure 2.1. Instead of separable workflows in the above three methods of Figure 2.1, our workflow is fully neural network based and can be end-to-end trained.

Considering the functional dataset $\boldsymbol{g}(t) = \{g_{ij}(t)\}_{i=1,j=1}^{N,p}$ consisting of N i.i.d. samples with p functional variable $g_{ij}(t)$ individually. Based on (1.6) and (1.7), we have

$$C_{jj'}(s,t) \approx \operatorname{cov}\left(g_{ij}^M(s), g_{ij'}^M(t) \mid g_k^M(\cdot), k \neq j, j'\right) = \sum_{k=1}^{M} \sigma_{jj'k}\phi_k(s)\phi_k(t) \tag{2.25}$$

where $\sigma_{jj'k} = \operatorname{cov}(a_{jk}, a_{j'k} | a_{hk}, h \neq j, j')$, which is the covariance matrix of PCA coefficients projected on kth basis $\phi_k(t)$ and $\Sigma_k = (\sigma_{jj'k})_{p\times p}$. We aim to estimate the connectivity between different nodes based on $C_{jj'}(s,t)$. Assuming the functional data $g_{ij}(t)$ can be lossless expressed through k basis functions, the connectivity could be defined as

$$Y_{jj'} = \begin{cases} 1 & \sigma_{jj'k} \neq 0 \text{ for } \forall k, \\ 0 & \text{otherwise.} \end{cases} \tag{2.26}$$

Among all existing methods, which provide estimator f from Σ to Y, they first calculate the PCA scores based on pre-determined basis functions, then use the covariance of PCA scores as input to

Algorithm 3 Procedure of Generating Synthetic Data Set

for $i = 1, \cdots, N$ **do**
  sample $G_i \sim \mathbb{P}(G)$
  sample $\Theta_k \sim \mathbb{P}(\Theta | G = G_i)$ for $k = 1, \cdots, K$
  for $k = 1, \cdots, K$ **do**
    $A_k \longleftarrow \{\boldsymbol{a}_{ik} \sim N(0, (\Theta_k)^{-1})\}$
  end
  $g_{ij}(t) \longleftarrow \sum_{k=1}^{K} a_{ijk} \cdot \phi_k(t)$ for $i = 1, \cdots, N, j = 1, \cdots, p$, where $\phi_k(t)$ is the pre-specific basis function
  Construct $(\boldsymbol{g}_i(t), Y_i)$ pairs from $(G_i, \boldsymbol{g}_i(t))$
end

infer the graph. In the contrast, by (2.25) and (2.26) we could consider the problem of graph learning by building the mapping from input $\boldsymbol{G}(t)$ to edge connectivity $\mathbf{Y}$, i.e.

$$f : \boldsymbol{G}(t) \to \boldsymbol{Y} \tag{2.27}$$

As we discussed in Section 1.4, the best f is the one with minimum risk $E(L(f(\boldsymbol{G}(t), \boldsymbol{Y})))$. To pick f among possible mappings, we follow the framework for graph discovering proposed by Belilovsky et al. (2016). First, we need to generate a number of multivariate functional dataset with known connectivity $\boldsymbol{Y}$. These graphs may have some specific sparsity or structures. Then the best mapping could be chosen by taking these synthetic data as input. Finally, we apply the mapping f on the real data to get the estimated connectivity $\hat{\boldsymbol{Y}}$. In other words, we convert the graph learning to an edge classification problem based on pre-generated synthetic data. This method is attractive since we have the ground truth, which can be used to fit neural network.

As discussed in Section 1.1, the nonzero element in the precision matrix corresponds to the connection between nodes. According to equation (2.26), edges in the graph are related to covariance of basis coefficients on the basis functions, i.e. $Y_{jj'} = I[\sigma_{jj'k} \neq 0, \exists k]$. Therefore, in this case, determining the underlying graph of the multivariate functional dataset is equivalent to determine the precision matrix Θ_k associated to graph projected on each basis. In summary, we generate $g_{ij}(t) = \sum_{i=1}^{K} a_{ijk} \cdot \phi_k(t)$ for $i = 1, \ldots, N$ and $j = 1, \ldots, p$ to consist $N \times p$ functional dataset, where $\boldsymbol{\phi}(t) = \{\phi_k(t)\}_{k=1}^{K}$ was K-dimensional pre-defined basis functions and a_{ijk} is element of $\boldsymbol{a}_{ik}$ sampled from multivariate Gaussian distribution with covariance matrix Σ_k. $\Theta_k = (\Sigma_k)^{-1}$ determine the graph structure and could be sampled among possible graphs with specific characteristics. The procedure is summarized in Algorithm 3.

The synthetic functional data could be adaptive to any types of domain knowledge or the real world data which is ground truth. For example, we could simulate synthetic data with desired sparsity level, graph structure, and also according to the property of functional data we could use appropriate basis functions. The synthetic data implicitly contains the prior information of graph. The prior information consists of the possible set of mapping f and make empirical risk more closely to the true risk. In the following section, we discuss to design a deep neural network with functional dataset as input to approximate the best edge classifier f to minimize the empirical risk.

2.3.2 Deep Functional Graphical Models

The architecture of our network (See Figure 2.2) is inspired by the properties of functional data illustrated in (2.25) and (2.26). Our input is a multivariate functional dataset and it has two main properties: 1) Infinite dimension. Unlike regular scalar data or time series, functional data such as EEG signal or fMRI is intrinsically infinite dimensional and generated by smooth underlying process. 2) Multivariate. Multivariate functional data is a set of functions for different nodes. The nodes are areas of interest from a space. Nodes may not be isolated and some nodes may form a meaningful subset. The graph information are sufficiently contained in the covariance matrices of basis scores. Our full network is visualized in Figure 2.2 consisting of Basis layer and convolutional layer. The basis layer helps to consume infinite functional data and project the graph information on the adpative basis functions. Further, the convolutional neural network is employed to take the covariance matrices to estimate the edge probability.

In order to make a model to consume the functional data set, we need to have a dimension reduction module like basis expansion. The conventional basis expansion is to project the function onto several pre-selected bases, i.e.,

$$a_{ijk} = \int \beta_k(t) \cdot g_{ij}(t) dt \tag{2.28}$$

and assume the truncated basis coefficients $\boldsymbol{a}_{ij} = (a_{ij1}, \cdots, a_{ijK})$ are lossless contained the information of functional data $g_{ij}(t)$. Unlike traditional pre-selection of basis functions, $\{\beta_k(t)\}_{k=1}^K$ in the architecture are approximated by K multi-layer perceptrons. The details of basis node is illustrated in Figure 2.3. The multi-layer perceptions take time point t, a 1-dimensional scalar value, as input. The inner product of approximated basis function $\hat{\beta}(t)$ and functional data $g_{ij}(t)$ (over identical

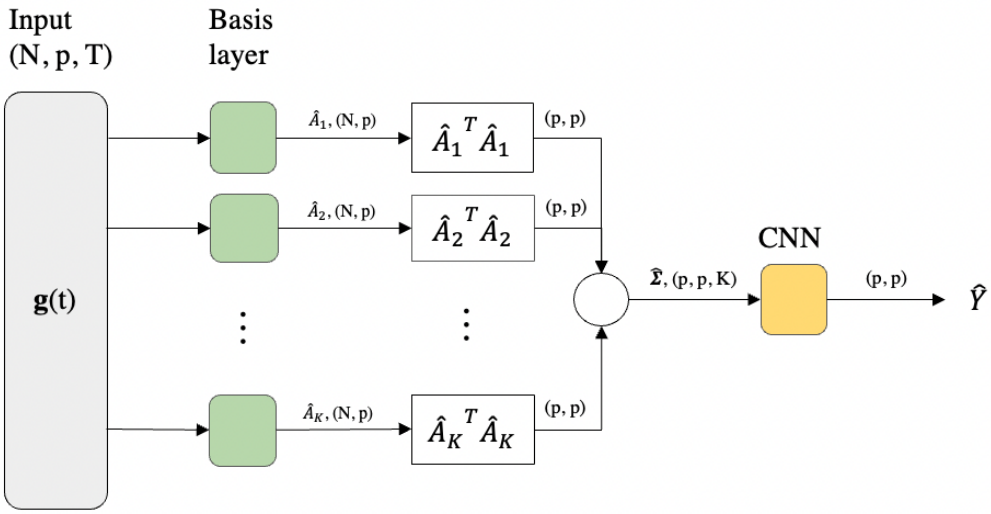

Figure 2.2: Deep FGM overview: the functional data set ($N \times p \times T$) is fed into basis layer. After basis layer, each curve conducts inner product operation with basis function through basis layer, then a aggregated covariance $\hat{\mathbf{\Sigma}} \in \mathcal{R}^{p \times p \times K}$ are convoluted by dilated-2D convolutional layer. Finally, we output the edge estimation based on neural network prediction. Basis layer consists of K basis nodes, which are multi-layer perceptron illustrated in Figure 2.3. "CNN" stands for convolutional neural network which stacks multiple convolutional layer demonstrated in Figure 2.4.

time grid) then produces the estimated basis coefficients. This method was first proposed by Yao et al. (2021a) for functional data regression problem, where regularization term for $\beta_k(t)$ could be added into loss function to achieve sparse and orthogonal bases.

Our architecture differ from other neural network that our input is the whole data set instead of individual data point. In other words, we could understand one functional data set as a data point in our case. Therefore, as the input dimension is $N \times p \times T$, where N is number of replication, p is the number of nodes and T is the number of time points collected for each function. Each function $g_{ij}(t)$ goes through basis layer and then we could have the estimated basis scores $\hat{A} = \{\hat{A}_k\}_{k=1}^K$ with dimension $N \times p \times K$. Further, transformation of $\hat{A}_k^T \hat{A}_k$ are made separately for each basis node output. The aggregated "covariance" $\hat{\mathbf{\Sigma}} = \{\hat{A}_k^T \hat{A}_k\}_{k=1}^K$ with dimension $p \times p \times K$ then is the input for the next convolutional layer.

We use convolutional neural network to take the $\hat{\Sigma}$, which represents the graph information of the input and is sufficient for edge classification based on conventional graph learning methods (Zhu et al., 2016; Qiao et al., 2019; Zhang et al., 2021a; Niu et al., 2021). The motivation of using convolutional neural network is that $\hat{\mathbf{\Sigma}}$ composes of K 2D arrays containing covariance matrix based

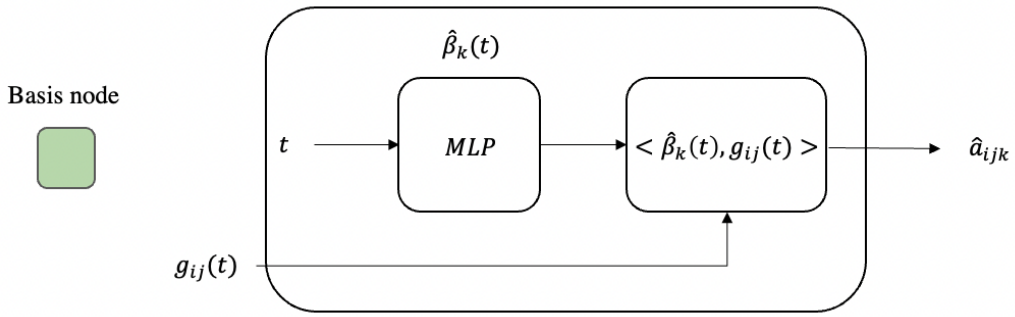

Figure 2.3: The kth basis node in the basis layer, where MLP indicates multi-layer perceptron with 3 hidden layers. $\langle \cdot, \cdot \rangle$ denotes inner product operation (Yao et al., 2021a). The input of the basis node is the function $g_{ij}(t)$, and its output is the basis coefficient corresponding to the learned basis.

on K basis functions. This input is similar with image data, which is a 3 slices of 2D array with R, G, B channels. In addition, convolutional neural network has characteristics of local connections, shared weights and stacked multiple layers (LeCun et al., 2015). We found these properties are important for graph learning, since we should leverage the local connection characteristic. Additionally, convolutional neural network use small size of shared kernels, so that the number of parameters is highly reduced compared with traditional spectral graph theory (Chung and Graham, 1997). Otherwise, the number of parameters would be exponentially increased with number of nodes, which is computationally intractable. Furthermore, convolutional neural network could contain multiple layers which capture the hierarchical patterns of graphs. Therefore, we apply convolutional neural network to graph structure learning. Note that adopting convolutional neural network to learn graph structure is not first proposed in our work. Scarselli et al. (2008) proposed the concept of graph convolutional neural network (GCN), which is very popular to process graph data. Belilovsky et al. (2016) also consider convolutional neural network as graph estimator for multivariate scalar data case. Their work use 2D convolution neural network instead of standard graph convolution network since they pre-compute the covariance matrix as data representation. To be more specific, $\hat{\mathbf{\Sigma}}$ flows into a standard convolutional neural network, shown in Figure 2.4. We use multiple layers of 2D convolution with RELU activation and dropout allowing us for fast training and avoid overfitting. Lastly, the feature representation is fed into a last 1×1 convolutional layer with sigmoid activation function to predict edges as binary classification task, which classify non-zero entries in the desired precision matrix.

The architecture (as shown in Figure 2.2) consists of two types of layers: a basis layer and a

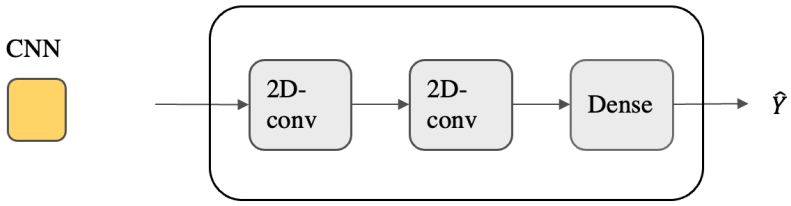

Figure 2.4: A convolutional module stacked with 2 hidden convolution layers and a dense layer with sigmoid activation function. The input of the CNN module is the covariance of basis coefficients $\hat{\boldsymbol{\Sigma}} \in \mathcal{R}^{p \times p \times K}$, and its output is the predicted edge probability.

convolutional layer. The input is the functional data set $\boldsymbol{g}(t) \in \mathbb{R}^{N \times p \times T}$, where N represents number of samples, p denotes number of nodes in each graph, and T denotes the number of measurement over time. We fetch $\boldsymbol{g}(t)$ into the basis layer, where basis layer is shown in Figure 2.3. We leverage basis layer implementation from Yao et al. (2021a). The basis layer functions as dimension reduction and the output is $\{A_k\}_{k=1}^{K}$ that represent the estimated basis scores based on the adaptive basis functions. Compare with conventional methods like PCA scores, our learned basis function offer the powerful non-linear approximation capability and it is fully data-driven.

Then, we transform the basis layer output $\hat{A}_k$ to be a covariance matrix by $\hat{A}_k^T \hat{A}_k$. We fetch the covariance matrix into a multiple layers of convolutional operations. The convolution neural network aims to predict the graph precision matrix, where each element in the precision matrix represents the corresponding edge connection. We stack multiple 2D convolution layer as shown in Figure 2.4. The detailed forward pass procedure is described in Algorithm 4.

During training phase, our goal is to reconstruct target graph from our synthetic data. We use binary cross-entropy loss as following:

$$L(f_W(\boldsymbol{g}(t)), Y) = \sum_{i \neq j} (Y^{ij} \log(f_W^{ij}(\boldsymbol{g}(t))) + (1 - Y^{ij}) \log(1 - f_W^{ij}(\boldsymbol{g}(t)))) \tag{2.29}$$

where Y is the ground truth of edge connection and f_W indicates the non-linear mappings between $\boldsymbol{G}(t)$ and Y which is approximated through our proposed Deep FGM. In order to obtain the ground truth, we generate synthetic data to emulate data sparsity and Gaussianity with different graph topology structures (ex: star, cluster, etc.) since deep models are data hungry. The detailed data generation flow can be found at Algorithm 3. Note that it is obvious that the model generalization is heavily relied on synthetic data generation process. Therefore, any prior knowledge can be injected

Algorithm 4 Deep Functional Graphical Model Forward Pass

Input: multivariate functional dataset $\{g_{ij}(t_1), \ldots, g_{ij}(t_T)\}$, for $i = 1, \ldots, N, j = 1, \ldots, p$.
Output: estimation for edges $\hat{Y}$.
Parameters: kth Basis layer neural network $f_{W^{\beta_k}}$ with weights W^{β_k}, for $k = 1, \ldots, K$, convoluational layer f_{W^c} with weights W^C, integration weights $w_1, \ldots, w_T$ for trapezoid rule;

```
for i = 1, ... N do
    for j = 1, ..., p do
        for k = 1, ..., K do
            â_ijk ⟵ < β_k(t), g_ij(t) > = Σ_{l=1}^{T} w_l · f_{W_k^β}(t_l) · g_ij(t_l)
        end
    end
end
```

$\hat{A}_k \longleftarrow (\hat{a}_{ijk})_{N \times p}$, for $k = 1, \ldots, K$
$\hat{\mathbf{\Sigma}} \longleftarrow (\hat{A}_1^T \hat{A}_1, \ldots, \hat{A}_K^T \hat{A}_K)$
$\hat{Y} = f_{W^c}(\hat{\mathbf{\Sigma}})$, where $\hat{\mathbf{\Sigma}}$ has dimension $p \times p \times K$.
return $\hat{Y}$

into synthetic data generation process to guarantee in-distribution prediction during testing. Based on Equation 2.29, we use batch gradient descent as optimizer to train our network parameters using Tensorflow (Abadi et al., 2015). Detailed training configurations can be found in Chapter 4.

After training, we obtain the optimized network parameters. During inference phase, we fetch the raw functional graphical data, and our network propagate it through adaptive basis layer to generate basis scores. Then, a covariance matrix which is transformed from basis scores flow through the convolution neural network to infer the corresponding graph.

Chapter 3

Causal Inference with Functional Confounding

In this chapter, we discuss the causal effect estimation with the functional covariates. First, we introduce background and current development of causal inference with application of machine learning in Section 3.1. In Section 3.2, we review the potential outcome framework and introduce two basic causal models, S-learner and T-learner. Further, In Section 3.3 the functional regression which is the basic method to handle functional covariate is discussed. Lastly, we propose causal models for functional confounders with implementation of neural networks in Section 3.4 and validate the proposal in Section 4.3 through simulation studies.

3.1 Causal Inference with Machine Learning

It is well known that the causality is different than statistical correlation. The causality can imply correlation, but in the contrast, correlation cannot infer causality. If two variables, say X and Y, are correlated, it may have following possibilities: X causes Y; Y causes X; X and Y have common causes but they do not cause each other; etc. Let's illustrate differences between causality and correlation through the famous ice cream and drowning example. In the most cities, the daily sales of ice cream display an increasing trend with the daily rates of drowning. Can we conclude that ice cream leads to the drowning? Apparently, such conclusion does not logically make sense. In fact,

people have more ice cream when the temperature is increasing, and people are more willing to go swimming when temperature is high, which causes the number of drowning increasing. Therefore, there exists no direct causal relationship between ice cream and drowning rate.

Pearl and Mackenzie (2018) describes problems of causal inference as a hierarchy of three types with increasing difficulty (often called "Causal Ladder"): 1) prediction, 2) intervention, and 3) counterfactuals. The first ladder is prediction, which is a typical task of the supervised machine learning, i.e. estimating $P(y|x)$. $P(y|x)$ is a conditional distribution which can be calculated from $P(y|x) = \frac{P(x,y)}{P(x)}$ which indicates the association between X and Y from observable data. However, if two variables are correlated, we cannot infer that the change of X determines the change of Y. The second level is intervention. In the concept of intervention, we are interested in that whether the outcome variable Y changes when we perturb the variable of interest X, i.e. estimating do-calculus $P(y|do(x))$ (Pearl, 2009) which is different with $P(y|x)$. More specifically, by performing an intervention on a system we could change its distribution, and thereby the original distribution $P(y|x)$ may not be valid. Considering the above example, let X be the ice cream sale rate and Y be the drowning rate in a particular city, and we assume the drowning rate functions properly $P(y|x)$ should be a unimodal distribution centered around βx. $P(y|do(x))$ would not be dependent on x and is generally the same as $p(y)$, which is the marginal distribution of the drowning rate. The is because an exogenous intervention of ice cream sale rate (say, ice cream duty free policy) won't actually cause the drowning rate to be increased or decreased. Since that the do caculus without intervention is equivalent to the original distribution, the intervention subsumes the prediction, justifying its higher level in the ladder. At the highest level – counterfactuals, we are allowed to ask questions like "Given the variables Z were observed to be z, if variables X were forced to be x then how likely is that variables Y would be y". Here the question is conditional on the new information Z. For example, "Given that Ben and John did not attend the seminar yesterday, if Ben joined the seminar, how likely John would also come?" Note that by setting Z as empty, we recover a family of interventional questions. Thus, the counterfactuals subsume the intervention, which is categorized as the top of the causal ladder. As our modeling evolving from prediction to counterfactual along the causal ladder, we need more sophisticated data and methodologies.

In applications where we try to control or choose x based on the conditions we estimated (the interventional level), we should use $P(y|do(x))$ instead of $P(y|x)$. For example, we are not merely interested in the association between treatment X and outcome Y. We prefer to measure

the effect of the treatment and proactively choose the treatment given our understanding of how it affects the outcome. Similar applications occur in a variety of the real-world scenarios, such as online advertising, online recommendation system, reinforcement learning, etc.

The effective way to estimate $P(y|do(x))$ is randomized controlled experiments or A/B tests where x is controlled. In the randomized experiments, we measure the outcome differences between treatment and control group to infer causality. Nevertheless, the true randomized controlled experiments may be impossible or at least impractical in many situations. For example, it is unethical in the experiment that forcing some subjects to smoke and the other to use placebo to study the effect of smoking on health. On the other hand, the randomized experiments only study the average of subjects, and it is not applicable to explain the causal effect for individuals. Instead of randomized experiments, we can estimate $P(y|do(x))$ through observational study. That means we have no control or intervention over subjects, and we simply observe the subjects and collect the observational data. But the core issue is we do not know the counterfacutal outcome. In other words, we do not know whether the subject would have a different outcome if it took a different treatment.

There are mainly two research efforts on causal inference from observational data: potential outcome framework (Rubin, 1974) and structural causal models (Pearl, 2009). The potential outcome model (also called the Neyman-Rubin causal model) is the most widely used framework for causal inference. In the randomized experiments, we use expected differences of outcome between treatment and control group to measure causal effect. But in observational study, it is impossible to observe both potential outcome simultaneously. This "missing data" issue is a fundamental problem in causal inference through observational study. Under the potential outcome framework, the missing counterfactuals are estimated from observational data then causal effect is calculated. Structural causal model is associated with a directed acyclic graph (DAG), which is a probabilistic graphical model that encodes the flow of data generating process. Nodes in the graph are divided into endogenous and exogenous variables, while each edge represents causal relationship among nodes modeled by simultaneous structural equations (Pearl, 2009). We can study the dependence or conditional dependence between variables through DAG based on observational data.

With the emerging development of machine learning, more powerful machine learning methods are applied in causal inference to measure the causal effects, understand the heterogeneous impact of interventions, and design targeted assignment mechanisms. These machine methods includes regularization methods (LASSO) (Bloniarz et al., 2016), tree-based methods (CART) (Athey and

Imbens, 2016), ensemble learning (random forest) (Wager and Athey, 2018), Bayesian nonparmatric (BART, Gaussian Process, Dirichlet Process) (Chipman et al., 2010; Alaa and Van Der Schaar, 2017; Schwartz et al., 2011), and also popular deep learning methods such as multi-layer perceptron (MLP), convolutional neural network, Recurrent neural networks (RNN), graph neural network (GNN), generative adversarial network (GAN), representation learning, reinforcement learning, etc. (Koch et al., 2021). Recalling that machine learning models are initially designed for prediction tasks with large and complex data, where it is not inherently designed for causal inference, one has to leverage existing machine learning tools in a delicate way so that we could overcome key challenges in causal inference, such that overlap, unconfoundedness, balance, etc (Yao et al., 2021b).

Machine learning as a tool extends the causal inference on complex data, such as text, time series, networks, and images. Bica et al. (2020) introduce the Counterfactual Recurrent Network (CRN), which is a sequence-to-sequence model to estimate time-varying treatment effects by RNN. Cheng et al. (2021) propose to estimate the long-term treatment effect through representation learning and two heads of CFRNet (Shalit et al., 2017) with an RNN by conditioning on time-varying confoundings. In network domain, Ma and Tresp (2021) explore the causal effect estimation within the TARNet framework by learning node representation via GNN, which aggregates treatment information of local neighbors and the graph structure separately. In summary, variety of research on latent confounding encoded in texts, images, graph and so on, are investigated in the causal inference area based on CNN, RNN, GNN, even large pre-trained language model (Devlin et al., 2018). More detailed examples can be found in Koch et al. (2021).

Within the existing framework of causal models, for time-varying data, most studies explore the causal effects in the assumption of data stationarity by using time-series analysis (Lim, 2018; Bica et al., 2020; Cheng et al., 2021). However, existing methods lacks of considering non-stationary cases, such as functional data ECG, EEG, fMRI, etc. Hence, the issue of how to estimate the causal effect and recover the individual treatment effect in the presence of functional confoundings is still under-developed, to our best knowledge.

In this work, we propose to investigate the causal effect estimation conditioning on the functional confounders under the general interference through functional data analysis. First, we review the potential outcome framework and the three key assumptions for causal inference, and functional regression in functional data analysis. Then we explore three causal models S-learner, T-learner and TARNet, integrating with the basis expansion in the functional data analysis. Our

experimental results in Section 4.3 show that the proposed methods could accurately estimate the causal effects.

3.2 Potential Outcome Framework

As we know, the ideal scenario to measure causal effect is applying different treatments to the same group. However, such scenario can only be achieved by a randomized controlled experiment. In the randomized experiment, we control the treatment assignment such that

$$\{Y_i(0), Y_i(1) \perp\!\!\!\perp W_i\}, \tag{3.1}$$

where we define outcome $Y_i \in \mathbb{R}$ and treatment assignment $W_i \in \{0, 1\}$. In the randomized experiment we can have that

$$\begin{aligned} ATE &= E[Y_i(1)] - E[Y_i(0)] \\ &= E[Y_i(1)|W_i = 1] - E[Y_i(0)|W_i = 0] \\ &= E[Y_i|W_i = 1] - E[Y_i|W_i = 0] \end{aligned} \tag{3.2}$$

where ATE stands for the average treatment effect, and our goal is to estimate the ATE. According to the equation above, it is highly impossible to have both $Y_i(0)$ and $Y_i(1)$ at the same time, but we can consistently estimate ATE in a randomized experiment. In this work, we mainly focus on average treatment effect estimation in the presence of functional confounders, as shown in Figure 3.1. For example, the outcome Y_i could be the quality-adjusted life years, the treatment W_i is smoking quitting therapy and the covariantes $X_i(t)$ are electrical medical records over a time period including weight, blood pressure etc. In this case, a therapist may prescribe the treatment to patients with weights or blood pressure change, and we want to study the causal effects in the setting where the treatment assignment may be associated with time-varying pre-treatment covariantes $X(t)$. If we directly use the differences of the average of observed outcome between treated and control group, it may lead to spurious effects like the famous Simpson's paradox. In addition, the confounders affect the treatment assignment that lead to selection bias, which is a phenomenen that the distribution of the treatment and control group are not matched. Thereby, selection bias would also bias the counterfactual outcome estimation. How to ease the bias caused by confounding variables is the one of the most important topics in causal inference, a procedure called deconfounding. If treatment

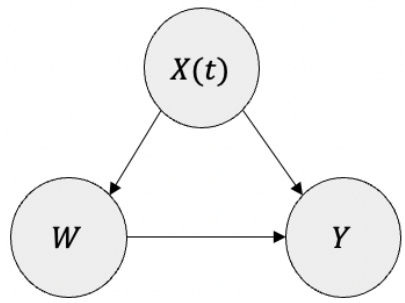

Figure 3.1: The three-variable causal DAG with functional confounders. W represents the treatment assignment indicator; $X(t)$ represents the functional covariate; Y represents the potential outcome.

assignment W_i is as good as random conditioning on X, it is sufficient to estimate the average causal effect by controlling covariates, i.e.,

$$\{Y_i(0), Y_i(1) \perp\!\!\!\perp W_i\} | X_i(t) \tag{3.3}$$

This assumption is often named as unconfoundedness assumption (Rosenbaum and Rubin, 1983). Besides, two other commonly used assumptions are stable unit treatment value assumption (SUTVA) and positivity. SUTVA contains consistency assumption and non-interference assumption: consistency assumes that the observed treatments is the same as the potential interventions we are interested in; the non-interference assumes the independence of each subject. On the other hand, positivity assumes that for any value of covariate, treatment assignment is not deterministic, because it will be meaningless to study the treatment effect with deterministic treatment assignment. Given the unconfoundedness we can express the ATE in terms of conditional outcomes as

$$\begin{aligned} ATE &= E[Y_i(1) - Y_i(0)] \\ &= E[E[Y_i(1)|X_i(t)] - E[Y_i(0)|X_i(t)]] \\ &= E[E[Y_i(1)|X_i(t), W_i = 1] - E[Y_i(0)|X_i(t), W_i = 0]] \\ &= E[\mu_{(1)}(X_i(t)) - \mu_{(0)}(X_i(t))] \end{aligned} \tag{3.4}$$

where $\mu_{(w)}(X_i(t)) = E[Y_i|X_i(t), W_i = w]$. This directly suggests that estimation strategy, i.e.,

$$A\hat{T}E = \frac{1}{n}\sum_{i}^{n}\left(\hat{\mu}_{(1)}(X_i(t)) - \hat{\mu}_{(0)}(X_i(t))\right) \tag{3.5}$$

We could consider machine learning methods such as linear regression, random forest, or neural networks to fit the potential outcome functions $\mu_{(w)}(X_i(t))$. Usually this type of model deconfounds through regression on covariates and blocks the back-door path between confounders and outcome, also called outcome modelling. Recall in the Figure 3.1, the direct path from W to Y is the causal path, whereas the back-door path is the path "$W - X(t) - Y$".

In contrast, we could also block the back-door path between treatment and confounders, which introduces the treatment modelling. The most important tool in treatment modelling is propensity score,

$$e(x(t)) = P(W_i|X_i(t) = x(t)) \tag{3.6}$$

i.e., the propensity score is defined upon conditional probability of being treated. We can use the variability of the propensity score to measure how far we are from a randomized experiment. Under the unconfoundedness assumption, we can express the average treatment effect as

$$\begin{aligned} ATE &= E[Y_i(1) - Y_i(0)] \\ &= E\left[\frac{W_i Y_i}{e(X_i(t))} - \frac{(1-W_i)Y_i}{1-e(X_i(t))}\right] \end{aligned} \tag{3.7}$$

which implies the following unbiased inverse propensity weighted estimator for the average treatment effect, i.e.,

$$\hat{ATE}_{IPW} = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{W_i Y_i}{\hat{e}(X_i(t))} - \frac{(1-W_i)Y_i}{1-\hat{e}(X_i(t))}\right). \tag{3.8}$$

where $\hat{e}(X(t))$ could be estimated through machine learning methods like logistic regression, or neural networks.

However, the accuracy of potential outcome estimation and especially propensity score estimation will affect the average treatment effect estimation (Imai and Ratkovic, 2014). Double robust modelling (Funk et al., 2011) is a method that combines estimates from both the outcome modelling $\hat{\mu}_{(w)}(x(t))$ and treatment modelling $\hat{e}(x(t))$, so that the estimation is robust even when one of the outcome models or propensity score is not appropriate. This methods is also named as augmented inverse propensity weighting (AIPW). Given estimates $\hat{\mu}_{(w)}(x(t))$ and $\hat{e}(X_i(t))$ from any machine

learning methods, the estimated average treatment effect can be expressed as

$$\begin{aligned} \hat{ATE}_{AIPW} = \frac{1}{n}\sum_{i=1}^{n} \big(\hat{\mu}_{(1)}(X_i(t)) - \hat{\mu}_{(0)}(X_i(t)) + \\ \frac{W_i}{\hat{e}(X_i(t))}\big(Y_i - \hat{\mu}_{(1)}(X_i(t)\big) - \frac{1-W_i}{1-\hat{e}(X_i(t))}\big(Y_i - \hat{\mu}_{(0)}(X_i(t))\big) \end{aligned} \tag{3.9}$$

In the above equation, the term $\frac{1}{n}\sum_{i=1}^{n}\big(\hat{\mu}_{(1)}(X_i(t)) - \hat{\mu}_{(0)}(X_i(t))\big)$ is the direct outcome estimator and the rest term is an IPW estimator applied to the residuals. In another words, AIPW employs propensity score re-weighting on the residuals to debias the direct outcome estimate. For more details about the application of machine learning and deep learning methods in causal effect estimation, we refer to Chernozhukov et al. (2018); Farrell et al. (2021).

Instead of using population-level ATE, we are also interested in how treatment effects vary across people. The individual treatment effect of the ith subject is defined as

$$ITE_i = Y_i(1) - Y_i(0). \tag{3.10}$$

It is almost impossible to estimate ITE_i since we cannot have both potential outcomes at the same time. So in practice, to study the treatment heterogeneity we collect some covariates and see how the treatment effects change with them, i.e., the conditional average treatment effect (CATE)

$$CATE = E[Y_i(1) - Y_i(0)|X_i(t) = x(t)]. \tag{3.11}$$

CATE is also called heterogeneous treatment effect and represents the average treatment effect among all subjects who have the same covariate values.

Under unconfoundedness, we can express the CATE as

$$\begin{aligned} CATE &= E[Y_i(1) - Y_i(0)|X_i(t) = x(t)] \\ &= E[Y_i(1)|X_i(t) = x(t), W_i = 1] - E[Y_i(0)|X_i(t) = x(t), W_i = 0] \\ &= E[Y_i|X_i(t) = x(t), W_i = 1] - E[Y_i|X_i(t) = x(t), W_i = 0] \\ &= \mu_{(1)}(x(t)) - \mu_{(0)}(x(t)). \end{aligned} \tag{3.12}$$

The most straightforward way to estimate the $CATE$ is to use machine learning to predict $\mu_{(1)}(x(t))$

and $\mu_{(0)}(x(t))$, then

$$C\hat{ATE} = \hat{\mu}_{(1)}(x(t)) - \hat{\mu}_{(0)}(x(t)). \tag{3.13}$$

In this case, if we fit a single model $\hat{\mu}(X_i(t), W_i)$ to all the data and $C\hat{ATE} = \hat{\mu}(X_i(t), 1) - \hat{\mu}(X_i(t), 0)$, this is called **S-learner** (Künzel et al., 2019); if we fit separate models on the treated and control models, i.e., $\hat{\mu}_{(1)}(x(t))$ and $\hat{\mu}_{(0)}(x(t))$ and $C\hat{ATE} = \hat{\mu}_{(1)}(x(t)) - \hat{\mu}_{(0)}(x(t))$, this is called **T-learner** (Künzel et al., 2019; Johansson et al., 2020).

In order to use S/T-learner to learn the heterogeneous treatment effect, we need to choose a method for predicting Y from $X(t)$. We could use functional regression in the functional data analysis, which will be discussed in next section.

3.3 Functional Regression

We review the most commonly used functional regression – Functional Linear Models (FLM) as following (Ramsay and Dalzell, 1991; Hastie and Mallows, 1993). The FLM is an extension of linear model to functional covariates, i.e.,

$$Y_i = B_0 + \int X_i(t)B(t)dt + \epsilon_i \tag{3.14}$$

where B_0 is the intercept, $Y_i \in \mathbb{R}$ is scalar response, $X_i(t) \in L(\mathcal{T})$ is a functional covariate, $B(t) \in L(\mathcal{T})$ is coefficient function, and residual term $\epsilon_i \sim N(0, \sigma^2)$. By the KL expansion (1.5), $X_i(t) = \sum_{k=1}^{\infty} a_{ik}\phi_k(t)$, and $B(t) = \sum_{k=1}^{\infty} b_k\phi_k(t)$. Then FLM can be expressed as

$$Y_i = B_0 + \sum_{k=1}^{\infty} b_k a_{ik} + \epsilon_i. \tag{3.15}$$

Therefore, the general functional regression model can be written as

$$Y_i = f(\boldsymbol{a}_i) + \epsilon_i. \tag{3.16}$$

where $\boldsymbol{a}_i = (a_{i1}, a_{i2}, \cdots)$ We can observe that functional regression can be converted to regression on the basis representation. The basis functions here could be Fourier basis functions, B-spline, or functional principal component calculated from spectral decomposition of covariance function of $X(t)$. We discuss the drawback of the pre-determined basis selection in Section 2.1.2. So in the

following proposed work, we employ the similar strategy in Section 2.3, i.e., adaptive basis expansion through neural network (Yao et al., 2021a).

3.4 Causal Models with Functional Confoundings

In this section, we explore three causal models, i.e., S-learner, T-learner and TARNet (Shi et al., 2019) for functional confounders. As we discussed in section 3.2, S-learner is an outcome model with a single prediction for all data. The S-learner takes covariate $X(t)$ and treatment W as input, and uses the single prediction to produce two estimated potential outcomes: $\hat{\mu}(x(t), W)$ and $\hat{\mu}(x(t), 1-W)$. To produce the counterfactual potential outcome, we need to set the treatment input with $1-W$. Then we can simply estimate the CATEs based on $\hat{\mu}(x(t), 1) - \hat{\mu}(x(t), 0)$. To handle the functional covariate, we use the adpative basis (Yao et al., 2021a) then use the vector of basis scores a_{ik} as input. If we use the multi-layer perceptron to implement $\hat{\mu}(\cdot, \cdot)$, the architecture for S-learner with functional confounder is shown in Figure 3.2. The basis layer approximates each basis function through neural network and produces a vector of basis scores $\boldsymbol{a}$, then we concatenate treatment W and $\boldsymbol{a}$ together and feed them into a standard multi-layer perceptron network. To learn basis functions and estimate outcomes, the loss function aim to minimize the mean squared error:

$$L_s\left(Y, \mu(X(t), W)\right) = MSE\left[Y, \mu(X(t), W)\right] \tag{3.17}$$

where the networks weights are updated based on backpropogation algorithms. The regularization term for basis orthogonality and sparsity could be added with objective function. Details are illustrated in Yao et al. (2021a). During testing phase, we compute CATEs based on two estimated outcome by flipping input treatment signals.

An improved version is T-learner which is based on two heads, that fit separate models on the treated and controls. Similar to S-learner, T-learner produces two estimated potential outcome $\hat{\mu}_0(X(t))$ and $\hat{\mu}_{(1)}(X(t))$, but T-learner use separate models to approximate each outcome. We use the adaptive basis to convert the functional covariate into basis representation the same as above S-learner. The transformed functional data are fed into two independent models (two multi-layer perceptrons). Similarly, we implement it through neural network minimizing the MSE loss between predicted and observed outcomes. In order to train two models separately, we set $W = 1$, where the

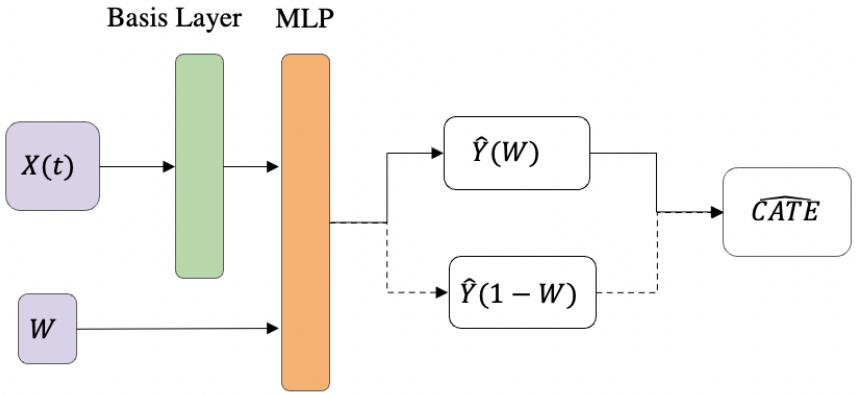

Figure 3.2: S-learner with functional confounder. W, $X(t)$ and $\hat{Y}(\cdot)$ represent treatment indicator, functional covariate and corresponding outcomes respectively. The details of basis node is illustrated in Figure 2.3. "MLP" represents the regular multi-layer perceptron.

head for $1-W$ would be disabled, and vice versa. The architecture of the T-learner with functional confounder is illustrated in Figure 3.3. The joint loss function for T-learner is

$$\begin{aligned} L_t(Y, \mu_{(0)}(X(t), 0)) &= MSE[Y, \mu_{(1)}(X(t), 1)] + MSE[Y, \mu_{(0)}(X(t), 0)] \\ &= \frac{1}{n}\sum_{i=1}^{n}(W_i(Y - \hat{Y}_{i1})^2) + ((1 - W_i)(Y - \hat{Y}_{i0})^2). \end{aligned} \tag{3.18}$$

In the testing phase, T-learner produces both treated and control predictions independently, then we could use their differences to estimate the heterogeneous treatment effect.

Lastly, TARNet proposed by Shi et al. (2019) extends based on T-learner with an extra representation layer as shown in Figure 3.3. The modification from T-learner to TARNet is straightforward: the TARNet aims to learn a shared representation function $\Phi(X(t))$ (also called "embedding" in deep learning terms) where data are balanced in the representation space. In other words, the distributions of the representations $\Phi(X(t)|W = 0)$ and $\Phi(X(t)|W = 1)$ are similar to each other after de-confounding. In our implementation, we simply insert a shared multi-layer perceptron network in front of two outcome network. We use the same training objective and prediction procedure as T-learner. As we show in our experiments, TARNet outperform S-learner and T-learner in most cases, demonstrating that learning a good shared representation of treatment and control groups is essential for causal effects estimation (Shi et al., 2019).

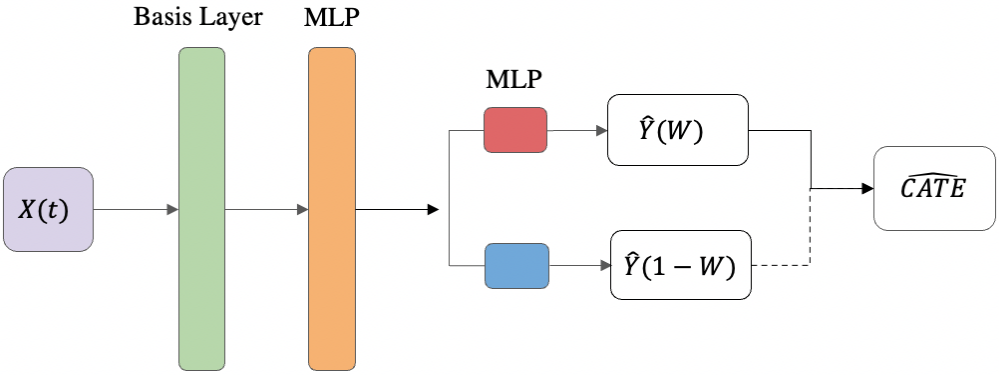

Figure 3.3: TARNet architecture for causal inference with functional input. Note that T-learner has the same architecture as TARNet except the MLP module. Both red and blue heads are implemented by multi-layer perceptron, but they have different pairs of parameters. W, $X(t)$ and $\hat{Y}(\cdot)$ represent treatment indicator, functional covariate and corresponding outcomes respectively.

Chapter 4

Numerical Experiments

In this chapter, we benchmark proposed approach and baseline based on simulation dataset. In Section 4.1, we first designed simulation studies to assess the performance of posterior inference using the Bayesian functional graphical lasso and horseshoe outlined in Section 2.2. This allows us to compare the frequentist fglasso of Qiao et al. (2019), Bayesian fglasso, and functional graphical horseshoe to each other across a variety of scenarios. We assess classification accuracy and fidelity of the estimates of both the zero and non-zero entries of the precision matrices in Section 4.1. As a matter of practicality for researchers interested in implementing the proposed techniques, we also compare the computational expense associated with each procedure across different computing environments (`R`, `NumPy`, and `TensorFlow`). In Section 4.2, we compare the proposed deep model with the frequentist fglasso of Qiao et al. (2019) and Bayesian functional graphical horseshoe in five different settings corresponding to different types of graph structures. We evaluate the performance of proposed deep model through the area under the ROC curves. Lastly, to benchmark the proposed causal models in Section 3.4, we simulate synthetic and semi-synthetic data with ground truth of treatment effect. We evaluate our methods on ATE, CATE, precision estimation of Heterogeneous Effect (PEHE) and test MSE in three senarios, i.e., randomized trials, observational study with selection bias as well as observational study under complex treatment heterogeneity.

4.1 Experiments for Bayesian Functional Graphical Models

For our simulation studies, we considered different sample sizes ($N = 5, 20, 100, 200$), graph sizes ($p = 10, 30, 50$), two different types of networks, and both sparse- and dense-sampled functions. Similar to the simulation studies considered by Qiao et al. (2019), we simulate functional data with $g_{ij}(t) = \mathbf{s}(t)^T \boldsymbol{\delta}_{ij},\ i = 1, \ldots, n,\ j = 1, \ldots, p$, where $\mathbf{s}(t) \in L(\mathcal{T})^5$ contains the first five Fourier basis functions, and $\boldsymbol{\delta}_{ij} \in \mathbb{R}^5$ is a zero mean Gaussian random vector. Hence, $\boldsymbol{\delta}_i = (\boldsymbol{\delta}_{i1}^T, \ldots, \boldsymbol{\delta}_{ip}^T)^T \in \mathbb{R}^{5p}$ follows a multivariate Gaussian distribution with covariance matrix $\boldsymbol{\Sigma} = \boldsymbol{\Theta}^{-1}$, where the underlying graph is determined by the sparsity pattern of $\boldsymbol{\Theta}$. We consider here two types of networks:

- *Network 1*: A block banded matrix $\boldsymbol{\Theta}$ with $\boldsymbol{\Theta}_{jj} = \mathbf{I}_5$, $\boldsymbol{\Theta}_{j,j-1} = \boldsymbol{\Theta}_{j-1,j} = 0.4\mathbf{I}_5$, and $\boldsymbol{\Theta}_{j,j-2} = \boldsymbol{\Theta}_{j-2,j} = 0.2\mathbf{I}_5$ for $j = 1, \cdots, p$, and 0 elsewhere. The network results in each node being connected to its immediate neighbors, and weaker connection to its second-order neighbors.

- *Network 2*: For $j = 1, \ldots, 10, 21, \ldots, 30, \ldots,$ the corresponding submatrices in $\boldsymbol{\Theta}$ are the same as those in Network 1 with $p = 10$, indicating every alternating block of 10 nodes are connected as Network 1. For $j = 11, \ldots, 20, 31, \ldots, 40, \ldots,$ we set $\boldsymbol{\Theta}_{jj} = \mathbf{I}_5$, so the remaining nodes are fully isolated.

For each network, we generate n realizations of $\boldsymbol{\delta} \sim N_{5p}(\mathbf{0}, \boldsymbol{\Theta}^{-1})$. The observed data are then generated as $h_{ijk} = g_{ij}(t_{ik}) + e_{ijk}, \quad e_{ijk} \sim N(0, 0.5^2), \quad k = 1, \ldots, T$, where subject i was observed at time points $t_{i1}, \ldots, t_{iT} \in [0, 1]$. We consider two sampling schemes from the functions:

- *Dense design with equally spaced measurements*: Each function was recorded on a regular grid between 0 and 1, i.e., $t_{i1} = 0, \ldots, t_{iT} = 1$ and $T = 100,\ i = 1, \ldots, N$.

- *Sparse design with irregularly-spaced measurements*: Each function was recorded randomly; i.e., t_{ik} are drawn randomly between 0 and 1 for $k = 1, \ldots, 9$.

Our proposed graphical models work with any choice of basis representation, but we choose the data-driven functional PCA approach, due to the mean-square optimality discussed in Section 1.2 and the smoothness of the simulated data. Consistently, we also implement FPCA in our application part. Thus, to implement any of the approaches considered, we need to compute the first M estimated principal components scores of g_{ij}. We use the PACE algorithm (Yao et al., 2005) for the irregularly sampled setting via the `fdapace` package in `R` (Carroll et al., 2020b).

Network 1, $p = 10$, dense data	N = 20	N = 100	N = 200
Bayes	0.65 (0.10)	**0.88** (0.04)	**0.94** (0.03)
FGM	**0.66** (0.06)	0.84 (0.03)	0.90 (0.03)

Network 1, $p = 10$, N = 100	Dense	Sparse
Bayes	**0.88** (0.04)	**0.83** (0.04)
FGM	0.84 (0.03)	0.80 (0.04)

Network 1, $N = 100$, dense data	p = 10	p = 30	p = 50
Bayes	**0.88** (0.04)	**0.86** (0.02)	0.83 (0.02)
FGM	0.84 (0.03)	0.84 (0.02)	**0.85** (0.02)

$p = 10$, $N = 100$, dense data	Network 1	Network 2
Bayes	**0.88** (0.04)	0.91 (0.04)
FGM	0.84 (0.03)	**0.92** (0.04)

Table 4.1: The mean area under the ROC curves for Bayesian fglasso and FGM. Standard errors are shown in parentheses.

For the regularly sampled case, we use ordinary singular value decomposition. We determine the truncation level M in (1.7) using the minimum number of principal component to capture 95% of the variability over all nodes for the SVD method and PACE algorithm. For both the Bayesian functional graphical lasso and the Bayesian functional graphical horseshoe, a total of 10,000 MCMC iterations are performed after 1000 burn-in iterations. Convergence is assessed via trace plots of randomly selected elements of $\mathbf{\Theta}$.

We first compare the frequentist fglasso to our proposed Bayesian fglasso model with a fixed regularization parameter λ. The primary differences then are the point estimates of $\mathbf{\Theta}$ (MAP estimate versus posterior mean) and the thresholding rule used to select the graph. With MAP estimation, the zeros are automatically produced as part of the optimization. For the Bayesian procedure, elementwise equal-tailed credible intervals are constructed from the MCMC output, whence elements of $\hat{\mathbf{\Theta}}$ are selected via those intervals that do not contain zero. Then $\|\hat{\mathbf{\Theta}}_{ij}\|_F$ is the estimate for edge between node i and j, which follows Qiao et al. (2019). For both methods and for a grid of λ values, we compute the true positive rate, $TPR_\lambda = TP/(TP + FN)$, and false positive rate, $FPR_\lambda = FP/(FP+TN)$, where TP and TN stand for true positives and negatives, respectively in terms of network edges correctly identified, and similarly for FP and FN. With these measures we can compute the areas under the associated ROC curves (AUCs), where values closer to 1 indicate better discriminative power between the true zero and non-zero edges in the true graph.

Table 4.1 displays the AUC values for the various settings described above. As expected from the correspondence between the frequentist and Bayesian formulations of the objective, we see very similar performance between the two approaches across each scenario considered. This suggests that, as far as graphical model selection is concerned, the frequentist and Bayesian implementations of the fglasso have equivalent discriminative ability with fixed a regularization parameter. However,

		Network 1			***Network 2***		
		p=10	p=30	p=50	p=10	p=30	p=50
fglasso	FPR (%)	6.07 (3.59)	0.95 (0.46)	13.87 (0.65)	8.61 (4.38)	1.13 (0.53)	16.06 (1.05)
	FNR (%)	38.82 (6.00)	62.46 (5.61)	32.27 (4.45)	24.44 (9.69)	50.00 (3.80)	27.78 (5.69)
	ERR (%)	18.44 (3.73)	9.01 (1.01)	15.33 (0.67)	11.78 (4.34)	4.16 (0.48)	16.41 (1.06)
	F 1	0.71 (0.06	0.52 (0.06)	0.41 (0.02)	**0.72(0.09)**	0.60 (0.04)	0.21 (0.02)
	Sparsity (%)	26.89	5.75	18.14	22	4.16	17.71
fghorse	FPR (%)	18.57 (8.27)	2.14 (0.60)	1.03 (0.36)	10.83 (6.97)	0.74 (0.64)	0.40 (0.21)
	FNR (%)	22.35 (9.41)	36.14 (3.35)	41.81 (4.18)	24.44 (9.69)	41.11 (3.08)	43.06 (3.11)
	ERR (%)	20.00 (6.13)	6.60 (0.78)	4.26 (0.37)	13.56 (6.16)	3.24 (0.52)	1.66 (0.23)
	F 1	**0.75(0.08)**	**0.72(0.03)**	**0.68(0.03)**	0.70 (0.11)	**0.69(0.03)**	**0.67(0.04)**
	Sparsity (%)	40.89 (6.38)	10.23 (0.57)	5.56 (0.54)	23.78 (5.63)	4.34 (0.72)	2.07 (0.22)
FGM	FPR (%)	81.07 (10.11)	44.58 (4.48)	31.35 (4.08)	50.00 (16.76)	19.39 (6.44)	11.14 (2.22)
	FNR (%)	4.71 (6.86)	12.28 (2.48)	19.28 (4.68)	1.11 (3.33)	17.04 (5.79)	19.72 (5.19)
	ERR (%)	52.22 (5.28)	40.34 (3.85)	30.39 (3.44)	40.22 (13.42)	19.24 (5.87)	11.40 (2.10)
	F 1	0.58 (0.03)	0.36 (0.02)	0.30 (0.02)	0.51 (0.09)	0.36 (0.06)	0.30 (0.04)
	Sparsity (%)	86.44	50.23	35.26	59.78	23.33	13.18
True sparsity (%)		37.78	13.1	7.92	20	6.21	2.94

Table 4.2: Summary statistics of false positive (FPR), false positive rate (FNR), mislassification rate (ERR), F1 (Dice) score and estimated graph sparsity of graph estimation with 10 data sets generated by dense functional data with underground Network 1 and 2 separately. "fglasso" refers to the Bayesian functional graphical lasso method with gamma prior for shrinkage parameter λ; "fghorse" refers to the functional graphical horseshoe method, while "FGM" refers to the frequentist version of functional graphical lasso model proposed by Qiao et al. (2019). The means are reported here, the standard errors are shown in paratheses.

the Bayesian approach facilitates additional flexibility in how the regularization parameter is treated. While it is possible to estimate λ with, e.g., cross-validation, it can be computationally expensive to do so. On the other hand, assigning a prior distribution to the regularization term, or circumventing an explicit λ altogether via, e.g., the horseshoe, allows the appropriate penalty to be learned from the data along with the remaining parameters in the model, so that it only needs to be fit once. We consider this approach next.

Next we compare the frequentist fglasso to the hierarchical Bayesian fglasso (as opposed to the fixed-λ Bayesian fglasso considered above) and the functional graphical horseshoe in terms of misclassification error and graph similarity when the regularization parameter is determined as it would be in practice. The frequentist fglasso requires tuning of the regularization parameter, either through cross-validation or by setting it to yield a desired sparsity level. For the frequentist fglasso we use 10-fold cross-validation for model selection, since in practice the *true* sparsity of the underlying graph will most likely be unknown. In frequentist fglasso, the sparsity of estimated graph could be tuned by regularized parameter λ. In the Bayesian framework, the credible level could also impact the sparsity. As the Bayesian fglasso behaves differently than the horseshoe, and to reduce the number of false positives, we use 90% credible intervals for the hierarchical Bayesian fglasso, and

50% credible intervals for thresholding with the functional graphical horseshoe. Table 4.2 reports the false positive rate (FPR), false negative rate (FNR), overall misclassification rate, the F1 score (also known as Dice coefficient), and resulting sparsity levels for $p = 10, 30, 50$ nodes, and $N = 100$ subjects with densely-sampled functions under both Network 1 and 2. The scores are averaged over 10 replications of each scenario. With the exception of $p = 10$ nodes in Network 2, the functional graphical horseshoe always has the highest F1 score, indicating the strongest graph similarity. We further see that the functional graphical horseshoe has the results with closest sparsity level to the true sparsity level. The frequentist fglasso generally performs the worst, due to the overly dense graphs that it produces, reflecting known risks of using log-likelihood-based cross-validation in this type of model (Wasserman and Roeder, 2009).

Further we turn our attention to a direct comparison between the Bayesian fglasso and the functional graphical horseshoe. One advantage of the horseshoe prior is that the global/local shrinkage in the model automatically adapts to the observed data. To allow for such "automatic" adaptation in the Bayesian fglasso, we use the augmented version of the model in which a hyperprior is assigned, $\lambda^2 \sim \text{Gamma}(1, 0.01)$. In this comparison, we consider fpc scores generated from $p = 10$ nodes with $N = 100$ observations and rank $M = 5$ in the basis expansion, using Network 1 defined above. For evaluation, we construct credible regions for each model. Given confidence level, the volume of credible regions is proportional to $|\Sigma_{\hat{\theta}_{ij}}|^{\frac{1}{2}}$. Performance characteristics are quantified in Table 4.3, where normalized logarithm of $|\Sigma_{\hat{\theta}_{ij}}|^{\frac{1}{2}}$ are reported. In addition, Bayesian FDR p_{ij}^{δ} are calculated, where $1 - p_{ij}^{\delta}$ can be considered "q-values", or estimates of the local false discovery rate (Storey, 2003).

Figure 4.1 depicts the Bayesian FDR-based "q-value" for the off-diagonal blocks, separated by the zeros, those with edge weight 0.2, and those with edge weight 0.4. We use the identical practical significance threshold δ (60% quantile of off-diagonal $\|\hat{\mathbf{\Theta}}_{ij}\|_F$) for both Bayesian functional lasso and horseshoe. We can make several observations from this figure. First, the Bayesian fglasso results in more false positives, compared to the lower false positive rate with the same rule applied to the functional graphical horseshoe. Thus, the Bayesian functional graphical lasso exhibits behavior similar to that which is known about its scalar counterpart. The stronger mass near the origin applied by the horseshoe compared to the Bayesian graphical lasso (Li et al., 2019, Theorem 3.2) results in much better identification of the true zeros in the model. Second, for the truly non-zero entries in $\mathbf{\Theta}$, the functional graphical horseshoe q-values have a wider gap compared with the

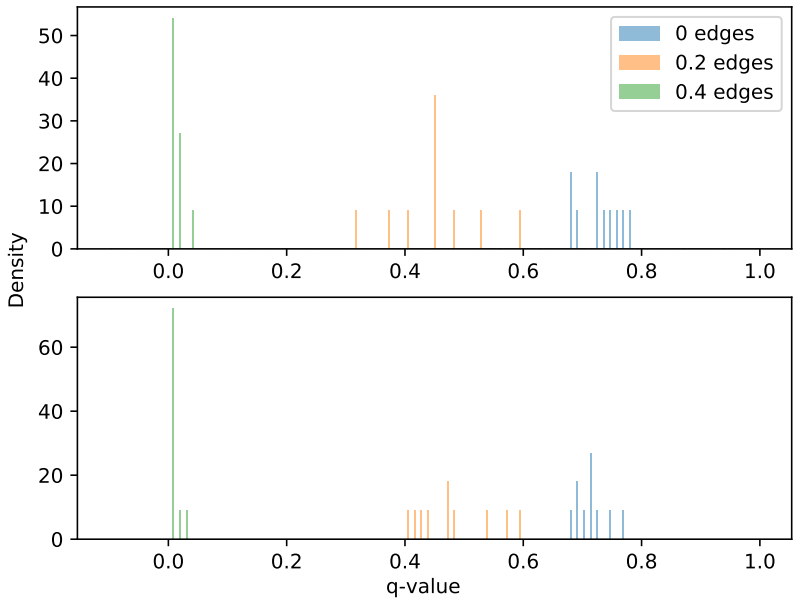

Figure 4.1: Histograms of "q-values" (Storey, 2003) for Bayesian functional lasso (top) and horseshoe (bottom).

Bayesian fglasso. This does not necessarily suggest superior estimation fidelity, however. In the Table 4.3 we see the volume of the functional graphical horseshoe credible regions growing as the signal size goes from 0.2 to 0.4, as would be expected from the results of Van Der Pas et al. (2014), whereas the hierarchical Bayesian fglasso exhibits no such behavior. Performance characteristics of confidence regions are quantified in Table 4.3. In terms of normalized logarithm of confidence region volume, the horseshoe again shows very faithful recovery of the true zeros due to its local-global shrinkage property. For the non-zero estimates, however, we observe more uncertainty with the Bayesian functional graphical horseshoe growing with signal size from 0.2 to 0.4, while fglasso has less uncertainty with the strong signal. We recognize that edge selection via Bayesian FDR / credible regions under the horseshoe tends to be conservative and that the volumes of the credible regions tends to grow with size of the true signal (Van Der Pas et al., 2014).

Lastly, as a pragmatic matter, we evaluate the computational speed and the scalability of our proposed block Gibbs samplers in three different environments, `R`, `NumPy`, and `TensorFlow`. Our proposed Gibbs sampler for the functional graphical horseshoe has similar computational complexity as the Bayesian fglasso, so we only show the results of Bayesian functional graphical lasso. We use

	0 edges	0.2 edges	0.4 edges
fglasso	0.304 (0.056)	0.019 (0.079)	-0.962 (0.150)
fghorse	-0.579 (0.037)	0.169 (0.152)	1.650 (0.060)

Table 4.3: Summary of average credible region volumes (log value) corresponding to edges with weights 0, 0.2 and 0.4 for Bayesian functional glasso and horseshoe based on 10 replications. The numbers in the parentheses are standard deviation.

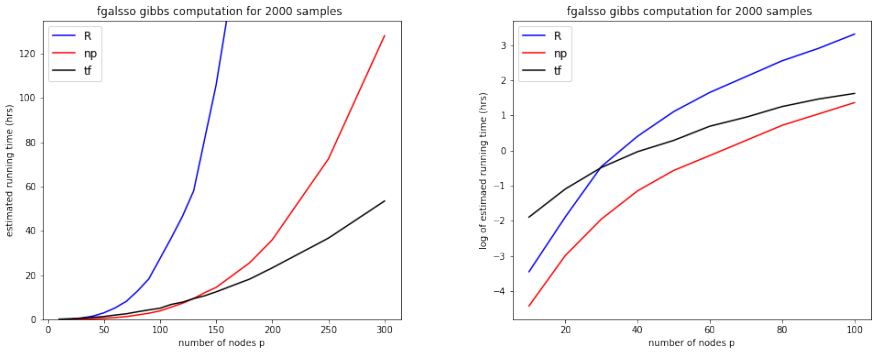

Figure 4.2: Computational cost as a function of p for the block Gibbs sampler under three environments, `R` (R), `NumPy` (np), `Tensorflow` (tf). Estimated wall time (left) and its logarithm (right) versus number of nodes p. Note the difference in the range of p in the left and right plots, and the change in ranking at lower numbers of nodes versus higher numbers.

simulated data from Network 1 with $N = 100$ and rank $M = 5$ while varying the number of nodes up to $p = 300$. All computations are run on a standard x86 machine (28-core CPU with 120G memory), using `R` or `Python`. For each such simulated data set, we measure the wall time required to generate 2,000 full iterations of the Gibbs sampler. We provide implementation code for the different environments at `https://github.com/jjniu/BayesFGM`.

Figure 4.2 displays the results, where the right panel is on the log scale for $p = 1, \ldots, 100$ for finer distinction. It takes about 1.9 min, 0.72 min, and 9 min to generate 2,000 iterations under $p = 10$ nodes with `R`, `NumPy` and `TensorFlow`, respectively. On the other hand, with $p = 200$, pure `R` takes 262.29 hours, `NumPy` takes 128.12 hours, and `TensorFlow` takes 53.5 hours. In general, we see that `TensorFlow` is the slowest language with small graphs, but that it also scales at a much lower rate that `NumPy` or `R`. Of the approaches considered, `NumPy` is the fastest for graphs of size up $p \approx 100$, but `TensorFlow` is preferable for extremely large graphs. It is no surprise that base `R` is the slowest and scales at the worst rate. `R` is known for its tendency to be slow with iteration-intense tasks due to its repeated data type conversion on each iteration. It is likely that interfacing `R` with `C++` via the `Rcpp` package (Eddelbuettel and François, 2011) would yield much better results when

working in pure `R`, but we did not consider that approach here.

4.2 Experiments for Deep Functional Graphical Models

In the experiments of Deep FGM, we use the procedure demonstrated in Algorithm 3 to generate a number of functional data sets corresponding to five different kinds of graph structures, i.e. random, cluster, small-world, star and scale-free graphs. Random graph is generated with sparse random connections; cluster graph is formed from the disjoint union of complete graphs; small-world graph means that most nodes are not neighbors of one another, but they can be connected by small number of steps via other nodes; star graph means a single central node exists which connected to all rest nodes; scale-free network means the distribution of number of neighbors for each node follows a heavy-tailed distribution. We provide examples in Figure 4.3. We use the `scikit-learn` (Pedregosa et al., 2011) to produce sparse precision matrices with 95% sparsity level, `BDgraph` R-package to generate data from cluster networks based on the G-Wishart distribution (Mohammadi et al., 2015) and `rags2ridges` R-package (Peeters et al., 2021) to generate data from small-world, star and scale-free networks. We use first four Fourier bases to generate functional data. Note that we consider identical Θ_k in Algorithm 3 for each basis coefficient to simplify the generation process of synthetic data. We repeat the experiment for 100 different graphs (of synthetic testing set for deep models) with 20, 40 nodes and sample observations with different size ranging 15, 40, 100. In neuroscience, one often has limited number of samples for inferring graph structure, and thereby we evaluated the robustness of Deep FGM by applying our model with small sample sizes ($N = 40, 15$).

We compare our neural network based solution (called 'Deep FGM' in Table 4.4) with baselines including both the frequentist and Bayesian methods: functional graphical lasso (Qiao et al., 2019) and functional graphical Horseshoe in Section 2.2.2. The functional graphical lasso (denoted as 'FGM' in Table 4.4) is a frequentest method imposing a block sparsity constraint on the blockwise precision matrix of basis coefficients. Functional graphical Horseshoe is a Bayesian functional graphical method with Horseshoe regularization. For dimension reduction, each baseline uses functional principal components or Fourier bases. We select the number of principal components based on 95% of variation explained (FVE), or use first 5 Fourier bases, which guarantee to cover the true bases in synthetic generation procedure.

For Deep FGM configuration, we use Adam optimization (Kingma and Ba, 2014) in Ten-

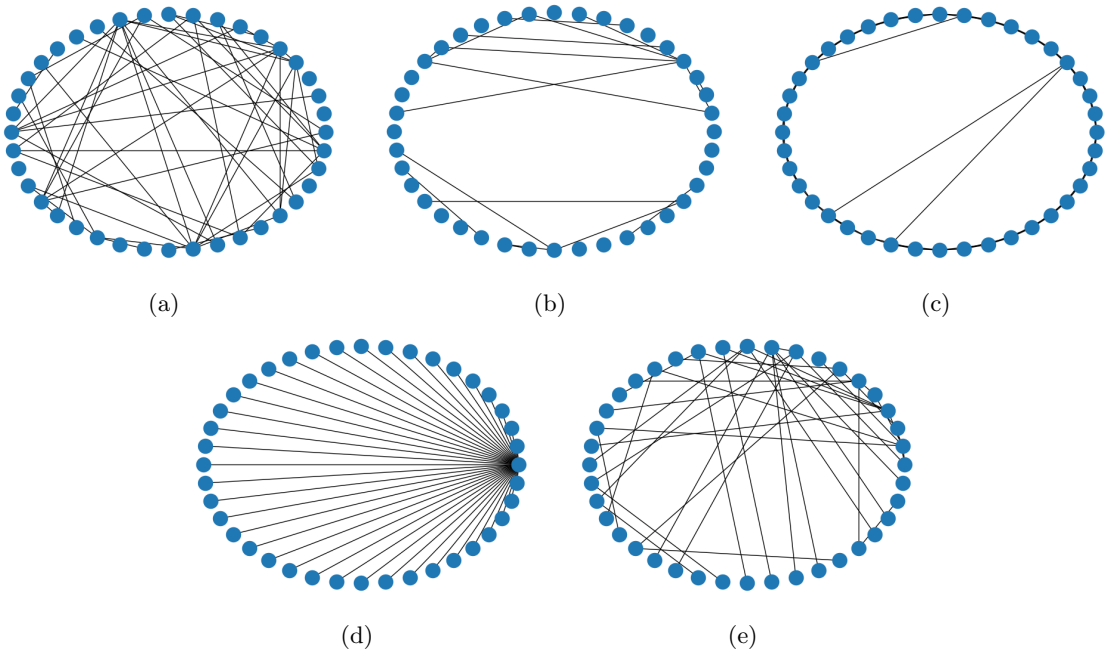

Figure 4.3: Samples of generated synthetic data with following topology: (a) random (b) cluster (c) small-world (d) star (c) scale-free graphs

sorflow with learning rate set to be 3×10^{-4}. We set neural network in basis node with 3 layers, and each layer is 64 dimensions. We set the convolution kernel size to be 50 with dropout rate to be 0.1. For the adaptive basis layer, we set number of basis nodes to be 4. We use RELU activation for all layers execept the last layer, which we use sigmoid function to produce edge classification probability.

To evaluate the estimation performance, we use the same metrics as Qiao et al. (2019), which is the average area under the ROC curve (AUC). For the frequentist method, fglasso directly returns adjacency matrix with weights under the optimal regularizer setting on the testing data; Bayesian functional Horseshoe returns MCMC samples of blockwise precision matrix and the Frobenius norm of each block of posterior mean is used for edge estimation. For Deep FGM, we prepare 5000 training/validation data sets to train deep FGM with different input configurations, and we train each network for 100 epochs. After each epoch, we evaluate the model using the validation data, and keep the model with the highest AUC.

Table 4.4 reports the average AUCs with the corresponding standard deviation for Deep FGM and baseline methods in 5 graph types, i.e. random, cluster, small-world, star and scale-free. Deep FGMs outperform baselines in most cases even with small data sets (sample size is 40

Graph	p	N	Deep FGM		FGM				fghorse			
					FPCA		Fourier		FPCA		Fourier	
Random	20	100	0.85	(0.09)	0.85	(0.08)	0.81	(0.07)	0.88	(0.08)	**0.89**	(0.08)
		40	**0.83**	(0.10)	0.76	(0.08)	0.76	(0.08)	0.81	(0.08)	0.81	(0.08)
		15	**0.77**	(0.09)	0.69	(0.09)	0.69	(0.09)	0.70	(0.10)	0.69	(0.12)
	40	100	0.79	(0.05)	0.76	(0.04)	0.76	(0.04)	**0.79**	(0.02)	0.79	(0.03)
		40	**0.76**	(0.05)	0.70	(0.03)	0.70	(0.03)	0.71	(0.04)	0.71	(0.03)
		15	**0.67**	(0.04)	0.63	(0.03)	0.63	(0.03)	0.64	(0.04)	0.64	(0.03)
Cluster	20	100	**0.93**	(0.09)	0.85	(0.12)	0.86	(0.12)	0.84	(0.12)	0.84	(0.13)
		40	**0.87**	(0.10)	0.81	(0.13)	0.81	(0.13)	0.79	(0.22)	0.80	(0.18)
		15	**0.80**	(0.12)	0.74	(0.14)	0.75	(0.15)	0.74	(0.15)	0.74	(0.15)
	40	100	**0.90**	(0.05)	0.84	(0.06)	0.84	(0.06)	0.87	(0.05)	0.86	(0.05)
		40	**0.85**	(0.06)	0.74	(0.07)	0.75	(0.07)	0.79	(0.08)	0.79	(0.08)
		15	**0.78**	(0.08)	0.75	(0.07)	0.75	(0.07)	**0.78**	(0.08)	**0.78**	(0.08)
Small-World	20	100	**0.99**	(0.01)	0.93	(0.03)	0.93	(0.03)	0.93	(0.03)	0.93	(0.03)
		40	**0.94**	(0.02)	0.68	(0.04)	0.68	(0.04)	0.77	(0.04)	0.77	(0.04)
		15	**0.83**	(0.03)	0.60	(0.04)	0.60	(0.04)	0.61	(0.04)	0.61	(0.04)
	40	100	**0.98**	(0.01)	0.93	(0.02)	0.93	(0.02)	0.93	(0.01)	0.93	(0.01)
		40	**0.93**	(0.01)	0.69	(0.04)	0.68	(0.04)	0.78	(0.04)	0.78	(0.04)
		15	**0.82**	(0.02)	0.60	(0.03)	0.64	(0.02)	0.60	(0.03)	0.63	(0.03)
Star	20	100	**1.00**	(0.00)	0.83	(0.05)	0.83	(0.05)	0.78	(0.02)	0.78	(0.02)
		40	**1.00**	(0.00)	0.70	(0.05)	0.79	(0.05)	0.71	(0.03)	0.71	(0.04)
		15	**1.00**	(0.01)	0.76	(0.08)	0.76	(0.08)	0.64	(0.05)	0.65	(0.04)
	40	100	**1.00**	(0.00)	0.77	(0.04)	0.77	(0.04)	0.69	(0.04)	0.69	(0.04)
		40	**1.00**	(0.01)	0.65	(0.04)	0.65	(0.04)	0.66	(0.06)	0.66	(0.06)
		15	**0.95**	(0.12)	0.73	(0.07)	0.73	(0.07)	0.65	(0.05)	0.65	(0.05)
Scale-Free	20	100	**1.00**	(0.00)	0.98	(0.02)	0.98	(0.02)	0.98	(0.02)	0.98	(0.02)
		40	**0.98**	(0.01)	0.81	(0.06)	0.85	(0.05)	0.88	(0.05)	0.86	(0.05)
		15	**0.91**	(0.04)	0.68	(0.07)	0.68	(0.07)	0.68	(0.05)	0.68	(0.06)
	40	100	0.98	(0.02)	0.95	(0.03)	0.94	(0.03)	**0.99**	(0.01)	**0.99**	(0.01)
		40	**0.93**	(0.02)	0.82	(0.04)	0.81	(0.04)	0.89	(0.04)	0.89	(0.03)
		15	**0.90**	(0.03)	0.71	(0.05)	0.71	(0.05)	0.75	(0.07)	0.75	(0.08)

Table 4.4: The area under the ROC curves. The means are demonstrated here with the standard errors in parenthesis. "Deep FGM" refers to the proposed deep learning model; "fghorse" refers to the functional graphical horseshoe method proposed by Niu et al. (2021); "FGM" refers to the frequentist version of functional graphical lasso model proposed by Qiao et al. (2019).

and 15) marginally. In the case of random Gaussian graphs, Bayesian functional Horseshoe has superior performance with larger data sets ($N = 100$). The frequentist FGM shows fair performance compared with the other two. For cluster graphs we can see that, Deep FGM performs particularly well in larger data sets settings compared with two baselines. Bayesian functional Horseshoe has similar performance as Deep FGM when graph has more nodes but small dataset ($p = 40, N = 15$). For smaller graph, The frequentist FGM shows better performance than Bayesian method, whereas Bayesian approach outperform frequentist approach when graph is large. Particularly, our deep models outperforms on AUC (close to 1) on small-world and star networks, which is more close to real world dataset. The AUCs of Deep FGM for large data sets in both $p = 20, 40$ cases are 1 or close to 1 and also work much better than baseline in other cases. In the last scenario, scale-free graph, our networks still have best performance in the most settings and improve the AUC marginally compared with baselines in small dataset cases. Table 4.4 shows that our model is performance

superior other approaches even with very the limited number of observations to $N = 15$. Overall, the experiment results demonstrate that our method is competitive to baselines and can be adaptive to specific graph structures.

Note that our experiments can be further enhanced with different basis functions. Since our proposal aims to validate the whole framework, experiments on specific basis settings is beyond the scope of this work, and we leave it in the future work.

4.3 Experiments for Deep Causal Models

Our experiment is based on synthetic data since it is almost impossible to know the counterfactual outcome based on real world data. Additionally, it is difficult to have a perfect observational data with ground truth of ATE or ITE. To evaluate our causal model with ground truth, we consider three different simulation data settings. The first settings is a randomized trial with no treatment propensity. The second settings is to mimic observational data with selection bias. In the third scenario, we use the Infant Health and Development Program (IHDP) dataset in a naturalistic simulation introduced in Hill (2011) to generate semi-synthetic functional dataset. For the third scenario, we adopt 25 covariates that are collected from designed randomized experiment of IHDP, and these variables are relevant to infant and his/her mother, such as birth weight, head circumference, prenatal care, education, alcohol, etc. In the treatment group, infants are treated with more intensive care and visit. The outcomes are generated based on certain generation procedures to mimic the conditions with selection bias and complex confoundedness in the realistic observational data. The strategy is inspired from Hill (2011) and Yao et al. (2021a). Note that the original IHDP data is not a typical functional data. To create a functional confounding experiment, we convert each covariate feature as functional data by setting covariates as functional basis coefficients.

We consider the covariate $X_i(t) = \sum_{k=1}^{50} a_{ik}\phi_k(t)$. For the fist two scenarios, the coefficient $a_{ik} = z_{ik}r_{ik}$, r_{ik} are i.i.d. sampled from uniform distribution on $[-\sqrt{3}, \sqrt{3}]$ and $\phi_k(t)$ are kth Fourier basis function. We set the number of Fourier basis to be 50. The discrete observable data are sampled as $X_i(t_k), k = 1, \ldots, T$, where subjects i was recorded at $t_1, \ldots, t_T \in [0, 1]$.

- **Scenario 1** (Randomized Trial): $z_{i1} = z_{i3} = 5, z_{i2} = z_{i10} = 3$, and $z_{ik} = 1$ for the rest k. The

outcome is generated as

$$
\begin{aligned}
Y_i(0) &= a_{i5}^2 = (\langle X, \phi_5 \rangle)^2 \\
Y_i(1) &= a_{i5}^2 + 4 = (\langle X, \phi_5 \rangle)^2 + 4
\end{aligned}
\tag{4.1}
$$

where $\langle \cdot, \cdot \rangle$ is the inner product operation, and the functional covariate depends on the 5th basis. It is a randomized experiment with treatment fraction 0.5, and the treatment propensities don't depend on the covariate. The treatment effect function is constant and the main effect is nonlinear. Thereby, the outcome surfaces are parallel across treatment and control groups; Both ITEs and ATE are 4.

- **Scenario 2** (Selection Bias): We use the similar setting as Scenario 1: $z_{i1} = z_{i3} = 5, z_{i2} = z_{i10} = 3$, and $z_{ik} = 1$ for the rest k. Except we define the outcome as

$$
\begin{aligned}
e_i &= logit^{-1}(a_{i5} - 2) \\
W_i &= Bernoulli(e_i) \\
Y_i(W_i) &= a_{i5}^2 + 4W_i = (\langle X, \phi_5 \rangle)^2 + 4W_i
\end{aligned}
\tag{4.2}
$$

where e_i indicates the propensity score for ith subject. In this case, it is generally not a randomized experiment, since the treatment propensities depend on $X(t)$. The propensity score is correlated with the main effect, and we set the treatment effect to be 4 as ground truth. The total sample size including treatment and control group is 1000 for both Scenario 1 and 2.

- **Scenario 3** (Treatment Effect Heterogeneity): We leverage the IHDP data which contains 25 covariates for 139 treated subjects and 608 control subjects. As we mentioned before, we treat these 25 covariates as functional data, and thereby each covariates is corresponding basis coefficient. Formally, We define IHDP data as $A_i = (a_{i1}, \cdots, a_{i25}) \in \mathbb{R}^{25}$, where we set first 25 basis coefficients as IHDP covariates, and the rest of a_{ik} to be 0. Then generate nonlinear and non-parallel outcome across treatment conditions as follow

$$
\begin{aligned}
Y_i(0) &= \exp((A_i + W_i)\beta) = \exp(\sum_{k=1}^{25} (\langle X_i, \phi_k \rangle + 0.5)\beta_k), \\
Y_i(1) &= A_i\beta - w^i = \sum_{k=1}^{25} \langle X_i, \phi_k \rangle \beta_k - w^i.
\end{aligned}
\tag{4.3}
$$

Following Hill (2011), W_i is an offset vector with all the entries equal to 0.5, β is a vector of coefficients randomly generated with a discrete distribution based on values $(0, 0.1, 0.2, 0.3, 0.4)$ of probability $(0.6, 0.1, 0.1, 0.1, 0.1)$. In the ith generation, w^i was set such that the ITE equals to 4.

We use 80% training/validation and 20% testing split for all experiments. Among 80% split, we use 15% as validation set to select proper hyper-parameters via evaluation, and rest of them as training set. After training, we calculate the CATE and the ATE on the entire simulation dataset. Following Hill (2011), we reported the Precision Estimation of Heterogeneous Effects (PEHE) to evaluate model performance on simulations. Smaller PEHE is better and optimal PEHE is 0. PEHE is defined as

$$PEHE = \sqrt{\frac{1}{N}\sum_{i=1}^{n}(CATE_i - C\hat{A}TE_i)^2}. \tag{4.4}$$

We also report ATE, the bias $|ATE - A\hat{T}E|$ and MSE on testing data for evaluation. We train 1000 epochs and evaluate the model using the validation set after each epoch, then keep the model with the best PEHE. After training we can estimate causal effects in the whole dataset or held-out testing dataset. Here we apply the whole dataset for inference ATE/CATE and testing dataset for prediction (metric MSE). We can get two predictions for each unit: $\hat{Y}(W)$ and $\hat{Y}(1-W)$, then the estimated CATE for the out-of-sample data is

$$C\hat{A}TE = (1-2W)(\hat{Y}(1-W) - \hat{Y}(W)) \tag{4.5}$$

and the estimated average treatment effect can be calculated by,

$$A\hat{T}E = \frac{1}{N}\sum_{i=1}^{N} C\hat{A}TE_i. \tag{4.6}$$

In the Table 4.5, we present the results for S-learner, T-learner and TARNet combining with functional data analysis. To easily demonstrate our experiments, the underlying true treatment effect value is set as 4 among all three simulation cases, and the ideal ATE should be as close as 4. As shown in the table, in case 1 (randomized experiment), S-learner works best in case 1 with PEHE 0.16, but it also shows S-learner has worst estimation on CATE, which is reflected by Test MSE. T-learner and TARNet have close results and generalize well on CATE of testing set. In case 2,

which is not an randomized controlled experiment, we find T-learner and TARNet are perform worse then S-learner on PEHE, and the variance of ATE is much larger than S-learner. This is because the main effect is correlated with propensity score and the three methods are not robust enough to capture it. In the case 3 with semi-synthetic observational data, S-learner perform worst and is not capable of handling the unbalance and treatment-effect heterogeneous data. Since S-learner (as shown in Figure 3.2) is a predictor for the outcome, it is reasonable that the S-learner perform well when the treatment is a strong indicator for outcome, but performs bad if the confounding has strong correlation with treatment or outcome. In case 3, the treatment is a weak indicator, and confounding is close to real world and much complicated than first two scenario. Therefore, the S-learner tends to be biased towards zero and shrink the treatment effect to the halfway. The T-learner outperform S-learner on PEHE, since T-learner fit two independent outcome models based on different treatments. TARNet with an additional MLP, which model the non-linearity of $X(t)$ and share the same feature representation of treatment and control group, outperform the others in case 3.

Overall, we observe that three methods in most cases produce reasonable ATE for the functional confoundings (except S-learner in case 3), which illustrates that the causal inference is still effective within the functional data analysis framework, even adopting with deep model. The TARNet often outperform S-learner and T-learner on Test MSE in case 1 and 2, but worst in case 3, indicating that TARNet generalize well when the confounding is simply constructed, but may be overfitting when confounding is complicated like real-world data. Additionally, by adding selection bias (scenario 2) and leverage real world data (scenario 3), the causal estimation becomes vulnerable for functional data, meaning that we still have gap to effectively handle real world causal inference in functional data.

Scenario	Methods	ATE	PEHE	Bias	Test MSE
1	S-learner	3.94 (0.15)	**0.16**	0.06	0.163
	T-learner	3.77 (0.21)	0.31	0.23	0.005
	TARNet	3.83 (0.21)	0.27	0.17	0.003
2	S-learner	4.24 (0.14)	**0.28**	0.24	0.278
	T-learner	3.61 (1.41)	1.46	0.39	0.170
	TARNet	4.11 (1.13)	1.14	0.11	0.040
3	S-learner	1.79 (6.38)	6.75	2.21	0.416
	T-learner	3.82 (2.73)	2.73	0.18	0.289
	TARNet	3.94 (1.62)	**1.63**	0.06	0.421

Table 4.5: Summary of results in experiments. The method with the best PEHE is marked in bold. The "ATE" is the expected value of the overall $C\hat{A}TE$; the standard deviation of $C\hat{A}TE$ is reported in parentheses and "Bias" denotes the bias to true treatment effect 4. For Bias, PEHE, and Test MSE, **smaller is better**. The ideal ATE is 4.

Chapter 5

Applications to Neuroimaging Data

In this chapter, *application on alcoholic study* and *structural connectivity after traumatic brain injury* leverage Bayesian functional graphical model or deep FGM to infer the brain connectivity based on real world data.

5.1 Application on Alcoholic Study

Here we apply our proposed functional graphical horseshoe method to an electroencephalography (EEG) dataset from an alcoholism study originally reported by Zhang et al. (1995). The data, freely available at `https://archive.ics.uci.edu/ml/datasets/eeg+database`, consist of 122 subjects, 77 of whom were identified as alcoholics, and 45 in the control group. Signals were initially collected from 64 electrodes placed on subjects' scalps at standard positions, and captured voltage signals at 256 Hz during a one-second time period. 120 trials were collected per subject. During each trial, the subject was exposed to either a single stimulus (a single picture) or two stimuli (a pair of pictures) shown on a computer monitor. The interest here is in estimating a graphical network representing functional connections between different brain regions. Assuming that the data $g_1(t), \ldots, g_{64}(t)$ jointly follow from a 64-dimensional multivariate Gaussian process (MGP), Zhu et al. (2016) and Qiao et al. (2019) both analyzed the data via functional graphical

model methodology. We filtered the signals through a banded filter between to obtain α frequencies between 8 and 12.5 Hz, as the α band has been shown to differentiate between alcoholic and control groups under this task. Moreover, we take the average of all trials for each subject, resulting in a single event-related potential curve at each electrode for each subject. The filtering was performed by applying the `eegfilter` function in the `eegkit` package (Helwig, 2018) in `R`. Since the data are regularly and densely sampled, we employed the regular SVD method to compute the principal component scores $\hat{\mathbf{a}}$. The truncated number of fpcs are selected so that at least 95% of the variation in the filtered signal trajectories for control and alcoholic curves are captured by the basis representations. For each MCMC run, 10,000 iterations were retained after an initial burn-in period of 1000 iterations. To be conservative and achieve a higher level of sparsity, we are interested in finding the edges through Bayesian FDR-based inference by thresholding block $\|\hat{\mathbf{\Theta}}_{ij}\|_F$ by ϕ_α^δ, that controls the overall average FDR at level $\alpha = 0.01 \sim 0.05$ and practice significance δ is 60% quantile of off-diagonal $\|\hat{\mathbf{\Theta}}_{ij}\|_F$.

The results are summarized in Figure 5.1. The weight of each edge is evaluated as p_{ij}^δ, the posterior probability of Frobenius norm of the estimated matrix for that edge of at least δ; i.e., $\Pr\{\|\hat{\mathbf{\Theta}}_{ij}\|_F > \delta\}$. As p_{ij}^δ is greater than ϕ_α^δ, it indicates independence between node i and j; on the other hand, if p_{ij}^δ is less than ϕ_α^δ, we infer that node i and node j have conditional dependence. The threshold ϕ_α^δ is a cutpoint on the posterior probabilities that control the average Bayesian FDR at level α. The practical threshold δ is selected to be the 60% percentile of $\|\hat{\mathbf{\Theta}}_{ij}\|_F$ and α is 0.01 and 0.05 for upper and lower graphs in Figure 5.1. In the weighted graphs, thicker edges indicate larger weights. We can see that most of the common edges have strong weights. By contrasting the alcoholic and control graphs, we see that the alcoholic group contains more edges connecting the frontal-central regions than the control group. It also appears that the right parietal region tends to have more connection in the alcoholic group than in the control group. Finding a more densely connected frontal region and differences in the right parietal region agrees with that which was found by Zhu et al. (2016). There are clear differences, of course, just as there were between functional graphical models estimated by Qiao et al. (2019) and Zhu et al. (2016). We remark that Qiao et al. (2019) specifically tuned their functional graphical lasso to obtain only 5% sparsity for ease of presentation. Overall, however, we find qualitative agreement with the analysis of Zhu et al. (2016), despite the fact that their assumed model was quite different from our functional graphical horseshoe, and the fact that our approach does not assume a decomposable graph.

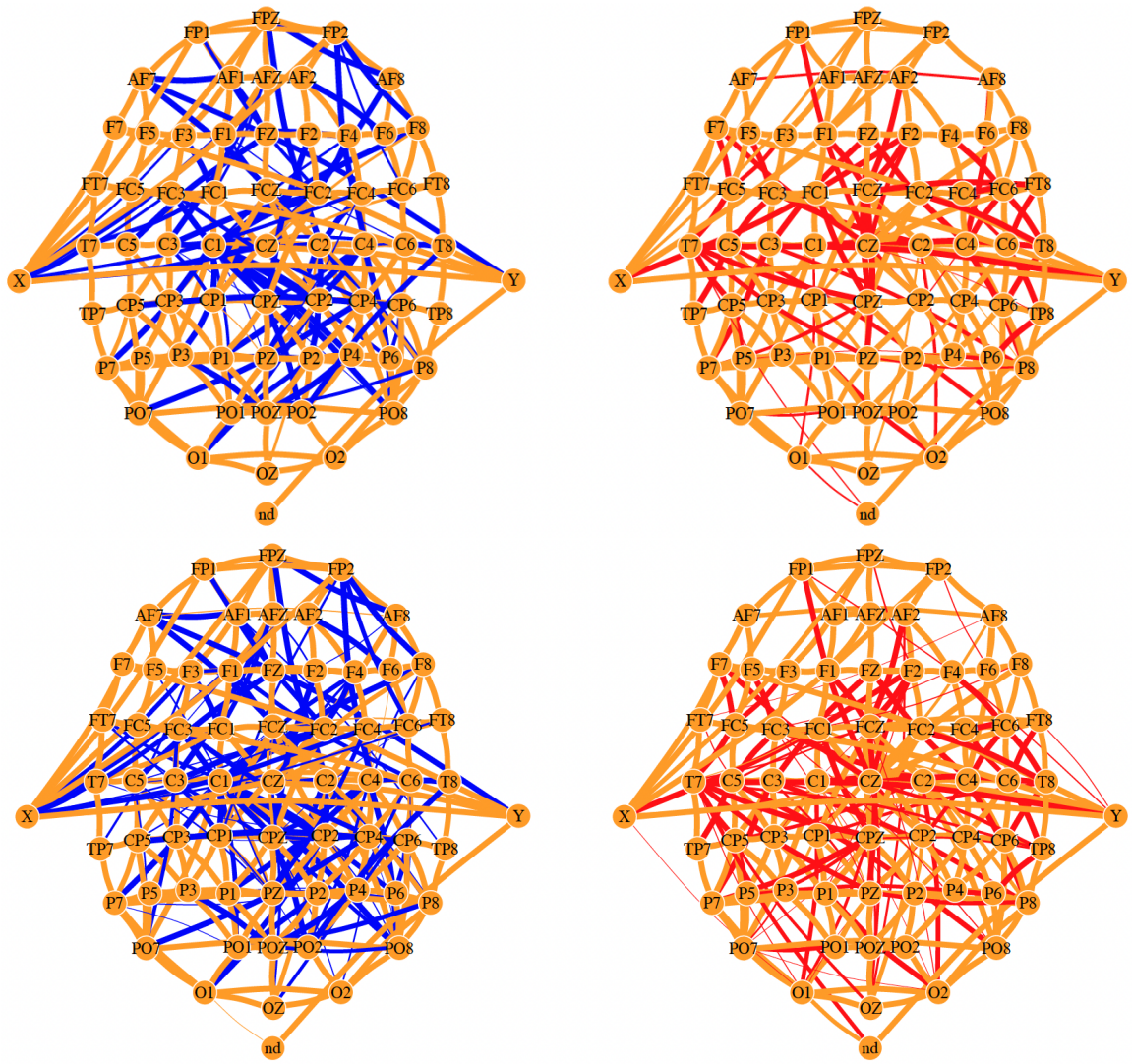

Figure 5.1: Functional brain connectivities for the alcoholic (left) and control (right) groups constructed by Bayesian functional graphical Horseshoe by controlling false discovery rate α 1% (upper) and 5% (lower)

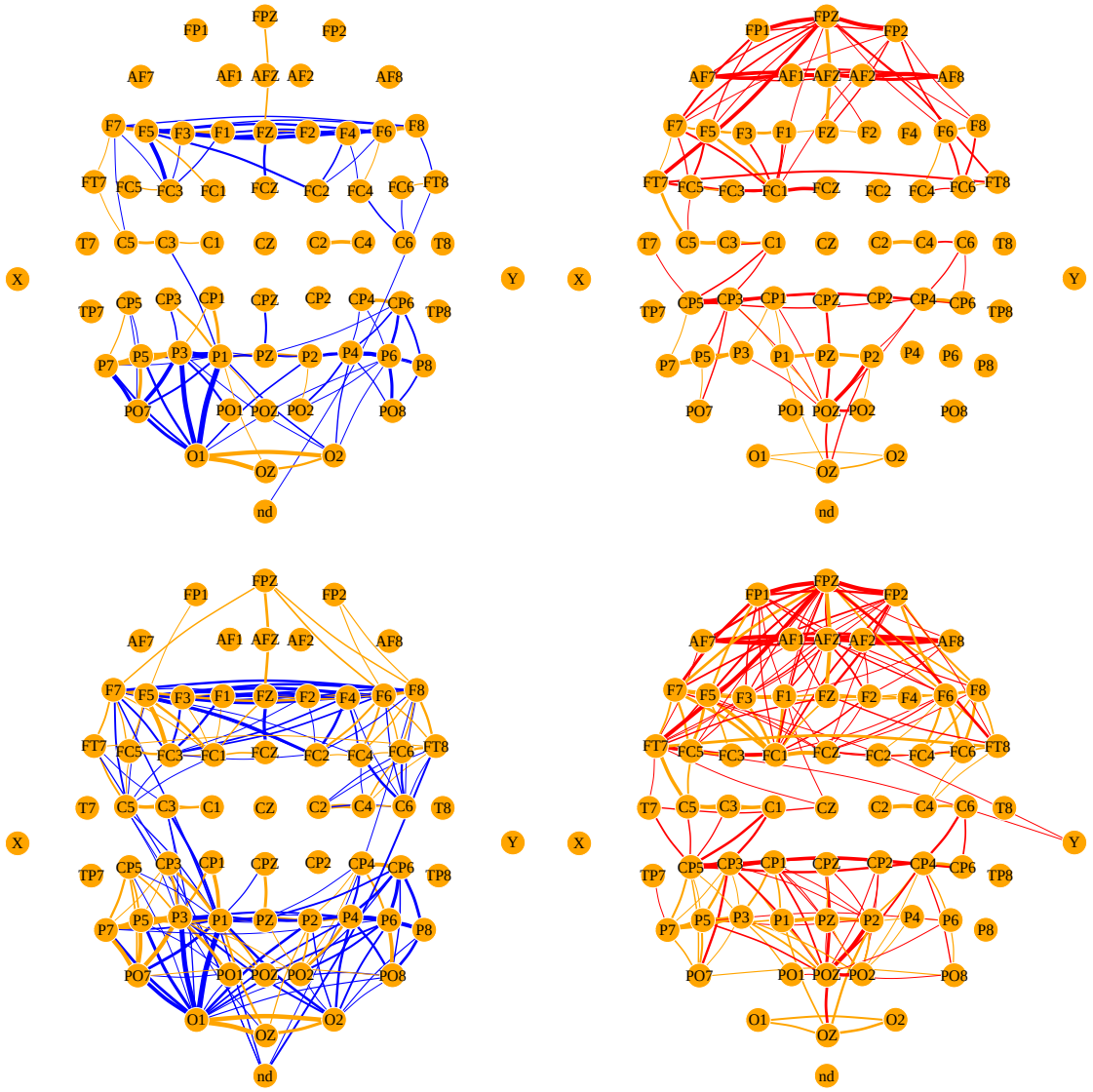

Figure 5.2: Functional brain connectivities for the alcoholic (left) and control (right) groups constructed by Deep FGM with 95% (upper) and 90% (lower) sparsity level. The thickness of edges indicates the weights. Blue lines denote edges identified only in the alcoholic group, red denote edges identified only in the control group, and orange denote edges identified both by in alcoholic and control groups.

In addition, we apply Deep FGM proposed in Section 2.3 to the real world EEG data, and we are interested at getting insights from deep learning method. We train two models separately for alcoholic and control groups. Both models are trained on synthetic data using random Gaussian graphs with 95% sparsity level. For two group of dataset, we set $N = 77$ and $N = 45$ respectively (see details in section 4.2). The reason we train two different models instead of identical one is that the strength of regularization could be different with size of data sets. Besides the regularization property, both networks are implicitly trained with the same assumptions due that the synthetic data generation process are same. The average testing AUCs for synthetic data are 0.723 and 0.725 for alcoholic and control graphs based on 100 replications. Figure 5.2 shows the estimated networks for the alcoholic and non-alcoholic groups at roughly 95% and 90% sparsity level. From the result, we see that the brain networks have some interesting patterns for the two groups. Firstly, the alcoholic and control groups share similar cluster patterns, which may reveal the existence of some regional patterns for brain activity. Moreover, both alcoholic and control group have few connectivity in the parietal area. Furthermore, we observe that less connected central areas are in both alcoholic and control graphs and increased functional connectivity exists in the occipital area in the alcoholic graph compared with control one. Overall, our findings have some similar patterns with results in Qiao et al. (2019); Zhu et al. (2016); Solea and Dette (2021).

5.2 Structural Connectivity after Traumatic Brain Injury

It is known that traumatic brain injuries can cause acute disconnections in white matter tracts (Rutgers et al., 2008), and there is interest in studying what happens to these connections during the chronic phase after such injuries. To address this question, we applied our proposed Bayesian functional graphical horseshoe to diffusion tensor imaging data (DTI) obtained from the Alzheimer's Disease Neuroimaging Initiative (ADNI). The data consist of 34 subjects, 17 of whom have been identified as having experienced traumatic brain injury (TBI) with the remaining 17 being healthy controls. The control subjects were selected from a much larger group via propensity score matching to control for confounding variables such as patient's age, sex, whole brain volume, and Alzheimer's disease status. The data contain initial and follow-up measurements of fractional anisotropy in each of $p = 26$ regions of interest (ROI) in the brain, resulting in irregularly measured longitudinal data for each ROI. For each subject, anywhere from one to nine time points are available,

each separated by several months. The data preprocessing includes eddy-correction (Andersson and Sotiropoulos, 2016), brain extraction, (Smith, 2002) and intensity normalization (Jenkinson and Smith, 2001; Jenkinson et al., 2002). For each voxel in a brain image, the fractional anisotropy (FA) is calculated as $FA = (3/2)((\lambda_1 - \bar{\lambda})^2 + (\lambda_2 - \bar{\lambda})^2 + (\lambda_3 - \bar{\lambda})^2)^{1/2}(\lambda_1^2 + \lambda_2^2 + \lambda_3^2)^{-1/2}$, where λ_1, λ_2 and λ_3 are the eigenvalues associated with the $x-$, $y-$ and $z-$ directions of the diffusion tensor and $\bar{\lambda} = \frac{\lambda_1+\lambda_2+\lambda_3}{3}$ is the mean diffusivity. FA indicates the degree of anisotropy of a diffusion process valued between 0 and 1. If the FA is close to 0, the diffusion is unrestricted in all directions, indicating loosely structured (i.e., deteriorated) white matter, whereas FA close to 1 means that the diffusion mainly occurs only along one axis, thus indicating stronger white matter in that area. The voxel-wise FA values are averaged to summarize the observed FA in each ROI at each time point. In total, we focus on exploring the connectivity (conditional dependence in brain atrophy) between 52 ROIs. Each ROI considered is listed in Table 5.1.

Similar to the EEG study, we assume the longitudinal data $g_1(t), \ldots, g_{52}(t)$ jointly follow a 52-dimensional MGP. We use the PACE algorithm of Yao et al. (2005) to carry out FPCA using `fdapace` package (Carroll et al., 2020a), where the truncation level of fpc scores is selected by capturing 95% of the variance in the curves. Then we fit our functional graphical horseshoe model via the Gibbs sampler discussed in 2.2.2, running the MCMC for 100,000 iterations after an initial 10,000-iteration burn-in period. The estimated graphs are constructed based on Bayesian FDR-based inference from the approximate posterior sample, at level $\alpha = 0.1$ and practice significance δ is 60% quantile of off-diagonal $\|\hat{\mathbf{\Theta}}_{ij}\|_F$.

The estimated FA networks for the TBI and control groups are plotted in top panel of Figure 5.3. The sparsity levels for the TBI and control groups are 4.60% (62 edges) and 2.94% (39 edges), respectively. Twelve edges are common to both graphs, indicated by orange lines in the figure. The estimated FA networks by FGM from Qiao et al. (2019) for TBI and control groups are also given in bottom panel of Figure 5.3. In both results, the TBI group tends to have more connections between different ROIs, particularly within right hemisphere. In most types of brain damage and dysfunction, performance IQ (PIQ) tends to deteriorate faster than verbal IQ (VIQ), leading to a PIQ/VIQ discrepancy. The increased connectivity in the right hemisphere observed here may reflect degraded right hemisphere functioning as PIQ deteriorates.

We see overall a denser set of connections in the TBI group than in the healthy controls. This agrees with results reported by Kook et al. (2021) in a study of children with a history of

Label	Hemisphere	ROI
CST	left, right	Corticospinal tract
ICP	left, right	Inferior cerebellar peduncle
ML	left, right	Medial lemniscus
SCP	left, right	Superior cerebellar peduncle
CP	left, right	Cerebral peduncle
ALIC	left, right	Anterior limb of internal capsule
PLIC	left, right	Posterior limb of internal capsule
PTR	left, right	Posterior thalamic radiation, includes optic radiatio
ACR	left, right	Anterior corona radiata
SCR	left, right	Superior corona radiata
PCR	left, right	Posterior corona radiata
CGC	left, right	Cingulum, cingulate gyrus
CGH	left, right	Cingulum (hippocampus), cingulate gyrus
FX_ST	left, right	Fornix (cres) / Stria terminalis
SLF	left, right	Superior longitudinal fasciculus
SFO	left, right	Superior fronto-occipital fasciculus, could be a part of anterior internal capsule
IFO	left, right	Inferior fronto-occipital fasciculus
SS	left, right	Sagittal stratum, includes inferior longitudinal fasciculus and inferior fronto-occipital fasciculus
EC	left, right	External capsule
UNC	left, right	Uncinate fasciculus
FX	left, right	Fornix, column and body of fornix
GCC	left, right	Genu of corpus callosum
BCC	left, right	Body of corpus callosum
SCC	left, right	Splenium of corpus callosum
RLIC	left, right	Retrolenticular part of internal capsule
TAP	left, right	Tapatum

Table 5.1: Index of the "Eve" white matter atlas labels corresponding to Figure 5.3 the TBI connectivity study.

TBI compared to a group with extra-cranial injury. The increased structural connectivity observed here in the TBI patients may reflect so-called "axonal sprouting" occurring as disconnected neurons attempt to reconnect to a network from which they have been isolated. This results in "improved" connectivity, but nevertheless a deterioration of functional activation and task performance. It is also possible that there are no entirely new structural connections between areas, but that the weight of the white matter tracts in the areas may increase, on average, because targeted selection of cortical modules necessary to participate in a given task breaks down in the presence of a brain injury. Indeed, enlargement of the activation field in stroke recovery has been observed in the literature (Lindow et al., 2016). Our results here support the previously observed phenomenon in which white matter tracts evince increased connectivity in the chronic phase after disconnection of the tracts in the acute phase (Rutgers et al., 2008).

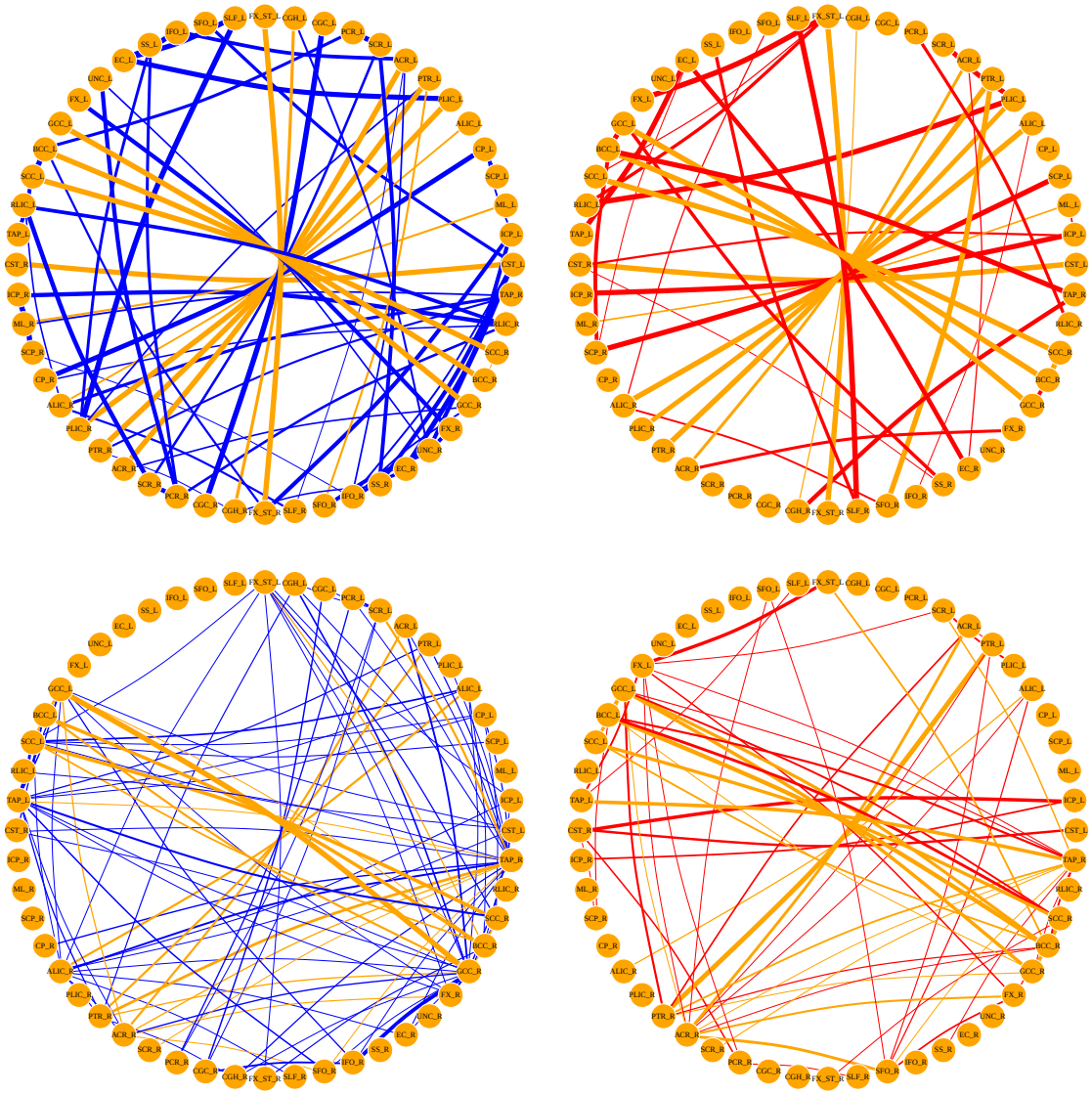

Figure 5.3: The estimated weighted brain connectivity graphs for TBI (left) and control (right) groups by Bayesian horseshoe (upper) and the frequentist fglasso (lower). Nodes names are marked with abbreviation of ROIs, which are defined in Table 5.1. The thicker edges indicate larger weights; Blue lines denote edges identified only by the TBI group, red denote edges identified only by the control group, and orange denote edges identified by both groups.

Chapter 6

Conclusions and Future works

In this dissertation, we have discussed two different approaches for constructing undirected graphical models for multivariate functional data. We also study how to apply causal inference with functional covariates based on deep model. On the heels of many outstanding contributions, our explorations provide additional pathways to estimate functional graphical model via Bayesian framework, and improves the existing functional graphical model architecture through deep learning techniques. In addition, we demonstrate how our methods can be particularly applied in the neuroimaging domain of EEG and DTI data. In this chapter, we conclude the thesis and some directions for future research.

6.1 Conclusions

In this this thesis, we first consider a Bayesian framework for graphical models associated with functional data. We proposed a fully Bayesian version of the functional graphical lasso as well as a novel functional graphical horseshoe prior. We provide also easily implemented Gibbs samplers via auxiliary variables to induce conditional conjugacy and adapting matrix partitioning techniques that have been used for other MCMC implementations of Bayesian graphical models. We compared these models to each other in several simulated scenarios. The Bayesian fglasso with fixed regularization term and the frequentist fglasso performed almost identically in terms of edge selection, as would be expected, with the exception that the Bayesian approach allows access to the full posterior distribution so that any quantity of interest can be obtained, not just the posterior

mode. The hierarchical Bayesian extension of the fglasso and the functional graphical horseshoe were directly compared to each other. The simulation results demonstrated that the functional graphical horseshoe is much better at avoiding false positives than either the frequentist or Bayesian fglasso and is still able to detect relatively weak signals. The superior balance between false positives and false negatives results in the functional graphical horseshoe exhibiting generally superior similarity to the underlying true graph. We also applied the functional graphical horseshoe to two applications in neuroimaging, one a previously studied EEG example, and the other involving DTI measurements to compare white matter integrity in people with and without a history of traumatic brain injury. The ability of the functional graphical horseshoe to avoid false positives is a critical characteristic for application in functional neuroimaging, as this field is often criticized for its abundance of false positives (Eklund et al., 2016). In addition, both EEG and MRI are notorious for having weak signal-to-noise ratios, making methods that are powerful at detecting weak signals quite useful. Indeed, the results in Section 5.2 suggest that our proposed methods may be helpful for understanding functional reorganization, a process in which many weak connections form.

In order to leverage powerful nonlinear approximation capability of deep model, we propose a graph estimator based on deep learning, called Deep FGM. Our architecture directly takes the functional data as input and does not require handcrafted bases to manually tuned functional basis transformation compared with conventional functional graphical models. The optimal bases could be learned for particular data and graph structure is trained in an end-to-end procedure. Unlike traditional dimension reduction methods like basis expansion or FPCA, which try to retain as much functional data information as possible, deep learning techniques allows us to back-propagate the task relevant gradient to adaptive FPCA module, so the projected scores in basis space specifically contains feature variation that are most relevant to the graph learning task. In addition, the learned graph estimator can generalize to different graph properties and structures. The specific domain knowledge about the graphical structure and also about data could be contained in the synthetic data and then learned by deep model. This mechanism saves some tedious mathematical derivation for different assumption in different scenarios. In the proposed deep model, we could obtain empirical performance under the synthetic settings. Finally the experiment results show that our learned graph estimator outperforms strong baseline methods in variety of scenarios and we could find some meaningful graphs in the alcoholic studies based on previous researches.

Furthermore, we investigate a model architecture that takes the functional covariate for

treatment effect estimation. Our method links the observed functional data by ordinary causal models. Based on simulation data results, our experiments demonstrate the effectiveness by embedding deep model for functional data within causal inference architecture. Causal inference with functional confounders is relatively under-exploited compared with other data settings, such as tabular data, image, text, etc. The proposed method is essential to unlock causal inference to be adopted in functional data domain. On the other hand, the counterfactual predictions of causal models have the potential to be used as part of clinical decision support system to address relevant medical challenge involving some functional confounders like ECG signal, fMRI, EEG, etc.

6.2 Implications and Recommendations for Further Works

While our work suggests promising results, particularly by extending the graphical horseshoe and deep neural networks to functional graphical models, much work still remains. For Bayesian models, the selection of the basis may impact the graph estimation, the sensitivity of basis based on the methods could be explored in the future works. Our proposed Gibbs samplers are efficient in small to moderately-sized scenarios, but the computational experiments show the practical limitations as the size of the graph grows large. Another future line of research would be investigating more computationally efficient MCMC techniques for exploring the posterior distributions, or other posterior approximation methods to infer in extremely high dimensional problems. Alternatively, the current algorithm and its implementation could be further improved by using efficient parallel computations among different blocks/nodes. Also, a particular bottleneck is the computation of inverse matrix at each iteration, and we could make use of the structure of the problem to accelerate the algorithm. For example, the whole inverse may not be explicitly required, and approximate matrix inversion algorithms might be feasible enough. There also remains deeper theoretical development of Bayesian approaches to functional graphical models, which to date the amount of data are still limited. Furthermore, the current Bayesian methods estimates the separate functional networks independently even though they may share similar patterns. We could consider Bayesian joint functional graphical lasso/horseshoe to estimate separate functional graphical models for different groups under the assumption that the groups can share similar graph properties. Lastly, our work focus on a static network structure over function domain among variables, while another possible future direction could extend our model to dynamic graphs changing over time.

Though the deep functional graphical model shows better performance on AUCs compared with Bayesian and Frequentest methods, it highly rely on the design and generation of synthetic data. With more specific domain knowledge about the type of graph structure, sparsity level and data characteristics, the in-sample distribution will be closer to the out-of-sample distribution, which can result in more accurate graph estimation. With little prior information, the standard random synthetic data may introduce some bias. Therefore, the methodology and framework of synthetic generation could be further developed for more accurate graph inference. In addition, theoretical consistency for deep neural network is still needed to be further developed.

On the other hand, the causal research we present in Chapter 4 still has gaps when functional confounding involves complicated real world data. The current gap which is caused by overfitting is still not clear: whether the overfitting is caused by functional basis layer or unobserved confoundings. Additionally, we will aim to build better balancing representations and to provide theoretical guarantees for the expected error on the counterfactual.

Bibliography

M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL `https://www.tensorflow.org/`. Software available from tensorflow.org.

O. Abdel-Hamid, A.-r. Mohamed, H. Jiang, L. Deng, G. Penn, and D. Yu. Convolutional neural networks for speech recognition. *IEEE/ACM Transactions on audio, speech, and language processing*, 22(10):1533–1545, 2014.

A. M. Alaa and M. Van Der Schaar. Bayesian inference of individualized treatment effects using multi-task gaussian processes. *Advances in neural information processing systems*, 30, 2017.

J. L. Andersson and S. N. Sotiropoulos. An integrated approach to correction for off-resonance effects and subject movement in diffusion mr imaging. *Neuroimage*, 125:1063–1078, 2016.

J. A. Aston, J.-M. Chiou, and J. P. Evans. Linguistic pitch analysis using functional principal component mixed effect models. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 59(2):297–317, 2010.

S. Athey and G. Imbens. Recursive partitioning for heterogeneous causal effects. *Proceedings of the National Academy of Sciences*, 113(27):7353–7360, 2016.

D. L. Bailey, M. N. Maisey, D. W. Townsend, and P. E. Valk. *Positron emission tomography*, volume 2. Springer, 2005.

V. Baladandayuthapani, R. Talluri, Y. Ji, K. R. Coombes, Y. Lu, B. T. Hennessy, M. A. Davies, and B. K. Mallick. Bayesian sparse graphical models for classification with application to protein expression data. *The annals of applied statistics*, 8(3):1443, 2014.

O. Banerjee, L. El Ghaoui, and A. d'Aspremont. Model selection through sparse maximum likelihood estimation for multivariate gaussian or binary data. *Journal of Machine Learning Research*, 9: 485–516, 2008.

S. Banerjee and S. Ghosal. Posterior convergence rates for estimating large precision matrices using graphical models. *Electronic Journal of Statistics*, 8(2):2111–2137, 2014.

E. Belilovsky, K. Kastner, G. Varoquaux, and M. Blaschko. Learning to discover graphical model structures. *arXiv preprint arXiv:1605.06359*, 2016.

I. Bica, A. M. Alaa, J. Jordon, and M. van der Schaar. Estimating counterfactual treatment outcomes over time through adversarially balanced representations. *arXiv preprint arXiv:2002.04083*, 2020.

A. Bloniarz, H. Liu, C.-H. Zhang, J. S. Sekhon, and B. Yu. Lasso adjustments of treatment effect estimates in randomized experiments. *Proceedings of the National Academy of Sciences*, 113(27): 7383–7390, 2016.

D. Bosq. *Linear processes in function spaces: theory and applications*, volume 149. Springer Science & Business Media, 2012.

T. Cai, W. Liu, and X. Luo. A constrained 1 minimization approach to sparse precision matrix estimation. *Journal of the American Statistical Association*, 106(494):594–607, 2011.

T. T. Cai, H. Li, W. Liu, and J. Xie. Joint estimation of multiple high-dimensional precision matrices. *Statistica Sinica*, 26(2):445, 2016.

C. Carroll, A. Gajardo, Y. Chen, X. Dai, J. Fan, P. Hadjipantelis, K. Han, H. Ji, H. Müller, and J. Wang. fdapace: Functional data analysis and empirical dynamics. *R package version 0.5*, 5, 2020a.

C. Carroll, A. Gajardo, Y. Chen, X. Dai, J. Fan, P. Z. Hadjipantelis, K. Han, H. Ji, H.-G. Mueller, and J.-L. Wang. *fdapace: Functional Data Analysis and Empirical Dynamics*, 2020b. URL `https://CRAN.R-project.org/package=fdapace`. R package version 0.5.5.

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

L. Chen, K. Lu, A. Rajeswaran, K. Lee, A. Grover, M. Laskin, P. Abbeel, A. Srinivas, and I. Mordatch. Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34:15084–15097, 2021.

L. Cheng, R. Guo, and H. Liu. Long-term effect estimation with surrogate representation. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, pages 274–282, 2021.

V. Chernozhukov, D. Chetverikov, M. Demirer, E. Duflo, C. Hansen, W. Newey, and J. Robins. Double/debiased machine learning for treatment and structural parameters, 2018.

H. A. Chipman, E. I. George, and R. E. McCulloch. Bart: Bayesian additive regression trees. *The Annals of Applied Statistics*, 4(1):266–298, 2010.

F. R. Chung and F. C. Graham. *Spectral graph theory*, volume 92. American Mathematical Soc., 1997.

A. P. Dawid and S. L. Lauritzen. Hyper markov laws in the statistical analysis of decomposable graphical models. *Annals of Statistics*, 21(3):1272–1317, 1993.

A. P. Dempster. Covariance selection. *Biometrika*, 32:95–108, 1972.

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

D. Eddelbuettel and R. François. Rcpp: Seamless R and C++ integration. *Journal of Statistical Software*, 40(8):1–18, 2011. doi: 10.18637/jss.v040.i08. URL `http://www.jstatsoft.org/v40/i08/`.

A. Eklund, T. E. Nichols, and H. Knutsson. Cluster failure: Why fMRI inferences for spatial extent have inflated false-positive rates. *Proceedings of the National Academy of Sciences*, 113 (28):7900–7905, 2016. doi: 10.1073/pnas.1602413113.

M. H. Farrell, T. Liang, and S. Misra. Deep neural networks for estimation and inference. *Econometrica*, 89(1):181–213, 2021.

J. Friedman, T. Hastie, and R. Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–441, 2008.

M. J. Funk, D. Westreich, C. Wiesen, T. Stürmer, M. A. Brookhart, and M. Davidian. Doubly robust estimation of causal effects. *American journal of epidemiology*, 173(7):761–767, 2011.

A. E. Gelfand and A. F. Smith. Sampling-based approaches to calculating marginal densities. *Journal of the American Statistical Association*, 85(410):398–409, 1990.

I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

K. Greenlaw, E. Szefer, J. Graham, M. Lesperance, F. S. Nathoo, and A. D. N. Initiative. A bayesian group sparse multi-task regression model for imaging genetics. *Bioinformatics*, 33(16):2513–2522, 2017.

M. Hämäläinen, R. Hari, R. J. Ilmoniemi, J. Knuutila, and O. V. Lounasmaa. Magnetoencephalography—theory, instrumentation, and applications to noninvasive studies of the working human brain. *Reviews of modern Physics*, 65(2):413, 1993.

T. Hastie and C. Mallows. [a statistical view of some chemometrics regression tools]: Discussion. *Technometrics*, 35(2):140–143, 1993.

N. Helwig. eegkit: Toolkit for electroencephalography data. *URL: http://CRAN. R-project. org/-package= eegkit. R package version*, pages 1–0, 2018.

J. L. Hill. Bayesian nonparametric modeling for causal inference. *Journal of Computational and Graphical Statistics*, 20(1):217–240, 2011.

S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

K. Imai and M. Ratkovic. Covariate balancing propensity score. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 76(1):243–263, 2014.

M. Jenkinson and S. Smith. A global optimisation method for robust affine registration of brain images. *Medical image analysis*, 5(2):143–156, 2001.

M. Jenkinson, P. Bannister, M. Brady, and S. Smith. Improved optimization for the robust and accurate linear registration and motion correction of brain images. *Neuroimage*, 17(2):825–841, 2002.

F. D. Johansson, U. Shalit, N. Kallus, and D. Sontag. Generalization bounds and representation learning for estimation of potential outcomes and causal effects. *arXiv preprint arXiv:2001.07426*, 2020.

I. Jolliffe. *Principal Component Analysis*. Springer, New York, 2nd edition, 2002.

K. Karhunen. Zur Spektraltheorie Stochastischer Prozesse. *Annales Academiæ Scientiarum Fennicæ*, 34:1–7, 1946.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

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

B. Koch, T. Sainburg, P. Geraldo, S. Jiang, Y. Sun, and J. G. Foster. Deep learning of potential outcomes. *arXiv preprint arXiv:2110.04442*, 2021.

J. H. Kook, K. A. Vaughn, D. M. DeMaster, L. Ewing-Cobbs, and M. Vannucci. Bvar-connect: A variational bayes approach to multi-subject vector autoregressive models for inference on brain connectivity networks. *Neuroinformatics*, 19:39–56, 2021.

A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

S. R. Künzel, J. S. Sekhon, P. J. Bickel, and B. Yu. Metalearners for estimating heterogeneous treatment effects using machine learning. *Proceedings of the national academy of sciences*, 116 (10):4156–4165, 2019.

M. Kyung, J. Gill, M. Ghosh, G. Casella, et al. Penalized regression, standard errors, and bayesian lassos. *Bayesian Analysis*, 5(2):369–411, 2010.

C. Lam and J. Fan. Sparsistency and rates of convergence in large covariance matrix estimation. *Annals of statistics*, 37(6B):4254, 2009.

Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

B. Li and E. Solea. A nonparametric graphical model for functional data with application to brain networks based on fmri. *Journal of the American Statistical Association*, 113(524):1637–1655, 2018.

Y. Li and T. Hsing. Uniform convergence rates for nonparametric regression and principal component analysis in functional/longitudinal data. *The Annals of Statistics*, 38(6):3321–3351, 2010.

Y. Li, B. A. Craig, and A. Bhadra. The graphical horseshoe estimator for inverse covariance matrices. *Journal of Computational and Graphical Statistics*, pages 1–24, 2019.

B. Lim. Forecasting treatment responses over time using recurrent marginal structural networks. *advances in neural information processing systems*, 31, 2018.

J. Lindow, M. Domin, M. Grothe, U. Horn, S. B. Eickhoff, and M. Lotze. Connectivity-based predictions of hand motor outcome for patients at the subacute stage after stroke. *Frontiers in human neuroscience*, 10:101, 2016.

M. Loeve. *Probability Theory. Van Nostrand.* 1963.

Y. Ma and V. Tresp. Causal inference under networked interference and intervention policy enhancement. In *International Conference on Artificial Intelligence and Statistics*, pages 3700–3708. PMLR, 2021.

E. Makalic and D. F. Schmidt. A simple sampler for the horseshoe estimator. *IEEE Signal Processing Letters*, 23:179–182, 2016.

W. S. McCulloch and W. Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.

G. J. McLachlan. Mahalanobis distance. *Resonance*, 4(6):20–26, 1999.

N. Meinshausen, P. Bühlmann, et al. High-dimensional graphs and variable selection with the lasso. *The annals of statistics*, 34(3):1436–1462, 2006.

A. Mohammadi, E. C. Wit, et al. Bayesian structure learning in sparse gaussian graphical models. *Bayesian Analysis*, 10(1):109–138, 2015.

J. S. Morris, V. Baladandayuthapani, R. C. Herrick, P. Sanna, and H. Gutstein. Automated analysis of quantitative image data using isomorphic functional mixed models, with application to proteomics data. *The annals of applied statistics*, 5(2A):894, 2011.

J. Niu, B. Hur, J. Absher, and D. A. Brown. Bayesian regularization for functional graphical models. *arXiv preprint arXiv:2110.05575*, 2021.

J. Pearl. *Causality*. Cambridge university press, 2009.

J. Pearl and D. Mackenzie. *The book of why: the new science of cause and effect*. Basic books, 2018.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.

C. Peeters, B. Bilgrau, and W. Wieringen. Rags2ridges: Ridges estimation of precision matrices from high-dimesnional data. r package version 2.2. 5. 2021.

J. Peng, P. Wang, N. Zhou, and J. Zhu. Partial correlation estimation by joint sparse regression models. *Journal of the American Statistical Association*, 104(486):735–746, 2009.

D. Perdices, J. E. L. de Vergara, and J. Ramos. Deep-fda: Using functional data analysis and neural networks to characterize network services time series. *IEEE Transactions on Network and Service Management*, 18(1):986–999, 2021.

X. Qiao, S. Guo, and G. M. James. Functional graphical models. *Journal of the American Statistical Association*, 114(525):211–222, 2019.

X. Qiao, C. Qian, G. M. James, and S. Guo. Doubly functional graphical models in high dimensions. *Biometrika*, 107(2):415–431, 2020.

H. Qiu, F. Han, H. Liu, and B. Caffo. Joint estimation of multiple graphical models from high dimensional time series. *Journal of the Royal Statistical Society. Series B, Statistical Methodology*, 78(2):487, 2016.

B. Rajaratnam, H. Massam, and C. M. Carvalho. Flexible covariance estimation in graphical gaussian models. *The Annals of Statistics*, 36(6):2818–2849, 2008.

J. O. Ramsay and C. Dalzell. Some tools for functional data analysis. *Journal of the Royal Statistical Society: Series B (Methodological)*, 53(3):539–561, 1991.

M. D. M. Reddy, M. S. M. Basha, M. M. C. Hari, and M. N. Penchalaiah. Dall-e: Creating images from text. *UGC Care Group I Journal*, 8(14):71–75, 2021.

P. R. Rosenbaum and D. B. Rubin. The central role of the propensity score in observational studies for causal effects. *Biometrika*, 70(1):41–55, 1983.

D. B. Rubin. Estimating causal effects of treatments in randomized and nonrandomized studies. *Journal of educational Psychology*, 66(5):688, 1974.

H. Rue and L. Held. *Gaussian Markov random fields: theory and applications*. CRC press, 2005.

D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

D. Rutgers, F. Toulgoat, J. Cazejust, P. Fillard, P. Lasjaunias, and D. Ducreux. White matter abnormalities in mild traumatic brain injury: a diffusion tensor imaging study. *American Journal of Neuroradiology*, 29(3):514–519, 2008.

F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini. The graph neural network model. *IEEE transactions on neural networks*, 20(1):61–80, 2008.

S. L. Schwartz, F. Li, and F. Mealli. A bayesian semiparametric approach to intermediate variables in causal inference. *Journal of the American Statistical Association*, 106(496):1331–1344, 2011.

U. Shalit, F. D. Johansson, and D. Sontag. Estimating individual treatment effect: generalization bounds and algorithms. In *International Conference on Machine Learning*, pages 3076–3085. PMLR, 2017.

H. Shappell, B. S. Caffo, J. J. Pekar, and M. A. Lindquist. Improved state change estimation in dynamic functional connectivity using hidden semi-markov models. *NeuroImage*, 191:243–257, 2019.

C. Shi, D. M. Blei, and V. Veitch. Adapting neural networks for the estimation of treatment effects. *arXiv preprint arXiv:1906.02120*, 2019.

B. W. Silverman. Smoothed functional principal components analysis by choice of norm. *The Annals of Statistics*, 24(1):1–24, 1996.

S. M. Smith. Fast robust automated brain extraction. *Human brain mapping*, 17(3):143–155, 2002.

E. Solea and H. Dette. Nonparametric and high-dimensional functional graphical models. *arXiv preprint arXiv:2103.10568*, 2021.

E. Solea and B. Li. Copula gaussian graphical models for functional data. *Journal of the American Statistical Association*, pages 1–13, 2020.

J. D. Storey. The positive false discovery rate: a bayesian interpretation and the q-value. *The Annals of Statistics*, 31(6):2013–2035, 2003.

T. Sun and C.-H. Zhang. Sparse matrix inversion with scaled lasso. *The Journal of Machine Learning Research*, 14(1):3385–3418, 2013.

B. Thind, K. Multani, and J. Cao. Deep learning with functional inputs. *Journal of Computational and Graphical Statistics*, (just-accepted):1–27, 2022.

S. L. Van Der Pas, B. J. Kleijn, and A. W. Van Der Vaart. The horseshoe estimator: Posterior concentration around nearly black vectors. *Electronic Journal of Statistics*, 8(2):2585–2618, 2014.

L. Vandenberghe, S. Boyd, and S.-P. Wu. Determinant maximization with linear matrix inequality constraints. *SIAM journal on matrix analysis and applications*, 19(2):499–533, 1998.

V. N. Vapnik. An overview of statistical learning theory. *IEEE transactions on neural networks*, 10 (5):988–999, 1999.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

S. Wager and S. Athey. Estimation and inference of heterogeneous treatment effects using random forests. *Journal of the American Statistical Association*, 113(523):1228–1242, 2018.

H. Wang. Bayesian graphical lasso models and efficient posterior computation. *Bayesian Analysis*, 7(4):867–886, 2012.

S. Wang and G. Cao. Robust deep neural network estimation for multi-dimensional functional data. *arXiv preprint arXiv:2205.09604*, 2022.

S. Wang, G. Cao, Z. Shang, and A. D. N. Initiative. Estimation of the mean function of functional data via deep neural networks. *Stat*, 10(1):e393, 2021.

L. Wasserman and K. Roeder. High dimensional variable selection. *Annals of statistics*, 37(5A): 2178, 2009.

R. Xiang, K. Khare, and M. Ghosh. High dimensional posterior convergence rates for decomposable graphical models. *Electronic Journal of Statistics*, 9(2):2828–2854, 2015.

Y. Yan, J. Zhu, M. Duda, E. Solarz, C. Sripada, and D. Koutra. Groupinn: Grouping-based interpretable neural network for classification of limited, noisy brain data. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 772–782, 2019.

F. Yao, H.-G. Müller, and J.-L. Wang. Functional data analysis for sparse longitudinal data. *Journal of the American Statistical Association*, 100(470):577–590, 2005.

J. Yao, J. Mueller, and J.-L. Wang. Deep learning for functional data analysis with adaptive basis layers. *arXiv preprint arXiv:2106.10414*, 2021a.

L. Yao, Z. Chu, S. Li, Y. Li, J. Gao, and A. Zhang. A survey on causal inference. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 15(5):1–46, 2021b.

M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(1):49–67, 2006.

M. Yuan and Y. Lin. Model selection and estimation in the Gaussian graphical model. *Biometrika*, 94(1):19–35, 2007.

J. Zapata, S.-Y. Oh, and A. Petersen. Partial separability and functional graphical models for multivariate gaussian processes. *arXiv preprint arXiv:1910.03134*, 2019.

L. Zhang, V. Baladandayuthapani, Q. Neville, K. Quevedo, and J. S. Morris. Bayesian functional graphical models. arXiv preprint arXiv:2108.05034, 2021a.

L. Zhang, V. Baladandayuthapani, Q. Neville, K. Quevedo, and J. S. Morris. Bayesian functional graphical models. *arXiv preprint arXiv:2108.05034*, 2021b.

X. L. Zhang, H. Begleiter, B. Porjesz, W. Wang, and A. Litke. Event related potentials during object recognition tasks. *Brain Research Bulletin*, 38(6):531–538, 1995.

H. Zhu, N. Strawn, and D. B. Dunson. Bayesian graphical models for multivariate functional data. *The Journal of Machine Learning Research*, 17(1):7157–7183, 2016.

Y. Zhu, X. Shen, and W. Pan. Structural pursuit over multiple undirected graphs. *Journal of the American Statistical Association*, 109(508):1683–1696, 2014.