# Online Machine Learning for Inference from Multivariate Time-series

Rohan T. Money

UiA University of Agder

Doctoral Dissertations at
the University of Agder 429

# Online Machine Learning for Inference from Multivariate Time-series

Rohan T. Money

# Online Machine Learning for Inference from Multivariate Time-series

Doctoral Dissertation for the Degree *Philosophiae Doctor (Ph.D.)*

University of Agder
Faculty of Engineering and Science
2023

Doctoral Dissertations at the University of Agder 429
ISSN: 1504-9272
ISBN: 978-82-8427-146-0

Printed by Make!Graphics
Kristiansand

# Preface and Acknowledgments

I am delighted to present this Ph.D. thesis, which reflects the culmination of my research work carried out at the Center Intelligent Signal Processing and Wireless Networks (WISENET), Department of Information and Communication Technology (ICT), University of Agder, Grimstad, Norway, under the guidance of Prof. Baltasar Beferull-Lozano. It would have been impossible for me to complete this dissertation without the support of many people to whom I am deeply indebted.

In the first place, I would like to express my profound gratitude to Prof. Baltasar Beferull-Lozano, my Ph.D. supervisor. Over the past three years, he has demonstrated immense dedication and commitment to his supervisory duties. I would like to thank him for sharing his wonderful ideas and immense knowledge with me. Next, I would like to thank my co-supervisor, Dr. Joshin P. Krishnan. It would not have been possible for this dissertation to be completed without his constant support and guidance. I would like to thank also Prof. Elvin Isufi (T.U. Delft) for the insights and guidance provided in the latter half of my Ph.D.

All members of the WISENET Center deserve a big thank you for their wonderful camaraderie. My sincere gratitude goes out also to Julia for her administrative assistance during all my period at WISENET. I am thankful to Emma, Kristine, Lief, and all other administrative staff members who supported me throughout my Ph.D. journey.

I would like to thank my friends Cheecku, Diego, Emilio , Hareesh , Juan, Kevin, Mohamed, Preeti, Rahul, Ravi, Sarang, Surendar for making my stay in Grimsatd memorable. I also thank the UiA football group for all the wonderful matches. Additionally, I would like to thank Aleena, Arun, Dasppan, Hashim, Jacob, Karthik, Kumo, Kunju, Lakshmi, Regish, Silpa for staying in touch with me throughout the last three years.

My sincere thanks go out to my family for their unwavering love and trust during my Ph.D. journey. My mother's constant support and encouragement kept me motivated, while my father's inspiring words helped me stay focused on my goals. I am also grateful for the love and encouragement of my siblings, who stood by me through thick and thin. I also want to express my gratitude to Appuchettan for his continuous calls and unwavering support during my Ph.D. program.

Rohan T. Money
Oslo
02/05/2023

# Sammendrag

Inferens og dataanalyse over nettverk har blitt viktige forskningsområder på grunn av den økende utbredelsen av sammenkoblede systemer og det økende volumet av data de produserer. Mange av disse systemene genererer data i form av multivariat tidsserier som er samlinger av tidsseriedata som observeres samtidig på tvers av flere variabler. For eksempel, EEG-målinger av hjernen gir multivariat tidsseriedata som registrerer den elektriske aktiviteten i ulike hjerneområder over tid. Cyberfysiske systemer genererer multivariate tidsserier som fanger opp oppførselen til fysiske systemer som respons på kybernetisk input. Tilsvarende gjenspeiler finansielle tidsserier dynamikken i flere finansielle instrumenter eller markedsindekser over tid. Ved å analysere disse tidsseriene kan man avdekke viktige detaljer om systemets oppførsel, oppdage mønstre og komme med forutsigelser. Derfor er det viktig å utvikle effektive metoder for dataanalyse og inferens i nettverk av tidsserier med flere variabler. Dette er et viktig forskningsområde med mange bruksområder på ulike felt. I denne doktorgradsavhandlingen fokuserer vi på å identifisere de rettede relasjonene mellom tidsserier og å utnytte denne informasjonen til å designe algoritmer for prediksjon av data og imputering av manglende data.

# Abstract

Inference and data analysis over networks have become significant areas of research due to the increasing prevalence of interconnected systems and the growing volume of data they produce. Many of these systems generate data in the form of multivariate time series, which are collections of time series data that are observed simultaneously across multiple variables. For example, EEG measurements of the brain produce multivariate time series data that record the electrical activity of different brain regions over time. Cyber-physical systems generate multivariate time series that capture the behaviour of physical systems in response to cybernetic inputs. Similarly, financial time series reflect the dynamics of multiple financial instruments or market indices over time.

Through the analysis of these time series, one can uncover important details about the behavior of the system, detect patterns, and make predictions. Therefore, designing effective methods for data analysis and inference over networks of multivariate time series is a crucial area of research with numerous applications across various fields. In this Ph.D. Thesis, our focus is on identifying the directed relationships between time series and leveraging this information to design algorithms for data prediction as well as missing data imputation.

This Ph.D. thesis is organized as a compendium of papers, which consists of seven chapters and appendices. The first chapter is dedicated to motivation and literature survey, whereas in the second chapter, we present the fundamental concepts that readers should understand to grasp the material presented in the dissertation with ease. In the third chapter, we present three online nonlinear topology identification algorithms, namely NL-TISO, RFNL-TISO, and RFNL-TIRSO. In this chapter, we assume the data is generated from a sparse nonlinear vector autoregressive model (VAR), and propose online data-driven solutions for identifying nonlinear VAR topology. We also provide convergence guarantees in terms of dynamic regret for the proposed algorithm RFNL-TIRSO. Chapters four and five of the dissertation delve into the issue of missing data and explore how the learned topology can be leveraged to address this challenge. Chapter five is distinct from other chapters in its exclusive focus on edge flow data and introduces an online imputation strategy based on a simplicial complex framework that leverages the known network structure in addition to the learned topology. Chapter six of the dissertation takes a different approach, assuming that the data is generated from nonlinear structural equation models. In this chapter, we propose an online topology identification algorithm using a time-structured approach, incorporating information from both the data and the model evolution. The algorithm is shown to have convergence guarantees achieved by bounding the dynamic regret. Finally, chapter seven of the dissertation provides concluding remarks and outlines potential future research directions.

# Publications

The following papers are included in this dissertation and are appended in Appendices, A-F at the end of the Ph.D. Thesis.

- PAPER A **R. Money**, J. Krishnan and B. Beferull-Lozano, "Online Nonlinear Topology Identification from Graph-connected Time Series," 2021 IEEE Data Science and Learning Workshop, 2021, pp. 1-6.

- PAPER B **R. Money**, J. Krishnan and B. Beferull-Lozano, "Random Feature Approximation for Online Nonlinear Graph Topology Identification," 2021 IEEE 31st International Workshop on Machine Learning for Signal Processing (MLSP), 2021, pp. 1-6.

- PAPER C **R. Money**; J. Krishnan; B. Beferull-Lozano (2023): Sparse Online Learning with Kernels using Random Features for Estimating Nonlinear Dynamic Graphs. Preprint. https://doi.org/10.36227/techrxiv.19210092.v3 (IEEE Transactions on Signal Processing, accepted)

- PAPER D **R. Money**, J. Krishnan and B. Beferull-Lozano, "Online Joint Nonlinear Topology Identification and Missing Data Imputation over Dynamic Graphs," 2022 30th European Signal Processing Conference (EUSIPCO), 2022, pp. 687-691.

- PAPER E **R. Money**; J. Krishnan; B. Beferull-Lozano; E. Isufi (2022): Online Edge Flow Imputation on Networks. IEEE Signal Processing Letters.

- PAPER F **R. Money**; J. Krishnan; B. Beferull-Lozano; E. Isufi (2023): Scalable and Privacy-aware Online Learning of Nonlinear Structural Equation Models. IEEE Open Journal of Signal Processing.

# Contents

# List of Figures

# List of Tables

# Abbreviations

| | |
|---|---|
| **COMID:** | Composite Objective Mirror Descent |
| **KF:** | Kalman Filter |
| **NL-TISO:** | Nonlinear Topology Identification by Sparse Online learning |
| **O&G:** | Oil and gas |
| **RF:** | Random Features |
| **RFNL-TIRSO:** | Random Feature approximation Nonlinear Topology Identification by Recursive Sparse Online learning |
| **RFNL-TISO:** | Random Feature approximation Nonlinear Topology Identification by Sparse Online learning |
| **RKHS:** | Reproducing Kernel Hilbert Space |
| **RL:** | Reinforcement learning |
| **SC:** | Simplicial Complex |
| **SEM:** | Structural Equation Models |
| **SVAR:** | Structural Vector Autoregressive |
| **VAR:** | Vector Autoregressive |

# Chapter 1

# Introduction

## 1.1 Motivation

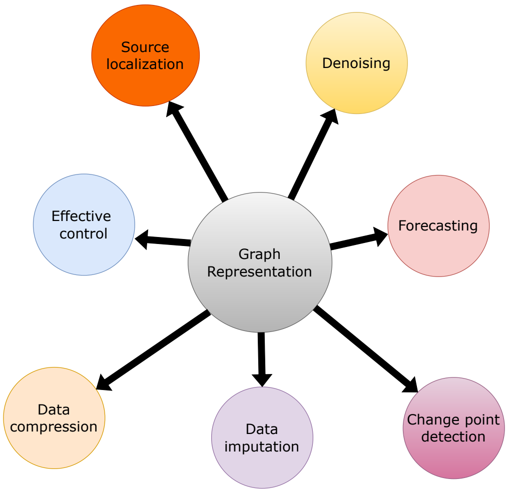


Figure 1.1: Application of graph representation.

Multivariate time series analysis has paramount importance in network science, as they are ubiquitous in real-world networks such as water distribution networks, social networks, transportation networks, etc. Data generated in the form of multivariate time series are mostly interdependent. It is possible to represent these networks in the form of a graph; in such a representation, each time series represents a node, and the relations between time series are expressed as edges connecting nodes. The graph structure of the network can be utilized for a wide variety of tasks. In financial systems, for example, predicting the future values of time series data like stock prices or exchange rates is crucial. By constructing a graph where nodes represent stocks and edges denote the relationships between their prices, graph-based algorithms can be leveraged to make accurate predictions based on historical data. In sensor networks, noisy signals are a common problem. By modelling the network as a graph, relationships between different sensors can be utilized to improve signal quality. Similarly, in social networks, identifying groups of users who share common interests or characteristics can be achieved by constructing a graph where nodes represent users and edges represent social connections between them. Community detection algorithms can then be applied to identify clusters of users with close connections and shared characteristics. Graph structures can also be employed in designing control strategies for complex systems like power grids or transportation networks. By modelling these systems as graphs, graph-based algorithms can be used to optimize control strategies. For instance, a power grid can be modeled as a graph with nodes representing power plants or substations and edges representing transmission lines. By employing graph-based optimization algorithms, the optimal control strategy can be determined, minimizing energy losses and ensuring stable operation. In most cases, the graph structure might not be physically observable and identifying the graph structure in the network itself becomes a challenging task. As there are countless practical applications for a graph representation of networks (see Fig. 1.1), learning the graph structure from multivariate data has gained significant research interest.

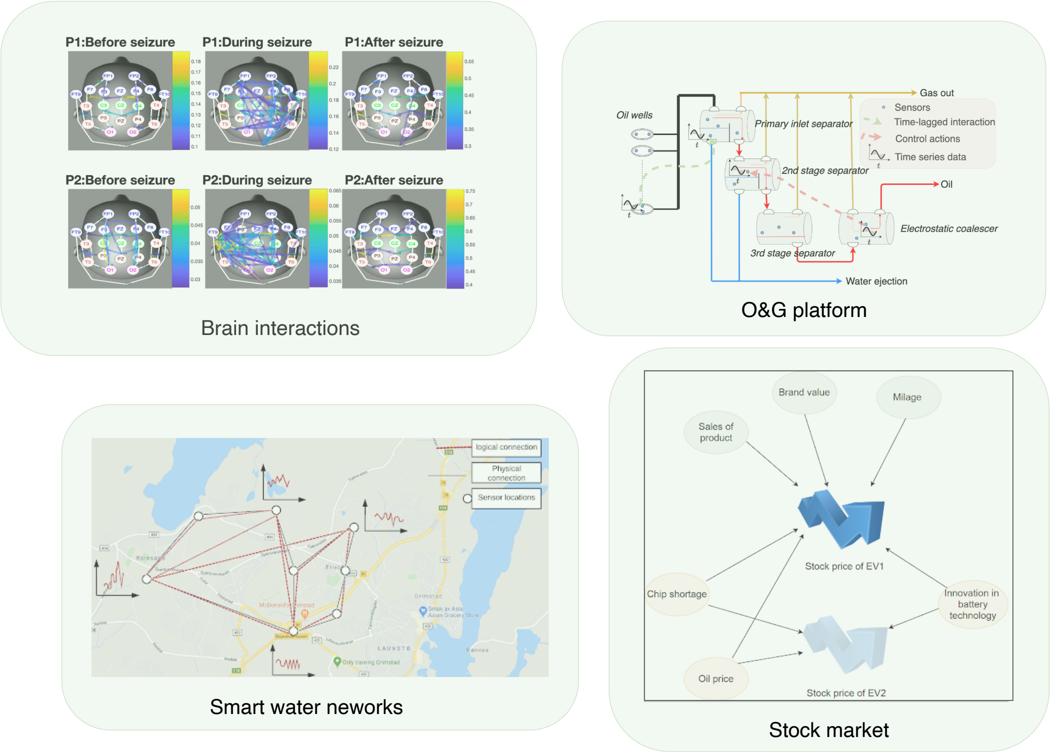


Figure 1.2: Systems with complex nonlinear interactions.

The majority of existing literature on graph learning considers an undirected or symmetric relationship between nodes. Such an approach is not always the best course of action since the interactions between nodes of real-world interconnected physical systems are often nonsymmetrical and bidirectional. As an example, assume that we want to categorize users as leaders and followers on a social network; the directionality of the graph is essential. Similarly, when the data under consideration is flow, such as power transfer in an electric grid, traffic flow in transportation network, flow in a water network etc., the direction is inherently associated with the data. The directional graphs can represent the system's hidden underlying structures or topology, for example, in neuroscience, social and sensor networks, etc. Moreover, the edges in a directional graph representing complex real-world dynamic systems can be seen as an abstract representation of causal relationships corresponding to the time-lagged interactions within the system. In this thesis, we will interchangeably use the terms causality, interactions, and dependency to represent directional relationships and topology within graph structures.

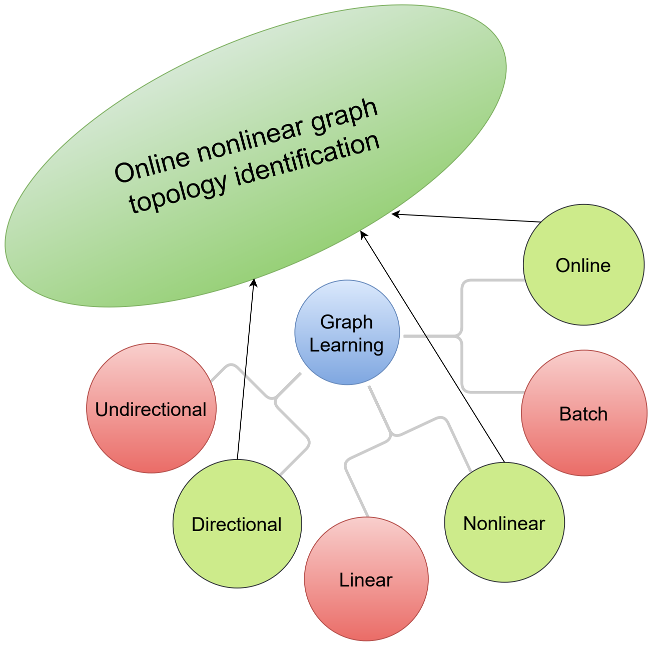


Figure 1.3: In this dissertation, we present algorithms that can learn nonlinear directional dependencies online.

Identifying a graph structure or topology is not straightforward since it is often not directly perceivable. Consider the case of oil and gas plants. The system consists of numerous actuators and hundreds of sensors. Each of the sensors generates time series data which are possibly dependent on each other. The dependencies are due to various physical equations governing the dynamics of the system and control actions. Solving the complex differential equations for such a complex system is a daunting task, and data-driven approaches are gaining popularity, which facilitates the analysis of multiple parameters of the system simultaneously to unveil the underlying physical laws. Even if an oil and gas (O&G) plant is mentioned as an example, such complex relationships are present in many other scenarios, such as brain networks, finance networks, various cyber-physical systems, etc. (please see Fig. 1.2).

When the underlying structure is unknown, and relationships can only be deduced from observed data, the topology identification task can already be challenging, and this is further compounded by the nonlinear nature of physical equations that govern real-world systems. In many cases, a linear approximation of the nonlinear system fails to express the fundamental nature of the system. As a result, it is vital to take into account the nonlinearity when the topology identification is performed. Apart from that, the dependencies between the multiple time series can also vary across time. For instance, consider the case of social media networks based on user activity, where the network structure varies based on various sociopolitical events. During the time of the election, individuals with similar political inclinations tend to create a stronger connection, or during a football match, sports enthusiasts generate a stronger connection. Motivated by the nonlinearity, non-stationarity, and directionality exhibited by real-world networks, the Ph.D. thesis proposes novel algorithms for identifying online nonlinear directed graph topologies (see Fig. 1.3).

Leveraging on the proposed topology estimation algorithms, the thesis also designs methods to impute missing time-series data, which is an important network science problem related to several applications. Social networks, for instance, might have privacy concerns or might be impractical to collect information from all the nodes simultaneously due to their size. In the case of sensor networks, it might be due to sensor or communication failures or nonuniform sampling. In such cases, both the physical structure of the network and the hidden graph structure in the network can be used to impute the missing data. Motivated by this, we propose two algorithms in the dissertation for joint topology identification and data imputation.

## 1.2 Summarized State of the Art

Learning graphs from multivariate time series has gained significant research interest, and a number of linear graph learning strategies have been proposed in recent years [1–7]. Several of these works focus on learning undirected graphs [1, 4, 5] by merely considering the correlation between time series. However, real-world network interactions are often directional, and learning directed graphs [2, 6] that can give meaningful insights about the actual system are the main focus of the thesis. In [2], the authors propose a novel idea for learning directed graphs from a system identification viewpoint and propose a batch solution for the topology identification problem by assuming the graph is stationary. Such an approach is not suitable when the graph structure varies with time. An online solution for the VAR topology identification problem is presented in [6]. The proposed method [6] is capable of learning dynamic graphs from streaming data in a computationally light manner. In [8], the authors use a similar approach to identify a time-varying structural equation model. However, all the aforestated approaches fail to incorporate the nonlinear nature of real-world networks.

Although nonlinear topology identification is less studied than linear topology identification, some interesting approaches have been developed [9–12]. Neural networks are a powerful tool to model nonlinearities, and some relevant works in this direction have been put forward [9, 10]. In [9], the authors propose a neural Granger causality algorithm with automatic lag selection, whereas [10] proposes a novel topology identification method when the associated nonlinearity is invertible. To the best of our knowledge, all the solutions proposed using neural networks provide batch (offline) solutions. Kernels are another important tool for modelling nonlinearity [13]. In [11], a nonlinear structural vector autoregressive (SVAR) model is considered, and nonlinearity is tackled using the kernel method, whereas [14] uses kernels to model a nonlinear structural equation model. The problem associated with kernel methods is the curse of dimensionality [15], i.e., as the number of data samples increases, the computational complexity becomes prohibitive at some point. For this reason, the literature on online learning of nonlinear graphs is very limited. In [12], the authors propose a kernel-based online nonlinear topology identification algorithm and the associated curse of dimensionality is tackled via the so-called dictionary method, where the dictionary elements are selected based on a budget-maintaining strategy. An alternative way to overcome the curse of dimensionality is to use Random feature (RF) approximation [16]. Unlike the dictionary method, RF approximation allows us to work in a fixed lower dimensional space and use standard convex optimization techniques. Apart from that, using RF approximation gives the additional benefit of nodal data privacy [17] because nodes share random features rather than actual data.

Knowledge about the graph structure learned from multivariate time series data can aid in filling in missing data samples. A number of approaches have been proposed in this line [6, 18–20]. In [19], the authors provide a pretrained batch solution to the data imputation problem using generative adversarial networks. An algorithm to jointly learn linear graph and missing signal is proposed in [18] using block coordinate descent and Kalman smoothing, whereas a computationally light online solution to the problem is provided in [6] using inexact proximal gradient descent. Apart from pure data-driven approaches, inherent knowledge about the system can also be used for missing data imputation. A Simplicial Complex formulation, exploiting prior knowledge from the system, for imputing time-invariant data is presented in [20]. However, to the best of our knowledge, there is no work on the problem of simplicial complex aided missing data imputation for multivariate time series data in the literature.

## 1.3 Problem Statements

This dissertation addresses four problem statements derived from the current State of the Art:

- **Online nonlinear topology identification:** Given a stream of multivariate time series data, learn the time-lagged nonlinear directional graph structure from the data in an online fashion.

- **Joint online topology identification and missing data imputation:** Given multivariate time series data with missing entries, jointly learn the directed graph and impute the missing data jointly in an online fashion.

- **Online data imputation over a network, when the network structure is given:** Given streaming data from a linear VAR process and the physical structure of the network, impute the missing data using available observation and the structure of the network.

- **Online learning of nonlinear structural equation model with privacy:** Given the streaming data from a time-varying nonlinear SEM model, estimate the nonlinear SEM topology in an online way with nodal data privacy.

## 1.4 Outline of the Dissertation

This Ph.D. thesis is based on six papers attached in the Appendix, which are organized as the following chapters.

- *Chapter 2* contents provides the background theory needed for the readers to follow the thesis. In this chapter, we introduce the two nonlinear topology identification models used in the Ph.D. thesis *(i)* nonlinear vector autoregressive model *(ii)* nonlinear structural equation model. We also introduce the basics of online learning and the performance metric of dynamic regret. Dynamic regret is a popular metric to evaluate the performance of an online algorithm. Next, we explain the fundamentals of the well-known Kalman filter, and finally, we present an overview of the simplicial complex.

- *Chapter 3* summarizes papers A ( [21] ), B ( [22] ), and C ( [23] ). This chapter discusses the problem of online nonlinear topology identification. We propose three algorithms:

    1. Nonlinear Topology Identification by Sparse Online learning (NL-TISO) [21] described in Appendix A: a kernel-based online algorithm to estimate the nonlinear topologies, which uses a forgetting window to tackle the dimensionality growth arising from kernel formulation.
    2. Random Feature approximation for Nonlinear Topology Identification by sparse Online learning (RFNL-TISO) [22] described in Appendix B: even if the NL-TISO is capable of identifying the graph structure, the solution provided is suboptimal due to the forgetting window; to solve this problem, we use RF approximation. RF approximation allows us to work in a fixed lower-dimensional state and use convex optimization tools to develop an efficient online nonlinear topology identification algorithm.
    3. Random Feature approximation for Nonlinear Topology Identification by Recursive Sparse Online learning (RFNL-TIRSO) [23] described in Appendix C: an alternative algorithm inspired by the recursive least squares (RLS) formulation is presented which is more robust to observation noise than RFNL-TISO. Apart from the RLS formulation, we also provide convergence guarantees in terms of dynamic regret, which is the typical performance metric for an online algorithm.

  We remark that all these three algorithms feature sparse graph estimates, inspired by the fact that real-world dependencies are sparse.

- *Chapter 4* summarizes paper D ( [24] ). This chapter proposes an algorithm for joint online topology identification and missing data imputation. In this chapter, we first formulate a nonconvex optimization problem and convexify it by making certain assumptions. The convex version of the problem is in a form that can be solved by a two-step approach: *(1)* estimation of the sparse dependencies by proximal updates and *(2)* reconstruction of missing data from the observed signal and previous topology estimate.

- *Chapter 5* summarizes paper E ( [25] ). The chapter considers the problem of online data imputation over a network when the network structure is given. In this chapter, we first learn a line graph; then, the learned graph is used as a model to describe the data. The learned model, along with observations, is used to impute the data using a Kalman filtering framework. The SC formulation of our Kalman filter also allows us to incorporate contextual information specific to a network (e.g. flow conservation in a water network), as opposed to the conventional Kalman filters.

- *Chapter 6* summarizes paper F ( [26] ). The chapter introduces an algorithm for the problem of online learning nonlinear SEM models with privacy. The proposed algorithm estimates nonlinear directed SEM topologies using a prediction correction approach. Additionally, the proposed approach is designed in a way that the nodes of the network do not have to share real data with each other. Finally, we also evaluate the dynamic regret of the algorithm.

- *Chapter 7* In this chapter, the main conclusions of the dissertation are discussed. The chapter also lists possible future directions to extend the thesis.

# Chapter 2

# Background Theory

## 2.1 Multivariate Time Series Models

For designing the topology identification algorithm, we first have to assume that a certain model is appropriate for describing the observed data. The choice of this model depends on various factors such as the application, the peculiarity of the underlying process, sampling time, etc. As the choice of model is cardinal in the topology identification, we describe the models used in the dissertation.

### 2.1.1 Linear Vector Autoregressive Model

Most of the real-world dependencies are time-lagged in nature, so the choice of vector autoregressive models is natural. Consider a multivariate time series $\{y_n[t]\}_{n=1}^{N}$ where $y_n[t]$ is the value of the time-series at time $t = 1, 2, \ldots, T$ measured at a given node $1 \leq n \leq N$. A $P$-th order linear VAR model is expressed as

$$y_n[t] = \sum_{n'=1}^{N} \sum_{p=1}^{P} a_{n,n'}^{(p)} y_{n'}[t-p] + u_n[t], \tag{2.1}$$

where $a_{n,n'}^{(p)}$ captures the influence of the $p$-lagged data at node $n'$ on the node $n$, and $u_n[t]$ is the process noise.

### 2.1.2 Nonlinear Vector Autoregressive Model

Most of the physical systems are nonlinear in nature, and a linear model only captures an approximate representation of the actual physical reality, which usually results in a suboptimal solution. Based on this fact, this dissertation proposes solutions based on nonlinear models. The first nonlinear model that we use in this dissertation is the nonlinear vector autoregressive model, which is capable of expressing time-lagged nonlinear directed interactions.

A $P$-th order nonlinear VAR model can be expressed as

$$y_n[t] = \sum_{n'=1}^{N} \sum_{p=1}^{P} f_{n,n'}^{(p)}(y_{n'}[t-p]) + u_n[t]. \tag{2.2}$$

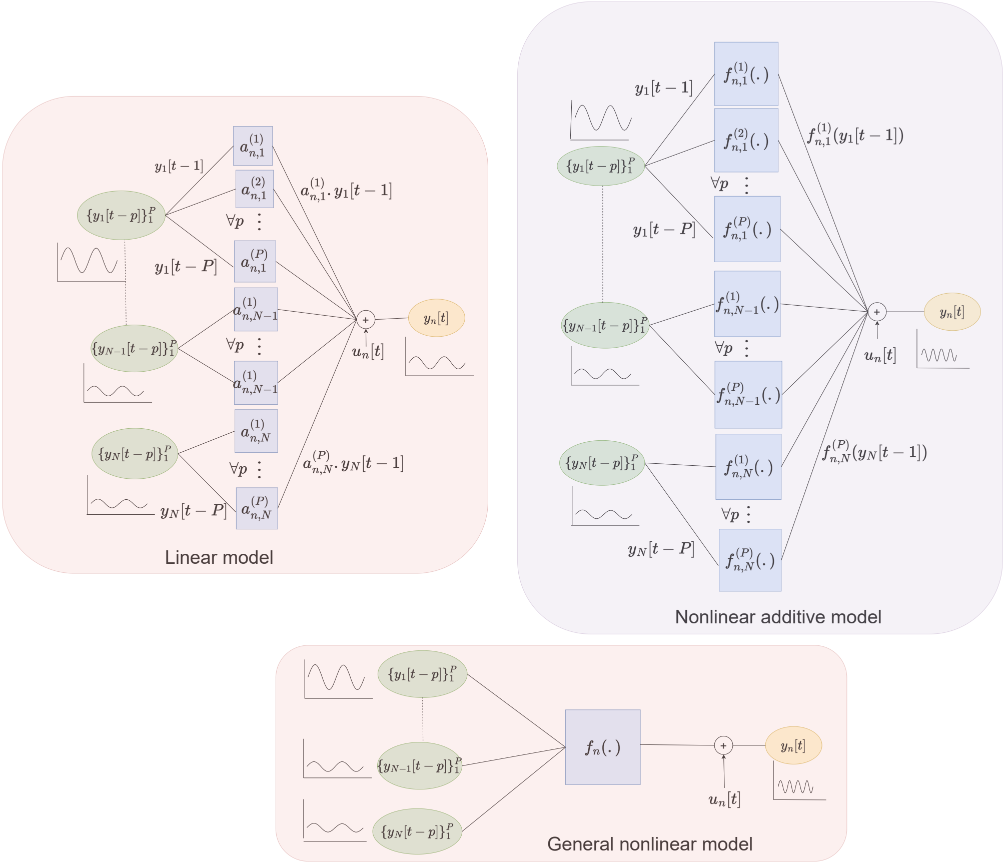


Figure 2.1: Various models to represent multivariate time series.

Equation (2.2) expresses the value of the $n$-th time series at time $t$ as a combination of $P$ time-lagged values of all the available time series. The function $f^{(p)}_{n,n'}$ encodes the nonlinear causal influence of the $p$-th time-lagged value of the $n'$-th time series on the $n$-th time series. Note that the model we consider is an additive nonlinear model. Despite the fact that a general nonlinear model may fit the data better in theory, we choose an additive model because of the following reasons: *(i)* it has more explanatory power (see, Fig. 2.1 in a nonlinear additive model, the contribution of each individual component or node can be separately identified, unlike in a general nonlinear model ), and *(ii)* it supports a solid framework necessary to the design of the online algorithms in the sequel. Also, the additive nonlinear models have been shown to be important in many applications, such as brain connectivity analysis [11]. Note that the equation (2.1) is a special case of the nonlinear additive model (2.2) under consideration.

### 2.1.3 Nonlinear Structural Equation Model

Structural equation models (SEM) are usually used to model multivariate time series when interactions are faster than the sampling time; such interactions are prevalent in brain networks, financial networks, social networks, etc. SEM is widely used to represent network relationships since they are simple and can express directional rather than correlation-based symmetrical relationships. Multivariate time series data can be modelled using a nonlinear SEM model as

$$y_n[t] = \sum_{n'=1, n' \neq n}^{N} f_{n,n'}(y_{n'}[t]) + u_n[t], \;\; n = 1, \ldots, N, \tag{2.3}$$

where $f_{n,n'}(.)$ is the influence of $n'$-th node on $n$-th node and $u_n[t]$ is the innovation noise.

## 2.2 Expressing Nonlinearity

Nonlinear input-output relationships are mostly modeled using kernel method [27,28] and neural networks in the literature. Since the goal of the thesis is to design online algorithms capable of tracking dynamic graphs, we focus on kernel methods.

### 2.2.1 Function in Reproducing Kernel Hilbert Space

In order to model the possible function using kernel methods to represent nonlinear relationships, we assume that the function belongs to a reproducing kernel Hilbert space (RKHS) [15]. Any continuous function $f(.)$ in an RKHS can be expressed as an infinite sum of kernel evaluations:

$$\mathcal{H} := \left\{ f(\cdot) \mid f\left(x[t]\right) = \sum_{t=1}^{\infty} \beta_t \; \kappa\left(x[t'], x[t]\right) \right\}, \tag{2.4}$$

where the function $\kappa\left(.,.\right)$ measures the similarity between the arguments and is termed the kernel. Every RKHS has a definite kernel associated with it and an inner product $\langle \kappa(y, x_1), \kappa(y, x_2) \rangle := \sum_{t=0}^{\infty} \kappa(y[t], x_1)\kappa(y[t], x_2)$, which characterizes the RKHS. Positive-definite kernels satisfy the reproducing property $\langle \kappa(y, x_1), \kappa(y, x_2) \rangle = \kappa(x_1, x_2)$, and a norm is induced as $\|f\|_{\mathcal{H}}^2 = \sum_{t=0}^{\infty} \sum_{t'=0}^{\infty} \beta_t \; \beta_{t'} \; \kappa(y[t], y[t'])$. Let us consider a functional optimization problem in RKHS:

$$\hat{f} = \arg \min_{\{f \in \mathcal{H}\}} \frac{1}{2} \sum_{\tau=0}^{T-1} \left[ x[\tau] - f\left(y[\tau]\right) \right]^2 + \lambda \Omega\left(||f||_{\mathcal{H}}\right), \tag{2.5}$$

where $x[\tau]$ and $y[\tau]$ represent an input-output pair, $f(.)$ encodes the functional relationship between them, $T$ is the total number of data samples available, and $\lambda\Omega\left(||f||_{\mathcal{H}}\right)$ is a regularization term which consist of nondecreasing function $\Omega(.)$ and a positive constant $\lambda > 0$. Here, the goal is to estimate the function $f(.)$ from the observations $\{x[\tau]\}_{\tau=1}^{T}$ and $\{y[\tau]\}_{\tau=1}^{T}$. In RKHS, the function is infinite-dimensional and requires an infinite number of kernel evaluations and parameters, making the optimization problem (2.5) infeasible. The solution to the optimization problem (2.5) can be expressed with a finite number of kernel evaluations using the Representer Theorem. As the function $\Omega(.)$ is non-decreasing, the Representer Theorem can be invoked, yielding a finite-dimensional solution

$$\hat{f}\left(y[\tau]\right) = \sum_{t=0}^{T-1} \beta_t \; \kappa(y[\tau], y[t])\,. \tag{2.6}$$

Note that here the number of kernel evaluations is equal to the number of data samples available, which circumvents the issues associated with the infinite dimension of RKHS. However, it is important to note that, as the number of data samples increases, the number of parameters required to express the function in (2.6) increases, resulting in prohibitive computational complexity. We use Random Features (RF) approximation to overcome this curse of dimensionality.

### 2.2.2 Random Features Approximation

According to the Johnson–Lindenstrauss Lemma [29], any two points in a higher dimensional space can be expressed in a lower dimension if the distance between points is preserved. In an infinite-dimensional RKHS, distance is measured using the norm induced by the inner product. In RF [16] approximation, the inner product is approximated in a fixed lower dimensional space, which allows us to embed the functions in RKHS into a lower dimensional space. For RKHS generating from shift-invariant kernels, the inner product can be approximated with a fixed number of random Fourier features using Bochner's theorem [30]. In this thesis, we exploit the above fact and approximate kernel evaluations using random Fourier features. In [31], the authors also explore the potential of employing RF approximation to protect privacy in graph-based inference tasks. Similarly, in Chapter 6, we also leverage RF approximation for privacy preservation purposes.

## 2.3 Online Learning

Input-output relationships can be expressed using the RF approximation with a fixed dimension. However, notice that it is not trivial to estimate the required function, as the optimization problem (2.5) proposed is in batch form. However, the optimization (2.2) is in a batch form, meaning that the problem is solved by finding the best fit of the data after collecting all the data points. Such a conventional offline approach has two major drawbacks: *(i)* it fails to track changes in the model as the entire data is used to fit the optimal parameters *(ii)* a high computational capability is required to perform such an optimization problem, and *(iii)* it is impossible to perform such an operation in a real-time applications where we do not have access to the entire batch of data. Online convex optimization is the paradigm where the model is updated on the fly according to the data stream. Consider an optimization problem

$$\arg\min_{\mathbf{a}} \frac{1}{T} \sum_{t=0}^{T-1} h_t(\mathbf{a}) \tag{2.7}$$

where $h_t(.)$ is the time-varying cost function and $\mathbf{a}$ the model parameter. Conventional batch solutions require all the $T$ data samples to solve the problem. Unlike batch solutions, in an online framework, the model is updated on the fly every time a new data stream is available. This not only reduces the computational bottleneck

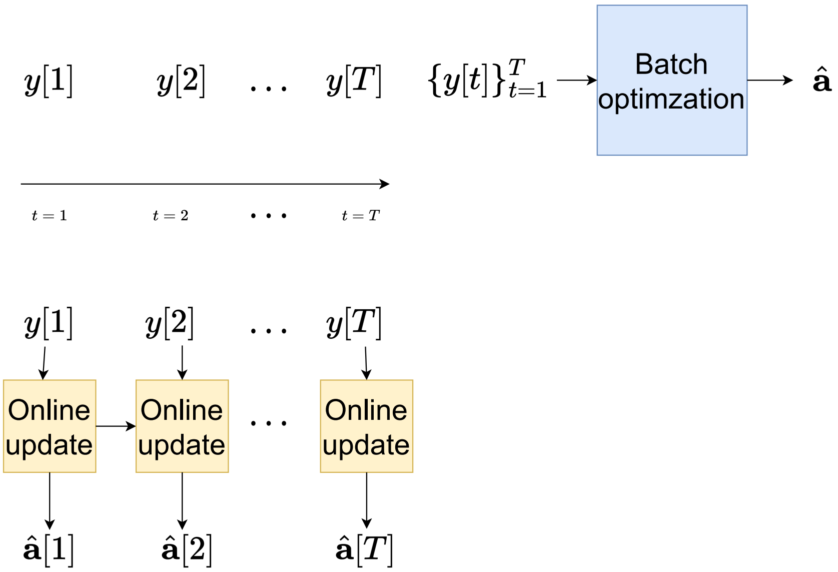


Figure 2.2: Data $y[t]$ is available at each time instant $1 \leq t \leq T$. The batch approach shown at the top collects all the data samples and solves a batch optimization problem to give a single model parameter at time $T$. The online approach is shown at the bottom; it updates the model parameter whenever a new data sample is available based on observed data and the previous state of the model. In the online approach, the evolution of parameters is tracked as $\{\hat{\mathbf{a}}[t]\}_{t=1}^{T}$.

of the batch solution but also tracks the change in model parameters, as shown in Fig. 2.2.

### 2.3.1 Dynamic Regret

Dynamic regret [32] is a popular metric to evaluate the capability of an online algorithm and is defined as

$$R[T] = \sum_{t=0}^{T-1} h_t(\mathbf{a}[t]) - h_t(\mathbf{a}^*[t]) \tag{2.8}$$

where $\mathbf{a}^*[t] \triangleq \arg\min_{\mathbf{a}} \; h_t(\mathbf{a})$ is the optimal minimizer of the cost function at time instant $t$ and $\mathbf{a}[t]$ is the estimate computed using the online algorithm. Dynamic regret provides a measure of how well the estimated solution tracks the optimum solution. If the dynamic regret $R[T]$ is sublinear, the estimated solution will converge to the optimal solution asymptotically, that is $\lim\limits_{T\to\infty} \frac{R[T]}{T} = 0$.

## 2.4 Kalman Filter

The observations from practical networks are often noisy. In a dynamical system, the Kalman filter can be used to infer system variables by combining information from the observations and model dynamics. Such inferences will be superior to the estimation techniques that rely just on observation. In Chapter 5 of this thesis,

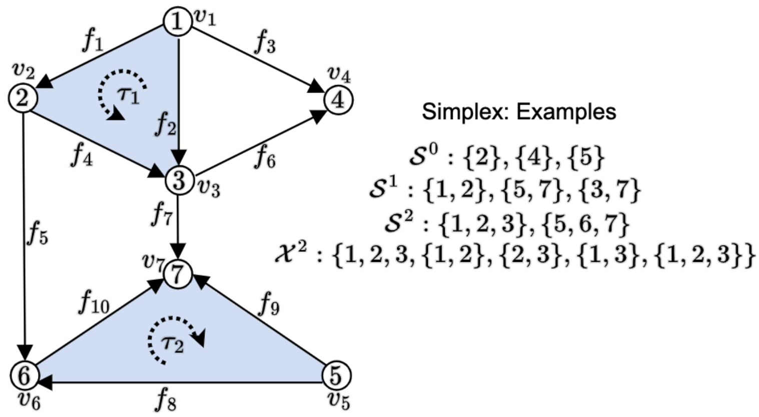

$v_1$
$v_2$
$v_3$
$v_4$
$v_5$
$v_6$
$v_7$
1
2
3
4
5
6
7
$f_1$
$f_2$
$f_3$
$f_4$
$f_5$
$f_6$
$f_7$
$f_8$
$f_9$
$f_{10}$
$\tau_1$
$\tau_2$
Simplex: Examples
$\mathcal{S}^0 : \{2\}, \{4\}, \{5\}$
$\mathcal{S}^1 : \{1,2\}, \{5,7\}, \{3,7\}$
$\mathcal{S}^2 : \{1,2,3\}, \{5,6,7\}$
$\mathcal{X}^2 : \{1,2,3, \{1,2\}, \{2,3\}, \{1,3\}, \{1,2,3\}\}$

Figure 2.3: A geometrical representation of a simplicial complex and associated simplicial signals.

we propose a data imputation strategy based on the Kalman filtering framework. Consider state space equations with a noisy model and noisy observations:

$$\mathbf{x}[t] = \mathbf{A}\mathbf{x}[t-1] + \mathbf{u}[\mathbf{t}] \tag{2.9}$$

$$\mathbf{y}[t] = \mathbf{C}\mathbf{x}[t] + \mathbf{v}[\mathbf{t}] \tag{2.10}$$

where $\mathbf{x}[t] \in \mathbb{R}^N$ is the value of state vector at time $t$, $\mathbf{A} \in \mathbb{R}^{N \times N}$ is the state transition matrix, $\mathbf{y}[t] \in \mathbb{R}^M$ is the observation of state measured using the measurement matrix $\mathbf{C} \in \mathbb{R}^{M \times N}$, $\mathbf{u}[t] \in \mathbb{R}^N$ and $\mathbf{v}[t] \in \mathbb{R}^M$ are model and observation noise respectively. For such a system, the best linear unbiased estimate of $\mathbf{x}[t]$ is given by the Kalman filter [33]. The Kalman filter works in two steps, prediction and update:
**prediction step:**

$$\hat{\mathbf{x}}_{t|t-1} = \mathbf{A}\mathbf{x}_{t-1|t-1} \tag{2.11}$$

$$\mathbf{P}_{t|t-1} = \mathbf{A}\mathbf{P}_{t-1|t-1}\hat{\mathbf{A}}^\top + \mathbf{Q}_t, \tag{2.12}$$

where $t|t-1$ is the estimate at time $t$ given the measurement up to $t-1$, $\mathbf{P}_{t|t-1} \in \mathbb{R}^{N \times N}$ is the prediction error covariance matrix and $\mathbf{Q}_t \in \mathbb{R}^{N \times N}$, the noise covariance matrix.
**update step:**

$$\mathbf{x}_{t|t} = \mathbf{x}_{t|t-1} + \mathbf{K}(\mathbf{C}\mathbf{y}[t] - \mathbf{x}_{t|t-1}), \tag{2.13}$$

$$\mathbf{P}_{t|t} = \mathbf{P}_{t|t-1} - \mathbf{K}\mathbf{C}\mathbf{P}_{t|t-1}, \tag{2.14}$$

where $\mathbf{K}{=}\mathbf{P}_{t|t-1}\mathbf{C}^\top(\mathbf{C}\mathbf{P}_{t|t-1}\mathbf{C}^\top{+}\mathbf{R}_t)^{-1}$ is the so-called Kalman gain and $\mathbf{R}_t$ is the covariance matrix of the observation noise. Note that when the observation and process noise are Gaussian, the Kalman filter is the minimum variance, unbiased estimator.

## 2.5 Simplicial Complex

We have discussed in the previous sections, how to learn hidden graph structures from observation. In some applications (e.g. water network), the physical structure of the network will be known, and this additional information can aid in various tasks such as topology identification, data imputation, denoising, etc. In the above tasks, we can incorporate such structural information using the simplicial complex (SC) representation of the network for data defined on higher-order topological structures (e.g., edges, triangles). Given the set of nodes $\mathcal{V}$, a $k$-simplex $\mathcal{S}^k$ is a subset of $\mathcal{V}$ having $k+1$ distinctive elements [34], [35]. A SC of order $K$, denoted as $\Psi^K$, is a set of $k$-simplices for $k = 0, 1 \ldots, K$ such that a simplex $\mathcal{S}^k \in \Psi^K$ only if all of its subsets also belong to $\Psi^K$. The proximities between different $k$-simplices in an SC can be represented using an incidence matrix $\mathbf{B}_k \in \mathbb{R}^{N_{k-1} \times N_k}, k \geq 1$, where the row and the column indices of $\mathbf{B}_k$ correspond to $(k-1)$- and $k$-simplices, respectively. The structure of an SC is encoded by Hodge Laplacians, constructed using $\mathbf{B}_k$'s as

$$\mathbf{L}_k = \begin{cases} \mathbf{B}_{k+1}\mathbf{B}_{k+1}^\top, & \text{for } k = 0, \\ \mathbf{B}_k^\top \mathbf{B}_k + \mathbf{B}_{k+1}\mathbf{B}_{k+1}^\top, & \text{for } 1 \leq k \leq K-1, \\ \mathbf{B}_K^\top \mathbf{B}_K, & \text{for } k = K, \end{cases} \tag{2.15}$$

In an SC, $k$-simplex signals are mappings from $k$-simplices to the real set $\mathbb{R}$. The 0-simplex, 1-simplex, and 2-simplex signals reside on the nodes, edges, and triangles, respectively (see, Fig. 2.3). For instance, consider a water distribution network. The demand at each node can be considered a 0-simplex signal, and the volume of water flow between two nodes is a 1-simplex signal.

# Chapter 3

# Online Nonlinear Topology Identification

This chapter summarizes Paper A ([21]), Paper B ([22]), and Paper C ([36]).

## 3.1 Motivation

Different methods for estimating time-varying spatio-temporal relationships among time series are mentioned in Section 1.2. Most of the proposed works in the literature are based on linear models. In this chapter, we focus on learning time-varying nonlinear dependencies. Using kernel-based nonlinear models and online convex optimization tools, we propose three online nonlinear topology identification algorithms, namely NL-TISO [21], RFNL-TISO [22], and RFNL-TIRSO [36]. The kernel framework poses a significant challenge due to the increase in dimensionality with the number of data samples. This dimensionality growth is circumvented in our first algorithm NL-TISO by discarding the past data samples using a forgetting window. However, such an approach can lead to suboptimal function learning because it discards data samples without assessing their significance in representing the functions to be learned. Hence, building upon NL-TISO, we propose the second algorithm RFNL-TISO, where the dimensionality growth is tackled through random feature approximation. The third work, RFNL-TIRSO, introduces a recursive least square loss in place of the least mean square loss used in RFNL-TISO. Next, we develop a kernel-based framework to model and learn these nonlinear functional dependencies.

## 3.2 Problem Formulation

Consider a multivariate time series with $N$ nodes. Let $y_n[t]$ be the value of time series at time $t = 0, 1, \dots, T-1$, observed at node $n$, $1 \leq n \leq N$. A $P$-th order nonlinear VAR model assuming additive functional dependencies can be formulated as

$$y_n[t] = \sum_{n'=1}^{N} \sum_{p=1}^{P} f_{n,n'}^{(p)}(y_{n'}[t-p]) + u_n[t], \tag{3.1}$$

where $f_{n,n'}^{(p)}$ is the function that encodes the nonlinear causal influence of the $p$-lagged data at node $n'$ on the node $n$ and $u_n[t]$ is the observation noise (although $f_{n,n'}^{(p)}$ changes with time, we chose to avoid time index in order to avoid complication in notation). Considering the model (3.1), topology identification can be defined as the estimation of the functional dependencies $\left\{f_{n,n'}^{(p)}(.)\right\}_{p=1}^{P}$ for $n = 1, 2, \ldots, N$ from the observed time series $\{y_n[t]\}_{n=1}^{N}$. In order to model nonlinearity, we utilize the kernel method and consider the Reproducing Kernel Hilbert Space (RKHS) framework.

## 3.3 Nonlinearity

We assume that the functions $f_{n,n'}^{(p)}$ in (3.1) belong to a reproducing kernel Hilbert space (RKHS):

$$\mathcal{H}_{n'}^{(p)} := \left\{ f_{n,n'}^{(p)} \mid f_{n,n'}^{(p)}(y) = \sum_{t=0}^{\infty} \beta_{n,n',t}^{(p)} \, \kappa_{n'}^{(p)}(y, y_{n'}[t-p]) \right\}, \tag{3.2}$$

where $\kappa_{n'}^{(p)} : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ is the kernel associated with the Hilbert space. The kernel measures the similarity between data points $y$ and $y_{n'}[t-p]$. Referring to (3.2), evaluation of the function $f_{n,n'}^{(p)}$ at $y$ can be represented as the linear combination of the similarities between $y$ and the data points $\{y_{n'}[t-p]\}_{t=0}^{t=\infty}$, with weights $\beta_{n,n',t}^{(p)}$. The inner product, $\langle \kappa_{n'}^{(p)}(y, x_1), \kappa_{n'}^{(p)}(y, x_2) \rangle := \sum_{t=0}^{\infty} \kappa_{n'}^{(p)}(y[t], x_1) \kappa_{n'}^{(p)}(y[t], x_2)$, is defined in the Hilbert space using kernels with reproducible property $\langle \kappa_{n'}^{(p)}(y, x_1), \kappa_{n'}^{(p)}(y, x_2) \rangle = \kappa_{n'}^{(p)}(x_1, x_2)$. Such a Hilbert space with the reproducing kernels is termed RKHS and the inner product described above induces a norm, $\|f_{n,n'}^{(p)}\|_{\mathcal{H}_{n'}^{(p)}}^2 = \sum_{t=0}^{\infty} \sum_{t'=0}^{\infty} \beta_{n,n',t}^{(p)} \, \beta_{n,n',t'}^{(p)} \, \kappa_{n'}^{(p)}(y_n[t], y_n[t'])$. We refer to [37] for further reading on RKHS. For a particular node $n$, the estimates of $\left\{f_{n,n'}^{(p)} \in \mathcal{H}_{n'}^{(p)}\right\}_{n',p}$ are obtained by solving the functional optimization problem:

$$\begin{aligned} \left\{\hat{f}_{n,n'}^{(p)}\right\}_{n',p} = \arg \min_{\left\{f_{n,n'}^{(p)} \in \mathcal{H}_{n'}^{(p)}\right\}} & \frac{1}{2} \sum_{\tau=P}^{T-1} \left[ y_n[\tau] - \sum_{n'=1}^{N} \sum_{p=1}^{P} f_{n,n'}^{(p)}(y_{n'}[\tau-p]) \right]^2 \\ & + \lambda \sum_{n'=1}^{N} \sum_{p=1}^{P} \Omega(||f_{n,n'}^{(p)}||_{\mathcal{H}_{n'}^{(p)}}). \end{aligned} \tag{3.3}$$

The objective function in equation (3.3) comprises two essential terms, namely the data fitting and sparsity-promoting regularizer. The primary aim of the former is to ensure that the model fits the available data correctly. On the other hand, the regularizer term promotes sparsity, which not only helps to prevent overfitting but also aids in the development of more interpretable graphs with fewer connections. By incorporating a sparsity-promoting regularizer in the objective function, the model is encouraged to identify and select only the most relevant features or connections, while penalizing the selection of too many non-zero weights or connections. This approach facilitates the creation of more efficient and interpretable models that are less susceptible to noise and overfitting. It is to be noted that in (3.3), the functions $\{f_{n,n'}^{(p)}\}$ belong to the RKHS defined in (3.2), which is an infinite dimensional space. However, For a non-decreasing function $\Omega$, by resorting to the Representer Theorem [38], the solution of (3.3) can be written using a finite number of data samples:

$$\hat{f}_{n,n'}^{(p)}(y_{n'}[\tau-p]) \quad = \sum_{t=p}^{p+T-1} \beta_{n,n',(t-p)}^{(p)} \kappa_{n'}^{(p)}(y_{n'}[\tau-p]), y_{n'}[t-p]). \tag{3.4}$$

Notice that the number of coefficients required to express the function increases with the number of data samples.

## 3.4 NL-TISO

Using (3.4), (3.3) can be reformulated as a parametric optimization problem involving the available data samples, as follows:

$$\left\{\widehat{\beta}^{(p)}_{n,n',t}\right\}_{n',p,t} = \arg \min_{\left\{\beta^{(p)}_{n,n',t}\right\}} \mathcal{L}^n\left(\beta^{(p)}_{n,n',t}\right), \tag{3.5}$$

where

$$\mathcal{L}^n\left(\beta^{(p)}_{n,n',t}\right) := \frac{1}{2}\sum_{\tau=P}^{T-1}\left[y_n[\tau] - \sum_{n'=1}^{N}\sum_{p=1}^{P}\sum_{t=p}^{p+T-1}\beta^{(p)}_{n,n',t}\kappa^{(p)}_{n'}\left(\tau,t\right)\right]^2, \tag{3.6}$$

and

$$\kappa^{(p)}_{n'}\left(\tau,t\right) := \kappa^{(p)}_{n'}\left(y_{n'}[\tau-p]), y_{n'}[t-p]\right). \tag{3.7}$$

The optimization problem in (3.5) can be reformulated as

$$\widehat{\boldsymbol{\beta}}_n = \arg\min_{\boldsymbol{\beta}_n} \frac{1}{2}\sum_{\tau=P}^{T-1}\left[y_n[\tau] - \boldsymbol{\beta}_n^\top \boldsymbol{\kappa}_\tau\right]^2 + \lambda\sum_{n'=1}^{N}\sum_{p=1}^{P}\|\boldsymbol{\beta}^{(p)}_{n,n'}\|_2. \tag{3.8}$$

where $\boldsymbol{\beta_n}$ and $\boldsymbol{\kappa}_\tau$ are obtained by stacking the parameters and kernel evaluations given in (3.4); and $\lambda\sum_{n'=1}^{N}\sum_{p=1}^{P}\|\boldsymbol{\beta}^{(p)}_{n,n'}\|_2$ is the sparsity providing regularizer (see Section 3.5 for the derivation and detailed explanation of the regularizer). The stacking operations are explained below:

$$\boldsymbol{\beta}_{n,n',t} := [\beta^{(1)}_{n,n',t}, \beta^{(2)}_{n,n',t}, \ldots, \beta^{(P)}_{n,n',t}]^\top \in \mathbb{R}^P \tag{3.9}$$

$$\boldsymbol{\beta}_{n,t} := [\boldsymbol{\beta}^\top_{n,1,t}, \boldsymbol{\beta}^\top_{n,2,t}, \ldots, \boldsymbol{\beta}^\top_{n,N,t}] \in \mathbb{R}^{NP} \tag{3.10}$$

$$\boldsymbol{\kappa}_{n'}\left(\tau,t\right) := [\kappa^{(1)}_{n'}\left(\tau,t\right), \kappa^{(2)}_{n'}\left(\tau,t\right), \ldots, \kappa^{(P)}_{n'}\left(\tau,t\right)]^\top \in \mathbb{R}^P \tag{3.11}$$

$$\boldsymbol{\kappa}\left(\tau,t\right) := [\boldsymbol{\kappa}_1\left(\tau,t\right)^\top, \boldsymbol{\kappa}_2\left(\tau,t\right)^\top, \ldots, \boldsymbol{\kappa}_N\left(\tau,t\right)^\top] \in \mathbb{R}^{NP} \tag{3.12}$$

$$\boldsymbol{\beta}_n := [\boldsymbol{\beta}_{n,0}, \boldsymbol{\beta}_{n,1}, \ldots, \boldsymbol{\beta}_{n,T-1}]^\top \in \mathbb{R}^{NPT} \tag{3.13}$$

$$\boldsymbol{\kappa}_\tau := [\boldsymbol{\kappa}\left(\tau,p\right), \boldsymbol{\kappa}\left(\tau,p+1\right), \ldots, \boldsymbol{\kappa}\left(\tau,p+T-1\right)]^\top \in \mathbb{R}^{NPT}, \tag{3.14}$$

The optimization problem presented in (3.8) is a batch (offline) solver, which requires access to all data samples $y_n[\tau]_{\tau=P}^{T-1}$ to find the optimal coefficients $\boldsymbol{\beta}_n$. However, this offline approach has two main drawbacks. Firstly, it is not suitable for real-time applications as it requires the solver to wait for the entire batch of data. Secondly, it suffers from high computation complexity and memory requirements, which increase super-linearly with the batch size.

To overcome these limitations, we propose an online algorithm for estimating the coefficients $\boldsymbol{\beta}_n$ in (3.8). This approach is designed to work with streaming data, where each data point is processed one at a time, and the coefficients are updated incrementally. By using an online approach, the computation complexity and memory requirements are significantly reduced, making it suitable for real-time applications. First replace the original loss function in (3.8) with the instantaneous loss function $l_\tau^n(\boldsymbol{\beta}_n) = \frac{1}{2}[y_n[\tau] - \boldsymbol{\beta}_n^\top \boldsymbol{\kappa}_\tau]^2$:

$$\widehat{\boldsymbol{\beta}}_n = \arg\min_{\boldsymbol{\beta}_n} l_\tau^n(\boldsymbol{\beta}_n) + \lambda \sum_{n'=1}^{N} \sum_{p=1}^{P} \|\boldsymbol{\beta}_{n,n'}^{(p)}\|_2. \tag{3.15}$$

The cost function presented in (3.15) consists of a loss function that is differentiable and a non-differentiable group-Lasso regularizer. To solve this optimization problem online, we can use the online subgradient descent (OSGD) or mirror descent (MD) methods. However, these methods work by linearizing the entire objective function in (3.15) using a subgradient of it. If the group-Lasso regularizer is linearized, its ability to induce sparsity in the estimates is compromised, resulting in non-sparse estimates.

To address this issue, we adopt an alternate optimization technique called composite objective mirror descent (COMID) [39]. This is a modified version of the MD algorithm that linearizes only the differentiable part of the objective function, while keeping the regularizer intact. By doing so, the sparsity-inducing property of the group-Lasso regularizer is preserved, and the resulting estimates remain sparse. The online COMID update can be written as

$$\boldsymbol{\beta}_n[t+1] = \arg\min_{\boldsymbol{\beta}_n} J_t^{(n)}(\boldsymbol{\beta}_n), \tag{3.16}$$

where

$$\begin{aligned} J_t^{(n)}(\boldsymbol{\beta}_n) \triangleq\ & \nabla \ell_t^n(\tilde{\boldsymbol{\beta}}_n[t])^\top \left(\boldsymbol{\beta}_n - \tilde{\boldsymbol{\beta}}_n[t]\right) \\ & + \frac{1}{2a_t}\|\boldsymbol{\beta}_n - \tilde{\boldsymbol{\beta}}_n[t]\|_2^2 + \lambda \sum_{n'=1}^{N} \sum_{p=1}^{P} \|\boldsymbol{\beta}_{n,n'}^{(p)}\|_2. \end{aligned} \tag{3.17}$$

The equation (3.17) defines $\tilde{\boldsymbol{\beta}}_n[t] \in \mathbb{R}^{PN(t+1)}$ as $[\boldsymbol{\beta}_n[t]; \mathbf{0}]$, where $\boldsymbol{\beta}_n[t] \in \mathbb{R}^{PNt}$ represents the estimated value of $\boldsymbol{\beta}_n$ using the samples up to time $t$. Here, $\mathbf{0} \in \mathbb{R}^{PN}$ denotes the initialization vector for the coefficients of the new kernel vector elements corresponding to the $(t+1)^{th}$ data sample. In (3.17), the first term is the gradient of the loss function, the second term is the Bregman divergence $B(\boldsymbol{\beta}_n, \tilde{\boldsymbol{\beta}}_n[t]) = \frac{1}{2}|\boldsymbol{\beta}_n - \tilde{\boldsymbol{\beta}}_n[t]|_2^2$, which is selected to have a closed-form solution for the COMID update [40]. The third term is a sparsity enforcing regularizer aimed at promoting sparsity in the updates, and $a_t$ represents the corresponding step size. Bregman divergence ensures that $\boldsymbol{\beta}_n[t+1]$ is similar to $\tilde{\boldsymbol{\beta}}_n[t]$ by assuming that the topology changes smoothly. The gradient in (3.17) is computed as

$$\mathbf{v}_n[t] := \nabla \ell_t^n(\tilde{\boldsymbol{\beta}}_n[t]) = \boldsymbol{\kappa}_\tau \left(\boldsymbol{\beta}_n^\top \boldsymbol{\kappa}_\tau - y_n[\tau]\right). \tag{3.18}$$

By ignoring the constants, the objective function in (3.17) can be expressed as

$$\begin{aligned} J_t^{(n)}(\boldsymbol{\beta}_n) &\propto \frac{\boldsymbol{\beta}_n^\top \boldsymbol{\beta}_n}{2a_t} + \boldsymbol{\beta}_n^\top \left(\mathbf{v}_n[t] - \frac{1}{a_t}\tilde{\boldsymbol{\beta}}_n[t]\right) + \lambda \sum_{n'=1}^{N} \sum_{p=1}^{P} \|\boldsymbol{\beta}_{n,n'}^{(p)}\|_2 \\ &= \sum_{n'=1}^{N} \sum_{p=1}^{P} \left[ \frac{{\boldsymbol{\beta}_{n,n'}^{(p)}}^\top \boldsymbol{\beta}_{n,n'}^{(p)}}{2a_t} + {\boldsymbol{\beta}_{n,n'}^{(p)}}^\top \left(\mathbf{v}_{n,n'}^{(p)}[t] - \frac{1}{a_t}\tilde{\boldsymbol{\beta}}_{n,n'}^{(p)}[t]\right) + \lambda \|\boldsymbol{\beta}_{n,n'}^{(p)}\|_2 \right]. \end{aligned} \tag{3.19}$$

Note that (3.19) is separable in $n'$, $m$ and $p$. Using (3.19), a closed form solution of (3.16) can be obtained in terms of multidimensional shrinkage-thresholding operator [41] as

$$\boldsymbol{\beta}^{(p)}_{n,n'}[t+1] = \left(\tilde{\boldsymbol{\beta}}^{(p)}_{n,n'}[t] - a_t\mathbf{v}^{(p)}_{n,n'}[t]\right) \times \left[1 - \frac{a_t\lambda}{\|\tilde{\boldsymbol{\beta}}^{(p)}_{n,n'}[t] -_t \mathbf{v}^{(p)}_{n,n'}[t]\|_2}\right]_+, \tag{3.20}$$

where $[x]_+ = \max\{0, x\}$. The term $\tilde{\boldsymbol{\beta}}^{(p)}_{n,n'}[t] - a_t\mathbf{v}^{(p)}_{n,n'}[t]$ in (3.20) performs a stochastic gradient update of $\boldsymbol{\beta}^{(p)}_{n,n'}$ in a direction that decreases the instantaneous loss function $l^n_\tau(\boldsymbol{\beta}_n)$ and the second term in (3.20) promotes group sparsity of $\boldsymbol{\beta}^{(p)}_{n,n'}$. The proposed NL-TISO algorithm is summarized as **Algorithm 1**.

**Algorithm 1:** NL-TISO Algorithm

```
Result: $\boldsymbol{\beta}^{(p)}_{n,n'}, for\ n, n' = 1,..,N$ and $p = 1,..,P$
Store $\{\boldsymbol{y}_n[t]\}_{t=1}^{P}$,
Initialize $\lambda$, $a_t$ (heuristically chosen) and kernel parameters depending on
  the type of the kernel.
for $t = P, P+1, \dots$ do
    Get data samples $y_n[t]$, $\forall n$ and compute $\boldsymbol{\kappa}_\tau$
    for $n = 1, \dots, N$ do
        compute $\mathbf{v}_n[t]$ using (3.18)
        for $n' = 1, \dots, N$ do
            compute $\boldsymbol{\beta}^{(p)}_{n,n'}[t+1]$ using (3.20)
        end
    end
end
```

Note that the solution (3.20) suffers from dimensionality growth of the kernel, i.e., the dimension of the variables increases with $t$. In NL-TISO, we address this issue by selecting only the most recent $T_w$ data points to compute (3.20), where $T_w$ is set to 2000 heuristically. Although this is a sub-optimal approach, it still provides competitive empirical performance, as demonstrated in the experiments section (see, paper [21]). As explained in the next section, in order to mitigate the problem, we use Random feature approximation which is explained in the following section.

## 3.5 RFNL-TISO

As mentioned, since the number of features increases with data samples, online learning becomes prohibitive at some point, referred to as the curse of dimensionality. The interesting fact about RKHS is that it is defined by an inner product. If we are able to approximate this inner product in a lower dimensional space, any function in the original infinite dimensional space can be approximately expressed in a lower dimensional space. In this work, in alignment with [42] and [43], we use Random Feature (RF) approximation to tackle the dimensionality growth. To invoke the RF approximation, we restrict our choice of kernels to the shift-invariant class, i.e., $\kappa^{(p)}_{n'}\,(y_{n'}[\tau - p]), y_{n'}[t-p]) = \kappa^{(p)}_{n'}\,(y_{n'}[\tau - p]) - y_{n'}[t-p])$; popular kernels such as Gaussian, Laplacian and radial basis function (RBF) are examples

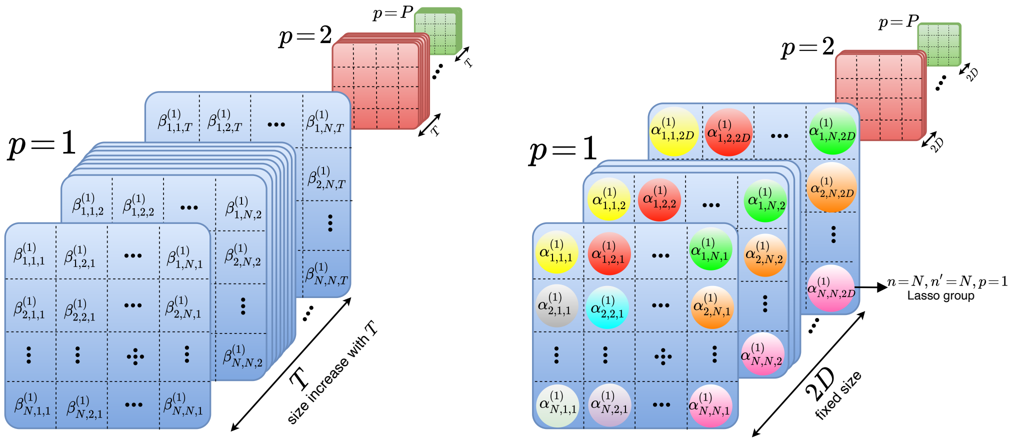


Figure 3.1: RKHS parameters (left) and fixed-size RF parameters (right). The Lasso groups of RF parameters are indicated in different colors.

of such class of kernels. Bochner's theorem [30] states that every shift-invariant kernel can be represented as an inverse Fourier transform of a probability distribution. Hence the kernel evaluation can be expressed as

$$\begin{aligned}\kappa_{n'}^{(p)}\left(y_{n'}[\tau-p]), y_{n'}[t-p]\right) &= \int \pi_{\kappa_{n'}^{(p)}}(v)\; e^{jv(y_{n'}[\tau-p]-y_{n'}[t-p])}dv \\ &= E_v[e^{jv(y_{n'}[\tau-p]-y_{n'}[t-p])}], \qquad (3.21)\end{aligned}$$

where $\pi_{\kappa_{n'}^{(p)}}(v)$ is the probability density function which depends on the type of the kernel, and $v$ is the random variable associated with it. If sufficient amount of i.i.d. samples $\{v_i\}_{i=1}^{D}$ are collected from the distribution $\pi_{\kappa_{n'}^{(p)}}(v)$, the real ensemble mean in (3.21) can be expressed as a sample mean:

$$\hat{\kappa}_{n'}^{(p)}\left(y_{n'}[\tau-p]), y_{n'}[t-p]\right) = \frac{1}{D}\sum_{i=1}^{D} e^{jv_i(y_{n'}[\tau-p])-y_{n'}[t-p])}, \qquad (3.22)$$

irrespective of the distribution $\pi_{\kappa_{n'}^{(p)}}(v)$. Note that an unbiased estimate of kernel evaluation in (3.22) involves a summation of a fixed $D$ number of terms. In general, computing the probability distribution corresponding to a kernel is a difficult task. In this work, the kernel under consideration is assumed to be Gaussian; for a Gaussian kernel $k_\sigma$ with variance $\sigma^2$, it is well known that the Fourier transform is also a Gaussian, with variance $\sigma^{-2}$. Considering the real part of (3.22), which is also an unbiased estimator, we can approximate (3.21) as

$$\hat{\kappa}_{n'}^{(p)}\left(y_{n'}[\tau-p], y_{n'}[t-p]\right) = \boldsymbol{z_v}\left(y_{n'}[\tau-p]\right)^\top \boldsymbol{z_v}\left(y_{n'}[t-p]\right), \qquad (3.23)$$

$$\text{where, } \boldsymbol{z_v}(x) = \frac{1}{\sqrt{D}}[\sin v_1 x, \ldots, \sin v_D x, \cos v_1 x, \ldots, \cos v_D x]^\top. \qquad (3.24)$$

Subsisting (3.24) in (3.4), we obtain a fixed dimension ($2D$ terms) approximation of the function $\hat{f}_{n,n'}^{(p)}$:

$$\begin{aligned}\hat{f}_{n,n'}^{(p)}\left(y_{n'}[\tau-p])\right) &= \sum_{t=p}^{p+T-1} \beta_{n,n',(t-p)}^{(p)} \boldsymbol{z_v}\left(y_{n'}[\tau-p]\right)^\top \boldsymbol{z_v}\left(y_{n'}[t-p]\right) \\ &= \boldsymbol{\alpha}_{n,n'}^{(p)\;\top} \boldsymbol{z_v}\left(y_{n'}[\tau-p]\right), \qquad (3.25)\end{aligned}$$

where $\boldsymbol{\alpha}_{n,n'}^{(p)}{}^{\top} = \sum_{t=p}^{p+T-1} \beta_{n,n',(t-p)}^{(p)} \boldsymbol{z_v} \left(y_{n'}[\tau - p]\right))^{\top}$. For the sake of brevity, in the succeeding steps, we define the following notation:

$$\boldsymbol{\alpha}_{n,n'}^{(p)} = [\alpha_{n,n',1}^{(p)}, \ldots, \alpha_{n,n',2D}^{(p)}]^{\top} \in \mathbb{R}^{2D}, \tag{3.26}$$

$$\boldsymbol{z_v} \left(y_{n'}[\tau - p]\right) = [z_{n',1}^{(p)} (\tau), \ldots z_{n',2D}^{(p)} (\tau)]^{\top} \in \mathbb{R}^{2D}. \tag{3.27}$$

The loss function (3.3) is reformulated as a parametric optimization problem using (3.25):

$$\left\{\widehat{\alpha}_{n,n',d}^{(p)}\right\}_{n',p,d} = \arg \min_{\left\{\alpha_{n,n',d}^{(p)}\right\}} \mathcal{L}^n \left(\alpha_{n,n',d}^{(p)}\right), \tag{3.28}$$

where

$$\mathcal{L}^n \left(\alpha_{n,n',d}^{(p)}\right) := \sum_{\tau=P}^{T-1} \frac{1}{2} \left[y_n[\tau] - \sum_{n'=1}^{N} \sum_{p=1}^{P} \sum_{d=1}^{2D} \alpha_{n,n',d}^{(p)} \; z_{n',d}^{(p)} (\tau)\right]^2. \tag{3.29}$$

For convenience, optimization parameters $\left\{\alpha_{n,n',d}^{(p)}\right\}$ and $\left\{z_{n',d}^{(p)} (\tau)\right\}$ are stacked in the lexicographic order of the indices $p$, $n'$, and $d$ to obtain the vectors $\boldsymbol{\alpha}_n \in \mathbb{R}^{2PND}$ and $\boldsymbol{z}_{\tau} \in \mathbb{R}^{2PND}$, respectively, and (3.33) can be rewritten as

$$\widehat{\boldsymbol{\alpha}}_n = \arg \min_{\boldsymbol{\alpha}_n} \mathcal{L}^n \left(\boldsymbol{\alpha}_n\right), \tag{3.30}$$

$$\text{where} \;\; \mathcal{L}^n(\boldsymbol{\alpha}_n) = \frac{1}{2} \sum_{\tau=P}^{T-1} \left[y_n[\tau] - \boldsymbol{\alpha}_n^{\top} \boldsymbol{z}_{\tau}\right]^2 \tag{3.31}$$

Following [43], the original regularization term in (3.3) can be converted to an equivalent parametric form as:

$$\begin{aligned} &\Omega(||f_{n,n'}^{(p)}||_{\mathcal{H}_{n,n'}^{(p)}}) \\ &= \Omega\left(\sqrt{\sum_{\tau=p}^{p+T-1} \sum_{t=p}^{p+T-1} \hat{\beta}_{n,n',(\tau-p)}^{(p)} \; \hat{\beta}_{n,n',(t-p)}^{(p)} \; k_{n'}^{(p)} (y_n(\tau), y_n(t))}\right) \\ &= \Omega\left(\sqrt{\sum_{\tau=p}^{p+T-1} \sum_{t=p}^{p+T-1} \hat{\beta}_{n,n',(\tau-p)}^{(p)} \hat{\beta}_{n,n',(t-p)}^{(p)} \boldsymbol{z}_{\boldsymbol{v},n'}^{(p)}(\tau)^{\top} \boldsymbol{z}_{\boldsymbol{v},n'}^{(p)}(t)}\right). \\ &= \Omega(||\boldsymbol{\alpha}_{n,n'}^{(p)}||_2). \end{aligned} \tag{3.32}$$

The function $\Omega$ in (3.32) is chosen to be $\Omega(.) = |.|$, where $|.|$ represents the absolute value function, in order to promote the group sparsity of $\boldsymbol{\alpha}_{n,n'}^{(p)}$ [11]. Such regularizers are typically known as *group-Lasso regularizers* (see, Fig. 3.1 for a visual representation of the Lasso groups). Note that the function $|.|$ is non-decreasing, thereby satisfying the regularization criteria to apply the Representer Theorem. Using (3.3) and (3.32), a parametric form of (3.3) can be constructed as follows:

$$\{\widehat{\boldsymbol{\alpha}}_n\}_{n'} = \arg \min_{\{\boldsymbol{\alpha}_n\}} \mathcal{L}^n \left(\boldsymbol{\alpha}_n\right) + \lambda \sum_{n'=1}^{N} \sum_{p=1}^{P} ||\boldsymbol{\alpha}_{n,n'}^{(p)}||_2. \tag{3.33}$$

However, notice that the batch formulation in (3.30) has some major limitations: **i)** requirement of entire batch of data points before estimation, **ii)** inability to track time-varying topologies, and **iii)** high computational complexity when $T$ is large, even if RF approximation is used. To mitigate these problems, we adopt an online optimization strategy, which is explained in the following section.
In this case, we replace the batch loss function $\mathcal{L}^n(\boldsymbol{\alpha}_n)$ in (3.30) with the stochastic (instantaneous) loss function $l_t^n(\boldsymbol{\alpha}_n) = \frac{1}{2}[y_n[t] - \boldsymbol{\alpha}_n^\top \boldsymbol{z}_t]^2$:

$$\widehat{\boldsymbol{\alpha}}_n = \arg\min_{\boldsymbol{\alpha}_n} l_t^n\left(\boldsymbol{\alpha}_n\right) + \lambda \sum_{n'=1}^{N} \sum_{p=1}^{P} \|\boldsymbol{\alpha}_{n,n'}^{(p)}\|_2. \tag{3.34}$$

Now we have a differentiable loss function and non-differentiable regularizer as discussed in the Section 3.4; we can solve such a problem using COMID, and closed form solution is obtained as:

$$\begin{aligned}\boldsymbol{\alpha}_{n,n'}^{(p)}[t+1] = &\left(\boldsymbol{\alpha}_{n,n'}^{(p)}[t] - a_t \mathbf{v}_{n,n'}^{(p)}[t]\right) \times \\ &\left[1 - \frac{a_t \lambda}{\|\boldsymbol{\alpha}_{n,n'}^{(p)}[t] - a_t \mathbf{v}_{n,n'}^{(p)}[t]\|_2}\right]_+,\end{aligned} \tag{3.35}$$

where $[\mathbf{v}_{n,n'}^{(1)}{}^\top, \mathbf{v}_{n,n'}^{(2)}{}^\top, \dots, \mathbf{v}_{n,n'}^{(P)}{}^\top]^\top \triangleq \mathbf{v}_{n,n'}\ \forall n'$, $[\mathbf{v}_{n,1}^\top, \mathbf{v}_{n,2}^\top, \dots, \mathbf{v}_{n,N}^\top]^\top \triangleq \nabla_{\boldsymbol{\alpha}} \tilde{\ell}_t^n(\boldsymbol{\alpha}_n[t])$ and $[x]_+ = \max\{0, x\}$. The required time-lagged dependencies are encoded in $\left\{\boldsymbol{\alpha}_{n,n'}^{(p)}[t+1]\right\}$; and the proposed RFNL-TISO is summarized in **Algorithm 2**. In the next section, we propose a new algorithm called RFNL-TIRSO, which addresses the instability of the LMS loss function in the presence of input noise. Unlike the NLTISO and RF-NLTISO algorithms, RFNL-TIRSO is designed to be robust to input noise. Additionally, we provide a dynamic regret analysis for RFNL-TIRSO, which guarantees its ability to track changes in the input.

**Algorithm 2:** RF-NLTISO Algorithm

```
Result: $\boldsymbol{\alpha}_{n,n'}^{(p)}, for\ n, n' = 1,..,N$ and $p = 1,..,P$
Store $\{\boldsymbol{y}_n[t]\}_{t=1}^{P}$,
Initialize $\lambda$, $a_t$, $D$ (heuristically chosen) and kernel parameters depending
 on the type of the kernel.
for $t = P, P+1, \dots$ do
    Get data samples $y_n[t]$, $\forall n$ and compute $\boldsymbol{z}_\tau$
    for $n = 1, \dots, N$ do
        compute $\nabla_{\boldsymbol{\alpha}} \tilde{\ell}_t^n(\boldsymbol{\alpha}_n[t])$
        for $n' = 1, \dots, N$ do
            compute $\boldsymbol{\alpha}_{n,n'}^{(p)}[t+1]$ using (3.35)
        end
    end
end
```

## 3.6 RFNL-TIRSO

The loss function $l_t^n(\boldsymbol{\alpha}_n) = \frac{1}{2}[y_n[t] - \boldsymbol{\alpha}_n^\top \boldsymbol{z}_t]^2$ introduced in the previous section is analogues to least mean squares (LMS) formulation. The algorithms in such structure suffer in practical settings due to instability produced by noise and dependence of convergence on condition number [44]. We modify the objective function by using the recursive least squares (RLS) principle,

$$\tilde{\ell}_t^n(\boldsymbol{\alpha}_n) = \mu \sum_{\tau=P}^{t} \gamma^{t-\tau} \ell_\tau^n(\boldsymbol{\alpha}_n). \tag{3.36}$$

Here, the instantaneous loss is replaced with a running average loss using an exponential window. The parameter $\gamma \in (0,1)$ is a forgetting factor, and $\mu = 1-\gamma$ is set to normalize the exponential weighting window. Expanding the function as follows:

$$\begin{aligned}\tilde{\ell}_t^n(\boldsymbol{\alpha}_n) =& \mu \sum_{\tau=P}^{t} \gamma^{t-\tau}(y_n^2[\tau] + \boldsymbol{\alpha}_n^\top \boldsymbol{z}_\tau \boldsymbol{z}_\tau^\top \boldsymbol{\alpha}_n - 2y_n[\tau]\boldsymbol{z}_\tau^\top \boldsymbol{\alpha}_n) & (3.37)\\ =& \frac{1}{2}\mu \sum_{\tau=P}^{t} \gamma^{t-\tau} y_n^2[\tau] + \boldsymbol{\alpha}_n^\top \boldsymbol{\phi}[t]\boldsymbol{\alpha}_n - \boldsymbol{r}_n^\top \boldsymbol{\alpha}_n 2y_n[\tau]\boldsymbol{z}_\tau^\top & (3.38)\end{aligned}$$

where

$$\boldsymbol{\phi}[t] = \mu \sum_{\tau=P}^{t} \gamma^{t-\tau} \boldsymbol{z}_\tau \boldsymbol{z}_\tau^\top \tag{3.39}$$

$$\boldsymbol{r}_n[t] = \mu \sum_{\tau=P}^{t} \gamma^{t-\tau} 2y_n[\tau]\boldsymbol{z}_\tau \tag{3.40}$$

As in RLS, these quantities can be updated recursively as $\boldsymbol{\phi}[t] = \gamma\boldsymbol{\phi}[t-1] + \mu \boldsymbol{z}_t \boldsymbol{z}_t^\top$ and $\boldsymbol{r}_n[t] = \gamma \boldsymbol{r}_n[t-1] + \mu y_n[t]\boldsymbol{z}_t$. The gradient of the loss function can be obtained as,

$$\nabla \tilde{\ell}_t^n(\boldsymbol{\alpha}_n) = \boldsymbol{\phi}[t]\boldsymbol{\alpha}_n - \boldsymbol{r}_n[t]. \tag{3.41}$$

The new objective function in the RLS analogous form can be expressed as,

$$\widehat{\boldsymbol{\alpha}}_n = \arg\min_{\boldsymbol{\alpha}_n} \tilde{\ell}_t^n(\boldsymbol{\alpha}_n) + \lambda \sum_{n'=1}^{N} \sum_{p=1}^{P} \|\boldsymbol{\alpha}_{n,n'}^{(p)}\|_2. \tag{3.42}$$

As the objective function has a differentiable loss function and a nondifferentiable regularizer, a closed-form solution for the required time-lagged dependencies is obtained, using the COMID update as :

$$\begin{aligned}\boldsymbol{\alpha}_{n,n'}^{(p)}[t+1] = &\left(\boldsymbol{\alpha}_{n,n'}^{(p)}[t] - a_t \mathbf{v}_{n,n'}^{(p)}[t]\right) \times\\ &\left[1 - \frac{a_t\lambda}{\|\boldsymbol{\alpha}_{n,n'}^{(p)}[t] - a_t \mathbf{v}_{n,n'}^{(p)}[t]\|_2}\right]_+,\end{aligned} \tag{3.43}$$

where $[\mathbf{v}_{n,n'}^{(1)\top}, \mathbf{v}_{n,n'}^{(2)\top}, \dots, \mathbf{v}_{n,n'}^{(P)\top}]^\top \triangleq \mathbf{v}_{n,n'}\ \forall n'$, $[\mathbf{v}_{n,1}^\top, \mathbf{v}_{n,2}^\top, \dots, \mathbf{v}_{n,N}^\top]^\top \triangleq \nabla \tilde{\ell}_t^n(\boldsymbol{\alpha}_n[t])$ and $[x]_+ = \max\{0, x\}$. The first term $\boldsymbol{\alpha}_{n,n'}^{(p)}[t] - a_t \mathbf{v}_{n,n'}^{(p)}[t]$ in (3.43) forces the stochastic gradient update of $\boldsymbol{\alpha}_{n,n'}^{(p)}$ in a way to descend the recursive loss function $\tilde{\ell}_t^n(\boldsymbol{\alpha_n})$, and the second term enforces group sparsity of $\boldsymbol{\alpha}_{n,n'}^{(p)}$. The required causal influence of $p-th$ time-lagged value of $n'-th$ sensor on $n-th$ sensor is encoded in $\boldsymbol{\alpha}_{n,n'}^{(p)}$. The proposed RFNL-TIRSO is summarized as **Algorithm 3**.

**Algorithm 3:** RFNL-TIRSO Algorithm

```
Result: $\left\{\boldsymbol{\alpha}_{n,n'}^{(p)}\right\}_{n,n',p}$
Store $\{\boldsymbol{y}_n[t]\}_{t=1}^{P}$,
Initialize $\lambda > 0$, $a_t > 0$, $\theta > 0$, $D$, $\sigma_n$ and $\boldsymbol{\Phi}(P-1) = \theta \boldsymbol{I}_{2PND}$
for $t = P, P+1, \dots$ do
    Collect data samples $y_n[t]$, $\forall n$ and compute $\boldsymbol{z_v}(t)$
    $\boldsymbol{\Phi}[t] = \gamma \boldsymbol{\Phi}[t-1] + \mu \boldsymbol{z_v}(t) \boldsymbol{z_v}(t)^\top$
    for $n = 1, \dots, N$ do
        $\boldsymbol{r}_n[t] = \gamma \boldsymbol{r}_n[t-1] + \mu y_n[t] \boldsymbol{z_v}(t)$
        compute $\nabla \tilde{\ell}_t^n(\boldsymbol{\alpha}_n[t])$ using (3.41)
        for $n' = 1, \dots, N$ do
            compute $\boldsymbol{\alpha}_{n,n'}^{(p)}[t+1]$ using (3.43)
        end
    end
end
```

## 3.7 Numerical Experiments

We provide a detailed numerical analysis using both synthetic and real data. The synthetic data sets that we consider are generated by considering nonlinear topologies having challenging dynamic nature. The real data set include data collected from an O&G platform as well as data from the EEG recordings of two pediatric subjects with intractable seizure. Through a series of numerical experiments, we show that our algorithms estimate interpretable topologies as well as outperform the state-of-the-art benchmarks (see, papers [21, 22, 36]).

## 3.8 Dynamic Regret

The dynamic regret bound for the proposed algorithm RFNL-TIRSO is based on the following assumptions

- $\boldsymbol{A1}$ Bounded samples: For all the time series samples, there exists $B_y > 0$ such that $\{|y_n[t]|^2\} \leq B_y \leq \infty, \forall n, t$.

- $\boldsymbol{A2}$ Shift-invariant kernels: kernels used are shift-invariant, i.e., $k(x_i, x_j) = k(x_i - x_j)$.

- $\boldsymbol{A3}$ Bounded minimum eigenvalue of $\boldsymbol{\Phi}[t]$: There exists $\rho_l > 0$ such that $\Lambda_{min}(\boldsymbol{\Phi}[t]) > \rho_l$, where $\Lambda_{min}(.)$ denotes the minimum eigenvalue.

- $\boldsymbol{A4}$ Bounded maximum eigenvalue of $\boldsymbol{\Phi}[t]$: There exists $L > 0$ such that $\Lambda_{max}(\boldsymbol{\Phi}[t]) < L < \infty$, where $\Lambda_{max}(.)$ denotes the maximum eigenvalue.

$\boldsymbol{A1}$ is generally true since real-world signals are mostly bounded. Kernels such as Gaussian, Laplacian, etc. satisfy the property $\boldsymbol{A2}$. $\boldsymbol{A3}$ will hold as long as the feature vectors are linearly independent and the condition is typically satisfied in practice. Note that $\boldsymbol{A3}$ is important for the strong convexity of the loss function, which is used in the derivation of regret bound. $\boldsymbol{A4}$ can be obtained by combining $\boldsymbol{A1}$ and the fact that the sum of eigenvalues of $\boldsymbol{\Phi}[t]$ is equal to its trace.

Dynamic regret is a popular way to test the capability of an online algorithm in a dynamic environment. Dynamic regret is defined as the cumulative sum of the difference between the estimated loss function and optimal loss function at each time instant(see (2.8), in Section 2.3.1). Dynamic regret of RFNL-TIRSO at time $T$ is bounded as

$$\boldsymbol{R}_n(T) \leq \Big(\Big(1 + \frac{L}{\rho_l}\Big)\sqrt{2PNDB_y} + \lambda\sqrt{PN}\Big) \times \Big(\|\boldsymbol{\alpha}_n^*[P]\|_2 + \boldsymbol{W}_n(T)\Big) + \epsilon L_h TC, \tag{3.44}$$

where $\boldsymbol{W}_n(T) = \sum_{t=P}^{T-1} \|\boldsymbol{\alpha}_n^*[t] - \boldsymbol{\alpha}_n^*[t-1]\|_2$ is the path length, $L_h$ is the Lipschitz constant and $\boldsymbol{\alpha}_n^*[P]$ is the optimal solution at time $P$ (Proof: see paper [36]). Therefore, it is guaranteed to achieve sub-linear dynamic regret by suitably choosing $\epsilon$ as long as $\boldsymbol{W}_T^n$ is sub-linear.

## 3.9 Chapter Summary

- This chapter proposes online algorithms to estimate the time-varying nonlinear topology from streaming multi-variate time series. We use a VAR model equipped with kernels to model the nonlinear spatio-temporal interaction among the time series. We propose a group Lasso-based online optimization framework to learn sparse model parameters, which is solved efficiently using the COMID algorithm.

- We design three successive algorithms, namely, NL-TISO, RFNL-TISO, and RFNL-TIRSO, which take into account important elements of online learning. NL-TISO deploys a forgetting window to mitigate kernel's dimensionality growth in online learning. The suboptimality associated with the forgetting window based approach is effectively addressed in RFNL-TISO by using RF approximation, which is further improved in RFNL-TIRSO by using an RLS loss function having better robustness to input noise.

- We provide a detailed theoretical analysis of the convergence of RFNL-TIRSO algorithm. Our analysis derives a sublinear upper bound for the dynamic regret of the algorithm under certain reasonable assumptions.

# Chapter 4

# Online Joint Topology Identification and Missing Data Imputation

This chapter summarizes Paper D ([24])

## 4.1 Motivation

In the previous chapter, we discussed topology identification when all the time series are fully observed. There are often times when the signals cannot be observed in their entirety. In this chapter, we discuss joint topology identification and missing data imputation. The proposed algorithm estimates topology from incomplete observations and uses the learned topology to impute the missing entries.

## 4.2 Problem Formulation

Let the multivariate time series $\{y_n[t]\}_{n=1}^{N}$ be generated from a $P$-th order nonlinear VAR model:

$$y_n[t] = \sum_{n'=1}^{N} \sum_{p=1}^{P} f_{n,n'}^{(p)}(y_{n'}[t-p]) + u_n[t], \tag{4.1}$$

where $f_{n,n'}^{(P)}(.)$ captures the influence of the $p$-th time-lagged value of the $n'$-th time series on the $n$-th time series, we represent the collection of time series at a time in a vector form as $\mathbf{y}[t] = [y_1[t], y_2[t], \dots, y_N[t]]^\top \in \mathbb{R}^N$. Unlike Chapter 3, now the full signal vector $\mathbf{y}[t]$ is not always observable. The observed signal at time $t$ can be expressed in a vector form as

$$\tilde{\mathbf{y}}[t] = \mathbf{m}[t] \odot (\mathbf{y}[t] + \mathbf{e}[t]), \tag{4.2}$$

where $\mathbf{m}[t] \in \{0, 1\}$ is a known masking vector, where the $n$-th element $m_n[t]$ is 0 only if the 88value of $n$-th series is missing at time $t$, $e[t]$ is the observation noise and $\odot$ is the Hadamard product [1]. In such a situation, the learned graph topology

[1] The Hadamard product is a binary operation between two matrices, $\mathbf{A}$ and $\mathbf{B}$, that have the same dimension. It is denoted by the symbol $\odot$, and is defined such that each element of the

can be exploited to impute the missing data. An illustrative example is shown in Fig. 4.1. At the time $t$, we have the masked observation vector $\tilde{\mathbf{y}}[t]$ and predicted graph topology. An estimate of the actual signal vector $\mathbf{y}[t]$ is constructed using the observation vector and predicted topology. The estimated signal vector $\hat{\mathbf{y}}[t]$ is then used to update the graph topology.

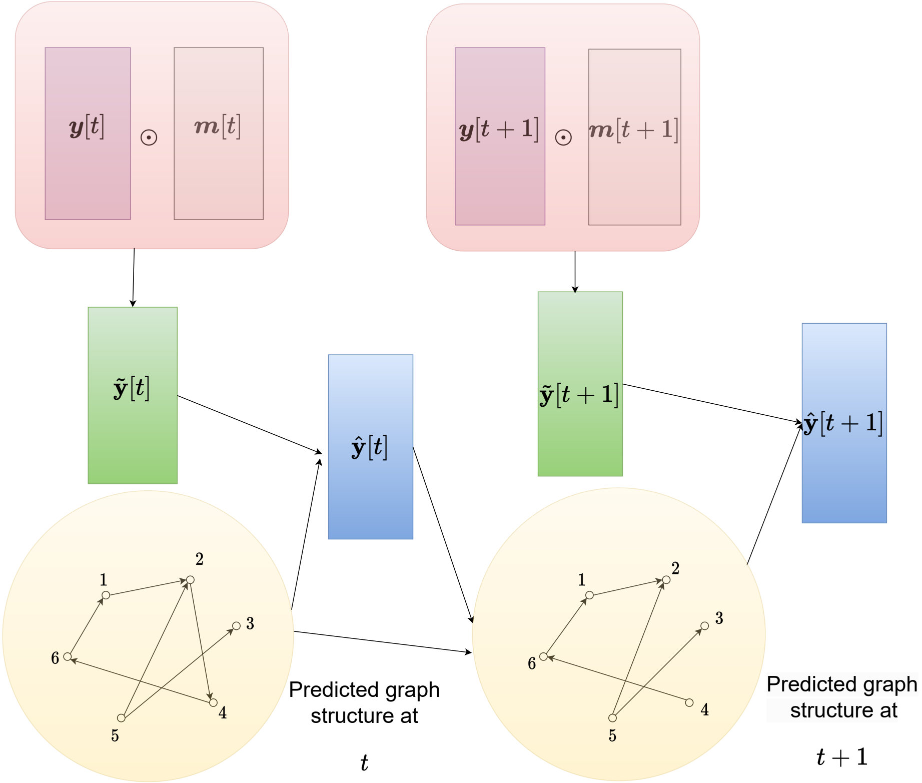


Figure 4.1: Illustration of proposed method.

Revisiting (3.25) the function $f^{p}_{n,n'}(.)$ can be approximated in parametric form using RF as $\hat{f}^{(p)}_{n,n'}(.) = \boldsymbol{\alpha}^{(p)\top}_{n,n'} \boldsymbol{z_v}(.)$, where $\boldsymbol{\alpha}^{(p)}_{n,n'}$ is the parameter vector and $\boldsymbol{z}_v(.)$ the data-dependent features (3.27). The parameter vectors $\{\boldsymbol{\alpha}^{(p)}_{n,n'}\}$ are stacked in lexicographic order of $n$, $p$, $n'$ obtaining the vector $\boldsymbol{\alpha}$, and an estimate of the signal vector is obtained :

$$\hat{\hat{\mathbf{y}}}[t] = \boldsymbol{\alpha}^{\top}\boldsymbol{z}_v[t]. \tag{4.3}$$

Now, we have an estimate of the signal based on the model and an observation vector. Combining this information, an online joint topology identification and missing data imputation problem can be formulated as

$$\begin{aligned}\{\hat{\boldsymbol{\alpha}}, \hat{\mathbf{y}}[\tau]\}_{\tau=P}^{\tau=T-1} = \arg\min_{\boldsymbol{\alpha},\mathbf{y}[\tau]} \sum_{\tau=P}^{T-1} \frac{1}{2}\|\mathbf{y}[\tau] - \boldsymbol{\alpha}^{\top}\boldsymbol{z}_v[\tau]\|_2^2 \\ + \lambda \sum_{n'=1}^{N}\sum_{d=1}^{2D}\|\boldsymbol{\alpha}_{n,n',d}\|_2 + \sum_{\tau=P}^{T-1}\frac{\nu}{2M_\tau}\|\tilde{\mathbf{y}}[\tau] - \mathbf{m}[\tau]\odot\mathbf{y}[\tau]\|_2^2.\end{aligned} \tag{4.4}$$

resulting matrix $(\mathbf{A}\odot\mathbf{B})_{ij}$ is equal to the product of the corresponding elements of $\mathbf{A}$ and $\mathbf{B}$, i.e., $(\mathbf{A}\odot\mathbf{B})_{ij} = \mathbf{A}_{ij}\mathbf{B}_{ij}$.

The optimization problem (4.4) consists of three terms *(i)* the topology identification part which fits the parametric model with data *(ii)* sparsity promoting regularizer as most of the real-world networks are sparse *(iii)* missing signal reconstruction term based on observation. The optimization problem (4.4) is nonconvex and computationally difficult to solve. In the next section, we introduce approximations to convexify the problem and drive an online solution.

## 4.3 Online Joint Topology Identification and Signal Reconstruction

The feature vector $\boldsymbol{z}_\nu[t]$ depends on the $P$ previous values of all the $N$ time series. Therefore, it is necessary to estimate the $P$ previous values along with the instantaneous values in the required online estimation strategy:

$$\{\widehat{\boldsymbol{\alpha}}, \hat{\mathbf{y}}[t], \{\hat{\mathbf{y}}[\tau]\}_{\tau=t-P}^{t-1}\} = \underset{\substack{\boldsymbol{\alpha}, \mathbf{y}[t] \\ \{\mathbf{y}[\tau]\}_{\tau=t-P}^{t-1}}}{\arg\min} \ell_t\left(\boldsymbol{\alpha}, \mathbf{y}[t], \{\mathbf{y}[\tau]\}_{\tau=t-P}^{t-1}\right) + \lambda \sum_{n'=1}^{N} \sum_{p=1}^{P} \|\boldsymbol{\alpha}_{n,n'}^{(p)}\|_2, \tag{4.5}$$

where the loss function $\ell_t(.)$ is defined as

$$\ell_t\left(\boldsymbol{\alpha}, \mathbf{y}[t], \{\mathbf{y}[\tau]\}_{\tau=t-P}^{t-1}\right) = \frac{1}{2}\|\mathbf{y}[t] - \boldsymbol{\alpha}^\top \boldsymbol{z}_v[t]\|_2^2 + \frac{\nu}{2M_t}\|\tilde{\mathbf{y}}[t] - \mathbf{m}[t] \odot \mathbf{y}[t]\|_2^2. \tag{4.6}$$

The optimization problem (4.5) is nonconvex as well as computationally expensive. In order to have a computationally light solution, we can convexify the optimization problem by assuming that $y[t]$ is independent of $\{\hat{y}\}_{\tau=t-P}^{t-1}$. Now, we can formulate the loss function without the arguments $\{\mathbf{y}[\tau]\}_{\tau=t-P}^{t-1}$:

$$\tilde{\ell}_t\left(\boldsymbol{\alpha}, \mathbf{y}[t]\right) = \frac{1}{2}\|\mathbf{y}[t] - \boldsymbol{\alpha}^\top \boldsymbol{z}_v[t]\|_2^2 + \frac{\nu}{2M_t}\|\tilde{\mathbf{y}}[t] - \mathbf{m}[t] \odot \mathbf{y}[t]\|_2^2. \tag{4.7}$$

The optimization problem (4.7) is separable with respect to $\boldsymbol{\alpha}$ and $\mathbf{y}[t]$, leading to subproblems that are convex in the respective variables, which can be solved using a two-step iterative block coordinate descent method. These two steps *(i)* signal reconstruction and *(ii)* topology identification, are explained below.

### 4.3.1 Signal Reconstruction

Assume that the estimates of dependency vectors $\{\boldsymbol{\alpha}_n[t]\}_{n=1}^N$ are available and substitute $\boldsymbol{\alpha}[t] = [\boldsymbol{\alpha}_1[t], \ldots, \boldsymbol{\alpha}_n[t]]^\top$ in place of $\boldsymbol{\alpha}$ in (4.7). Then, the estimate of the signal vector is obtained by solving the following optimization problem:

$$\hat{\mathbf{y}}[t] = \underset{\mathbf{y}[t]}{\arg\min}\ \frac{1}{2}\|\mathbf{y}[t] - \boldsymbol{\alpha}[t]^\top \boldsymbol{z}_v[t]\|_2^2 + \frac{\nu}{2M_t}\|\tilde{\mathbf{y}}[t] - \mathbf{m}[t] \odot \mathbf{y}[t]\|_2^2. \tag{4.8}$$

Note that the regularization term is not included in the formulation as it is independent of $\mathbf{y}[t]$. The optimization problem (4.8) can be separated into $n$ quadratic problems, each one solved for each node with respect to $y_n[t]$, $1 \leq n \leq N$:

$$\hat{y}_n[t] = \arg\min_{y_n[t]} \ell_t^n \left(y_n[t]\right), \tag{4.9}$$

where $\ell_t^n \left(y_n[t]\right) = \frac{1}{2}\Bigg[y_n[t] - \boldsymbol{\alpha}_n[t]^\top \boldsymbol{z}_v[t]\Bigg]^2 + \frac{\nu}{2M_t}(\tilde{y}_n[t] - m_n[t]y_n[t])^2$. As the optimization problem is quadratic, a closed-form solution can be readily obtained as

$$\hat{y}_n[t] = \frac{\nu m_n[t]\tilde{y}_n[t]}{M_t + \nu m_n[t]} + \frac{k_n[t]M_t}{\nu m_n[t] + M_t}, \tag{4.10}$$

where $k_n[t] = \boldsymbol{\alpha}_n[t]^\top \boldsymbol{z}_v[t]$. Let $\frac{\nu m_n[t]}{M_t + \nu m_n[t]} = q_n[t]$, then,

$$\hat{y}_n[t] = q_n[t]\tilde{y}_n[t] + [1 - q_n[t]]k_n[t]. \tag{4.11}$$

Now the reconstructed signals $\{\hat{y}_n[\tau]\}_{n=1}^N$ can be used to estimate parameter vectors $\{\boldsymbol{\alpha}_n[t+1]\}_{n=1}^N$.

### 4.3.2 Topology Identification

The estimates $\{\hat{y}_n[\tau]\}_{n=1}^N$ are substituted in (4.7) and then, the topology identification problem can be formulated as

$$\boldsymbol{\alpha}[t] = \arg\min_{\boldsymbol{\alpha}} \frac{1}{2}\|\hat{\mathbf{y}}[t] - \boldsymbol{\alpha}^\top \boldsymbol{z}_v[t]\|_2^2 + \lambda \sum_{n'=1}^{N} \sum_{p=1}^{P} \|\boldsymbol{\alpha}_{n,n'}^{(p)}\|_2. \tag{4.12}$$

Here, again, the optimization problem is node separable and the dependency vector for a particular node $n$ can be obtained by solving the optimization problem:

$$\widehat{\boldsymbol{\alpha}}_n = \arg \min_{\boldsymbol{\alpha}_n} \tilde{\ell}_t^n \left(\boldsymbol{\alpha}_n\right) + \lambda \sum_{n'=1}^{N} \sum_{p=1}^{P} \|\boldsymbol{\alpha}_{n,n'}^{(p)}\|_2. \tag{4.13}$$

where $\tilde{\ell}_t^n(\boldsymbol{\alpha}_n) = \frac{1}{2}[\hat{y}_n[t] - \boldsymbol{\alpha}_n^\top \boldsymbol{z}_{\boldsymbol{v}}[t]]^2$. Unlike the signal reconstruction (4.8), the optimization problem (4.13) is not quadratic. The objective function of (4.13) contains a differentiable loss function and a nondifferentiable regularizer, and such problems can be solved efficiently using COMID methods [22]. A closed-form solution for the COMID update is obtained via the multidimensional shrinkage-thresholding operator:

$$\begin{aligned}\boldsymbol{\alpha}_{n,n'}^{(p)}[t+1] = [&\boldsymbol{\alpha}_{n,n'}^{(p)}[t] - \gamma_t \mathbf{v}_{n,n'}^{(p)}[t]]\times \\ &\left[1 - \frac{\gamma_t \lambda}{\|\boldsymbol{\alpha}_{n,n'}^{(p)}[t] - \gamma_t \mathbf{v}_{n,n'}^{(p)}[t]\|_2}\right]_+,\end{aligned} \tag{4.14}$$

where $[\mathbf{v}_{n,n'}^{(1)}{}^\top, \mathbf{v}_{n,n'}^{(2)}{}^\top, \ldots, \mathbf{v}_{n,n'}^{(P)}{}^\top]^\top \triangleq \mathbf{v}_{n,n'}\ \forall n'$, $[\mathbf{v}_{n,1}^\top, \mathbf{v}_{n,2}^\top, \ldots, \mathbf{v}_{n,N}^\top]^\top \triangleq \nabla_{\boldsymbol{\alpha}} \tilde{\ell}_t^n(\boldsymbol{\alpha}_n[t])$. The proposed solution is explained in **Algorithm** 4.

**Algorithm 4:**

```
Result: $\left\{\boldsymbol{\alpha}^{(p)}_{n,n'}[t+1]\right\}_{n,n',p}$, $\hat{\mathbf{y}}[t]$
Initialize $\{\boldsymbol{y}_n[t]\}_{t=1}^{P}$, $\left\{\boldsymbol{\alpha}^{(p)}_{n,n'}[P]\right\}_{n,n',p}$ as all-ones vector, $\lambda$, kernel parameters, $\gamma$, $D$, $\nu$
  (heuristically chosen)
for $t = P, P+1, \ldots$ do
    Get data observation vector $\tilde{\mathbf{y}}_n[t]$ and masking vector $\mathbf{m}[t]$, compute $\boldsymbol{z_v}[t]$ (3.27)
    for $n = 1, \ldots, N$ do
        compute $\hat{y}_n[t]$ using (4.11)
        compute $\mathbf{v}^{(p)}_{n,n'}[t]$
        for $n' = 1, \ldots, N$ do
            compute $\boldsymbol{\alpha}^{(p)}_{n,n'}[t+1]$ using (4.14)
        end
    end
end
```

### 4.3.3 Numerical Experiments

We present numerical analysis using both synthetic and real data sets and show that the proposed algorithm outperforms the state-of-the-art benchmarks (see, paper [24]). The synthetic data sets feature dynamic nonlinear topologies with various missing data patterns, while the real data experiments involve data from Lundin's O&G platform subjected to different missing data scenarios.

## 4.4 Summary of Chapter

- This chapter presents an algorithm for joint nonlinear topology identification and missing signal reconstruction. While the original problem is nonconvex and computationally intensive, we propose a convexified version that can be solved using convex optimization techniques.

- Using the block coordinate decent method, the proposed method iteratively solves topology identification and missing data imputation problems.

# Chapter 5

# Online Data Imputation over Structure-aware Higher order Networks

This chapter summarizes Paper E ([25])

## 5.1 Motivation

As explained in previous chapters, missing data is very common in multivariate time series due to various practical reasons. In this chapter, we focus on missing data imputation when the signal is defined over the edges of the network. Traffic flow in the transportation network, information flow in the brain networks, and water flow in the water network are examples of signals defined on the edges. Imputation of missing flows can be performed by exploiting the prior information from the network structure as well as the data-driven features from the observations. Consider the case of a water distribution network, as shown in Fig. 5.1. The physical structure imposes priors, e.g., flow conservation at the junction of edges. Simplicial complex and algebraic topological tools [45] can be used to incorporate such priors in the formulation of missing data imputation algorithms. Apart from the physical structure, the time series data are also coupled through hidden interactions that are not physically observable. In Fig. 5.1, suppose that the demand in node 7 increases; automatically, water flowing through pipe 9 increases and in order to meet this increased demand, pipe 4 will draw more water from the reservoir. From this illustrative example, it is clear that the flow through pipe 9 has a time-lagged influence on the flow through pipe 4. In this chapter, we introduce an algorithm which utilizes both the physical structure and the time-lagged interaction for imputing the flows.

## 5.2 System Model

Consider a network $\mathcal{G}$ with node set $\mathcal{V}$, and the nodes are physically connected through edge set $\mathcal{E}$. Let $V \triangleq |\mathcal{V}|$ and $E \triangleq |\mathcal{E}|$ represent cardinality of $\mathcal{V}$ and $\mathcal{E}$,

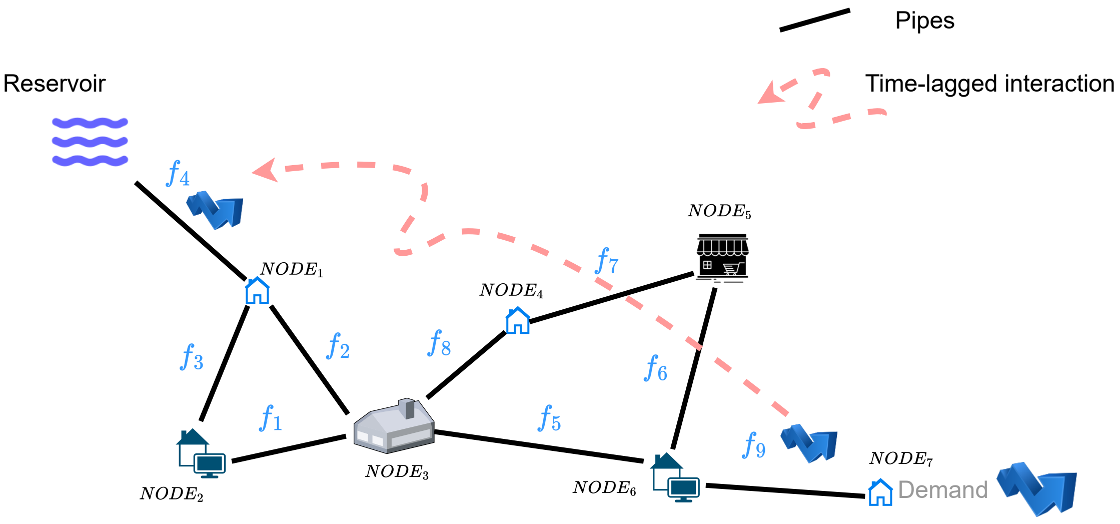


Figure 5.1: Schematic of a water distribution network.

respectively.

### 5.2.1 Model Flow as a Simplicial Signal

The network structure of $\mathcal{G}$ can be expressed using a simplicial complex (SC). In SC, a k-simplicial signal is a mapping from a k-simplex to $\mathbb{R}$. A 0-simplex signal resides on the node; similarly, a 1-simplex signal is on the edge, and a 2-simplex signal is on the triangle Fig. 2.3. In this work, we focus on the signal residing on edges, i.e., the 1-simplex signal. From here onwards, we call the 1-simplex signals flow signals. A flow signal at time $t$ between two nodes $i$ and $j$ is defined as $f_{(i,j)}[t] = -f_{(j,i)}[t], \; \forall \; (i,j) \in \mathcal{E}$. We can stack the flows into a vector $\tilde{\mathbf{f}}[t] = [f_1[t] \;\; f_2[t] \;\; \ldots \;\; f_E[t]]^\top \in \mathbb{R}^E$. The structure of SC can be represented in the form of Hodge Laplacian. First-order Hodge Laplacian is constructed based on incidence matrices $\mathbf{B}_1$ and $\mathbf{B}_2$, which measure proximities between the edges with respect to the nodes and triangles, respectively. The first order Hodge Laplacian is defined as follows

$$\mathbf{L}_1 = \mathbf{B}_1^\top \mathbf{B}_1 + \mathbf{B}_2 \mathbf{B}_2^\top, \tag{5.1}$$

where $\mathbf{B}_1^\top \mathbf{B}_1 = \mathbf{L}_1^\ell$ is termed as first-order lower Laplacian and $\mathbf{B}_2 \mathbf{B}_2^\top = \mathbf{L}_1^u$ is termed as first order upper Laplacian. Flows in networks such as water distribution and road transportation exhibit flow conservation, which can be mathematically expressed using the incidence matrix as $\mathbf{B}_1 \tilde{\mathbf{f}}[t] = \mathbf{0} \in \mathbb{R}^V$ [20]. The first-order lower Laplacian $\mathbf{L}_1^\ell$ can be used to model the flow conservation since it describes the relationship among the edges that incident on a node, which is given by

$$\|\mathbf{B}_1 \tilde{\mathbf{f}}[t]\|_2^2 = \tilde{\mathbf{f}}[t]^\top \mathbf{B}_1^\top \mathbf{B}_1 \tilde{\mathbf{f}}[t] = \tilde{\mathbf{f}}[t]^\top \mathbf{L}_1^\ell \tilde{\mathbf{f}}[t] = 0. \tag{5.2}$$

### 5.2.2 Line Graph Modeling

The simplicial signals are mostly interdependent in real-world systems, and these dependencies are physically unobservable. For example, a traffic block on a road

$'X'$ that is not directly connected to road $'Z'$ can affect vehicles flowing through $'Z'$ in a time-lagged manner. Similarly, the physical equations and pressure differences in the water network allow us to retrieve information about one flow from another. Many such interactions are time-lagged so that a VAR model can fit the process well. If we can learn a line graph that describes the model, it is possible to estimate the missing flows in a better way by combining the information from the learned line graph and available either define or use lower Laplacian. A $P$-th order VAR model with $E$ number of flows can be expressed as,

$$\tilde{\mathbf{f}}[t] = \sum_{p=1}^{P} \left[ \tilde{\mathbf{A}}^{(p)}[t]\tilde{\mathbf{f}}[t-p] + \mathbf{b}^{(p)}[t] \right] + \mathbf{u}[t], \tag{5.3}$$

where $\tilde{\mathbf{A}}^{(p)}[t] \in \mathbb{R}^{E \times E}$ is the weighted adjacency matrix, $\mathbf{u}[t]$ is the process noise and $\mathbf{b}^{(p)}[t] \in \mathbb{R}^{E}$ is the bias component. The model can be compactly written using an augumented matrix $\mathbf{A}^{(p)}[t] = [\tilde{\mathbf{A}}^{(p)}[t] \ \mathbf{b}^{(p)}[t]] \in \mathbb{R}^{E \times E+1}$ and the signal vector $\mathbf{f}[t] = [\tilde{\mathbf{f}}[t]^\top; 1]^\top \in \mathbb{R}^{E+1}$, as,

$$\mathbf{f}[t] = \sum_{p=1}^{P} \mathbf{A}^{(p)}[t]\mathbf{f}[t-p] + \mathbf{u}[t]. \tag{5.4}$$

## 5.3 Problem Formulation

Assume that at a particular time $t$, only a subset of flows is observable. The observed flow vector is $\mathbf{f}_o[t] = \mathbf{M}[t]\mathbf{f}[t] \in \mathbb{R}^{E+1}$, where $\mathbf{M}[t] \in \mathbb{R}^{(E+1)\times(E+1)}$ is a diagonal masking matrix, that is, $\mathbf{M}(n,n)[t] = 0$ if the $n$-$th$ flow is missing and $\mathbf{M}(n,n)[t] = 1$, otherwise. Unlike the previous chapter, in this setting, some flows can be permanently unobserved. The goal is to find, in an online fashion, both a sequence of line graphs $\{\mathbf{A}^{(p)}[t]\}_{p,t}$, representing the causal dependencies between flows, and the original signal $\mathbf{f}[t]$, from the partial observation $\mathbf{f}_o[t]$.

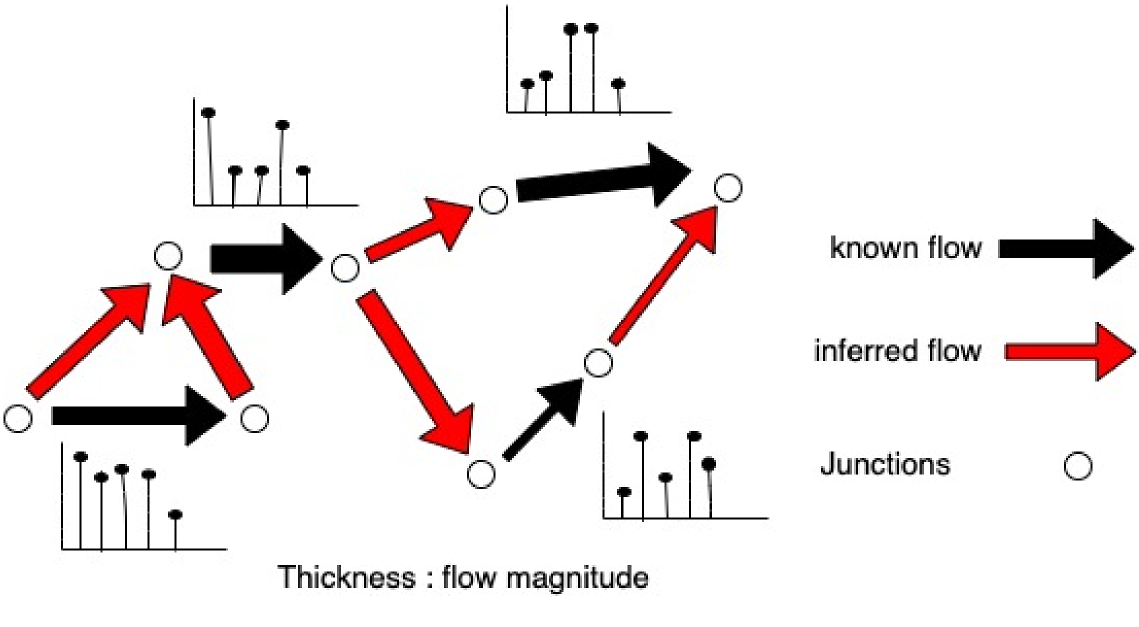


Figure 5.2: The problem under consideration is to infer the missing flows from available observation.

## 5.4 Proposed Solution

A joint direct optimization of $\mathbf{A}^{(p)}[t]$ and $\mathbf{f}[t]$ leads to a nonconvex optimization problem, which is computationally difficult to solve. Hence, in this work, we propose a bi-level optimization problem: *i*) *signal reconstruction*- missing flows are estimated using structure-aware Kalman Filter (KF) based on the observed flows and the learned line graph topology, and *ii*) *line graph identification*- line graph is estimated using the reconstructed signals.

### 5.4.1 Signal Reconstruction

Assume that we have an estimate of $\hat{\mathbf{A}}^{(p)}[t]$, $\forall p$ at time $t$ of the topology and estimates of the previous $P$ flow values $\{\hat{\mathbf{f}}[t-p]\}_{p=1}^{P}$. By rearranging the data as follows, it is possible to model flow in state space form:

$$\hat{\mathbf{A}}^{\mathcal{S}}[t] \triangleq \left[\begin{array}{c|c} \multicolumn{2}{c}{\underbrace{\hat{\mathbf{A}}^{(1:P)}[t]}_{E \times P(E+1)}} \\ \mathbf{I}_{P(E+1)-E} & \underbrace{\mathbf{0}}_{(P(E+1)-E) \times E} \end{array}\right], \ \mathbf{C}^{\mathcal{S}}[t] \triangleq \left[\begin{array}{c|c} \underbrace{\mathbf{M}[t]}_{(E+1) \times (E+1)} & \underbrace{\mathbf{0}}_{(E+1) \times (P-1)(E+1)} \\ \underbrace{\mathbf{0}}_{(P-1)(E+1) \times (E+1)} & \mathbf{I}_{(P-1)(E+1)} \end{array}\right],$$

$$\begin{aligned} \mathbf{y}^{\mathcal{S}}[t] &\triangleq [\mathbf{f}_o[t]^\top; \hat{\mathbf{f}}[t-1:t-P+1]^\top]^\top, \\ \hat{\mathbf{f}}^{\mathcal{S}}[t] &\triangleq [\hat{\mathbf{f}}[t]^\top; \hat{\mathbf{f}}[t-1]^\top; \ldots; \hat{\mathbf{f}}[t-P+1]^\top]^\top, \end{aligned} \tag{5.5}$$

The state space representation of the model is given as

$$\hat{\mathbf{f}}^{\mathcal{S}}[t] = \hat{\mathbf{A}}^{\mathcal{S}}[t]\hat{\mathbf{f}}^{\mathcal{S}}[t-1] + \mathbf{v}_t, \tag{5.6}$$

$$\mathbf{y}^{\mathcal{S}}[t] = \mathbf{C}^{\mathcal{S}}[t]\hat{\mathbf{f}}^{\mathcal{S}}[t] + \mathbf{w}_t, \tag{5.7}$$

where $\hat{\mathbf{f}}^{\mathcal{S}}[t] \in \mathbb{R}^{P(E+1)}$ is the current state vector, $\hat{\mathbf{A}}^{\mathcal{S}}[t] \in \mathbb{R}^{P(E+1) \times P(E+1)}$ is the state transition matrix and $\mathbf{y}^{\mathcal{S}}[t] \in \mathbb{R}^{P(E+1)}$, and $\mathbf{C}^{\mathcal{S}} \in \mathbb{R}^{P(E+1) \times P(E+1)}$ are the observed signal and the observation matrix, respectively. The process noise $\mathbf{v}_t$ and the observation noise $\mathbf{w}_t$ are assumed zero-mean Gaussian. The optimal estimates of $\hat{\mathbf{f}}^{\mathcal{S}}[t]$ can be obtained using a Kalman filter (KF) [33].

**1) Prediction:**

$$\hat{\mathbf{f}}^{\mathcal{S}}_{t|t-1} = \hat{\mathbf{A}}^{\mathcal{S}}[t]\hat{\mathbf{f}}^{\mathcal{S}}_{t-1|t-1}, \tag{5.8}$$

$$\mathbf{P}_{t|t-1} = \hat{\mathbf{A}}^{\mathcal{S}}[t]\mathbf{P}_{t-1|t-1}\hat{\mathbf{A}}^{\mathcal{S}}[t]^\top + \mathbf{Q}_t, \tag{5.9}$$

**2) Update**: The KF update of the state vector can be expressed as a convex optimization problem [46], [47]:

$$\begin{aligned} \underset{\hat{\mathbf{f}}^{\mathcal{S}}_{t|t}, \mathbf{w}_t}{\text{minimize}} \quad & \mathbf{w}_t^\top \mathbf{R}_t^{-1}\mathbf{w}_t + (\hat{\mathbf{f}}^{\mathcal{S}}_{t|t} - \hat{\mathbf{f}}^{\mathcal{S}}_{t|t-1})^\top \mathbf{P}^{-1}_{t|t-1}(\hat{\mathbf{f}}^{\mathcal{S}}_{t|t} - \hat{\mathbf{f}}^{\mathcal{S}}_{t|t-1}), \\ \text{subject to} \quad & \mathbf{y}^{\mathcal{S}}[t] = \mathbf{C}^{\mathcal{S}}[t]\hat{\mathbf{f}}^{\mathcal{S}}_{t|t} + \mathbf{w}_t. \end{aligned} \tag{5.10}$$

Solving (5.10) yields the standard KF update equation.

$$\hat{\mathbf{f}}^{\mathcal{S}}_{t|t} = \hat{\mathbf{f}}^{\mathcal{S}}_{t|t-1} + \mathbf{K}_t(\mathbf{y}^{\mathcal{S}}[t] - \mathbf{C}^{\mathcal{S}}[t]\hat{\mathbf{f}}^{\mathcal{S}}_{t|t-1}). \tag{5.11}$$

The covariance matrix can be updated as

$$\mathbf{P}_{t|t} = \mathbf{P}_{t|t-1} - \mathbf{K}_t\mathbf{C}^{\mathcal{S}}[t]\mathbf{P}_{t|t-1}. \tag{5.12}$$

**3) Flow-conservation update:** The KF update problem (5.10), penalized with the flow conservation (5.2), can be written as

$$\begin{aligned}
\underset{\hat{\mathbf{f}}^{\mathcal{S}}_{t|t}, \mathbf{w}_t}{\text{minimize}} \quad & \mathbf{w}_t^\top \mathbf{R}_t{}^{-1}\mathbf{w}_t + (\hat{\mathbf{f}}^{\mathcal{S}}_{t|t} - \hat{\mathbf{f}}^{\mathcal{S}}_{t|t-1})^\top \mathbf{P}^{-1}_{t|t-1}(\hat{\mathbf{f}}^{\mathcal{S}}_{t|t} - \hat{\mathbf{f}}^{\mathcal{S}}_{t|t-1}) \\
& \qquad\qquad + \mu\hat{\mathbf{f}}^{\mathcal{S}}_{t|t}[t]^\top \mathbf{L}\hat{\mathbf{f}}^{\mathcal{S}}_{t|t}[t], \\
\text{subject to} \quad & \mathbf{y}^{\mathcal{S}}[t] = \mathbf{C}^{\mathcal{S}}[t]\hat{\mathbf{f}}^{\mathcal{S}}_{t|t} + \mathbf{w}_t,
\end{aligned} \tag{5.13}$$

where

$$\mathbf{L} = \begin{bmatrix} \tilde{\mathbf{L}}^{\ell}_1 & \mathbf{0}_{(E+1)\times(P-1)(E+1)} \\ \mathbf{0}_{(P-1)(E+1)\times(E+1)} & \mathbf{0}_{(P-1)(E+1)\times(P-1)(E+1)} \end{bmatrix},$$

with $\tilde{\mathbf{L}}^{\ell}_1$=$[\mathbf{L}^{\ell}_1 \ \mathbf{0}_E; \mathbf{0}^\top_E \ 0] \in \mathbb{R}^{(E+1)\times(E+1)}$, the Laplacian $\mathbf{L}^{\ell}_1$ padded with zero vector $\mathbf{0}_E \in \mathbb{R}^E$ to nullify the bias component in $\mathbf{f}[t]$ and $\mu$ is a hyperparameter. The optimization problem (5.13) is a convex quadratic optimization problem that yields flow-conservation-based Kalman updates. We adopt a similar strategy as followed in [47] to obtain a closed-form solution. We first reformulate the problem (5.13) by substituting the constraint $\mathbf{w}_t = \mathbf{y}^{\mathcal{S}}[t] - \mathbf{C}^{\mathcal{S}}[t]\hat{\mathbf{f}}^{\mathcal{S}}_{t|t}$ in the objective function:

$$\begin{aligned}
\underset{\hat{\mathbf{f}}^{\mathcal{S}}_{t|t}}{\text{minimize}} \quad & (\mathbf{y}^{\mathcal{S}}[t] - \mathbf{C}^{\mathcal{S}}[t]\hat{\mathbf{f}}^{\mathcal{S}}_{t|t})^\top \mathbf{R}_t{}^{-1}(\mathbf{y}^{\mathcal{S}}[t] - \mathbf{C}^{\mathcal{S}}[t]\hat{\mathbf{f}}^{\mathcal{S}}_{t|t}) \\
& +(\hat{\mathbf{f}}^{\mathcal{S}}_{t|t} - \hat{\mathbf{f}}^{\mathcal{S}}_{t|t-1})\mathbf{P}^{-1}_{t|t-1}(\hat{\mathbf{f}}^{\mathcal{S}}_{t|t} - \hat{\mathbf{f}}^{\mathcal{S}}_{t|t-1})^\top + \mu\mathbf{f}[t]^\top\tilde{\mathbf{L}}^{\ell}_1\mathbf{f}[t]
\end{aligned} \tag{5.14}$$

Next, we differentiate the objective function with respect to $\hat{\mathbf{f}}^{\mathcal{S}}_{t|t}$ and equate to 0 to find the optimum $\hat{\mathbf{f}}^{\mathcal{S}}_{t|t}$:

$$\begin{aligned}
&-2\mathbf{C}^{\mathcal{S}}[t]^\top\mathbf{R}^{-1}_t(\mathbf{y}^{\mathcal{S}}[t] - \mathbf{C}^{\mathcal{S}}[t]\hat{\mathbf{f}}^{\mathcal{S}}_{t|t}) \\
&\qquad\qquad + 2\mathbf{P}^{-1}_{t|t-1}(\hat{\mathbf{f}}^{\mathcal{S}}_{t|t} - \hat{\mathbf{f}}^{\mathcal{S}}_{t|t-1}) + 2\mu\mathbf{L}\mathbf{f}[t] = 0
\end{aligned} \tag{5.15}$$

$$\begin{aligned}
\implies \hat{\mathbf{f}}^{\mathcal{S}}_{t|t} =& (\mathbf{C}^{\mathcal{S}}[t]^\top\mathbf{R}^{-1}\mathbf{C}^{\mathcal{S}}[t] + \mathbf{P}^{-1}_{t|t-1} + 2\mu\mathbf{L})^{-1}\times \\
&(\mathbf{C}^{\mathcal{S}^\top}\mathbf{R}^{-1}\mathbf{Y}^{\mathcal{S}}[t] + \mathbf{P}^{-1}_{t|t-1}\hat{\mathbf{f}}^{\mathcal{S}}_{t|t-1}),
\end{aligned} \tag{5.16}$$

which is the required flow-conservation-based Kalman filter solution.

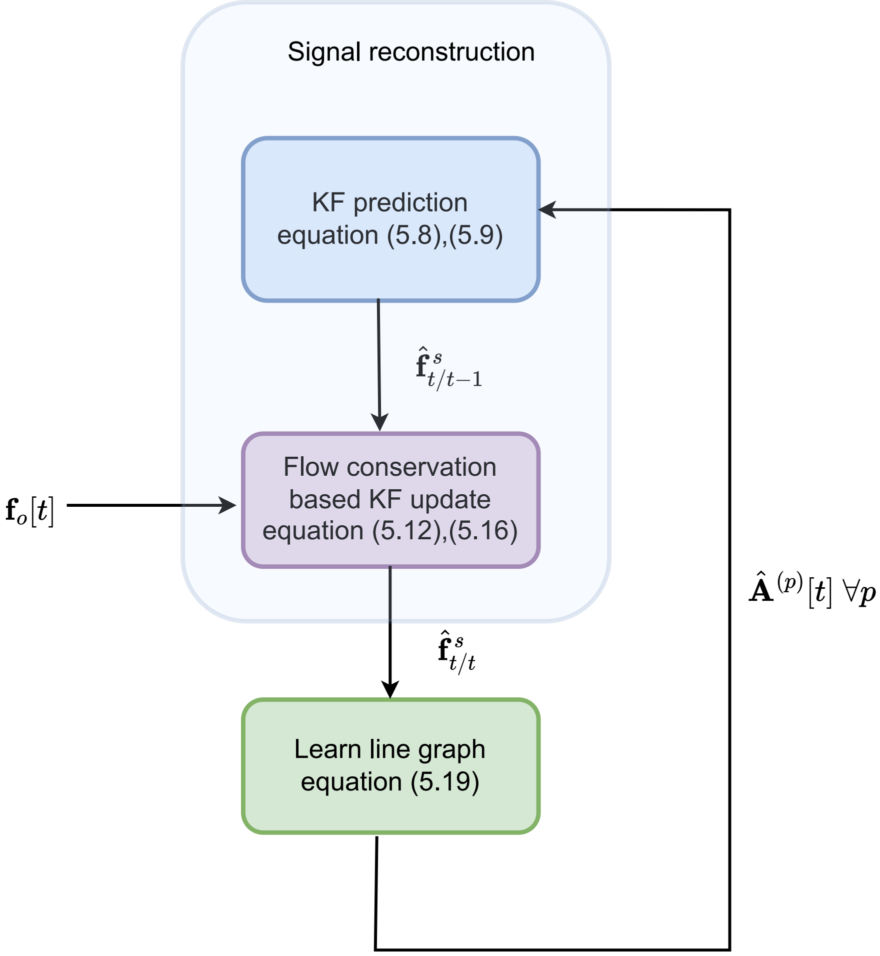

Signal reconstruction
KF prediction equation (5.8),(5.9)
$\hat{\mathbf{f}}^s_{t/t-1}$
$\mathbf{f}_o[t]$
Flow conservation based KF update equation (5.12),(5.16)
$\hat{\mathbf{A}}^{(p)}[t]\ \forall p$
$\hat{\mathbf{f}}^s_{t/t}$
Learn line graph equation (5.19)

Figure 5.3: Proposed algorithm.

### 5.4.2 Line Graph Identification

The node separable version of the model (5.4) is expressed as

$$f_n[t] = \sum_{n'=1}^{E+1} \sum_{p=1}^{P} a^{(p)}_{n,n'}[t] f_{n'}[t-p] + u_n[t], \tag{5.17}$$

where $a^{(p)}_{n,n'}$ is the coefficient which encodes relationship between $p$-th time-lagged value of $n'$-th sensor and $n$-th sensor. Assuming $P$ previous flows $\{f_n[t-p]\}_{p=1}^{P}\ \forall n$ are known, an online line graph learning problem can be formulated as [23, 48]

$$\widehat{\boldsymbol{a}}_n[t] = \arg \min_{\boldsymbol{a}_n \in \mathbb{R}^{(E+1)P}} \ell_t^n(\boldsymbol{a}_n) + \lambda \sum_{n'=1}^{E+1} \|\boldsymbol{a}_{n,n'}\|_2, \tag{5.18}$$

where the loss function $\ell_t^n(\boldsymbol{a}_n) = \frac{1}{2}[f_n[t] - \boldsymbol{a}_n^\top \hat{\mathbf{f}}^{\mathcal{S}}[t-1]]^2$ and $\boldsymbol{a}_n \in \mathbb{R}^{(E+1)P}$ is a column vector containing all the VAR coefficients that influence the sensor $n$ and is obtained by stacking $\left\{a^{(p)}_{n,n'}\right\}_{p=1}^{P} \forall n'$ in the lexicographic order of $p$ and $n'$. The optimization problem (5.18) has a regularization term which induces group sparsity by grouping the influence of all the time-lagged values of $n'$-th sensor on $n$-th sensor as $\boldsymbol{a}_{n,n'} = [a^{(1)}_{n,n'}, \ldots, a^{(P)}_{n,n'}]^\top \in \mathbb{R}^P$. The optimization problem has a differentiable loss function and a nondifferentiable regularizer, so the optimization problem can be solved using COMID. The closed-form solution for the problem is obtained as

$$\widehat{\boldsymbol{a}}_{n,n'}[t+1]=\left(\widehat{\boldsymbol{a}}_{n,n'}[t]-\gamma_t\mathbf{v}_{n,n'}[t]\right)\left[1-\frac{\gamma_t\lambda}{\|\widehat{\boldsymbol{a}}_{n,n'}[t]-\gamma_t\mathbf{v}_{n,n'}[t]\|_2}\right]_+, \tag{5.19}$$

where $[\mathbf{v}_{n,1}^\top, \mathbf{v}_{n,2}^\top, \ldots, \mathbf{v}_{n,N}^\top]^\top \triangleq \nabla_{\boldsymbol{a}}\ell_t^n(\boldsymbol{a}_n[t]) \in \mathbb{R}^{(E+1)P}$. The proposed algorithm is illustrated in Fig. 5.3.

## 5.5 Summary of the Chapter

- This chapter presents an algorithm for missing data imputation for edge data. First, the missing data is reconstructed using a structure-aware Kalman filter, and then, the reconstructed signal is used to learn a noisy model constructed using a line graph. The capability of the proposed algorithm for imputing both permanently and randomly missing flow data is tested using data generated from the EPANET software (see paper [25]).

- To the best of our knowledge, it is the first SC-based data imputation that has been proposed for time series data.

- Even if the problem is formulated as a missing data imputation problem, the proposed model can be extended for data denoising, time series forecasting from partial observation, etc.

# Chapter 6

# Scalable and Privacy-aware Online Learning of Nonlinear Structural Equation Models

This chapter summarizes Paper F ([26])

## 6.1 Motivation

The capability of the Structural Equation model (SEM) to express directional multivariate relationships is well studied. In some real-world applications, time-lag relationships cannot be observed due to swift interactions between variables. SEMs are adequate models in such situations where the interactions are faster than the sampling time. We have discussed in the previous chapters the importance of learning nonlinear relationships in time-varying systems. In this chapter, we propose a nonlinear time-varying SEM topology identification algorithm using a time-structured online optimization approach. Our approach considers the evolution of the model with time rather than the popular time-unstructured approach, which solely relies on observations. A linear SEM topology identification algorithm based on a time-structured approach has been proposed recently [49]. As opposed to existing approaches, we do not rely solely on linear symmetric relationships based on correlation. Instead, we propose a nonlinear and node-separable solution for the problem, where our node-separability feature enhances the scalability of the algorithm compared to other methods. Moreover, we utilize RF approximation, which enables nodes to maintain their data privacy, as data sharing between nodes is not required.

## 6.2 Problem Formulation

Consider $N$ interdependent time series, and let $y_n[t]$ be the value of the $n$-th time series at time $t$. We use a nonlinear SEM with no exogenous variables to model the dependencies among these time series:

$$y_n[t] = \sum_{n'=1, n' \neq n}^{N} f_{n,n'}(y_{n'}[t]) + u_n[t], \quad n = 1, \dots, N, \tag{6.1}$$

where $f_{n,n'}(\cdot)$ encodes the nonlinear influence of $n'$-th time series on $n$-th time series and $u_n[t]$ is the observation noise [14]. We assume that$\{f_{n,n'}(\cdot)\}_{n,n'}$ belongs to an RKHS, and identify the topology by estimating the nonlinear functions $\{f_{n,n'}(\cdot)\}_{n,n'}$. RF approximation allows us to express $f_{n,n'}\left(y_{n'}[\tau]\right)$ in random Fourier space with fixed dimension:

$$\tilde{\hat{f}}_{n,n'}\left(y_{n'}[\tau]\right) = \boldsymbol{\alpha}_{n,n'}{}^{\top} \boldsymbol{z}_{\boldsymbol{v},n'}[\tau], \tag{6.2}$$

where $\boldsymbol{\alpha}_{n,n'} \in R^{2(N-1)D}$ are the parameters that determine the function and $\boldsymbol{z}_{\boldsymbol{v},n'}[\tau]$ is the node-specific RF at time $\tau$. In the following sections, we formulate an optimization problem to estimate $\{\boldsymbol{\alpha}_{n,n'}\}_{n,n'}$.

## 6.3 Topology Identification

Using (6.2), we formulate a parametric optimization problem:

$$\begin{aligned}\{\widehat{\boldsymbol{\alpha}}_{n,n'}\}_{n'} = \arg\min_{\{\boldsymbol{\alpha}_{n,n'}\}} \frac{1}{2} \sum_{\tau=0}^{T-1} \left[ y_n[\tau] - \sum_{n'=1, n' \neq n}^{N} \boldsymbol{\alpha}_{n,n'}^{\top} \boldsymbol{z}_{\boldsymbol{v},n'}[\tau] \right]^2 \\ + \lambda \sum_{n'=1, n' \neq n}^{N} ||\boldsymbol{\alpha}_{n,n'}||_2,\end{aligned} \tag{6.3}$$

where the group Lasso regularizer is introduced to promote sparse solutions. We stack the vectors $\boldsymbol{\alpha}_{n,n'}$ and $\boldsymbol{z}_{\boldsymbol{v},n'}[t]$ along the index $n' = 1, \dots, N,\ n' \neq n$ to form $\boldsymbol{\alpha}_n \in \mathbb{R}^{2(N-1)D}$ and $\boldsymbol{z}_n[t] \in \mathbb{R}^{2(N-1)D}$, and compactly write (6.3) as

$$\widehat{\boldsymbol{\alpha}}_n = \arg\min_{\boldsymbol{\alpha}_n} \mathcal{L}^n\left(\boldsymbol{\alpha}_n\right) + \lambda \sum_{n'=1, n' \neq n}^{N} ||\boldsymbol{\alpha}_{n,n'}||_2, \tag{6.4}$$

$$\text{where } \mathcal{L}^n(\boldsymbol{\alpha}_n) = \frac{1}{2} \sum_{\tau=0}^{T-1} \left[ y_n[\tau] - \boldsymbol{\alpha}_n^{\top} \boldsymbol{z}_n[\tau] \right]^2. \tag{6.5}$$

The proposed optimization problem 6.4 is in batch form, which requires a high computational capability to be solved, and moreover, it does not allow tracking the time-varying nature of real-world dependencies.

## 6.4 Time-varying Solution

Following the online optimization framework, we replace the batch loss in (6.5) with a recursive least squared (RLS) loss using an exponential window:

$$\tilde{\ell}_t^n(\boldsymbol{\alpha}_n) = \mu \sum_{\tau=0}^{t} \gamma^{t-\tau} \ell_\tau^n(\boldsymbol{\alpha}_n). \tag{6.6}$$

where $\ell_\tau^n(\boldsymbol{\alpha}_n) = \frac{1}{2}[y_n[\tau] - \boldsymbol{\alpha}_n^{\top} \boldsymbol{z}_n[\tau]]^2$ is the instantaneous loss function, $\gamma \in (0, 1)$ is the forgetting factor of the window, and $\mu = 1 - \gamma$ normalizes the window. The RLS loss function can be expanded as

$$\begin{aligned}\tilde{\ell}_t^n(\boldsymbol{\alpha}_n) =& \frac{1}{2}\mu \sum_{\tau=0}^{t} \gamma^{t-\tau}\Big(y_n^2[\tau] + \boldsymbol{\alpha}_n^\top \boldsymbol{z}_n[\tau]\boldsymbol{z}_n[\tau]^\top \boldsymbol{\alpha}_n \\ &- 2y_n[\tau]\boldsymbol{z}_n[\tau]^\top \boldsymbol{\alpha}_n\Big) \\ =& \frac{1}{2}\mu \sum_{\tau=0}^{t} \gamma^{t-\tau} y_n^2[\tau] + \frac{1}{2}\boldsymbol{\alpha}_n^\top \boldsymbol{\Phi}_n[t]\boldsymbol{\alpha}_n - \boldsymbol{r}_n^\top \boldsymbol{\alpha}_n, \end{aligned} \tag{6.7}$$

where

$$\boldsymbol{\Phi}_n[t] = \mu \sum_{\tau=0}^{t} \gamma^{t-\tau} \boldsymbol{z}_n[\tau]\boldsymbol{z}_n[\tau]^\top, \tag{6.8}$$

$$\boldsymbol{r}_n[t] = \mu \sum_{\tau=0}^{t} \gamma^{t-\tau} y_n[\tau]\boldsymbol{z}_n[\tau]. \tag{6.9}$$

The new optimization problem using the RLS loss becomes

$$\arg\min_{\boldsymbol{\alpha}_n} \tilde{\ell}_t^n(\boldsymbol{\alpha}_n) + \lambda \sum_{n'=1, n'\neq n}^{N} \|\boldsymbol{\alpha}_{n,n'}\|_2. \tag{6.10}$$

The objective function in (6.10) has a differentiable loss but a non-differentiable regularizer. We solve it using COMID [39]. The COMID update can be solved in closed form for each lasso group $\boldsymbol{\alpha}_{n,n'} \in \boldsymbol{\alpha}_n$ [cf. (6.3)] using the multidimensional shrinkage thresholding operator (MSTO) [41]:

$$\begin{aligned}\boldsymbol{\alpha}_{n,n'}^{(1)}[t+1] =& (\boldsymbol{\alpha}_{n,n'}[t] - \nu_t \mathbf{v}_{n,n'}) \times \\ & \left[1 - \frac{\nu_t \lambda}{\|\boldsymbol{\alpha}_{n,n'}[t] - \nu_t \mathbf{v}_{n,n'}\|_2}\right]_+, \end{aligned} \tag{6.11}$$

where $[\mathbf{v}_{n,1}^\top, \mathbf{v}_{n,2}^\top, \dots, \mathbf{v}_{n,N}^\top]^\top \triangleq \nabla_{\boldsymbol{\alpha}} \tilde{\ell}_t^n(\boldsymbol{\alpha}_n[t])$ and $[x]_+ = \max\{0, x\}$. The MSTO solution (6.11) involves a one-step COMID update. For brevity of the succeeding formulation, we represent the $K$-step version of (6.11) as

$$\boldsymbol{\alpha}_n^{(K)}[t+1] = \mathrm{MSTO}^{(K)}(\tilde{\ell}_t^n(\boldsymbol{\alpha}_n[t]), \nu_t, \lambda), \tag{6.12}$$

which computes the $K$-step descent update of $\boldsymbol{\alpha}_{n,n'}[t]$ as in (6.11), for $n' = 1, \dots, N, n' \neq n$, for the loss function $\tilde{\ell}_t^n(\cdot)$ with the parameters $\nu_t$ and $\lambda$, and stacks them to form $\boldsymbol{\alpha}_n^{(K)}[t+1]$.

## 6.5 Prediction Correction

Solving the optimization problem (6.6) online is possible using a standard time unstructured approach. Using such a method has the limitation that it neglects the model's evolution [50]. The optimization problem (6.6) is presented in a way that allows time-structured approaches to be used. In this work, we use a prediction-correction algorithm under time-structured optimization. It is necessary for the prediction correction algorithm to have a strongly convex loss function and a properly convex regularizer. As the optimization problem (6.6) satisfies this property, we follow such an approach for solving the required problem. The prediction correction algorithm works in two steps: *(i)* Predict the yet unobserved loss function based on available information and make an estimate based on the predicted loss function *(ii)*

Correct the predicted estimate whenever additional information is available in the form of the data stream.

**Prediction:** The first step is to predict at time $t$, the yet unobserved loss function $\tilde{\ell}^n_{t+1}(\boldsymbol{\alpha}_n)$ using Taylor series expansion (because the function is strongly convex):

$$\begin{aligned}\tilde{\ell}^{n,pr}_{t+1}(\boldsymbol{\alpha}_n) = \boldsymbol{\alpha}_n^\top \nabla_{\boldsymbol{\alpha}\boldsymbol{\alpha}} \tilde{\ell}^n_t(\boldsymbol{\alpha}_n)\boldsymbol{\alpha}_n + [\nabla_{\boldsymbol{\alpha}} \tilde{\ell}^n_t(\boldsymbol{\alpha}_n[t]) \\ + \nabla_{t\boldsymbol{\alpha}} \tilde{\ell}^n_t(\boldsymbol{\alpha}_n[t]) - \nabla_{\boldsymbol{\alpha}\boldsymbol{\alpha}} \tilde{\ell}^n_t(\boldsymbol{\alpha}_n[t])\boldsymbol{\alpha}_n[t]]^\top \boldsymbol{\alpha}_n\end{aligned} \tag{6.13}$$

In order to predict the evolution of the loss function, we require, the gradient $\nabla_{\boldsymbol{\alpha}} \tilde{\ell}^n_t(\boldsymbol{\alpha}_n[t])$, the Hessian $\nabla_{\boldsymbol{\alpha}\boldsymbol{\alpha}} \tilde{\ell}^n_t(\boldsymbol{\alpha}_n[t])$ and the partial derivative of $\nabla_{\boldsymbol{\alpha}} \tilde{\ell}^n_t(\boldsymbol{\alpha}_n[t])$ w.r.t. time $\nabla_{t\boldsymbol{\alpha}} \tilde{\ell}^n_t(\boldsymbol{\alpha}_n[t])$ which have the forms

$$\nabla_{\boldsymbol{\alpha}} \tilde{\ell}^n_t(\boldsymbol{\alpha}_n[t]) = \boldsymbol{\Phi}_n[t]\boldsymbol{\alpha}_n - \boldsymbol{r}_n[t], \tag{6.14}$$

$$\nabla_{\boldsymbol{\alpha}\boldsymbol{\alpha}} \tilde{\ell}^n_t(\boldsymbol{\alpha}_n[t]) = \boldsymbol{\Phi}_n[t], \tag{6.15}$$

$$\nabla_{t\boldsymbol{\alpha}} \tilde{\ell}^n_t(\boldsymbol{\alpha}_n[t]) = (\boldsymbol{\Phi}_n[t] - \boldsymbol{\Phi}_n[t-1])\boldsymbol{\alpha} - (\boldsymbol{r}_n[t] - \boldsymbol{r}_n[t-1]). \tag{6.16}$$

The regularizer is time-invariant; hence the prediction step is not required for it. Using the predicted loss (6.13) in place of (6.10), we predict the RF coefficients as

$$\boldsymbol{\alpha}_n^{pr}[t+1] = \mathrm{MSTO}^{(P)}(\tilde{\ell}^{n,pr}_{t+1}(\boldsymbol{\alpha}_n[t]), \nu_t, \lambda), \tag{6.17}$$

where $\boldsymbol{\alpha}_n^{pr}[t+1]$ denotes the $P$-step COMID descent of $\boldsymbol{\alpha}_n[t]$ under the predicted loss. The gradient of the predicted loss involved in the MSTO operation (6.17) can be obtained from (6.13) as

$$\begin{aligned}\nabla_{\boldsymbol{\alpha}} \tilde{\ell}^{n,pr}_{t+1}(\boldsymbol{\alpha}_n[t]) =& (2\boldsymbol{\Phi}_n[t-1] - \boldsymbol{\Phi}_n[t-2])\boldsymbol{\alpha}_n \\ &+ 2\boldsymbol{r}_n[t-1] - \boldsymbol{r}_n[t-2].\end{aligned} \tag{6.18}$$

**Correction:** At time $t+1$, the loss $\tilde{\ell}^n_{t+1}(\cdot)$ [cf. the one appearing in (6.10)] becomes available, and the predicted RF coefficients $\boldsymbol{\alpha}_n^{pr}[t+1]$ are corrected via $C$-step COMID descents:

$$\boldsymbol{\alpha}_n[t+1] = \mathrm{MSTO}^{(C)}(\tilde{\ell}^n_{t+1}(\boldsymbol{\alpha}_n^{pr}[t+1]), \nu_t, \lambda), \tag{6.19}$$

The illustration of the proposed algorithm for node $n$ is shown in Fig. 6.1. As we can see from the Fig. 6.1 model, predict an estimate based on the trajectory of the loss function and update the prediction when new node-specific random features are available. It is possible for the nodes to maintain nodal data privacy since they share random features instead of actual data.

## 6.6 Dynamic Regret

The dynamic regret analysis is derived under the following mild assumptions:

A1) Bounded time series: there exists $B_y > 0$ such that $\{|y_n[t]|^2\}_{n,t} \le B_y \le \infty$,

A2) Shift-invariant kernels: the kernels are shift-invariant, i.e., $k(x_i, x_j) = k(x_i - x_j)$.

A3) Bounded minimum eigenvalue of $\boldsymbol{\Phi}_n[t]$ : There exists $\rho_l > 0$ such that $\Lambda_{\min}(\boldsymbol{\Phi}_n[t]) \ge \rho_l$, $\forall t$, where $\Lambda_{\min}(\cdot)$ is the minimum eigenvalue operator.

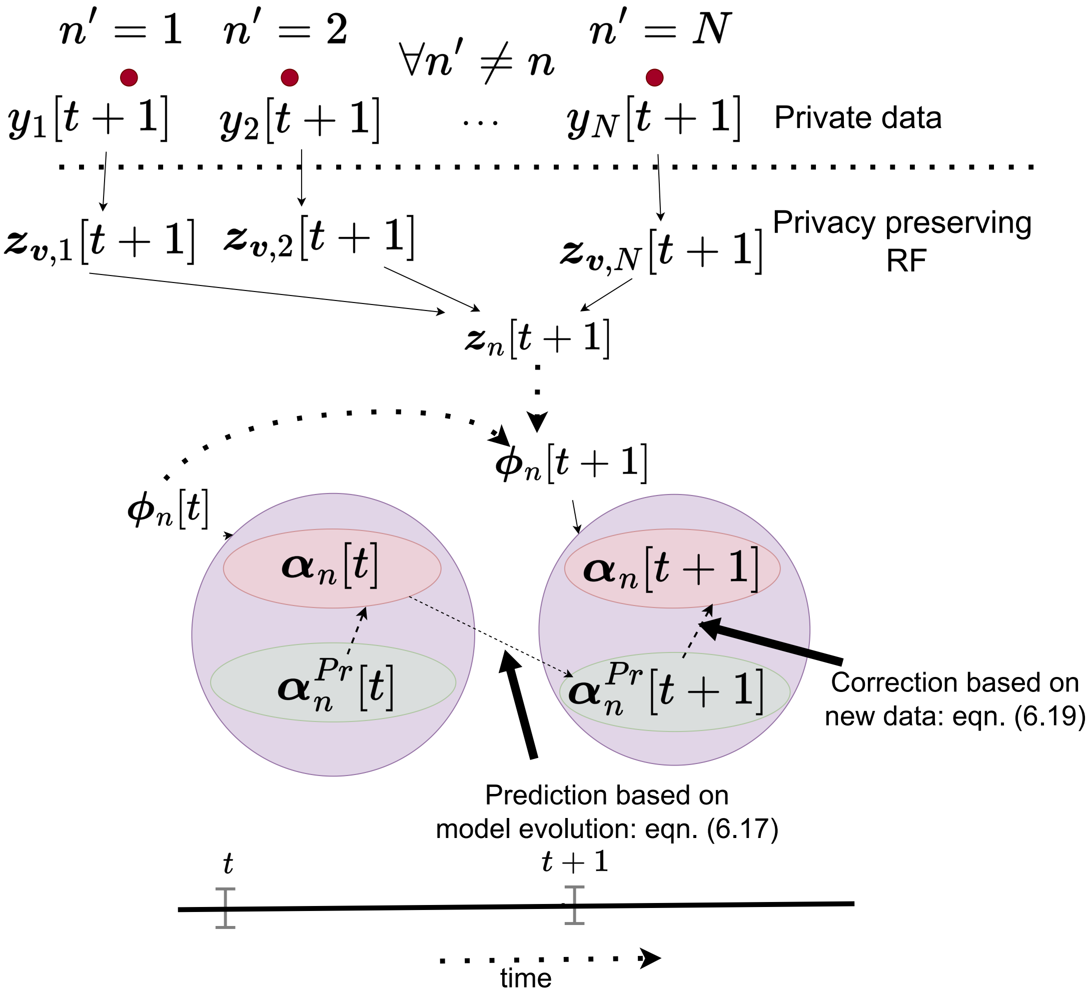


Figure 6.1: Proposed Algorithm.

A4) Bounded maximum eigenvalue: there exists $L > 0$ such that $2\Lambda_{\max}(\mathbf{\Phi}_n[t]) < L < \infty$, $\forall t$, where $\Lambda_{\max}(\cdot)$ is the maximum eigenvalue operator.

Under assumptions A1, A2, A3, and A4, the dynamic regret $R_n(T)$ satisfies

$$R_n(T) \leq \left(\left(1 + \frac{L}{2\rho_l}\right)\sqrt{2(N-1)DB_y} + \lambda\sqrt{N-1}\right) \times$$
$$T\left(q^{(P+C)}\|\boldsymbol{\alpha}_n^*[0]\|_2 + q^{(P+C)}d + q^{(P+C+1)}l\right) + \epsilon\eta L_h T,$$

where $\eta > 0$ is a constant, $L_h$ is the Lipschitz continuity parameter of function $h_t^n(\cdot,\cdot)$, $d$ is the maximum temporal variation in the optimal solution $\|\boldsymbol{\alpha}_n^*[t] - \boldsymbol{\alpha}_n^*[t-1]\|_2$, and $l$ is the maximum error in the optimal prediction $\|\boldsymbol{\alpha}_n^*[t] - \boldsymbol{\alpha}_n^{pr*}[t]\|_2$ with $\boldsymbol{\alpha}_n^{pr*}[t]$ the optimum prediction at time $t$. The quantity $q \in (0, 1)$ is the contraction coefficient, and its value for various optimization techniques is provided in [51](The proof is provided in [26]).

The dynamic regret bound is linear in time, which implies that $\lim_{t\to\infty} R_n(T)/T = constant$, where $constant$ is the steady state error, which depends on $l = \|\boldsymbol{\alpha}_n^*[t] - \boldsymbol{\alpha}_n^{pr*}[t]\|_2$, $d = \|\boldsymbol{\alpha}_n^*[t] - \boldsymbol{\alpha}_n^*[t-1]\|_2$, and the constant $\epsilon \geq 0$. This means that if $d$ and $l$ are low (slowly varying systems), it is possible to have a very low bound for the asymptotic $R_n(T)/T$ by controlling $\epsilon$ at the expense of model complexity.

## 6.7 Summary of the Chapter

- This chapter presents an online algorithm for SEM topology identification using a time-structured approach, in which the evolution of the model is also exploited along with data. The Optimization problem is formulated in a way that is privacy-aware and scalable.

- We also derive a dynamic regret bound for the algorithm.

- We test the capability of the algorithm with both synthetic and real data (see paper [26]).

# Chapter 7

# Concluding Remarks

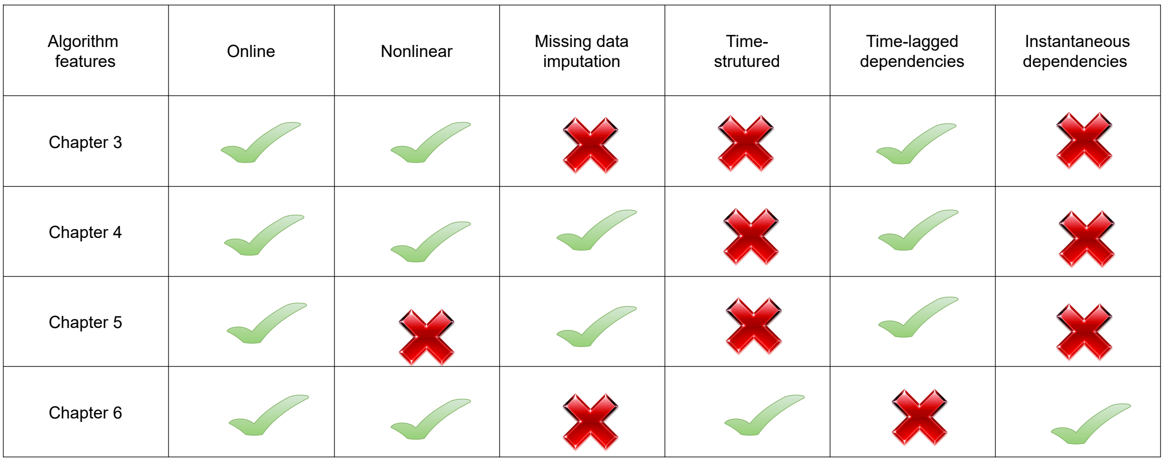
## 7.1 Conclusion

This dissertation proposes various algorithms for online inference from multiple time series. The two major areas covered in the dissertation are; *(i)* online nonlinear topology identification, *(ii)* online missing data imputation. A detailed description of the proposed algorithms can be found in Chapters 3, 4, 5, and 6, after providing the motivation and background in Chapters 1 and 2.

In Chapter 3, we assume that the observed data is generated from a nonlinear VAR model (2.2) and propose three algorithms (NL-TISO, RFNL-TISO, and RFNL-TIRSO) to solve the problem of online topology identification. Motivated by the sparse interactions in real-world networks, we formulate convex optimization problems with differentiable loss functions and non-differentiable group Lasso regularizers. Such optimization is then solved using a composite objective mirror descent technique, resulting in online topology estimation algorithms with fixed computational complexity per iteration. We leverage the kernel methods to handle the nonlinearities. The curse of dimensionality associated with kernel methods is mitigated using a forgetting window in NL-TISO, whereas RFNL-TISO and RFNL-TIRSO use random feature approximation. Compared to RFNL-TIRSO, RFNL-TISO is computationally less demanding; however, RFNL-TIRSO is more resilient to noise. The strong convexity of the RFLN-TIRSO loss function, allows us to derive an upper bound for the dynamic regret and conduct a theoretical investigation into the convergence assurance.

In Chapter 4, we propose a kernel-based online framework using random feature approximation to jointly estimate nonlinear VAR topologies and missing data from partial observations of streaming multivariate time series data. We convexify the joint optimization problem and solve it using a two-step approach: *(i)* estimate the signal based on observations and the current model, and *(ii)* update the model based on the estimated signals.

Chapter 5 presents a novel online algorithm for imputing missing data in networks with edge-defined signals. We use a bi-level optimization scheme that takes advantage of the known physical structure of the network. Our proposed algorithm involves two steps: *(i)* a sparse line graph identification step obtained by



| Algorithm features | Online | Nonlinear | Missing data imputation | Time-strutured | Time-lagged dependencies | Instantaneous dependencies |
|---|---|---|---|---|---|---|
| Chapter 3 | ✓ | ✓ | ✗ | ✗ | ✓ | ✗ |
| Chapter 4 | ✓ | ✓ | ✓ | ✗ | ✓ | ✗ |
| Chapter 5 | ✓ | ✗ | ✓ | ✗ | ✓ | ✗ |
| Chapter 6 | ✓ | ✓ | ✗ | ✓ | ✗ | ✓ |

Figure 7.1: Key features of the algorithms proposed in Chapters 3-6.

solving a group-Lasso-based optimization framework via composite objective mirror descent to exploit the spatio-temporal dependencies among the edge signals; and *(ii)* a Kalman-filtering-based signal reconstruction step developed using the simplicial complex (SC) formulation to exploit the flow conservation of the edge signals. Unlike the preceding chapters, this chapter focuses on signals defined on edges. Furthermore, the SC formulation employed in this chapter gives the algorithm the capacity to integrate flow-conservation properties of edge signals, assisting in imputing permanently missing signals, which is not possible with the imputation schemes proposed in Chapter 4.

In Chapter 6, we assume that the dependencies are instantaneous and propose an online algorithm using the RF-based kernel formulation for estimating nonlinear SEM topology. In our proposed method, data is used for both learning the model and tracking its evolution. The SEM parameters are updated using the predicted model parameters and the new data samples through a time-structured prediction-correction strategy. Our proposed approach possesses three key properties. First, it enables node-separable learning, which promotes scalability in large networks. Second, it provides privacy in SEM learning by substituting the actual data with node-specific Random Features (RF). Third, its performance can be characterized theoretically via a dynamic regret analysis, demonstrating that a linear dynamic regret bound can be achieved under mild assumptions.

A summary of the key differences between the algorithms proposed in each chapter is provided in Fig. 7.1.

## 7.2 Future Work

- The existing literature mostly assumes the underlying network has a graph structure with pairwise relationships. In many real-world networks like the brain network, gene regulatory network, etc; the interactions are higher order; identifying these higher-order interactions and representing the networks in the form of hypergraphs or simplicial complex (SC) is a challenging problem, and it comes as a natural extension to my Ph.D. work.

- Inference over higher-order network: Representing networks as simplicial complexes (SC) offers an efficient way to capture higher-order interactions and utilize algebraic topological tools for network inference. By incorporating context through the SC representation, tasks such as denoising, missing data imputation, and time series forecasting in flow networks (e.g., transportation, water, and brain information flow networks) can be improved by biasing solutions based on existing knowledge, such as divergence and curl.

- Controller design for the graph-structured network: Apart from signal processing over networks, I have a deep interest in control theory. In graph-connected networks, it is possible to manipulate the behavior of entire nodes by controlling a subset of nodes, provided the graph is fully connected. Optimal control strategies for such networks have applications in various fields, including epidemic control, stock market management, and data-driven control for sensor networks.

- Graph-informed Reinforcement learning (RL): A research area that I find particularly interesting is the development of interpretable graph-informed reinforcement learning (RL) algorithms. This is a promising area for designing control strategies in large-scale dynamical systems that involve subsystems with intricate spatio-temporal interactions, such as wind farms. Traditionally, researchers and engineers learn the interactions within the dynamical systems using physics-based models (e.g., computational fluid dynamics model). However, such models are often computationally expensive, restricting their use in RL algorithms for real-time applications. I intend to investigate the possibility of designing RL algorithms that utilize computationally light online graph learning techniques. This would improve control strategies by exploiting the subsystem interactions embedded in the graph.

# Appendix A

# PAPER A

**Title**: Online Non-linear Topology Identification from Graph-connected Time Series

**Authors**: **R. Money**, J. Krishnan, B. Beferull-Lozano

**Conference**: IEEE Data Science and Learning Workshop 2021

# Online Non-linear Topology Identification from Graph-connected Time Series

R. Money, J. Krishnan, B. Beferull-Lozano

**Abstract:** **Estimating the unknown causal dependencies among graph-connected time series plays an important role in many applications, such as sensor network analysis, signal processing over cyber-physical systems, and finance engineering. Inference of such causal dependencies, often know as topology identification, is not well studied for non-linear non-stationary systems, and most of the existing methods are batch-based which are not capable of handling streaming sensor signals. In this paper, we propose an online kernel-based algorithm for topology estimation of non-linear vector autoregressive time series by solving a sparse online optimization framework using the composite objective mirror descent method. Experiments conducted on real and synthetic data sets show that the proposed algorithm outperforms the state-of-the-art methods for topology estimation.**

## A.1 Introduction

Recent advancements in cyber-physical systems (CPS) and sensor networks call for advanced research on data analysis of structured or inter-linked spatio-temporal signals. Such structured signals can be meaningfully represented using graph-connected time series. Graph representation is a prevalent tool to model the inter-dependency of data [52], and it plays a vital role in countless practical applications such as time series prediction [53], change point detection [54], data compression [55], etc. Many of the functional dependencies in real-world time series are causal [56], and inferring the causal dependencies, which we term as *topology identification*, generates a more informative representation of the multivariate data. These dependencies may not be physically observable in some cases; instead, there can be logic connections between data nodes that are not physically connected due to control mechanisms, and inferring such typologies is a challenging task. Linear models, such as structural equation models (SEM), vector auto-regressive (VAR) models, and structural vector auto-regressive (SVAR) models [57] are widely used to study the causal dependencies among the graph-connected time series. SEM being a memory-less model, does not accommodate the temporal dependencies among the data, whereas the VAR is an ideal choice for modeling the time-lagged interactions; however, it fails to capture the instantaneous causal relations. SVAR is a slightly modified model that unifies both SEM and VAR. The choice of the model depends on the physical nature of the system; for instance, SVAR is a useful model for brain connectivity analyses. However, VAR deserves special attention since the nodal dependencies on many practical sensor networks (e.g., water networks, oil and gas networks) involve mainly time-lagged interactions.

A significant challenge connected to topology identification is that the real-world systems are usually non-stationary, meaning that the statistical properties of dependencies vary over time. The commonly used batch-based off-line methods [11] have two major drawbacks: **i)** they are not effective in tracking the topology of non-stationary systems and **ii)** from a pure computational point of view, they suffer from processing large batch of data; hence, it is necessary to develop online estimation algorithms [48]. Online topology estimation algorithms have been developed for linear models, meaning that the causal dependencies among the data time-series hold a linear relation. For instance, in [48], a novel online linear topology identification algorithm have been proposed by minimizing a group-lasso-regularized [58] objective function.

Although the linear topology identification is a well-studied problem, many practical systems have non-linear dependencies [59]. As an example, in a smart water network, the causal dependencies are non-linear due to various control systems, saturation in valves or pumps, and non-linear physical equations governing the system. Similarly, essential non-linear dependencies are present in most of the real-world systems such as brain networks and finance networks. The ability of nonparametric techniques [60] and deep neural networks to learn non-linear functions is well studied, which has been exploited also in topology identification [9], [11], [14]. However, once again most of these algorithms are batch-based. Kernel-based representations are powerful tools to model the non-linear dependencies [15], which can be exploited to develop algorithms for online non-linear topology identification. For instance, in [61], authors have proposed an online algorithm based on functional gradient descent by considering a SVAR model. In [12], authors used a more general non-additive model for topology identification and a dictionary-based approach to solve the computational complexity imposed by the kernels. But [12] restricts the choice of the kernel functions to be twice differentiable to learn a sparse topology.

This paper proposes an online topology identification algorithm based on a non-linear VAR model using kernels. The proposed algorithm learns sparse and time-varying non-linear typology by solving an online optimization framework using composite objective iterations [39]. We provide strong empirical evidence using real and synthetic data sets, which show that the proposed algorithm outperforms its state-of-the-art counterpart.

## A.2 Problem Formulation

Consider a collection of $N$ time series, connected by a directed graph and let $y_n[t]$ be the value of time series at time $t = 0, 1, \ldots, T-1$ measured at node $1 \leq n \leq N$. A $P$-th order non-linear VAR model of the time series can be formulated as

$$y_n[t] = \sum_{n'=1}^{N} \sum_{p=1}^{P} a_{n,n'}^{(p)} f_{n,n'}^{(p)}(y_{n'}[t-p]) + u_n[t], \tag{A.1}$$

where $f_{n,n'}^{(p)}$ is a non-linear function that captures the causal influence of the $p$-lagged data at node $n'$ on the node $n$, $a_{n,n'}^{(p)}$ is the corresponding entry of the graph adjacency matrix, and $u_n[t]$ is the measurement noise.Referring to (A.1), topology identification can be defined as the estimation of the non-linear dependencies expressed by $\left\{a_{n,n'}^{(p)} f_{n,n'}^{(p)}(.)\right\}_{p=1}^{P}$ for $n = 1, 2, \ldots, N$ from the observed time series $\{y_n[t]\}_{n=1}^{N}$.

To circumvent the challenges in topology identification, imposed by the non-linear dependencies, we assume that the functions $f_{n,n'}^{(p)}(.)$ in (A.1) belong to a reproducing kernel Hilbert space (RKHS):

$$\mathcal{H}_{n'}^{(p)} := \left\{ f_{n,n'}^{(p)} | f_{n,n'}^{(p)}(y) = \sum_{t=0}^{\infty} \beta_{n,n',t}^{(p)} \, \kappa_{n'}^{(p)}(y, y_{n'}[t-p]) \right\}, \tag{A.2}$$

where $\kappa_{n'}^{(p)} : \mathbb{R} \times \mathbb{R} \rightarrow \mathbb{R}$ is the Hilbert space basis function, often known as the kernel, which measures the similarities between the arguments of the basis function. Using (A.2), a function $f_{n,n'}^{(p)}$ evaluated at $y$ can be represented as the linear weighted sum of the similarities between $y$ and the data samples $\{y_{n'}[t-p]\}_{t=0}^{t=\infty}$, where the weights are denoted by $\beta_{n,n',t}^{(p)}$. We assume that the Hilbert space is characterized by the inner product $\langle \kappa_{n'}^{(p)}(y, x_1), \kappa_{n'}^{(p)}(y, x_2) \rangle := \sum_{t=0}^{\infty} \kappa_{n'}^{(p)}(y[t], x_1) \kappa_{n'}^{(p)}(y[t], x_2)$, with the kernel having the reproducible property $\langle \kappa_{n'}^{(p)}(y, x_1), \kappa_{n'}^{(p)}(y, x_2) \rangle = \kappa_{n'}^{(p)}(x_1, x_2)$. Such a Hilbert space with the reproducing kernel constitutes an RKHS with norm $\|f_{n,n'}^{(p)}\|_{\mathcal{H}_{n'}^{(p)}}^2 = \sum_{t=0}^{\infty} \sum_{t'=0}^{\infty} \beta_{n,n',t}^{(p)} \, \beta_{n,n',t'}^{(p)} \, \kappa_{n'}^{(p)}(y_n[t], y_n[t'])$. We refer to [37] for further reading on RKHS.

The least-squares (LS) estimate $\left\{f_{n,n'}^{(p)} \in \mathcal{H}_{n'}^{(p)}; n' = 1, \ldots, N, \ p = 1, \ldots, P\right\}$ for a particular node is obtained by solving the following non-parametric optimization problem:

$$\left\{\widehat{f}_{n,n'}^{(p)}\right\}_{n',p} = \arg \min_{\left\{f_{n,n'}^{(p)} \in \mathcal{H}_{n'}^{(p)}\right\}} \frac{1}{2} \sum_{\tau=P}^{T-1} \Bigg[ y_n[\tau] - \\ \sum_{n'=1}^{N} \sum_{p=1}^{P} a_{n,n'}^{(p)} f_{n,n'}^{(p)}(y_{n'}[\tau-p]) \Bigg]^2 . \tag{A.3}$$

It is to be noted that, in (A.3), the functions $\{f_{n,n'}^{(p)}\}$ belongs to the RKHS, defined in (A.2), which is an infinite dimensional space. However, by resorting to the Representer Theorem [38], the solution of (A.3) can be written using a finite number of data samples:

$$\widehat{f}_{n,n'}^{(p)}(y_{n'}[\tau-p]) = \sum_{t=p}^{p+T-1} \beta_{n,n',(t-p)}^{(p)} \kappa_{n'}^{(p)}(y_{n'}[\tau-p]), y_{n'}[t-p]) \, . \tag{A.4}$$

Using (A.4), (A.3) can be reformulated as a parametric optimization problem involving the available data samples, as follows:

$$\left\{\widehat{\alpha}^{(p)}_{n,n',t}\right\}_{n',p,t} = \arg\min_{\left\{\alpha^{(p)}_{n,n',t}\right\}} \mathcal{L}^n\left(\alpha^{(p)}_{n,n',t}\right), \tag{A.5}$$

where

$$\mathcal{L}^n\left(\alpha^{(p)}_{n,n',t}\right) := \frac{1}{2}\sum_{\tau=P}^{T-1}\left[y_n[\tau] - \sum_{n'=1}^{N}\sum_{p=1}^{P}\sum_{t=p}^{p+T-1} \alpha^{(p)}_{n,n',t}\kappa^{(p)}_{n'}(\tau,t)\right]^2, \tag{A.6}$$

$$\alpha^{(p)}_{n,n',t} := a^{(p)}_{n,n'}\beta^{(p)}_{n,n',(t-p)}, \tag{A.7}$$

and

$$\kappa^{(p)}_{n'}(\tau,t) := \kappa^{(p)}_{n'}(y_{n'}[\tau-p]), y_{n'}[t-p]). \tag{A.8}$$

We stack the entries of $\left\{\alpha^{(p)}_{n,n',t}\right\}$ and $\left\{\kappa^{(p)}_{n'}(\tau,t)\right\}$ in the lexicographic order of the indices $p$, $n'$, and $t$ to obtain the vectors $\boldsymbol{\alpha}_n \in \mathbb{R}^{PNT}$ and $\boldsymbol{\kappa}_\tau \in \mathbb{R}^{PNT}$, respectively, and rewrite (A.5) as

$$\widehat{\boldsymbol{\alpha}}_n = \arg\min_{\boldsymbol{\alpha}_n} \mathcal{L}^n(\boldsymbol{\alpha}_n), \tag{A.9}$$

where

$$\mathcal{L}^n(\boldsymbol{\alpha}_n) = \frac{1}{2}\sum_{\tau=P}^{T-1}\left[y_n[\tau] - \boldsymbol{\alpha}_n^\top\boldsymbol{\kappa}_\tau\right]^2 \tag{A.10}$$

Further, to avoid overfitting and to enforce group sparsity, we propose a regularized optimization framework:

$$\widehat{\boldsymbol{\alpha}}_n = \arg\min_{\boldsymbol{\alpha}_n} \mathcal{L}^n(\boldsymbol{\alpha}_n) + \lambda\sum_{n'=1}^{N}\sum_{p=1}^{P}\|\boldsymbol{\alpha}^{(p)}_{n,n'}\|_2, \tag{A.11}$$

where $\lambda \geq 0$ is the regularization parameter and $\boldsymbol{\alpha}^{(p)}_{n,n'} = (\alpha^{(p)}_{n,n',0}, \alpha^{(p)}_{n,n',1}, \ldots, \alpha^{(p)}_{n,n',T}) \in \mathbb{R}^T$. The second term in (A.11) is a *group-lasso* regularizer, which promote a *group-sparse structure* in $\boldsymbol{\alpha}^{(p)}_{n,n'}$, thereby exploiting the prior information that the number of causal dependencies are typically small for real-world graph-connected time series.

The parametric optimization given by (A.11) is a batch (offline) solver meaning that to solve (A.11), we require all data samples $\{y_n[\tau]\}_{\tau=P}^{T-1}$ to be available. Such an offline approach has two major drawbacks: **i)** it is not suitable for real-time applications since the solver has to wait for the entire batch of data and **ii)** it suffers from high computation complexity and memory requirements which grows super linearly with the batch size. In the following section, we propose an online algorithm to estimate the coefficients $\boldsymbol{\alpha}_n$ in (A.11).

## A.3 Online topology estimation

First replace the original loss function $\mathcal{L}^n(\boldsymbol{\alpha}_n)$ in (A.11) with the instantaneous loss function $l^n_\tau(\boldsymbol{\alpha}_n) = \frac{1}{2}[y_n[\tau] - \boldsymbol{\alpha}_n^\top\boldsymbol{\kappa}_\tau]^2$:

$$\widehat{\boldsymbol{\alpha}}_n = \arg\min_{\boldsymbol{\alpha}_n} l^n_\tau(\boldsymbol{\alpha}_n) + \lambda\sum_{n'=1}^{N}\sum_{p=1}^{P}\|\boldsymbol{\alpha}^{(p)}_{n,n'}\|_2. \tag{A.12}$$

A straightforward way to solve (A.12) is by applying the online subgradient descent (OSGD). However, it is to be remarked that the regularizer in (A.12) is non-differentiable and OSGD fails to provide sparse $\boldsymbol{\alpha}_{n,n'}^{(p)}$ since it linearizes the entire instantaneous objective function in (A.12) [48].To mitigate this issue, we use the composite objective mirror descent (COMID) [39] algorithm. The online COMID update can be written as

$$\boldsymbol{\alpha}_n[t+1] = \arg\min_{\boldsymbol{\alpha}_n} J_t^{(n)}(\boldsymbol{\alpha}_n), \tag{A.13}$$

where

$$\begin{aligned} J_t^{(n)}(\boldsymbol{\alpha}_n) &\triangleq \nabla \ell_t^n(\tilde{\boldsymbol{\alpha}}_n[t])^\top (\boldsymbol{\alpha}_n - \tilde{\boldsymbol{\alpha}}_n[t]) \\ &+ \frac{1}{2\gamma_t}\|\boldsymbol{\alpha}_n - \tilde{\boldsymbol{\alpha}}_n[t]\|_2^2 + \lambda \sum_{n'=1}^{N}\sum_{p=1}^{P} \|\boldsymbol{\alpha}_{n,n'}^{(p)}\|_2. \end{aligned} \tag{A.14}$$

In (A.14), $\tilde{\boldsymbol{\alpha}}_n[t] \in \mathbb{R}^{PN(t+1)}$ is defined as $[\boldsymbol{\alpha}_n[t]; \mathbf{0}]$, where $\boldsymbol{\alpha}_n[t] \in \mathbb{R}^{PNt}$ is the value of $\boldsymbol{\alpha}_n$ estimated by processing the samples up to time $t$. The zero vector $\mathbf{0} \in \mathbb{R}^{PN}$ is appended as an initialization for the coefficients of the new elements of the kernel vector corresponding to the $(t+1)^{th}$ data sample. In (A.14), the first term is the gradient of the loss function and the second term is the Bregman divergence $B(\boldsymbol{\alpha}_n, \tilde{\boldsymbol{\alpha}}_n[t]) = \frac{1}{2}\|\boldsymbol{\alpha}_n - \tilde{\boldsymbol{\alpha}}_n[t]\|_2^2$, chosen in such a way that the COMID update has a closed form solution [40] and $\gamma_t$ is the corresponding step size. Bregman divergence ensures that $\boldsymbol{\alpha}_n[t+1]$ is close to $\tilde{\boldsymbol{\alpha}}_n[t]$, in line with the assumption that the topology changes smoothly. The third term is a sparsity enforcing regularizer, in order to promote sparsity in the updates. The gradient in (A.14) is evaluated as

$$\mathbf{v}_n[t] := \nabla \ell_t^n(\tilde{\boldsymbol{\alpha}}_n[t]) = \boldsymbol{\kappa}_\tau \left(\boldsymbol{\alpha}_n^\top \boldsymbol{\kappa}_\tau - y_n[\tau]\right) \tag{A.15}$$

Expanding the objective function in (A.14) by omitting the constants leads to the following formulation:

$$\begin{aligned} J_t^{(n)}(\boldsymbol{\alpha}_n) &\propto \frac{\boldsymbol{\alpha}_n^\top \boldsymbol{\alpha}_n}{2\gamma_t} + \boldsymbol{\alpha}_n^\top \left(\mathbf{v}_n[t] - \frac{1}{\gamma_t}\tilde{\boldsymbol{\alpha}}_n[t]\right) + \lambda \sum_{n'=1}^{N}\sum_{p=1}^{P} \|\boldsymbol{\alpha}_{n,n'}^{(p)}\|_2 \\ &= \sum_{n'=1}^{N}\sum_{p=1}^{P} \left[ \frac{{\boldsymbol{\alpha}_{n,n'}^{(p)}}^\top \boldsymbol{\alpha}_{n,n'}^{(p)}}{2\gamma_t} + {\boldsymbol{\alpha}_{n,n'}^{(p)}}^\top \left(\mathbf{v}_{n,n'}^{(p)}[t] - \frac{1}{\gamma_t}\tilde{\boldsymbol{\alpha}}_{n,n'}^{(p)}[t]\right) \right. \\ &\qquad \left. + \lambda \|\boldsymbol{\alpha}_{n,n'}^{(p)}\|_2 \right]. \end{aligned} \tag{A.16}$$

Note that (A.16) is separable in $n'$, $m$ and $p$. Using (A.16), a closed form solution of (A.13) can be obtained in terms of multidimensional shrinkage-thresholding operator [41] as

$$\begin{aligned} \boldsymbol{\alpha}_{n,n'}^{(p)}[t+1] &= \left(\tilde{\boldsymbol{\alpha}}_{n,n'}^{(p)}[t] - \gamma_t \mathbf{v}_{n,n'}^{(p)}[t]\right) \times \\ &\quad \left[1 - \frac{\gamma_t \lambda \nVdash \{n \neq n'\}}{\|\tilde{\boldsymbol{\alpha}}_{n,n'}^{(p)}[t] - \gamma_t \mathbf{v}_{n,n'}^{(p)}[t]\|_2}\right]_+, \end{aligned} \tag{A.17}$$

where $[x]_+ = \max\{0, x\}$ and

$$\nVdash \{n \neq n'\} = \begin{cases} 1, & \text{if } n \neq n' \\ 0, & n = n'. \end{cases}$$

The term $\tilde{\boldsymbol{\alpha}}_{n,n'}^{(p)}[t] - \gamma_t \mathbf{v}_{n,n'}^{(p)}[t]$ in (A.17) performs a stochastic gradient update of $\boldsymbol{\alpha}_{n,n'}^{(p)}$ in a direction that decreases the instantaneous loss function $l_\tau^n(\boldsymbol{\alpha}_n)$ and the second term in (A.17) promotes group sparsity of $\boldsymbol{\alpha}_{n,n'}^{(p)}$. The function $\nVdash \{n \neq n'\}$ in the second term prevents the enforcement of sparsity of self-connections of the graph. One major issue with (A.17) is that the size of $\boldsymbol{v}_{n,n'}^{(p)}[t]$ becomes prohibitive as $t$ increases. To mitigate this issue we select the recent $T_w$ data points to calculate (A.17). For the experiments presented in this paper, we heuristically fix the value of $T_w$ to 2000. Although this sub-optimal approach affects the performance of the algorithm, we are getting quite competitive empirical performance as shown later in the experiment section.

The proposed algorithm, termed as *Nonlinear Topology Identification via Sparse Online learning* (NL-TISO), is summarized in **Algorithm 5**.

**Algorithm 5:** NL-TISO Algorithm

```
Result: $\boldsymbol{\alpha}_{n,n'}^{(p)}, for\ n, n' = 1, .., N$ and $p = 1, .., P$
Store $\{\boldsymbol{y}_n[t]\}_{t=1}^{P}$,
Initialize $\lambda$, $\gamma$ (heuristically chosen) and kernel parameters depending on
  the type of the kernel.
for $t = P, P+1, \dots$ do
    Get data samples $y_n[t]$, $\forall n$ and compute $\boldsymbol{\kappa}_\tau$
    for $n = 1, \dots, N$ do
        compute $\mathbf{v}_n[t]$ using (A.15)
        for $n' = 1, \dots, N$ do
            compute $\boldsymbol{\alpha}_{n,n'}^{(p)}[t+1]$ using (A.17)
        end
    end
end
```

## A.4 Experiments

In this section, we illustrate the effectiveness of the proposed NL-TISO algorithm using synthetic and real data. We compare our results with two state-of-the-art topology estimation algorithms: **i)** TIRSO [48]- a recent online topology estimation algorithm based on COMID update developed for linear causal dependencies and **ii)** functional gradient descent (FGD) algorithm [61]- an online kernal based topology estimation algorithm based on functional gradient descent updates

### A.4.1 Experiments using synthetic data

#### A.4.1.1 Identifying causal dependencies

We generated graph connected time series, based on the non-linear VAR model (C.1) with parameter values $N = 5$, $T = 3000$, and $P = 2$. The entries of the graph adjacency matrix $\left\{a_{n,n'}^{(p)}\right\}$ are drawn from a Gaussian distribution $\mathcal{N}(8,\ 3)$ with an edge probability $p_e = 0.1$. The initial $P$ samples of the time series are drawn randomly from a Gaussian distribution $\mathcal{N}(0,\ 0.1)$ and the remaining samples are generated using model (C.1). A Gaussian kernel centered at the dependent data points and having variance 0.03 is used to model the non-linear dependencies in (C.1), where the kernel coefficients $\beta^{(p)}_{n,n'}$ are drawn from a zero mean Gaussian distribution with variance 0.03. The noise $u_n[t]$ is generated from a zero mean Gaussian distribution with variance 0.01. The causal dependencies $\left\{\boldsymbol{\alpha}^{(p)}_{n,n'}[t]\right\}$ are estimated using the

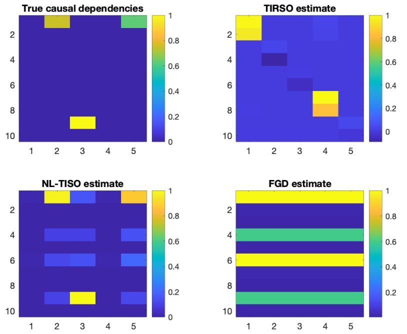


Figure A.1: Causal dependencies (normalized) estimated using different algorithms compared with the true dependency.

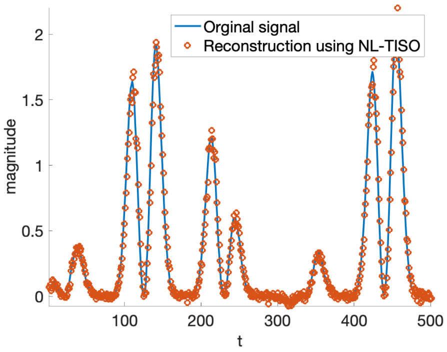


Figure A.2: Reconstruction of true signal in node 1 using estimated coefficients.

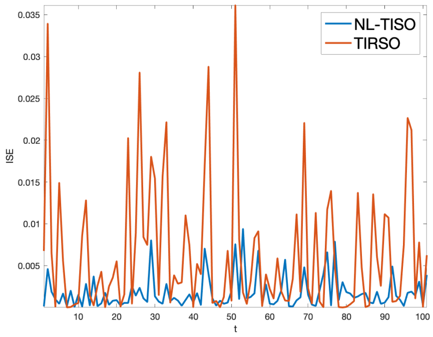


Figure A.3: ISE comparison of NL-TISO and TIRSO when the signal to be reconstructed is rapidly varying.

proposed NL-TISO algorithm using Gaussian kernel having variance 0.1 and with hyper-parameters $\lambda = 0.1$ and $\gamma = 10$. Since a stationary topology is considered in this experiment, we compute the $\ell_2$ norms $\widehat{b}^{(p)}_{n,n'} = \|\boldsymbol{\alpha}^{(p)}_{n,n'}[t]\|_2$ at $t = T$ and arrange them in a matrix structure similar to the graph adjacency matrix to visualize the causal dependencies. A similar strategy is adopted for the FGD and the TIRSO algorithms, and the estimated adjacency matrix is used to visualize the dependencies. The true and the estimated dependencies are shown in Fig. B.1, in which for each subplot, the $5 \times 5$ dependency matrices corresponding to $p = 1$ and 2 are concatenated, resulting in a size $10 \times 5$ size matrix. From Fig. B.1, it is clear that the NL-TISO algorithm outperforms others in identifying the causal relationship.

#### A.4.1.2 Signal Reconstruction Experiment

In this experiment, using the inferred causal dependencies, we reconstruct the time series and compare it with the true signals. In contrary to the previous experiment, a dynamic graph-topology is considered here using a time varying adjacency matrix

$$a^{(p)}_{n,n'}[t+1] = a^{(p)}_{n,n'}[t] + 0.01\sin(0.03 * t) \tag{A.18}$$

with a random initialization. We use a different non-linear dependency compared to the previous experiment to generate data:

$$f^{(p)}_{n,n'}(x) = 0.4\sin(\pi x^2) + 0.3\sin(2\pi x) + 0.3\sin(3\pi x). \tag{A.19}$$

Graph-connected time series ($N = 5$) are generated using (C.1), (A.18), and (A.19) in a similar manner as described in A.4.1.1.

The causal dependencies $\left\{\boldsymbol{\alpha}^{(p)}_{n,n'}\right\}$ are estimated from the time series using NL-TISO with a Gaussian kernel having variance 0.02 and with hyper-parameters $\lambda = 10^{-6}$ and $\gamma = 10$ . Using the same Gaussian kernel and the estimated dependencies, the time series are reconstructed. In Fig. A.2, a visual comparison of both the true and reconstructed time series at one of the five nodes is shown. We observed that the reconstructed signal is very close to the true one, although a Gaussian-based kernel is used to infer the non-linearity imposed by (A.19), which in turn indicates that kernel-based representations are a powerful tool in handling the non-linear causal dependencies. Further, the signal reconstruction quality of the state-of-the-art algorithms TIRSO [48] and FGD [61] are compared using *instantaneous squared error*, which is defined as $ISE(t) = (y_n(t) - \hat{y}_n(t))^2$ and is plotted in Fig. A.3, which concludes that NL-TISO outperforms TIRSO by a considerable margin for the non-linear signal models. We have also observed that the ISE of the FGD algorithm is much worse than NL-TISO and TIRSO and is not shown in the figure.

### A.4.2 Experiments using Real Data

In this section, we present experiments using real data collected from Lundin's offshore oil and gas (O&G) platform Edvard-Grieg[1]. We consider a directed graph with 24 nodes; each node corresponds to temperature (T), pressure (P), or oil-level (L) sensors. These sensors are placed in the separators of decantation tank that separates oil, gas, and water. The time series are obtained by uniformly sampling the sensor readings and applying normalization to have zero mean and unit sample

[1] https://www.lundin-energy.com/

variance. These time series are expected to exhibit causal dependencies due to the underlying physical coupling arising from the pipeline connections and the control systems.

The causal dependencies are learned using NL-TISO with a Gaussian kernel having a variance of 0.1 and with hyper parameter values $\lambda = 0.1$ and $\gamma = 10$. In Fig. A.4, we show one portion of the reconstructed signal corresponding to sensor-1, which is a pressure sensor, and it can be observed that the reconstructed signal is very close to the true sensor reading. Further, in Fig.B.6, we compare the reconstruction error of NL-TISO with TIRSO in terms of ISE for sensor-1 signal samples. We observe that NL-TISO outperforms TIRSO by a considerable margin, which supports the effectiveness of proposed algorithm in learning real world topology. The causal dependencies among the 24 time series obtained by averaging the NL-TISO estimates for one hour is shown in Fig. B.7.

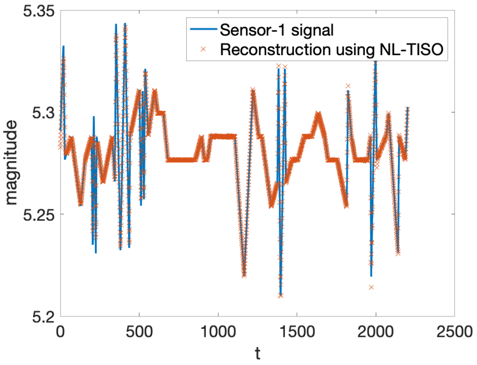


Figure A.4: Reconstruction of sensor-1 signal with sampling time 5s from Lundin data using estimated causal dependencies.

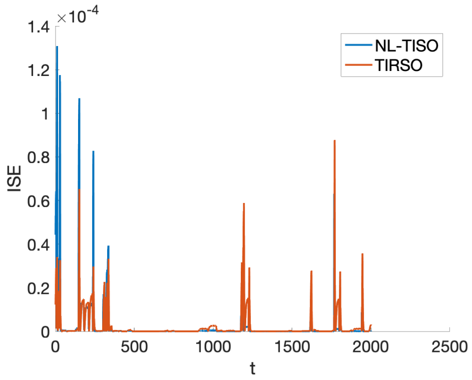


Figure A.5: ISE comparison of NL-TISO and TIRSO using real data.

## A.5 Conclusion

An online algorithm for non-linear topology identification from graph-connected time-series was proposed in this paper. Most of the state-of-the-art algorithms

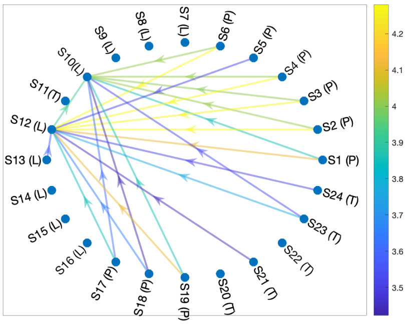


Figure A.6: Causality graph in oil and gas plant estimated by NL-TISO.

solve the topology estimation problem by assuming a linear and stationary topology. However, many real-world networks are highly dynamic and non-linear. The proposed algorithm, NL-TISO, is devised based on kernel representation to handle the non-linearities of the real-world sensor networks. Further, using a composite objective mirror descent method, NL-TISO estimates sparse topology in an online fashion aiming at dynamic system models. Qualitative and quantitative empirical evidence provided in the paper using real and synthetic data show that NL-TISO is an effective algorithm to infer the causal dependencies of real-world sensor networks. We identify two major limitations of the proposed framework: **i)** the computational complexity and memory requirements of kernel-based representations increases considerably with number of data points which is handled in NL-TISO by considering a time window to select recent samples and **ii)** the variance of the Gaussian kernels used in NL-TISO are heuristically chosen. These limitations could be handle by further research on dictionary-based multi-kernel representations, which will be devoted to our future work.

# Appendix B

# PAPER B

**Title**: Random Feature Approximation for Online Nonlinear Graph Topology Identification

**Authors**: **R. Money**, J. Krishnan, B. Beferull-Lozano

**Conference**: IEEE International Workshop on Machine Learning for Signal Processing 2021

# Random Feature Approximation for Online Nonlinear Graph Topology Identification

R. Money, J. Krishnan, B. Beferull-Lozano

**Abstract:** **Online topology estimation of graph-connected time series is challenging, especially since the causal dependencies in many real-world networks are nonlinear. In this paper, we propose a kernel-based algorithm for graph topology estimation. The algorithm uses a Fourier-based Random feature approximation to tackle the curse of dimensionality associated with the kernel representations. Exploiting the fact that the real-world networks often exhibit sparse topologies, we propose a group lasso based optimization framework, which is solve using an iterative composite objective mirror descent method, yielding an online algorithm with fixed computational complexity per iteration. The experiments conducted on real and synthetic data show that the proposed method outperforms its competitors.**

## B.1 Introduction

The amount of data generated from interconnected networks such as sensor networks, financial time-series, brain-networks, etc., are increasing rapidly. Extraction of meaningful information from such interconnected data, represented in the form of a graph can have many practical applications such as, signal denoising [62], change point detection [54], time series prediction [53], etc. Many of the functional relationships in such networks are causal and identification of this causal graph structure is termed topology identification. Many real world causal systems can be well described using vector autoregressive model (VAR) as naturally most of the dependencies are time-lagged in nature. Moreover under causal sufficiency, VAR causality implies well known Granger causality [63].

Topology identification based on the linear VAR model has been well-studied. In [48], an efficient way to estimate linear VAR coefficients from streaming data is proposed. However, such linear VAR models fail to capture the real-world nonlinear dependencies.A novel nonlinear VAR topology identification is proposed in [11] in which, the kernels are used to linearize the nonlinear dependencies by mapping them to a higher-dimensional Hilbert space. However, being a batch-based approach, [11] is computationally expensive and is not suitable for identifying the time-varying topologies.

The above shortcomings are tackled by kernel-based online algorithms [61], [21]. In [21], sparse VAR coefficients are recursively estimated using a composite objective mirror decent (COMID) approach, whereas [61] uses functional stochastic gradient descent (FSGD), followed by soft-thresholding. However, the kernel-based representations have a major drawback of unaffordable growth of computational complexity and memory requirement, which is commonly known as the “curse of dimensionality". Both [61] and [21] propose to circumvent this issue by restricting the numeric calculation to a limited number of time-series samples using a time window, which results in suboptimal performance.

A standard procedure to address the curse of dimensionality is to invoke the kernel dictionaries [64]. Often, the dictionary elements are selected based on a budget maintaining strategy. In large-scale machine learning problems, the dictionary size can go prohibitively high in order to maintain the budget. Recently, the random feature (RF) approximation [16] techniques are gaining popularity in approximating the kernels, which are shown to yield promising results compared to the budget maintaining strategies [16, 42].

In this work, we use RF approximation to avoid the curse of dimensionality in learning nonlinear VAR models. We approximate shift-invariant Gaussian kernels using a fixed number of random Fourier features. The major contributions of this paper are **i)** formulation of a kernel-based optimization framework in the function space, **ii)** reformulation of **i)** to a parametric optimization using RF approximation, and **iii)** an online algorithm to estimate the sparse nonlinear VAR coefficients using COMID updates. We provide numerical results showing the proposed method outperforms the state-of-the-art topology identification algorithms.

## B.2 Kernel Representation

Consider a multi-variate time series with $N$ nodes. Let $y_n[t]$ be the value of time series at time $t = 0, 1, \ldots, T-1$ observed at node $1 \leq n \leq N$. A $P$-th order nonlinear VAR model assuming additive functional dependencies can be formulated as

$$y_n[t] = \sum_{n'=1}^{N} \sum_{p=1}^{P} a_{n,n'}^{(p)} f_{n,n'}^{(p)}(y_{n'}[t-p]) + u_n[t], \tag{B.1}$$

where $f_{n,n'}^{(p)}$ is the function that encodes the nonlinear causal influence of the $p$-lagged data at node $n'$ on the node $n$, $a_{n,n'}^{(p)}$ is the corresponding entry of the graph adjacency matrix, and $u_n[t]$ is the observation noise. Considering the model (B.1), topology identification can be defined as the estimation of the functional dependencies $\left\{a_{n,n'}^{(p)} f_{n,n'}^{(p)}(.)\right\}_{p=1}^{P}$ for $n = 1, 2, \ldots, N$ from the observed time series $\{y_n[t]\}_{n=1}^{N}$.

We assume that the functions $f_{n,n'}^{(p)}$ in (B.1) belong to a reproducing kernel Hilbert space (RKHS):

$$\mathcal{H}_{n'}^{(p)} := \left\{ f_{n,n'}^{(p)} \mid f_{n,n'}^{(p)}(y) = \sum_{t=0}^{\infty} \beta_{n,n',t}^{(p)} \, \kappa_{n'}^{(p)}(y, y_{n'}[t-p]) \right\}, \tag{B.2}$$

where $\kappa_{n'}^{(p)} : \mathbb{R}\times\mathbb{R} \to \mathbb{R}$ is the kernel associated with the Hilbert space. The kernel measures the similarity between data points $y$ and $y_{n'}[t-p]$. Referring to (B.2), evaluation of the functional $f_{n,n'}^{(p)}$ at $y$ can be represented as the linear combination of the similarities between $y$ and the data points $\{y_{n'}[t-p]\}_{t=0}^{t=\infty}$, with weights $\beta_{n,n',t}^{(p)}$. The inner product, $\langle \kappa_{n'}^{(p)}(y,x_1), \kappa_{n'}^{(p)}(y,x_2)\rangle := \sum_{t=0}^{\infty} \kappa_{n'}^{(p)}(y[t],x_1)\kappa_{n'}^{(p)}(y[t],x_2)$, is defined in the Hilbert space using kernels with reproducible property $\langle \kappa_{n'}^{(p)}(y,x_1), \kappa_{n'}^{(p)}(y,x_2)\rangle = \kappa_{n'}^{(p)}(x_1,x_2)$. Such a Hilbert space with the reproducing kernels is termed as RKHS and the inner product induces a norm, $\|f_{n,n'}^{(p)}\|^2_{\mathcal{H}_{n'}^{(p)}} = \sum_{t=0}^{\infty}\sum_{t'=0}^{\infty} \beta_{n,n',t}^{(p)}\,\beta_{n,n',t'}^{(p)}\,\kappa_{n'}^{(p)}(y_n[t], y_n[t'])$. We refer to [37] for further reading on RKHS.

For a particular node $n$, the estimates of $\left\{f_{n,n'}^{(p)} \in \mathcal{H}_{n'}^{(p)}\right\}_{n',p}$ are obtained by solving the functional optimization problem:

$$\left\{\hat{f}_{n,n'}^{(p)}\right\}_{n',p} = \arg \min_{\left\{f_{n,n'}^{(p)} \in \mathcal{H}_{n'}^{(p)}\right\}} \frac{1}{2}\sum_{\tau=P}^{T-1}\left[y_n[\tau] - \sum_{n'=1}^{N}\sum_{p=1}^{P} a_{n,n'}^{(p)} f_{n,n'}^{(p)}(y_{n'}[\tau-p])\right]^2. \tag{B.3}$$

It is to be noted that in (B.3), the functions $\{f_{n,n'}^{(p)}\}$ belong to the RKHS defined in (B.2), which is an infinite dimensional space. However, by resorting to the Representer Theorem [38], the solution of (B.3) can be written using a finite number of data samples:

$$\hat{f}_{n,n'}^{(p)}(y_{n'}[\tau-p]) = \sum_{t=p}^{p+T-1} \beta_{n,n',(t-p)}^{(p)} \kappa_{n'}^{(p)}(y_{n'}[\tau-p]), y_{n'}[t-p]). \tag{B.4}$$

Notice that the number of coefficients required to express the function increases with the number of data samples. In the recent works [61], [21], this problem is solved by using a time window to fix the number of data points, resulting in suboptimality. However, in this work in align with [42] and [43], we use RF approximation to tackle the dimensionality growth.

## B.3 Random Feature Approximation

To invoke RF approximation, we assume the kernel to be shift-invariant, i.e., it satisfies the property $\kappa_{n'}^{(p)}(y_{n'}[\tau-p]), y_{n'}[t-p]) = \kappa_{n'}^{(p)}(y_{n'}[\tau-p]) - y_{n'}[t-p])$. Bochner's theorem [30] states that every shift-invariant kernel can be represented as an inverse Fourier transform of a probability distribution. Hence the kernel evaluation can be expressed as

$$\begin{aligned}\kappa_{n'}^{(p)}(y_{n'}[\tau-p]), y_{n'}[t-p]) &= \int \pi_{\kappa_{n'}^{(p)}}(v)\; e^{jv(y_{n'}[\tau-p]-y_{n'}[t-p])}dv \\ &= \mathbb{E}_v[e^{jv(y_{n'}[\tau-p]-y_{n'}[t-p])}],\end{aligned} \tag{B.5}$$

where $\pi_{\kappa_{n'}^{(p)}}(v)$ is the probability density function which depends on type of the kernel, and $v$ is the random variable associated with it. If sufficient amount of iid samples $\{v_i\}_{i=1}^{D}$ are collected from the distribution $\pi_{\kappa_{n'}^{(p)}}(v)$, the real ensemble mean in (B.5) can be expressed as a sample mean:

$$\hat{\kappa}_{n'}^{(p)}(y_{n'}[\tau-p]), y_{n'}[t-p]) = \frac{1}{D}\sum_{i=1}^{D} e^{jv_i(y_{n'}[\tau-p])-y_{n'}[t-p])}, \tag{B.6}$$

irrespective of the distribution $\pi_{\kappa^{(p)}_{n'}}(v)$. Note that the unbiased estimate of kernel evaluation in (B.6) involves a summation of fixed $D$ number of terms. In general, computing the probability distribution corresponding to a kernel is a difficult task. In this work the kernel under consideration is Gaussian; for a Gaussian kernel $k_\sigma$ with variance $\sigma^2$, it is well known that the Fourier transform is a Gaussian with variance $\sigma^{-2}$. Considering the real part of (B.6), which is also an unbiased estimator, (B.5) can be approximated as

$$\hat{\kappa}^{(p)}_{n'}\left(y_{n'}[\tau-p], y_{n'}[t-p]\right) = \boldsymbol{z_v}\left(y_{n'}[\tau-p]\right)^\top \boldsymbol{z_v}\left(y_{n'}[t-p]\right), \tag{B.7}$$

$$\textit{where } \boldsymbol{z_v}(x) = \frac{1}{\sqrt{D}}[\sin v_1 x, \ldots, \sin v_D x, \cos v_1 x, \ldots, \cos v_D x]^\top. \tag{B.8}$$

Subsisting (B.7) in (B.4), we obtain a fixed dimension ($2D$ terms) approximation of the function $\hat{f}^{(p)}_{n,n'}$:

$$\begin{aligned}\hat{f}^{(p)}_{n,n'}\left(y_{n'}[\tau-p]\right)) &= \sum_{t=p}^{p+T-1} \beta^{(p)}_{n,n',(t-p)} \boldsymbol{z_v}\left(y_{n'}[\tau-p]\right)^\top \boldsymbol{z_v}\left(y_{n'}[t-p]\right) \\ &= \boldsymbol{\theta}^{(p)\top}_{n,n'} \boldsymbol{z_v}\left(y_{n'}[\tau-p]\right), \end{aligned} \tag{B.9}$$

where $\boldsymbol{\theta}^{(p)\top}_{n,n'} = \sum_{t=p}^{p+T-1} \beta^{(p)}_{n,n',(t-p)} \boldsymbol{z_v}\left(y_{n'}[\tau-p]\right)^\top = [\theta^{(p)}_{n,n',1}, \ldots, \theta^{(p)}_{n,n',2D}] \in \mathbb{R}^{2D}$. For the sake of clarity, in the succeeding steps, we define the following notation:

$$\boldsymbol{\alpha}^{(p)}_{n,n'} = [\alpha^{(p)}_{n,n',1}, \ldots, \alpha^{(p)}_{n,n',2D}]^\top \in \mathbb{R}^{2D}, \tag{B.10}$$

$$\boldsymbol{z_v}\left(y_{n'}[\tau-p]\right) = [z^{(p)}_{n',1}(\tau), \ldots z^{(p)}_{n',2D}(\tau)]^\top \in \mathbb{R}^{2D}, \tag{B.11}$$

where $\alpha^{(p)}_{n,n',d} = \theta^{(p)}_{n,n',d} a^{(p)}_{n,n'}$. The functional optimization (B.3) is reformulated as a parametric optimization problem using (B.9):

$$\left\{\widehat{\alpha}^{(p)}_{n,n',d}\right\}_{n',p,d} = \arg \min_{\left\{\alpha^{(p)}_{n,n',d}\right\}} \mathcal{L}^n\left(\alpha^{(p)}_{n,n',d}\right), \tag{B.12}$$

where

$$\mathcal{L}^n\left(\alpha^{(p)}_{n,n',d}\right) := \sum_{\tau=P}^{T-1} \frac{1}{2}\left[y_n[\tau] - \sum_{n'=1}^{N}\sum_{p=1}^{P}\sum_{d=1}^{2D} \alpha^{(p)}_{n,n',d}\ z^{(p)}_{n',d}(\tau)\right]^2. \tag{B.13}$$

For convenience, optimization parameters $\left\{\alpha^{(p)}_{n,n',d}\right\}$ and $\left\{z^{(p)}_{n',d}(\tau)\right\}$ are stacked in the lexicographic order of the indices $p$, $n'$, and $d$ to obtain the vectors $\boldsymbol{\alpha}_n \in \mathbb{R}^{2PND}$ and $\boldsymbol{z}_\tau \in \mathbb{R}^{2PND}$, respectively, and (B.12) is rewritten as

$$\widehat{\boldsymbol{\alpha}}_n = \arg\min_{\boldsymbol{\alpha}_n} \mathcal{L}^n\left(\boldsymbol{\alpha}_n\right), \tag{B.14}$$

$$\text{where } \ \mathcal{L}^n(\boldsymbol{\alpha}_n) = \frac{1}{2}\sum_{\tau=P}^{T-1}\left[y_n[\tau] - \boldsymbol{\alpha}_n^\top \boldsymbol{z}_\tau\right]^2 \tag{B.15}$$

Now, in order to avoid overfitting, we propose a regularized optimization framework:

$$\widehat{\boldsymbol{\alpha}}_n = \arg\min_{\boldsymbol{\alpha}_n} \mathcal{L}^n\left(\boldsymbol{\alpha}_n\right) + \lambda \sum_{n'=1}^{N}\sum_{p=1}^{P} \|\boldsymbol{\alpha}^{(p)}_{n,n'}\|_2, \tag{B.16}$$

where $\lambda \geq 0$ is the regularization parameter and $\boldsymbol{\alpha}_{n,n'}^{(p)} = (\alpha_{n,n',1}^{(p)}, \alpha_{n,n',2}^{(p)}, \ldots, \alpha_{n,n',2D}^{(p)}) \in \mathbb{R}^{2D}$. The second term in (B.16) is a *group-lasso* regularizer, which promote a *group-sparse structure* in $\boldsymbol{\alpha}_{n,n'}^{(p)}$, supported by the assumption that most of the real world dependencies are sparse in nature.

However, notice that the batch formulation in (B.16) has some significant limitations: **i)** requirement of complete batch of data points before estimation, **ii)** inability to track time varying topologies, and **iii)** explosive computational complexity when $T$ is large even if RF approximation is used. To mitigate these problems, we adopt an online optimization strategy, which is explained in the following section.

## B.4 Online Topology Estimation

In this case, we replace the batch loss function $\mathcal{L}^n(\boldsymbol{\alpha}_n)$ in (B.16) with the stochastic (instantaneous) loss function $l_t^n(\boldsymbol{\alpha}_n) = \frac{1}{2}[y_n[t] - \boldsymbol{\alpha}_n^\top \boldsymbol{z}_t]^2$:

$$\widehat{\boldsymbol{\alpha}}_n = \arg\min_{\boldsymbol{\alpha}_n} l_t^n\left(\boldsymbol{\alpha}_n\right) + \lambda \sum_{n'=1}^{N} \sum_{p=1}^{P} \|\boldsymbol{\alpha}_{n,n'}^{(p)}\|_2. \tag{B.17}$$

Notice that the sparsity promoting group lasso regularizer is non-differentiable. The use of online subgradient descent (OSGD) is not advisable in this situation as it linearizes the entire objective function and fails to provide sparse iterates. To avoid this limitation of OSGD, we use the composite objective mirror descent (COMID) [39] algorithm which resembles the nature of proximal methods, hence improving convergence. The online COMID update can be written as

$$\boldsymbol{\alpha}_n[t+1] = \arg\min_{\boldsymbol{\alpha}_n} J_t^{(n)}(\boldsymbol{\alpha}_n), \tag{B.18}$$

$$\text{where } J_t^{(n)}(\boldsymbol{\alpha}_n) \triangleq \nabla \ell_t^n(\boldsymbol{\alpha}_n[t])^\top \left(\boldsymbol{\alpha}_n - \boldsymbol{\alpha}_n[t]\right) + \frac{1}{2\gamma_t}\|\boldsymbol{\alpha}_n - \boldsymbol{\alpha}_n[t]\|_2^2 + \lambda \sum_{n'=1}^{N} \sum_{p=1}^{P} \|\boldsymbol{\alpha}_{n,n'}^{(p)}\|_2. \tag{B.19}$$

In (B.19) $\boldsymbol{\alpha}_n[t] \in \mathbb{R}^{2PND}$ denotes the estimate of $\boldsymbol{\alpha}_n$ at time $t$. The first term in equation (B.19) is the gradient of the loss function $l_t^n(\boldsymbol{\alpha}_n)$, the second and third term are Bergman divergence and sparsity promoting regularizer respectively. The Bregman divergence is included to improve the stability of algorithm from adversaries by constraining $\boldsymbol{\alpha}_n[t+1]$ to be close to $\boldsymbol{\alpha}_n[t]$. The Bregman divergence $B(\boldsymbol{\alpha}_n, \boldsymbol{\alpha}_n[t]) = \frac{1}{2}\|\boldsymbol{\alpha}_n - \boldsymbol{\alpha}_n[t]\|_2^2$ chosen in such a way that the COMID update has a closed form solution [40] and $\gamma_t$ is the corresponding step size. The gradient in (B.19) is evaluated as

$$\mathbf{v}_n[t] := \nabla \ell_t^n(\boldsymbol{\alpha}_n[t]) = \boldsymbol{z}_t\left(\boldsymbol{\alpha}_n^\top \boldsymbol{z}_t - y_n[t]\right) \tag{B.20}$$

Expanding the objective function in (B.19) and omitting the constants leads to the following formulation:

$$
\begin{aligned}
J_t^{(n)}(\boldsymbol{\alpha}_n) &\propto \frac{\boldsymbol{\alpha}_n^\top \boldsymbol{\alpha}_n}{2\gamma_t} + \boldsymbol{\alpha}_n^\top \left(\mathbf{v}_n[t] - \frac{1}{\gamma_t}\boldsymbol{\alpha}_n[t]\right) + \lambda \sum_{n'=1}^{N}\sum_{p=1}^{P} \|\boldsymbol{\alpha}_{n,n'}^{(p)}\|_2 \\
&= \sum_{n'=1}^{N}\sum_{p=1}^{P} \Bigg[ \frac{\boldsymbol{\alpha}_{n,n'}^{(p)^\top} \boldsymbol{\alpha}_{n,n'}^{(p)}}{2\gamma_t} + \boldsymbol{\alpha}_{n,n'}^{(p)^\top} \left( \mathbf{v}_{n,n'}^{(p)}[t] - \frac{1}{\gamma_t}\boldsymbol{\alpha}_{n,n'}^{(p)}[t] \right) \\
&\qquad\qquad + \lambda \|\boldsymbol{\alpha}_{n,n'}^{(p)}\|_2 \Bigg]. \qquad \text{(B.21)}
\end{aligned}
$$

A closed form solution for (B.18) using (B.21) is obtained via the multidimensional shrinkage-thresholding operator [41]:

$$
\begin{aligned}
\boldsymbol{\alpha}_{n,n'}^{(p)}[t+1] = &\left(\boldsymbol{\alpha}_{n,n'}^{(p)}[t] - \gamma_t \mathbf{v}_{n,n'}^{(p)}[t]\right) \times \\
&\left[1 - \frac{\gamma_t \lambda}{\|\boldsymbol{\alpha}_{n,n'}^{(p)}[t] - \gamma_t \mathbf{v}_{n,n'}^{(p)}[t]\|_2}\right]_+, \qquad \text{(B.22)}
\end{aligned}
$$

where $[x]_+ = \max\{0, x\}$. The first term $\boldsymbol{\alpha}_{n,n'}^{(p)}[t] - \gamma_t \mathbf{v}_{n,n'}^{(p)}[t]$ in (B.22) forces the stochastic gradient update of $\boldsymbol{\alpha}_{n,n'}^{(p)}$ in a way to descend instantaneous loss function $l_t^n(\boldsymbol{\alpha}_n)$ and the second term in (B.22) enforces group sparsity of $\boldsymbol{\alpha}_{n,n'}^{(p)}$. Note that the close form solution (B.22) is separable in $n'$ and $p$.

The proposed algorithm, termed as *Random Feature based Nonlinear Topology Identification via Sparse Online learning* (RF-NLTISO), is summarized in **Algorithm 6**.

**Algorithm 6:** RF-NLTISO Algorithm

```
Result: $\boldsymbol{\alpha}_{n,n'}^{(p)}, for\ n, n' = 1, .., N$ and $p = 1, .., P$
Store $\{\boldsymbol{y}_n[t]\}_{t=1}^{P}$,
Initialize $\lambda$, $\gamma$, $D$ (heuristically chosen) and kernel parameters depending
  on the type of the kernel.
for $t = P, P+1, \dots$ do
    Get data samples $y_n[t]$, $\forall n$ and compute $\boldsymbol{z}_\tau$
    for $n = 1, \dots, N$ do
        compute $\mathbf{v}_n[t]$ using (B.20)
        for $n' = 1, \dots, N$ do
            compute $\boldsymbol{\alpha}_{n,n'}^{(p)}[t+1]$ using (B.22)
        end
    end
end
```

## B.5 Experiments

We compare the performance of the proposed algorithm, RF-NLTISO, with the the state-of-the-art online topology estimation algorithms. Experiments shown in this section are conducted using 1) synthetic datasets with topologies having different transition patterns and 2) real datasets collected from Lundin's offshore oil and Gas platform. For the performance comparison, we choose TIRSO [48] and NL-TISO [21] algorithms, which are the state-of-the-art counterparts of RF-NLTISO, to the best of our knowledge. TIRSO is developed based on a linear VAR model assumption, whereas NL-TISO, a kernel-based topology estimation algorithm, is developed for nonlinear VAR models. Although a kernel-based functional stochastic gradient based algorithm [61] is also available, its performance has been shown to be inferior compared to NL-TISO [21].

## B.5.1 Experiments using Synthetic Data

### B.5.1.1 Topology with switching edges

We generate a multi-variate time series using nonlinear VAR model (B.1) with $N = 5, P = 2$. An initial random graph with edge probability of 0.1 is generated and the graph adjacency coefficients $a_{n,n'}^{(p)}$ are drawn from a Uniform distribution $\mathcal{U}(0,\,1)$. After every 1000 samples, one of the active (non-zero) edge disappears and another one appears randomly, which brings an abrupt change in the graph topology. The nonlinearity in (B.1) is introduced using a Gaussian kernel with variance 0.01 and the kernel coefficients are chosen randomly from a zero mean Gaussian distribution with variance 30. Note that the initial $P$ data samples are generated randomly and rest of the data is generated using the model (B.1). The coefficients $\left\{\boldsymbol{\alpha}_{n,n'}^{(p)}[t]\right\}$ are estimated using the proposed RF-NLTISO algorithm with a Gaussian kernel having variance 0.1 and number of random features $D = 50$. The hyper-parameters $\lambda$ and $\gamma$ are heuristically chosen as 0.1 and 1000, respectively. To visualize causal relationships, we compute the $\ell_2$ norms $b_{n,n'}^{(p)}[t] = \|\boldsymbol{\alpha}_{n,n'}^{(p)}[t]\|_2$ and arrange them in a matrix similar to the graph adjacency matrix. A similar strategy is adopted for the NL-TISO and the TIRSO algorithms. The normalized version of true and the estimated dependencies at various time samples are shown in Fig. B.1, where in each subplot, the $5 \times 5$ dependency matrices corresponding to $p = 1$ and 2 are concatenated, resulting in a $10 \times 5$ size matrix. We normalized the coefficients by dividing each coefficients with highest value of coefficient in a pseudo adjacency matrix. From the Fig. B.1, it is clear that RF-NLTISO is able to perform equal or better compared to NL-TISO algorithm and clearly outperforms TIRSO.

Next we conduct the same experiments using RF-NLTISO with different numbers of random feature ($D \in \{10, 30, 50\}$). These experiments are repeated 1000 times to find probability of miss detection ($P_{\mathrm{MD}}$) and false alarm ($P_{\mathrm{FA}}$), which we define as

$$\begin{aligned} P_{\mathrm{MD}}[t] &\triangleq \frac{\sum_{n\neq n'}\sum_{p=1}^{P}\mathbb{E}\left[\nVdash\{\|\widehat{\boldsymbol{b}}_{n,n'}^{(p)}[t]\|_2 < \delta\}\nVdash\{\left\|\boldsymbol{\alpha}_{n,n'}\right\|_2 \geq \delta\}\right]}{\sum_{n\neq n'}\sum_{p=1}^{P}\mathbb{E}\left[\nVdash\{\left\|\boldsymbol{\alpha}_{n,n'}\right\|_2 \geq \delta\}\right]}, \\ P_{\mathrm{FA}}[t] &\triangleq \frac{\sum_{n\neq n'}\sum_{p=1}^{P}\mathbb{E}\left[\nVdash\{\|\widehat{\boldsymbol{b}}_{n,n'}^{(p)}[t]\|_2 > \delta\}\nVdash\{\left\|\boldsymbol{\alpha}_{n,n'}\right\|_2 \leq \delta\}\right]}{\sum_{n\neq n'}\sum_{p=1}^{P}\mathbb{E}\left[\nVdash\{\left\|\boldsymbol{\alpha}_{n,n'}\right\|_2 \leq \delta\}\right]}. \end{aligned} \tag{B.23}$$

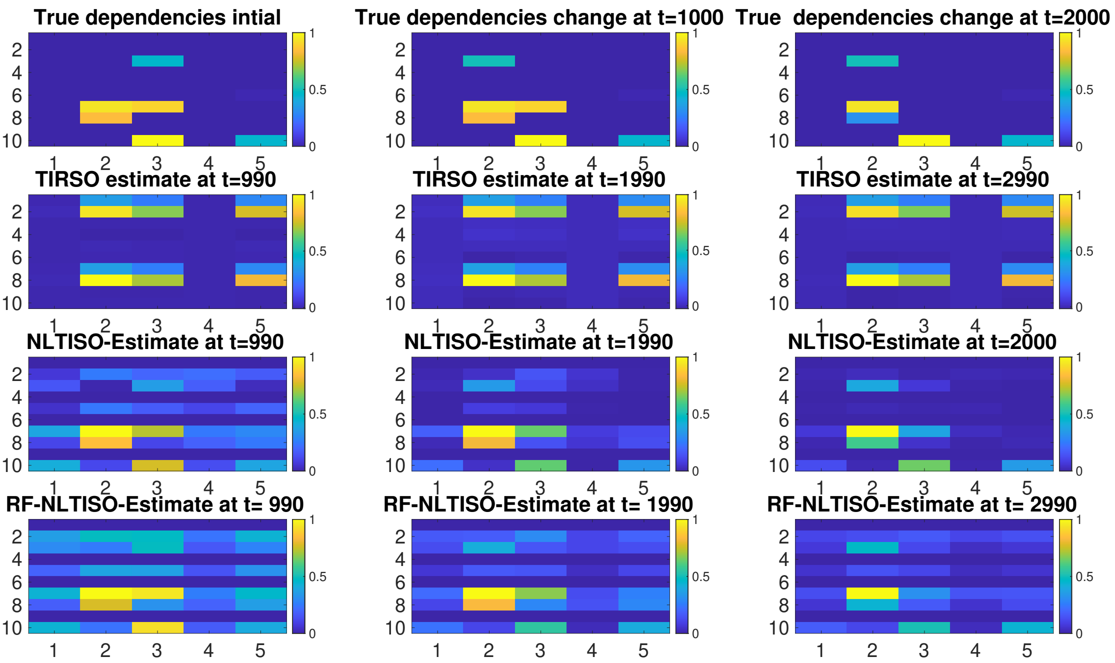


Figure B.1: Causal dependencies estimated using different algorithms compared with the true dependency.

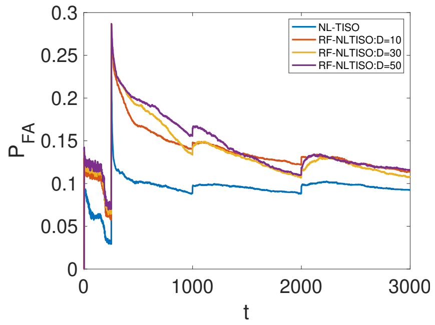


Figure B.2: Probability of False Alarm ($P_{FA}$).

0.12

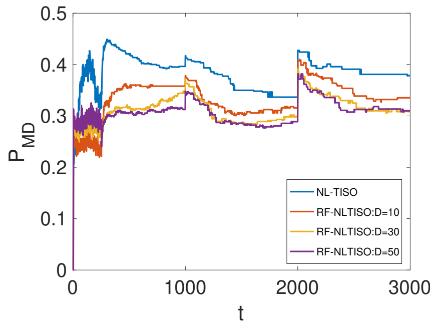
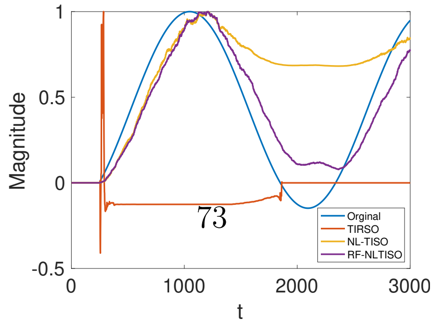


Figure B.3: Probability of Miss Detection ($P_{MD}$).



From Figs. B.2 and B.3, it is observed that for the given choices of $D$, $P_{MD}$ is better for RF-NLTISO compared to NL-TISO; however, NL-TISO is performing better in terms of $P_{FA}$. Both the figures show an overshoot at the topology-switching time instances. It is also observed that for the proposed algorithm, $P_{MD}$ decreases with $D$, whereas $P_{FA}$ increases with $D$, which in turn suggests a tuning for $D$ for an effective trade-off between $P_{FA}$ and $P_{MD}$.

#### B.5.1.2 Slowly varying topology

We compare the performance of RF-NLTISO with the state-of-the-art algorithms using a slowly varying graph topology. The same experiment setup as discussed in B.5.1.1 is adopted with the following more slowly time varying topology:

$$a_{n,n'}^{(p)}[t+1] = a_{n,n'}^{(p)}[t] + 0.01 \sin(0.03 * t) \tag{B.24}$$

The normalized values of one of the active edges is plotted in Fig. B.4. The figure also shows the normalized values of the corresponding estimated coefficients ($\widehat{b}_{n,n'}^{(p)}[t]$ for NL-TISO and RF-NLTISO and $\widehat{a}_{n,n'}^{(p)}[t]$ for TIRSO ). From the figure, it can be observed that the RF-NLTISO estimates are closer to the true value compared to the estimates from the other two algorithms. In this example, the quality of TIRSO estimates lags considerably behind the kernel-based algorithms due to the fact that the underlying VAR model is nonlinear.

### B.5.2 Experiments using Real Data

This section is dedicated to experiments using real data collected from Lundin's offshore oil and gas (O&G) platform Edvard-Grieg[1]. We have a multi-variate time series with 24 nodes; and the nodes corresponds to various temperature (T), pressure (P), or oil-level (L) sensors. The sensors are placed in the separators of decantation tanks that separate oil, gas, and water. The time series are obtained by uniformly sampling the sensor readings with a sampling rate of $5s$. We assume that hidden logic dependencies are present in the network due to various physical connections and various control actuators. The data obtained from the network is normalized by making it a zero mean unit variance signal, before applying the algorithm. The causal dependencies are learned using RF-NLTISO with $D = 10, 50, 100$ and a Gaussian kernel having a variance of 0.1 and with hyper parameter values $\lambda = 0.1$ and $\gamma = 10$. The signal is reconstructed using the estimated dependencies. Fig. B.6 shows the mean squared error ($MSE$), defined as $MSE(t) = \mathbb{E}((y_n(t) - \hat{y}_n(t))^2)$ for a particular sensor $n = 8$, of RF-NLTISO estimates in comparison with other algorithms. We observe that the RF-NLTISO estimates with random feature number $D \geq 50$ show better $MSE$ performance compare to NL-TISO. The causality graph estimated by RF-NLTISO is shown in Fig. B.7.

One of the main attractiveness of RF-NLTISO is that even though it is a kernel-based algorithm, it has a fixed computational complexity throughout the online

[1] https://www.lundin-energy.com/

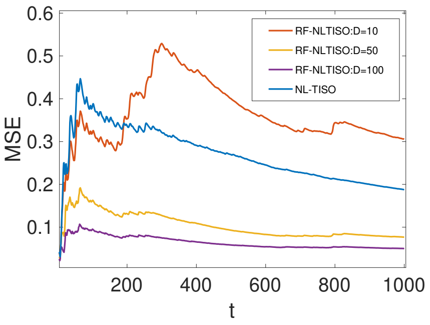


Figure B.6: MSE comparison of NL-TISO with RF-NLTISO.

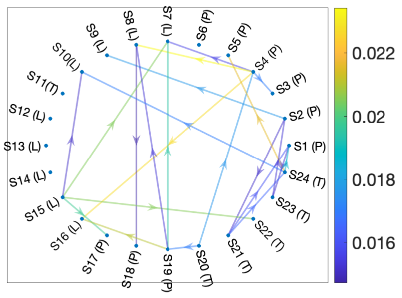


Figure B.7: Causality graph in oil and gas plant estimated by RF-NLTISO. P, T, L represent pressure, temperature, and oil level sensors, respectively.

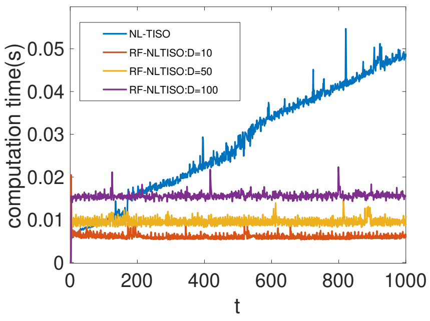


Figure B.8: Comparison of computation time of kernel-based algorithms.

Figure B.9: Real data.

iterations. To demonstrate this, in Fig. B.8, we plot the computation time required to estimate the coefficients at each time instant by NL-TISO and RF-NLTISO with different values of $D$. The experiment is conducted in a machine with processor 2.4 GHz 8-core Intel Core i9 and 16GB 2667 MHz DDR4 RAM. Fig. B.8 shows that the computation time of NL-TISO increases considerably with time but that of RF-NLTISO remains more or less constant for a particular value of $D$.

## B.6 Conclusion

We propose a kernel-based online topology identification method for interconnected networks of time-series with additive nonlinear dependencies. In this work, the curse of dimensionality associated with kernel representation is tackled using random feature approximation. Assuming that the real-world dependencies are sparse, we use composite objective mirror decent update to estimate the online sparse causality graph. The effectiveness of the proposed algorithm is illustrated through experiments conducted on synthetic and real data, which shows that the algorithm outperforms the state-of-the-art competitors. We devote the convergence and stability analysis of the proposed algorithm to our future work.

# Appendix C

# PAPER C

**Title**: Sparse Online Learning with Kernels using Random Features for Estimating Nonlinear Dynamic Graphs

**Authors**: **R. Money**, J. Krishnan, B. Beferull-Lozano

**Journal**: IEEE Transactions on Signal Processing 2023 (Accepted with minor revision)

# Sparse Online Learning with Kernels using Random Features for Estimating Nonlinear Dynamic Graphs

R. Money, J. Krishnan, B. Beferull-Lozano

**Abstract:** **Online topology estimation of graph-connected time series is challenging in practice, especially because the causal dependencies between the time-series in many real-world scenarios are nonlinear. In this paper, we propose an online kernel-based algorithm for graph topology estimation. The algorithm also performs a Fourier-based Random feature approximation to tackle the curse of dimensionality associated with the kernel representations. Exploiting the fact that the real-world networks often exhibit sparse topologies, we propose a group-Lasso based optimization framework, which is solved using an iterative composite objective mirror descent method, yielding an online algorithm with fixed computational complexity per iteration. We provide theoretical guarantees for the proposed algorithm and prove that the algorithm can achieve sublinear dynamic regret under certain reasonable assumptions. The experiments on real and synthetic data show that the proposed method outperforms its state-of-the-art competitors.**

## C.1 Introduction

Many practical networks such as large scale cyber-physical systems (CPS), financial networks, brain networks, etc., generate multivariate time series data. In such systems, the time series are interdependent and it is possible to represent the dependencies in the form of graphs, or we can say that the multivariate time series is graph connected. Some of these dependencies are often imperceptible by direct inspection. Inferring and exploiting the hidden graph structure of data can have a significant impact in many application fields. For instance, it can contribute to developing better control actions in CPS [65], explainable analysis in brain networks [66], and better forecast in financial time series [67], to name a few.

Real-world networks often exhibit time-delayed and directed dependencies between their components. For instance, consider an example of an oil and gas processing platform, as shown in Fig. C.1. The system consists of wells and separators. The raw oil is extracted from the well and is separated as oil, water, and gas in the separators. It is a highly dynamic and complex system with hundreds of sensors and

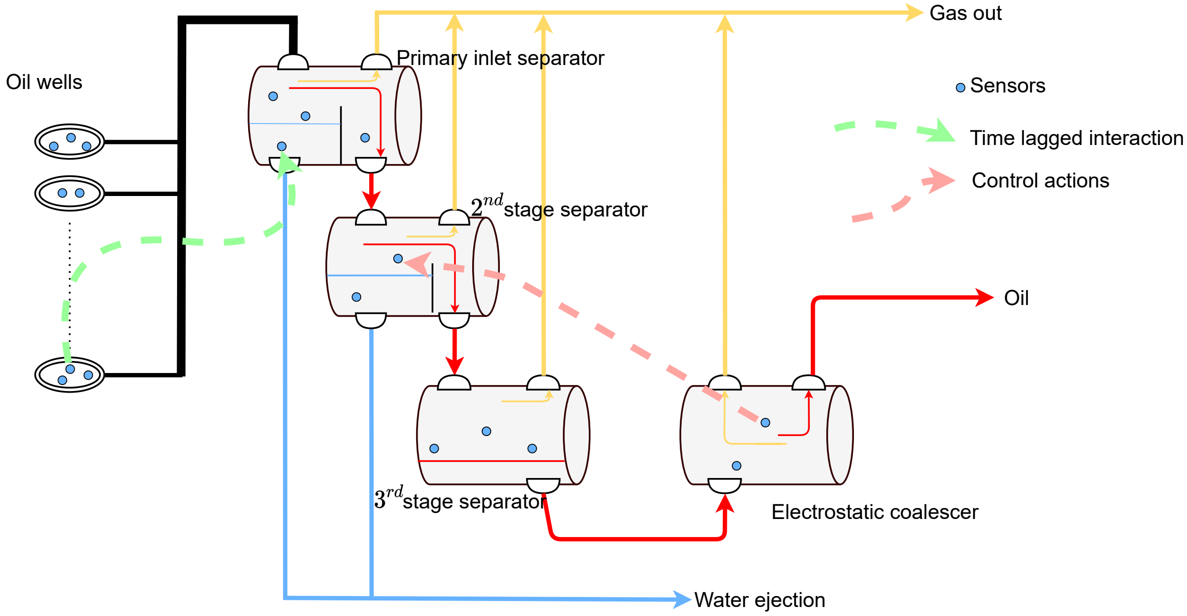


Figure C.1: Schematic of processing stages in an oil and gas platform.

actuators. If an event occurs in a well, its effect will be reflected in the separators after a delay. Similarly, the oil level in separator-2 depends on the pressure that is controlled by an actuator in separator-3. The data acquired from such a system form a multivariate time series, possibly having many directed time-lagged interactions, which can be represented using a graph structure. Any information related to these dependencies is highly beneficial since it helps to predict the evolution of sensor variables in the near future and the appropriate control actions in advance. Although a scenario related to the oil and gas platform is adopted here for illustration, such interactions have a vital role in many important networks, such as brain data, the stock market, and smart water networks (SWN), to name a few. Hereafter, we use the term *topology identification* to denote the estimation of such dependencies.

A significant challenge associated with the aforementioned real-world graph-connected networks is the non-stationary nature of the causal dependencies. There is extensive research on the field of online learning [68], [69], which outperforms classical batch solutions in terms of both computational complexity and ability to track changes. Such methods can be exploited and applied to topology identification in order to mitigate the problem of time varying dependencies. For instance, [48] proposes a sparse online solution for topology identification using proximal updates, whereas [49] introduces a prediction-correction algorithm based on a time-varying convex optimization framework that exhibits an intrinsic temporal-regularization of the graph topology.

In addition to the non-stationary nature, real-world systems such as the one shown in Fig. C.1, are further complicated due to the nonlinear nature of the dependencies. In CPSs such as Oil and Gas platforms or SWNs, this nonlinearity may arise from control mechanisms of the actuator, nonlinear liquid flows (see, e.g., [70]), saturation of tanks, etc. Similarly, the interactions in stock market networks and network structured data related to brain imaging techniques, such as electroencephalography (EEG), electrocorticography (ECoG), positron emission tomography (PET), etc., also exhibit a high level of nonlinearities [13]. In such applications, topology estimation based on simple linear models [48], [49] is inadequate, since many of the inherent nonlinear interactions within the system are discarded.

An effective way to deal with the nonlinearity is by invoking kernel machines, which can approximate any nonlinear continuous function, provided enough training samples are available. For instance, in [11], a novel topology identification algorithm based on the nonlinear structural vector auto-regressive (SVAR) model using kernels is proposed. On the other hand, deep neural networks (DNNs) are powerful alternatives to kernels for modelling nonlinear interactions.Nonlinear dependencies are estimated in [71] using a temporal convolutional neural network and an attention mechanism, while [10] uses a vector autoregressive (VAR) model with an invertible neural network approach to capture dependencies, and [9] applies a group-Lasso regularizer on neural weights to obtain sparse nonlinear dependencies. Although the above-mentioned kernel- and DNN-based methods are powerful tools to model the nonlinear dependencies, their batch-based (offline) nature makes them unsuitable for real-time applications that require online topology estimation with every new data sample to track changes in the system. In addition, such batch-based approaches also suffer from a high computational complexity since the algorithm must process the entire data batch together.

The above discussion motivates the need for algorithms that can learn nonlinear and dynamic topologies. Kernels are an ideal choice in this regard due to their interpretability and capability to learn functions online [21,72,73]. In kernel frameworks, the data points are transformed into a function space, where a linear relationship exists between them. However, working in a function space has some limitations in the context of online topology identification. First, the standard online convex optimization techniques cannot be readily used as the dimension of optimization variables is not fixed, and it increases with every new data sample. Second, the number of parameters required to express the function increases with the number of data samples, and the computational complexity becomes prohibitive at some point, which is typically known as the curse of dimensionality [74]. This dimensionality growth is circumvented in [21] by discarding the past data samples using a forgetting window. However, such an approach can lead to suboptimal function learning because it discards data samples without assessing their significance in representing the functions to be learned.

Sparse kernel dictionaries and random feature (RF) approximation are two popular techniques for tackling the curse of dimensionality associated with kernels. A parsimonious online learning algorithm for kernels has been developed in [64] using a functional stochastic gradient descent (FSGD) method featured by sparse function subspace projections. This is achieved by learning sparse kernel dictionaries using the kernel orthogonal matching pursuit (KOMP) technique. Despite its reported benefits [64] in terms of model complexity compared to RF-based techniques, the sparse FSGD method in [64] has two limitations that render it an unfitting choice for online topology identification of multivariate time series: *i*) the algorithm need to include several KOMP sub-iterations for every time series at each time instant, which results in high computational complexity, not being suitable for online algorithms, particularly when the number of time series exceeds a few hundred, as it is typical in real-world networks such as the one shown in Fig. C.1, and *ii*) in a multivariate setting with $N$ time series, the FSGD derivation in [64] results in identical functional dependencies between a time series $n$ and all other time series $n' = 1, 2, \ldots, N$ (as observed in [61]), which prevents distinguishing the different functional dependencies. In [75], an alternative approach to reduce the dimensionality growth of the kernel method for multivariate topology inference is presented, which involves learning a sparse kernel dictionary based on coherence criteria. Nevertheless, this algorithm's convergence guarantees assume that optimal parameters (representing the topology) do not change over time, which is impractical for time-varying systems.

On the other hand, the RF approximation approach not only addresses the problem of kernel dimensionality growth but also provides greater mathematical flexibility for modelling and learning the nonlinear interaction among multivariate time series, in addition to enabling a theoretical analysis. RF approximation was originally proposed in [16], and the idea has recently gained popularity in large-scale machine learning problems [42, 43, 76]. In addition to providing a computational boost in large-scale data sets, RF allows working in fixed lower dimensional spaces, which is very convenient for many online convex optimization routines. It has been shown that the RF approximation in kernels can be also used to understand neural networks [77], [78], and some researchers have shown equivalence in function approximation between neural networks and RF approximations [77]. Multiple Random Fourier features can be also utilized to initialize the learning process, and the best one can be kept to avoid overfitting [79, 80].

In this work, we propose a kernel-based online nonlinear topology identification algorithm using RF approximation. We assume that the dependencies of the system can be modelled using nonlinear additive sparse model. Notice that the sparsity assumption is not restrictive, since the interactions in real-world systems are often sparse due to the dominant local interactions. In fact, this prior information helps to avoid overfitting during learning. The proposed algorithm estimates nonlinear topologies in an online manner by generating sparse iterates at each time instant, using a proximal optimization technique known as Composite objective mirror descent (COMID). The algorithm features incremental updates to the model upon the arrival of new data samples, making it suitable for applications characterized by topology drifts [81, 82]. Through a combination of theoretical guarantees based on dynamic regret analysis and multiple numerical evidence, we show the effectiveness of our algorithm in tracking the changes in topology.

The main contributions of this work are listed below:

(**i**) This paper proposes an online algorithm with fixed computational complexity per iteration for nonlinear topology estimation. The proposed algorithm is termed *Random feature based nonlinear topology identification via recursive sparse online learning* (RFNL-TIRSO). This work is significantly different from our previous work in [22], where we used an instantaneous loss function, which is susceptible to noise and converges slowly. RFNL-TIRSO replaces the instantaneous loss function with an average running loss inspired by recursive least square (RLS) formulation, and compared to [22], it significantly improves convergence speed and robustness to the input noise.

(**ii**) We also provide theoretical guarantees regarding the convergence of RFNL-TIRSO, whereas no such theoretical guarantees were provided in [22]. The paper derives an upper bound for dynamic regret of RFNL-TIRSO based on strong convexity of the RLS loss function. Dynamic regret characterizes the tracking capability of an online algorithm [83], and we achieve a sublinear dynamic regret under certain assumptions that are reasonable in real-world applications. Our dynamic regret analysis includes three key elements: an online kernel-based nonlinear algorithm, a non-differentiable objective function, and a model with multiple decoupled functions representing topological connections to enable interpretable topology identification. None of the existing related analyses [32, 84–89] provides a complete coverage of all these three elements.

(**iii**) The performance of the proposed algorithm is tested with extensive experiments using both real and synthetic data. The algorithm estimates interpretable topologies using time series data collected from the sensors of an oil and gas plant. In addition to the CPS applications, we also demonstrate the capability of our algorithm in detecting epileptic seizure events using EEG signals.

The rest of the paper is organized as follows: Section C.2 presents the system model, kernel formulation, and random feature approximation. In Appendix C.3, we develop the RFNL-TIRSO algorithm. Theoretical analysis of RFNL-TIRSO is performed in Appendix C.4 and the numerical results are provided in Appendix C.5. Section C.6 concludes the paper.

Notations: Bold lowercase and uppercase letters denote column vectors and matrices, respectively. The operators $\nabla$, $(.)^\top$, $\mathbb{E}$, $\Lambda_{max}(.)$, $\Lambda_{min}(.), <.,.>$ respectively denote gradient, transpose, expectation, maximum eigen value, minimum eigen value, and inner product operators. The symbols $\mathbf{1}_N$ and $\boldsymbol{I}_N$ represent all-one vector of dimension $N$ and identity matrix of dimension $N \times N$, respectively.

## C.2 Nonlinear topology identification

### C.2.1 System Model

Consider a collection of $N$ sensors (nodes) generating a multi-variate time series denoted by $\mathbf{y}[t] \in \mathbb{R}^N$, where $t = 0, 1, \ldots, T-1$ denotes the time index. We assume that the dynamics of the sensor network can be captured by a $P$-th order VAR model with additive nonlinear functional dependencies:

$$y_n[t] = \sum_{n'=1}^{N} \sum_{p=1}^{P} f_{n,n'}^{(p)}(y_{n'}[t-p]) + u_n[t], \tag{C.1}$$

where $y_n[t]$ is the value of time series at time $t$ observed at node $1 \le n \le N$, $f_{n,n'}^{(p)}$ is a nonlinear function that captures the causal influence of the $p$-lagged data point of node $n'$ on node $n$, and $u_n[t]$ is the process noise, which is assumed to be zero mean i.i.d. random process. With respect to model (C.1), we define topology identification as the estimation of the functional dependencies $\left\{f_{n,n'}^{(p)}(.)\right\}_{p=1}^{P}$, $\forall n, n'$, from the observed time series $\{y_{n'}[t]\}_{n'=1}^{N}$.

### C.2.2 Kernel representation

Assume that the functions $f_{n,n'}^{(p)}$ in (C.1) belong to a reproducing kernel Hilbert space (RKHS):

$$\mathcal{H}_{n'}^{(p)} := \left\{ f_{n,n'}^{(p)} \mid f_{n,n'}^{(p)}(y) = \sum_{t=p}^{\infty} \beta_{n,n',(t-p)}^{(p)} \ \kappa_{n'}^{(p)}(y, y_{n'}[t-p]) \right\}, \tag{C.2}$$

where $\kappa_{n'}^{(p)} : \mathbb{R} \times \mathbb{R} \rightarrow \mathbb{R}$ is a positive definite kernel, which characterizes the RKHS. The kernel is a function measuring the similarity between the data points $y$ and $y_{n'}[t-p]$. The expression (F.2) follows from the fact that any function in RKHS can be expressed as an infinite combination of kernel evaluations [15], i.e., the function $f_{n,n'}^{(p)}(y)$ can be expressed as the linear combination of the similarities between $y$ and the data points $\{y_{n'}[t-p]\}_{t=p}^{t=\infty}$, with weights $\beta_{n,n',(t-p)}^{(p)}$. Here, we consider a Hilbert space with the inner product $\langle \kappa_{n'}^{(p)}(y, x_1), \kappa_{n'}^{(p)}(y, x_2) \rangle := \sum_{t=0}^{\infty} \kappa_{n'}^{(p)}(y[t], x_1) \kappa_{n'}^{(p)}(y[t], x_2)$ using kernels with reproducible property $\langle \kappa_{n'}^{(p)}(y, x_1), \kappa_{n'}^{(p)}(y, x_2) \rangle = \kappa_{n'}^{(p)}(x_1, x_2)$. Such a Hilbert space with the reproducing kernels is termed as RKHS, and the inner product induces the RKHS norm, $\|f_{n,n'}^{(p)}\|_{\mathcal{H}_{n'}^{(p)}}^2 = \sum_{t=0}^{\infty} \sum_{t'=0}^{\infty} \beta_{n,n',t}^{(p)} \ \beta_{n,n',t'}^{(p)} \ \kappa_{n'}^{(p)}(y_n[t], y_n[t'])$. We refer to [37] for further reading on RKHS.

The required causal dependencies $\left\{f_{n,n'}^{(p)} \in \mathcal{H}_{n'}^{(p)}\right\}_{n',p}$ at a particular node $n$ can be obtained by solving the following non-parametric optimization problem in batch form, considering all the samples at once:

$$\begin{aligned} \left\{\hat{f}_{n,n'}^{(p)}\right\}_{n',p} &= \arg \min_{\left\{f_{n,n'}^{(p)} \in \mathcal{H}_{n'}^{(p)}\right\}} \frac{1}{2} \sum_{\tau=P}^{T-1} \Bigg[ y_n[\tau] \\ &- \sum_{n'=1}^{N} \sum_{p=1}^{P} f_{n,n'}^{(p)}(y_{n'}[\tau-p]) \Bigg]^2 + \lambda \sum_{n'=1}^{N} \sum_{p=1}^{P} \Omega(||f_{n,n'}^{(p)}||_{\mathcal{H}_{n'}^{(p)}}). \end{aligned} \tag{C.3}$$

For a non-decreasing function $\Omega$, the solution of (F.3), denoted as $\left\{\hat{f}_{n,n'}^{(p)}\right\}_{n',p}$ can be obtained in terms of finite kernel evaluation by invoking the Representer Theorem [38]:

$$\hat{f}_{n,n'}^{(p)}(y_{n'}[\tau-p]) = \sum_{t=p}^{p+T-1} \hat{\beta}_{n,n',(t-p)}^{(p)} \ \kappa_{n'}^{(p)}(y_{n'}[\tau-p], y_{n'}[t-p]) \,. \tag{C.4}$$

Although the solution (F.4) entails only a finite number (equal to $T$) of kernel evaluations, its computational complexity becomes prohibitively high for a large value of $T$. This is a major drawback associated with the kernel formulations, which is commonly referred to as *the curse of dimensionality.* In alignment with [43], [72], we use RF approximation to solve the curse of dimensionality.

### C.2.3 RF approximation

From Appendix C.2.2, we remark that the RKHS is characterized by an inner product. Resorting to the theory of RF approximation, the inner product can be expressed in a random Fourier space, which facilitates the approximation of an RKHS function to a function in a fixed low dimensional space, thereby preventing the dimensionality growth. In addition to tackling the curse of dimensionality, working on a fixed low dimensional space will enable us to use the standard convex optimization tools to solve the topology identification.

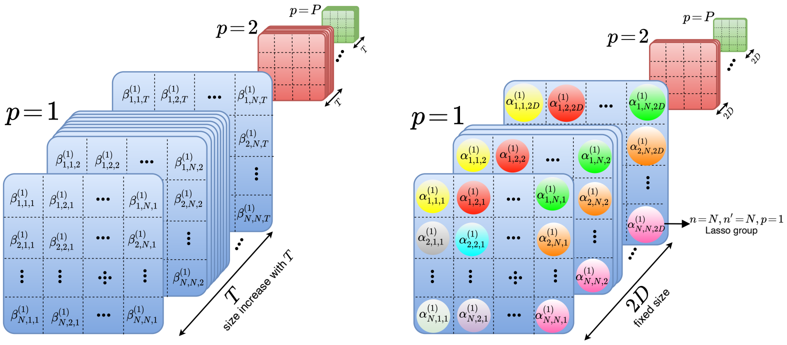


Figure C.2: RKHS parameters (left) and fixed-size RF parameters (right). The Lasso groups of RF parameters are indicated in different colours.

The RF approximation requires that the kernel defining the RKHS should be shift invariant, i.e., $\kappa^{(p)}_{n'}\left(y_{n'}[\tau-p], y_{n'}[t-p]\right) = \kappa^{(p)}_{n'}\left(y_{n'}[\tau-p] - y_{n'}[t-p]\right)$. There are many popular kernels that are shift invariant, such as the Laplacian, the Cauchy, and the Gaussian kernels. By the Bochner's Theorem [30], every shift-invariant kernel can be expressed as an inverse Fourier transform of a probability density function. Following this theorem, the kernel evaluation can be expressed as

$$\begin{aligned}
&\kappa^{(p)}_{n'}\left(y_{n'}[\tau-p], y_{n'}[t-p]\right) \\
&\quad = \int_{\mathbb{R}} \pi_{\kappa^{(p)}_{n'}}(v)\; e^{jv(y_{n'}[\tau-p]-y_{n'}[t-p])} dv \\
&\quad = \mathbb{E}_v\big[e^{jv(y_{n'}[\tau-p]-y_{n'}[t-p])}\big],
\end{aligned} \tag{C.5}$$

where $\mathbb{E}$ is the expectation operation, $\pi_{\kappa^{(p)}_{n'}}(v)$ is the probability density function corresponding to the kernel under consideration, and $v$ is the random variable associated with the probability density function. Using a sufficient amount of i.i.d. samples $\{v_i\}_{i=1}^{D}$ from the distribution $\pi_{\kappa^{(p)}_{n'}}(v)$, we can approximate the expectation in (F.5) as a sample mean (weak law of large numbers):

$$\begin{aligned}
&\hat{\kappa}^{(p)}_{n'}\left(y_{n'}[\tau-p], y_{n'}[t-p]\right) = \\
&\frac{1}{D}\sum_{i=1}^{D} e^{jv_i(y_{n'}[\tau-p]-y_{n'}[t-p])},
\end{aligned} \tag{C.6}$$

irrespective of the distribution $\pi_{\kappa^{(p)}_{n'}}(v)$. Notice that (F.6) is an unbiased estimator of the kernel evaluation in (F.5) [54]. Finding the probability distribution which is the inverse Fourier transform of a kernel is a difficult task in general. However, for a Gaussian kernel with variance $\sigma^2$, the Fourier transform is also a Gaussian with variance $\sigma^{-2}$. Hence, in this work, we restrict our choice of kernel to Gaussian kernels. Further, the real part of (F.6) is also an unbiased estimator of the kernel evaluation [42], and (F.5) can be expressed in vector form using only the real components as

$$\hat{\kappa}_{n'}^{(p)}\left(y_{n'}[\tau-p], y_{n'}[t-p]\right) = \boldsymbol{z}_{\boldsymbol{v},n'}^{(p)}(\tau)^\top \boldsymbol{z}_{\boldsymbol{v},n'}^{(p)}(t), \tag{C.7}$$

$$\begin{aligned} \textit{where } \boldsymbol{z}_{\boldsymbol{v},n'}^{(p)}(\tau) = \frac{1}{\sqrt{D}}\Big[&\sin\left(v_1 y_{n'}[\tau-p]\right), \ldots, \sin\left(v_D y_{n'}[\tau-p]\right), \\ &\cos\left(v_1 y_{n'}[\tau-p]\right), \ldots, \cos\left(v_D y_{n'}[\tau-p]\right)\Big]^\top. \end{aligned} \tag{C.8}$$

Substitute (C.7) in (F.4) to obtain an approximation of the function $\hat{f}_{n,n'}^{(p)}$ in a fixed dimension (2D):

$$\begin{aligned} \hat{\hat{f}}_{n,n'}^{(p)}\left(y_{n'}[\tau-p]\right)) &= \sum_{t=p}^{p+T-1} \hat{\beta}_{n,n',(t-p)}^{(p)} \boldsymbol{z}_{\boldsymbol{v},n'}^{(p)}(\tau)^\top \boldsymbol{z}_{\boldsymbol{v},n'}^{(p)}(t) \\ &= \boldsymbol{\alpha}_{n,n'}^{(p)\top} \boldsymbol{z}_{\boldsymbol{v},n'}^{(p)}(\tau), \end{aligned} \tag{C.9}$$

where $\boldsymbol{\alpha}_{n,n'}^{(p)} = \sum_{t=p}^{p+T-1} \hat{\beta}_{n,n',(t-p)}^{(p)} \boldsymbol{z}_{\boldsymbol{v},n'}^{(p)}(t)$. For the sake of simplicity, we define the following notations:

$$\boldsymbol{\alpha}_{n,n'}^{(p)} = [\alpha_{n,n',1}^{(p)}, \ldots, \alpha_{n,n',2D}^{(p)}]^\top \in \mathbb{R}^{2D}, \tag{C.10}$$

$$\boldsymbol{z}_{\boldsymbol{v},n'}^{(p)}(\tau) = [z_{\boldsymbol{v},n',1}^{(p)}(\tau), \ldots z_{\boldsymbol{v},n',2D}^{(p)}(\tau)]^\top \in \mathbb{R}^{2D}, \tag{C.11}$$

$$z_{\boldsymbol{v},n',k}^{(p)}(\tau) = \begin{cases} \sin(v_k y_{n'}[\tau-p]), & \text{if } k \leq D \\ \cos(v_{k-D} y_{n'}[\tau-p]), & \text{otherwise.} \end{cases}$$

The functional optimization (F.3) can be reformulated as a parametric optimization problem using (F.8). First, we define the parametric form of the loss function in (F.3):

$$\mathcal{L}^n\left(\boldsymbol{\alpha}_{n,n'}^{(p)}\right) := \sum_{\tau=P}^{T-1} \frac{1}{2}\left[y_n[\tau] - \sum_{n'=1}^{N}\sum_{p=1}^{P} \boldsymbol{\alpha}_{n,n'}^{(p)\top}\ \boldsymbol{z}_{\boldsymbol{v},n'}^{(p)}(\tau)\right]^2, \tag{C.12}$$

which can be expanded in terms of RF components as

$$\mathcal{L}^n\left(\alpha_{n,n',d}^{(p)}\right) := \sum_{\tau=P}^{T-1} \frac{1}{2}\left[y_n[\tau] - \sum_{n'=1}^{N}\sum_{p=1}^{P}\sum_{d=1}^{2D} \alpha_{n,n',d}^{(p)}\ z_{\boldsymbol{v},n',d}^{(p)}(\tau)\right]^2.$$

For convenience, the variables $\left\{\alpha_{n,n',d}^{(p)}\right\}$ and $\left\{z_{\boldsymbol{v},n',d}^{(p)}(\tau)\right\}$ are stacked in the lexicographic order of the indices $p$, $n'$, and $d$ to obtain the vectors $\boldsymbol{\alpha}_n \in \mathbb{R}^{2PND}$ and $\boldsymbol{z}_{\boldsymbol{v}}(\tau) \in \mathbb{R}^{2PND}$, respectively, and loss function can be compactly rewritten as:

$$\mathcal{L}^n(\boldsymbol{\alpha}_n) = \frac{1}{2}\sum_{\tau=P}^{T-1}\left[y_n[\tau] - \boldsymbol{\alpha}_n^\top \boldsymbol{z}_{\boldsymbol{v}}(\tau)\right]^2. \tag{C.13}$$

Following [43], the original regularization term in (F.3) can be converted to an equivalent parametric form as:

$$\begin{aligned}
&\Omega(||f^{(p)}_{n,n'}||_{\mathcal{H}^{(p)}_{n,n'}}) \\
&= \Omega\left(\sum_{\tau=p}^{p+T-1}\sum_{t=p}^{p+T-1} \hat{\beta}^{(p)}_{n,n',(\tau-p)}\, \hat{\beta}^{(p)}_{n,n',(t-p)}\, k^{(p)}_{n'}(y_n(\tau), y_n(t))\right) \\
&= \Omega\left(\sum_{\tau=p}^{p+T-1}\sum_{t=p}^{p+T-1}\hat{\beta}^{(p)}_{n,n',(\tau-p)}\hat{\beta}^{(p)}_{n,n',(t-p)}\boldsymbol{z}^{(p)}_{\boldsymbol{v},n'}(\tau)^\top \boldsymbol{z}^{(p)}_{\boldsymbol{v},n'}(t)\right). \\
&= \Omega(||\boldsymbol{\alpha}^{(p)}_{n,n'}||_2). \qquad (C.14)
\end{aligned}$$

The function $\Omega$ in (C.14) is chosen to be $\Omega(.) = |.|$, where $|.|$ represents the absolute value function, in order to promote the group sparsity of $\boldsymbol{\alpha}^{(p)}_{n,n'}$ [11]. Such regularizers are typically known as *group-Lasso regularizers* (see, Fig. C.2 for a visual representation of the Lasso groups). Note that the function $|.|$ is non-decreasing, thereby satisying the regularization criteria to apply the Representer Theorem. Using (F.11) and (C.14), a parametric form of (F.3) can be constructed as follows:

$$\{\widehat{\boldsymbol{\alpha}}_n\}_{n'} = \arg\min_{\{\boldsymbol{\alpha}_n\}} \mathcal{L}^n(\boldsymbol{\alpha}_n) + \lambda\sum_{n'=1}^{N}\sum_{p=1}^{P}||\boldsymbol{\alpha}^{(p)}_{n,n'}||_2. \qquad (C.15)$$

Although the topology can be estimated by solving (F.10), this approach has several drawbacks since it is a batch formulation, meaning that (F.10) requires the entire batch of the time series samples $y_n[t],\ t = 0, 1, \ldots, T-1$ from all the nodes. In addition, the batch formulation is not useful when the data is available in a streaming manner and cannot be used to track the instantaneous time-varying topologies of non-stationary systems. Moreover, since the batch optimization computes the solutions using an entire batch of data, the computational complexity can often become prohibitively high, especially when batch size is huge. Hence, motivated by the above factors, we propose an online topology estimation strategy with a lower computational complexity in the following section.

## C.3 Online learning

To formulate an an online optimization framework, we replace the batch loss function $\mathcal{L}^n(\boldsymbol{\alpha}_n)$ in (F.10) with a stochastic (instantaneous) loss function $\ell^n_t(\boldsymbol{\alpha}_n) = \frac{1}{2}[y_n[t] - \boldsymbol{\alpha}_n^\top \boldsymbol{z}_{\boldsymbol{v}}(t)]^2$:

$$\widehat{\boldsymbol{\alpha}}_n = \arg\ \min_{\boldsymbol{\alpha}_n} \ell^n_t(\boldsymbol{\alpha}_n) + \lambda\sum_{n'=1}^{N}\sum_{p=1}^{P}||\boldsymbol{\alpha}^{(p)}_{n,n'}||_2. \qquad (C.16)$$

The loss function $l^n_t(\boldsymbol{\alpha}_n)$ in (C.16) is analogous to a Least Mean Square (LMS) formulation. However, notice that the estimates of LMS are prone to observation noise and can be unstable in practice. To avoid this problem, we formulate (C.16) in a recursive least square (RLS) sense, which further provides necessary stability in addition to faster convergence:

$$\tilde{\ell}_t^n(\boldsymbol{\alpha}_n) = \mu \sum_{\tau=P}^{t} \gamma^{t-\tau} \ell_\tau^n(\boldsymbol{\alpha}_n). \tag{C.17}$$

In (F.12), we replace the instantaneous loss with a running average loss using an exponential window. The parameter $\gamma \in (0,1)$ is the forgetting factor of the window, and $\mu = 1-\gamma$ is set to normalize the exponential weighting window. We expand the RLS loss function as follows:

$$\tilde{\ell}_t^n(\boldsymbol{\alpha}_n) = \frac{1}{2}\mu \sum_{\tau=P}^{t-1} \gamma^{t-\tau} \Big( y_n^2[\tau] + \boldsymbol{\alpha}_n^\top \boldsymbol{z}_{\boldsymbol{v}}(\tau)\boldsymbol{z}_{\boldsymbol{v}}(\tau)^\top \boldsymbol{\alpha}_n - 2y_n[\tau]\boldsymbol{z}_{\boldsymbol{v}}(\tau)^\top \boldsymbol{\alpha}_n \Big) \tag{C.18}$$

$$= \frac{1}{2}\mu \sum_{\tau=P}^{t-1} \gamma^{t-\tau} y_n^2[\tau] + \frac{1}{2}\boldsymbol{\alpha}_n^\top \boldsymbol{\Phi}[t]\boldsymbol{\alpha}_n - \boldsymbol{r}_n[t]^\top \boldsymbol{\alpha}_n, \tag{C.19}$$

where

$$\boldsymbol{\Phi}[t] = \mu \sum_{\tau=P}^{t} \gamma^{t-\tau} \boldsymbol{z}_{\boldsymbol{v}}(\tau)\boldsymbol{z}_{\boldsymbol{v}}(\tau)^\top, \tag{C.20}$$

$$\boldsymbol{r}_n[t] = \mu \sum_{\tau=P}^{t} \gamma^{t-\tau} y_n[\tau]\boldsymbol{z}_{\boldsymbol{v}}(\tau). \tag{C.21}$$

As in a typical RLS formulation, these quantities can be updated recursively as $\boldsymbol{\Phi}[t] = \gamma\boldsymbol{\Phi}[t-1] + \mu\boldsymbol{z}_{\boldsymbol{v}}(t)\boldsymbol{z}_{\boldsymbol{v}}(t)^\top$ and $\boldsymbol{r}_n[t] = \gamma\boldsymbol{r}_n[t-1] + \mu y_n[t]\boldsymbol{z}_{\boldsymbol{v}}(t)$. The gradient of the loss function can be obtained as

$$\nabla\tilde{\ell}_t^n(\boldsymbol{\alpha}_n) = \boldsymbol{\Phi}[t]\boldsymbol{\alpha}_n - \boldsymbol{r}_n[t]. \tag{C.22}$$

Finally, using the RLS loss function, the topology can be estimated by solving

$$\arg\min_{\boldsymbol{\alpha}_n} \tilde{\ell}_t^n(\boldsymbol{\alpha}_n) + \lambda \sum_{n'=1}^{N} \sum_{p=1}^{P} \|\boldsymbol{\alpha}_{n,n'}^{(p)}\|_2. \tag{C.23}$$

The cost function in (F.16) consists of a differentiable loss function and a non-differentiable group-Lasso regularizer.The online subgradient descent (OSGD) or the mirror descent (MD) method can be used to solve (F.16) online. However, these methods work by linearizing the entire objective function in (F.16) using a subgradient of it. If the group-Lasso regularizer is linearized, its ability to induce sparsity is compromised, resulting in non-sparse estimates. Hence, we choose an alternate optimization technique known as composite objective mirror descent (COMID) [39], a modified version of the MD algorithm, in which the differentiable part of the objective function is linearized, whereas the regularizer is kept intact. The online

COMID updates can be written as

$$\boldsymbol{\alpha}_n[t+1] = \arg\min_{\boldsymbol{\alpha}_n} J_t^{(n)}(\boldsymbol{\alpha}_n), \tag{C.24}$$

$$\text{where } J_t^{(n)}(\boldsymbol{\alpha}_n) \triangleq \nabla \tilde{\ell}_t^n(\boldsymbol{\alpha}_n[t])^\top (\boldsymbol{\alpha}_n - \boldsymbol{\alpha}_n[t]) + \frac{1}{2a_t}\|\boldsymbol{\alpha}_n - \boldsymbol{\alpha}_n[t]\|_2^2 + \lambda \sum_{n'=1}^{N}\sum_{p=1}^{P} \|\boldsymbol{\alpha}_{n,n'}^{(p)}\|_2, \tag{C.25}$$

where $\boldsymbol{\alpha}_n[t] \in \mathbb{R}^{2PND}$ is the estimate of $\boldsymbol{\alpha}_n$ at time $t$. The objective function $J_t^{(n)}$ in (C.25) consists of 3 parts: (i) gradient of loss function given by (F.18), (ii) a Bregman divergence term with $a_t$ as the step size, and (iii) a sparsity enforcing group-Lasso regularizer. The Bregman divergence [40] improves the stability of the online algorithms by constraining the value of the new estimate $\boldsymbol{\alpha}_n[t+1]$ within the proximity of the previous estimate $\boldsymbol{\alpha}_n[t]$. The Bregman divergence $B(\boldsymbol{\alpha}_n, \boldsymbol{\alpha}_n[t]) = \frac{1}{2}\|\boldsymbol{\alpha}_n - \boldsymbol{\alpha}_n[t]\|_2^2$ is selected in such a way that the optimization problem (E.17) has a closed form solution [40]. For notational convenience, we denote the gradient in (C.25) as

$$\mathbf{v}_n[t] := \nabla \tilde{\ell}_t^n(\boldsymbol{\alpha}_n[t]). \tag{C.26}$$

The objective function in (C.25) is expanded by omitting the constants leading to the following formulation:

$$J_t^{(n)}(\boldsymbol{\alpha}_n) \propto \frac{\boldsymbol{\alpha}_n^\top \boldsymbol{\alpha}_n}{2a_t} + \boldsymbol{\alpha}_n^\top \left(\mathbf{v}_n[t] - \frac{1}{a_t}\boldsymbol{\alpha}_n[t]\right) + \lambda \sum_{n'=1}^{N}\sum_{p=1}^{P} \|\boldsymbol{\alpha}_{n,n'}^{(p)}\|_2$$

$$= \sum_{n'=1}^{N}\sum_{p=1}^{P}\Bigg[\frac{\boldsymbol{\alpha}_{n,n'}^{(p)\top}\boldsymbol{\alpha}_{n,n'}^{(p)}}{2a_t} + \boldsymbol{\alpha}_{n,n'}^{(p)\top}\left(\mathbf{v}_{n,n'}^{(p)}[t] - \frac{1}{a_t}\boldsymbol{\alpha}_{n,n'}^{(p)}[t]\right) + \lambda\|\boldsymbol{\alpha}_{n,n'}^{(p)}\|_2\Bigg]. \tag{C.27}$$

A closed form solution for (E.17) using (C.27) can be obtained via the multidimensional shrinkage-thresholding operator [41]:

$$\boldsymbol{\alpha}_{n,n'}^{(p)}[t+1] = \left(\boldsymbol{\alpha}_{n,n'}^{(p)}[t] - a_t\mathbf{v}_{n,n'}^{(p)}[t]\right) \times \left[1 - \frac{a_t\lambda}{\|\boldsymbol{\alpha}_{n,n'}^{(p)}[t] - a_t\mathbf{v}_{n,n'}^{(p)}[t]\|_2}\right]_+, \tag{C.28}$$

where $[x]_+ = \max\{0, x\}$. The first part $\boldsymbol{\alpha}_{n,n'}^{(p)}[t] - \gamma_t\mathbf{v}_{n,n'}^{(p)}[t]$ in (C.28) forces the stochastic gradient update of $\boldsymbol{\alpha}_{n,n'}^{(p)}$ in a way to descend the recursive loss function $\tilde{\ell}_t^n(\boldsymbol{\alpha}_n)$, and the second part in (C.28) enforces group sparsity of $\boldsymbol{\alpha}_{n,n'}^{(p)}$. This closed-form expression estimates the required dependency between the time series $y_n$ and the $p$-th time lagged value of time series $y_{n'}$ at time instant $t+1$, in terms of the parameter vector $\boldsymbol{\alpha}_{n,n'}^{(p)}[t+1]$. We name the proposed algorithm as *Random feature based nonlinear topology identification via recursive sparse online learning* (RFNL-TIRSO), which is shown in **Algorithm 8**.

**Algorithm 7:** RFNL-TIRSO Algorithm

```
Result: $\left\{\boldsymbol{\alpha}_{n,n'}^{(p)}\right\}_{n,n',p}$
Store $\{\boldsymbol{y}_n[t]\}_{t=1}^{P}$,
Initialize $\lambda > 0$, $a_t > 0$, $\theta > 0$, $D$, $\sigma_n$ and $\boldsymbol{\Phi}(P-1) = \theta \boldsymbol{I}_{2PND}$
for $t = P, P+1, \ldots$ do
    Get data samples $y_n[t]$, $\forall n$ and compute $\boldsymbol{z}_{\boldsymbol{v}}(t)$
    $\boldsymbol{\Phi}[t] = \gamma \boldsymbol{\Phi}[t-1] + \mu \boldsymbol{z}_{\boldsymbol{v}}(t) \boldsymbol{z}_{\boldsymbol{v}}(t)^\top$
    for $n = 1, \ldots, N$ do
        $\boldsymbol{r}_n[t] = \gamma \boldsymbol{r}_n[t-1] + \mu y_n[t] \boldsymbol{z}_{\boldsymbol{v}}(t)$
        compute $\mathbf{v}_n[t]$ using (F.18), (D.29)
        for $n' = 1, \ldots, N$ do
            compute $\boldsymbol{\alpha}_{n,n'}^{(p)}[t+1]$ using (C.28)
        end
    end
end
```

## C.4 Theoretical results

The performance analysis and convergence guarantee of RFNL-TIRSO are presented in this section using dynamic regret. Regret is a popular metric to measure the performance of an online algorithm [90]. Despite being originally developed for static learning problems, numerous online algorithms involving dynamic regret analysis have been developed [32,84–86] to solve problems in a dynamic environment; however all of them belong to the class of linear algorithms. Moreover, [84–86] assume differentiable objective functions, and hence they cannot be leveraged in RFNL-TIRSO. Dynamic regret bounds for nonlinear algorithms are proposed in [87–89]. In [87], the problem under consideration is limited to positive functions, whereas our problem formulation does not have such a limitation. The regret analysis presented in [88] differs significantly from the proposed method for several reasons. First, the objective function used in [88] must be differentiable, while in our proposed method, the regularizer is non-differentiable. Second, in contrast to [88], the regret analysis in the proposed method involves multiple decoupled functions representing interpretable topological connections. Although [89] provides a logarithmic regret bound using second-order information, the objective function under consideration is differentiable. Our theoretical analysis is based on the following assumptions:

- ***A*1** : Bounded samples: For all the time series samples, there exists $B_y > 0$ such that $\{|y_n[t]|^2\}_{n,t} \leq B_y \leq \infty$.
- ***A*2** : Shift-invariant kernels: kernels used are shift-invariant, i.e., $k(x_i, x_j) = k(x_i - x_j)$.
- ***A*3** : Bounded minimum eigenvalue of $\boldsymbol{\Phi}[t]$: There exists $\rho_l > 0$ such that $\Lambda_{min}(\boldsymbol{\Phi}[t]) > \rho_l$, where $\Lambda_{min}(.)$ denotes the minimum eigenvalue.

- ***A*4** : Bounded maximum eigenvalue of $\mathbf{\Phi}[t]$: There exists $L > 0$ such that $\Lambda_{max}(\mathbf{\Phi}[t]) < L < \infty$, where $\Lambda_{max}(.)$ denotes the maximum eigenvalue.

***A*1** is reasonable in practice as the signals from real-world applications are bounded. ***A*2** is always true for typical kernels like Gaussian, Laplacian, etc. Since $\mathbf{\Phi}(t)$ is a sum of rank one matrices formed using feature vectors, ***A*3** will hold as long as the feature vectors are linearly independent. This is quite a reasonable assumption in practice when a sufficient amount of data is available. Note that ***A*3** is important for the strong convexity assumption of the loss function, which is used in the following sections. ***A*4** can be obtained by combining ***A*1** and the fact that the sum of eigenvalues of $\mathbf{\Phi}[t]$ is equal to its trace.

### C.4.1 Dynamic Regret Analysis

Dynamic regret is a popular metric to quantify the performance of online algorithms in a dynamic environment [83]. As a preliminary step to the regret analysis, we define the optimum RKHS and RF coefficients.

***Optimum RKHS coefficients***: Using the batch form solution (F.4), which exploits the Representer Theorem, a parametric autoregressive representation at time $t$ can be obtained as

$$\widehat{y}_n[t] = \hat{\boldsymbol{\beta}}_n^\top \boldsymbol{\kappa}_t, \tag{C.29}$$

where $\hat{\boldsymbol{\beta}}_n \in \mathbb{R}^{NPt}$ and $\boldsymbol{\kappa}_t \in \mathbb{R}^{NPt}$ are respectively obtained by stacking the variables $\hat{\beta}^{(p)}_{n,n',(\tau-p)}$ and the kernel evaluations in (F.4) along the lexicographic order of the indices $n'$,$p$, and the time index up to $t$. The optimum RKHS coefficients $\boldsymbol{\beta}_n^*[t]$ for each node $n$ at time $t$ can be obtained by solving

$$\boldsymbol{\beta}_n^*[t] = \arg\min_{\hat{\boldsymbol{\beta}}_n} h_t^n(\hat{\boldsymbol{\beta}}_n), \tag{C.30}$$

where the cost function $h_t^n(\hat{\boldsymbol{\beta}}_n)$ in (C.30) is composed of two terms: $h_t^n(\hat{\boldsymbol{\beta}}_n) = \tilde{\ell}_t^n(\hat{\boldsymbol{\beta}}_n) + \omega^n(\hat{\boldsymbol{\beta}}_n)$, where $\tilde{\ell}_t^n(.)$ is the RLS loss function defined in (F.12) with instantaneous losses computed as $\ell_t^n(\hat{\boldsymbol{\beta}}_n) = \frac{1}{2}[y_n[t] - \hat{\boldsymbol{\beta}}_n^\top \boldsymbol{\kappa}_t]^2$, and $\omega^n(.)$ is the group-Lasso regularizer defined as $\omega^n(\hat{\boldsymbol{\beta}}_n) = \lambda \sum_{n'=1}^N \sum_{p=1}^P \|\hat{\boldsymbol{\beta}}^{(p)}_{n,n'}\|_2$.

***Optimum RF coefficients***: Following the same procedure, we define the optimum RF coefficients $\boldsymbol{\alpha}_n^*[t]$ at time $t > P$ as

$$\boldsymbol{\alpha}_n^*[t] = \arg\min_{\boldsymbol{\alpha}_n} h_t^n(\boldsymbol{\alpha}_n), \tag{C.31}$$

where $h_t^n(\boldsymbol{\alpha}_n) = \tilde{\ell}_t^n(\boldsymbol{\alpha}_n) + \omega^n(\boldsymbol{\alpha}_n)$, and $\tilde{\ell}_t^n(.)$ is the RLS loss defined in (F.12) and $\omega^n(\boldsymbol{\alpha}_n) = \lambda \sum_{n'=1}^N \sum_{p=1}^P \|\boldsymbol{\alpha}^{(p)}_{n,n'}\|_2$. It should be noticed that the optimum RF coefficients $\boldsymbol{\alpha}_n^*[t]$ is different from the RFNL-TIRSO estimate $\boldsymbol{\alpha}_n[t]$ obtained by the computationally light COMID algorithm, as RFNL-TIRSO only makes one COMID update per time instant.

***Dynamic Regret***: Dynamic Regret is defined as the cumulative sum of the difference between the estimated cost function and the optimal cost function over all time instants. In our framework, it can be expressed as

$$\boldsymbol{R}_n[T] = \sum_{t=P}^{T-1} \left[ h_t^n(\boldsymbol{\alpha}_n[t]) - h_t^n(\boldsymbol{\beta}_n^*[t]) \right]. \tag{C.32}$$

Our aim is to find a theoretical bound for $\boldsymbol{R}_n[T]$. Since our online algorithm works in the RF space, we perform the regret analysis with reference to the optimal cost function in the RF space, i.e., $h_t^n(\boldsymbol{\alpha}_n^*[t])$. Notice that this is without loss of generality because there is a one-to-one mapping. Adding and subtracting $h_t^n(\boldsymbol{\alpha}_n^*[t])$ in (F.27) yields

$$\boldsymbol{R}_n[T] = \boldsymbol{R}_n^{\mathrm{rf}}[T] + \boldsymbol{\xi}_n[T], \tag{C.33}$$

where $\boldsymbol{R}_n^{\mathrm{rf}}[T] = \sum_{t=P}^{T-1} \left( h_t^n(\boldsymbol{\alpha}_n[t]) - h_t^n(\boldsymbol{\alpha}_n^*[t]) \right)$ is the regret with respect to optimal cost in RF space and $\boldsymbol{\xi}_n[T] = \sum_{t=P}^{T-1} \left( h_t^n(\boldsymbol{\alpha}_n^*[t]) - h_t^n(\boldsymbol{\beta}_n^*[t]) \right)$ is the cumulative RF approximation error caused by the dimensionality reduction.

#### C.4.1.1 Bounding the regret w.r.t. optimal cost function in RF space

**Lemma 2** bounds $\boldsymbol{R}_n^{\mathrm{rf}}(T)$.

**Theorem 1.** *Under the assumptions of A1, A3, A4, and letting $a_t = \frac{1}{L}$, the dynamic regret of RFNL-TIRSO (**Algorithm 8**) w.r.t. the optimal cost function in the RF space satisfies*

$$\begin{aligned} \boldsymbol{R}_n^{rf}(T) \leq & \left( \left(1 + \frac{L}{\rho_l}\right) \sqrt{2PNDB_y} + \lambda\sqrt{PN} \right) \times \\ & \left( \|\boldsymbol{\alpha}_n^*[P]\|_2 + \boldsymbol{W}_n(T) \right), \end{aligned}$$

where $\boldsymbol{W}_n(T) = \sum_{t=P}^{T-1} \|\boldsymbol{\alpha}_n^*[t] - \boldsymbol{\alpha}_n^*[t-1]\|_2$.

*Proof:* See Appendix C.7.
From **Lemma 2**, it can be readily seen that if $\boldsymbol{W}_n(T)$ is sublinear, then the regret will also be sublinear.

#### C.4.1.2 Bounding the cumulative RF approximation error

**Lemma 3** provides a bound for $\boldsymbol{\xi}_n(T)$.

**Theorem 2.** *Under assumptions A1 and A2, there exists $\epsilon \geq 0$ such that the cumulative approximation error $\boldsymbol{\xi}_n[T]$ of RFNL-TIRSO (**Algorithm** 8) satisfies*

$$\boldsymbol{\xi}_n(T) \leq \epsilon L_h T C.$$

*Proof:* See Appendix C.8.
Finally, we bound the dynamic regret $\boldsymbol{R}_n(T)$ using **Lemma 2** and **Lemma 3**.

**Theorem 3.** *Under the assumptions of A1, A2, A3, and A4, the dynamic regret $\boldsymbol{R}_n(T)$ of RF-NLTIRSO (**Algorithm** 8) satisfies*

$$\boldsymbol{R}_n(T) \leq \Big(\Big(1+\frac{L}{\rho_l}\Big)\sqrt{2PNDB_y}+\lambda\sqrt{PN}\Big)\times \\ \Big(\|\boldsymbol{\alpha}_n^*[P]\|_2+\boldsymbol{W}_n(T)\Big)+\epsilon L_h TC.$$

*Proof:* **Theorem 4** can be directly and readily proved by substituting **Lemma 2** and **Lemma 3** in (F.30). Notice that if we have setting $\epsilon = \mathcal{O}(\frac{1}{\sqrt{T}})$, this results in a dynamic regret of $\mathcal{O}(\boldsymbol{W}_n(T)+\sqrt{T})$.In such cases, the dynamic regret is sublinear, if $\boldsymbol{W}_n(T)$ is sublinear. Ideally, an online algorithm must obtain a sublinear dynamic regret, which implies that $\boldsymbol{R}_n(T)/T \rightarrow 0$ as $T \rightarrow \infty$, or in the worst case, a linear regret which implies $\boldsymbol{R}_n(T)/T \rightarrow$ *constant*, where *constant* is known as the steady state error. Notice that in our case, this steady state error when $\boldsymbol{W}_n(T)$ is sublinear is $\epsilon L_f C$. If $\epsilon$ is small, the resulting study state error will also be small. As shown in appendix $\boldsymbol{B}$ , we can make $\epsilon$ sufficiently small by increasing the number of random features $D$ by trading off with complexity [16].

## C.5 Experimental results

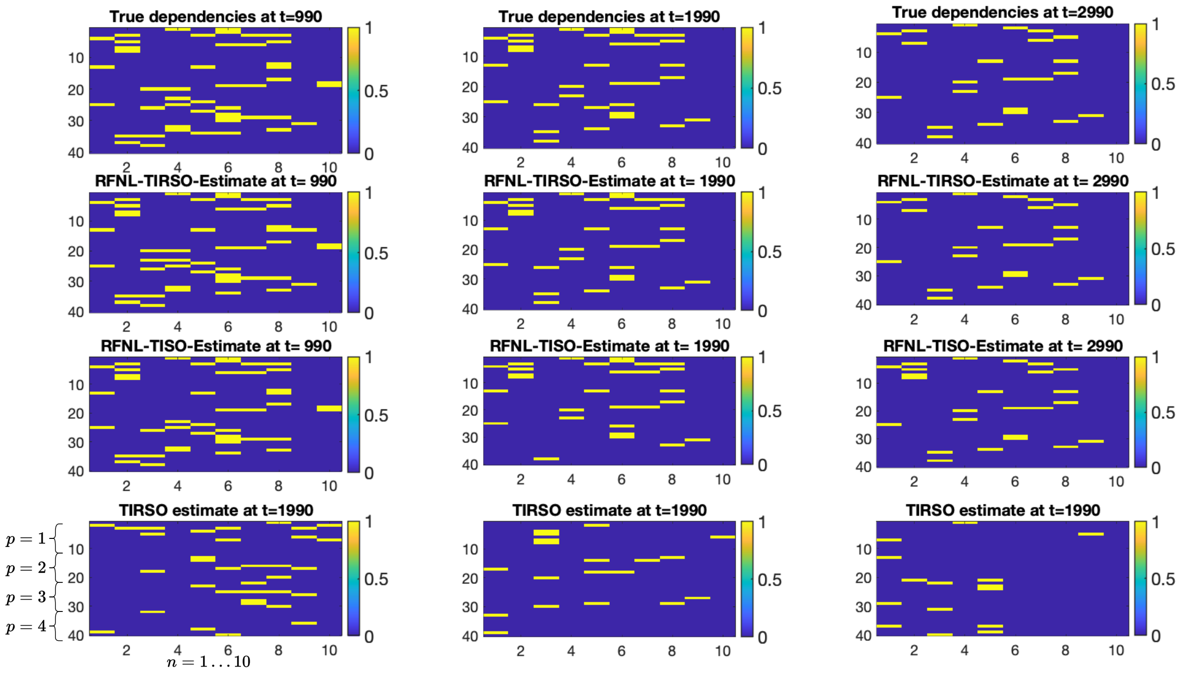


Figure C.3: True topology plotted against topology estimated using various algorithms for $g(x) = g_1(x)$. In each subfigure, the x-axis corresponds to nodes $n = 1,\dots,10$, and the y-axis corresponds to nodes $n = 1,\dots,10$ for time lags $p = 1,\dots,4$. The edge values are indicated by the colour code.

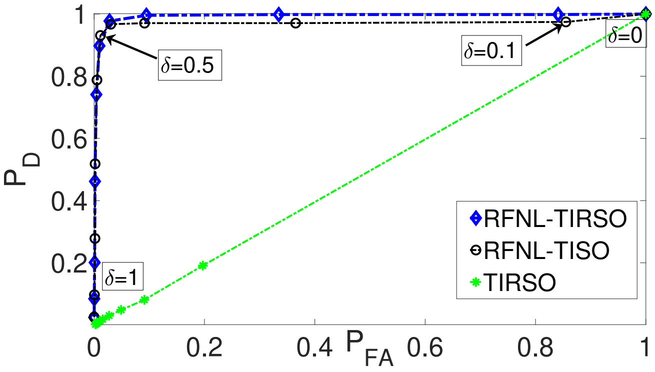


Figure C.4: Function $g_1$.

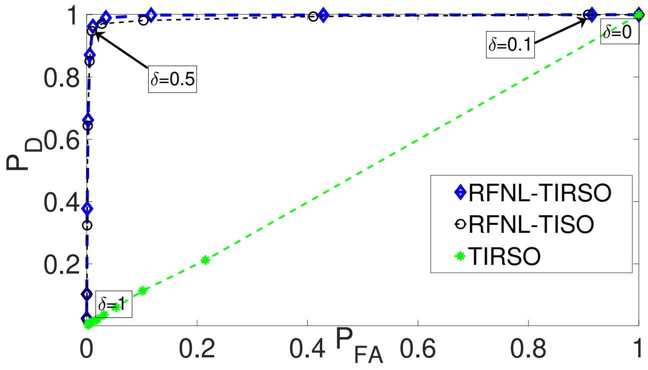


Figure C.5: Function $g_2$.

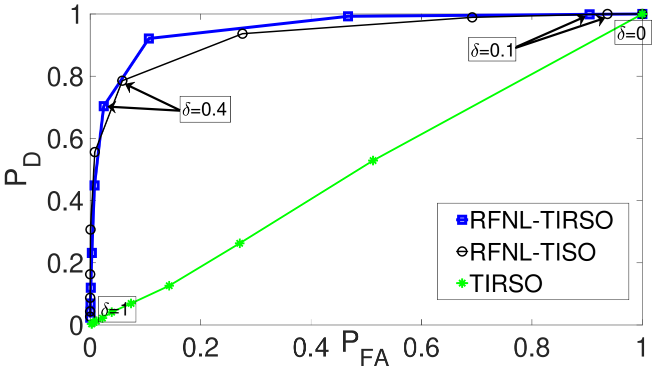


Figure C.6: Function $g_3$.

Figure C.7: Receiver-Operating Curve for each of the realization of the nonlinear function $g(x)$.

In this section, we analyze the performance of RFNL-TIRSO using extensive numerical experiments. We choose TIRSO [48], RFNL-TISO [22] and PDIS [75,91], as the state-of-the-art competitors to compare the performance of RFNL-TIRSO. It is to be remarked that TIRSO is an online topology algorithm designed by assuming linear VAR models. TIRSO is selected in order to show the advantages of the proposed nonlinear algorithm RFNL-TIRSO, compared to its linear counterpart. The second algorithm RFNL-TISO is an online nonlinear topology estimation algorithm designed by considering an instantaneous loss function. Based on the discussions in Appendix C.3, RFNL-TIRSO is expected to show better performance compared to RFNL-TISO since it incorporates an RLS-based loss function. The third algorithm, PDIS [75, 91], is a recent online nonlinear topology identification algorithm using dictionaries of kernel functions based on partial-derivative-imposed sparsity. To the best of our knowledge, these three algorithms are the best benchmarks to compare the performance of RFNL-TIRSO, and although some other batch-based algorithms are available [11], [9], [10], they are not comparable to our algorithm, since they are offline algorithms.

The per node computational complexity of RFNL-TIRSO, RFNL-TISO, and TIRSO, are in the order of $\mathcal{O}\big(N^2P^2D^2\big)$, $\mathcal{O}\big(NPD\big)$, and $\mathcal{O}\big(N^2P^2\big)$, respectively. Although RFNL-TIRSO is computationally heavier than the competitors, it provides robustness, and theoretical performance guarantees, which is not the case for the competing algorithms and which we demonstrate through several numerical experiments in this section.

Experiments shown in this section are conducted using both synthetic and real data sets. The synthetic dataset includes graph-connected time series data generated by assuming different topology transition patterns to highlight the ability of the online algorithms to track non-stationary topologies. The real data sets include (i) time seires data collected from the Lundin's offshore oil and Gas platform[1] and (ii) Epileptic seizure data [92].

## C.5.1 Experiments using Synthetic Data Sets

### C.5.1.1 Piecewise stationary topology

We generate a multivariate time series using a nonlinear VAR model (C.1) with $N = 10, P = 4$. The nonlinear function in (C.1) is taken as $f^{(p)}_{n,n'}(x) = a^{(p)}_{n,n'}(x)g(x)$, where $g(x)$ is a nonlinear function and $a^{(p)}_{n,n'}(x) \in \{0, 1\}$. The experiments are conducted with three different realizations of $g(x)$: $g_1(x) = 0.25\sin(x^2) + 0.25\sin(2x) + 0.5\sin(x)$, $g_2(x) = 0.25\cos(x^2) + 0.25\cos(2x) + 0.5\cos(x)$, and with a Gaussian kernel, i.e., $g_3(x) = (1/\sqrt{2\pi})\exp(-x^2/2)$. We refer to $a^{(p)}_{n,n'}$ as an *edge*, and $a^{(p)}_{n,n'}(.) = 0/1$ means that the $p$-th time-lagged dependency between $n$ and $n'$ is disabled/enabled. A graph-connected time series is generated by restricting the number of active edges to be 30% of the total available edges. Further, we introduce abrupt changes in the topology after every 1000 time step by randomly cutting off 30% of the available

[1] https://www.lundin-energy.com/

active edges. Notice that the initial $P$ data samples are generated randomly, and the rest of the data are generated using model (C.1). The hyperparameters of all the algorithms used in the experiments are tuned heuristically to get the maximum area under the receiver operating curve, which is explained below. The hyperparameter settings for RFNL-TIRSO are $(\sigma_n, \lambda, a_t) = (2.5, 0.01, 0.1/\Lambda_{max}(\boldsymbol{\phi}[t]))$, for $g_1$ and $g_2$, and $(1, 0.01, 0.1/\Lambda_{max}(\boldsymbol{\phi}[t]))$ for $g_3$. The top row of Fig. C.3 contains the true edges $\left\{a^{(p)}_{n,n'}\right\}$ at different time steps, which are arranged in matrices of size $N \times N$, for $p = 1, 2, \ldots, P$, and stacked vertically, resulting in matrices of size $NP \times N$. The estimated dependencies $\left\{\hat{a}^{(p)}_{n,n'}\right\}$ using different algorithms are shown in the bottom rows. After computing the normalized $\ell_2$ norms $b^{(p)}_{n,n'}[t] = \|\boldsymbol{\alpha}^{(p)}_{n,n'}[t]\|_2/(\max_{n'}\|\boldsymbol{\alpha}^{(p)}_{n,n'}[t]\|_2)$, the presence of an edge is detected using a threshold $\delta$ as $\hat{a}^{p}_{n,n'} = 1\{b^{(p)}_{n,n'}[t] < \delta\}$, where $1\{x\} = 1/0$, if $x$ is true/false. It is clear from Fig. C.3 that the estimates of RFNL-TISO are very close to the ground truth, and they outperform others.

A numerical comparison of the performances of the algorithms is made using the probability of false alarm ($P_{FA}$) and the probability of detection ($P_D$). The probability of false alarm ($P_{FA}$) refers to the probability that the algorithm reports the presence of a dependency in the network that is not actually present. On the other hand, the probability of detection($P_D$), refers to the probability that the algorithm detects a dependency that is truly present in the network. In our experiment, we assume there is a presence of a detected edge from the $p-th$ time-lagged value of $n'-th$ sensor to the present value of the $n-th$ sensor if the value of coefficient $b^{(p)}_{n,n'}[t]$ is greater than a threshold $\delta \in [0, 1]$, and define $P_{FA}$ and $P_D$ as

$$P_{\mathrm{D}}[t] \triangleq 1 - \frac{\sum_{n\neq n'}\sum_{p=1}^{P}\mathbb{E}\left[1\{b^{(p)}_{n,n'}[t] < \delta\}1\{a_{n,n'} = 1\}\right]}{\sum_{n\neq n'}\sum_{p=1}^{P}\mathbb{E}[1\{a_{n,n'} = 1\}]},$$
$$P_{\mathrm{FA}}[t] \triangleq \frac{\sum_{n\neq n'}\sum_{p=1}^{P}\mathbb{E}\left[1\{b^{(p)}_{n,n'}[t] > \delta\}1\{a_{n,n'} = 0\}\right]}{\sum_{n\neq n'}\sum_{p=1}^{P}\mathbb{E}\left[1\{a_{n,n'} = 0\}\right]}, \tag{C.34}$$

where $1\{x\} = 1/0$, if $x$ is true/false and $\delta$ is a threshold. From (C.34), it is clear that when $\delta = 0$, both $P_D$ and $P_{FA}$ become one. With an increase in $\delta$, both $P_D$ and $P_{FA}$ decrease, eventually reaching zero when $\delta$ equals one.

The Receiver-Operating curve (ROC) of the different algorithms at time $t = 990$ is plotted in Fig. C.7 by varying $\delta$ from 0 to 1, with $P_{FA}$ in the x-axis and $P_D$ in the y-axis. The area under the ROC curve (AUC) is computed to evaluate the performance of the algorithm. A topology identification algorithm with a high AUC value is characterized by by a high $P_D$ and low $P_{FA}$, indicating that it can accurately identify network topologies while minimizing the occurrence of false positives. From Fig. C.7, it can be observed that the area under ROC (AUC) of the RFNL-TIRSO is substantially better than TIRSO and slightly better than RFNL-TISO for all three nonlinearity functions. These observations are more evident from Table F.1, where the computed AUC values are tabulated. We further analyze the AUC of RFNL-TIRSO for different RF space dimensions, i.e., $D \in \{20, 30, 50\}$, at different time instants in Table C.2, for the nonlinear function $g(x) = g_1(x)$. As expected, the AUC increases with $D$ and the number of data samples. A similar AUC trend as in Table C.2 was obtained for the other two nonlinear functions $g_1$ and $g_2$.

Table C.1: AUC for different algorithms.

| AUC | $g_1$ | $g_2$ | $g_3$ |
|---|---|---|---|
| $RFNL-TIRSO$ | **0.9914** | **0.9949** | **0.9543** |
| $RFNL-TISO$ | 0.9741 | 0.9817 | 0.9317 |
| $TIRSO$ | 0..4967 | 0.5 | .5123 |

Table C.2: AUC curve for different values of D.

| AUC | $t = 990$ | $t = 1990$ | $t = 2990$ |
|---|---|---|---|
| $D = 20$ | 0.9500 | 0.9762 | 0.9732 |
| $D = 30$ | 0.9568 | 0.9827 | 0.9835 |
| $D = 50$ | 0.9721 | 0.9887 | **0.9901** |

### C.5.1.2 Lorenz graph

We also present experiments with synthetic data sets generated using the Lorenz graph [93]. We consider a discretized version of the Lorenz graph involving 3 time series exhibiting the following nonlinear dependencies:

$$\begin{pmatrix} y_1[t+1] \\ y_2[t+1] \\ y_3[t+1] \end{pmatrix} = 0.01\begin{pmatrix} 10(y_2[t]-y_1[t]) \\ y_1[t](28-y_3[t])-y_2[t] \\ y_1[t]y_2[t]-\frac{8}{3}y_3[t] \end{pmatrix}+\begin{pmatrix} y_1[t] \\ y_2[t] \\ y_3[t] \end{pmatrix} \tag{C.35}$$

Compared to the causality model used in C.5.1.1, the Lorenz graph model (C.35) involves only order one ($P = 1$) time lag dependencies among the nodes. Moreover, note that (C.35) involves nonadditive nonlinear interactions among the nodes, which is different from the VAR assumption in (C.1). The performance of the RFNL-TIRSO and the PDIS [91] algorithms are compared in this section, whereas TIRSO is omitted since the algorithm implementation assumes $P > 1$. To ensure a fair comparison, we follow exactly the same experiment set up as in [91], in which, the performance is measured using the *edge identification error rate* (EIER), defined as $EIER = \frac{\|\mathbf{A}-\hat{\mathbf{A}}\|_0}{N(N-1)} \times 100$, where $\mathbf{A}$ is the true dependency matrix and $\hat{\mathbf{A}}$ is the estimated dependency matrix. For RFNL-TIRSO, $\hat{\mathbf{A}}$ is computed using $b^{(1)}_{n,n'}$. The hyperparameters are tuned heuristically to obtain minimum EEIR resulting in a setting $(\sigma_n, \lambda, a_t) = (1, .3, 1/(t\Lambda_{max}(\boldsymbol{\phi}[t])))$. The estimated and true binary adjacency matrix (excluding self-dependencies) are shown in Fig. C.11, and the EIER till $t = 1750$ is plotted in Fig. C.12. We remark that although the PDIS algorithm is designed by assuming nonadditive nonlinear interactions, its performance lags behind the proposed RFNL-TIRSO algorithm, which assumes additive nonlinearities.

This is because the RFNL-TIRSO algorithm employs an RLS loss function, which results in an improved convergence speed compared to the LMS loss used in PDIS.

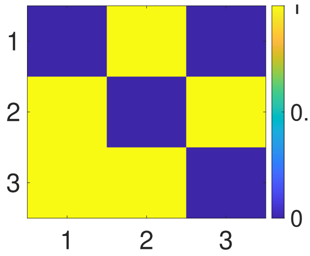


Figure C.8: Orginal dependency.

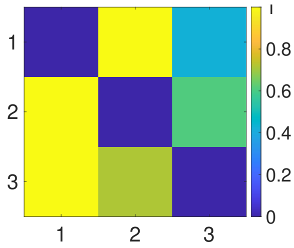


Figure C.9: Estimated dependency.

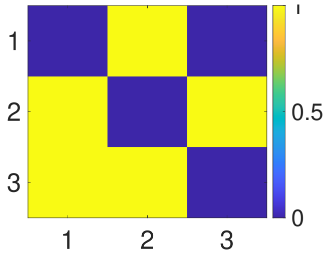


Figure C.10: Estimated binary dependency.

Figure C.11: Lorenz graph detection using RFNL-TIRSO: (a) True Binary dependency, (b) Estimated dependency, (c) Binary estimated dependency by stetting threshold as .5.

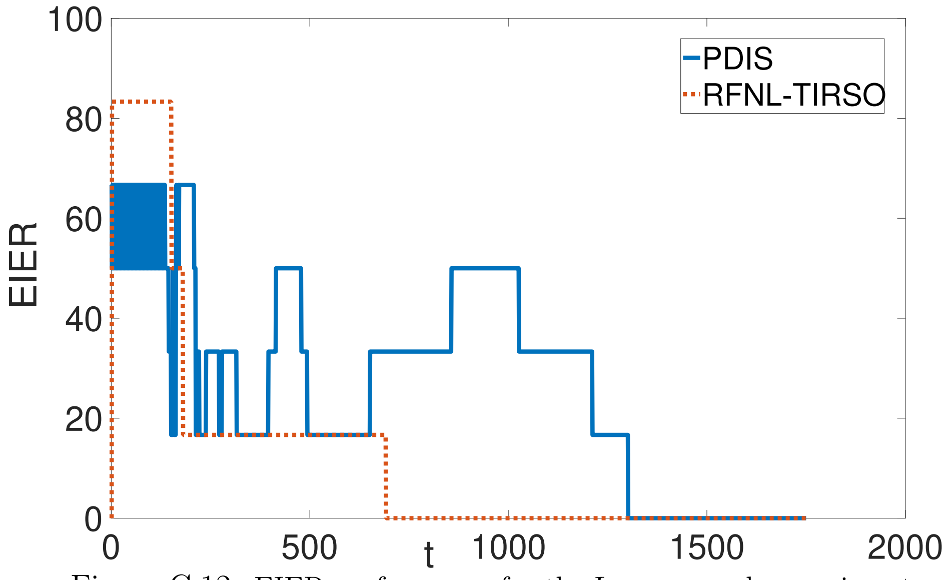


Figure C.12: EIER performance for the Lorenz graph experiment.

#### C.5.1.3 Numerical Evaluation of Dynamic Regret

A theoretical bound of the dynamic regret $\boldsymbol{R}_n[T]=\boldsymbol{R}_n^{\mathrm{rf}}[T]+\boldsymbol{\xi}_n[T]$ has been derived in Appendix C.4.1. In this section, using experiments conducted on synthetic data, we numerically compute the dynamic regret of RFNL-TIRSO w.r.t. the optimal cost in the RF space, defined as $\boldsymbol{R}_n^{\mathrm{rf}}[t]=\sum_{\tau=P}^{t-1}(h_\tau^n(\boldsymbol{\alpha}_n[\tau])-h_\tau^n(\boldsymbol{\alpha}_n^*[\tau]))$, for $t=1,\dots,1000$. This allows validating experimentally our theoretical results. Here, $\boldsymbol{\alpha}_n[\tau]$ is the RF coefficient estimated using RFNL-TIRSO at time $\tau$, and $\boldsymbol{\alpha}_n^*[\tau]$ is the optimum RF coefficient, computed using a standard gradient descent algorithm until convergence. We remark that the estimation of $\boldsymbol{\alpha}_n^*[\tau]$ involves very high computational complexity compared to that of $\boldsymbol{\alpha}_n[\tau]$. In Fig. C.13, we plot $\boldsymbol{R}_n^{\mathrm{rf}}[t]$ and its rate of change w.r.t. time $\boldsymbol{R}_n^{\mathrm{rf}}[t]/t$. In this experiment, we used the same data generation mechanism involving the nonlinear dependencies $g_1$ and $g_2$, as explained in C.5.1.1, having topology change points at $t=250$ and $t=500$. Figure C.13 shows that $\mathbf{R}^{\mathrm{rf}}[t]$ is sublinear w.r.t. $t$ and $\mathbf{R}^{\mathrm{rf}}[t]/t$ is negligibly small, which is in agreement with the theoretical results stated in **Lemma 2**. We note that a numerical evaluation of the second component of the dynamic regret $\boldsymbol{\xi}_n[t]$ is a daunting, complex process since it involves finding the optimal parameters in a high dimensional RKHS. However, as shown in (C.67) we remark that $\boldsymbol{\xi}_n[t]/t$ is theoretically bounded by the value $\epsilon L_f C$, where $\epsilon$ is a user-controlled parameter. $\boldsymbol{\xi}_n[t]/t$ can be made small, yielding a value of $\boldsymbol{R}_n[t]/t$ that can be upper bounded by a small constant for $t\to\infty$.

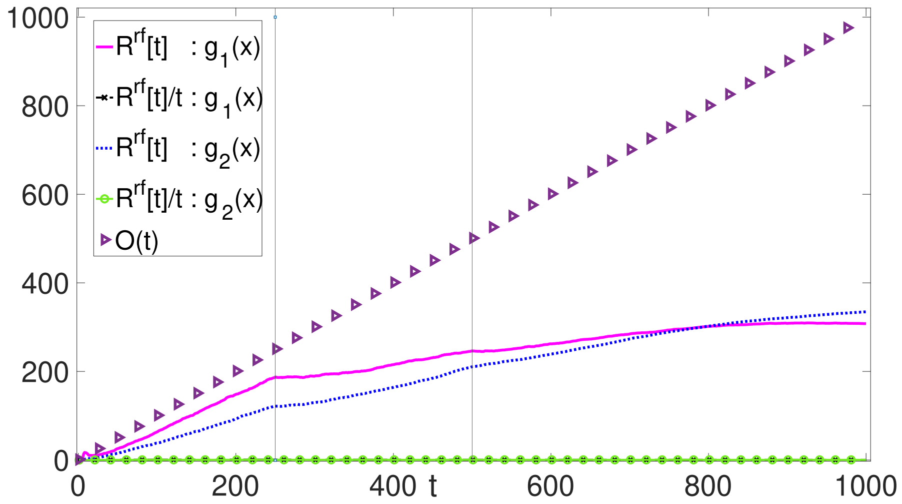


Figure C.13: Regret w.r.t. optimal cost function in RF space. Vertical lines indicate the topology change points.

### C.5.2 Experiments using Real Data Sets

#### C.5.2.1 Oil and Gas Platform Data

This section is dedicated to experiments using real data collected from Lundin's Offshore Oil and Gas (O and G) platform Edvard-Grieg[2]. We collected multivariate

[2] https://www.lundin-energy.com/

time series data from 24 nodes (numbered as $n = 1, 2, \ldots, 24.$) of the plant corresponding to various temperature (T), pressure (P), and oil-level (L) sensors. The sensors are placed in the separators of decantation tanks separating oil, gas and water. The time series are obtained by uniformly sampling the sensor readings with a sampling rate of 5 seconds. We assume that the hidden logic dependencies are present in the network due to the various existing physical connections and control actuators. The data obtained from the sensors are preprocessed by normalizing them to zero mean unit variance signals.

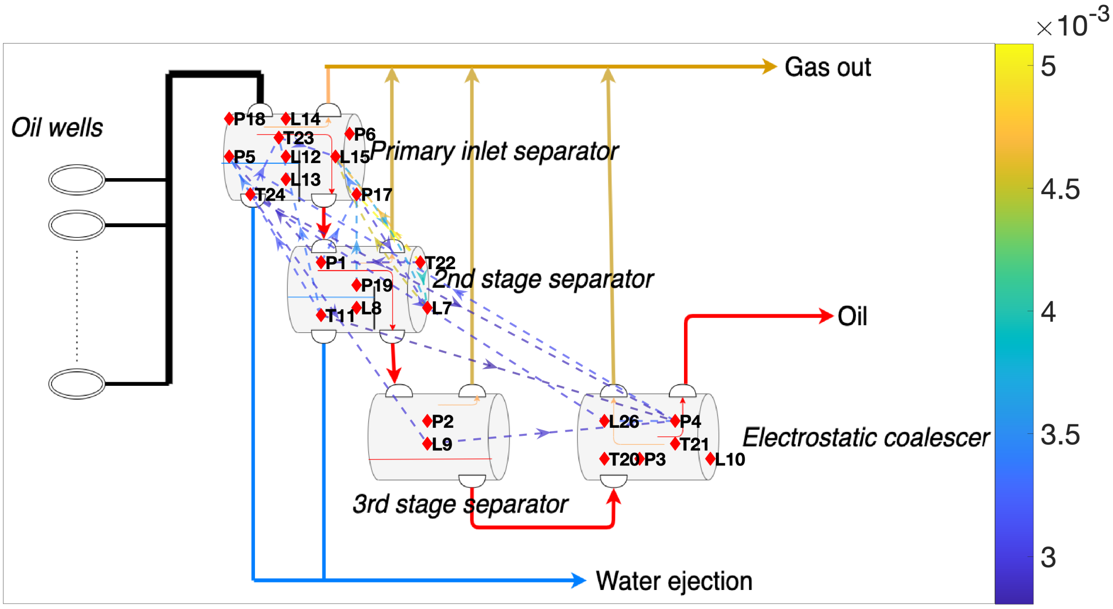


Figure C.14: Causality graph estimated using RFNL-TIRSO for Oil and Gas platform.

The dependencies are learned using RFNL-TIRSO ($D = 10$), RFNL-TISO, and TIRSO by assuming a VAR model of order $P=12$. A Gaussian kernel having a variance of 1 is used in all the experiments with hyperparameter setting $\lambda = 0.1$ and step size $a_t = 1/\Lambda_{max}(\boldsymbol{\phi}[t])$ (tuned to obtain minimum NMSE). The estimated dependencies are visualized in Fig. C.14 using the $\ell_2$ norms $\|\boldsymbol{\alpha}_{n,n'}[t]\|_2$. RFNL-TIRSO identifies interpretable connections; for instance, two pressure sensors in the same separator are connected, and the oil level in separator-1 is connected to the pressure variation in separator-2. Further, as expected, most of the identified interactions are local (e.g., interactions inside a separator), with very few long-distance interactions (e.g., interactions between two separators). The strong local interactions among variables such as temperature, pressure, and oil level inside a container are directly linked to fluid dynamics of the oil and gas in the closed chamber as dictated by the differential equations governing these variables [94]. However, various control mechanisms governing the whole oil and gas platform and the physical connections across different chambers can also cause some longer-distance non-trivial interactions, although they will not typically be as predominant as the local interactions.

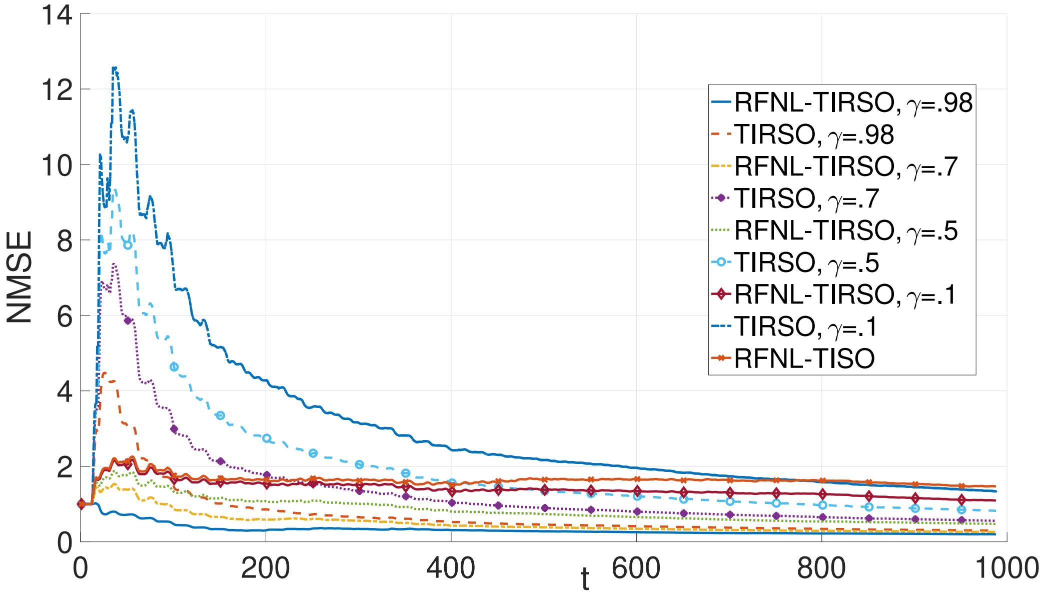


Figure C.15: NMSE comparison: data from the Oil and Gas platform.

For instance, the primary inlet separator and the electrostatic coalescer can interact despite not being physically connected. When there are changes in the oil level within the coalescer, it can affect the head of the system, leading to changes in the pressure and oil level within the primary inlet separator that operates based on gravity.

We wish to note that the estimated dependencies can be interpreted as an abstract graph representation of various physics-based equations describing the space-temporal variation of the signals. Ground truth dependencies are not available in this experiment, and evaluating the estimated graph using the underlying differential physics-based equations governing the space-time system is a tedious procedure that is beyond the scope of this study. However, we demonstrate the ability of the algorithms to learn causal dependencies based on the accuracy of time series forecasting using the learned VAR model. A good prediction accuracy implies that the estimated causal dependencies are close to the underlying unknown real dependencies. As a side note, we highlight that the time series forecasting is a challenging problem having enormous applications in various fields such as finance engineering, traffic forecast, sensor network etc. The prediction accuracy is computed using normalized mean squared error (NMSE):

$$\text{NMSE (T)} = \frac{\sum_{t=1}^{T}(y_n[t+t_{step}] - \hat{y}_n[t+t_{step}])^2}{\sum_{t=1}^{T}(y_n[t+t_{step}])^2}, \tag{C.36}$$

where $\hat{y}_n[t+t_{step}]$ is the estimate of the time series generated by the $n^{th}$ node at time instant $t+t_{step}$ based on the VAR model learned at time $t$. Figure C.15 shows the NMSE of the estimated signals corresponding to a particular sensor $n = 8$ using various algorithms. We discard the PDIS algorithm in this experiment since it is not designed for signal prediction. NMSE is calculated according to (C.36) with $t_{step} = 12$, which refers to one minute ahead prediction. For RFNL-TIRSO and TIRSO, we conduct the experiments by varying the forgetting factor $\gamma \in \{0.1, 0.5, 0.7, 0.98\}$. We note that the best NMSE of the RFNL-TIRSO algorithm is obtained at $\gamma = .98$, and it outperforms all the competitors. It is interesting to observe that as $\gamma$ reduces, the performance of RFNL-TIRSO becomes close to RFNL-TISO, as expected from (F.12). Additionally, we plot the dynamic regret and cumulative variation of the optimal parameter estimates in Fig. C.16, which shows that our algorithm is able to track the topology even if the optimal topology is changing.

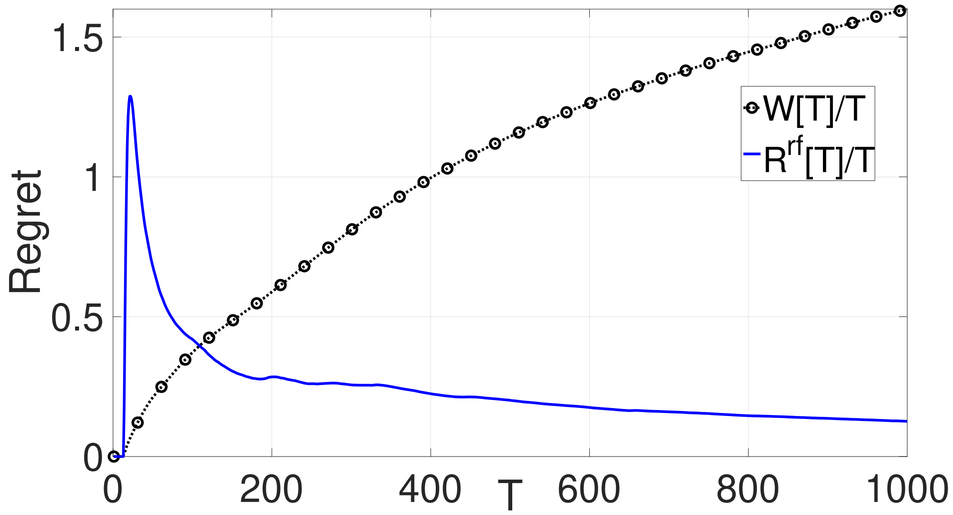


Figure C.16: Rate of change of regret and path length: data from Edvard-Grieg Oil and Gas platform.

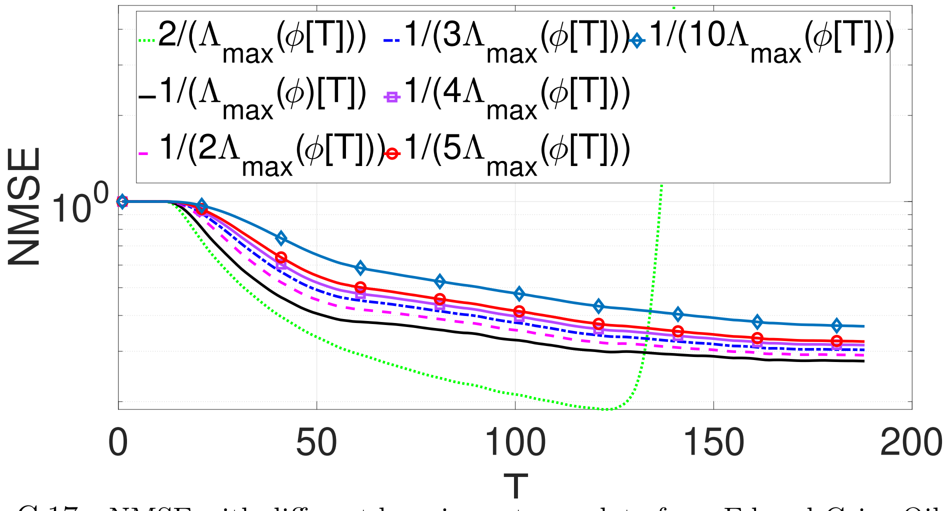


Figure C.17: NMSE with different learning rate $a_t$: data from Edvard-Grieg Oil and Gas platform.

In section Appendix C.4, we show that the RFNL-TIRSO converges if the learning rate is less than $1/L$, where $L$ is the upper bound of $\Lambda_{max}(\boldsymbol{\phi}[T])$. The performance of RFNL-TIRSO under various learning rates is shown in Fig. C.17. Intuitively as the learning rate increases, RFNL-TIRSO converges faster; and when the learning rate is increased beyond $1/L$, convergence is not guaranteed, as evidenced in Fig. C.17. Note that if the data has high variance, the value of $\Lambda_{max}(\boldsymbol{\phi}[T])$ will be obviously high, necessitating the use of a lower learning rate to ensure the algorithm convergence.

#### C.5.2.2 Epileptic data set

The dataset used for this experiment [92] is collected from the Children's Hospital Boston, and it consists of EEG recordings from pediatric subjects with intractable seizures. Subjects were monitored during several days following withdrawal of anti-seizure medication to characterize their seizures and assess their candidacy for surgical intervention. The electrode positions and the nomenclature used during the EEG recordings were based on the well known *International 10-20 system* standard. All signals were sampled at 64 samples per second, and there is a total of 23 Channels: FP1:F7, F7:T7, T7:P7, P7:O1, FP1:F3, F3:C3, C3:P3, P3:O1, FP2:F4, F4:C4, C4:P4, P4:O2, FP2:F8, F8:T8, T8:P8, P8:O2, FZ:CZ, CZ:PZ, P7:T7, T7:FT9, FT9:FT10, FT10:T8, and 2T8:P8, which measures the potential difference between the corresponding electrodes.

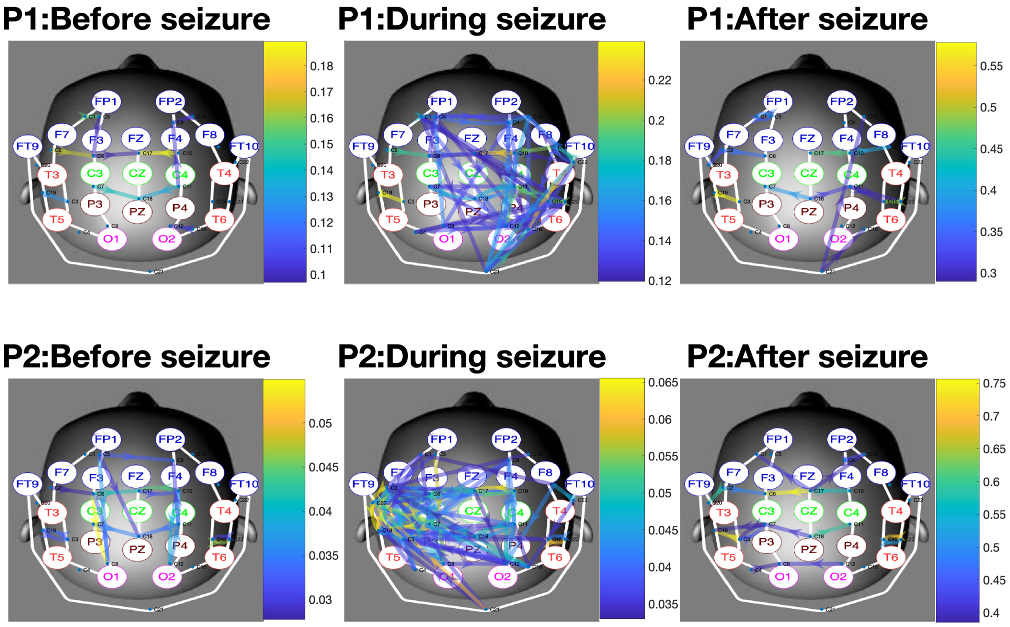


Figure C.18: Estimated brain topology for the subjects P1 and P2 during various stages of seizure.

The estimated brain topology using RFNL-TIRSO ($P = 2, D = 20$) at various time instants (before the seizure, during a seizure, after seizure), visualized using the $\ell_2$ norms $\|\boldsymbol{\alpha}_{n,n'}[t]\|_2$, are shown in Fig. C.18. It is observed that the estimated topologies before and after the seizure are very similar, with connections concentrated across certain brain regions. However, during the seizure, the topologies get more disrupted, which agrees with the observations in [95]. This disruption can be attributed to an increase in pathogenic neural discharge during the seizure [96].

The brain can be divided into several regions, namely, temporal, frontal, occipital, parietal and central. Epilepsies are generally classified according to the region of the brain where they originate, with common classifications including temporal

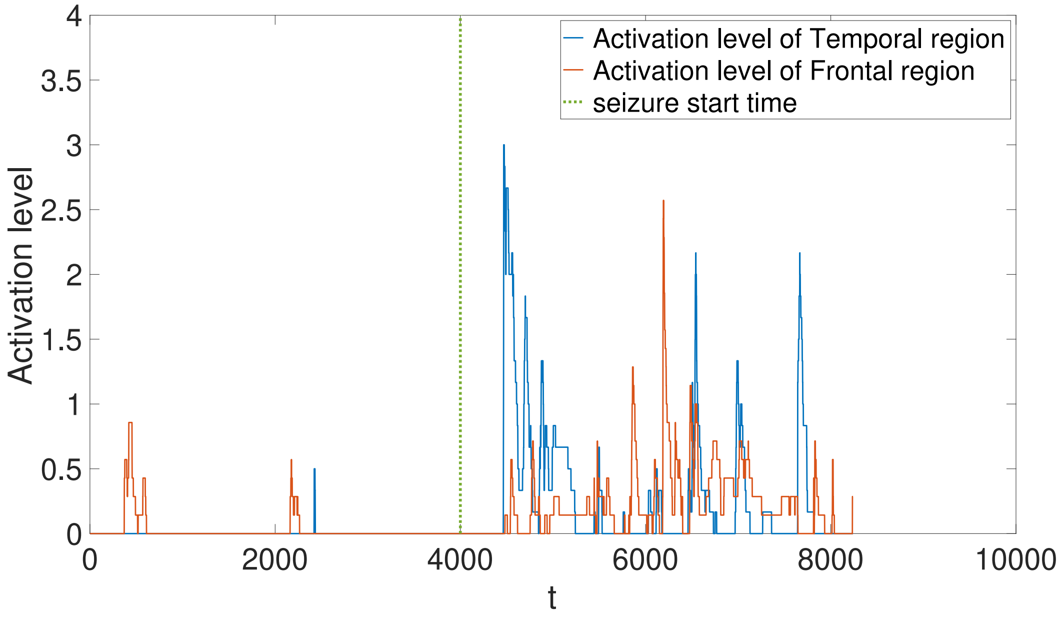


Figure C.19: Subject: P1, Category: Temporal Lobe Epilepsy.

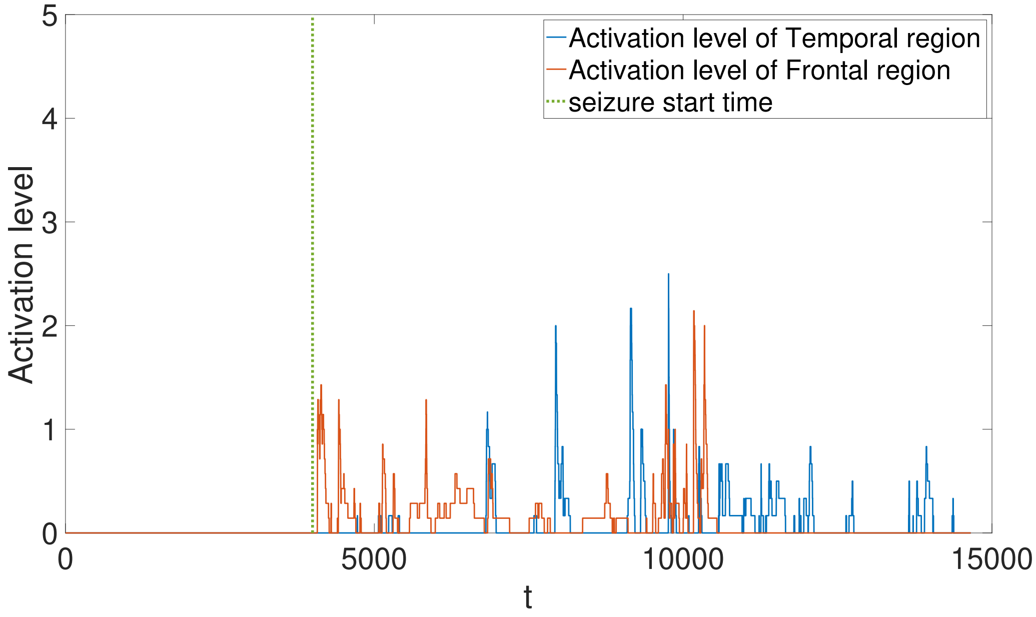


Figure C.20: Subject: P2, Category: Frontal Lobe Epilepsy.

Figure C.21: Activation levels in 'T' and 'F' regions of the brain.

lobe (TL) epilepsy and frontal lobe (FL) epilepsy [97]. In TL epilepsy, more inter-region connections will originate from the temporal region, whereas in FL epilepsy, such connections are originated from the frontal region. To showcase this, we next present an experiment with the brain data of P1 and P2, respectively, belonging to the TL and FL epilepsy categories [98]. To measure the activity level of different brain regions, we group all the channels connected to the 'temporal' region into one group (group-T) and the 'frontal' region into another group (group-F). Note that all the connections between the 'frontal' and the 'temporal' regions are excluded in this experiment. We define the activation level of a group as the sum of the degrees of all the nodes belonging to the group divided by the total number of nodes present in the group, where the degree of a node refers to the total number of edges connected to the node. The activation level of each group for P1 and P2 are shown in Fig. C.19 and Fig. C.20, respectively. From the figures, it is observed that for P1 and P2, the activation levels of group-T and group-F, respectively, spike first, and then the activation spreads across the other brain region. These observations align with the characteristics of TL and FL epilepsies.

## C.6 Conclusion

An online nonlinear topology identification algorithm termed RFNL-TIRSO is proposed in this paper. The multivariate time series data received in sequential form are processed online to estimate time-varying nonlinear dependencies. It has been proven that RFNL-TIRSO follows a sublinear dynamic regret, which guarantees the tracking capability of the algorithm in dynamic environments. The performance of RFNL-TIRSO is evaluated using real and synthetic data sets, and the algorithm outperforms the state-of-the-art online topology estimation methods.

## C.7 Proof of Lemma 2

In this section, we derive a theoretical upper bound for $\boldsymbol{R}_n^{\mathrm{rf}}(T)$. Since the function $h_t^n$ is convex

$$\begin{aligned}\boldsymbol{R}_n^{\mathrm{rf}}(T) &= \sum_{t=P}^{T-1}\left[h_t^n(\boldsymbol{\alpha}_n[t]) - h_t^n(\boldsymbol{\alpha}_n^*[t])\right] \\ &\leq \sum_{t=P}^{T-1} \nabla h_t^n(\boldsymbol{\alpha}_n[t])^\top(\boldsymbol{\alpha}_n[t] - \boldsymbol{\alpha}_n^*[t]).\end{aligned} \tag{C.37}$$

Apply Cauchy-Schwarz inequality on right hand side to get

$$\begin{aligned}\boldsymbol{R}_n^{\mathrm{rf}}(T) =&\sum_{t=P}^{T-1}\left[h_t^n(\boldsymbol{\alpha}_n[t]) - h_t^n(\boldsymbol{\alpha}_n^*[t])\right] \\ \leq&\sum_{t=P}^{T-1} \|\nabla h_t^n(\boldsymbol{\alpha}_n[t])\|_2 \; \|\boldsymbol{\alpha}_n[t] - \boldsymbol{\alpha}_n^*[t]\|_2.\end{aligned} \tag{C.38}$$

The optimality gap of any proximal gradient descent algorithm with an objective function having 1) a strongly convex and Lipschitz smooth loss function and 2) a Lipschitz continuous regularizer is derived in [32]. We can show that RFNL-TIRSO is a proximal gradient descent algorithm by following the proofs provided in [48]. Hence, the cumulative optimality gap is bounded as

$$\sum_{t=P}^{T-1} \|\boldsymbol{\alpha}_n[t] - \boldsymbol{\alpha}_n^*[t]\|_2 = \|\boldsymbol{\alpha}_n^*[P]\|_2 + \boldsymbol{W}_n(T), \tag{C.39}$$

where $\boldsymbol{W}_n(T) = \sum_{t=P}^{T-1} \|\boldsymbol{\alpha}_n^*[t] - \boldsymbol{\alpha}_n^*[t-1]\|_2$ is the path length, which is a measure of cumulative variation of the optimality gap. Next, we bound for the term $\|\nabla h_t^n(\boldsymbol{\alpha}_n[t])\|_2$ in (C.38).

**Lemma 1.** *Under the assumptions A1, A3 and A4,*

$$\|\nabla h_t^n(\boldsymbol{\alpha}_n[t])\|_2 \leq \Big(\Big(1 + \frac{L}{\rho_l}\Big)\sqrt{2PNDB_y} + \lambda\sqrt{PN}\Big).$$

*Proof*: The cost function consists of a differentiable loss function $\tilde{\ell}_t^n$ and a non-differentiable regularizer $\boldsymbol{\omega}^n$. We introduce the notation $\boldsymbol{u}^n$ to denote a subgradient of the regularizer $\boldsymbol{\omega}^n(\boldsymbol{\alpha}_n[t])$. The gradient of the entire cost function can be bounded by bounding the gradient of these two terms:

$$\|\nabla h_t^n(\boldsymbol{\alpha}_n[t])\|_2 \leq \|\nabla \tilde{\ell}_t^n(\boldsymbol{\alpha}_n[t])\|_2 + \|\boldsymbol{u}^n\|_2. \tag{C.40}$$

The term $\|\nabla \tilde{\ell}_t^n(\boldsymbol{\alpha}_n[t])\|_2$ is bounded in **Lemma 1.2** using **Lemma 1.1**, and the term $\|\boldsymbol{u}^n\|_2$ is bounded in **Lemma 1.3**.

**Lemma 1.1.** Under assumptions A1 and A3

$$\|\boldsymbol{\alpha}_n[t+1]\|_2 \leq (1 - a_t\rho_l)\|\boldsymbol{\alpha}_n[t]\|_2 + a_t\sqrt{2PNDB_y}.$$

*Proof:* From **Lemma 7** in [48] we have,

$$\|\boldsymbol{\alpha}_n[t+1]\|_2 \leq (1 - a_t\rho_l)\|\boldsymbol{\alpha}_n[t]\|_2 + a_t\|\boldsymbol{r}_n[t]\|_2. \tag{C.41}$$

Using (F.15), we can bound $\|\boldsymbol{r}_n[t]\|_2$ as

$$\begin{aligned}
\|\boldsymbol{r}_n[t]\|_2 &= \|\mu \sum_{\tau=P}^{t} \gamma^{t-\tau} y_n[\tau]\boldsymbol{z}_{\boldsymbol{v}}(\tau)\|_2 \\
&\leq \mu\| \sum_{\tau=P}^{t} \gamma^{t-\tau} y_n[\tau]\mathbf{1}_{2PND}\|_2 &\text{(C.42)}\\
&\leq \mu\sqrt{2PNDB_y}\gamma^t \sum_{\tau=P}^{t} (\frac{1}{\gamma})^\tau &\text{(C.43)}\\
&= \sqrt{2PNDB_y}(1 - \gamma^{t-P+1}) &\text{(C.44)}\\
&\leq \sqrt{2PNDB_y}. &\text{(C.45)}
\end{aligned}$$

Inequality (C.42) is obtained by replacing the RF vector (sinusoidal components) with an all-one vector having a higher norm, (C.43) is obtained using the assumption A1, (C.44) follows from $u = 1 - \gamma$, and (C.45) follows from $\gamma \leq 1$. **Lemma 1.1** is proved by substituting (C.45) in (C.41).

**Lemma 1.2.** Under assumptions A1, A3, and A4, the RFNL-TIRSO algorithm with step size parameter $a_t = \frac{1}{L}$ satisfies

$$\|\nabla \tilde{\ell_t^n}(\boldsymbol{\alpha}_n[t])\|_2 \leq \Big(1 + \frac{L}{\rho_l}\big)\sqrt{2PNDB_y}.$$

*Proof:* Invoke **Lemma 1.1**, set $a_t = a$, and let $\delta = (1 - a\rho_l)$ and $0 \leq \delta \leq 1$, to get

$$\|\boldsymbol{\alpha}_n[t+1]\|_2 \leq \delta\|\boldsymbol{\alpha}_n[t]\|_2 + a_t\sqrt{2PNDB_y} \tag{C.46}$$

The bound of $\|\boldsymbol{\alpha}_n[t+1]\|_2$ in terms of the norm of the initial estimate $\|\boldsymbol{\alpha}_n[P]\|_2$ is obtained by $t - P + 1$ recursion of (C.46):

$$\begin{aligned}
\|\boldsymbol{\alpha}_n[t+1]\|_2 &\leq \delta^{t-P+1}\|\boldsymbol{\alpha}_n[P]\|_2 + a\sqrt{2PNDB_y}\sum_{i=0}^{t-P}\delta^i \\
&= \frac{a\sqrt{2PNDB_y}(1 - \delta^{t-P+1})}{1 - \delta} \qquad (C.47) \\
&\leq \frac{a\sqrt{2PNDB_y}}{1 - (1 - a\rho_l)}) = \frac{1}{\rho_l}\sqrt{2PNDB_y} \qquad (C.48)
\end{aligned}$$

In (C.47), we assumed that the RF coefficients are initialized with zeros, i.e., $\boldsymbol{\alpha}_n[P] = \mathbf{0}_{2PND}$.

Using (C.48) and (C.45), we can bound gradient:

$$\begin{aligned}
\|\nabla \tilde{\ell_t^n}(\boldsymbol{\alpha}_n[t])\|_2 &= \|\boldsymbol{\phi}[t]\boldsymbol{\alpha}_n[t] - \boldsymbol{r}_n[t]\|_2 \text{ (from (F.18))} \\
&\leq \|\boldsymbol{\phi}[t]\boldsymbol{\alpha}_n[t]\|_2 + \|\boldsymbol{r}_n[t]\|_2 \\
&\leq \Lambda_{max}(\boldsymbol{\phi}[t])\|\boldsymbol{\alpha}_n[t]\|_2 + \|\boldsymbol{r}_n[t]\|_2 \qquad (C.49) \\
&= L\frac{\sqrt{2PNDB_y}}{\rho_l} + \sqrt{2PNDB_y} \qquad (C.50) \\
&\leq \Big(1 + \frac{L}{\rho_l}\Big)\sqrt{2PNDB_y} \qquad (C.51)
\end{aligned}$$

Inequality (C.49) holds since spectral norm of $\boldsymbol{\phi}[t] = \Lambda_{max}(\boldsymbol{\phi}[t])$, whereas (C.50) is obtained by combining the Assumption A4, (C.48), and (C.45). Next, we bound $\|\boldsymbol{u}^n\|_2$.

**Lemma 1.3.** The norm of a subgradient of the regularizer can be bounded as

$$\|\boldsymbol{u}^n\|_2 \leq \lambda\sqrt{PN}.$$

*Proof:* To prove **Lemma 1.3**, we apply **Lemma 2.6** from [68] which states that every subgradient of $\omega^n(.)$ is bounded by its Lipschitz continuity parameter $L_{\omega^n}$. In the following, we show that $L_{\omega^n} = \lambda\sqrt{PN}$. Lipschitz continuity of $\omega^n$ means there exists $L_{\omega^n} > 0$ such that

$$|\omega^n(\boldsymbol{a}) - \omega^n(\boldsymbol{b})| \leq L_{\omega^n}\|\boldsymbol{a} - \boldsymbol{b}\|_2 \tag{C.52}$$

for all real $\boldsymbol{a}$ and $\boldsymbol{b}$. From the group-Lasso regularizer, we have

$$\omega^n(\boldsymbol{x_n}) = \lambda \sum_{n'=1}^{N} \sum_{p=1}^{P} \|\boldsymbol{x}_{n,n'}^{(p)}\|_2. \tag{C.53}$$

Expanding the left-hand side of (C.52) using (C.53) yields

$$
\begin{aligned}
&|\omega^n(\boldsymbol{a}_n) - \omega^n(\boldsymbol{b}_n)| && \text{(C.54)}\\
&= \lambda \Big| \sum_{n'=1}^{N} \sum_{p=1}^{P} \|\boldsymbol{a}_{n,n'}^{(p)}\|_2 - \sum_{n'=1}^{N} \sum_{p=1}^{P} \|\boldsymbol{b}_{n,n'}^{(p)}\|_2 \Big| && \text{(C.55)}\\
&= \lambda \Big| \sum_{n'=1}^{N} \sum_{p=1}^{P} \|\boldsymbol{a}_{n,n'}^{(p)}\|_2 - \|\boldsymbol{b}_{n,n'}^{(p)}\|_2 \Big| && \text{(C.56)}\\
&\leq \lambda \sum_{n'=1}^{N} \sum_{p=1}^{P} \Big| \|\boldsymbol{a}_{n,n'}^{(p)}\|_2 - \|\boldsymbol{b}_{n,n'}^{(p)}\|_2 \Big| && \text{(C.57)}\\
&\leq \lambda \sum_{n'=1}^{N} \sum_{p=1}^{P} \|\boldsymbol{a}_{n,n'}^{(p)} - \boldsymbol{b}_{n,n'}^{(p)}\|_2 && \text{(C.58)}\\
&\leq \lambda \sqrt{PN} \|\boldsymbol{a}_n - \boldsymbol{b}_n\|_2. && \text{(C.59)}
\end{aligned}
$$

In the above derivation, inequality (C.57) follows from the triangle inequality, inequality (C.58) from the reverse triangle inequality and (C.59) from the basic inequality $\|q\|_1 \leq \sqrt{M}\|q\|_2,\ q \in R^M$. From (C.59), we obtain the required Lipschitz parameter to be $\lambda\sqrt{PN}$.

Substitute the bounds of $\|\nabla l_t^n(\boldsymbol{\alpha}_n[t])\|_2$ given by **Lemma 1.2** and $\|\boldsymbol{u}^n\|_2$ given by **Lemma 1.3** in (C.40) to complete the proof of **Lemma 1**. Finally, the proof of **Lemma 2** can be completed by substituting **Lemma 1** and (C.39) in (C.38).

## C.8 Proof of Lemma 3

The cumulative approximation error due to the RF approximation is

$$\boldsymbol{\xi}_n[T] \leq \left| \sum_{t=P}^{T-1} \Big[ h_t^n(\boldsymbol{\alpha}_n^*[t]) - h_t^n(\boldsymbol{\beta}_n^*[t]) \Big] \right|. \tag{C.60}$$

Using the triangle inequality,

$$
\begin{aligned}
\boldsymbol{\xi}_n[T] &\leq \sum_{t=P}^{T-1} \Big| h_t^n(\boldsymbol{\alpha}_n^*[t]) - h_t^n(\boldsymbol{\beta}_n^*[t]) \Big| \\
&\leq \sum_{t=P}^{T-1} L_h \Bigg| \sum_{n'=1}^{N} \sum_{p=1}^{P} \sum_{t'=P}^{t+p-1} \beta_{n,n',(t'-p)}^{(p)*} \boldsymbol{z}_{\boldsymbol{v},n'}^{(p)}(t)^{\top} \boldsymbol{z}_{\boldsymbol{v},n'}^{(p)}(t') \\
&\qquad\qquad - \beta_{n,n',(t'-p)}^{(p)*} k_{n'}^{(p)} \left(y_{n'}[t-p], y_{n'}[t'-p]\right) \Bigg| \qquad \text{(C.61)}
\end{aligned}
$$

$$\leq \sum_{t=P}^{T-1} L_h \sum_{n'=1}^{N} \sum_{p=1}^{P} \sum_{t'=p}^{t+p-1} \left|\beta^{(p)*}_{n,n',(t'-p)}\right| \times$$

$$\left|\boldsymbol{z}^{(p)}_{\boldsymbol{v},n'}(t)^\top \boldsymbol{z}^{(p)}_{\boldsymbol{v},n'}(t) - k^{(p)}_{n'}(y_{n'}[t-p], y_{n'}[t'-p])\right|. \qquad \text{(C.62)}$$

Inequality (C.61) is obtained from the Lipschitz continuity of the cost function ($L_h > 0$ is the Lipschitz continuity parameter) and (C.62) follows from Cauchy-Schwarz inequality. As shown in [16], it can be proved that for a given shift-invariant kernel $k^{(p)}_{n'}$ (assumption A2), the approximation error due to the random Fourier approximation is bounded by

$$\sup_{y_n(t)} \left|\boldsymbol{z}^{(p)}_{\boldsymbol{v},n'}(t)^\top \boldsymbol{z}^{(p)}_{\boldsymbol{v},n'}(t) - k^{(p)}_{n'}\left(y_{n'}[t-p], y_{n'}[t'-p]\right)\right| \leq \epsilon^p_{n'} \qquad \text{(C.63)}$$

with a probability given by $1 - 2^8(\sigma^p_{n'}/\epsilon^p_{n'})^2 \exp(-D\epsilon^p_{n'}/12)$. Here, $\epsilon^p_{n'} \geq 0$ is a constant and $\sigma^p_{n'}$ is the variance of random feature vector norm. Using (C.63),

$$\boldsymbol{\xi}_n[T] \leq \sum_{t=P}^{T-1} L_h \sum_{n'=1}^{N} \sum_{p=1}^{P} \sum_{t'=P}^{t+p-1} \left|\beta^{(p)*}_{n,n',(t'-p)}\right| \epsilon^p_{n'}. \qquad \text{(C.64)}$$

Let $\epsilon = \max \epsilon^p_{n'}$, which leads to

$$\begin{aligned}
\boldsymbol{\xi}(T) &\leq \sum_{t=P}^{T-1} L_h \epsilon \sum_{n'=1}^{N} \sum_{p=1}^{P} \sum_{t'=P}^{t+p-1} \left|\beta^{(p)*}_{n,n',(t'-p)}\right| && \text{(C.65)}\\
&\leq \sum_{t=P}^{T-1} \epsilon L_h C && \text{(C.66)}\\
&\leq \epsilon L_h T C, && \text{(C.67)}
\end{aligned}$$

where $C$ is a constant and (C.66) follows from the assumption A1: since $y_n(t)$ is bounded, the optimal parameters should also be bounded.

# Appendix D

# PAPER D

**Title**: Online Joint Nonlinear Topology Identification and Missing Data Imputation over Dynamic Graphs

**Authors**: **R. Money**, J. Krishnan, B. Beferull-Lozano

**Conference**: European Signal Processing Conference 2022

# Online Joint Nonlinear Topology Identification and Missing Data Imputation over Dynamic Graphs

R. Money, J. Krishnan, B. Beferull-Lozano

**Abstract:** **Extracting causal graph structures from multivariate time series, termed topology identification, is a fundamental problem in network science with several important applications. Topology identification is a challenging problem in real-world sensor networks, especially when the available time series are partially observed due to faulty communication links or sensor failures. The problem becomes even more challenging when the sensor dependencies are nonlinear and nonstationary. This paper proposes a kernel-based online framework using random feature approximation to jointly estimate nonlinear causal dependencies and missing data from partial observations of streaming graph-connected time series. Exploiting the fact that real-world networks often exhibit sparse topologies, we propose a group lasso-based optimization framework for topology identification, which is solved online using alternating minimization techniques. The ability of the algorithm is illustrated using several numerical experiments conducted using both synthetic and real data.**

## D.1 Introduction

Data analytics on complex networked systems such as large-scale sensor networks, social networks, brain networks, etc., have gained much research attention in the last decade. Most such complex networks generate data in the form of multivariate time series, which are often interdependent. These dependencies can be represented in the form of a graph. Representing and processing data on graph structures have become increasingly important due to diverse range of applications, such as data compression, denoising, change point detection, etc. Often, such dependencies are not directly observable and must be inferred. Identification of causal graph structure from multivariate time series is termed *topology identification*, which is a challenging task due to the nonstationary and nonlinear nature of the dependencies.

It is essential to have sufficient and good quality data when solving a topology identification problem; however, data might not be fully observable in many real-world situations. Sensor networks, for instance, transmit data captured by sensors through communication channels to an end-user for processing. These networks are susceptible to data loss due to sensor failures or communication impairments, making it challenging to identify the topology. A practically significant algorithm for topology identification must be *(i)* capable of working online to handle nonstationary dependencies, *(ii)* capable of recognizing nonlinear dependencies, and *(iii)* capable of dealing with noisy and incomplete observations.

Online linear topology identification is fairly well studied in the literature [1,48]. In [48], an optimization problem is formulated by taking into account the sparse nature of real-world dependencies and solving the problem using composite objective mirror descent (COMID), and in [1], a time-varying convex optimization framework has been used for topology identification. Recently, several works on nonlinear topology identification have been proposed [9–11,21,22,36], among which [21,22,36] propose online solutions for nonlinear topology identification problems, whereas [11] and [9] propose batch solutions using kernel and neural networks, respectively.

While the aforementioned works demonstrate promising results in topology estimation, all assume complete data availability with no sensor failures or communication issues. A joint linear topology identification and missing data imputation using block coordinate descent and Kalman smoothing have been recently proposed in [18]. Similarly, [6] proposes a computationally light approach using inexact proximal gradient descent. However, [18] and [6] assume linear causality, which does not make sense for most real-world time series.

In this paper, we propose an online nonlinear topology identification algorithm accounting for missing data by solving a group lasso-based optimization framework. Considering the well-established underlying theory and the ability to carry out online training, kernels are used to model nonlinearity, which are approximated using random features [16] to control the computational complexity. To the best of our knowledge, this is the first attempt to address jointly *(i)* nonlinearity, *(ii)* nonstationarity, and *(iii)* missing data in topology identification.

# D.2 Problem formulation

## D.2.1 Nonlinear topology identification

A $P$-th order nonlinear vector autoregressive (VAR) process with $N$ number of nodes can be expressed as

$$y_n[t] = \sum_{n'=1}^{N} \sum_{p=1}^{P} f_{n,n'}^{(p)}(y_{n'}[t-p]) + u_n[t], \tag{D.1}$$

where $y_n[t]$ is the observation of the $n$-th time series at time $t$, $f_{n,n'}^{(p)}(.)$ encodes the causal influence of $p$-th time-lagged value of $n'$-th time series on $n$-th time series, and $u_n[t]$ is the observation noise. The nonlinear VAR model is a suitable model owing to the fact that the causal dependencies in the real world are time-lagged in nature. Moreover, the VAR model implies the famous causality hypothesis proposed by Granger [99], under certain assumptions [63].

### D.2.1.1 Kernel representation

We assume that the function in (D.1) belongs to a reproducing kernel Hilbert space (RKHS):

$$\mathcal{H}_{n'}^{(p)} := \left\{ f_{n,n'}^{(p)} \mid f_{n,n'}^{(p)}(y) = \sum_{t=p}^{\infty} \beta_{n,n',t}^{(p)} \, \kappa_{n'}^{(p)}(y, y_{n'}[t-p]) \right\}, \tag{D.2}$$

where $\kappa_{n'}^{(p)}(.,.)$ is a positive definite function that measures the similarity between its arguments, and is termed *kernel*. Every positive definite kernel is associated to a RKHS with inner product $\langle \kappa_{n'}^{(p)}(y, x_1), \kappa_{n'}^{(p)}(y, x_2) \rangle := \sum_{t=0}^{\infty} \kappa_{n'}^{(p)}(y[t], x_1) \kappa_{n'}^{(p)}(y[t], x_2)$ and it satisfies the reproducing property $\langle \kappa_{n'}^{(p)}(y, x_1), \kappa_{n'}^{(p)}(y, x_2) \rangle = \kappa_{n'}^{(p)}(x_1, x_2)$, thus thereby inducing the RKHS norm $\|f_{n,n'}^{(p)}\|_{\mathcal{H}_{n'}^{(p)}}^2 = \sum_{t=0}^{\infty} \sum_{t'=0}^{\infty} \beta_{n,n',t}^{(p)} \, \beta_{n,n',t'}^{(p)} \, \kappa_{n'}^{(p)}(y_n[t], y_n[t'])$. As any function in the RKHS can be expressed as an infinite combinations of kernel evaluations, $f_{n,n'}^{(p)}$ can be expressed as (D.2), with $\beta_{n,n',t}^{(p)}$ being the weight associated with each kernel evaluation. A functional optimization problem can be formulated to obtain the required causal dependency for a given node $n$:

$$\begin{aligned} \left\{ \hat{f}_{n,n'}^{(p)} \right\}_{n',p} &= \underset{\left\{ f_{n,n'}^{(p)} \in \mathcal{H}_{n'}^{(p)} \right\}}{\arg\min} \; \frac{1}{2} \sum_{\tau=P}^{T-1} \Bigg[ y_n[\tau] - \\ & \sum_{n'=1}^{N} \sum_{p=1}^{P} f_{n,n'}^{(p)}(y_{n'}[\tau-p]) \Bigg]^2 + \lambda \sum_{n'=1}^{N} \sum_{p=1}^{P} \Omega \left( \|f_{n,n'}^{(p)}\|_{\mathcal{H}_{n'}^{(p)}} \right), \end{aligned} \tag{D.3}$$

where $\sum_{n'=1}^{N} \sum_{p=1}^{P} \Omega \left( \|f_{n,n'}^{(p)}\|_{\mathcal{H}_{n'}^{(p)}} \right)$ is the regularizer and $\lambda$ is the hyperparemter associated with it. If $\Omega(.)$ is nondecreasing, the solution of (D.3) can be expressed with a finite number of kernel evaluations using Representer Theorem [38]:

$$\hat{f}_{n,n'}^{(p)}(y_{n'}[\tau-p]) = \sum_{t=p}^{p+T-1} \beta_{n,n',(t-p)}^{(p)} \kappa_{n'}^{(p)}(y_{n'}[\tau-p], y_{n'}[t-p]). \tag{D.4}$$

Here, the number of kernel evaluations required is equal to the number of data samples. As the number of data samples increases, the number of optimization variables increases, which is commonly known as the *curse of dimensionality* in kernel formulations. We use the random feature (RF) approximation to mitigate this problem.

### D.2.1.2 RF approximation

RF approximation addresses the curse of dimensionality by restricting the kernel evaluations to an approximate fixed lower-dimensional Fourier space. Furthermore, linear optimization techniques are easier to use in random Fourier space than in infinite-dimensional RKHS. We use shift-invariant kernels to facilitate RF approximation, i.e., $\kappa_{n'}^{(p)}(y_{n'}[\tau], y_{n'}[t]) = \kappa_{n'}^{(p)}(y_{n'}[\tau] - y_{n'}[t])$. According to Bochner's theorem [30], a shift invariant kernel can be represented using an inverse Fourier transform of a probability distribution:

$$\begin{aligned} \kappa_{n'}^{(p)}(y_{n'}[\tau-p], y_{n'}[t-p]) &= \int \pi_{\kappa_{n'}^{(p)}}(v) \; e^{jv(y_{n'}[\tau-p] - y_{n'}[t-p])} dv \\ &= \mathbb{E}_v[e^{jv(y_{n'}[\tau-p] - y_{n'}[t-p])}], \end{aligned} \tag{D.5}$$

where $\mathbb{E}$ is the expectation operator, $\pi_{\kappa^{(p)}_{n'}}(v)$ is the kernel specific probability density function (pdf) and $v$ is the random variable corresponding to the pdf. With sufficient number of i.i.d. samples $\{v_i\}_{i=1}^{D}$, the expectation in (D.5) can be replaced with sample mean:

$$\hat{\kappa}^{(p)}_{n'}\left(y_{n'}[\tau-p], y_{n'}[t-p]\right)=\frac{1}{D}\sum_{i=1}^{D} e^{jv_i(y_{n'}[\tau-p]-y_{n'}[t-p])}. \tag{D.6}$$

Note that (D.6) is an unbiased estimator of the kernel evaluation with a fixed number $D$ of terms [43]. For a Gaussian kernel with variance $\sigma^2$, the inverse Fourier transform can be shown to be also a Gaussian with variance $\sigma^{-2}$. Using this information, the real part of (D.6), which is also an unbiased estimator of kernel evaluation, can be expressed as

$$\hat{\kappa}^{(p)}_{n'}\left(y_{n'}[\tau-p], y_{n'}[t-p]\right) = \boldsymbol{z}^{(p)}_{\boldsymbol{v},n'}[\tau]^{\top}\boldsymbol{z}^{(p)}_{\boldsymbol{v},n'}[t], \tag{D.7}$$

$$\begin{aligned} where,\ \boldsymbol{z}^{(p)}_{\boldsymbol{v},n'}[\tau] = \frac{1}{\sqrt{D}}\Big[&\sin\left(v_1 y_{n'}[\tau-p]\right), \ldots, \sin\left(v_D y_{n'}[\tau-p]\right), \\ &\cos\left(v_1 y_{n'}[\tau-p]\right), \ldots, \cos\left(v_D y_{n'}[\tau-p]\right)\Big]^{\top}. \end{aligned} \tag{D.8}$$

A fixed dimensional ($2D$) approximation of the function $\hat{f}^{(p)}_{n,n'}$ is readily obtained by substituting (D.7) in (D.4):

$$\begin{aligned} \tilde{\hat{f}}^{(p)}_{n,n'}\left(y_{n'}[\tau-p]\right) &= \sum_{t=p}^{p+T-1} \beta^{(p)}_{n,n',(t-p)}\boldsymbol{z}^{(p)}_{\boldsymbol{v},n'}[\tau]^{\top}\boldsymbol{z}^{(p)}_{\boldsymbol{v},n'}[t] \\ &= \boldsymbol{\alpha}^{(p)\top}_{n,n'}\boldsymbol{z}^{(p)}_{\boldsymbol{v},n'}[\tau], \end{aligned} \tag{D.9}$$

where $\boldsymbol{\alpha}^{(p)}_{n,n'} = \sum_{t=p}^{p+T-1}\beta^{(p)}_{n,n',(t-p)}\boldsymbol{z}^{(p)}_{\boldsymbol{v},n'}[t]$. The following notations are introduced to simplify the formulations:

$$\boldsymbol{\alpha}^{(p)}_{n,n'} = [\alpha^{(p)}_{n,n',1}, \ldots, \alpha^{(p)}_{n,n',2D}]^{\top} \in \mathbb{R}^{2D}, \tag{D.10}$$

$$\boldsymbol{z}^{(p)}_{\boldsymbol{v},n'}[\tau] = [z^{(p)}_{\boldsymbol{v},n',1}[\tau], \ldots z^{(p)}_{\boldsymbol{v},n',2D}[\tau]]^{\top} \in \mathbb{R}^{2D}, \tag{D.11}$$

$$z^{(p)}_{\boldsymbol{v},n',k}[\tau] = \begin{cases} \sin(v_{k} y_{n'}[\tau-p]), & \text{if } k \leq D \\ \cos(v_{k-D} y_{n'}[\tau-p]), & \text{otherwise.} \end{cases}$$

The functional optimization (D.3) is reformulated as a parametric optimization problem using (D.9):

$$\left\{\widehat{\boldsymbol{\alpha}}^{(p)}_{n,n'}\right\}_{n',p} = \arg\min_{\left\{\boldsymbol{\alpha}^{(p)}_{n,n'}\right\}} \mathcal{L}^{n}\left(\boldsymbol{\alpha}^{(p)}_{n,n'}\right) + \lambda\sum_{n'=1}^{N}\sum_{p=1}^{P}\Omega(||\boldsymbol{\alpha}^{(p)}_{n,n'}||_2), \tag{D.12}$$

where

$$\mathcal{L}^{n}\left(\boldsymbol{\alpha}^{(p)}_{n,n'}\right) := \sum_{\tau=P}^{T-1}\frac{1}{2}\left[y_n[\tau] - \sum_{n'=1}^{N}\sum_{p=1}^{P}\boldsymbol{\alpha}^{(p)\top}_{n,n'}\ \boldsymbol{z}^{(p)}_{\boldsymbol{v},n'}[\tau]\right]^2, \tag{D.13}$$

which can be expanded in terms of RF components as

$$\mathcal{L}^n\left(\alpha_{n,n',d}^{(p)}\right) := \sum_{\tau=P}^{T-1} \frac{1}{2}\left[y_n[\tau] - \sum_{n'=1}^{N}\sum_{p=1}^{P}\sum_{d=1}^{2D} \alpha_{n,n',d}^{(p)}\ z_{\boldsymbol{v},n',d}^{(p)}[\tau]\right]^2. \tag{D.14}$$

For convenience, the parameters $\{\alpha_{n,n',d}^{(p)}\}$ and $\{z_{\boldsymbol{v},n',d}^{(p)}[\tau]\}$ are stacked in the lexicographic order of the indices $p$, $n'$, and $d$ to obtain the vectors $\boldsymbol{\alpha}_n \in \mathbb{R}^{2PND}$ and $\boldsymbol{z}_{\boldsymbol{v}}[\tau] \in \mathbb{R}^{2PND}$, respectively, which allows to rewrite the loss function as

$$\mathcal{L}^n(\boldsymbol{\alpha}_n) = \frac{1}{2}\sum_{\tau=P}^{T-1}\left[y_n[\tau] - \boldsymbol{\alpha}_n^\top \boldsymbol{z}_{\boldsymbol{v}}[\tau]\right]^2. \tag{D.15}$$

### D.2.2 Missing data

To formulate the topology identification problem with missing data and noisy observation, we assume that only a subset of the nodes is observed. The motif of missing data is represented by the masking vector $\mathbf{m}[t] \in R^N$, where $m_n[t], n = 1, ..., N$, are i.i.d Bernoulli random variables. The observed vector signal $\tilde{\mathbf{y}}[t]$ at time $t$ is given by

$$\tilde{\mathbf{y}}[t] = \mathbf{m}[t] \odot (\mathbf{y}[t] + \mathbf{e}[t]), \tag{D.16}$$

where $\mathbf{y}[t] = [y_1[t], ..., y_n[\tau]]\top \in \mathbb{R}^N$ and $\mathbf{e}[t] \in \mathbb{R}^N$ are the original signal and observation noise in vector form and $\odot$ represents the element wise multiplication.

### D.2.3 Nonlinear topology identification with missing data

A batch formulation for the joint topology identification and missing data imputation can be formulated similarly to [18] and [6] as follows:

$$\begin{aligned}\{\hat{\boldsymbol{\alpha}}, \hat{\mathbf{y}}[\tau]\}_{\tau=P}^{\tau=T-1} &= \arg\min_{\boldsymbol{\alpha},\mathbf{y}[\tau]} \sum_{\tau=P}^{T-1} \frac{1}{2}\|\mathbf{y}[\tau] - \boldsymbol{\alpha}^\top \boldsymbol{z}_v[\tau]\|_2^2 \\ &+ \lambda \sum_{n'=1}^{N}\sum_{d=1}^{2D} \|\boldsymbol{\alpha}_{n,n',d}\|_2 + \sum_{\tau=P}^{T-1} \frac{\nu}{2M_\tau}\|\tilde{\mathbf{y}}[\tau] - \mathbf{m}[\tau] \odot \mathbf{y}[\tau]\|_2^2,\end{aligned} \tag{D.17}$$

where $\boldsymbol{\alpha} = [\boldsymbol{\alpha}_1^\top, \ldots, \boldsymbol{\alpha}_N^\top] \in \mathbb{R}^{2PND} \times \mathbb{R}^N$, $M_\tau$ is cardinality of $\mathbf{m}[\tau]$, and $\nu$ is a hyperparameter that regulates the signal reconstruction part.

## D.3 Joint online estimation of nonlinear topology and missing data

Note that $\boldsymbol{z}_\nu$ depends on $P$ previous values of all the $N$ time series. Hence the required online estimation strategy should estimate $P$ previous values of the time series along with the instantaneous values:

$$\begin{aligned}&\{\widehat{\boldsymbol{\alpha}}, \hat{\mathbf{y}}[t], \{\hat{\mathbf{y}}[\tau]\}_{\tau=t-P}^{t-1}\} = \\ &\arg\min_{\substack{\boldsymbol{\alpha},\mathbf{y}[t] \\ \{\mathbf{y}[\tau]\}_{\tau=t-P}^{t-1}}} \ell_t \left(\boldsymbol{\alpha}, \mathbf{y}[t], \{\mathbf{y}[\tau]\}_{\tau=t-P}^{t-1}\right) + \lambda \sum_{n'=1}^{N} \sum_{p=1}^{P} \|\boldsymbol{\alpha}_{n,n'}^{(p)}\|_2, \end{aligned} \tag{D.18}$$

where the non decreasing function $\Omega(.) = |.|$ and the loss function is defined as

$$\begin{aligned}&\ell_t \left(\boldsymbol{\alpha}, \mathbf{y}[t], \{\mathbf{y}[\tau]\}_{\tau=t-P}^{t-1}\right) = \\ &\frac{1}{2}\|\mathbf{y}[t] - \boldsymbol{\alpha}^\top \boldsymbol{z}_v[t]\|_2^2 + \frac{\nu}{2M_t}\|\tilde{\mathbf{y}}[t] - \mathbf{m}[t] \odot \mathbf{y}[t]\|_2^2. \end{aligned} \tag{D.19}$$

We relax the formulation (D.18) since it is computationally expensive as well as nonconvex. We assume that $\{\hat{\mathbf{y}}[\tau]\}_{\tau=t-P}^{t-1}$ are independent realizations of random variables $\{\mathbf{y}[\tau]\}_{\tau=t-P}^{t-1}$ [6] and obtain a new loss function:

$$\begin{aligned}\tilde{\ell}_t \left(\boldsymbol{\alpha}, \mathbf{y}[t]\right) = &\frac{1}{2}\|\mathbf{y}[t] - \boldsymbol{\alpha}^\top \boldsymbol{z}_v[t]\|_2^2 \\ &+ \frac{\nu}{2M_t}\|\tilde{\mathbf{y}}[t] - \mathbf{m}[t] \odot \mathbf{y}[t]\|_2^2. \end{aligned} \tag{D.20}$$

Now the loss function is convex and separable across $n$. Hence the optimization problem for a node can be expressed as

$$\{\widehat{\boldsymbol{\alpha}}_n, \hat{y}_n[t]\} = \arg\min_{\boldsymbol{\alpha}_n, y_n[t]} \ell_t^n \left(\boldsymbol{\alpha}_n, y_n[t]\right) + \lambda \sum_{n'=1}^{N} \sum_{p=1}^{P} \|\boldsymbol{\alpha}_{n,n'}^{(p)}\|_2, \tag{D.21}$$

$$where \ell_t^n \left(\boldsymbol{\alpha}_n, y_n[t]\right) = \frac{1}{2}\left[y_n[t] - \boldsymbol{\alpha}_n^\top \boldsymbol{z}_v[t]\right]^2 + \frac{\nu}{2M_t}(\tilde{y}_n[t] - m_n[t] y_n[t])^2. \tag{D.22}$$

We use the alternating minimization method in which (D.21) is solved by alternating between two sub-problems that are convex and have closed-form solutions. Since the optimization problem with respect to $y_n[t]$ (the signal reconstruction problem) is quadratic, a closed-form solution can be obtained. The second optimization problem with respect to $\boldsymbol{\alpha}_n$ (topology identification) is in a form similar to the one discussed in [22], where it is solved in a closed form using composite objective mirror descent (COMID) method.

### D.3.1 Signal reconstruction

The signal reconstruction problem can be formulated as

$$\hat{y}_n[t] = \arg\min_{y_n[t]} \ell_t^n \left(\boldsymbol{\alpha}_n, y_n[t]\right). \tag{D.23}$$

The solution of (D.23) is obtained by finding the zero derivative point of the objective function:

$$\hat{y}_n[t] = \frac{\nu m_n[t] \tilde{y}_n[t]}{M_t + \nu m_n[t]} + \frac{k_n[t] M_t}{\nu m_n[t] + M_t}, \tag{D.24}$$

where $k_n[t] = \boldsymbol{\alpha}_n^\top \boldsymbol{z}_v[t]$. Let $\frac{\nu m_n[t]}{M_t + \nu m_n[t]} = q_n[t]$, then,

$$\hat{y}_n[t] = q_n[t] \tilde{y}_n[t] + [1 - q_n[t]] k_n[t]. \tag{D.25}$$

### D.3.2 Topology identification

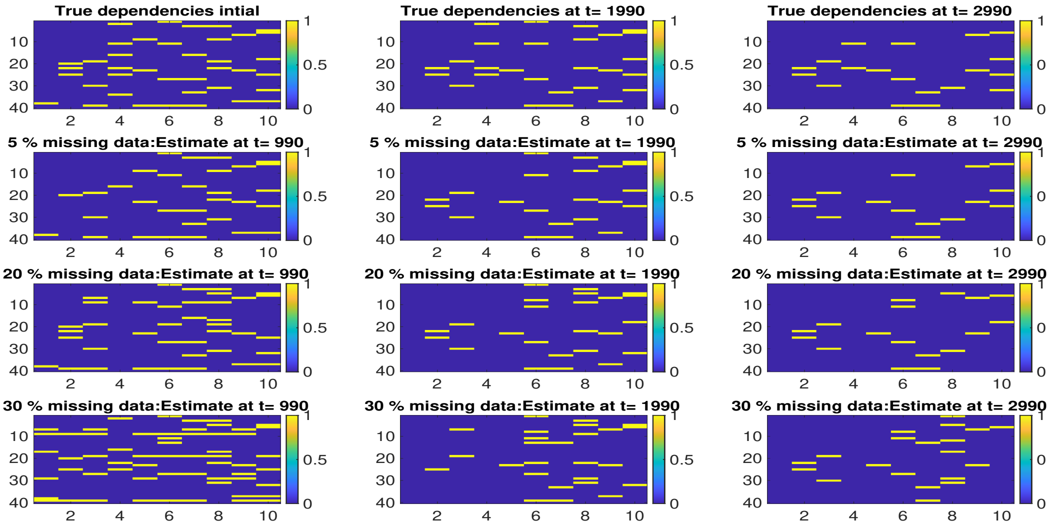


Figure D.1: True edges $(a_{n,n'}^{(p)})$ and estimated weights $(\widehat{b}_{n,n'}^{(p)})$ for various missing data scenarios.

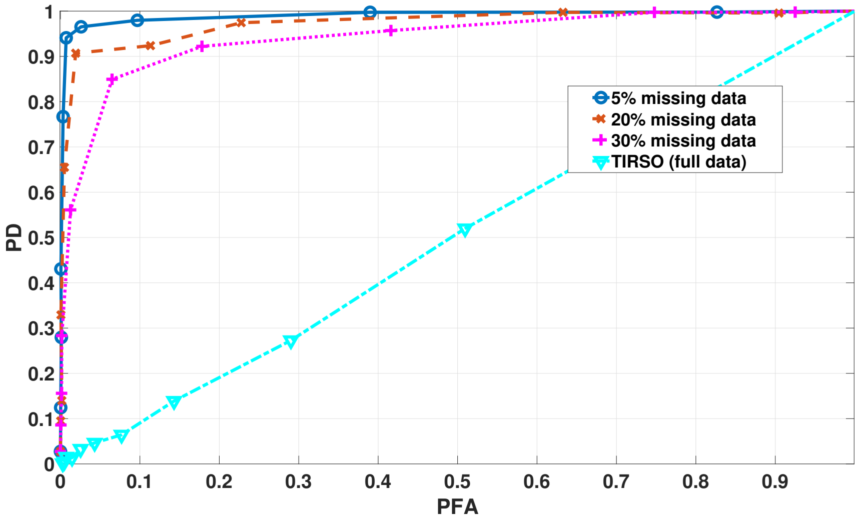


Figure D.2: ROC curve.

Figure D.3: Results: Experiment using synthetic data.

We use the estimates $\{\hat{y}_n[\tau]\}_{\tau=t-P}^{t}$ obtained using (D.25) to find the topology. This sub-problem can be formulated as

$$\widehat{\boldsymbol{\alpha}}_n = \arg \min_{\boldsymbol{\alpha}_n} \ell_t^n(\boldsymbol{\alpha}_n) + \lambda \sum_{n'=1}^{N} \sum_{p=1}^{P} \|\boldsymbol{\alpha}_{n,n'}^{(p)}\|_2. \tag{D.26}$$

where $\ell_t^n(\boldsymbol{\alpha}_n) = \frac{1}{2}[\hat{y}_n[t] - \boldsymbol{\alpha}_n^\top \boldsymbol{z}_{\boldsymbol{v}}[t]]^2$. The convex objective function in (D.26) contains two terms: a smooth loss function and a non-smooth regularizer. Such problems can be solved efficiently using COMID methods [22]. The online COMID update is given by

$$\boldsymbol{\alpha}_n[t+1] = \arg\min_{\boldsymbol{\alpha}_n} J_t^{(n)}(\boldsymbol{\alpha}_n), \tag{D.27}$$

$$\text{where } J_t^{(n)}(\boldsymbol{\alpha}_n) \triangleq \nabla \ell_t^n(\boldsymbol{\alpha}_n[t])^\top [\boldsymbol{\alpha}_n - \boldsymbol{\alpha}_n[t]] + \frac{1}{2\gamma_t}\|\boldsymbol{\alpha}_n - \boldsymbol{\alpha}_n[t]\|_2^2 + \lambda \sum_{n'=1}^{N}\sum_{p=1}^{P} \|\boldsymbol{\alpha}_{n,n'}^{(p)}\|_2. \tag{D.28}$$

In (D.28), $\boldsymbol{\alpha}_n[t] \in \mathbb{R}^{2PND}$ is the estimate of $\boldsymbol{\alpha}_n$ at time $t$. The objective function $J_t^{(n)}(.)$ consists of three terms: *(i)* gradient of the loss function, *(ii)* Bregman divergence $\|\boldsymbol{\alpha}_n - \boldsymbol{\alpha}_n[t]\|_2^2$ chosen such that the optimization problem (D.28) has a closed-form solution ($\gamma_t$ is the step size associated with the divergence), and *(iii)* a sparsity promoting group lasso regularizer. Note that the Bregman divergence term increases stability of the online algorithm by enforcing the next iterate $\boldsymbol{\alpha}_n[t+1]$ to be closer to current iterate $\boldsymbol{\alpha}_n[t]$. The gradient in (D.28) is evaluated as

$$\mathbf{v}_n[t] := \nabla \ell_t^n(\boldsymbol{\alpha}_n[t]) = \boldsymbol{z}_v[t][\boldsymbol{\alpha}_n^\top \boldsymbol{z}_v[t] - \hat{y}_n[t]]. \tag{D.29}$$

A closed-form solution for (D.27) is obtained via the multidimensional shrinkage-thresholding operator:

$$\boldsymbol{\alpha}_{n,n'}^{(p)}[t+1] = [\boldsymbol{\alpha}_{n,n'}^{(p)}[t] - \gamma_t \mathbf{v}_{n,n'}^{(p)}[t]] \times \left[1 - \frac{\gamma_t \lambda}{\|\boldsymbol{\alpha}_{n,n'}^{(p)}[t] - \gamma_t \mathbf{v}_{n,n'}^{(p)}[t]\|_2}\right]_+, \tag{D.30}$$

where $[x]_+ = \max\{0, x\}$. The above solution is a product of two terms: first term minimizes the loss function $\ell_t^n(\boldsymbol{\alpha}_n)$ and the second term enforces sparsity on the updates. The proposed algorithm for jointly estimating the topology and the missing data is summarized in Algorithm 8.

**Algorithm 8:**

```
Result: $\left\{\boldsymbol{\alpha}_{n,n'}^{(p)}[t+1]\right\}_{n,n',p}$, $\hat{\mathbf{y}}[t]$
Initialize $\{\boldsymbol{y}_n[t]\}_{t=1}^{P}$, $\left\{\boldsymbol{\alpha}_{n,n'}^{(p)}[P]\right\}_{n,n',p}$ as all-ones vector, $\lambda$, kernel parameters, $\gamma$, $D$, $\nu$
  (heuristically chosen)
for $t = P, P+1, \ldots$ do
    Get data observation vector $\tilde{\mathbf{y}}_n[t]$ and masking vector $\mathbf{m}[t]$, compute $\boldsymbol{z}_{\boldsymbol{v}}[t]$
    for $n = 1, \ldots, N$ do
        compute $\hat{y}_n[t]$ using (D.25)
        compute $\mathbf{v}_n[t]$ using (D.29)
        for $n' = 1, \ldots, N$ do
            compute $\boldsymbol{\alpha}_{n,n'}^{(p)}[t+1]$ using (D.30)
        end
    end
end
```

## D.4 Experiment

In this section, we test the capability of our algorithm using both synthetic and real data. We generate graph-connected time series with known topologies and varying levels of missing data for synthetic data experiments, whereas, in the second part, we use real data from Lundin's offshore oil and gas platform[1]. The $\ell_2$ norms of the estimated weights ($\widehat{b}_{n,n'}^{(p)}[t] := \|\boldsymbol{\alpha}_{n,n'}^{(p)}[t]\|_2$) are used to visualize the dependencies among the time series. For all the experiments, we used Gaussian reproducing kernel $k$ with variance $\sigma_k^2 = 5$.

### D.4.1 Experiments using Synthetic data

The data used in this experiment are generated using nonlinear VAR model described in (D.1) with $N = 10, P = 4$ and random Gaussian noise with mean 0 and variance 0.01. The nonliner function in (D.1) is taken as $f_{n,n'}^{(p)}(x) = a_{n,n'}^{(p)}(x)g(x)$, $\forall n, n', p$, where $g(x) = 0.25\sin(x^2) + 0.25\sin(2x) + 0.5\sin(x)$ and $a_{n,n'}^{(p)}(x) \in \{0, 1\}$. We term $a_{n,n'}^{(p)}$ as *edge* and when $a_{n,n'}^{(p)} = 0$, it disables the dependencies between the nodes $n$ and $n'$ for the time lag $p$. Furthermore, $a_{n,n'}^{(p)}(x) = 0$, when $g(x) = 0$. The time series are initialized randomly using samples drawn from uniform distribution $\mathcal{U}(0,\, 1)$. To bring time variance in the topology, 30% of the active edges are made to disappear after every 1000 time stamps, and new equal number of different edges are made active. To simulate various missing data scenarios, we generate different masks $\mathbf{m}[t]\ \forall\ t$, whose samples are drawn from Bernoulli distribution with probabilities $0.95, 0.75, 0.65$, corresponds to $5\%, 25\%, 35\%$ of missing data respectively.

In Fig. D.1, we compare the true edges $a_{n,n'}^{(p)}$ and estimated causal weights $\widehat{b}_{n,n'}^{(p)}$ at three different time instants having different edge patterns. The edges and the estimated weights are arranged in a matrix form of size $N \times N$ for $p = 1, 2, \ldots, P$ and are stacked in Fig. D.1, such that the resulting matrices are of size $NP \times N$. The estimated weights are normalized and hard-thresholded to 0 or 1 to have the best match with the edges. It can be observed in Fig. D.1 that for 5% of missing data, the proposed algorithm estimates most of the edges accurately, and as the number of missing data increases, the estimation accuracy decreases. The ROC curve corresponding to the time stamp $t = 990$ is plotted in Fig. D.2 by computing the probability of detection (PD) and the probability of false alarm (PFA). Figure D.2 shows that the areas under all the three curves are close to 1, indicating the characteristics of a good ROC curve. It can also be observed that the area under the curve deviates more from 1 as the number of missing data increases. Also, the ROC curve for a recent online linear topology estimation algorithm termed TIRSO [48] is included in Fig. D.2. Note that TIRSO's ROC is computed based on full data; even then, its performance significantly lags behind the proposed algorithm. Intuitively, JSTIRSO [6], the extension to TIRSO that accounts for missing data, should also perform inferiorly to the proposed algorithm. These observations illustrate how

[1] https://www.lundin-energy.com/

effectively the proposed algorithm identifies nonlinear topologies compared to its linear counterparts.

### D.4.2 Experiments using Real data

We use real data from Lundin's oil and gas plant, consisting of time series recorded from multiple pressure (P), temperature (T), and oil level (L) sensors from $system_{20}$ of the plant. The $system_{20}$ is a plant section where oil, gas, and water are separated from the well extracts. There are 24 sensors in total recording 24 time series, sampled at intervals of $5s$. Below, we examine two different missing data scenarios.

#### D.4.2.1 Missing data due to limited communication capacity

Assume that only a subset of the sensor values can be transmitted at each timestamp due to the limited capacity of the communication channel. We randomly select 8 out of the 24 sensors ($\sim 33.33\%$) at each time stamp and jointly estimate the topologies and the missing data. The true and observed time series of a sensor, along with the reconstructed values, are shown in Fig. D.4, which shows that the proposed algorithm reconstructs the signal even with a high amount of missing data. Since the ground truth dependencies are unavailable, we compare the dependencies estimated from the partial observations  with that from a full observation  in Fig. D.5, which shows that the algorithm can estimate most of the dependencies from the partial observations.

#### D.4.2.2 Missing data due to sensor failure

Here we consider the case where the recording from a particular sensor is missing for a certain period of time due to a sensor failure. In the experiment, time series from sensor-2 are masked from time instant $t = 4000$ to $t = 4200$, which constitutes about 16 minutes of data. Figure D.6 shows that the proposed algorithm reconstructs sensor-2 signals accurately during the missing data interval without having access to any information from sensor-2. This clearly showcases the advantage of exploiting causal dependencies in missing data imputations.

## Conclusion

This paper presents a novel algorithm for joint nonlinear topology identification and missing data imputation. The nonlinear causal dependencies are modeled using a computationally light kernel formulation based on random feature approximations. Experiments on real and synthetic data have demonstrated the effectiveness of the proposed algorithm under various missing data scenarios.

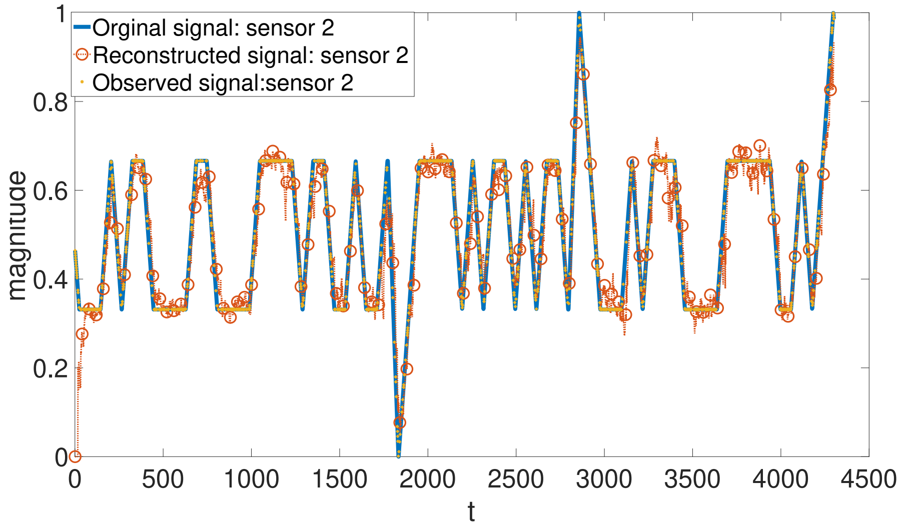


Figure D.4: Original and reconstructed signal when only 33.33% of data is observable.

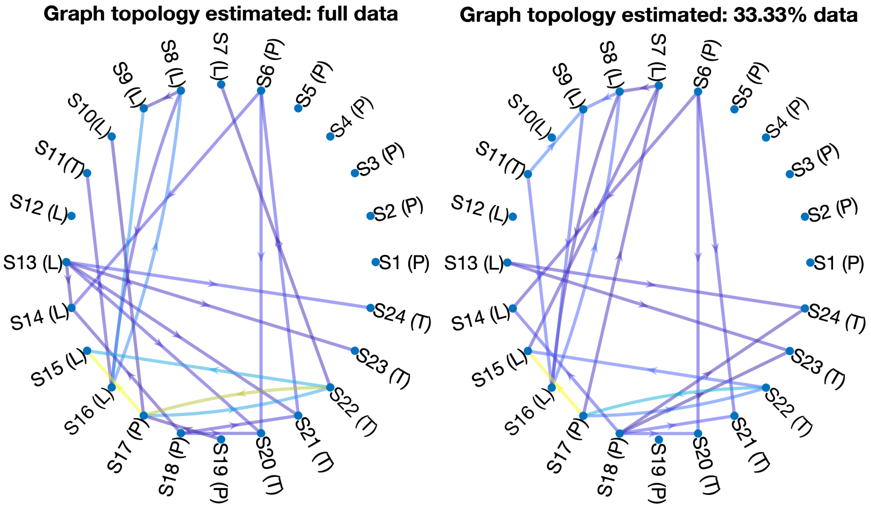


Figure D.5: Causality graph estimated for oil and gas platform (Only the significant edges are shown).

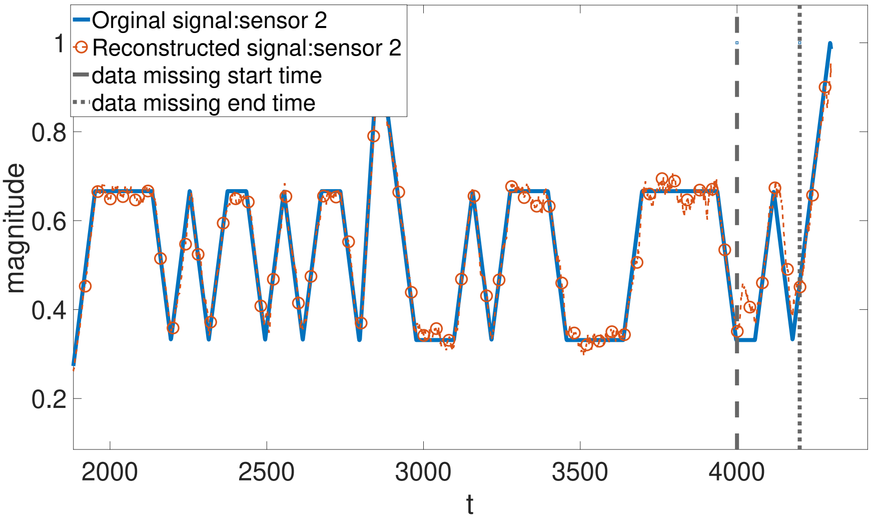


Figure D.6: Comparison of real and reconstructed signal when an interval of data is missing for a sensor.

Figure D.7: Results: Experiment using real data.

# Appendix E

# PAPER E

**Title**: Online Edge Flow Imputation on Networks

**Authors**: **R. Money**, J. Krishnan, B. Beferull-Lozano, E. Isufi

**Journal**: IEEE Signal Processing Letters 2022

# Online Edge Flow Imputation on Networks

R. Money, J. Krishnan, B. Beferull-Lozano, E. Isufi

**Abstract:** **An online algorithm for missing data imputation for networks with signals defined on the edges is presented. Leveraging the prior knowledge intrinsic to real-world networks, we propose a bi-level optimization scheme that exploits the causal dependencies and the flow conservation, respectively via *(*i*)* a sparse line graph identification strategy based on a group-Lasso and *(*ii*)* a Kalman filtering-based signal reconstruction strategy developed using simplicial complex (SC) formulation. The advantages of this first SC-based attempt for time-varying signal imputation have been demonstrated through numerical experiments using EPANET models of both synthetic and real water distribution networks.**

## E.1 Introduction

Multivariate time series analysis is paramount in sensor, brain, and social networks, to name a few. Data generated from such interdependent systems can be represented as a time-varying graph, in which the recorded signals may be linked to the nodes [100, 101], or the edges [102], depending on the task at hand. Many applications including anomaly detection [103], time series forecasting [104], and missing data imputation [6] can benefit from learning and exploiting the graph structure. Among these applications, it is worth paying special attention to the missing data imputation [6, 18, 19] since many real-world systems are partially observed because of Re.g., sensor or communication failure, or simply the impossibility to have sensors in all locations. This paper focuses on time-varying data imputation on the edges of networks, such as water or traffic networks, referred to as *flow-based networks*. While there are methods for imputing data at the nodes [6, 18, 19, 105, 106], extending them to flow-based networks is not immediate.

Imputation in flow-based networks can benefit from simplicial complex (SC) formulations [45, 107], using algebraic tools from Hodge theory [108], [109] to encapsulate the adjacencies among the flow signals, e.g., the flow conservation in the network. In addition to this spatial information that SC encapsulates, one can also exploit the temporal priors, such as causal dependencies among the signals [1–4, 11, 21–23]. The flow signals are mostly interdependent in real-world systems, and their dependencies are often time-lagged in nature and cannot be observed physically. For instance, the flow in a pipe of a water network can influence the flow in another non-directly con-

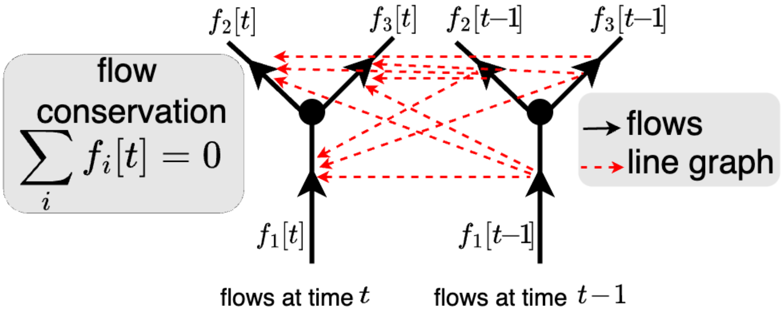


Figure E.1: Causal influence of $(t-1)$-th flows on $t$-th flows, represented using a line graph.

nected pipe in a time-lagged way. Similarly, a traffic block on a road can causally affect the traffic on another road. In such real-world networks, imputation can be enhanced by exploiting causal interactions between the flows. Imputation strategies utilizing both spatial and temporal dependencies have not been explored in flow-based networks.

This paper proposes a data imputation algorithm exploiting the spatio-temporal priors related to flow conservation and causal dependencies among flows. The algorithm learns a line graph connecting the flows, which stands in for an abstract representation of the time-lagged causal dependencies, as illustrated in Fig. E.1. One major challenge here is that a batch-based offline strategy is impractical in applications requiring real-time imputation of streaming flows. The proposed strategy learns a line graph in an online fashion. Using the learned line graph at each time step, a flow-conservation-based Kalman filter estimates the missing flows from streaming partial observations. The main contributions of this work are:

1. A method to estimate sparse causal dynamic dependencies among flows. This is achieved via a vector autoregressive model and a group-Lasso-based optimization framework. The latter is solved in an online fashion via composite objective mirror descent.

2. A Kalman-filter-based data imputation technique for streaming flows by exploiting the learned causality and the flow conservation devised via simplicial complexes.

3. The proposed algorithm can impute permanently unobserved flows, benefiting from the joint exploitation of the flow conservation and the causal dependencies.

To the best of our knowledge, this is the first work that considers multivariate time series data over simplicial complex. This work opens the door to the exploitation of learned line graphs and adjacency relationships among the time-varying signals over simplices (e.g., edge flows), which is useful in various applications such as forecasting, control strategy design, and change point detection.

## E.2 Preliminaries

Consider a physically connected network $\mathcal{G} \triangleq (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ and $\mathcal{E}$ denote the sets of nodes and edges with cardinalities $V \triangleq |\mathcal{V}|$ and $E \triangleq |\mathcal{E}|$, respectively. We consider a flow-based network, for example, a water network with nodes as junctions, edges as pipes, and water flows as signals on the edges.

### E.2.1 Modelling Flow Conservation in a Simplicial Complex

Given the set of nodes $\mathcal{V}$, a $k$-simplex $\mathcal{S}^k$ is a subset of $\mathcal{V}$ having $k+1$ distinctive elements [34], [35]. A simplicial complex (SC) of order $K$, denoted as $\Psi^K$, is a set of $k$-simplices for $k\!=\!0,1\ldots,K$ such that a simplex $\mathcal{S}^k\!\in\Psi^K$ only if all of its subsets also belong to $\Psi^K$. The typical low-order simplices, named after their geometrical shapes, are nodes (0-simplex), edges defined by two nodes (1-simplex), and triangles defined by three nodes (2-simplex). Let the number of $k$-simplices in $\Psi^K$ be $N_k$. The proximities between different $k$-simplices in an SC can be represented using an incidence matrix $\mathbf{B}_k \in \mathbb{R}^{N_{k-1}\times N_k}, k \geq 1$, where the row and the column indices of $\mathbf{B}_k$ correspond to $(k-1)$- and $k$-simplices, respectively. The structure of an SC is encoded by Hodge Laplacians, constructed using $\mathbf{B}_k$'s as

$$\mathbf{L}_k = \begin{cases} \mathbf{B}_{k+1}\mathbf{B}_{k+1}^\top, & \text{for } k=0, \\ \mathbf{B}_k^\top\mathbf{B}_k + \mathbf{B}_{k+1}\mathbf{B}_{k+1}^\top, & \text{for } 1 \leq k \leq K-1, \\ \mathbf{B}_K^\top\mathbf{B}_K, & \text{for } k=K, \end{cases} \tag{E.1}$$

where $\mathbf{L}_0$ is the graph Laplacian. The higher-order Laplacians $\mathbf{L}_k$, for $1\!\leq\! k\!\leq K\!-\!1$, consist of two terms: *i*) the *lower Laplacian*, $\mathbf{L}_k^l\!\triangleq\!\mathbf{B}_k^\top\mathbf{B}_k$, which encodes the adjacencies w.r.t. next-low-order simplices; and *ii*) the *upper Laplacian*, $\mathbf{L}_k^u \triangleq \mathbf{B}_{k+1}\mathbf{B}_{k+1}^\top$, which encodes the adjacencies w.r.t. next-high-order simplices.

In a SC, $k$-simplex signals are mappings from $k$-simplices to the real set $\mathbb{R}$. The 0-simplex, 1-simplex, and 2-simplex signals reside on the nodes, edges, and triangles, respectively. For flow-based networks, we consider 1-simplex signals or simply the flow signals. The flow signal at time $t$ between two nodes $i$ and $j$ is defined as $f_{(i,j)}[t] = -f_{(j,i)}[t],\ \forall\ (i,j) \in \mathcal{E}$. We stack the flows into a vector $\tilde{\mathbf{f}}[t] = [f_1[t]\ f_2[t]\ \ldots\ f_E[t]]^\top$. The node-to-edge incidence matrix $\mathbf{B}_1 \in \mathbb{R}^{V\times E}$ has entries $\mathbf{B}_1(m,n) = 1$, if the flow $n$ is leaving the node $m$, $-1$ if entering the node $m$, and 0 if the flow is not connected to $m$. According to the flow conservation principle, the sum of flows entering and leaving a node is zero, i.e., $\mathbf{B}_1\tilde{\mathbf{f}}[t] = \mathbf{0} \in \mathbb{R}^V$ [20]. The first-order lower Laplacian $\mathbf{L}_1^l$, can be used to model the flow conservation since it describes the relationship among the edges incidenting on a node, which is given by

$$\|\mathbf{B}_1\tilde{\mathbf{f}}[t]\|_2^2 = \tilde{\mathbf{f}}[t]^\top\mathbf{B}_1^\top\mathbf{B}_1\tilde{\mathbf{f}}[t] = \tilde{\mathbf{f}}[t]^\top\mathbf{L}_1^l\tilde{\mathbf{f}}[t] = 0. \tag{E.2}$$

One can also exploit the edge-to-triangle relationship of flows using $\mathbf{B}_2$, but we do not consider it since there is no contextual prior associated with $\mathbf{B}_2$.

### E.2.2 Modelling Causal Dependencies using Line Graphs

We also take advantage from the fact that flows in a real-world network exhibit causal interactions. We construct a dynamic line graph connecting the flows using a $P$-th order dynamic VAR model to describe the time-lagged causal dependencies among the flows:

$$\tilde{\mathbf{f}}[t] = \sum_{p=1}^{P} \Big[ \tilde{\mathbf{A}}^{(p)}[t]\tilde{\mathbf{f}}[t-p] + \mathbf{b}^{(p)}[t] \Big] + \mathbf{u}[t], \tag{E.3}$$

where $\tilde{\mathbf{A}}^{(p)}[t] \in \mathbb{R}^{E \times E}$ is the unknown weighted adjacency matrix of the line graph that captures the influence of the $p$-th time-lagged vector flow on the vector flow at time $t$, and $\mathbf{u}[t]$ is the process noise, Rwhich is assumed to be temporarily white and zero mean. The term $\mathbf{b}^{(p)}[t] \in \mathbb{R}^{E}$ is the bias component, which makes the model slightly different from a standard VAR model. We include the bias term since the normalization of the flow signals, which is a requirement for the subsequent formulation, cannot easily be achieved for permanently unobserved flows. Using an augumented matrix $\mathbf{A}^{(p)}[t] = [\tilde{\mathbf{A}}^{(p)}[t] \; \mathbf{b}^{(p)}[t]] \in \mathbb{R}^{E \times E+1}$ and the signal vector $\mathbf{f}[t] = [\tilde{\mathbf{f}}[t]^\top; 1]^\top \in \mathbb{R}^{E+1}$, (E.3) can be compactly written as

$$\mathbf{f}[t] = \sum_{p=1}^{P} \mathbf{A}^{(p)}[t]\mathbf{f}[t-p] + \mathbf{u}[t]. \tag{E.4}$$

# E.3 Problem formulation

Assume that at a particular time $t$, only a subset of flows is observable. The observed flow vector is $\mathbf{f}_o[t] = \mathbf{M}[t]\mathbf{f}[t] \in \mathbb{R}^{E+1}$, where $\mathbf{M}[t] \in \mathbb{R}^{(E+1)\times(E+1)}$ is a diagonal masking matrix, with $\mathbf{M}(n,n)[t] = 0$ if the *n-th* flow is missing and $\mathbf{M}(n,n)[t] = 1$, otherwise. In this setting, some flows can be permanently unobserved. The goal is to find in an online fashion both a sequence of line graphs $\{\mathbf{A}^{(p)}[t]\}_{p,t}$, representing the causal dependencies between flows and the original signal $\mathbf{f}[t]$ from the partial observation $\mathbf{f}_o[t]$.

# E.4 Online estimation of the line graph and data

A naive one-step optimization strategy to estimate $\mathbf{A}^{(p)}[t]$ and $\mathbf{f}[t]$ leads to nonconvex formulations that are difficult to solve [6]. Hence, we propose a bi-level optimization problem with the following steps: *i*) *signal reconstruction-* missing flows are estimated using the observed flows by assuming a known line graph topology; and *ii*) *line graph identification-* line graph is estimated using the reconstructed signals.

## E.4.1 Signal Reconstruction

Assume that we have an estimate at time $t$ of the topology $\hat{\mathbf{A}}^{(p)}[t]$, $\forall p$ and estimates of $P$ previous flow values $\{\hat{\mathbf{f}}[t-p]\}_{p=1}^{P}$. We propose a Kalman-filtering-based strategy for signal reconstruction, and to facilitate the formulation, the available data are arranged as

$$\hat{\mathbf{A}}^{\mathcal{S}}[t] \triangleq \left[\begin{array}{c|c} \multicolumn{2}{c}{\underbrace{\hat{\mathbf{A}}^{(1:P)}[t]}_{E \times P(E+1)}} \\ \mathbf{I}_{P(E+1)-E} & \underbrace{\mathbf{0}}_{(P(E+1)-E)\times E} \end{array}\right], \mathbf{C}^{\mathcal{S}}[t] \triangleq \left[\begin{array}{c|c} \underbrace{\mathbf{M}[t]}_{(E+1)\times(E+1)} & \underbrace{\mathbf{0}}_{(E+1)\times(P-1)(E+1)} \\ \underbrace{\mathbf{0}}_{(P-1)(E+1)\times(E+1)} & \mathbf{I}_{(P-1)(E+1)} \end{array}\right],$$

$$\begin{aligned} \mathbf{y}^{\mathcal{S}}[t] &\triangleq [\mathbf{f}_o[t]^\top; \hat{\mathbf{f}}[t-1:t-P+1]^\top]^\top, \\ \hat{\mathbf{f}}^{\mathcal{S}}[t] &\triangleq [\hat{\mathbf{f}}[t]^\top; \hat{\mathbf{f}}[t-1]^\top; \ldots; \hat{\mathbf{f}}[t-P+1]^\top]^\top, \end{aligned} \tag{E.5}$$

where $\hat{\mathbf{A}}^{(1:P)}[t] = [\hat{\mathbf{A}}^{(1)}[t], \ldots, \hat{\mathbf{A}}^{(P)}[t]]$ and $\mathbf{I}_N$ denotes $N \times N$ identity matrix. A state-space representation capturing the VAR relationships (E.15) and the missing data modelling is

$$\hat{\mathbf{f}}^{\mathcal{S}}[t] = \hat{\mathbf{A}}^{\mathcal{S}}[t]\hat{\mathbf{f}}^{\mathcal{S}}[t-1] + \mathbf{v}_t, \tag{E.6}$$

$$\mathbf{y}^{\mathcal{S}}[t] = \mathbf{C}^{\mathcal{S}}[t]\hat{\mathbf{f}}^{\mathcal{S}}[t] + \mathbf{w}_t, \tag{E.7}$$

where $\hat{\mathbf{f}}^{\mathcal{S}}[t] \in \mathbb{R}^{P(E+1)}$ is current state vector, $\hat{\mathbf{A}}^{\mathcal{S}}[t] \in \mathbb{R}^{P(E+1)\times P(E+1)}$ is the state transition matrix and $\mathbf{y}^{\mathcal{S}}[t] \in \mathbb{R}^{P(E+1)}$, and $\mathbf{C}^{\mathcal{S}} \in \mathbb{R}^{P(E+1)\times P(E+1)}$ are the observed signal and the observation matrix, respectively. The process noise $\mathbf{v}_t$ and the observation noise $\mathbf{w}_t$ are assumed zero-mean Gaussian. The optimal estimates of $\hat{\mathbf{f}}^{\mathcal{S}}[t]$ can be obtained using a Kalman filter (KF) [33].

**1) Prediction:**

$$\hat{\mathbf{f}}^{\mathcal{S}}_{t|t-1} = \hat{\mathbf{A}}^{\mathcal{S}}[t]\hat{\mathbf{f}}^{\mathcal{S}}_{t-1|t-1}, \tag{E.8}$$

$$\mathbf{P}_{t|t-1} = \hat{\mathbf{A}}^{\mathcal{S}}[t]\mathbf{P}_{t-1|t-1}\hat{\mathbf{A}}^{\mathcal{S}}[t]^\top + \mathbf{Q}_t, \tag{E.9}$$

where $t|t-1$ refers to the estimate at time $t$ given the observation up to $t-1$, $\mathbf{P}_{t|t-1} \in \mathbb{R}^{(E+1)P\times(E+1)P}$ is the prediction error covariance matrix and $\mathbf{Q}_t \in \mathbb{R}^{(E+1)P\times(E+1)P}$, the noise covariance matrix.

**2) Update:** The KF update of the state vector can be expressed as convex optimization problem [46], [47]:

$$\begin{aligned} \underset{\hat{\mathbf{f}}^{\mathcal{S}}_{t|t}, \mathbf{w}_t}{\text{minimize}} \quad & \mathbf{w}_t^\top \mathbf{R}_t{}^{-1}\mathbf{w}_t + (\hat{\mathbf{f}}^{\mathcal{S}}_{t|t} - \hat{\mathbf{f}}^{\mathcal{S}}_{t|t-1})^\top \mathbf{P}^{-1}_{t|t-1}(\hat{\mathbf{f}}^{\mathcal{S}}_{t|t} - \hat{\mathbf{f}}^{\mathcal{S}}_{t|t-1}), \\ \text{subject to} \quad & \mathbf{y}^{\mathcal{S}}[t] = \mathbf{C}^{\mathcal{S}}[t]\hat{\mathbf{f}}^{\mathcal{S}}_{t|t} + \mathbf{w}_t. \end{aligned} \tag{E.10}$$

Solving (E.10) yields the standard KF update equation:

$$\hat{\mathbf{f}}^{\mathcal{S}}_{t|t} = \hat{\mathbf{f}}^{\mathcal{S}}_{t|t-1} + \mathbf{K}_t(\mathbf{y}^{\mathcal{S}}[t] - \mathbf{C}^{\mathcal{S}}[t]\hat{\mathbf{f}}^{\mathcal{S}}_{t|t-1}). \tag{E.11}$$

The covariance matrix can be updated as

$$\mathbf{P}_{t|t} = \mathbf{P}_{t|t-1} - \mathbf{K}_t\mathbf{C}^{\mathcal{S}}[t]\mathbf{P}_{t|t-1}. \tag{E.12}$$

where $\mathbf{K}_t = \mathbf{P}_{t|t-1}\mathbf{C}^{\mathcal{S}}[t]^\top(\mathbf{C}^{\mathcal{S}}[t]\mathbf{P}_{t|t-1}\mathbf{C}^{\mathcal{S}}[t]^\top + \mathbf{R}_t)^{-1}$ is the Kalman gain and $\mathbf{R}_t$ is the covariance matrix of the observation noise.

**3) Flow-conservation update:** The KF update problem (E.10), penalized with the flow conservation (E.2), can be written as

$$\begin{aligned}
\underset{\hat{\mathbf{f}}^{\mathcal{S}}_{t|t},\mathbf{w}_t}{\text{minimize}} \quad & \mathbf{w}_t^\top \mathbf{R}_t{}^{-1}\mathbf{w}_t + (\hat{\mathbf{f}}^{\mathcal{S}}_{t|t} - \hat{\mathbf{f}}^{\mathcal{S}}_{t|t-1})^\top \mathbf{P}^{-1}_{t|t-1}(\hat{\mathbf{f}}^{\mathcal{S}}_{t|t} - \hat{\mathbf{f}}^{\mathcal{S}}_{t|t-1}) \\
& \qquad\qquad\qquad\qquad + \mu \hat{\mathbf{f}}^{\mathcal{S}}_{t|t}[t]^\top \mathbf{L}\hat{\mathbf{f}}^{\mathcal{S}}_{t|t}[t], \\
\text{subject to} \quad & \mathbf{y}^{\mathcal{S}}[t] = \mathbf{C}^{\mathcal{S}}[t]\hat{\mathbf{f}}^{\mathcal{S}}_{t|t} + \mathbf{w}_t,
\end{aligned} \tag{E.13}$$

where

$$\mathbf{L} = \begin{bmatrix} \tilde{\mathbf{L}}^l_1 & \mathbf{0}_{(E+1)\times(P-1)(E+1)} \\ \mathbf{0}_{(P-1)(E+1)\times(E+1)} & \mathbf{0}_{(P-1)(E+1)\times(P-1)(E+1)} \end{bmatrix},$$

with $\tilde{\mathbf{L}}^l_1 = [\mathbf{L}^l_1 \ \mathbf{0}_E; \mathbf{0}_E^\top \ 0] \in \mathbb{R}^{(E+1)\times(E+1)}$, the Laplacian $\mathbf{L}^l_1$ padded with zero vector $\mathbf{0}_E \in \mathbb{R}^E$ to nullify the bias component in $\mathbf{f}[t]$ and $\mu$ is a hyperparameter. We regularize flow conservation instead of imposing it as a constraint, based on the assumption that the flow conservation is not strictly satisfied in real-world networks. The optimization problem (E.13) is quadratic with a closed-form solution (see, E.7.1):

$$\begin{aligned}
\hat{\mathbf{f}}^{\mathcal{S}}_{t|t} =& (\mathbf{C}^{\mathcal{S}}[t]^\top \mathbf{R}_t^{-1}\mathbf{C}^{\mathcal{S}}[t] + \mathbf{P}^{-1}_{t|t-1} + 2\mu\mathbf{L})^{-1} \times \\
& (\mathbf{C}^{\mathcal{S}}[t]^\top \mathbf{R}_t{}^{-1}\mathbf{y}^{\mathcal{S}}[t] + \mathbf{P}^{-1}_{t|t-1}\hat{\mathbf{f}}^{\mathcal{S}}_{t|t-1}),
\end{aligned} \tag{E.14}$$

### E.4.2 Line Graph Identification

The element-wise version of (E.15) for the $n^{th}$ flow is

$$f_n[t] = \sum_{n'=1}^{E+1}\sum_{p=1}^{P} a^{(p)}_{n,n'}[t] f_{n'}[t-p] + u_n[t], \tag{E.15}$$

where $a^{(p)}_{n,n'}[t] \in \mathbb{R}$ represents the influence of the $p$-th time-lagged value of flow $n'$ on flow $n$. For notational convenience, we stack the elements of $a^{(p)}_{n,n'}[t]$ in the lexicographic order of the indices $p$, and $n'$ to obtain $\boldsymbol{a}_n[t] \in \mathbb{R}^{(E+1)P}$ and also stack the same elements along index $p$ to obtain $\boldsymbol{a}_{n,n'}[t] \in \mathbb{R}^P$. Assuming flows are known, the online topology identification can be formulated as [23, 48]

$$\widehat{\boldsymbol{a}}_n[t] = \arg \min_{\boldsymbol{a}_n \in \mathbb{R}^{(E+1)P}} \ell^n_t(\boldsymbol{a}_n) + \lambda \sum_{n'=1}^{E+1} \|\boldsymbol{a}_{n,n'}\|_2, \tag{E.16}$$

where $\ell^n_t(\boldsymbol{a}_n) = \frac{1}{2}[f_n[t] - \boldsymbol{a}_n^\top \hat{\mathbf{f}}^{\mathcal{S}}[t-1]]^2$ is the instantaneous loss function for a node $n$ and $\lambda$ is a hyperparameter. The second term is a group-lasso regularizer added in line with the assumption that the real-world dependencies are sparse.

In general, proximal algorithms can solve objective functions of the form (E.16) having a differentiable loss function and a non-differentiable regularizer. Following

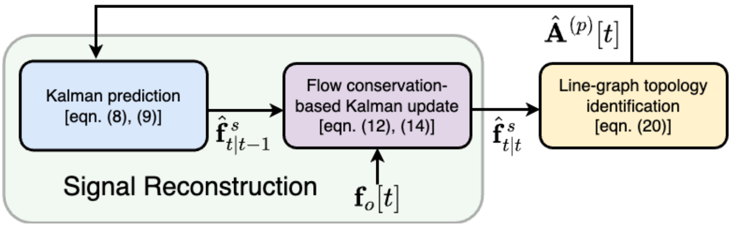

$\hat{\mathbf{A}}^{(p)}[t]$
Kalman prediction
[eqn. (8), (9)]
$\hat{\mathbf{f}}^s_{t|t-1}$
Flow conservation-based Kalman update
[eqn. (12), (14)]
$\hat{\mathbf{f}}^s_{t|t}$
Line-graph topology identification
[eqn. (20)]
Signal Reconstruction
$\mathbf{f}_o[t]$

Figure E.2: Schematic representation of the proposed algorithm.

[48], we use online composite objective mirror descent (COMID), which is effective and comes with convergence guarantees. The online COMID update is

$$\widehat{\boldsymbol{a}}_n[t+1] = \arg \min_{\boldsymbol{a}_n \in \mathbb{R}^{(E+1)P}} J_t^{(n)}(\boldsymbol{a}_n), \tag{E.17}$$

$$\begin{aligned} \text{where } J_t^{(n)}(\boldsymbol{a}_n) &\triangleq \nabla \ell_t^n(\widehat{\boldsymbol{a}}_n[t])^\top (\boldsymbol{a}_n - \widehat{\boldsymbol{a}}_n[t]) \\ &+ \frac{1}{2\gamma_t}\|\boldsymbol{a}_n - \widehat{\boldsymbol{a}}_n[t]\|_2^2 + \lambda \sum_{n'=1}^{E+1} \|\boldsymbol{a}_{n,n'}\|_2. \end{aligned} \tag{E.18}$$

Equation (E.18) has the gradient of the loss $\ell_t^n(\boldsymbol{a}_n)$ as the first term, and the Bregman divergence and sparsity-promoting regularizer as the second and the third terms, respectively. Bregman divergence makes the algorithm more stable by constraining $\widehat{\boldsymbol{a}}_n[t+1]$ to be close to $\widehat{\boldsymbol{a}}_n[t]$ and it is chosen to be $B(\boldsymbol{a}_n, \widehat{\boldsymbol{a}}_n[t]) = \frac{1}{2}\|\boldsymbol{a}_n - \widehat{\boldsymbol{a}}_n[t]\|_2^2$ so that the COMID update has a closed-form solution [40] and $\gamma_t > 0$ is the corresponding step size. The gradient in (E.18) is evaluated as

$$\mathbf{v}_n[t] \triangleq \nabla \ell_t^n(\widehat{\boldsymbol{a}}_n[t]) = \hat{\mathbf{f}}^{\mathcal{S}}[t-1]\left(\boldsymbol{a}_n^\top \hat{\mathbf{f}}^{\mathcal{S}}[t-1] - f_n[t]\right) \tag{E.19}$$

The optimization problem is separable across nodes and a closed-form solution for (E.17) is obtained via the multidimensional shrinkage-thresholding operator [41]:

$$\widehat{\boldsymbol{a}}_{n,n'}[t+1] = \left(\widehat{\boldsymbol{a}}_{n,n'}[t] - \gamma_t \mathbf{v}_{n,n'}[t]\right)\left[1 - \frac{\gamma_t \lambda}{\|\widehat{\boldsymbol{a}}_{n,n'}[t] - \gamma_t \mathbf{v}_{n,n'}[t]\|_2}\right]_+, \tag{E.20}$$

where $[x]_+ = \max\{0, x\}$. A schematic representation of the proposed algorithm is shown in Fig. F.1. The computational complexity of the algorithm is mainly contributed by (E.14), and it is of order $\mathcal{O}\big(P^3(E+1)^3\big)$.

## E.5 Experimental Results

We use flow data from a real water network and a synthetic network, both generated using the EPANET software. The flow signals are the hourly sampled volume of water in $m^3/h$. A demand-driven model is used to generate data such that the water flows meet the time-varying water demands at the nodes. We compare the results with the state-of-the-art algorithms Graph-based Semi-supervised learning for Edge Flows *(FlowSSL)* [20] and *Joint Signal and Topology Identification via Recursive Sparse Online learning (JSTIRSO)* [6]. FlowSSL exploits the flow conservation of the flows, whereas JSTIRSO uses a causal graph structure to impute the missing data. We compare the algorithms via the normalized mean squared error (NMSE):

$$\text{NMSE}_n(T) = \frac{\sum_{t=1}^{T}(f_n(t) - \hat{f}_n(t))^2}{\sum_{t=1}^{T} f_n(t)^2}. \tag{E.21}$$

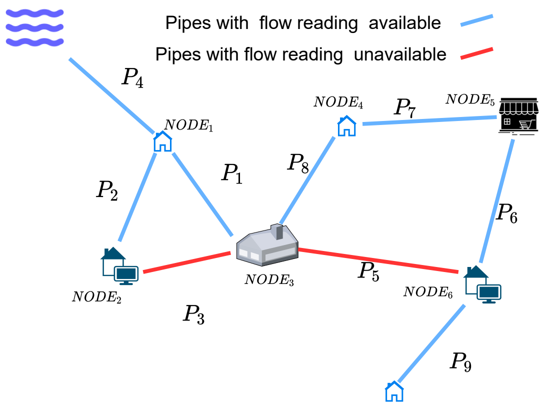


Figure E.3: Physical graph.

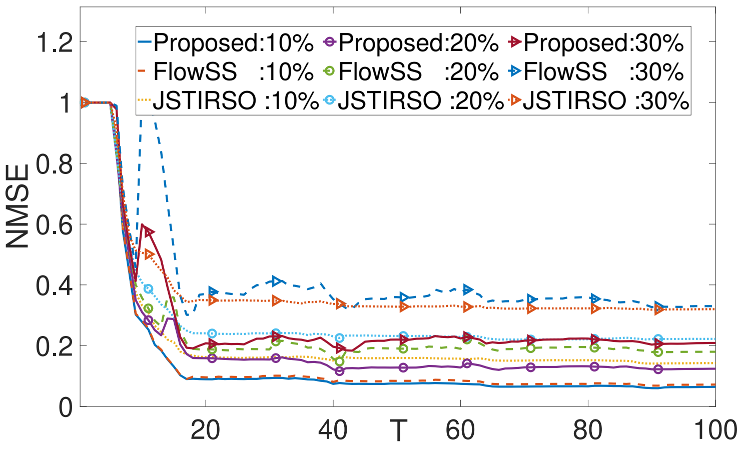


Figure E.4: Time varying random missing-flow patterns.

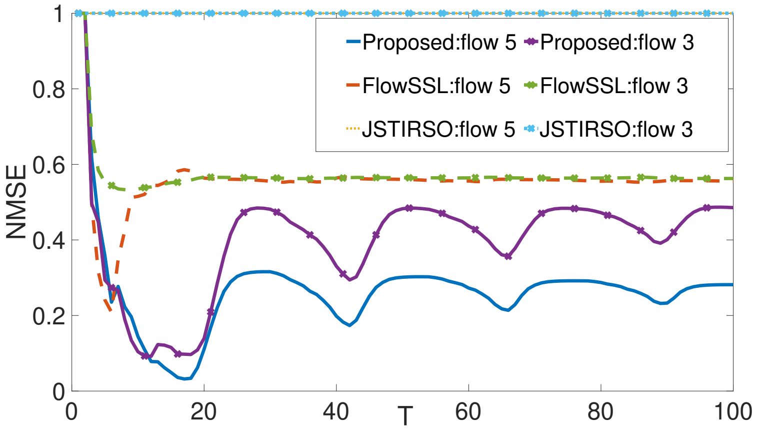


Figure E.5: Permanently unobserved flows.

Figure E.6: Synthetic Water Network Topology.

A total of 125 data samples are generated, and the initial 25 samples are used to tune the hyperparameters of all the algorithms to achieve the lowest NMSE averaged across all edges via grid search. The line graph is initialized with random values drawn from $\mathcal{N}(0, 1)$. The NMSEs are averaged over 50 runs of experiments.

## E.5.1 Synthetic Water Network

A water distribution model, shown in E.3, is simulated, which consists of 1 reservoir, 9 pipes, and 8 nodes. Below, we examine two types of missing data patterns with the hyperparameter setting $(\mu, \lambda) = (0.5, 0.1)$.

### E.5.1.1 Random variation in missing-flows

We assume that $10\%, 20\%$, and $30\%$ of randomly chosen flows are missing at each time instant. NMSEs are plotted in Appendix E.4.2, which shows that the proposed method is better than the competitors because, unlike them, it takes full advantage of the flow conservation and causal dependencies. Going beyond 30% of missing data results in very high NMSEs by all algorithms, and is not included in Appendix E.4.2 to maintain the legibility.

### E.5.1.2 Permanently unobserved flows

We consider flow-3 and flow-5 are permanently missing. The NMSEs for both the missing flows are shown in E.5. The proposed method provides better imputation performance compared to FlowSSL [20], whereas JSTIRSO [6] fails to reconstructs the missing signal since it does not exploit the flow conservation.

## E.5.2 Cherry Hills Water Networks

Cherry Hills is a real water network consisting of 40 pipes and 36 nodes [110]. We assume a reference flow direction as in Fig. E.7, and the hyperparamters are tuned to $(\mu, \lambda) = (50, 0.04)$. We examine four different scenarios in which $20\%, 30\%, 40\%$, and $50\%$ of the flows are randomly missing at each time stamp. The average NMSEs computed from the estimates of random missing flows are plotted in Fig. E.11, where the proposed method outperforms the other two algorithms, especially with a significant margin for the 50% missing case. NMSEs of all algorithms is very high when more than 50% of flows are missing. The experiment is repeated with $15\%, 20\%$, and $25\%$ of permanently missing flows, and the results are plotted in Fig. E.10, where the proposed algorithm outperforms the competitors in all the cases.

One instance of the learned line graph ($T$=100, $p$=3) is shown in Fig. E.8. We wish to note that the line graph is an abstract graph induced by the various physics-based equations describing the space-temporal variation of the flows. Although one could attempt to analyse the line graph using the underlying differential equations governing the space-time system, this is a daunting complex process, which is beyond the scope of this study. However, a good prediction implies necessarily that the data-driven line graph is close to the unknown real graph. To demonstrate the importance of the learned line graph, we repeat the Kalman prediction using a random line graph without considering any relation to the data. NMSEs obtained for permanently missing flows at $t$=100, using random and learned line graphs, are 1.08 and 0.06, respectively. Similar results were obtained for all the other experiments highlighting the role of the learned line graph.

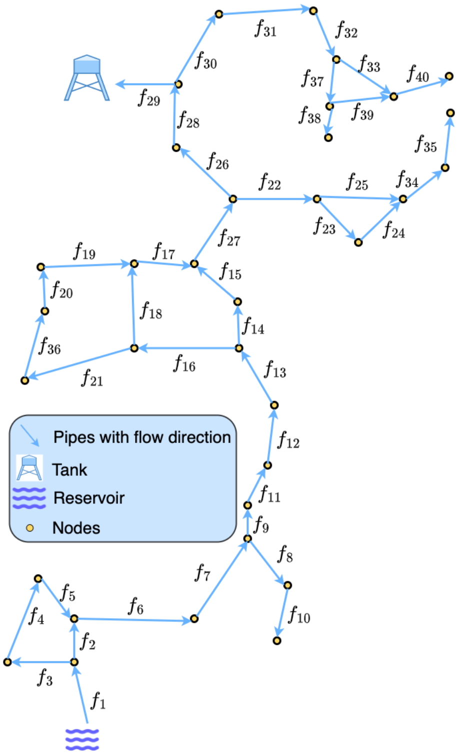


Figure E.7: Cherry Hills Flows.

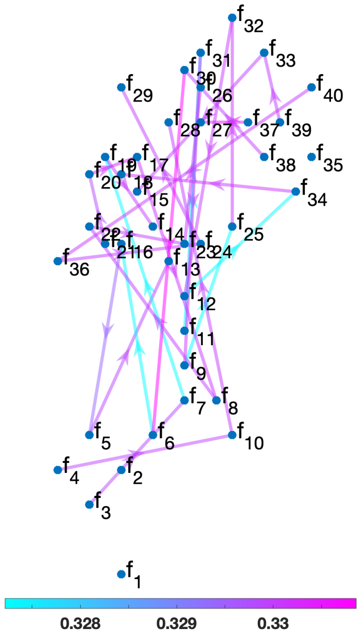


Figure E.8: Estimated Line Graph.
Figure E.9: Cherry Hills Water Network.

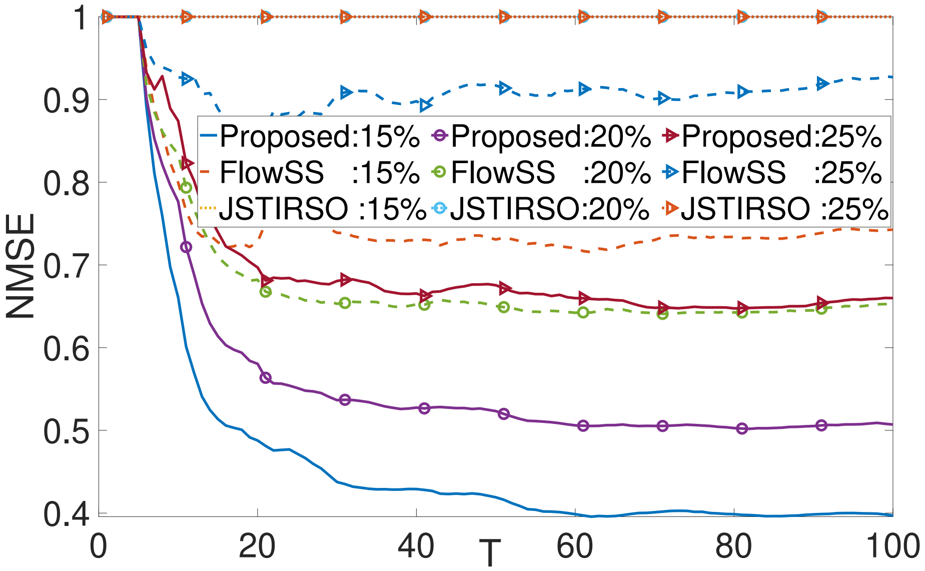


Figure E.10: Permanently Missing Flows.

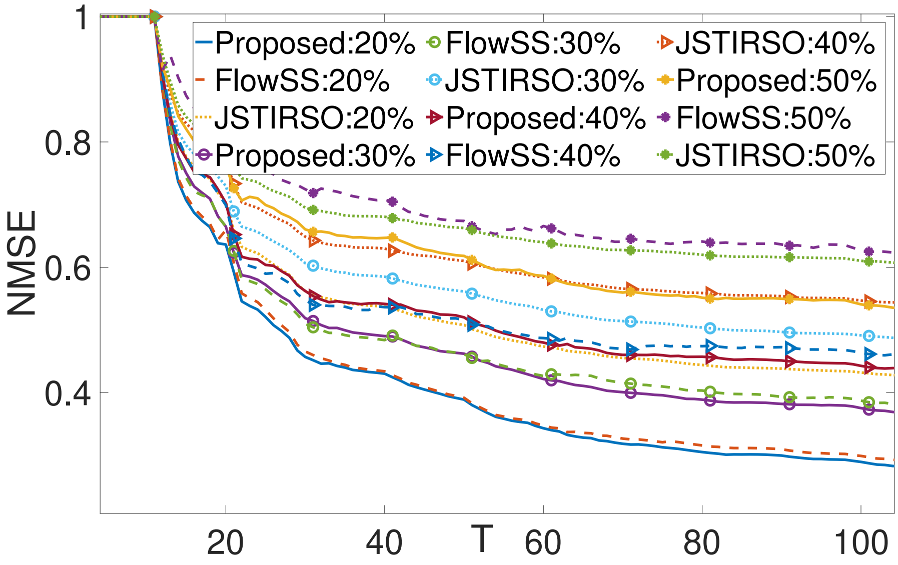


Figure E.11: Randomly Missing Flows.

Figure E.12: Cherry Hills Water Network:NMSE.

## E.6 Conclusion

We proposed a novel missing data imputation scheme for flow-based networks. The proposed algorithm comprises a simplicial-complex-based Kalman filter and a group-lasso-based optimization strategy to take advantage of the flow conservation and causal dependency of real-world networks. This study paves the way for exploring higher order connectivity in real-life networks using simplicial complexes.

## E.7 Supplementary Material

### E.7.1 Derivation of Flow-Conservation-based Kalman Filter

The optimization problem (E.13) is a convex quadratic optimization problem that yields flow-conservation-based Kalman updates. We adopt a similar strategy as followed in [47] to obtain a closed-form solution. We first reformulate the problem (E.13) by substituting the constraint $\mathbf{w}_t = \mathbf{y}^{\mathcal{S}}[t] - \mathbf{C}^{\mathcal{S}}[t]\hat{\mathbf{f}}^{\mathcal{S}}_{t|t}$ in the objective function:

$$\begin{aligned}\underset{\hat{\mathbf{f}}^{\mathcal{S}}_{t|t}}{\text{minimize}} \quad & (\mathbf{y}^{\mathcal{S}}[t] - \mathbf{C}^{\mathcal{S}}[t]\hat{\mathbf{f}}^{\mathcal{S}}_{t|t})^{\top}\mathbf{R}_t{}^{-1}(\mathbf{y}^{\mathcal{S}}[t] - \mathbf{C}^{\mathcal{S}}[t]\hat{\mathbf{f}}^{\mathcal{S}}_{t|t}) \\ & +(\hat{\mathbf{f}}^{\mathcal{S}}_{t|t} - \hat{\mathbf{f}}^{\mathcal{S}}_{t|t-1})\mathbf{P}^{-1}_{t|t-1}(\hat{\mathbf{f}}^{\mathcal{S}}_{t|t} - \hat{\mathbf{f}}^{\mathcal{S}}_{t|t-1})^{\top} + \mu(\hat{\mathbf{f}}^{\mathcal{S}}_{t|t})^{\top}\mathbf{L}\hat{\mathbf{f}}^{\mathcal{S}}_{t|t},\end{aligned} \tag{E.22}$$

where

$$\underbrace{\mathbf{L}}_{P(E+1)\times P(E+1)} \triangleq \left[\begin{array}{c|c} \underbrace{\tilde{\mathbf{L}}^{l}_{1}}_{(E+1)\times(E+1)} & \underbrace{\mathbf{0}}_{(E+1)\times(P-1)(E+1)} \\ \hline \underbrace{\mathbf{0}}_{(P-1)(E+1)\times(E+1)} & \underbrace{\mathbf{0}}_{(P-1)(E+1)\times(P-1)(E+1)} \end{array}\right].$$

Next, we differentiate the objective function with respect to $\hat{\mathbf{f}}^{\mathcal{S}}_{t|t}$ and equate to 0 to find the optimum $\hat{\mathbf{f}}^{\mathcal{S}}_{t|t}$:

$$\begin{aligned}-2\mathbf{C}^{\mathcal{S}}[t]^{\top}\mathbf{R}_t^{-1}(\mathbf{y}^{\mathcal{S}}[t] - \mathbf{C}^{\mathcal{S}}[t]\hat{\mathbf{f}}^{\mathcal{S}}_{t|t}) \\ + 2\mathbf{P}^{-1}_{t|t-1}(\hat{\mathbf{f}}^{\mathcal{S}}_{t|t} - \hat{\mathbf{f}}^{\mathcal{S}}_{t|t-1}) + 2\mu\mathbf{L}\mathbf{f}[t] = 0\end{aligned} \tag{E.23}$$

$$\begin{aligned}\implies \hat{\mathbf{f}}^{\mathcal{S}}_{t|t} =& (\mathbf{C}^{\mathcal{S}}[t]^{\top}\mathbf{R}^{-1}\mathbf{C}^{\mathcal{S}}[t] + \mathbf{P}^{-1}_{t|t-1} + 2\mu\mathbf{L})^{-1}\times \\ & (\mathbf{C}^{\mathcal{S}^{\top}}\mathbf{R}^{-1}\mathbf{Y}^{\mathcal{S}}[t] + \mathbf{P}^{-1}_{t|t-1}\hat{\mathbf{f}}^{\mathcal{S}}_{t|t-1}),\end{aligned} \tag{E.24}$$

which is the required flow-conservation-based Kalman filter solution.

# Appendix F

# PAPER F

**Title**: Scalable and Privacy-aware Online Learning of Nonlinear Structural Equation Models

**Authors**: **R. Money**, J. Krishnan, B. Beferull-Lozano, E. Isufi

**Journal**: IEEE Open Journal of Signal Processing 2023

# Scalable and Privacy-aware Online Learning of Nonlinear Structural Equation Models

R. Money, J. Krishnan, B. Beferull-Lozano, E. Isufi

**Abstract: An online topology estimation algorithm for nonlinear structural equation models (SEM) is proposed in this paper, addressing the nonlinearity and the non-stationarity of real-world systems. The nonlinearity is modeled using kernel formulations, and the curse of dimensionality associated with the kernels is mitigated using random feature approximation. The online learning strategy uses a group-lasso-based optimization framework with a prediction-corrections technique that accounts for the model evolution. The proposed approach has three properties of interest. First, it enjoys node-separable learning, which allows for scalability in large networks. Second, it offers privacy in SEM learning by replacing the actual data with node-specific random features. Third, its performance can be characterized theoretically via a dynamic regret analysis, showing that it is possible to obtain a linear dynamic regret bound under mild assumptions. Numerical results with synthetic and real data corroborate our findings and show competitive performance w.r.t. state-of-the-art alternatives.**

## F.1 INTRODUCTION

Structural Equation Models (SEM) are a prevalent tool to model interactions in real-world networks due to their simplicity and ability to express instantaneous directed relationships between interacting entities [111–113]. The advantages of SEM over simple correlation-based models lie in leveraging the directionality, which is key to many applications, such as modeling the functional connectivity between brain regions [114] and interactions in financial networks [115], to name a few. SEM modeling and its topology estimation are challenging because real-life networks are large, dynamic, and comprise nonlinear interactions, as well as leveraging directly node-specific data may raise privacy concerns [111].

Although SEM-based topology estimation has been explored in literature, most of the approaches are developed for stationary linear systems and provide offline (batch-based) solutions [116, 117]. Modeling time-varying systems call for online optimization strategies, which can be classified into *time-unstructured* and *time-*

*structured* methods [50, 118]. The former update the model only when a new data sample arrives [68], whereas the latter first predict the model based on its evolution and then correct the prediction when the new data sample is available [119]. The time-structured algorithms are expected to perform better since they take advantage of the prior related to the model evolution but typically have a slightly higher computational cost. A SEM-based online topology estimation has been proposed in [8], but it adopts the time-unstructured strategy and fails to exploit the model evolution; hence, suboptimal. On the other hand, [118] and [1] propose time-structured online SEM learning strategies, but the models are restricted to linear interactions. Moreover, the node operations of these algorithms are computationally expensive, and they assume symmetric interactions of the network data, which destroys SEM's directionality features.

Aiming to overcome the above challenges, we propose an online topology learning algorithm for nonlinear and directed SEM using a time-structured optimization framework. The nonlinearity is tackled using kernel methods, and the curse of dimensionality of kernels is mitigated through random feature (RF) approximation. Kernel techniques are conventionally used for nonlinear topology estimation [11,21,27] and help transform the problem into an amenable form. Instead, RF is typically used to reduce the complexity of nonlinear models as well as to ensure that connectivity is inferred without revealing nodal attributes [22,36,42,43,76,120]. Through a series of design choices and theoretical derivations, we show how kernels and RFs can be incorporated into the online nonlinear SEM model and show that the proposed algorithm has the following four properties:

1. Sparse model evolution: The proposed SEM learning strategy uses a prediction-correction approach to model the SEM evolution. Exploiting the fact that real-world networks exhibit sparse directed interactions, we introduce a group-lasso-based regularizer to learn sparse models.

2. Scalability: The proposed algorithm is separable across the nodes with a fixed computational complexity per iteration, thereby facilitating scalability in large graphs.

3. Privacy: The node separability and the random features avoid sharing the true data, thus, ensuring node privacy.

4. Convergence Guarantee: A dynamic regret analysis of the proposed algorithm is conducted, guaranteeing convergence, and showing the role played by the different components of the proposed method.

Numerical experiments on synthetic data and real data from neuroscience and finance corroborate the above contributions and show superior performance to competing alternatives.

The rest of the paper is organized as follows. Section F.2 presents the nonlinear SEM, kernel formulation, and random feature approximation. Appendix F.3 develops an online strategy for learning the nonlinear SEM using a prediction-correction algorithm. The dynamic regret analysis of the proposed algorithm is performed in Appendix F.4, and the numerical results are provided in F.5. F.6 concludes the paper. All proofs are collected in the Appendix.

## F.2 Problem formulation

Consider $N$ interdependent time series, and let $y_n[t]$ be the value of the $n$-th time series at time $t$. A nonlinear SEM with no exogenous variables models the dependencies among these time series as

$$y_n[t] = \sum_{n'=1, n'\neq n}^{N} f_{n,n'}(y_{n'}[t]) + u_n[t], \ \ n = 1, \dots, N, \tag{F.1}$$

where $f_{n,n'}(\cdot)$ encodes the nonlinear influence of $n'$-th time series on $n$-th time series, and $u_n[t]$ is the observation noise [14]. For example, in the context of brain networks, $y_n[t]$ represents the electroencephalogram (EEG) recorded at the $n$-th node (sensor) at time $t$, and $f_{n,n'}(\cdot)$ encodes the functional connectivity between the nodes $n$ and $n'$.

**Kernel representation.** The nonlinear structure in (F.1) allows modeling a broader range of problems, but at the same time makes it more difficult to analyse and model the time series interactions. A typical way to approach these challenges is to consider the nonlinear function in (F.1) belonging to a reproducing kernel Hilbert space (RKHS):

$$\mathcal{H}_{n'} := \left\{ f_{n,n'}(\cdot) \mid f_{n,n'}\left(y_n[t']\right) = \sum_{t=0}^{\infty} \beta_{n,n',t}\ \kappa_{n'}\left(y_n[t'], y_{n'}[t]\right) \right\}, \tag{F.2}$$

where $\kappa_{n'}(\cdot,\cdot)$ is a positive definite kernel function, measuring the similarity between its arguments. Every positive definite kernel has an associated RKHS characterized by the inner product: $\langle \kappa_{n'}(y,x_1), \kappa_{n'}(y,x_2)\rangle := \sum_{t=0}^{\infty} \kappa_{n'}(y[t],x_1)\kappa_{n'}(y[t],x_2)$. RKHS kernels satisfy the reproducing property $\langle \kappa_{n'}^{(p)}(y,x_1), \kappa_{n'}(y,x_2)\rangle = \kappa_{n'}(x_1,x_2)$, and induces a norm $\|f_{n,n'}\|^2_{\mathcal{H}_{n'}} = \sum_{t=0}^{\infty}\sum_{t'=0}^{\infty} \beta_{n,n',t}\ \beta_{n,n',t'}\ \kappa_{n'}(y_n[t], y_n[t'])$. It is possible to express any function in RKHS as an infinite sum of kernel evaluations weighted by $\beta_{n,n',t}$ [11].

For a node $n$, the functional dependency can be obtained by solving

$$\left\{\hat{f}_{n,n'}\right\}_{n'} = \underset{\{f_{n,n'}\in\mathcal{H}_{n'}\}}{\arg\min}\ \frac{1}{2}\sum_{\tau=0}^{T-1}\left[y_n[\tau] - \sum_{n'=1,n'\neq n}^{N} f_{n,n'}(y_{n'}[\tau])\right]^2$$

$$+\lambda \sum_{n'=1,n'\neq n}^{N} \Omega\left(||f_{n,n'}||_{\mathcal{H}_{n'}}\right), \tag{F.3}$$

where $\Omega(\cdot)$ is a regularizing function with the hyperparameter $\lambda > 0$. We consider $\Omega(x) = |x|$ to induce a sparse SEM model. In (F.3), the function $f_{n,n'}(\cdot)$ belongs to the RKHS, which is an infinite dimensional space [cf. (F.2)]. However, for non-decreasing regularizing functions such as $\Omega(x) = |x|,\ x \geq 0$, the solution of (F.3) can be expressed with a finite number of parameters using the Representer Theorem [38]:

$$\hat{f}_{n,n'}\left(y_{n'}[\tau]\right) = \sum_{t=0}^{T-1} \beta_{n,n',t}\ \kappa_{n'}(y_{n'}[\tau], y_{n'}[t])\,. \tag{F.4}$$

As the number of data samples increases, the number of kernel evaluations in (F.4) and the parameters required to express the function also increase. We overcome this curse of dimensionality using random feature (RF) approximation.

**RF approximation.** RF approximation limits the kernel evaluations to a fixed lower-dimensional Fourier space for kernels with a shift-invariant property, i.e., $\kappa_{n'}\left(y_{n'}[\tau], y_{n'}[t]\right) = \kappa_{n'}\left(y_{n'}[\tau] - y_{n'}[t]\right)$; thus, preventing the dimensionality growth. According to Bochner's theorem [30], an inverse Fourier transform of a probability distribution can represent a shift-invariant kernel:

$$\begin{aligned}\kappa_{n'}\left(y_{n'}[\tau], y_{n'}[t]\right) &= \int_{\mathbb{R}} \pi_{\kappa_{n'}}(v)\; e^{jv(y_{n'}[\tau]-y_{n'}[t])} dv \\ &= \mathbb{E}_v\left[e^{jv(y_{n'}[\tau]-y_{n'}[t])}\right], \end{aligned} \tag{F.5}$$

where $\pi_{\kappa_{n'}}(v)$ is the kernel-specific probability density function (pdf), $v$ is the random variable associated to the pdf, and $\mathbb{E}[\cdot]$ is the expectation operator. Given a sufficient number $D$ of i.i.d. samples $\{v_i\}_{i=1}^{D}$ drawn from distribution $\pi_{\kappa_{n'}}(v)$, the expectation is estimated by the sample mean:

$$\hat{\kappa}_{n'}\left(y_{n'}[\tau], y_{n'}[t]\right) = \frac{1}{D}\sum_{i=1}^{D} e^{jv_i(y_{n'}[\tau]-y_{n'}[t])}. \tag{F.6}$$

Finding the probability distribution which is the inverse Fourier transform of a kernel is a difficult task in general. However, choosing a Gaussian kernel with variance $\sigma^2$ avoids this difficulty since its Fourier transform is also a Gaussian with variance $\sigma^{-2}$. This allows writing the real part of (F.6) as

$$\begin{aligned}&\hat{\kappa}_{n'}\left(y_{n'}[\tau], y_{n'}[t]\right) = \boldsymbol{z}_{\boldsymbol{v},n'}[\tau]^\top \boldsymbol{z}_{\boldsymbol{v},n'}[t], where \quad \boldsymbol{z}_{\boldsymbol{v},n'}[\tau] = \frac{1}{\sqrt{D}}\Big[\sin\left(v_1 y_{n'}[\tau]\right), \ldots, \sin\left(v_D y_{n'}[\tau]\right), \\ &\qquad \cos\left(v_1 y_{n'}[\tau]\right), \ldots, \cos\left(v_D y_{n'}[\tau]\right)\Big]^\top. \end{aligned} \tag{F.7}$$

A fixed dimensional ($2D$) representation of the function $\hat{f}_{n,n'}(\cdot)$ is obtained by substituting (C.7) into (F.4):

$$\begin{aligned}\tilde{\hat{f}}_{n,n'}\left(y_{n'}[\tau]\right) &= \sum_{t=0}^{T-1} \beta_{n,n',t} \boldsymbol{z}_{\boldsymbol{v},n'}[\tau]^\top \boldsymbol{z}_{\boldsymbol{v},n'}[t] \\ &= \boldsymbol{\alpha}_{n,n'}{}^\top \boldsymbol{z}_{\boldsymbol{v},n'}[\tau], \end{aligned} \tag{F.8}$$

where $\boldsymbol{\alpha}_{n,n'} = \sum_{t=0}^{T-1} \beta_{n,n',t} \boldsymbol{z}_{\boldsymbol{v},n'}[t]$. Using (F.8), we can reformulate the non-parametric problem (F.3) into a parametric optimization problem:

$$\begin{aligned}\{\widehat{\boldsymbol{\alpha}}_{n,n'}\}_{n'} = \arg\min_{\{\boldsymbol{\alpha}_{n,n'}\}} \frac{1}{2}\sum_{\tau=0}^{T-1}\Bigg[y_n[\tau] - \sum_{n'=1, n'\neq n}^{N} \boldsymbol{\alpha}_{n,n'}^\top \boldsymbol{z}_{\boldsymbol{v},n'}[\tau]\Bigg]^2 \\ + \lambda \sum_{n'=1, n'\neq n}^{N} ||\boldsymbol{\alpha}_{n,n'}||_2, \end{aligned} \tag{F.9}$$

The regularizer in (F.9) is a group-lasso regularizer to enforce sparsity in the RF coefficient $\boldsymbol{\alpha}_{n,n'} \in \mathbb{R}^{2D}$. For brevity, we stack the vectors $\boldsymbol{\alpha}_{n,n'}$ and $\boldsymbol{z}_{\boldsymbol{v},n'}[t]$ along the index $n' = 1, \ldots, N,\ n' \neq n$ to form $\boldsymbol{\alpha}_n \in \mathbb{R}^{2(N-1)D}$ and $\boldsymbol{z}_n[t] \in \mathbb{R}^{2(N-1)D}$, and compactly write (F.9) as

$$\widehat{\boldsymbol{\alpha}}_n = \arg\min_{\boldsymbol{\alpha}_n} \mathcal{L}^n\left(\boldsymbol{\alpha}_n\right) + \lambda \sum_{n'=1,n'\neq n}^{N} ||\boldsymbol{\alpha}_{n,n'}||_2, \tag{F.10}$$

$$where\mathcal{L}^n(\boldsymbol{\alpha}_n) = \frac{1}{2}\sum_{\tau=0}^{T-1}\left[y_n[\tau] - \boldsymbol{\alpha}_n^\top \boldsymbol{z}_n[\tau]\right]^2. \tag{F.11}$$

Solving problem (F.10) requires access to all the batch of time series $\{y_n[\tau]\}_{\tau=0}^{T-1}$ which may be practically infeasible as they evolve over time and, at the same time, it is computationally demanding. Targeting real-world nonstationary systems with streaming data, we develop an online strategy enhanced by prediction correction mechanisms [119] that exploit the nonlinear SEM evolution. However, the group-lasso regularizer, required to enforce sparse dependencies is non-differentiable, making the deployment of prediction-correction methods not straightforward.

## F.3 Time-varying solution

### F.3.1 Online loss function

Following online optimization, we replace the batch loss in (F.11) with a recursive least square loss (RLS) using an exponential window:

$$\tilde{\ell}_t^n(\boldsymbol{\alpha}_n) = \mu \sum_{\tau=0}^{t} \gamma^{t-\tau} \ell_\tau^n(\boldsymbol{\alpha}_n). \tag{F.12}$$

where $\ell_\tau^n(\boldsymbol{\alpha}_n) = \frac{1}{2}[y_n[\tau] - \boldsymbol{\alpha}_n^\top \boldsymbol{z}_n[\tau]]^2$ is the instantaneous loss function, $\gamma \in (0,1)$ is the forgetting factor of the window, and $\mu = 1-\gamma$ normalizes the window. The RLS loss function can be expanded as

$$\begin{aligned}\tilde{\ell}_t^n(\boldsymbol{\alpha}_n) =& \frac{1}{2}\mu \sum_{\tau=0}^{t} \gamma^{t-\tau}\Big(y_n^2[\tau] + \boldsymbol{\alpha}_n^\top \boldsymbol{z}_n[\tau]\boldsymbol{z}_n[\tau]^\top \boldsymbol{\alpha}_n \\ &- 2y_n[\tau]\boldsymbol{z}_n[\tau]^\top \boldsymbol{\alpha}_n\Big) \\ =& \frac{1}{2}\mu \sum_{\tau=0}^{t} \gamma^{t-\tau} y_n^2[\tau] + \frac{1}{2}\boldsymbol{\alpha}_n^\top \boldsymbol{\Phi}_n[t]\boldsymbol{\alpha}_n - \boldsymbol{r}_n^\top \boldsymbol{\alpha}_n,\end{aligned} \tag{F.13}$$

where

$$\boldsymbol{\Phi}_n[t] = \mu \sum_{\tau=0}^{t} \gamma^{t-\tau} \boldsymbol{z}_n[\tau]\boldsymbol{z}_n[\tau]^\top, \tag{F.14}$$

$$\boldsymbol{r}_n[t] = \mu \sum_{\tau=0}^{t} \gamma^{t-\tau} y_n[\tau]\boldsymbol{z}_n[\tau]. \tag{F.15}$$

The new optimization problem using the RLS loss becomes

$$\arg\min_{\boldsymbol{\alpha}_n} \tilde{\ell}_t^n\left(\boldsymbol{\alpha}_n\right) + \lambda \sum_{n'=1,n'\neq n}^{N} \|\boldsymbol{\alpha}_{n,n'}\|_2. \tag{F.16}$$

The objective function in (F.16) has a differentiable loss but a non-differentiable regularizer. We solve it using composite objective mirror descent (COMID) [39] with the online updates:

$$\boldsymbol{\alpha}_n^{(1)}[t+1] = \arg\min_{\boldsymbol{\alpha}_n} \Bigg[ \nabla_{\boldsymbol{\alpha}} \tilde{\ell}_t^n(\boldsymbol{\alpha}_n[t])^\top (\boldsymbol{\alpha}_n - \boldsymbol{\alpha}_n[t]) \\ + \frac{1}{2\nu_t} \|\boldsymbol{\alpha}_n - \boldsymbol{\alpha}_n[t]\|_2^2 + \lambda \sum_{n'=1, n' \neq n}^{N} \|\boldsymbol{\alpha}_{n,n'}\|_2 \Bigg], \tag{F.17}$$

where $\boldsymbol{\alpha}_n^{(1)}[t+1]$ denotes the one-step COMID descent of $\boldsymbol{\alpha}_n[t]$, $\nu_t$ the step size, and $\nabla_{\boldsymbol{\alpha}} \tilde{\ell}_t^n(\boldsymbol{\alpha}_n[t])$ the gradient of $\tilde{\ell}_t^n(\boldsymbol{\alpha}_n[t])$ w.r.t. $\boldsymbol{\alpha}_n$, which can be computed from (F.13) as

$$\nabla_{\boldsymbol{\alpha}} \tilde{\ell}_t^n(\boldsymbol{\alpha}_n[t]) = \boldsymbol{\Phi}_n[t]\boldsymbol{\alpha}_n - \boldsymbol{r}_n[t]. \tag{F.18}$$

In an online setting, the parameters $\boldsymbol{\Phi}_n[t]$ and $\boldsymbol{r}_n[t]$ can be estimated recursively as $\boldsymbol{\Phi}_n[t] = \gamma \boldsymbol{\Phi}_n[t-1] + \mu \boldsymbol{z}_{\boldsymbol{v}}[t] \boldsymbol{z}_n[t]^\top$ and $\boldsymbol{r}_n[t] = \gamma \boldsymbol{r}_n[t-1] + \mu y_n[t] \boldsymbol{z}_n[t]$ [cf. (F.14) and (F.15)].

The COMID update (F.17) can be solved in closed-form for each lasso group $\boldsymbol{\alpha}_{n,n'} \in \boldsymbol{\alpha}_n$ [cf. (F.9)] using the multidimensional shrinkage thresholding operator (MSTO) [41]:

$$\boldsymbol{\alpha}_{n,n'}^{(1)}[t+1] = (\boldsymbol{\alpha}_{n,n'}[t] - \nu_t \mathbf{v}_{n,n'}) \times \\ \left[ 1 - \frac{\nu_t \lambda}{\|\boldsymbol{\alpha}_{n,n'}[t] - \nu_t \mathbf{v}_{n,n'}\|_2} \right]_+, \tag{F.19}$$

where $[\mathbf{v}_{n,1}^\top, \mathbf{v}_{n,2}^\top, \ldots, \mathbf{v}_{n,N}^\top]^\top \triangleq \nabla_{\boldsymbol{\alpha}} \tilde{\ell}_t^n(\boldsymbol{\alpha}_n[t])$ and $[x]_+ = \max\{0, x\}$. The MSTO solution (F.19) involves a one-step COMID update. For brevity of the succeeding formulation, we represent the $K$-step version of (F.19) as

$$\boldsymbol{\alpha}_n^{(K)}[t+1] = \mathrm{MSTO}^{(K)}(\tilde{\ell}_t^n(\boldsymbol{\alpha}_n[t]), \nu_t, \lambda), \tag{F.20}$$

which computes the $K$-step descent update of $\boldsymbol{\alpha}_{n,n'}[t]$ as in (F.19), for $n' = 1, \ldots, N, n' \neq n$, for the loss function $\tilde{\ell}_t^n(\cdot)$ with the parameters $\nu_t$ and $\lambda$, and stacks them to form $\boldsymbol{\alpha}_n^{(K)}[t+1]$.

### F.3.2 Prediction-Correction Algorithm

Although we can follow a time-unstructured learning strategy by directly using (F.20), such an approach discards the model evolution and leads to a suboptimal solution. Problem (F.16) features a strongly convex time-varying loss function and a properly convex regularizer, and such an optimization problem can be solved online using time-structured optimization methods that account for the model evolution. We follow the prediction-correction strategy as proposed in [119].

**Prediction.** The first step is to predict at time $t$, the yet unobserved loss function $\tilde{\ell}_{t+1}^n(\boldsymbol{\alpha}_n)$ using Taylor series expansion:

$$\tilde{\ell}_{t+1}^{n,pr}(\boldsymbol{\alpha}_n) = \boldsymbol{\alpha}_n^\top \nabla_{\boldsymbol{\alpha}\boldsymbol{\alpha}} \tilde{\ell}_t^n(\boldsymbol{\alpha}_n) \boldsymbol{\alpha}_n + [\nabla_{\boldsymbol{\alpha}} \tilde{\ell}_t^n(\boldsymbol{\alpha}_n[t]) \\ + \nabla_{t\boldsymbol{\alpha}} \tilde{\ell}_t^n(\boldsymbol{\alpha}_n[t]) - \nabla_{\boldsymbol{\alpha}\boldsymbol{\alpha}} \tilde{\ell}_t^n(\boldsymbol{\alpha}_n[t]) \boldsymbol{\alpha}_n[t]]^\top \boldsymbol{\alpha}_n \tag{F.21}$$

In addition to the gradient computed in (F.18), prediction (F.21) requires computing the Hessian $\nabla_{\boldsymbol{\alpha}\boldsymbol{\alpha}} \tilde{\ell}_t^n(\boldsymbol{\alpha}_n[t])$ and the partial derivative of $\nabla_{\boldsymbol{\alpha}} \tilde{\ell}_t^n(\boldsymbol{\alpha}_n[t])$ w.r.t. time $\nabla_{t\boldsymbol{\alpha}} \tilde{\ell}_t^n(\boldsymbol{\alpha}_n[t])$ which have the forms

$$\nabla_{\boldsymbol{\alpha}\boldsymbol{\alpha}} \tilde{\ell}_t^n(\boldsymbol{\alpha}_n[t]) = \boldsymbol{\Phi}_n[t], \tag{F.22}$$

$$\nabla_{t\boldsymbol{\alpha}} \tilde{\ell}_t^n(\boldsymbol{\alpha}_n[t]) = (\boldsymbol{\Phi}_n[t] - \boldsymbol{\Phi}_n[t-1])\boldsymbol{\alpha} - (\boldsymbol{r}_n[t] - \boldsymbol{r}_n[t-1]). \tag{F.23}$$

**Algorithm 9:** Proposed Algorithm

```
Result: $\{\boldsymbol{\alpha}_{n,n'}\}_{n,n'}$
Initialize $\lambda > 0$, $\nu_t > 0$, $D$, $\sigma_n$, P and C
for $t = 1, 2, \dots$ do
    Get data samples $y_n[t]$, $\forall n$ and compute $\boldsymbol{z_n}[t]$, $\forall n$
    for $n = 1, \dots, N$ do
        $\boldsymbol{\Phi}_n[t] = \gamma \boldsymbol{\Phi}_n[t-1] + \mu \boldsymbol{z}_n[t] \boldsymbol{z}_n[t]^\top$
        $\boldsymbol{r}_n[t] = \gamma \boldsymbol{r}_n[t-1] + \mu y_n[t] \boldsymbol{z}_n[t]$
        compute $\tilde{\ell}_{t+1}^{n,pr}(\boldsymbol{\alpha}_n)$ using (F.21)
        compute $\boldsymbol{\alpha}_n^{pr}[t+1]$ using (F.24)
        compute $\boldsymbol{\alpha}_n[t+1]$ using (F.26)
    end
end
```

The group-lasso regularizer is a time-invariant function and just performs the thresholding operation in (F.19), irrespective of the time indices. Hence, it does not require prediction. Using the predicted loss (F.21) in place of (F.16), we predict the RF coefficients as

$$\boldsymbol{\alpha}_n^{pr}[t+1] = \mathrm{MSTO}^{(P)}(\tilde{\ell}_{t+1}^{n,pr}(\boldsymbol{\alpha}_n[t]), \nu_t, \lambda), \tag{F.24}$$

where $\boldsymbol{\alpha}_n^{pr}[t+1]$ denotes the $P$-step COMID descent of $\boldsymbol{\alpha}_n[t]$ under the predicted loss. The gradient of the predicted loss involved in the MSTO operation (F.24) can be obtained from (F.21) as

$$\begin{aligned} \nabla_{\boldsymbol{\alpha}} \tilde{\ell}_{t+1}^{n,pr}(\boldsymbol{\alpha}_n[t]) =& (2\boldsymbol{\Phi}_n[t-1] - \boldsymbol{\Phi}_n[t-2])\boldsymbol{\alpha}_n \\ &+ 2\boldsymbol{r}_n[t-1] - \boldsymbol{r}_n[t-2]. \end{aligned} \tag{F.25}$$

**Correction.** At time $t+1$, the loss $\tilde{\ell}_{t+1}^{n}(\cdot)$ [cf. the one appearing in (F.16)] becomes available, and the predicted RF coefficients $\boldsymbol{\alpha}_n^{pr}[t+1]$ are corrected via $C$-step COMID descents:

$$\boldsymbol{\alpha}_n[t+1] = \mathrm{MSTO}^{(C)}(\tilde{\ell}_{t+1}^{n}(\boldsymbol{\alpha}_n^{pr}[t+1]), \nu_t, \lambda), \tag{F.26}$$

A pseudocode of the proposed prediction-correction algorithm is provided in Algorithm 9. The computational complexity of the proposed algorithm is mainly contributed by the gradient evaluation steps (F.25) and (F.18); and it is of order $\mathcal{O}(N^2 D^2)$ per node.

## F.4 Dynamic Regret

To characterize the performance of the proposed online algorithm, we analyse its dynamic regret [83], which characterizes the distance of the online loss function from the optimal counterpart in each time instant. The regret analysis is derived under the following mild assumptions:

A1) Bounded time series: there exists $B_y > 0$ such that $\{|y_n[t]|^2\}_{n,t} \le B_y \le \infty$,

A2) Shift-invariant kernels: the kernels are shift-invariant, i.e., $k(x_i, x_j) = k(x_i - x_j)$.

A3) Bounded minimum eigenvalue of $\mathbf{\Phi}_n[t]$ : There exists $\rho_l > 0$ such that $\Lambda_{\min}(\mathbf{\Phi}_n[t]) \geq \rho_l,\ \forall t$, where $\Lambda_{\min}(\cdot)$ is the minimum eigenvalue operator.

A4) Bounded maximum eigenvalue: there exists $L > 0$ such that $2\Lambda_{\max}(\mathbf{\Phi}_n[t]) < L < \infty,\ \forall t$, where $\Lambda_{\max}(\cdot)$ is the maximum eigenvalue operator.

Dynamic regret is defined as the sum of differences between the online estimated cost function and optimal cost function:

$$R_n[T] = \sum_{t=0}^{T-1} \Big[ h_t^n(\boldsymbol{\alpha}_n[t], \boldsymbol{z}_n[t]) - h_t^n(\boldsymbol{\beta}_n^*[t], \boldsymbol{\kappa}_n[t]) \Big], \tag{F.27}$$

where $\boldsymbol{\alpha}_n[t]$ collects the estimated RF coefficients [cf. (F.26)] and $\boldsymbol{z}_n[t]$ is the RF features; and $\boldsymbol{\beta}_n^*[t] \in \mathbb{R}^{(N-1)t}$ and $\boldsymbol{\kappa}_n[t]$ are the optimal coefficients and the kernel-based features in RKHS, respectively. The function $h_t^n(\cdot,\cdot)$ is defined as

$$h_t^n(\mathbf{w}, \mathbf{x}) = \frac{1}{2}[y_n[t] - \mathbf{w}^\top \mathbf{x}]^2 + \lambda \sum_{n'=1}^{N} \|\mathbf{w}_{n,n'}\|_2, \tag{F.28}$$

which is related to (F.10) by replacing the cumulative loss by an instantaneous loss. We also define the optimal RF coefficients $\boldsymbol{\alpha}_n^*[t]$ as

$$\boldsymbol{\alpha}_n^*[t] = \arg\min_{\boldsymbol{\alpha}_n} h_t^n(\boldsymbol{\alpha}_n, \boldsymbol{z}_n[t]). \tag{F.29}$$

Adding and subtracting $h_t^n(\boldsymbol{\alpha}_n^*[t], \boldsymbol{z}_n[t])$ in (F.27) gives

$$\begin{aligned} R_n[T] =& \underbrace{\sum_{t=0}^{T-1} (h_t^n(\boldsymbol{\alpha}_n[t], \boldsymbol{z}_n[t]) - h_t^n(\boldsymbol{\alpha}_n^*[t], \boldsymbol{z}_n[t]))}_{R_n^{\mathrm{rf}}[T]} \\ &+ \underbrace{\sum_{t=0}^{T-1} (h_t^n(\boldsymbol{\alpha}_n^*[t], \boldsymbol{z}_n[t]) - h_t^n(\boldsymbol{\beta}_n^*[t], \boldsymbol{\kappa}_n[t]))}_{\xi_n[T]}, \end{aligned} \tag{F.30}$$

where $R_n^{\mathrm{rf}}[T]$ is the regret w.r.t. optimal cost in RF space and $\xi_n[T]$ is the cumulative error in RF approximation. Dynamic regret can be bounded by bounding $R_n^{\mathrm{rf}}[T]$ and $\xi_n[T]$.

**Theorem 4.** *Under assumptions A1, A2, A3, and A4, the dynamic regret $R_n(T)$ satisfies*

$$\begin{aligned} R_n(T) \leq &\Big(\Big(1 + \frac{L}{2\rho_l}\Big)\sqrt{2(N-1)DB_y} + \lambda\sqrt{N-1}\Big)\times \\ &T\Big(q^{(P+C)}\|\boldsymbol{\alpha}_n^*[0]\|_2 + q^{(P+C)}d + q^{(P+C+1)}l\Big) + \epsilon\eta L_h T, \end{aligned}$$

*where $\eta > 0$ is a constant, $L_h$ is the Lipschitz continuity parameter of function $h_t^n(\cdot,\cdot)$, $d$ is the maximum temporal variation in the optimal solution $\|\boldsymbol{\alpha}_n^*[t] - \boldsymbol{\alpha}_n^*[t-1]\|_2$, and $l$ is the maximum error in the optimal prediction $\|\boldsymbol{\alpha}_n^*[t] - \boldsymbol{\alpha}_n^{pr*}[t]\|_2$ with $\boldsymbol{\alpha}_n^{pr*}[t]$ the optimum prediction at time $t$. The quantity $q \in (0,1)$ is the contraction coefficient, and its value for various optimization techniques is provided in [51].*

**Proof**: See Appendix F.7.

The dynamic regret bound in Theorem 4 is linear in time, which implies that $\lim_{t\to\infty} R_n(T)/T = \mathit{constant}$, where *constant* is the steady state error, which depends on $l = \|\boldsymbol{\alpha}_n^*[t] - \boldsymbol{\alpha}_n^{pr*}[t]\|_2$, $d = \|\boldsymbol{\alpha}_n^*[t] - \boldsymbol{\alpha}_n^*[t-1]\|_2$, and the constant $\epsilon \geq 0$.

This means that if $d$ and $l$ are low (slowly varying systems), it is possible to have a very low bound for the asymptotic $R_n(T)/T$ by controlling $\epsilon$ in the expense of model complexity.

## F.5 Numerical Experiments

This section compares the proposed algorithm with competing alternatives using both synthetic data from Erdös-Rényi graph models and real data from epileptic seizure and financial time series. We compare the proposed approach with the following alternatives:

- Pro-SEM: the time-unstructured linear time-varying SEM from [8], based on a proximal online gradient framework;
- TV-SEM: the time-structured linear time-varying SEM from [49];
- MSTO: A nonlinear SEM by merely performing a one-step multidimensional shrinkage thresholding [cf. (21)] without any prediction-correction steps.

The first two alternatives are considered as baselines as they have also shown superior performance to other online learning strategies in the respective papers. Instead, the third alternative is considered to highlight the importance of the proposed time-structured strategy.

In all experiments, the proposed algorithm has one-step prediction ($P = 1$) and one-step correction ($C = 1$). Wherever the SEM topologies are plotted for visualization, we use the normalized $\ell_2$ norms of the RF coefficients as the topology estimates, defined as $b_{n,n'}[t] := \|\boldsymbol{\alpha}_{n,n'}[t]\|_2/(\max_m \|\boldsymbol{\alpha}_{n,m}[t]\|_2)$.

### F.5.1 Synthetic data

In this experiment, we consider simulated data from a slowly-varying SEM model. We generate graph-connected time series using the following nonlinear SEM model:

$$\mathbf{y}[t] = 0.1(\mathbf{I}-\mathbf{W}[t])^{-1}\mathbf{u}[t] + 0.1\sin((\mathbf{I}-\mathbf{W}[t])^{-1}\mathbf{u}[t]), \tag{F.31}$$

where $\mathbf{y}[t] \in \mathbb{R}^5$ is the signal at time $t$, $\mathbf{u}[t] \sim \mathcal{N}(0, 0.1)$, $\mathbf{I} \in \mathbb{R}^{5\times 5}$ is the identity matrix, and the operator $\sin(\cdot)$ acts element-wise to introduce non-linearities. The matrix $\mathbf{W}[t] \in \mathbb{R}^{5\times 5}$ is constructed such that it attributes slowly-evolving model dynamics to (F.31), and is of the form:

$$\mathbf{W}[t+1] = \mathbf{W}[t] + 0.001\sin(0.01t)\mathbf{W}[t], \tag{F.32}$$

where $\mathbf{W}[0] \in \mathbb{R}^{5\times 5}$ is constructed using an Erdös-Rényi random graph with diagonal entries zero[1].

Our synthetic data set consists of 100 multi-variate time series, generated using (F.31), each having $T = 5000$ signal samples. Out of the 100 multi-variate time series, 20 are used to tune the hyperparameter of all the algorithms based on a grid

[1] We choose a small Erdös-Rényi graph of size $5 \times 5$ to corroborate the dynamic regret, which involves high computational complexity.

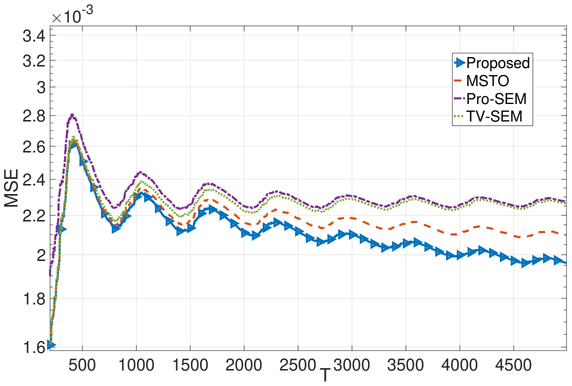


Figure F.1: MSE comparison on the synthetic data set.

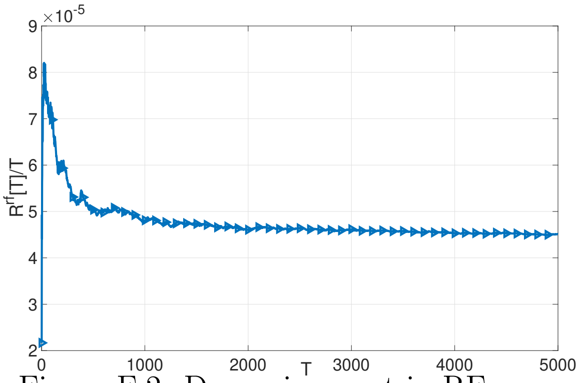


Figure F.2: Dynamic regret in RF space.

search for the best model fitness. The model fitness is measured via *Mean Squared Error* (MSE), defined as

$$\mathrm{MSE}[T] = \frac{\sum_{t=0}^{T-1} \|\mathbf{y}[t] - \hat{\mathbf{y}}[t]\|_2^2}{NT}, \tag{F.33}$$

where $\hat{\mathbf{y}}[t] \in \mathbb{R}^5$ is the signal estimated using the learned SEM model. The hyperparameter values of the algorithm are $(\sigma_n, \lambda, \gamma, \nu_t) = (5, 0.0009, 0.98, 2/\max\{\Lambda_{\max}(\mathbf{\Phi}_n[t])\}_n)$ and the RF count is $D = 5$. The MSEs averaged across the remaining 80 multivariate time series are plotted in Fig. F.1, which shows that the proposed method outperforms all alternatives. This is because the alternatives do not exploit the evolution of the model or cannot learn non-linearities, whereas the proposed algorithm features both.

**Dynamic Regret.** In Fig. F.2, we plot the rate of change of the dynamic regret w.r.t. optimal cost function in RF space $R_n^{\mathrm{rf}}[T]/T$. The convergence of $R^{\mathrm{rf}}[T]/T$ is evident from Fig. F.2, which supports our theoretical analysis in Theorem 4. We wish to note that a numerical evaluation of the second component of the dynamic regret $\xi_n[T]$ is a daunting, complex process since it involves finding the optimal parameters in a high dimensional RKHS. However, $\xi_n[T]/T$ is upper bounded by the value $\epsilon\eta L_h$ [cf. Lemma 3], where $\epsilon$ is a user-controlled parameter. By setting $\epsilon$ to be very small, the rate of change of the overall dynamic regret $R_n[T]/T$ can be made closer to $R^{\mathrm{rf}}[T]/T$, when $T \to \infty$.

### F.5.2 Real data: Epileptic seizure

In this experiment, we examine the functional connectivities among different brain regions via learned SEM topologies using an EEG dataset. Our goal is to distinguish between the normal and epileptic dynamics in the brain networks. We use an EEG dataset of children with intractable seizures collected from the Children's Hospital, Boston [92]. The data set consists of multivariate time series of potential differences between electrodes inserted in the brain. There are a total of 23 times series measuring EEG activities in different brain regions. We fit this data using different algorithms and test their capability to distinguish the pre-seizure and the seizure events. We measure the performance via the *Maximum Mean Discrepancy* (MMD) of the distribution of nodal degrees, which is a standard approach used to measure the distance between two graphs [121,122]. The MMD is defined as

$$\begin{aligned}\mathrm{MMD}^2(p_1||p_2) =&\mathbb{E}_{\mathbf{x},\mathbf{y}\sim p_1}\big[k(\mathbf{x},\mathbf{y})\big] + \mathbb{E}_{\mathbf{x},\mathbf{y}\sim p_2}\big[k(\mathbf{x},\mathbf{y})\big] \\ &- 2\mathbb{E}_{\mathbf{x}\sim p_1,\mathbf{y}\sim p_2}\big[k(\mathbf{x},\mathbf{y})\big] \qquad \text{(F.34)}\end{aligned}$$

where $k(\mathbf{x},\mathbf{y})$ is the radial basis kernel function computing the distance between $\mathbf{x}$ and $\mathbf{y}$, and $\mathrm{MMD}^2$ measures the distance between distributions $p_1$ and $p_2$. In this experiment, $p_1$ and $p_2$ correspond to the distributions of nodal degrees for the pre-seizure and seizure events, respectively.

We used the proposed method with the RF count $D = 5$ along with the hyperparameters $(\sigma_n, \lambda, \gamma, \nu_t) = (1, 0.1, .98, 2/\max\{\Lambda_{\max}(\mathbf{\Phi}_n[t])\})$, obtained using a grid search for the best MMD. The hyperparameters of other algorithms are also tuned using the same strategy.

Table F.1 compares the MMD of the different algorithms using the seizure data from two subjects, $S_1$ and $S_2$. The MMD of the proposed algorithm is an order-one magnitude higher compared to alternatives, which highlights that the proposed algorithm distinguishes the seizure and the pre-seizure events better. This is due to the fact that the functional connectivities in brain are highly nonlinear [11], and all alternatives, except MSTO, discard the nonlinear components in the connectivity. MSTO, on the other hand, can accommodate the non-linearities; however, it does not take advantage of the brain connectivity evolution, and is at the second place in the comparison.

A snapshot of the estimated graph topology before seizure and after seizure is shown in Fig. F.3 and Fig. F.4, respectively. Before the seizure, the connections are concentrated across certain regions, and during the seizure, they get more disrupted, which agrees with the observations in [95]. The reason for the disrupted topology is the increase in pathogenic neural discharge during seizure [96].

We further compare the per-node computational complexity of the proposed method and the time-structured benchmark TV-SEM. The experiment is conducted in a machine with specifications: 2.4 GHz 8-core Intel Core $i9$ and $16GB$ $2667$ $MHz$ $DDR4$ $RAM$. In Fig. F.6, we plot the cumulative computation time of the prediction and the correction steps, where it can be observed that the proposed model performs the prediction and the correction much faster. The shorter computation time stems from the node separability feature, which the TV-SEM does

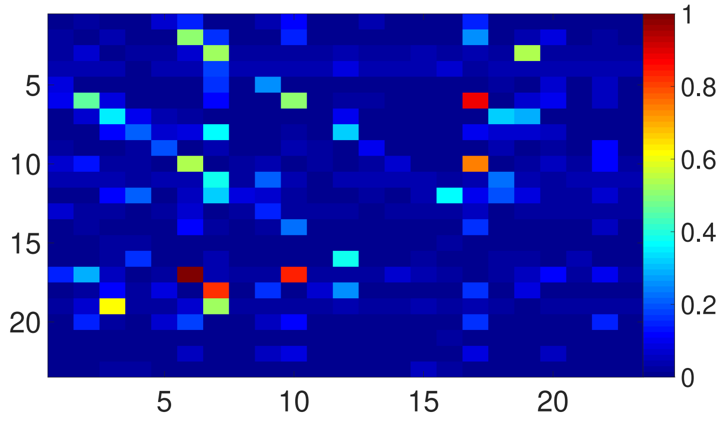


Figure F.3: Before seizure.

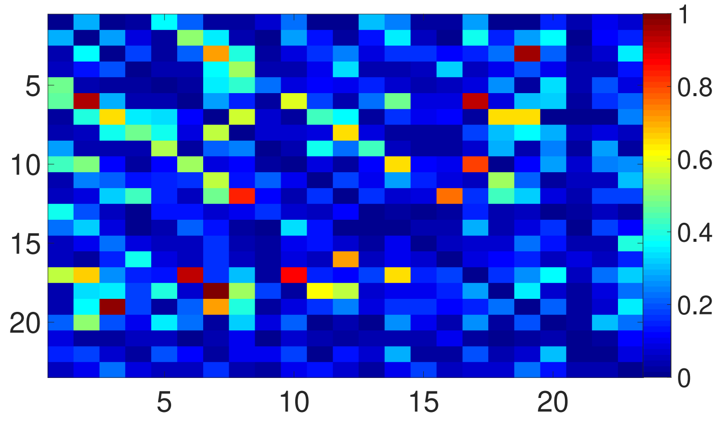


Figure F.4: During seizure.

Figure F.5: Snapshots of estimated topologies: (a) before seizure, (b) during seizure.

not have. The other alternatives are not considered in Fig. F.6 since they are time-unstructured algorithms that do not take advantage of the model evolution, and hence, are faster than the time-structured methods.

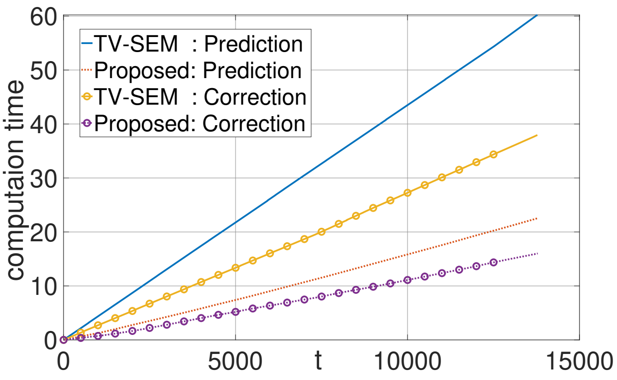


Figure F.6: Comparison of cumulative computational time on epileptic data.

### F.5.3 Financial time series

We consider financial time series belonging to three categories: airline industry, oil industry, and cryptocurrency, which are listed in Table F.2. The data set includes

Table F.1: Maximum Mean Discrepancy for node degree on EEG data.

| MMD | $S_1$ | $S_2$ |
|---|---|---|
| Proposed | **0.0532** | **0.0550** |
| TV-SEM | 0.0037 | 0.0038 |
| Pro-SEM | 0.0032 | 0.0013 |
| MSTO | 0.0067 | 0.0070 |

Table F.2: Categorized list of financial times series.

| Groups | Stocks |
|---|---|
| Group-1 (Airlines) | Delta Air Lines (DAL), Air Canada (AC), Air France (AF), United Airlines (UAL), American Airlines (AAL). |
| Group-2 (Oil) | British Petroleum (BP), ConocoPhillips (COP), Chevron (CVX), Shell (SHEL), ExxonMobil (XOM). |
| Group-3 (Crypto) | Bitcoin (BTC), Dogecoin (DOGE), Ripple (XRP), Cardano (ADA), Ethereum (ETH). |

Table F.3: Clustering coefficients of stock groups under COVID and post-COVID market dynamics, computed using (F.35).

| | Algorithm | *Airlines* | *Oil* | *Crypto* |
|---|---|---|---|---|
| COVID | Proposed | 0.45 | **0.54** | **.54** |
| | TV-SEM | 0.45 | 0.44 | 0.45 |
| | Pro-SEM | 0.23 | 0.33 | 0.00 |
| | MSTO | 0.38 | 0.45 | 0.40 |
| post-COVID | Proposed | **0.81** | **0.80** | 1.00 |
| | TV-SEM | 0.60 | 0.40 | 0.44 |
| | Pro-SEM | 0.20 | 0.42 | 1.00 |
| | MSTO | 0.63 | 0.54 | 1.00 |

15 time series of 879 samples each, which are the closing price values of the stocks from 01-06-2019 to 14-10-2022, including the COVID-19 outbreak. The pandemic had a serious impact on world economy, affecting the natural dynamics of the stock market. A high dip in the S&P 500 index was observed around 25-02-2020 to 25-06-2020, which we mark as the pandemic period. Our goal in this experiment is to identify clusters in the data using the learned SEM topologies and examine the

variations in the clusters during and after the pandemic. Since the stock groups in Table F.2 are formed by selecting the stocks from similar industries, they are expected to show stronger intra-group dependencies than intergroup dependencies, under the normal market conditions [123].

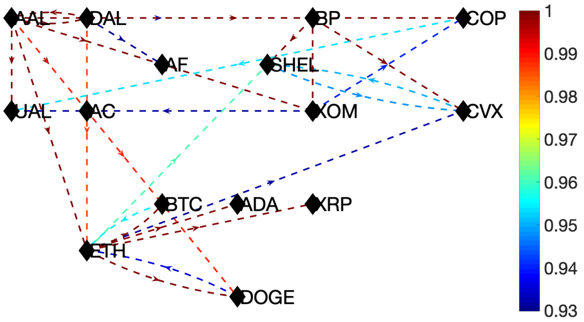


Figure F.7: Estimated SEM topology on 05-05-2020 (during COVID).

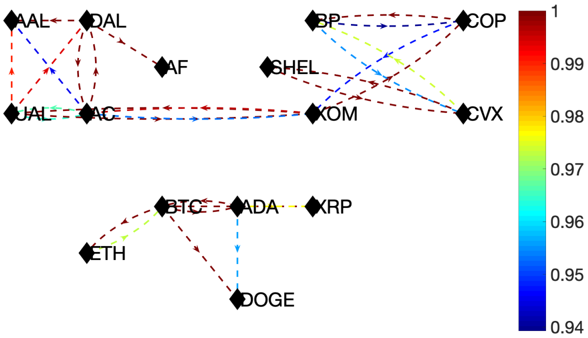


Figure F.8: Estimated SEM topology on 08-12-2021 (after COVID).

Let $\mathcal{V}_i = 1, 2, 3$, denote the set of nodes corresponding to the stocks in each group. We measure the performance via the clustering coefficient $\rho_i$ that computes the ratio of the number of edges within group-$i$ to the total number of edges connected to group-$i$ members:

$$\rho_i = \frac{\sum_{n \in \mathcal{V}_i} 1(b_{n,n'} > \delta | n' \in \mathcal{V}_i)}{\sum_{n \in \mathcal{V}_i} 1(b_{n,n'} > \delta) + \sum_{n' \in \mathcal{V}_i} 1(b_{n,n'} > \delta)}, \tag{F.35}$$

where $\delta$ is a threshold selected to consider the strongest $2N$ edges for clustering; and $1(\cdot)$ is an indicator function defined as $1(x) = 1$, when $x$ is *true*, and 0, otherwise. A high value of $\rho_i$ indicates that intra-group interactions in group-$i$ are stronger compared to its intergroup interactions. The first 20% of the data samples are used to tune the hyperparameter for the lowest MSE resulting in $(\sigma_n, \lambda, \gamma, \nu_t) = (1, 1, .98, 2/\max\{\Lambda_{\max}(\mathbf{\Phi}_n[t])\})$ and RF count for the experiment is $D = 10$.

Table F.3 lists the clustering coefficients of the three groups, averaged across 80 days, randomly sampled from the COVID and post-COVID intervals. As expected, the clustering is more predominant with post-COVID market dynamics than with the COVID market dynamics. The proposed method identifies better such clusters compared with the alternatives. The MSTO algorithm is next in the comparison. This observation is supported by the fact that the interactions among the financial time series are complex [124], which cannot be effectively modeled using the linear Pro-SEM and TV-SEM. It is further interesting to note here as the crypto cluster is much easier identified in the post-COVID period. This follows the intuition that the airline and oil sectors have more financial transactions between them, whereas cryptocurrencies are exchanged only with each other.

Further, the SEM topologies estimated using the proposed algorithm for a COVID-affected market day and a post-COVID day are shown in Fig. F.7 and Fig. F.8, respectively. In line with the expectation, more intra-group market interactions can be observed in Fig. F.8, whereas these interactions get disrupted in Fig. F.7.

## F.6 Conclusion

This paper proposed an online algorithm to learn the nonlinear structural equation model (SEM), targeting the streaming data from real-world systems with nonlinear dynamics. The proposed method leverages the kernel formulation with random feature approximation to obtain a low-dimensional representation of the nonlinear dynamics. The algorithm uses a prediction-correction strategy equipped with a group-lasso-based optimization framework, solved via composite object mirror descent. Unlike the state-of-the-art algorithms, the proposed method offers data privacy at the network node through node separability and random features. In addition, the proposed online problem is separable across nodes, improving scalability in large graphs. A dynamic regret analysis has been derived to ensure the theoretical guarantee of the algorithm. Using synthetic, epileptic, and financial data, we demonstrated that the SEM topology learned using the proposed model fits the data better and can distinguish between the changes in the system dynamics with less computational complexity compared to the state-of-the-art alternatives.

## F.7 Proof of Theorem 4

Theorem 4 provides an upper bound for the dynamic regret $R_n(T) = R_n^{\text{rf}}(T) + \xi_n(T)$. We prove the theorem by bounding $R_n^{\text{rf}}(T)$ and $\xi_n(T)$ using the following two lemmas.

**Lemma 2.** *Under assumptions A1, A3, and A4, and letting $\nu_t = \frac{2}{L}$, the dynamic regret w.r.t. the optimal cost function in the RF space is upper bounded by*

$$R_n^{rf}(T) \leq \Big(\Big(1 + \frac{L}{2\rho_l}\Big)\sqrt{2(N-1)DB_y} + \lambda\sqrt{N-1}\Big) \times \\ T\Big(\|\boldsymbol{\alpha}_n^*[0]\|_2 + q^{(P+C)}d + q^{(P+C+1)}l\Big).$$

**Proof**: The Cauchy-Schwarz inequality allows us to bound $R_n^{\text{rf}}[T]$ by bounding the cumulative optimality gap $\sum_{t=0}^{T-1} \|\boldsymbol{\alpha}_n[t] - \boldsymbol{\alpha}_n^*[t]\|_2$ and the gradient of the loss function $\|\nabla\tilde{\ell}_t^n(\boldsymbol{\alpha}_n[t])\|_2$ [32].

The bound for optimality gap is given by Proposition-1 in [118]:

$$\|\boldsymbol{\alpha}_n[t] - \boldsymbol{\alpha}_n^*[t]\|_2 \leq q^C(q^P\|\boldsymbol{\alpha}_n[t-1] - \\ \boldsymbol{\alpha}_n^*[t-1]\|_2 + q^P d + (1+q^P)l) \qquad \text{(F.36)}$$

Since $q < 1$, we can express cumulative error in terms of the initial optimal solution $\boldsymbol{\alpha}_n^*[0]$. Setting $\boldsymbol{\alpha}_n[0] = 0$, we bound the cumulative optimality gap as

$$\sum_{t=0}^{T-1} \|\boldsymbol{\alpha}_n[t] - \boldsymbol{\alpha}_n^*[t]\|_2 \leq Tq^{(P+C)}\|\boldsymbol{\alpha}_n^*[0]\|_2 + Tq^{(P+C)}d + Tq^{(P+C+1)}l \tag{F.37}$$

The gradient of the loss is bounded by following Lemma 1.2 in [36]:

$$\|\nabla\tilde{\ell}_t^n(\boldsymbol{\alpha}_n[t])\|_2 \leq \left(\left(1 + \frac{L}{2\rho_l}\right)\sqrt{2(N-1)DB_y} + \lambda\sqrt{N-1}\right) \tag{F.38}$$

The claim can be then proved by adding (F.37) and (F.38). □

**Lemma 3.** *Under assumptions A1 and A2, there exists a constant $\epsilon \geq 0$ such that the cumulative approximation error $\xi_n[T]$ satisfies*

$$\xi_n(T) \leq \epsilon\eta L_h T.$$

**Proof**: The proof follows from Theorem 2 in [36]. □

# Bibliography

[1] Alberto Natali, Elvin Isufi, Mario Coutino, and Geert Leus. Learning time-varying graphs from online data. *IEEE Open Journal of Signal Processing*, 2022.

[2] Mario Coutino, Elvin Isufi, Takanori Maehara, and Geert Leus. State-space network topology identification from partial observations. *IEEE Transactions on Signal and Information Processing over Networks*, 2020.

[3] Mishfad Veedu, Doddi Harish, and Murti V. Salapaka. Topology learning of linear dynamical systems with latent nodes using matrix decomposition. *IEEE Transactions on Automatic Control*, 2021.

[4] Rafael E. Carrillo, Martin Leblanc, Baptiste Schubnel, Renaud Langou, Cyril Topfel, and Pierre-Jean Alet. High-resolution pv forecasting from imperfect data: A graph-based solution. *Energies*, 13, 2020.

[5] Koki Yamada, Yuichi Tanaka, and Antonio Ortega. Time-varying graph learning based on sparseness of temporal variation. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019.

[6] B. Zaman, L. M. L. Ramos, and B. Beferull-Lozano. Online joint topology identification and signal estimation with inexact proximal online gradient descent. *10.48550/ARXIV.2012.05957*, 2020.

[7] David Hallac, Youngsuk Park, Stephen P. Boyd, and Jure Leskovec. Network inference via the time-varying graphical lasso. *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2017.

[8] Bakht Zaman, Luis Miguel Lopez Ramos, and Baltasar Beferull-Lozano. Dynamic regret analysis for online tracking of time-varying structural equation model topologies. 2020.

[9] Alex Tank, Ian Covert, Nicholas Foti, Ali Shojaie, and Emily Fox. Neural granger causality for nonlinear time series. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[10] L. Lopez-Ramos, K. Roy, and B. Beferull-Lozano. Explainable nonlinear modelling of multiple time series with invertible neural networks. *INTAP*, 2021.

[11] Y. Shen, G. Giannakis, and B. Baingana. Nonlinear structural vector autoregressive models with application to directed brain networks. *IEEE Transactions on Signal Processing*, 67:5325–5339, 2019.

[12] Mircea Moscu, Ricardo Borsoi, and Cédric Richard. Online kernel-based graph topology identification with partial-derivative-imposed sparsity. In *Signal Processing (EUSIPCO), 28th European Conference on*, 2020.

[13] C.J. Stam. *Nonlinear brain dynamics*. Nova Biomedical, 2006.

[14] Y. Shen, B. Baingana, and G. B. Giannakis. Kernel-based structural equation models for topology identification of directed networks. *IEEE Transactions on Signal Processing*, 65(10):2503–2516, 2017.

[15] B. Scholkopf and Alexander J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond.* MIT Press, Cambridge, MA, USA, 2001.

[16] A. Rahimi and B. Recht. Random features for large-scale kernel machines. In *NeurIPS*, NIPS'07, 2007.

[17] Yanning Shen, Geert Leus, and Georgios B. Giannakis. Online graph-adaptive learning with scalability and privacy. *IEEE Transactions on Signal Processing*, 2019.

[18] 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*, 2019.

[19] Amarlingam Madapu, Santiago Segarra, Sundeep Chepuri, and Antonio G. Marques. Generative adversarial networks for graph data imputation from signed observations. *IEEE ICASSP*, 2020.

[20] Junteng Jia, M. Schaub, Santiago Segarra, and A. Benson. Graph-based semi-supervised active learning for edge flows. *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery Data Mining - KDD '19*, 2019.

[21] R. Money, J. Krishnan, and B. Beferull-Lozano. Online non-linear topology identification from graph-connected time series. *IEEE DSLW*, 2021.

[22] R. Money, J. Krishnan, and B. Beferull-Lozano. Random feature approximation for online nonlinear graph topology identification. *IEEE MLSP Workshop*, 2021.

[23] R. Money, J. Krishnan, and B. Beferull-Lozano. Online Inference of Nonlinear Causal Topology from Multivariate Time Series. *10.36227/techrxiv.19210092.v2*, 2022.

[24] Rohan Money, Joshin Krishnan, and Baltasar Beferull-Lozano. Online joint nonlinear topology identification and missing data imputation over dynamic graphs. *2022 30th European Signal Processing Conference (EUSIPCO)*, 2022.

[25] Rohan Money, Joshin Krishnan, Baltasar Beferull-Lozano, and Elvin Isufi. Online edge flow imputation on networks. *IEEE Signal Processing Letters*, 2023.

[26] Rohan Money, Joshin Krishnan, Baltasar Beferull-Lozano, and Elvin Isufi. Scalable and privacy-aware online learning of nonlinear structural equation models. *IEEE Open Journal of Signal Processing*, 2023.

[27] D. Marinazzo, M. Pellicoro, and S. Stramaglia. Kernel-granger causality and the analysis of dynamical networks. *Physical Review E*, 2008.

[28] Vitor Rosa Meireles Elias, Vinay Chakravarthi Gogineni, Wallace A. Martins, and Stefan Werner. Kernel regression over graphs using random fourier features. *IEEE Transactions on Signal Processing*, 70:936–949, 2022.

[29] Kasper Green Larsen and Jelani Nelson. Optimality of the johnson-lindenstrauss lemma. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, 2017.

[30] S. Bochner. Lectures on fourier integrals. *Princeton University press*, 42, 1959.

[31] Yanning Shen, Geert Leus, and Georgios B. Giannakis. Online graph-adaptive learning with scalability and privacy. *IEEE Transactions on Signal Processing*, 67(9):2471–2483, 2019.

[32] Rishabh Dixit, Amrit Singh Bedi, Ruchi Tripathi, and Ketan Rajawat. Online learning with inexact proximal online gradient descent algorithms. *IEEE Transactions on Signal Processing*, 67(5):1338–1352, 2019.

[33] R. E. Kalman. A New Approach to Linear Filtering and Prediction Problems. *Journal of Basic Engineering*, 1960.

[34] Maosheng Yang, Elvin Isufi, M. T. Schaub, and Geert Leus. Simplicial convolutional filters. *arXiv preprint arXiv:2201.11720*, 2022.

[35] Elvin Isufi and Maosheng Yang. Convolutional filtering in simplicial complexes. In *IEEE ICASSP*, 2022.

[36] Rohan Money, J. Krishnan, and B. Beferull-Lozano. Sparse online learning with kernels using random features for estimating nonlinear dynamic graphs. *10.36227/techrxiv.19210092.v3.*, 2022.

[37] G. Wahba. *Spline Models for Observational Data.* SIAM Press, Society for Industrial and Applied Mathematics, 1990.

[38] B. Olkopf, R. Herbrich, A. Smola, and Robert Williamson. A generalized representer theorem. *Computational Learning Theory*, 42, 06 2000.

[39] J. Duchi, S. Shwartz, and A. Tewari. Composite objective mirror descent. *COLT'10*, pages 14–26, 2010.

[40] M. Gutmann and J. Hirayama. Bregman divergence as general framework to estimate unnormalized statistical models. In *Proceedings of UAI*, UAI'11, page 283–290, Arlington, Virginia, USA, 2011. AUAI Press.

[41] A. Puig, A. Wiesel, and A. Hero. A multidimensional shrinkage-thresholding operator. volume 18, pages 113 – 116, 10 2009.

[42] J. Lu, S. Hoi, J. Wang, P. Zhao, and Z. Liu. Large scale online kernel learning. *Journal of Machine Learning Research*, 17(47):1–43, 2016.

[43] Y. Shen, T. Chen, and G. Giannakis. Random feature-based online multi-kernel learning in environments with unknown dynamics. *J. Mach. Learn. Res.*, 20(1):773–808, January 2019.

[44] Simon Haykin. *Adaptive filter theory.* Prentice Hall, 4th edition, 2002.

[45] Sergio Barbarossa and S Stefania. Topological signal processing over simplicial complexes. *IEEE Transactions on Signal Processing*, 2020.

[46] John Mattingley and Stephen Boyd. Real-time convex optimization in signal processing. *IEEE Signal Processing Magazine*, 2010.

[47] L Hongqing, L Yong, Z Yi, and T Trieu-Kien. Sparse kalman filter. *IEEE ChinaSIP*, 2015.

[48] B. Zaman, L. M. L. Ramos, D. Romero, and B. Beferull-Lozano. Online topology identification from vector autoregressive time series. *IEEE Transactions on Signal Processing*, 69:210–225, 2021.

[49] N. Alberto, C. Mario, I. Elvin, and L. Geert. Online time-varying topology identification via prediction-correction algorithms. In *IEEE ICASSP*, 2021.

[50] Andrea Simonetto, Emiliano Dall'Anese, Santiago Paternain, Geert Leus, and Georgios B. Giannakis. Time-varying convex optimization: Time-structured algorithms and applications. *Proceedings of the IEEE*, 2020.

[51] Amir Beck. *First-Order Methods in Optimization.* Society for Industrial and Applied Mathematics, Philadelphia, PA, 2017.

[52] X. Dong, D. Thanou, M. Rabbat, and P. Frossard. Learning graphs from data: A signal representation perspective. *IEEE Signal Processing Magazine*, 36(3):44–63, 2019.

[53] B. Zaman, L. M. Lopez-Ramos, D. Romero, and B. Beferull-Lozano. Online topology estimation for vector autoregressive processes in data networks. In *2017 IEEE CAMSAP*, pages 1–5, 2017.

[54] L. Lopez-Ramos, D. Romero, B. Zaman, and B. Beferull-Lozano. Dynamic network identification from non-stationary vector autoregressive time series. In *2018 IEEE GlobalSIP*, pages 773–777, Nov 2018.

[55] A. Chatterjee, R. J. Shah, and S. Sen. Pattern matching based algorithms for graph compression. In *2018 Fourth International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN)*, pages 93–97, 2018.

[56] C. J. Quinn, N. Kiyavash, and T. P. Coleman. Equivalence between minimal generative model graphs and directed information graphs. In *2011 IEEE International Symposium on Information Theory Proceedings*, pages 293–297, 2011.

[57] G. Giannakis, Y. Shen, and G. Karanikolas. Topology identification and learning over graphs: Accounting for nonlinearities and dynamics. *Proceedings of the IEEE*, 106(5):787–807, May 2018.

[58] Noah Simon, Jerome Friedman, Trevor Hastie, and Robert Tibshirani. A sparse-group lasso. *Journal of Computational and Graphical Statistics*, 22(2):231–245, 2013.

[59] G. Chen, D. Glen, Z. Saad, J. Hamilton, E. Thomason, H. Gotlib, and R. Cox. Vector autoregression, structural equation modeling, and their synthesis in neuroimaging data analysis. *Computers in biology and medicine*, 41(12):1142—1155, December 2011.

[60] Charles A. Micchelli, Yuesheng Xu, and Haizhang Zhang. Universal kernels. *J. Mach. Learn. Res.*, 7:2651–2667, December 2006.

[61] Y. Shen and G. B. Giannakis. Online identification of directional graph topologies capturing dynamic and nonlinear dependencies†. In *2018 IEEE Data Science Workshop (DSW)*, pages 195–199, 2018.

[62] S. Mollaebrahim and B. Beferull-Lozano. Design of asymmetric shift operators for efficient decentralized subspace projection. *IEEE Transactions on Signal Processing*, 69:2056–2069, 2021.

[63] H. Lütkepohl. *New introduction to multiple time series analysis*. Springer, Berlin [u.a.], 2005.

[64] A. Koppel, G. Warnell, E. Stump, and A. Ribeiro. Parsimonious online learning with kernels via sparse projections in function space. In *2017 IEEE ICASSP*, pages 4671–4675, March 2017.

[65] F. Youping, L. Jingjiao, and Z. Dai. A method for identifying critical elements of a cyber-physical system under data attack. *IEEE Access*, 6, 2018.

[66] H. Weiyu, G. Leah, W. Nicholas, G. Scott, B. Danielle, and R. Alejandro. Graph frequency analysis of brain signals. *IEEE Journal of Selected Topics in Signal Processing*, 10(7), 2016.

[67] D. Cheng, F. Yang, S. Xiang, and J. Liu. Financial time series forecasting with multi-modality graph neural network. *Pattern Recognition*, 121, 2022.

[68] Shai Shalev-Shwartz. Online learning and online convex optimization. *Foundations and Trends® in Machine Learning*, 4(2):107–194, 2012.

[69] A. Simonetto, E. Dall'Anese, S. Paternain, G. Leus, and G. B. Giannakis. Time-varying convex optimization: Time-structured algorithms and applications, 2020.

[70] M. Singh and V. Kekatos. Optimal scheduling of water distribution systems. *IEEE Transactions on Control of Network Systems*, pages 1–1, 2019.

[71] N. Meike, B. Doina, and S. Christin. Causal discovery with attention-based convolutional neural networks. *Machine Learning and Knowledge Extraction*, 1, 2019.

[72] J. Lu, s. C.H. Hoi, J. Wang, P. Zhao, and Z. Liu. Large scale online kernel learning. *Journal of Machine Learning Research*, 2016.

[73] L. Zhang, J. Yi, R. Jin, M. Lin, and X. He. Online kernel learning with a near optimal sparsity bound, 2013.

[74] V. Michel and F. Damien. The curse of dimensionality in data mining and time series prediction. In *Computational Intelligence and Bioinspired Systems*. Springer Berlin Heidelberg, 2005.

[75] Mircea Moscu, Ricardo A. Borsoi, Cédric Richard, and José-Carlos M. Bermudez. Graph topology inference with derivative-reproducing property in rkhs: Algorithm and convergence analysis. *IEEE Transactions on Signal and Information Processing over Networks*, 2022.

[76] T. Nguyen, T. Le, H. Bui, and D. Phung. Large-scale online kernel learning with random feature reparameterization. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*.

[77] G. Yehudai and O. Shamir. On the power and limitations of random features for understanding neural networks, 2020.

[78] R. Sato, M. Yamada, and H. Kashima. Random features strengthen graph neural networks, 2021.

[79] Basarbatu Can and Huseyin Ozkan. A neural network approach for online nonlinear neyman-pearson classification. *IEEE Access*, 8:210234–210250, 2020.

[80] Fatih Porikli and Huseyin Ozkan. Data driven frequency mapping for computationally scalable object detection. *2011 8th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 30–35, 2011.

[81] Huseyin Ozkan, N. Denizcan Vanli, and Suleyman S. Kozat. Online classification via self-organizing space partitioning. *IEEE Transactions on Signal Processing*, 2016.

[82] Gregory Ditzler, Manuel Roveri, Cesare Alippi, and Robi Polikar. Learning in nonstationary environments: A survey. *IEEE Computational Intelligence Magazine*, 2015.

[83] Lijun Zhang, Tianbao Yang, Rong Jin, and Zhi-Hua Zhou. Dynamic regret of strongly adaptive methods. *ICML*, 2018.

[84] Aryan Mokhtari, Shahin Shahrampour, Ali Jadbabaie, and Alejandro Ribeiro. Online optimization in dynamic environments: Improved regret rates for strongly convex problems. *2016 IEEE 55th Conference on Decision and Control (CDC)*, pages 7195–7201, 2016.

[85] Omar Besbes, Yonatan Gur, and Assaf Zeevi. Non-stationary stochastic optimization. *Operations Research*, 63(5):1227–1244, 2015.

[86] Ali Jadbabaie, Alexander Rakhlin, Shahin Shahrampour, and Karthik Sridharan. Online Optimization : Competing with Dynamic Comparators, 09–12 May 2015.

[87] Abhishek Chakraborty, Ketan Rajawat, and Alec Koppel. Sparse representations of positive functions via first- and second-order pseudo-mirror descent. *IEEE Transactions on Signal Processing*, 70:3148–3164, 2022.

[88] Amrit Singh Bedi, Alec Koppel, Ketan Rajawat, and Brian M. Sadler. Trading dynamic regret for model complexity in nonstationary nonparametric optimization. *2020 American Control Conference (ACC)*, pages 321–326, 2020.

[89] Daniele Calandriello, Alessandro Lazaric, and Michal Valko. Second-order kernel online convex optimization with adaptive sketching. *International Conference on Machine Learning*, 2017.

[90] Martin A. Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *International Conference on Machine Learning*, 2003.

[91] Mircea Moscu, Ricardo Borsoi, and Cédric Richard. Online kernel-based graph topology identification with partial-derivative-imposed sparsity. *2020 28th European Signal Processing Conference (EUSIPCO).*

[92] A. Shoeb. Application of machine learning to epileptic seizure onset detection and treatment. *PhD Thesis, Massachusetts Institute of Technology*, 2009.

[93] Edward N. Lorenz. Computational chaos-a prelude to computational instability. *Physica D: Nonlinear Phenomena*, 1989.

[94] Shangfei Song, Xuanzhang Liu, Chenxuan Li, Zhe Li, Shijia Zhang, Wei Wu, Bohui Shi, Qi Kang, Haihao Wu, and Jing Gong. Dynamic simulator for three-phase gravity separators in oil production facilities. *ACS Omega*, 8(6):6078–6089, 2023.

[95] Yuejing Hu, Qizhong Zhang, Rihui Li, Thomas Potter, and Yingchun Zhang. Graph-based brain network analysis in epilepsy: an EEG study. *2019 9th International IEEE/EMBS Conference on Neural Engineering (NER)*, 2019.

[96] F Pittau, F Fahoum, R Zelmann, F Dubeau, and J. Gotman. Negative bold response to interictal epileptic discharges in focal epilepsy. *Brain Topogr*, 2013.

[97] M. Manford, D. Fish, and S. Shorvon. An analysis of clinical seizure patterns and their localizing value in frontal and temporal lobe epilepsies. *Brain : a journal of neurology*, 1, 1996.

[98] N. Saadat and P. Hossein. Epileptic seizure onset detection algorithm using dynamic cascade feed-forward neural networks. In *2011 International Conference on Intelligent Computation and Bio-Medical Instrumentation*, pages 196–199, 2011.

[99] C.Granger. Investigating causal relations by econometric models and cross-spectral methods. *Econometrica, 37(3)*, page 424–438, 1969.

[100] Fallani F De Vico, J Richiardi, M Chavez, and S. Achard. Graph analysis of functional brain networks: practical issues in translational neuroscience. *Philos Trans R Soc Lond B Biol Sci.*, 2014.

[101] Xiaowen Dong, Dorina Thanou, Michael Rabbat, and Pascal Frossard. Learning graphs from data: A signal representation perspective. *IEEE Signal Processing Magazine*, 2019.

[102] C Giusti, R Ghrist, and D Bassett. Two's company, three (or more) is a simplex. *Journal of Computational Neuroscience*, 2016.

[103] Ailin Deng and Bryan Hooi. Graph neural network-based anomaly detection in multivariate time series. *arXiv*, 2021.

[104] Elvin Isufi, Andreas Loukas, Nathanaël Perraudin, and Geert Leus. Forecasting time series with varma recursions on graphs. *IEEE Transactions on Signal Processing*, 2019.

[105] Kai Qiu, Xianghui Mao, Xinyue Shen, Xiaohan Wang, Tiejian Li, and Yuantao Gu. Time-varying graph signal reconstruction. *IEEE Journal of Selected Topics in Signal Processing*, 2017.

[106] Jhony H. Giraldo, Arif Mahmood, Belmar Garcia-Garcia, Dorina Thanou, and Thierry Bouwmans. Reconstruction of time-varying graph signals via sobolev smoothness. *IEEE Transactions on Signal and Information Processing over Networks*, 2022.

[107] Michael T. Schaub and Santiago Segarra. Flow smoothing and denoising: Graph signal processing in the edge-space. In *2018 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, 2018.

[108] B Christian, G Elizabeth, H. A. Harrington, and M T. Schaub. What are higher-order networks? *arXiv*, 2021.

[109] M. T. Schaub, Y. Zhu, J. Seby, T. M. Roddenberry, and S. Segarra. Signal processing on higher-order networks: Livin' on the edge... and beyond. *Signal Processing*, 2021.

[110] Lewis A. Rossman, Robert M. Clark, and Walter M. Grayman. Modeling chlorine residuals in drinking water distribution systems. *Journal of Environmental Engineering.*

[111] G. Giannakis, Y. Shen, and G. Karanikolas. Topology identification and learning over graphs: Accounting for nonlinearities and dynamics. *Proceedings of the IEEE*, 106(5):787–807, 2018.

[112] Gonzalo Mateos, Santiago Segarra, Antonio Marques, and Alejandro Ribeiro. Connecting the dots: Identifying network structure via graph signal processing. *IEEE Signal Processing Magazine*, 2019.

[113] Antonio Marques, Santiago Segarra, and Gonzalo Mateos. Signal processing on directed graphs: The role of edge directionality when processing and learning from network data. *IEEE Signal Processing Magazine*, 2020.

[114] A. McLntosh and F. Gonzalez-Lima. Structural equation modeling and its application to network analysis in functional brain imaging. *Human Brain Mapping*, 1994.

[115] Maruf Maxim and Abu Ashif. A new method of measuring stock market manipulation through structural equation modeling (sem). *MPRA Paper 82891, University Library of Munich, Germany*, 2017.

[116] Yanning Shen, Xiao Fu, Georgios B. Giannakis, and Nicholas D. Sidiropoulos. Topology identification of directed graphs via joint diagonalization of correlation matrices. *IEEE Transactions on Signal and Information Processing over Networks*, 2020.

[117] Brian Baingana and Georgios B. Giannakis. Switched dynamic structural equation models for tracking social network topologies. *2015 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, 2015.

[118] Alberto Natali, Elvin Isufi, Mario Coutino, and Geert Leus. Online graph learning from time-varying structural equation models. In *2021 55th Asilomar Conference on Signals, Systems, and Computers*, 2021.

[119] Andrea Simonetto and Emiliano Dall'Anese. Prediction-correction algorithms for time-varying constrained optimization. *IEEE Transactions on Signal Processing*, 2017.

[120] Yanning Shen, Geert Leus, and Georgios Giannakis. Online graph-adaptive learning with scalability and privacy. *IEEE Transactions on Signal Processing*, 67(9):2471–2483, 2019.

[121] Karolis Martinkus, Andreas Loukas, Nathanaël Perraudin, and Roger Wattenhofer. Spectre: Spectral conditioning helps to overcome the expressivity limits of one-shot graph generators. *arxiv.org/abs/2204.01613*, 2022.

[122] Renjie Liao, Yujia Li, Yang Song, Shenlong Wang, William L. Hamilton, David Duvenaud, Raquel Urtasun, and Richard Zemel. Efficient graph generation with graph recurrent attention networks. *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, 2019.

[123] Matthew Jackson. *Social and economic networks.* Princeton University Press, 2008.

[124] Fredj Jawadi and William Barnett. Nonlinear modeling of economic and financial time-series. *International Symposia in Economic Theory and Econometrics*, 20, 2020.