Topology Inference of Directed Graphs by Gaussian Processes with Sparsity Constraints

Chen Cui, *Student member, IEEE*, Paolo Banelli, *Senior member, IEEE*, Petar M. Djurić, *Fellow, IEEE*

***Abstract*—In machine learning applications, the data are often high-dimensional and intricately related. It is often of interest to find the underlying structure and the causal relationships among the data and represent the findings with directed graphs. In this paper, we study multivariate time series, where each series is associated with a node of a graph, and where the objective is estimating the topology of a sparse graph that reflects how the nodes of the graph affect each other, if at all. We propose a novel fully Bayesian method with a sparse prior on the hyperparameters. The proposed method allows for nonlinear and multiple lag relationships among the time series. The method is based on Gaussian processes, and it treats the entries of the graph adjacency matrix as hyperparameters. The method employs a modified automatic relevance determination (ARD) kernel and allows for learning the mapping function from selected past data to current data as edges of a graph. We show that the resulting adjacency matrix provides the intrinsic structure of the graph and answers questions related to causality. Numerical tests show that the proposed method has comparable or better performance than state-of-the-art methods.**

***Index Terms*—Topology inference, Gaussian processes, causality, ARD kernel, Bayes, Hamiltonian Monte Carlo**

I. Introduction

In many science and engineering problems, it is essential to determine the underlying structure of observed data/signals as structures provide valuable insights about the system where the data originate. Given the significance of the problem, it is not surprising that learning underlying structures/topology of multidimensional data has been well studied in many fields, including biology [1], social sciences [2], and finance [3]. For example, in biology, searching the functional connectivity within different brain areas; in social science, analyzing relationships between individuals on social platforms and even entire societies; and in finance, studying the interdependence of financial entities [4] have been of great interest.

There are several approaches to the problem of finding the topology of a graph. One common class of methods is to estimate the Laplacian or adjacency matrix of the graph under the assumption that the data are generated according to a Gaussian Markov Random Field [5], [6]. However, these methods produce undirected graphs because the estimated adjacency matrices are symmetric, which in turn limits the applicability of the methods. To overcome this hindrance, there has been significant research on estimating directed graphs. A prominent approach among them is based on vector autoregression (VAR) models [7]. For instance, in [8] and [9], VAR models were used to quantify Granger causality of fMRI data and to recover gene regulatory networks, respectively. Further, kernel-based VAR models were proposed to identify non-linear topologies [10], [11].

Another category of methods for inference of directed graph topology is based on structural equation models (SEMs) [12], [13]. They were initially proposed for causality learning [14], [15]. SEMs are used to infer simultaneous relationships while taking exogenous variables into account. This overcomes common problems associated with the existence of exogenous nodes or variables in topology inference and causality learning. In [7], it was proposed to combine VAR and SEM models, ending with SVAR models. These models consider both simultaneous and lagged dependencies of time series. In [16] and [17], an SVAR model was used for inference of dynamic, and non-linear, directed graphs, respectively. In addition to the aforementioned approaches, from the perspective of graph signal processing, the adjacency matrix can be viewed as a graph filter [18]–[20]. The adjacency matrix can also be considered to be related to Granger causality [21], motivating the interpretation of a network of structural data dependencies as an evidence of causal relationships.

In [22] we proposed a method based on Gaussian processes (GPs) to reveal the structure and causality of a graph by assuming that the function that generates data on each node is drawn from a Gaussian process rather than being deterministic. While this method performed well under various conditions, it is still susceptible to overfitting [23], particularly for large networks with numerous parameters and resulting in the estimation of a dense graph topology. To address this issue, we propose a novel fully Bayesian method to infer the topology of directed graphs from time-series data observed at the nodes. We assume nonlinear functional relationships among the signals that evolve on the nodes of the graph, where the functions, too, need to be estimated. Specifically, we suggest employing fully Bayesian inference of Gaussian processes with sparse priors as a tool for learning the unknown structural mappings. We recall that the GPs are data-efficient, flexible, and can cooperate with information from the priors. In this paper, the arguments of the functions are not only past local data but also past data from other nodes. The GPs are based on

C. Cui is with the Department of Electrical and Computer Engineering, Stony Brook University, Stony Brook, NY, USA (e-mail:chen.cui@stonybrook.edu)

P. Banelli is with the Department of Engineering, University of Perugia, Perugia, Italy (e-mail: paolo.banelli@unipg.it)

P. M. Djurić is with the Department of Electrical and Computer Engineering, Stony Brook University, Stony Brook, NY, USA (e-mail:petar.djuric@stonybrook.edu)

a predefined kernel, where the hyperparameters of the kernel encode the relevance of each argument and, thus, allow for automatic relevance determination. We introduce a modified version of the automatic relevance determination (ARD) kernel to set priors on the hyperparameters. Our method enables simultaneous learning of the strength of influence from all the selected previous data, which means that the method is not dependent on the knowledge of specific time delays. In addition, the proposed method produces marginalized posteriors of the hyperparameters.

The thought behind the ARD kernel has been widely used in machine learning since it was formulated in the framework of neural networks [24]. For example, [25] and [26] employed ARD kernels for feature selections in support vector machines (SVMs). Additionally, [27] proposed a method to infer causality between two-time series using the hyperparameter of Gaussian processes with an ARD kernel. Moreover, the idea of making inferences based on hyperparameters has been successfully applied in many ways. For instance, [28] introduced an online method for detecting change points, and [29] developed a time-varying hyperparameter model to estimate time-varying functions.

The main contribution of this paper is the introduction of a novel fully Bayesian Gaussian process (fBGP)-based method for inferring a *sparse* graph topology from time series data observed at the nodes. The proposed method does not make any assumptions regarding the functional relationship between the signals of the nodes, except that the functions are sufficiently smooth. The method also allows for the presence of multiple time lag dependencies. Further, the method provides the posterior of the hyperparameters of the model, enabling the evaluation of their uncertainty and providing additional insights into the graph topology. The proposed method can also be used to make hard decisions if edges between nodes exist or not and thereby produce a binary graph topology.

The rest of the paper is organized as follows: In Section II, we give a brief overview of graphs, GPs, the ARD kernel, and the Hamiltonian Monte Carlo method. Then in Sections III and IV, we describe the proposed model and solution, respectively. In Section V, we present in detail our algorithm. In Section VI, we show numerical results on different cases, and in the last section, we provide concluding remarks and outline directions for future work.

II. Preliminaries

A. Graph and Graph Signal

Consider a graph denoted by $\mathcal{G}(\mathcal{V},\mathcal{E},\mathbf{W})$, where $\mathcal{V}$ is a set of N nodes, $\mathcal{E}$ is a set of edges, and $\mathbf{W}$ is the graph's adjacency matrix whose elements provide information about the relationships between the nodes on the graph. This matrix can be symmetric or asymmetric and, thus, implying if the graph is undirected or directed, respectively. The (n,m)th entry of $\mathbf{W}$ is denoted by $w_{nm} \in [0,+\infty)$, and its value represents the strength of coupling of the nth and mth nodes. For an undirected graph, we have $w_{nm} = w_{mn}$, and if its value is non-zero, there is an edge *between* node m and node n. Similarly, for a directed graph, if w_{nm} is non-zero, there is an edge *pointing from* node m *to* node n, and the value of w_{nm} represents the strength of how much node m affects node n. With a binary adjacency matrix $\mathbf{A}$, an entry $a_{nm}=1$ indicates the presence of an edge *pointing from* node m *to* node n.

On a given graph, its signals are defined as follows. Consider an unordered set of data $\mathcal{S}=\{s_{k_1},...,s_{k_N}\}$, which are associated with $\mathcal{G}$. We assume that to each node in $\mathcal{G}$ we attach data from $\mathcal{S}$, s_{k_l}, where k_l, $l=1,\ldots,N$ are indices of the nodes. We can then order the data according to the node indices, and this yields the N-tuple $\mathbf{s}=\{s_1,...s_n,...,s_N\}$. We can think of $\mathbf{s}$ as a graph signal over $\mathcal{G}$ [30]. The nth element s_n in $\mathbf{s}$ is indexed by the node n of $\mathcal{G}$.

TABLE I
List of some symbols and notations

t,i,j	–	discrete time indices
T	–	number of observed data samples
n,m	–	node indices
N	–	size of the graph
λ	–	a discrete delay time index
Λ	–	maximum delay lag
$\mathbf{x}_t \in \mathbb{R}^{N\Lambda\times 1}$	–	an input vector at time t
$\mathbf{X} \in \mathbb{R}^{(T-\Lambda)\times N\Lambda}$	–	a matrix with rows composed of the input vectors
$\mathsf{y}_t = \mathbf{Y}_{1:N,t}$	–	a vector of graph signals on graph $\mathcal{G}$ at time t
$\mathbf{y}_n = \mathbf{Y}^\top_{n,\Lambda+1:T}$	–	a vector of graph signals at node n
$\mathbf{Y} \in \mathbb{R}^{N\times T}$	–	a matrix that collects all the graph signals
$\mathbf{W}^\lambda$	–	weighted adjacency matrix for delay λ
$w^\lambda_{i,j}$	–	(i,j)th element of $\mathbf{W}^\lambda$
$\mathbf{A}^\lambda$	–	binary adjacency matrix for delay λ
$a^\lambda_{i,j}$	–	(i,j)th element of $\mathbf{A}^\lambda$
$\mathbf{f}_n$	–	function at node n
$\mathcal{GP}(\cdot,\cdot)$	–	a Gaussian process (GP)
$\mathbf{m}(\cdot)$	–	mean function of a GP
$\mathbf{K}(\cdot)$	–	kernel function of a GP
l	–	length-scale of a kernel function
σ^2_n	–	amplitude of a kernel function
σ^2_v	–	noise variance
a,b,α,β	–	hyper-hyperparameters of the priors

B. Gaussian Processes

Gaussian processes are a class of stochastic processes, which are used in machine learning for modeling functions [23]. More specifically, let $\{\mathbf{x}_t, y_t\}$, $t=1,2,\ldots,T$, be T input-output values, with $\mathbf{x}^\top_t$ being a row vector and y_t being a scalar, $\mathbf{y}=[y_1\ y_2\ldots y_T]^\top$, $\mathbf{y}=\mathbf{f}(\mathbf{X})$, with $\mathbf{f}\in\mathbb{R}^{T\times 1}$, and $\mathbf{X}\in\mathbb{R}^{T\times N}$ being a matrix whose rows represent the inputs to the function $\mathbf{f}$, that is,[1]

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_1^\top \\ \mathbf{x}_2^\top \\ \vdots \\ \mathbf{x}_T^\top \end{bmatrix}, \qquad \mathbf{y} = \mathbf{f}(\mathbf{X}) = \begin{bmatrix} f(\mathbf{x}_1) \\ f(\mathbf{x}_2) \\ \vdots \\ f(\mathbf{x}_T) \end{bmatrix}. \tag{1}$$

The idea behind GPs is to assume that the function's samples are jointly drawn from a Gaussian distribution instead of being deterministic. Mathematically, we have

$$\mathbf{f} \sim \mathcal{GP}\left(\mathbf{m}(\mathbf{X}), \mathbf{K}_{\boldsymbol{\theta}}(\mathbf{X})\right), \tag{2}$$

[1]In this section, the matrix $\mathbf{X}$ is of size $T\times N$. In the rest of the paper, it is $(T-\Lambda)\times N$, as suggested in Table I.

where $\mathbf{m}(\mathbf{X})$ is the mean function, $\mathbf{K}_{\boldsymbol{\theta}}(\mathbf{X})$ is the covariance (kernel) function of the process, and $\boldsymbol{\theta}$ is a vector of hyper-parameters of the GP, i.e.,

$$\begin{aligned} \mathbf{m}(\mathbf{X}) &= \mathbb{E}[\mathbf{f}(\mathbf{X})], \\ k_{\boldsymbol{\theta}}(\mathbf{x}_i, \mathbf{x}_j) &= \mathbb{E}[(f(\mathbf{x}_i) - m(\mathbf{x}_i))(f(\mathbf{x}_j) - m(\mathbf{x}_j))]. \end{aligned} \tag{3}$$

In practice, without loss of generality, we let the mean function to be $\mathbf{0}$, and by definition, the kernel must be positive definite [23].

C. Automatic Relevance Determination Kernel

A commonly used kernel for GPs is the squared exponential (SE) kernel with the following form:

$$k_{\boldsymbol{\theta}}(\mathbf{x}_i, \mathbf{x}_j) = \sigma^2 \exp\left(-\frac{(\mathbf{x}_i - \mathbf{x}_j)^\top(\mathbf{x}_i - \mathbf{x}_j)}{2l^2}\right), \tag{4}$$

where $\boldsymbol{\theta} = \{\sigma^2, l\}$, σ^2 is one hyperparameter of the GP that represents the amplitude of the kernel, and l is another hyperparameter, also called a characteristic *length−scale*. The symbol l reflects the relationship between the distance one moves in the input space and how the function value changes in the output space [23]. Informally, if l is very small, the output is very sensitive to the change of the input, but if l is very large, small changes in the input do not affect the output much.

The ARD kernel is an extension of the SE kernel with the following form:

$$k_{\boldsymbol{\theta}}^{ARD}(\mathbf{x}_i, \mathbf{x}_j) = \sigma^2 \exp\left(-\sum_{m=1}^{N} \frac{(x_{i,m} - x_{j,m})^2}{2l_m^2}\right), \tag{5}$$

where $x_{m,i}$ is the mth entry of $\mathbf{x}_i$. Different from the SE kernel, for each component of the input vector, the ARD kernel assigns a different length−scale, l_m. If we think the input $\mathbf{x}$ is a vector of features, we can use the length−scale to decide which features to discard from the input (the ones with small contributions or no contributions) and which not. In the sequel, we exploit the ARD kernel to infer (causal) relationships among the observed data.

D. Hamiltonian Monte Carlo

Hamiltonian Monte Carlo (HMC) is a powerful algorithm of the MCMC family that uses Hamiltonian dynamics for drawing samples from a target distribution that are more different than current samples while maintaining small probability of rejecting these samples. Unlike other MCMC methods, HMC is not plagued by issues like random walk behavior and sensitivity to correlated parameters [31].

In HMC, we define the target variable by z, and we introduce an auxiliary variable r. We refer to z as the state of a modeled dynamical system (or its location) and to r as its momentum variable. We are interested in the dynamics of z under continuous time t. From physics we have the relationship

$$r = \frac{\mathrm{d}z}{\mathrm{d}t}. \tag{6}$$

Further, we can express the probability distribution of z by

$$p(z) \propto \exp(-E(z)), \tag{7}$$

where $E(z)$ is the potential energy of the dynamical system when it is in state z. For the momentum we have

$$\frac{\mathrm{d}r}{\mathrm{d}t} = \frac{\mathrm{d}E(z)}{\mathrm{d}z}. \tag{8}$$

The total energy of the system is

$$H(z, r) = E(z) + V(r), \tag{9}$$

where $V(r)$ is the kinetic energy of the dynamical system, which we express by

$$V(r) = \frac{1}{2}r^2, \tag{10}$$

and H is the Hamiltonian function.

Then, using (6), (8), (9), and (10), we can write

$$\frac{dz}{dt} = \frac{\partial H}{\partial r} = \frac{\partial V}{\partial r}, \tag{11}$$

$$\frac{dr}{dt} = -\frac{\partial H}{\partial z} = -\frac{\partial V}{\partial z} - \frac{\partial E}{\partial z}. \tag{12}$$

The joint distribution of z and r can be expressed by

$$\pi(z, r) \propto \exp(-H(z, r)). \tag{13}$$

During the evolution of the dynamical system, the value of H remains constant. Due to Liouville's theorem, the Hamiltonian system also preserves its volume in phase space (meaning that the region of the space of (z, r) changes its shape but not its volume). The constancy of H and the preservation of volume in phase space suggest that the Hamiltonian dynamics induces the invariance of $\pi(z, r)$. The invariance of H notwithstanding, the variable z and r vary. When we integrate the Hamiltonian dynamics over a finite time interval, we can make large changes to z systematically and still have high probability of acceptance.

In practice, the integration of the Hamiltonian equations when implemented straightforwardly introduces numerical errors. One method that allows for Liuoville's theorem to hold exactly is known as leapfrog discretization and it is based on alternative updates of the target and auxiliary variables. The updating of the discrete-time approximations of $\hat{z}$ and $\hat{r}$ according to this scheme is given by

$$\begin{aligned} \hat{r}_{t+\Delta t/2} &= \hat{r}_t - \frac{\Delta t}{2}\frac{\partial V}{\partial z}|_{\hat{z}_t}, \\ \hat{z}_{t+\Delta t} &= \hat{z}_t + \Delta t \hat{r}_{t+\Delta t/2}, \\ \hat{r}_{t+\Delta t} &= \hat{r}_{t+\Delta t/2} - \frac{\Delta t}{2}\frac{\partial V}{\partial z}|_{\hat{z}_{t+\Delta t}}, \end{aligned} \tag{14}$$

where Δt is the step size. For a detailed tutorial on HMC sampling, interested readers could refer to [32].

III. Model Description

Assume $\mathbf{Y} \in \mathbb{R}^{N\times T}$ collects the graph signals on a graph $\mathcal{G}$ with N nodes over the entire timeline from $t = 1$ to T. A column vector y_t[2] of $\mathbf{Y}$ is composed of measurements at time

[2]Note that y_t is a column vector identical to the tth column of $\mathbf{Y}$ and $\mathbf{y}_n$ is a column vector whose elements are identical to the elements of the nth row of $\mathbf{Y}$.

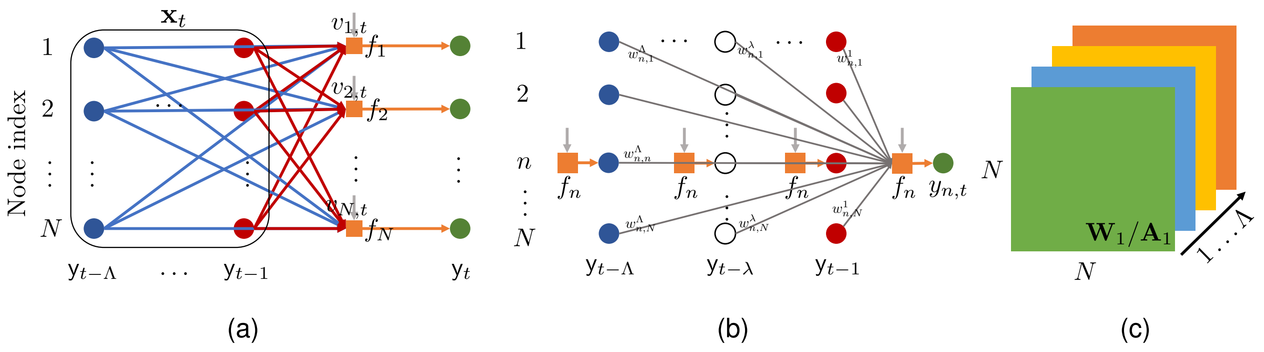

Fig. 1. Description of the model. (a) Illustration of the network, where the arrows' directions suggest directions of influence. Different colors represent different time lags. (b) Illustration of the dimension of the adjacency matrix A/W, where each layer represents the adjacency matrix for a specific lag. (c) Illustration of the decomposed network.

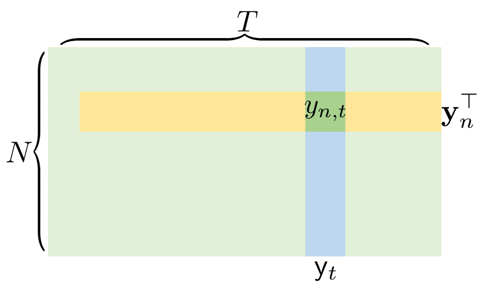

Fig. 2. Data description. The green rectangular represents the graph signal $\mathbf{Y}$, the column vector refers to the graph signals at time t over the graph, and the row vector refers to the graph signals on node n but time changes.

t, and $y_{n,t}$ is a graph signal of node n at t. Further, assume that $y_{n,t}$ is a function of the previous data on all (or some of) the nodes of $\mathcal{G}$. Specifically, we consider the data model

$$y_{n,t} = f_n(\mathbf{x}_t) + v_{n,t}, \tag{15}$$

where $f_n : \mathbf{x}_t \to y_{n,t}$ is an unknown function, $\mathbf{x}_t \in \mathbb{R}^{N\Lambda\times 1}$ is a vector that contains the signals from all the nodes from time instant $t-\Lambda$ to $t-1$, and where Λ is a positive integer representing the largest time lag of a sample from the graph that can affect a current sample of signal on the graph. Figure 1a displays the considered model.

For the signal of the nth node, we have

$$\mathbf{y}_n = \mathbf{f}_n(\mathbf{X}) + \mathbf{v}_n, \tag{16}$$

where $\mathbf{v}_n := [v_{n,\Lambda+1}, \dots, v_{n,T}]^\top \in \mathbb{R}^{(T-\Lambda)\times 1}$, $\mathbf{X} \in \mathbb{R}^{(T-\Lambda)\times N\Lambda}$ is a collection of inputs $\mathbf{x}_t$ from $t = \Lambda+1$ to $t = T$, and $\mathbf{y}_n \in \mathbb{R}^{(T-\Lambda)\times 1}$ is the graph signal of node n from $t = \Lambda+1$ to $t = T$, given as

$$\mathbf{y}_n := [y_{n,\Lambda+1} \dots y_{n,T}]^\top. \tag{17}$$

The function $\mathbf{f}_n : \mathbf{X} \to \mathbf{y}_n$ is modeled as a Gaussian process, i.e.,

$$\mathbf{f}_n \sim \mathcal{GP}(\mathbf{0}, \mathbf{K}_n), \tag{18}$$

where $\mathbf{K}_n \in \mathbb{R}^{T-\Lambda\times T-\Lambda}$ is the kernel function and has the following form:

$$\mathbf{K}_n = \sigma_n^2 \begin{bmatrix} k^n_{1,1} & k^n_{1,2} & \dots & k^n_{1,T-\Lambda} \\ & \vdots & k^n_{i,j} & \vdots \\ k^n_{T-\Lambda,1} & k^n_{T-\Lambda,2} & \dots & k^n_{T-\Lambda,T-\Lambda} \end{bmatrix}, \tag{19}$$

where σ_n^2 is a scalar hyperparameter, while the entries $k^n_{i,j}$ of the kernel matrix, which models the cross-covariance of the ith and jth lags for the nth node signal depend on the past graph signals at all the other nodes and on some length-scale hyperparameters, as we will explain in detail in the following section. By formulating the model by (16), we basically partition the entire graph into N star subgraphs, each one associated to a single GP, that has to be estimated by observing the (common) input and the node-specific output $\mathbf{y}_n$. This approach simplifies the inference problem by breaking it down into smaller more manageable subproblems. Throughout the rest of this paper, we focus on node n, and all derivations are based on this node for simplicity. Fig. 1b provides a visualization of the proposed model.

IV. Proposed Solution

A. Modified Automatic Relevance Determinant Kernel

We propose to model the entry $k^n_{i,j}$ in (19) by

$$k^n_{i,j} = \exp\left(-\sum_{\lambda=1}^{\Lambda}\sum_{m=1}^{N}(w^\lambda_{nm})^2 \frac{(x_{i,(\lambda-1)N+m} - x_{j,(\lambda-1)N+m})^2}{2}\right), \tag{20}$$

where the $w^\lambda_{nm} \in [0, +\infty)$ are the length-scale hyperparameters.[3] With (20), as anticipated by (16), we model the inputs of a specific node n by the *past* Λ values of the graph signals coming from all the graph nodes, including n itself. To infer the topology of the directed graph, we propose to use the set of hyperparameters $\mathbf{w}_n \in \mathbb{R}^{\Lambda N\times 1}$, where

$$\mathbf{w}_n^\top = [w^1_{n1}\ w^1_{n2} \dots w^1_{nN}\ w^2_{n1} \dots\ w^2_{nN} \dots\ w^\Lambda_{nN}]. \tag{21}$$

[3]Note that from here on we work with w^λ_{nm} rather than ℓ as defined in (4), where $w^\lambda_{nm} = 1/\ell_{nm}$. The reason is explained further in the text.

Indeed, w_{nm}^{λ} can indicate which set of nodes $m = 1, \ldots, N$ contributes to the evolution of the nth signal and which do not, thus revealing which edges exist in the graph. Specifically, small values of w_{nm}^{λ} suggest that a slight change in the history of node m will statistically cause a small change to node n for a specific delay index λ, while large values of w_{nm}^{λ} indicate the opposite.

Equation (20) is a modified version of the ARD kernel, which allows for reasonable sparse priors on the length-scale hyperparameters, making it more convenient to use. In the original ARD kernel, the lack of an edge requires the length scale to be infinite. However, the modified ARD kernel models the length-scale hyperparameters for non-edge entries as zeros. Plenty of distributions have sharp spikes at zero (high probability at zero), and they can be used as priors to enforce sparsity in the estimated graph topology.

To conceptualize the model further, we can think of each delay as a different layer of a weighted adjacency matrix organized as a tensor, as shown in Fig. 1c.

B. Fully Bayesian GPs with Sparse Constraints

In practice, a common method to train a GP model and obtain the optimal set of hyperparameters is to maximize the marginal likelihood in equation (25), rather than the marginal hyperparameter posterior in (26), both presented below. This is because the marginal likelihood is analytically tractable when the additive observation noise is Gaussian, as is commonly assumed in GP regression [23] [33]. This approach is called type II maximum likelihood (ML-II) estimation. Although the performance of ML-II has been shown to be good enough in many cases [27], there are still some problems to be considered. First, the non-convexity of the marginal likelihood surface with multiple modes can easily lead to overfitting of the model, particularly when dealing with many hyperparameters. Second, the results might converge to a local optimal near the starting point, making the ML-II method susceptible to the choice of initial points of the gradient-based optimizer [23] [33].

We address these issues by proposing an fBGP method. We refer to it as a fully Bayesian method because we estimate all the unknowns of the GP model under the Bayesian inference paradigm via HMC. Thus, by exploiting the posterior distribution, which combines the likelihood and prior, our model can provide uncertainty estimates and is more robust to overfitting. However, the posterior distribution over the hyperparameters is highly intractable, and consequently we rely on a Markov Chain Monte Carlo (MCMC) method for sampling. The proposed approach can estimate a sparse graph topology and is robust to different thresholds for deciding a true edge.

1) The FBGP Framework: Given the data model described by (16), the main distributions of the fBGP's framework are given by

$$\text{priors over the hyperparameters:} \quad \begin{cases} \mathbf{w}_n & \sim \ p(\mathbf{w}_n) \\ \sigma_n^2 & \sim \ p(\sigma_n^2) \\ \sigma_v^2 & \sim \ p(\sigma_v^2) \end{cases}, \tag{22}$$

$$\text{a prior over the function:} \quad \mathbf{f}_n | \mathbf{X}, \mathbf{w}_n, \sigma_n^2 \sim \mathcal{N}(\mathbf{0}, \mathbf{K}_n), \tag{23}$$

$$\text{a data likelihood:} \quad \mathbf{y}_n | \mathbf{f}_n, \sigma_v^2 \sim \mathcal{N}(\mathbf{f}_n, \sigma_v^2 \mathbf{I}). \tag{24}$$

Based on the generative model (22)-(24), the marginal likelihood of the hyperparameters, marginalized over the function $\mathbf{f}_n$, can be written as

$$\begin{aligned} & p(\mathbf{y}_n | \mathbf{X}, \mathbf{w}_n, \sigma_n^2, \sigma_v^2) \\ & = \int p(\mathbf{y}_n | \mathbf{f}_n, \mathbf{X}, \mathbf{w}_n, \sigma_n^2, \sigma_v^2) p(\mathbf{f}_n | \mathbf{X}, \mathbf{w}_n, \sigma_n^2, \sigma_v^2) d\mathbf{f}_n. \end{aligned} \tag{25}$$

By Bayes' rule, the marginal posterior of the hyperparameters is obtained by the marginal likelihood and the priors over the hyperparameters according to

$$\begin{aligned} & p(\mathbf{w}_n, \sigma_n^2, \sigma_v^2 | \mathbf{X}, \mathbf{y}_n) \\ = \ & \frac{p(\mathbf{y}_n | \mathbf{X}, \mathbf{w}_n, \sigma_n^2, \sigma_v^2) p(\mathbf{w}_n) p(\sigma_n^2) p(\sigma_v^2)}{\mathcal{Z}} \\ \propto \ & p(\mathbf{y}_n | \mathbf{X}, \mathbf{w}_n, \sigma_n^2, \sigma_v^2) p(\mathbf{w}_n) p(\sigma_n^2) p(\sigma_v^2), \end{aligned} \tag{26}$$

where $\mathcal{Z}$ is the normalization constant, given by

$$\mathcal{Z} = \int p(\mathbf{y}_n | \mathbf{X}, \mathbf{w}_n, \sigma_n^2, \sigma_v^2) p(\mathbf{w}_n) p(\sigma_n^2) p(\sigma_v^2) d\mathbf{w}_n d\sigma_n^2 d\sigma_v^2. \tag{27}$$

It is important to note that $\mathbf{w}_n \in \mathbb{R}^{N\Lambda}$ is a very high-dimensional when considering a large graph or multiple delays scenario.

2) Selection of priors: Let us bear in mind that the values of the entries of the adjacency matrix represent the weights of the corresponding edges. Therefore, to enforce sparsity of the learned graph topology, it is reasonable to choose priors that assign high probabilities to the elements of the adjacency matrix that are close to zero.

Consider the sparse prior given in [34], written as:

$$p(\boldsymbol{\theta}) \propto \exp\left(- \sum_{m=1}^{N} \theta_m^{\gamma} \right), \tag{28}$$

where $\boldsymbol{\theta}$ is a vector of parameters that need to be estimated and where θ_m is the mth entry of $\boldsymbol{\theta}$, and the parameter $\gamma > 0$ is considered to control the sparsity level. When the parameter γ is equal to zero, we count the number of non-zero θ_m (i.e., we rely on the ℓ_0-norm). Let $\mathbf{w}_n$ be the argument instead of $\boldsymbol{\theta}$; then in our problem $\gamma = 0$ corresponds to counting the number of existing edges. However, we cannot have stable sampling from the posterior when $\gamma = 0$ because optimization problems involving ℓ_0-norms are often infeasible [35]. So to ensure the stability of the algorithm, we let $\gamma = 1$. Then the prior is given as

$$p(\mathbf{w}_n) = \exp\left(- \sum_{m=1}^{N} \sum_{\lambda=1}^{\Lambda} w_{nm}^{\lambda} \right). \tag{29}$$

The probability density function in (29) has a spike at zero, indicating that the probability is high when the input is close to zero, which is exactly what is desired by the sparsity requirement.

We have several choices for the priors of the scalar hyper-parameter σ_n^2 and the noise variance σ_v^2, such as

$$p(\sigma^2) \propto C, \tag{30}$$

$$p(\sigma^2) \propto \frac{1}{\sigma^2}, \tag{31}$$

$$p(\sigma^2) \propto \frac{1}{(\sigma^2)^{\alpha-1}} \exp\left(-\frac{\beta}{\sigma^2}\right), \tag{32}$$

$$p(\sigma^2) \propto \exp\left(-\frac{(\sigma^2-b)^2}{2a^2}\right), \tag{33}$$

where $\sigma^2 \in [0,+\infty)$ may represent either σ_n^2 or σ_v^2. Equation (30) is a uniform distribution with C as a constant, (31) is a non-informative prior often used for modeling noise variance, (32) is the inverse Gamma probability density function with parameters $\alpha > 0$ and $\beta > 0$, and the last equation is the truncated Gaussian distribution, with $a \in \mathbb{R}$ and $b > 0$. The performance of the method is usually not sensitive to the parameters of the adopted prior.

Still, one needs to be careful when choosing the priors because they can affect the performance of the sampling [36]. From the experiments we have done, the constant prior and the non-informative prior work well under a mild condition, say, low dimension and few data points. However, the half-normal prior and inverse gamma prior are stable under various sampling procedures, even in the 20-dimensional data set case, we used to test our algorithm, as described in the following.

V. Topology Inference Algorithm

Given the fBGP framework and the posterior of the marginal hyperparameters, we employ the HMC sampler to implement the sampling from the marginal likelihood. As the HMC exploits the gradient of the target distribution, it is necessary to derive the equation of the target distribution, i.e., the marginal hyperparameter posterior in (26), and its partial derivative with respect to each hyperparameter.

We start by considering the logarithm of the marginal likelihood, which for additive Gaussian noise $\mathbf{v}_n \sim \mathcal{N}(\mathbf{0}, \sigma_v^2\mathbf{I})$ is given by

$$\begin{aligned}&\log p(\mathbf{y}_n|\mathbf{X}, \mathbf{w}_n, \sigma_n^2, \sigma_v^2)\\ &= -\frac{1}{2}\mathbf{y}_n^\top \mathbf{K}^{-1}\mathbf{y}_n - \frac{1}{2}\log|\mathbf{K}| - \frac{T-\Lambda}{2}\log 2\pi,\end{aligned} \tag{34}$$

where $\mathbf{K} = \mathbf{K}_n + \sigma_v^2\mathbf{I}$.

By combining the marginal likelihood with the priors, we can express the log-marginal posterior distribution as,

$$\begin{aligned}\mathcal{L}_{\text{MP}} &= \log p(\mathbf{w}_n, \sigma_n^2, \sigma_v^2|\mathbf{y}_n, \mathbf{X})\\ &= -\frac{1}{2}\mathbf{y}_n^\top \mathbf{K}^{-1}\mathbf{y}_n - \frac{1}{2}\log|\mathbf{K}| - \frac{T-\Lambda}{2}\log 2\pi\\ &- \sum_{m=1}^{N}\sum_{\lambda=1}^{\Lambda} w_{nm}^\lambda + \log(p(\sigma_n^2)) + \log(p(\sigma_v^2)) - \log(\mathcal{Z}).\end{aligned} \tag{35}$$

To avoid the positive constraint, we worked with the logarithms of the hyperparameters. We represent them by adding tilde over their symbols. The partial derivative of the log-marginal posterior, with respect to each logarithm of the hyperparameters, is expressed by

$$\frac{\partial \mathcal{L}_{\text{MP}}}{\partial \widetilde{w}_{nm}^\lambda} = \frac{1}{2}\text{tr}\left((\boldsymbol{\alpha}\boldsymbol{\alpha}^\top - \mathbf{K}^{-1})\frac{\partial \mathbf{K}}{\partial \widetilde{w}_{nm}^\lambda}\right) + \frac{\partial \log p(\mathbf{w}_n)}{\partial \widetilde{w}_{nm}^\lambda}, \tag{36}$$

$$\vdots$$

$$\frac{\partial \mathcal{L}_{\text{MP}}}{\partial \widetilde{\sigma}_n} = \frac{1}{2}\text{tr}\left((\boldsymbol{\alpha}\boldsymbol{\alpha}^\top - \mathbf{K}^{-1})\frac{\partial \mathbf{K}}{\partial \widetilde{\sigma}_n}\right) + \frac{\partial \log p(\sigma_n^2)}{\partial \widetilde{\sigma}_n}, \tag{37}$$

$$\frac{\partial \mathcal{L}_{\text{MP}}}{\partial \widetilde{\sigma}_v} = \frac{1}{2}\text{tr}\left((\boldsymbol{\alpha}\boldsymbol{\alpha}^\top - \mathbf{K}^{-1})\frac{\partial \mathbf{K}}{\partial \widetilde{\sigma}_v}\right) + \frac{\partial \log p(\sigma_v^2)}{\partial \widetilde{\sigma}_v}, \tag{38}$$

where $\boldsymbol{\alpha} = \mathbf{K}^{-1}\mathbf{y}_n$, and

$$\left[\frac{\partial \mathbf{K}}{\partial \widetilde{w}_{nm}^\lambda}\right]_{i,j} = -\exp(2\widetilde{\sigma}_n) d_{i,j}^m k_{i,j}^n \exp(2\widetilde{w}_{nm}^\lambda), \tag{36a}$$

$$\left[\frac{\partial \mathbf{K}}{\partial \widetilde{\sigma}_n}\right]_{i,j} = 2\exp(2\widetilde{\sigma}_n) k_{i,j}^n, \tag{37a}$$

$$\left[\frac{\partial \mathbf{K}}{\partial \widetilde{\sigma}_v}\right]_{i,i} = 2\exp(2\widetilde{\sigma}_v), \tag{38a}$$

where $d_{i,j}^m = (y_{m,i+\lambda-1} - y_{m,j+\lambda-1})^2$.

We organize the partial gradient as a vector, i.e., $d\mathcal{L}_{\text{MP}} = [\frac{\partial \mathcal{L}_{\text{MP}}}{\partial \widetilde{w}_{nm}^\lambda}; \frac{\partial \mathcal{L}_{\text{MP}}}{\partial \widetilde{\sigma}_n}; \frac{\partial \mathcal{L}_{\text{MP}}}{\partial \widetilde{\sigma}_v}]$. Based on the marginal posterior and the gradient, we apply an HMC sampler. When the sampling is completed, we use the obtained sample set $\widetilde{\boldsymbol{\Phi}} = \{\widetilde{\mathbf{w}}_n^{(s)}, \widetilde{\sigma}_n^{(s)}, \widetilde{\sigma}_v^{(s)}\}_{s=1}^S$ to construct the marginal posterior distributions of the unknowns. From this posterior we can obtain desired estimates of the unknowns. For example, the minimum mean square estimate (MMSE) of σ_n^2 is obtained by

$$\widehat{\sigma}_n^2 = \frac{1}{S}\sum_{s=1}^{S}\exp(\widetilde{\sigma}_n^{(s)}). \tag{39}$$

The approximate marginal posterior of the hyperparameters of the length scales are given by

$$q^S(w_{nm}^\lambda|\mathbf{X}, \mathbf{y}_n) = \frac{1}{S}\sum_{s=1}^{S}\delta(w_{nm}^\lambda - \exp(\widetilde{w}_{nm}^{\lambda\,(s)})). \tag{40}$$

where $\delta(\cdot)$ is the Dirac delta function.

To find an effective binary topology, i.e., the nonweighted adjacency matrix, we set a threshold ϵ based on the following criterion:

$$a_{nm}^\lambda = \begin{cases} 0, & \text{if } w_{nm}^\lambda < \epsilon \\ 1, & \text{otherwise} \end{cases}. \tag{41}$$

We provide the steps of our procedure in Algorithm 1.

Algorithm 1 fBGP Topology Inference Algorithm

```
1:  Input: $\mathbf{Y}$, $\Lambda$, $\epsilon$, $S$
2:  for $n = 1, \ldots N$ (in parallel) do
3:      Initialize: $\mathbf{y}_n = \mathbf{Y}_{\Lambda+1:T,n}$, $\mathbf{X} = \mathbf{Y}_{1:T-\Lambda,:}$
            $\widetilde{\mathbf{w}}_n^{(0)} = \mathbf{1}$, $\widetilde{\sigma}_n^{(0)} = \widetilde{\sigma}_v^{(0)} = \log(var(\mathbf{y}_n))$
4:      for $s = 1, \ldots S$ do
5:          Calculate $\mathcal{L}_{\text{MP}}$ via Eq. (35)
6:          Calculate $d\mathcal{L}_{\text{MP}}$ via Eqs. (36), (37) and (38)
7:          Sample $\widetilde{\mathbf{\Phi}}_n^{(s)}$ given $\mathcal{L}_{\text{MP}}$ and $d\mathcal{L}_{\text{MP}}$
8:      end for
9:      Estimate the unknown parameters from $\widetilde{\mathbf{\Phi}}$ (as per (39))
10:     Find the weighted adjacency matrix by $[\mathbf{W}]_{n,:} = \mathbf{w}_n^*$
11: end for
12: Determine the binary adjacency matrix $\mathbf{A}$ via (41)
13: return $\mathbf{W}, \mathbf{A}$
```

In the section on numerical results, we modified the code from the gpml toolbox [37] to implement the fBGP model. We also implemented the HMC sampler using the Statistics and Machine Learning Toolbox in MATLAB [38]. The complexity of the algorithm depends on the complexity of inverting the kernel matrix and the size of the network. With N nodes and T data points, it is $\mathcal{O}(NT^3)$.

VI. Numerical Results

We evaluate the performance of the proposed method on three non-linear dynamic systems previously studied in our work [22]. The results of our previous study showed that the GP-based method outperformed the kernel-based method and LASSO, demonstrating the feasibility of detecting multi-lag causality. In this section, we show results of the proposed fBGP method and compare them with results obtained by the GP-based method (with deterministic parameters) on data generated by the same model as in [22]. We reiterate that both methods do not make any specific functional assumptions except that the functions are relatively smooth.

In the first experiment, we considered a small dynamical system [39], [40], while in the second one we worked with a larger Lorenz 96 model [41]. Finally, with the third experiment we demonstrate the applicability of the proposed method for estimating causality (connectivity) over multiple delays.

A. A Discretized Lorenz Attractor

We considered a three-node network associated with a discretized version of the Lorenz attractor [42], [43], described by

$$\begin{bmatrix} y_{1,t+1} \\ y_{2,t+1} \\ y_{3,t+1} \end{bmatrix} = \begin{bmatrix} y_{1,t} \\ y_{2,t} \\ y_{3,t} \end{bmatrix} + 0.01 \begin{bmatrix} 10(y_{2,t} - y_{1,t}) \\ y_{1,t}(28 - y_{3,t}) - y_{1,t} \\ y_{1,t}y_{2,t} - \frac{8}{3}y_{3,t} \end{bmatrix} + \begin{bmatrix} v_{1,t} \\ v_{2,t} \\ v_{3,t} \end{bmatrix}, \tag{42}$$

which is clearly characterized by a single (one-lag) memory (causality). The process of learning this topology has also been addressed in [39] using a kernel-based algorithm that was specifically designed to handle the nonlinear generative model of the data. Our previous work had already demonstrated, under the same condition as in [39] (i.e., with $v_{1,t}, v_{2,t}, v_{3,t}$ set to zero, and initial conditions $y_{1,0} = y_{2,0} = y_{3,0} = 0.01$), that the GP-based method in [22] (with $\Lambda = 1$, $N = 3$) outperforms the kernel-based method in terms of recovering the graph structure, using only $T = 90$ samples in comparison to the 250 samples required by [39].

Note that, we tested the proposed method on a noisy version of the Lorenz attractor to ensure that our approach is robust to observation noise. We compared the performance of the fBGP method with the GP-based method from [22] to demonstrate that the fBGP method can effectively enforce the network's sparsity by proper priors on the GPs length-scale parameters and by sampling from the marginal posteriors of

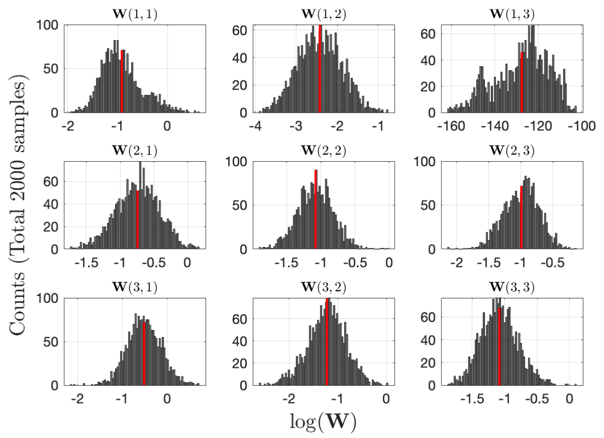

Fig. 3. Discretized Lorenz attractor. An example of a histogram of the samples obtained from the log-marginal posterior with noise variance $\sigma_v^2 = 0.5$. The subplots are arranged according to the index of the 3×3 adjacency matrix. For each subplot, the abscissa represents the logarithm of the samples' value, and the ordinate represents the count in each bin.

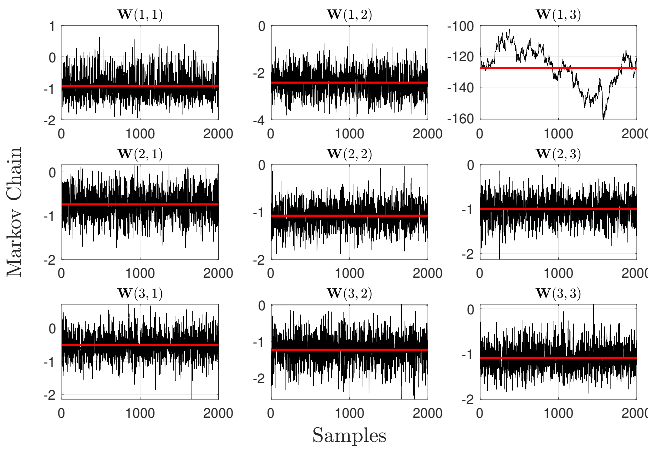

Fig. 4. Discretized Lorenz attractor. An example of Markov Chain samples where 2000 of them are generated with the HMC sampler with a noise variance $\sigma_v^2 = 0.5$. The subplots are arranged according to the index of the adjacency matrix. For each subplot, the abscissa represents the index of the samples, the ordinate represents the value of the samples, and the red lines are the MMSE estimates.

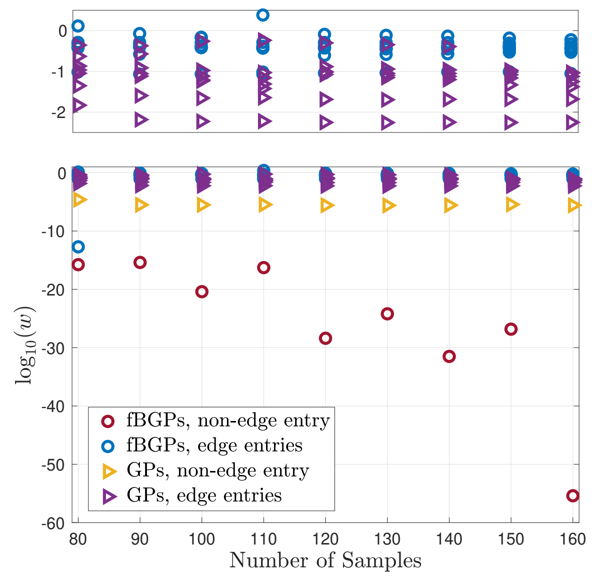

Fig. 5. One example of the elements of the inferred weighted adjacency matrix by the fBGP and the GP-based methods with noise variance $\sigma_v^2 = 0.5$. The upper small figure is the zoomed-in part of the lower figure. The circle marks represent values inferred by the fBGP method, with red indicating the non-edge entry and with blue indicating the edge entries. The triangle marks show the results estimated via the GP-based method, with yellow depicting the non-edge entry and with purple, the edge entries.

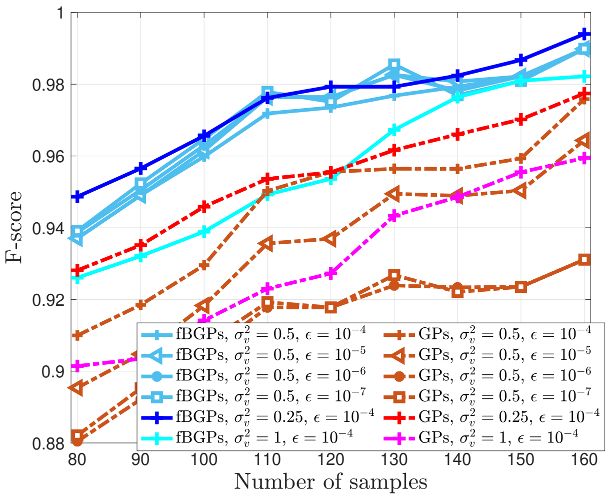

Fig. 6. Discretized Lorenz attractor. The F-score computed by averaging 50 independent runs is shown in the figure. The lines provide the F-scores obtained by four different thresholds by the fBGP method (in solid blue) and the GP-based method (in dash red). The logarithm of the threshold ranges from -7 to -4 with a step size of 1. Additionally, the lines show the F-score for different noise variances σ_v^2 of 0.25, 0.5, and 1, with the logarithmic threshold set to -4. The plot is shown for different time series lengths from 80 in steps of 10 samples to 160 samples.

the hyperparameters. Further, the simulation results show that the proposed method is not too sensitive to the selection of the thresholds in (41) that are used for determining causality in the network.

To generate synthetic data, we set σ_v^2 equal to 0.25, 0.5, and 1 and used the same initial value as in the previous work. The priors for σ_n^2 and σ_v^2 were chosen as the half-normal distribution in (33) with $a = 0$ and $b = 1$. We note that, for benchmarking performance, the weighted adjacency matrix $\mathbf{W}$ provides more informative results than the binary connectivity matrix $\mathbf{A}$. Therefore, we plot the estimated marginal posterior of the weights $\mathbf{w}$ and the associated Markov chain in Figs. 3 and 4, respectively, employing $T = 160$ samples.

In Fig. 5, we give an intuitive comparison of the fBGP method with the GP-based method from [22], with different lengths of the time series (starting with a series that is $T = 80$ samples long and increasing their sizes to 160 samples in increments of 10 samples). For the proposed method, the red circle marks the only entry that quantifies as a missing edge, while the blue circle marks depict the entries of the adjacency matrix that correspond to existing edges. For the GP-based method, the purple triangle marks represent edges, and the yellow represent the only non-edge. The results show that the entries associated with edges estimated by the two approaches are very similar. However, there is a considerable gap between the value of the non-edge entry learned by the two methods, with the proposed method compressing the non-edge value to zero and thereby demonstrating its ability to model sparsity effectively.

When studying the binary adjacency matrix, we first determined the effective graph via (41). To provide a quantified comparison, we assessed performance by the F-score, which is expressed by

$$\text{F-score} = \frac{\text{TP}}{\text{TP} + \frac{1}{2}(\text{FN} + \text{FP})}, \tag{43}$$

where TP stands for the number of true positives, FP for the false positives, and FN for the false negatives. Figure 6 shows the F-score for the two approaches, with the log-threshold $\log_{10}(\epsilon)$ increasing from -7 to -4. The results show that, different from the GP-based method, the threshold value does not significantly affect the estimated topology of the new fBGP method, thus, demonstrating its robustness to threshold selection for edge detection.

B. Lorenz 96

The Lorenz 96 model is defined as

$$\frac{dy_n}{dt} = (y_{n+1} - y_{n-2})y_{n-1} - y_n + F, \quad n = 1, 2, \ldots, N. \tag{44}$$

Lorenz originally proposed the system to investigate fundamental issues in forecasting spatially extended chaotic systems such as the atmosphere [41]. We use this system to illustrate the proposed method's ability to recover the topology of a high-dimensional graph. Specifically, we generated a directed graph with $N = 20$ nodes using (44). We obtained the synthetic signals of this graph through Runge-Kutta numerical integration with a step size of 0.1, initializing all the nodes with $\mathbf{y}_0 = F\mathbf{1} = 8\mathbf{1}$ and then adding a value of 0.01 to the initial value of node 1, with an observation noise $\sigma_v^2 = 0.5$. In this case, one lag refers to one sample rather than one second, and we consider a one-delay system, i.e., $\Lambda = 1$. The priors of σ_v^2 and σ_n^2 are given by (32) and (33) with $a = 0$ and $b = 1$, and $\alpha = 3$ and $\beta = 0.5$, respectively.

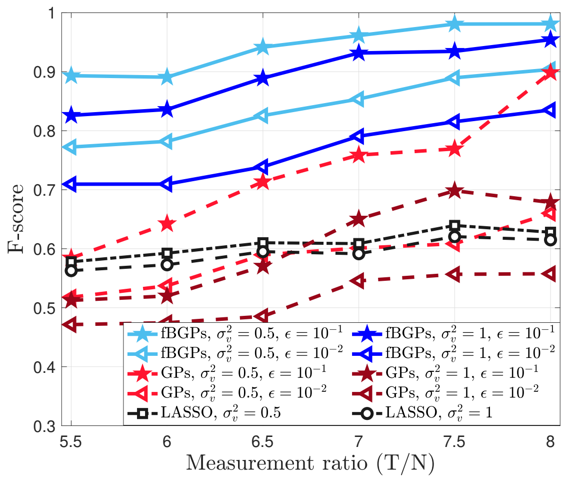

Fig. 7. The Lorenz 96 F-score was computed by averaging 50 independent runs. Lines provide the F-scores obtained by two different thresholds by the fBGP method (in solid blue), the GP-based method (in dash red), and the graph LASSO (in dot black). Additionally, the lines show the F-score for different noise variances σ_v^2 of 0.5 and 1, with the logarithmic threshold set to -1 and -2, respectively. The plot is shown for different measurement ratios (T/N) starting at 5.5 to 8 with a step size of 0.5.

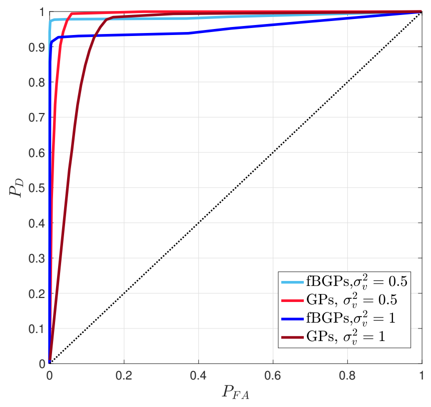

Fig. 8. Lorenz 96. Comparison of ROC curves for the fBGP method with HMC sampling and the GP-based method with ML-II.

We compared the performance of three methods, namely the fBGP method, the GP-based method, and LASSO [44] by measuring their F-score defined by (43). The F-score was computed by averaging 50 independent trials, where each trial was independently run for different values of measurement ratios (T/N). The results, presented in Fig. 7, show that the fBGP method outperformed the other two approaches for two different thresholds. For LASSO, the regularization parameter was chosen to be the one with the best performance. We did not display the results based on other thresholds because those thresholds almost denied or accepted all the edges.

The F-score in this test shows some stronger dependence on the threshold ϵ. Actually, to complete the edge-detection performance of the proposed test, we plot the receiver operating characteristic (ROC) for $\epsilon \in [0, +\infty)$, as shown in Fig. 8. The symbol P_D is the correct detection probability of an existing edge, and P_{FA} is the false alarm probability (for existence

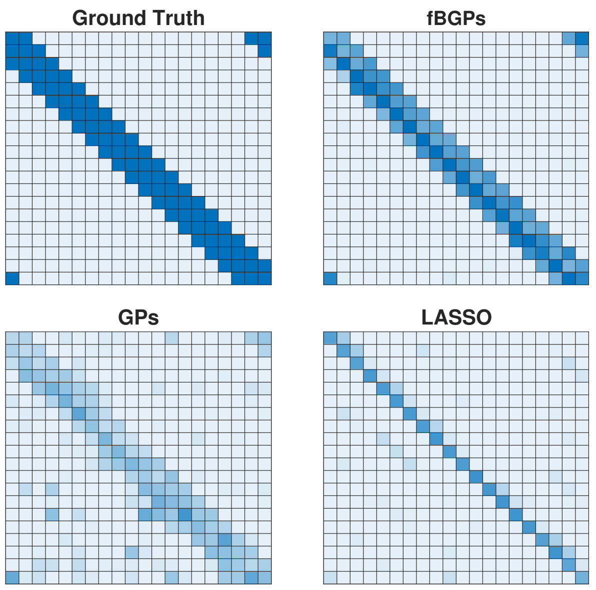

Fig. 9. Lorenz 96. A heatmap is presented with a measurement ratio of 8. The first row displays the weighted topology heatmap of the ground truth (left) and the fBGP method (right). The second row showcases the weighted topology heatmap of the GP-based method (left) and the LASSO approach (right). The darker shades of blue indicate stronger edges, and the color scale ranges from 0 to 1.

of an edge). Each point in the figure represents one pair of (P_{FA}, P_D), associated to a specific threshold ϵ. The area under the curve (AUC) is a criterion for detection performance with a higher value indicating better performance. The AUCs suggest that the fBGP method improves the detection performance with respect to the GP-based approach.

To provide an intuitive sense of the edge detection performance, Fig. 9 represents the heatmap of the ground truth and the simulation results of the fBGP, GP-based, and LASSO methods, respectively. The results show that the fBGP method can detect most of the actual edges with fewer errors than the other two methods.

C. Multi-lag system

In the third experiment, we designed a small network to evaluate the detection capabilities of the proposed approach in a multiple-delay problem. Each node in the small graph interacted with a different delay time. The data model is shown in Fig. 10, which graphically represents the dynamical system described by the following set of equations:

$$\begin{aligned} y_{1,t} &= y_{4,t-3}^2 + y_{4,t-3} + v_{1,t}, \\ y_{2,t} &= \sin(y_{4,t-3}) + 1 + v_{2,t}, \\ y_{3,t} &= y_{4,t-2}y_{2,t-2} + y_{4,t-2}y_{1,t-2} + y_{3,t-2}y_{1,t-2} - 3 + v_{3,t}, \\ y_{4,t} &= \sin(y_{3,t-1})/y_{3,t-1} + v_{4,t}. \end{aligned} \tag{45}$$

The initial values of y_1, y_2, y_3, and y_4 were independently drawn from a Gaussian distribution with zero mean and variance of 0.5^2, i.e., $y_{1,0}, y_{2,0}, y_{3,0}, y_{4,0} \sim \mathcal{N}(0, 0.5^2)$. The noise was drawn from a Gaussian distribution with zero mean and a standard deviation of $\sigma_v^2 = 0.25$ and $\sigma_v^2 = 0.5$,

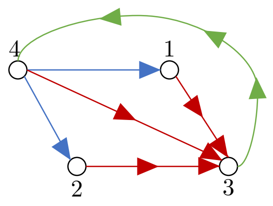

Fig. 10. Data model description: the arrow directions reflect the direction of causation. The different colors represent different delays, $\lambda = 3$ (blue), $\lambda = 2$ (red), and $\lambda = 1$ (green).

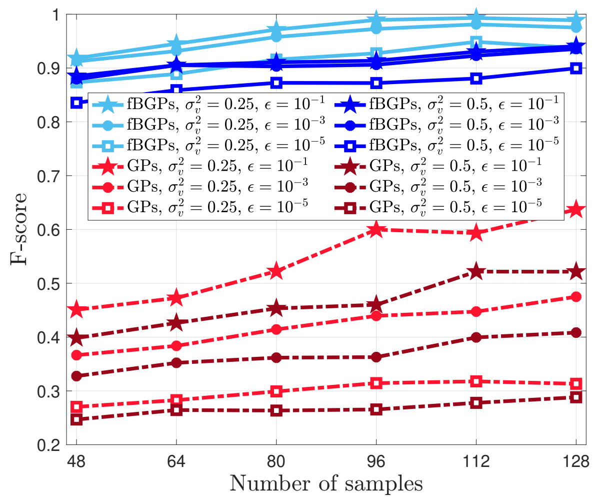

Fig. 11. Multiple delays. The F-score computed by averaging 50 independent runs. The lines provide the F-scores obtained by three different thresholds by the fBGP method (in solid blue) and the GPs method (in dash red). The logarithm of the threshold ranges from -1 to -5 with a step size of 2. The plot presents different lengths of time series, ranging from 48 to 128 samples, in steps of 12.

respectively. The priors were the same as the example in the previous subsection.

For the proposed method and the GP-based method, Fig. 11 shows the F-score against the length of the time series. The results were computed by averaging 50 independent trials. They clearly show that the fBGP method can identify the multiple delays better than the GP-based method under different thresholds. Figure 12 displays the ROC curve for the two approaches. Even though when the false alarm P_{FA} is larger than 0.2, the detection probability P_D of the GPs method is greater than the fBGP method, based on the AUC criterion, the fBGP method significantly improves the detection performance.

It is important to note that certain parameters of the HMC sampler must be selected to implement the method including the update size and number of leapfrog steps. Throughout the experiments shown in this work, we consistently set the step size to 0.05 and the number of leapfrog steps to 50. Further, we note that these parameters have a direct impact on the sampling performance. This is often addressed by employing the well-known No U-Turn Sampler [45], which is an extended version of the HMC sampler that controls the number of steps automatically. This method offers a potential solution to alleviate the need of setting these parameters manually.

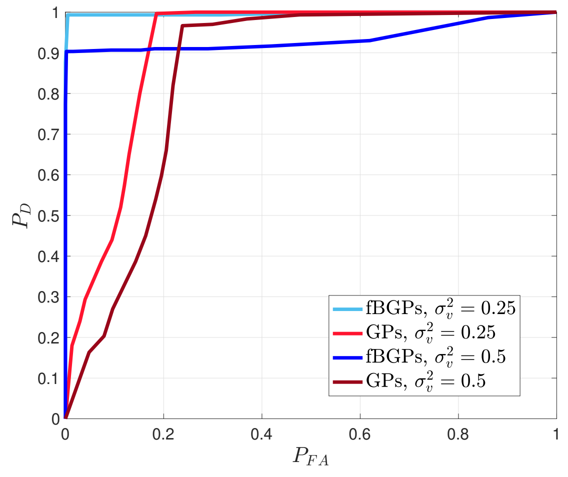

Fig. 12. Multiple delays. Comparison of ROC curves of the fBGP and GP-based methods.

VII. Conclusion

In this paper, we proposed a fully Bayesian Gaussian process-based method for estimating the topology of a graph from observed signals on the graph. The proposed method models the entry of the weighted adjacency matrix as the coefficient of the modified ARD kernel. Our model assumptions are mild, and we do not specify the functional forms of the relationships among the graph signals, allowing our method to detect causation among the signals on the graph. We showed that the performance of the proposed method is very good under different conditions, including non-linear dynamics, large systems, and even scenarios with multiple delays. Our proposed method uses priors of the hyperparameters that promote sparsity and samples from the marginal posterior using the HMC sampler. The experimental results suggest that that method avoids overfitting.

Future directions of work include the following:

1) Extension of the proposed method to dynamic networks where the topology of the network varies with time. One promising direction involves developing algorithms that leverage nonstationary Gaussian processes to handle dynamic networks with time-varying topology.
2) Investigation of the scalability of the method. Note that when the system delays Λ and the network size N increase, the number of parameters that need to be estimated is $N^2\Lambda$. If N and Λ are large, this becomes problematic. The ongoing work focuses on employing feature-based Gaussian processes that reduce the dimensionality issues.
3) Automatic threshold selection. Since the proposed method provides the marginal posterior of the parameters, one possibility will be exploiting a binary hypothesis test to determine the presence or absence of the edges automatically.

References

[1] S. Lebre, J. Becq, F. Devaux, M. P. Stumpf, and G. Lelandais, "Statistical inference of the time-varying structure of gene-regulation networks,"

BMC systems biology, vol. 4, no. 1, pp. 1–16, 2010.

[2] S. Myers and J. Leskovec, "On the convexity of latent social network inference," *Advances in Neural Information Processing systems*, vol. 23, 2010.

[3] M. Iloska, Y. El-Laham, and M. F. Bugallo, "Graphical network and topology estimation for autoregressive models using Gibbs sampling," *Signal Processing*, vol. 190, p. 108303, 2022.

[4] A. Nagurney and K. Ke, "Financial networks with intermediation," *Quantitative Finance*, vol. 1, no. 4, p. 441, 2001.

[5] X. Dong, D. Thanou, P. Frossard, and P. Vandergheynst, "Learning laplacian matrix in smooth graph signal representations," *IEEE Transactions on Signal Processing*, vol. 64, no. 23, pp. 6160–6173, 2016.

[6] H. E. Egilmez, E. Pavez, and A. Ortega, "Graph learning from data under Laplacian and structural constraints," *IEEE Journal of Selected Topics in Signal Processing*, vol. 11, no. 6, pp. 825–841, 2017.

[7] G. Chen, D. R. Glen, Z. S. Saad, J. P. Hamilton, M. E. Thomason, I. H. Gotlib, and R. W. Cox, "Vector autoregression, structural equation modeling, and their synthesis in neuroimaging data analysis," *Computers in Biology and Medicine*, vol. 41, no. 12, pp. 1142–1155, 2011.

[8] G. Chen, J. P. Hamilton, M. E. Thomason, I. H. Gotlib, Z. S. Saad, and R. W. Cox, "Granger causality via vector auto-regression tuned for fMRI data analysis," in *Proc Intl Soc Mag Reson Med*, vol. 17, 2009, p. 1718.

[9] G. Michailidis and F. d'Alché Buc, "Autoregressive models for gene regulatory network inference: Sparsity, stability and causality issues," *Mathematical Biosciences*, vol. 246, no. 2, pp. 326–334, 2013.

[10] R. Money, J. Krishnan, and B. Beferull-Lozano, "Online non-linear topology identification from graph-connected time series," in *2021 IEEE Data Science and Learning Workshop (DSLW)*, 2021, pp. 1–6.

[11] M. A. Vosoughi and A. Wismüller, "Large-scale kernelized granger causality (lskgc) for inferring topology of directed graphs in brain networks," in *Medical Imaging 2022: Biomedical Applications in Molecular, Structural, and Functional Imaging*, vol. 12036. SPIE, 2022, p. 1203603.

[12] B. Liu, A. de La Fuente, and I. Hoeschele, "Gene network inference via structural equation modeling in genetical genomics experiments," *Genetics*, vol. 178, no. 3, pp. 1763–1776, 2008.

[13] X. Cai, J. A. Bazerque, and G. B. Giannakis, "Inference of gene regulatory networks with sparse structural equation models exploiting genetic perturbations," *PLoS Computational Biology*, vol. 9, no. 5, p. e1003068, 2013.

[14] J. Pearl, *Causality*. Cambridge university press, 2009.

[15] J. Peters, D. Janzing, and B. Schölkopf, *Elements of Causal Inference: Foundations and Learning Algorithms*, 2017.

[16] V. N. Ioannidis, Y. Shen, and G. B. Giannakis, "Semi-blind inference of topologies and dynamical processes over dynamic graphs," *IEEE Transactions on Signal Processing*, vol. 67, no. 9, pp. 2263–2274, 2019.

[17] Y. Shen, G. B. Giannakis, and B. Baingana, "Nonlinear structural vector autoregressive models with application to directed brain networks," *IEEE Transactions on Signal Processing*, vol. 67, no. 20, pp. 5325–5339, 2019.

[18] R. Shafipour, S. Segarra, A. G. Marques, and G. Mateos, "Directed network topology inference via graph filter identification," in *2018 IEEE Data Science Workshop (DSW)*. IEEE, 2018, pp. 210–214.

[19] J. Mei and J. M. Moura, "Signal processing on graphs: Causal modeling of unstructured data," *IEEE Transactions on Signal Processing*, vol. 65, no. 8, pp. 2077–2092, 2016.

[20] G. Mateos, S. Segarra, A. G. Marques, and A. Ribeiro, "Connecting the dots: Identifying network structure via graph signal processing," *IEEE Signal Processing Magazine*, vol. 36, no. 3, pp. 16–43, 2019.

[21] A. Bolstad, B. D. Van Veen, and R. Nowak, "Causal network inference via group sparse regularization," *IEEE Transactions on Signal Processing*, vol. 59, no. 6, pp. 2628–2641, 2011.

[22] C. Cui, P. Banelli, and P. M. Djurić, "Gaussian processes for topology inference of directed graphs," in *2022 30th European Signal Processing Conference (EUSIPCO)*. IEEE, 2022, pp. 2156–2160.

[23] C. E. Rasmussen, "Gaussian processes in machine learning," in *Summer school on machine learning*. Springer, 2003, pp. 63–71.

[24] C. M. Bishop and N. M. Nasrabadi, *Pattern Recognition and Machine Learning*. Springer, 2006, vol. 4, no. 4.

[25] C. Gold, A. Holub, and P. Sollich, "Bayesian approach to feature selection and parameter tuning for support vector machine classifiers," *Neural Networks*, vol. 18, no. 5-6, pp. 693–701, 2005.

[26] T. Wang, H. Huang, S. Tian, and J. Xu, "Feature selection for SVM via optimization of kernel polarization with gaussian ard kernels," *Expert Systems with Applications*, vol. 37, no. 9, pp. 6663–6668, 2010.

[27] G. Feng, J. G. Quirk, and P. M. Djurić, "Inference about causality from cardiotocography signals using Gaussian processes," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 2852–2856.

[28] Y. Saatçi, R. D. Turner, and C. E. Rasmussen, "Gaussian process change point models," in *ICML*, 2010.

[29] Y. Liu and P. M. Djurić, "Gaussian process state-space models with time-varying parameters and inducing points," in *2020 28th European Signal Processing Conference (EUSIPCO)*. IEEE, 2021, pp. 1462–1466.

[30] P. M. Djurić and C. Richard, *Cooperative and Graph Signal Processing: Principles and Applications*. Academic Press, 2018.

[31] M. Nishio and A. Arakawa, "Performance of Hamiltonian Monte Carlo and no-u-turn sampler for estimating genetic parameters and breeding values," *Genetics Selection Evolution*, vol. 51, no. 1, pp. 1–12, 2019.

[32] M. Betancourt, "A conceptual introduction to hamiltonian monte carlo," *arXiv preprint arXiv:1701.02434*, 2017.

[33] V. Lalchand and C. E. Rasmussen, "Approximate inference for fully Bayesian Gaussian process regression," in *Symposium on Advances in Approximate Bayesian Inference*. PMLR, 2020, pp. 1–12.

[34] D. P. Wipf and B. D. Rao, "Sparse Bayesian learning for basis selection," *IEEE Transactions on Signal Processing*, vol. 52, no. 8, pp. 2153–2164, 2004.

[35] H. Dong, K. Chen, and J. Linderoth, "Regularization vs. relaxation: A conic optimization perspective of statistical variable selection," *arXiv preprint arXiv:1510.06083*, 2015.

[36] M. Betancourt, "How the shape of a weakly informative prior affects inferences," *Stan User's Guide. March*, vol. 17, 2017.

[37] C. E. Rasmussen and H. Nickisch, "Gaussian processes for machine learning (gpml) toolbox," *The Journal of Machine Learning Research*, vol. 11, pp. 3011–3015, 2010.

[38] M. MathWorks *et al.*, "Statistics and machine learning toolbox user's guide," 2017.

[39] M. Moscu, R. Borsoi, and C. Richard, "Online graph topology inference with kernels for brain connectivity estimation," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 1200–1204.

[40] G. Sugihara, R. May, H. Ye, C.-h. Hsieh, E. Deyle, M. Fogarty, and S. Munch, "Detecting causality in complex ecosystems," *Science*, vol. 338, no. 6106, pp. 496–500, 2012.

[41] A. Karimi and M. R. Paul, "Extensive chaos in the Lorenz-96 model," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 20, no. 4, p. 043105, 2010.

[42] N. Biggs, N. L. Biggs, and B. Norman, *Algebraic Graph Theory*. Cambridge University Press, 1993, no. 67.

[43] M. A. Kramer, E. D. Kolaczyk, and H. E. Kirsch, "Emergent network topology at seizure onset in humans," *Epilepsy Research*, vol. 79, no. 2-3, pp. 173–186, 2008.

[44] J. Ranstam and J. Cook, "Lasso regression," *Journal of British Surgery*, vol. 105, no. 10, pp. 1348–1348, 2018.

[45] M. D. Hoffman, A. Gelman *et al.*, "The no-u-turn sampler: adaptively setting path lengths in Hamiltonian Monte Carlo." *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1593–1623, 2014.