ABSTRACT

Title of Dissertation: DEVELOPING MACHINE LEARNING TECHNIQUES
FOR NETWORK CONNECTIVITY INFERENCE
FROM TIME-SERIES DATA

Amitava Banerjee
Doctor of Philosophy, 2022

Dissertation Directed by: Professor Edward Ott
Department of Physics
Department of Electrical and Computer Engineering
Institute for Research in Electronics and Applied Physics

Inference of the connectivity structure of a network from the observed dynamics of the states of its nodes is a key issue in science, with wide-ranging applications such as determination of the synapses in nervous systems, mapping of interactions between genes and proteins in biochemical networks, distinguishing ecological relationships between different species in their habitats etc. In this thesis, we show that certain machine learning models, trained for the forecasting of experimental and synthetic time-series data from complex systems, can automatically learn the causal networks underlying such complex systems. Based on this observation, we develop new machine learning techniques for inference of causal interaction network connectivity structures underlying large, networked, noisy, complex dynamical systems, solely from the time-series of their nodal states.

In particular, our approach is to first train a type of machine learning architecture, known as

the ‘reservoir computer’, to mimic the measured dynamics of an unknown network. We then use the trained reservoir computer system as an *in silico* computational model of the unknown network to estimate how small changes in nodal states propagate in time across that network. Since small perturbations of network nodal states are expected to spread along the links of the network, the estimated propagation of nodal state perturbations reveal the connections of the unknown network. Our technique is noninvasive, but is motivated by the widely used invasive network inference method, whereby the temporal propagation of active perturbations applied to the network nodes are observed and employed to infer the network links (e.g., tracing the effects of knocking down multiple genes, one at a time, can be used infer gene regulatory networks).

We discuss how we can further apply this methodology to infer causal network structures underlying different time-series datasets and compare the inferred network with the ground truth whenever available. We shall demonstrate three practical applications of this network inference procedure in (1) inference of network link strengths from time-series data of coupled, noisy Lorenz oscillators, (2) inference of time-delayed feedback couplings in opto-electronic oscillator circuit networks designed the laboratory, and, (3) inference of the synaptic network from publicly-available calcium fluorescence time-series data of *C. elegans* neurons. In all examples, we also explain how experimental factors like noise level, sampling time, and measurement duration systematically affect causal inference from experimental data.

The results show that synchronization and strong correlation among the dynamics of different nodal states are, in general, detrimental for causal network inference. Features that break synchrony among the nodal states, e.g., coupling strength, network topology, dynamical noise, and heterogeneity of the parameters of individual nodes, help the network inference. In fact, we show in this thesis that, for parameter regimes where the network nodal states are not synchronized, we can often achieve perfect causal network inference from simulated and experimental time-series data, using machine learning techniques, in a wide variety of physical systems. In cases where effects like observational noise, large sampling time, or small sampling duration hinder such perfect network inference, we show that it is possible to utilize specially-designed surrogate time-series data for assigning statistical confidence to individual inferred network links.

Given the general applicability of our machine learning methodology in time-series prediction and network inference, we anticipate that such techniques can be used for better model-building, forecasting, and control of complex systems in nature and in the lab.

DEVELOPING MACHINE LEARNING TECHNIQUES FOR NETWORK CONNECTIVITY INFERENCE FROM TIME-SERIES DATA

by

Amitava Banerjee

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2022

Advisory Committee:
Professor Edward Ott, Chair/Advisor
Professor Rajarshi Roy
Professor Michelle Girvan
Professor Yanne K. Chembo
Professor Pratyush Tiwary

© Copyright by
Amitava Banerjee
2022

Preface

Causality in complex systems

This thesis is about inference of cause and effect relations from data. In complex systems where there are many interacting agents, such causal interactions form networks that ultimately dictate how the system as a whole behaves over time. Examples of such systems include neurons in our brain affecting firing patterns of one another, predators and preys in an ecosystems causing growth and decline of each other's population, genes in our cells turning each other on and off, and so on. A general theme of the study of networked dynamical systems has been identifying how these individual causal interactions between parts of such complex systems result in the collective dynamics of these systems as a whole (e.g., how do neurons in the visual cortex fire together when we see an image). An artist's impression of how the functioning of a multi-component system could depend on the cause and effect interactions among its parts is depicted in the drawing of an example of a scheme known as 'Rube Goldberg machines' in Fig. 1.

Since the functioning of a complex system depends crucially on the causal interactions among its parts, it is important to identify such interactions to construct mathematical models of the system, or for the purpose of predicting or controlling such systems. Regarding this, a natural question to ask is: given only the observed behavior of different parts of a complex system over time, is it possible to construct the network of cause and effect interactions that connects them? For example, looking at how neurons in the visual cortex fire over time, can we reconstruct how they are coupled via synaptic interactions? This brings us to the question of casual inference -

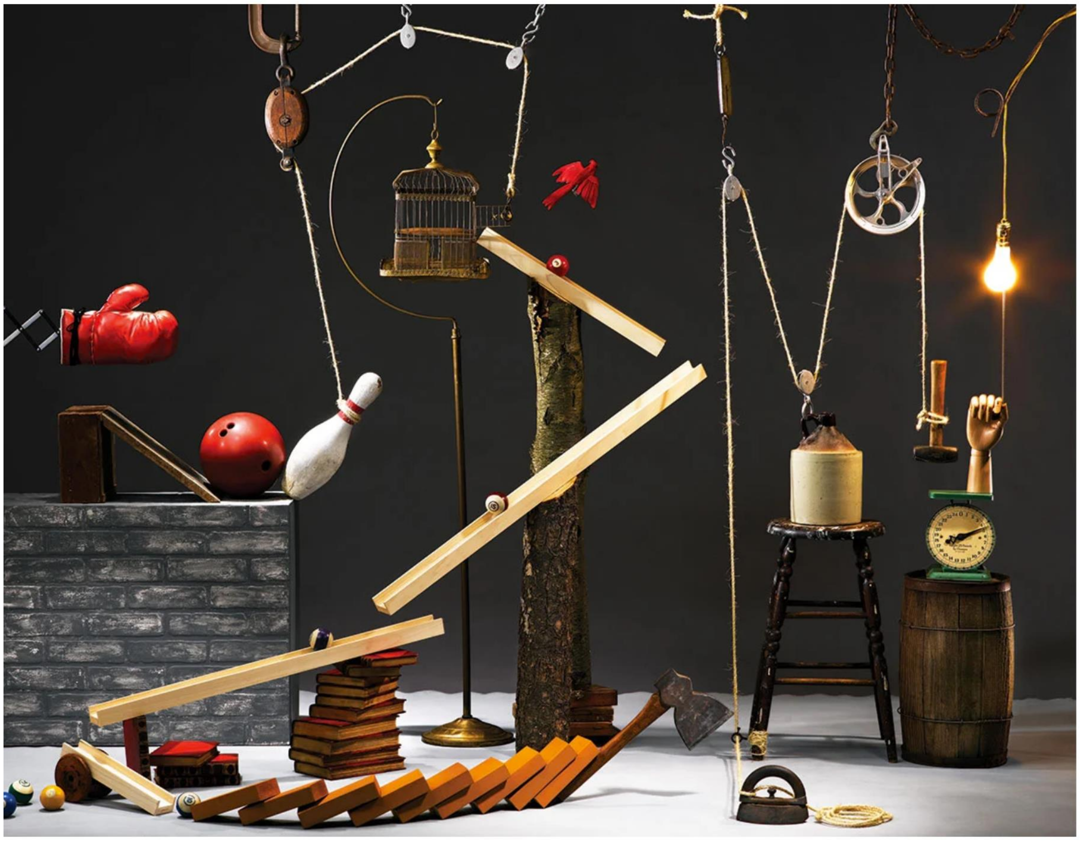

Figure 1: An example of a Rube Goldberg machine to light up the electric bulb shown in the right with the boxing glove and spring shown in the left, from Scout Life website.

connecting a set of observed events or dynamical agents with causal interactions. In some cases, such causal interactions among different agents can be simple and propagating linearly from one agent to another, without any feedback and with each agent being causally affected by only one other agent and causally affecting only one (a different) agent - like falling dominoes. An example is given by the machine in Fig. 1. There, cause and effect interactions flow along the following sequence, from left to right, as the extending of the gloved spring on the left eventually lights up the bulb on the right: boxing glove and spring, bowling ball, bowling pin, left string, birdcage, small falling ball, dominoes, axe, right string, hammer, weighing scale, model of human hand, light bulb. But, in more general cases, if there are many agents in a complex system, the cause and effect interactions among them constitute a directed network. An example is shown in Fig. 2, where the causal predator-prey interactions form a network among the Chesapeake Bay waterbirds. In such cases, identifying causal interactions amounts to inferring the directed network of cause and effect relationships among the agents of a complex system. Moreover, as we shall see later in this thesis, unlike the example network shown in Fig. 2, there could be bi-directional causal effects between different agents, adding to a further layer of complication in causal inference. In this thesis, we shall devise and test methods to infer these causal networks in various complex systems.

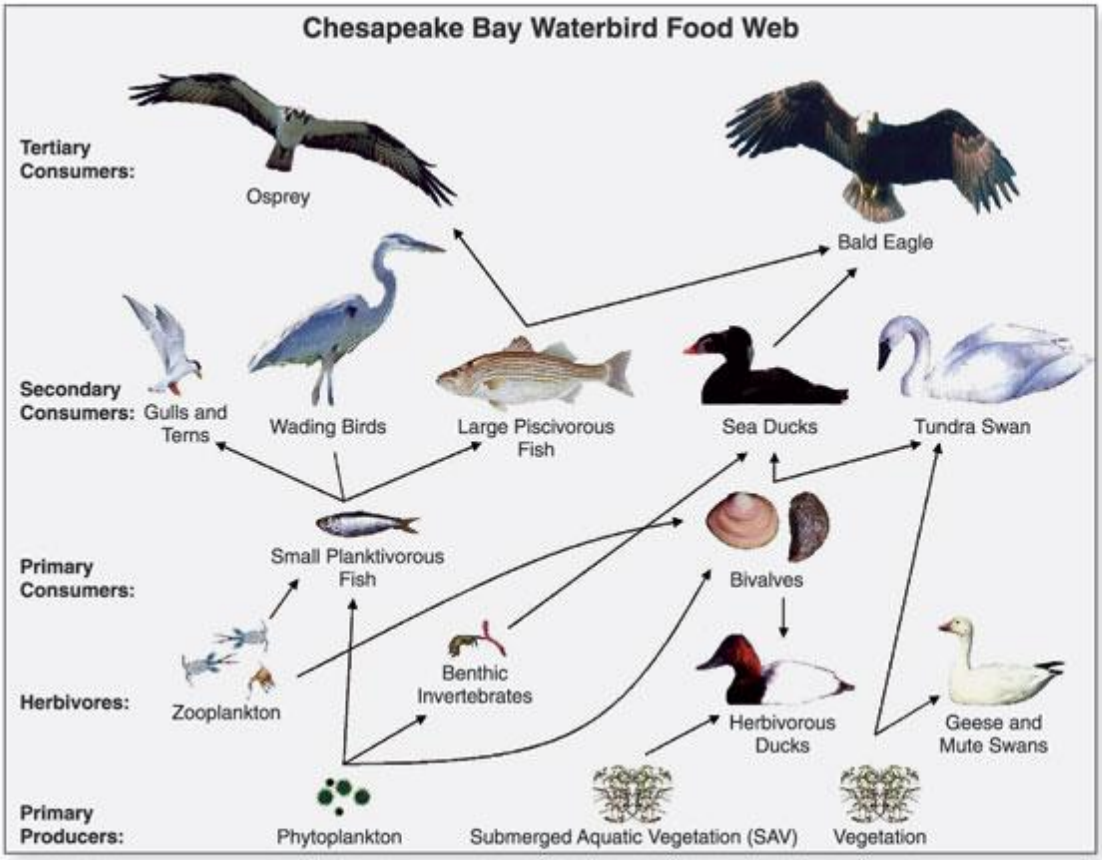

Figure 2: An example of a directed network of predator-prey interactions, from Wikipedia.

The ladder of causation

Since inference of causality in complex systems is of utmost importance in understanding their inner workings, it is imperative to think about the different levels at which the task of causal inference can be achieved. A formalization of this is found from 'The Book of Why' by Judea Pearl and Dana Mackenzie [1], where the authors classify causal inference to three levels - association ('seeing'), intervention ('doing'), and counterfactual ('imagining') - forming the 'ladder of causation', as shown in Fig. 3, since each of the levels build on its lower levels. To consider this in more details, let us use the example of baking butter cookies. Suppose we wish to understand whether adding butter causes the cookie to have a certain texture. How do we go about it?

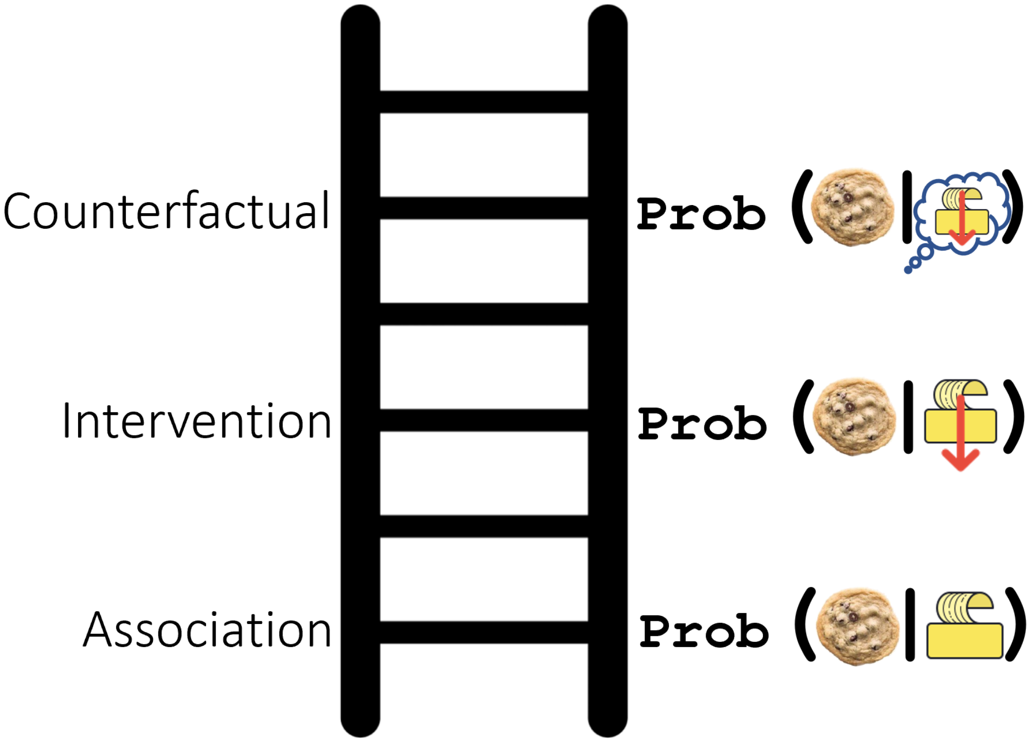

Figure 3: The ladder of causation showing three levels of abilities of a causal learner, for the example of baking a butter cookie.

To begin with, we need data. Suppose we have baked many butter cookies with different proportions of the baking ingredients and recorded the outcomes each time. The data relates the amount of butter in the cookies to different textures of the cookies, but this relationship is merely an association, or, ‘correlation’, in the sense of statistics. In the language of probability, it means to be able to calculate only the probability of a cookie to have a certain texture, given a certain amount of butter in it (Fig. 3). This does not tell us for sure if the amount of butter is causing an attribute of the texture of the cookie. So, this is the most basic information that we can extract from the situation, if we only have the data available. At this stage, which is the first rung of the ladder of causation, we can predict the texture of the cookie by the amount of butter in it - but we do not know for sure whether it was the amount of the butter that caused this texture. This is because we cannot answer questions like ‘what would the texture be like if the amount of butter was halved?’ simply by looking at the data, since the amount of other ingredients are also varying in the data. So, the change in texture when the amount of butter is changed may not actually be due to the butter but, say because of the differing amounts of flour and sugar. So, relationships based on simple associations can be misleading. How to know for sure that the butter is causing a particular aspect of the texture?

The way to deduce if it was really the amount of butter that caused a particular texture of the cookie is to do skillfully designed controlled experiments - baking many cookies with different amounts of butter with the amount of all other ingredients and baking conditions held fixed. This implies directly interacting with the system, using tools to actively intervene into it. Such controlled experiments enable us to calculate the probability of a cookie to have a certain texture, if the butter in it is changed by a certain amount (Fig. 3). This is the second rung of the ladder of causation, and it involves doing the experiments, like the ones that generated the data of rung one, in controlled way. As such, it assumes that we have an access into the system we wish to model.

The final rung of the ladder is imagining what would have happened if we changed the conditions in a certain baking experiment, but, without doing any more experiment. So, suppose we have done some experiments with different amounts of butter like in rung two, and maybe others involving changing other ingredients as well. Then, we ask ”what would have happened if we changed ingredient X in experiment number Y?” Unlike rung two, we cannot do an experiment again. So this requires an additional level of sophistication - the intuition of an experienced baker.

What does this imply for causal inference in complex systems? If we observe different parts of a complex systems over a period of time, we can record the changes of measured states of these different parts (e.g., membrane potentials or calcium fluorescence from individual neurons in the brain). The first thing to do with this data is to find correlations between states of different parts of the complex system. While we just argued that such correlations are the most primitive forms of causal modeling and may not necessarily imply causal interactions, we do obtain a wealth of information relating different parts of a complex system, which are useful particularly if we cannot make interventions into the system, and hence, cannot go up the ladder of causation easily. For example, correlations between activities of different parts of the human brain can be used to calculate functional connectivity, a measure that yields useful information, e.g., in studies on schizophrenia [2] and learning [3].

However, simple associations between compositions of a complex systems with its characteristics, without a mechanistic understanding of the underlying processes, has been shown to be dangerous in science. An example is linking a person’s genes to their behavioral traits. ‘Some companies see a market in reading DNA like a fortune-teller reads tea leaves’, remarks a Nature commentary [4], warning us against the consequences of misinterpreting these associations as causal effects.

Indeed, examples of misuse of associations abound in modern machine learning. It is challenging for artificial intelligence (AI) to work by identifying causal relations in the data, going beyond probabilistic associations that most of them are trained on [5]. As consequences, mainstream AI still struggles to transfer learned skills to new environments, requires data-heavy training, and often produces opaque machines reproducing and reinforcing non-causal associations and biases in training data. ‘Widespread, mysterious, and destructive’ [6], Cathy O’Neil labels the algorithms affecting our lives, for example, by performing facial recognition in surveillance and calculating everything from credit scores, college rankings, to the amount of prison sentences. Opaque AI algorithms learning by identifying non-causal associations have been shown to, e.g., misclassify gender of a face depending on race [7], produce language models showing stereotypical association and negative sentiments towards specific groups [8], misclassify objects to be identified for steering angle prediction in self-driving cars [9] and produce false positive melanoma recognition depending on spurious surgical skin markings in medical images [10]. Thus, a general goal in machine learning is to produce AI algorithms that learn not by identifying statistical associations, but by identifying causal relationships in the training data. In order to do so, we need a way to lift the AI algorithms to one rung up the ladder of causation. This thesis is about machine learning architectures that are shown to be able to learn by identifying causal interactions in complex systems.

Taking machines up the ladder of causation

What does it require for a causal learner to go one level up the ladder of causation? It is the ability to do specifically designed controlled experiments. In the case of genetics experiments, the approach of ‘perturbation biology’ [11] involves knocking down genes and observing downstream effects of such knockdowns on other genes to infer causal gene interaction networks. However, while this example allows experimental access to the system under study, such might not be true in more general cases. For example, in the case of ecological predator-prey relationships, it is not desirable, in general, to have any human interventions to the population dynamics of different species. Considering such cases, trained machine learning algorithms should be able to accurately predict the effects of interventions, if they are to be placed on the second rung of the ladder of causation. Is there a way that it can be achieved?

In the case of complex dynamical systems with many components connected by a network of causal interactions, we show in this thesis that it is possible to have machine learning architectures trained only on the time-series data from the components of a complex system, that can not only fit to the training data (i.e., learn to model associations), but can also predict the causal interactions among the components. In particular, our trained machines would be shown to accurately answer questions like "would component X of a complex system change at time t' as a result of a change in component Y at time t, with $t' > t$"? In many cases, a correct answer to these questions would mean inference of the causal network underlying the complex system. In our examples in the following chapters, we shall test the goodness of causal network inference by comparing the inferred causal network to the ground truth causal network, with the latter being known in all of our examples. The results will show that there are hopes of lifting mostly associational AI algorithms of today one step up the ladder of causation, where they would be able to predict the effect of interventions and different controlled experiments on the systems they would be trained on.

Dedication

This thesis is dedicated to my partner, Ms. Promita Ghosh, doctoral candidate at the Rosalind and Morris Goodman Cancer Institute, McGill University, for her constant support, advice, and inspiration.

Acknowledgments

I shall start with the Land Acknowledgement Statement. The following is copied from the UMD Office of Diversity and Inclusion website.

> **Every community owes its existence and strength to the generations before them, around the world, who contributed their hopes, dreams, and energy into making the history that led to this moment.**
>
> **Truth and acknowledgement are critical in building mutual respect and connections across all barriers of heritage and difference.**
>
> **So, we acknowledge the truth that is often buried: We are on the ancestral lands of the Piscataway People, who are the ancestral stewards of this sacred land. It is their historical responsibility to advocate for the four-legged, the winged, those that crawl and those that swim. They remind us that clean air and pristine waterways are essential to all life.**
>
> **This Land Acknowledgement is a vocal reminder for each of us as two-leggeds to ensure our physical environment is in better condition than what we inherited, for the health and prosperity of future generations.**

আমার প্রভুর পায়ের তলে   শুধুই কি রে মানিক জ্বলে ।
চরণে তাঁর লুটিয়ে কাঁদে লক্ষ মাটির ঢেলা রে ॥
  আমার গুরুর আসন-কাছে   সুবোধ ছেলে ক'জন আছে ।
  অবোধ জনে কোল দিয়েছেন, তাই আমি তাঁর চেলা রে ।

রবীন্দ্রনাথ ঠাকুর

Where my God dwells   do you only see the precious jewels?
For I see the dirt weep there in abundance.
  Where my Gurus take the seat   how few good ones they care to greet?
  For the naive get a home there, I am a follower hence.

Rabindranath Tagore
(English translation mine)

The graduate school life at UMD Physics taught me that gratefulness is something to be practiced mindfully. Looking back four years into the past, I realize how difficult it is for me to acknowledge everyone who helped me survive, thrive, and learn during my grad school. So, I am beginning with acknowledging my own limitation and apologizing for my ungratefulness to anyone not mentioned here. I plan to host an updated version of this section in my personal website: https://banerjeeamitava.github.io/ under the section called ‘Writings’.

These years have not been easy at all. The only easy part was probably running the computer codes to produce the results that make the bulk of this thesis. I was never able to go back to my country, India, during my entire grad school, even when my friends died before the COVID vaccine could be distributed. As if the pandemic was not enough, within four years of setting foot in this country, I saw political violence, murderous police officers, free and cheap racialized hatred, discrimination, anti-LGBTQ legislation, malfunctioning courts, mass shootings, and decay and deliberate dismantling of human rights and dignity. If that was not enough, we had to see a full-scale war happening in some of my friends' homeland, and countless other deadly conflicts killing people all over the world. Amongst all of this, I have been able to survive, thanks to some of the most wonderful people I have been privileged to have met, who made me feel loved and blessed.

Since a lot of time during my PhD was spent at home due to the pandemic, I spent a considerable amount of time with my housemates, and they are partly the reason I was able to get through. So I am thankful to my current and former housemates: Abu Saleh Musa Patoary, for cooking N number of dishes with (/for) me and traveling, drinking, and watching Netflix with me among doing many other stuff, including getting the the defense champagne; Gautam Nambiar, for his always-positive presence, numerous Malyali treats, getting the defense cake, and for the drive to the COVID vaccine center; Elizabeth Joy Paul, Arthur Lin, and Wonseok Hwang for numerous drives to the grocery stores, helping with moving, and for teaching me about etiquette; and Wrick Sengupta, Sagar Airen, Amit Vikram Anand, and Aritra Das for great conversations, among many other memories that are making me smile now. Speaking of my friends, I must also thank Sourav Das and Sohitri Ghosh for numerous Bengali treats and all the great conversations; Subhayan Sahu and Anshuman Swain for being my constant and robust supports; and Naren Manjunath, Prakhar Gupta, and Xiaozhen Fu for numerous dinner invites to their place.

In the Summer of 2019 I went for a hike to the Old Rag mountains in the Shenandoah National Park. This difficult hike was the first hike ever in my life, and I realized that it symbolized my PhD journey. I am thankful to Joseph Hart who literally lifted me up and carried my burden during the Old Rag hike and accompanied me in the hike to make sure I do not get stuck. I am also thankful to be in the presence of Ian Abel, William Dorland and Sarah Penniston-Dorland, Adelle Wright, Arinjoy De, and Rupak Mukherjee in the hikes.

I am grateful to have served in and served by several DEI and planning committees, and I got to know and learn from some of the best human beings that life has given me: Allison Carter, Mika Chmielewski, Kathleen Hamilton-Campos, Natalia Pankratova, and others from the Women in Physics; Daniel Serrano, Thomas Murphy, Kevin Daniels, Roxanne Defendini, Marc Swisdak, Heidi Myers, Taylor Prendargast, Sai Kanth Dacha, Carlos Rios Ocampo, and Meredith Pettit from the IREAP-ROLE; Milena Crnogorčević, Charlotte Ward, Andrew Guo, Katya Leidig, Arjun Savel, Stuart Vogel, and others from the GRAD-MAP; Madison Anderson, Ananya Sitaram, Landry Horimbere, Edgar Perez, Donna Hammer, and others from the Physics Climate Committee; Jennifer Enriquez, Justine Suegay, Shabab Ahmed Mirza, and others from MICA; Luanjiao Aggie Hu, Fatima Ali, and others from the GSL; Brian Medina from the BISS; Amanda Vu, Jackie Liu, Alythia Vo and many others from the APIDA Heritage Month Planning Committee, and many more. These people have been the best teachers in my grad school. They have convinced me that I am worthy of love and care and also capable of receiving them.

I am also thankful to be served by the Physics GRAD-COMM and Mental Health Task Force, the Office of Diversity and Inclusion, and the LGBTQ+ Equity Centre. I am grateful to Physics, IREAP, and IPST staff and coordinators Josiland Chambers, Jessica Crosby, Pauline Rirksopa, Paulina Alejandro, Judy Gorski, Dorothea Brosius, Catha Stewart, and many others. They have been the lifesavers and made me feel home here. Also thanks to Leslie Miller from the the CMNS Office who made me feel special by writing a feature story on my research.

I am also indebted to the cleaning staff, the nurses who gave me covid vaccines, and the numerous Uber drivers who have delivered food to me and taken me to places. I am particularly grateful to the Uber driver Michael who was exceptionally kind to have drove me through the Six

Flag vaccination site when I had no idea that the site was drive-thru-only beforehand.

I am thankful to people whom I travelled with to the International Physics of Living Systems conference in Montpellier this summer. These include Vishavdeep Vashisht, Spandan Pathak, Nathan Zimmerberg, Elissa Moller, Frank Fazekas, and others and, for leading and planning the trip, Arpita Upadhyaya.

I am grateful to my mentees, Neil Shah, Waley Wang, and Paulson Obiniyi Jr., who have showed me that I am capable of giving research and life advice that are useful. I am also thankful to all the amazing GRAD-MAP mentees whom I was lucky to have served, including Autumn Jackson Bartholomew, who was very kind to come to my thesis and helped me with the arrangements.

I am thankful to my research collaborators: Jaideep Pathak, Sarthak Chandra, Juan Restrepo, William Fagan, Alireza Seif, Mohammad Hafezi, Alexey Gorshkov, Ravi Chepuri, and others, for everything I learned from them.

I am thankful to my dissertation committee members Michelle Girvan, Pratyush Tiwary and Yanne Chembo for the great feedback and encouragements. I am grateful to all the people people who came to my defense talk, congratulated me on my graduation and celebrated with me.

I am grateful to my partner Promita Ghosh for being a constant, loving, and supporting figure who was kind enough to be a safe haven whenever my mental health faltered. She proves that brilliant scientists can also be good human beings.

I am extremely lucky to have Rajarshi Roy as my major collaborator and advisor. They say you need a lot of mentors for different things in grad school: one mentor to guide your research, one to help you network, one to help you look for jobs, one to go walk with, one who helps you when you get into bad troubles, one to vent out to during your sad days, one to shed tears to, and one who prays for you silently, and so on. I had Raj.

Finally, I am thankful to my doctoral thesis advisor Edward Ott for being so kind as to accept me as his student and providing me with his experienced guidance throughout my PhD journey. Needless to say, the thesis would not have been done, had he not abundantly supported me and my work, in every way possible, though everything that happened.

Table of Contents

Preface ii

Dedication x

Acknowledgements xi

Table of Contents xvii

List of Figures xx

List of Abbreviations and Symbols xxi

Chapter 1: Introduction 1
- 1.1 A network science view of complex dynamical systems . . . 1
- 1.2 Network inference techniques and applications . . . 2
  - 1.2.1 Model free network inference techniques . . . 4
  - 1.2.2 Model-based network inference techniques . . . 5
- 1.3 Network inference with reservoir computing . . . 6
  - 1.3.1 Properties of computational models for network inference . . . 6
  - 1.3.2 Basic structure of a reservoir computer . . . 8
  - 1.3.3 Reservoir computer models of time-series data and network inference . . 11
- 1.4 Organization of this thesis . . . 14
  - 1.4.1 RC inference of short-term causal dependence . . . 15
  - 1.4.2 RC inference of time-delayed causal dependence . . . 15
  - 1.4.3 Effects of sampling conditions on RC causal inference and statistical analysis 16

Chapter 2: Inference of short term causal interactions: applications in coupled Lorenz oscillators 18
- 2.1 Overview . . . 18
- 2.2 Introduction . . . 19
- 2.3 Short Term Causal Dependence (STCD) . . . 21
- 2.4 Using Reservoir Computer to Determine Short Term Causal Dependence . . . 25
- 2.5 Tests of Machine Learning Inference of STCD . . . 29
- 2.6 Discussion . . . 37
- 2.7 Acknowledgements . . . 38

Chapter 3: Inference of time-delayed causal interactions: applications in optoelectronic oscillator networks 39
3.1 Overview 39
3.2 Introduction 41
3.3 Reservoir Computing Methodology for Network Inference 44
3.3.1 The General Delay-coupled Network 44
3.3.2 Time Series Prediction with a Reservoir Computer 46
3.3.3 Our Network Inference Procedure 49
3.4 Opto-electronic Oscillator Networks 54
3.4.1 Description of the experiment 54
3.4.2 Mathematical Model and Numerical Simulations of the Opto-Electronic Network 57
3.5 Results of Link Inference Tests 61
3.5.1 Performance on Simulated Data - Homogeneous Delays 62
3.5.2 Performance on Experimental Data 69
3.5.3 Performance on Simulated Data - Heterogeneous Delays 70
3.6 Discussion 71
3.7 Acknowledgements 74
3.8 Appendices 74
3.8.1 Determination of the time-delay from cross-correlation 74
3.8.2 Derivation of the discrete-time equation for simulating the opto-electronic system 76

Chapter 4: Network inference application for *C. elegans* neuronal calcium fluorescence dynamics 80
4.1 Overview 80
4.2 Introduction 81
4.3 Physical systems and datasets used 86
4.3.1 Calcium imaging time-series data from *C. elegans* 86
4.3.2 Coupled Lorenz oscillator network simulations 87
4.4 Link-score distributions for the coupled Lorenz Oscillator Network 88
4.5 Statistical analysis of candidate link scores for link inference 93
4.5.1 Brief review of surrogate data generation techniques for network link inference 93
4.5.2 Method of surrogate time-series generation and statistical significance analysis of inferred connections 94
4.6 Tests on Example Systems 95
4.6.1 Application to Lorenz network model 95
4.6.2 Applications to *C. elegans* data 98
4.6.3 Transfer entropy and Granger causality 100
4.6.4 Another Surrogate Data Generation Technique 101
4.7 Discussion 102
4.8 Methods 105
4.8.1 Reservoir-computer-based connection inference technique 105
4.8.2 Transfer Entropy 109

4.8.3 Granger Causality 109
4.9 Acknowledgements 110

Chapter 5: Summary 111

Bibliography 113

Bibliography 113

Curriculum vitae 127

List of Figures

1 Schematic of a Rube Goldberg machine iii
2 The food web of waterbirds from Chesapeake Bay iv
3 The ladder of causation v

1.1 Schematics of a reservoir computer 8
1.2 Properties of time-series prediction by reservoir computers 12

2.1 Schematic of the reservoir computing architecture 26
2.2 Performance of link inference for noiseless Lorenz oscillator networks 33
2.3 Effect of dynamical and observational noise on network link inference 35
2.4 Network link strength inference for arbitrary networks 37

3.1 Schematics of a reservoir computer for time-delayed prediction 46
3.2 Illustration of opto-electronic oscillator and coupling scheme 55
3.3 Examples of experimental and simulated time series from two opto-electronic oscillator networks 59
3.4 List of possible connected directed 4-node networks 60
3.5 Synchronization error for simulated time series of 4-node networks 62
3.6 Network inference performance with varying noise strength 63
3.7 Network inference performance with varying coupling strength 65
3.8 Distribution of individual link scores 68
3.9 Performance of network inference method on experimental data 69
3.10 Inference of networks with heterogeneous link delays 72
3.11 Time delay determination by cross-correlation of measured time series 75

4.1 *C. elegans* neural connectome inference from calcium fluorescence time-series . . 82
4.2 Separable and non-separable network link inference score distributions 83
4.3 Dependence of network link inference score distributions on sampling limitations 89
4.4 score distributions from different link inference methods 99
4.5 Score distribution and surrogate data for AAFT surrogate 103
4.6 Schematic of the reservoir computer 107

List of Abbreviations and Symbols

AAFT	Amplitude-Adjusted Fourier Transform
AI	Artificial Intelligence
CDSD	Causality-Destroyed Surrogate Data
ML	Machine Learning
OEO	Opto-Electronic Oscillator
RC	Reservoir Computer
STCD	Short Term Causal Dependence
W_{in}	Input-to-Reservoir Coupling Matrix
W_{out}	Reservoir-to-Output Coupling Matrix

Chapter 1: Introduction

In this thesis, we infer causal interactions in complex dynamical systems from machine learning models of their time-series data. This chapter serves as a background material for the rest of the thesis. It introduces both the research questions that we shall address in the rest of this thesis, and the main framework that we shall address them within.

We start by explaining with examples why an understanding of the network of causal interactions underlying a complex system is essential for modeling and prediction of complex systems. Then we discuss several traditional and contemporary methods developed for such purpose. Finally, we shall introduce a new machine-learning-based technique for inference of causal interaction network structure underlying complex dynamical systems, that we shall employ in the next chapters.

1.1 A network science view of complex dynamical systems

The science of agents, objects, or systems connected by graphs - network science - has flourished in the last two decades and provided us with a wealth of useful analytical and computational techniques [12, 13] to describe and study complex systems in nature and society. Examples of networked systems range from neuronal networks [14] in the brains of human and other organisms, and gene regulation and protein interaction networks inside cells [15], to the internet and telecommunication systems. Network science has provided us with a 'systems' viewpoint of natural and social processes, that has been instrumental in their understanding and control. A particularly important subfield of network science deals with collective properties of complex dynamical systems connected via networks. Applications of such studies include epidemic spreading [16], synchronization of dynamics [17], and growth of political polarization [18] on complex networks, to mention a few. As such, networked complex systems are platforms of studying a wide variety of dynamical phenomena arising from the interactions between network topology and time-evolution of agents, acting as the network nodes, connected by the network links.

1.2 Network inference techniques and applications

While studying the collective properties and dynamics of networked agents has been interesting, the inverse problem, i.e., inference of network links from the observed dynamics of the network nodal states, is also an important and challenging problem in itself. Applications of network inference techniques include determination of the synaptic connectivity in nervous systems [2, 19, 20], mapping of interactions between genes [21] and proteins in biochemical networks [22], distinguishing relationships between species in ecological networks [23, 24], and understanding the causal dependencies between elements of the global climate [25]. In all such cases, a basic network structure among the agents of these complex systems (e.g., individual neurons, genes, proteins, and species) can be found by measuring correlations among the time-dependent states of the agents. Examples include functional connectivity in neuroscience which measures the correlation of neural activity among different parts of the brain [2, 14], gene co-expression networks that link genes which are statistically expressed together [26], co-occurrence/co-exclusion networks of microbes [27] that map which microbes are statistically present or absent together in the human microbiome etc. Such networks based on correlation, co-occurrence or associations among different agents have been useful in, for example, relating neural activity to neurodegenerative diseases [28, 29] and other conditions [2, 14] and relating changes in gut microbial co-occurrence patterns to inflammatory bowel disease [30]. However, despite the usefulness of correlation-based network models of complex systems, in the absence of causal models, they are generally unable to predict how the complex systems would respond to external perturbations. For example, it is impossible to predict how knocking down of a specific gene would affect the expression of other genes and stimulating a specific neuron would affect the firing pattern of other neurons, if the causal, directed interaction networks among the genes and neurons are not known. So, therapeutic interventions into these systems will not be possible without a thorough causal understanding of these systems. Thus, a broad goal of this thesis is to extract the causal network connecting different components of a complex system, solely from the time evolution of such components.

Given the usefulness of causal network inference, a number of methods have been proposed for such purpose. We shall now briefly review some prominent network inference methods. It is helpful to classify them into two types: (1) model-free, i.e., network inference techniques that do not assume any particular mathematical model of the observed dynamics, and, (2) model-based, i.e., network inference techniques which assume the dynamics of the system to be governed by a particular class of mathematical models (e.g., a set of linear coupled ordinary differential equations) with parameters which are fitted using the available time-series data. We now describe some examples of the two types of network inference methods. For connectivity inference from neuronal recordings, a review of different network inference methods and their comparisons can be found in Ref. [19].

1.2.1 Model free network inference techniques

1.2.1.1 Transfer entropy

Transfer entropy is an information theoretic causal network inference technique, which measures the rate of transfer of information from one time-series to another [31]. For a pair of agents (i, j) with dynamical states in a complex system, transfer entropy from i to j is defined by the conditional mutual information between the future states of j and past states of i conditioned by past states of j. It represents the amount by which the uncertainty in the future dynamics of j is reduced by knowing the past dynamics of i conditioned on the past dynamics of j. Transfer entropy is usually calculated pairwise, and can be used to construct a weighted, directed network connecting the agents of a complex system based on the directional information flow between them. Several generalizations and variations of transfer entropy also exist [19], to account for experimental limitations to acquisition of time-series data.

1.2.1.2 Supervised learning of causal interactions

Model-free techniques of estimating causal interaction networks are rare because effects of causal interactions between dynamical agents can potentially affect their dynamics in multiple, unknown ways. In cases where there are available examples of dynamics with known ground truth causal connectivity, one can use such examples to train machine learning algorithm to detect the presence of causal interactions from dynamics [19, 32]. However, such techniques require data-heavy training and availability of ground truth causal interaction networks which are often unknown. Furthermore, in many cases, such models lack interpretability if it is unclear what features of a complex system's dynamics are being used to predict the causal interactions underlying the dynamics.

1.2.2 Model-based network inference techniques

In model-based techniques the causal connectivity is estimated by mathematically modeling the processes that generates the sampled time-series data, and fitting model parameters to the observed data. Some examples of such models are the following.

1.2.2.1 Granger causality

For a pair of agents (i, j) with dynamical states in a complex system, we say that j does not Granger cause i if and only if i, conditional on its own past dynamics, is independent of the past dynamics of j [33–36]. Alternatively, if we can have information on the past dynamics of i from the past dynamics of j, then j would Granger cause i. The typical way to test this type of dependency between two time-series involves fitting a vector autoregressive model for the time-series for i, and measuring whether inclusion of the time-series of j in that model makes the fitting error significantly lower. In such generative time-series models applied to test the predictive powers of different time-series, one can simultaneously consider more than two time-series at once, and can simultaneously include multiple time-delays in the generative model. However, these models are linear, and stationary in nature whereas the original dynamics might be nonlinear and non-stationary [37, 38]. Furthermore, the presence of unmeasured external inputs to the dynamics, noise, and finite sampling rate is also known to affect Granger causality [37, 38].

1.2.2.2 Bayesian network model

Bayesian networks model the conditional probability distributions of time-series from different agents with a directed, acyclic graph [19, 24, 39] with parameters corresponding to, e.g., causal connectivity strengths. These parameters of the model are optimized to fit to the observed data and assuming a particular form of the underlying probability distribution (e.g., Gaussian) of the time-series variables from different agents [39].

1.3 Network inference with reservoir computing

1.3.1 Properties of computational models for network inference

In the preceding section, we saw some examples of popular techniques for causal connectivity inference from time-series of complex systems. While model-free methods like transfer entropy can capture causal connectivity by measuring information flow between different dynamical agents of a complex system, they do not provide any generative model for the sampled time-series data. In this thesis, we wish to extract causal interaction network connections from a model of the observed time-series, that can accurately fit to and generalize the measured data. Thus, we aim for a mathematical or computational model of the observed time-series from the complex system that has the following three properties.

(i) Accurate fit to the sampled data: The model is sufficiently complex, potentially containing a large number of adjustable parameter so that it can fit to the sampled time-series data accurately.

(ii) Prediction or generalization of the sampled data: The model is able to predict the time-series of the complex system for some time duration in the future, or, at least be able to generate time-series data which is statistically similar (e.g., has similar power spectrum) to the original sampled time-series data.

(iii) Causal inference extraction from observed data: The model should be able to predict the effect of small changes to the states of individual agents on the states of other agents at a later time in the complex system. That is, it should have the capability to extract causal connectivity structure among the dynamical states of different agents, using only the observed time-series data that it has been fitted on.

In order to faithfully infer causal network connectivity from a model of time-series data, all three of the above properties are essential. Since causal interactions can affect the dynamics of individual agents in many ways, we wish to make sure that our model of the dynamics is sufficiently complex to capture all of them, so we wish our model to fit accurately to the observed time-series data. However, goodness of this fit alone does not guarantee that the model is indeed capturing the ground truth dynamical processes of the actual system, as the model could overfit to the available data. In order to avoid this overfitting, we ensure that the model is indeed able to autonomously predict the future time evolution of the actual system, or generate time-series samples which are statistically similar to the observed time-series of the complex system. Furthermore, to ensure that the model is able to fit to and generalize the observed time-series not by identifying simple statistical association between the time-series of different agents, we further require that the model must be able to predict the effect of interventions to the time-series of the different agents. In this thesis, we shall show that it is possible to obtain a model that satisfies the above three properties, using a specialized machine learning architecture called the 'reservoir computer' (RC). In the next section, we describe the structure of a RC in more details.

1.3.2 Basic structure of a reservoir computer

In its most basic form, the reservoir computer is a machine learning architecture that takes in multiple time-series at its input and produces multiple time-series at its output (Fig. 1.1). We denote the time-series that has been sampled from the complex system by the time-dependent vector $\mathbf{u}(t)$, of dimension D_u, with its components denoting the time-series corresponding to the states of individual agents of the system (note that, if the states of the agents are given by vectors themselves, then multiple components of the time-series vector $\mathbf{u}(t)$ may be sampled from the same agent). If the sampling time is δt, we have this time series vector measured at times $t = 0, \delta t, 2\delta t, ..., T\delta t$. Based on the properties of causal models from the previous section, we wish to obtain a computational model of the observed time-series yielding a corresponding time-series $\tilde{\mathbf{u}}(t)$ of the dynamics that (i) fits to the observed time-series data $\mathbf{u}(t)$ (which we shall call the 'training data'), (ii) can generate time-series data $\tilde{\mathbf{u}}(t)$ with similar statistical properties as $\mathbf{u}(t)$, and (iii) can predict the input data for a time duration (say, τ) into the future, i.e., given $\mathbf{u}(t)$, the model can predict $\tilde{\mathbf{u}}(t+\tau)$ at its output. Fig. 1.1 shows how this is done with an RC.

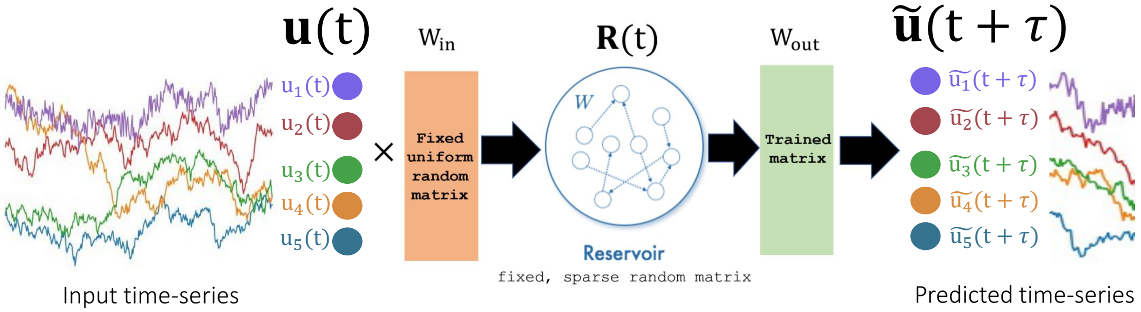

Figure 1.1: Schematics of a reservoir computer predicting a five-variable time-series $\mathbf{u}(t)$ a time τ ahead into the future.

As depicted in Fig. 1.1, the RC is a network consisting of a large number of nodes. The number of nodes, D_r, is chosen such that $D_r >> D_u$. Each of the nodes has a time-dependent scalar state, all of which are collected in a time-dependent column vector $\mathbf{R}(t)$ of length D_r. These reservoir nodes receive measured inputs of the system state vector $\mathbf{u}(t)$. At every time-point t, this system state vector $\mathbf{u}(t)$ is fed into the reservoir via a $D_r \times D_u$ input-to-reservoir coupling matrix $\mathbf{W}_{\text{in}}$ (Fig. 1.1). Furthermore, the reservoir nodes are coupled among themselves with a $D_r \times D_r$ adjacency matrix $\mathbf{W}$. The time evolution of the reservoir node states $\mathbf{R}$ over a small time step Δt are given by the equation,

$$\mathbf{R}\left(t + \Delta t\right) = \sigma\left(\mathbf{W}_{\text{in}}\mathbf{u}(t), \mathbf{W}\mathbf{R}(t)\right), \tag{1.1}$$

which maps the reservoir state at time t to the reservoir state at the next time step, $t + \Delta t$, where σ is a general nonlinear 'activation' function of the input and reservoir states, acting componentwise on its vector argument (which must the same dimension, D_r, as the reservoir state $\mathbf{R}$). For simplicity, in this instance we can assume that $\delta t = \Delta t$, even though in more general cases, it suffices that the reservoir time-evolution iteration time-step Δt is an integer multiple of the sampling time-step δt of the system. Based on the available time series from the system $\mathbf{u}(t)$ for T time steps, Eq. 1.1 can be iterated to generate a corresponding time series for the reservoir state, $\mathbf{R}(t)$.

Since the goal is to predict the future values of the sampled time-series $\mathbf{u}(t + \tau)$ from their current observed values $\mathbf{u}(t)$, we may utilize the reservoir state vectors $\mathbf{R}(t)$ for this purpose. In our case, this is done by using a suitable linear combination of the reservoir node states with a $D_u \times D_r$ reservoir-to-output coupling matrix $\mathbf{W}_{\text{out}}$ (Fig. 1.1) according to the equation,

$$\tilde{\mathbf{u}}(t) = \mathbf{W}_{\text{out}}\mathbf{R}(t), \tag{1.2}$$

where the tilde in $\tilde{\mathbf{u}}$ indicates that the vector is a prediction from the RC, as opposed to being sampled from the actual system. Given the form of Eq. 1.2, the matrix $\mathbf{W}_{\text{out}}$ can be found by a linear regression between the vectors $\mathbf{u}(t)$ and $\mathbf{R}(t)$. This linear regression constitutes the training process of the RC, and the matrix $\mathbf{W}_{\text{out}}$ is the only element of the RC with adjustable parameters - the other matrices, like $\mathbf{W}_{\text{in}}$ and $\mathbf{W}$, are chosen randomly at the beginning of iteration of eq. 1.1, kept fixed throughout. Since the training constitutes of a linear regression, RCs can be trained much faster than conventional deep learning neural networks which contain many adjustable parameters which have to be trained by gradient descent. A comparison of the training time of RC and other machine learning architectures is done in Ref. [40].

While the subsequent chapters in this thesis will contain more details about the form of the nonlinear function σ, as well as the structure of the fixed matrices $\mathbf{W}_{\text{in}}$ and $\mathbf{W}$, we note here the generality of RCs. Indeed Eq. 1.1 shows that any dynamical system with a sufficiently large number of 'nodes' or computing units, having some nonlinearity, and a mechanism of receiving inputs and providing readable outputs can serve as a potential RC. Indeed, as reviews on RC realizations with physical systems demonstrate [41], RCs can be experimentally realized with many different systems irrespective of the microscopic details of their dynamics. Such physical implementation of the reservoir computer [41–43], include photonic [44–46], electronic and opto-electronic [47, 48], neuronal [49], molecular [50], chemical [51], spin [52] systems, or even a tank of water [53]. This flexibility of physical realization makes RCs a prime candidate for 'neuromorphic computing' [43], an unconventional computing paradigm that seeks inspiration from neural systems to create novel computing architectures.

1.3.3 Reservoir computer models of time-series data and network inference

RCs have been used for a variety of applications in time-series analysis, including forecasting of spatially extended [54] and networked [55] chaotic systems, data-based reconstruction of long-term statistical properties of dynamics like the Lyapunov spectrum [56, 57], separation of mixed signals from two sources [53, 58], and fast classification of spoken words [44], to mention a few. Among these applications, the first one is the most relevant for our purpose of network inference. In terms of properties (i) and (ii) from Sec. 1.3.1, RCs with a moderate size ($D_r \approx 1000$) are known to fit to the training data of size $D_u \approx 10$ accurately. Furthermore, as Fig. 1.2(a) shows, it is possible to use trained RCs for autonomous long-term time-series forecasting by running them in a closed loop configuration where the output time-series from the RC $\tilde{\mathbf{u}}(t)$ at a given time-step t is fed back as the input to the RC, thus eliminating the need to have any inputs from the original time-series data. As shown in the example of Fig. 1.2(b), such predictions remain accurate for a long time, even for chaotic systems. It is to be noted here that, for chaotic systems, time-series starting from two different initial conditions separated by a small amount δu, are expected to separate by an amount $e^{\Lambda_{max} t}\delta u$ after a time t, where $\Lambda_{max} > 0$ is the maximum Lyapunov exponent of the system. Forecasting of such chaotic trajectories several Lyapunov times into the future thus shows that the closed-loop RC can serve as an accurate model of the observed dynamics. Fig. 1.2(b) further shows that, even when the accuracy of the long-term RC prediction breaks down (say after a time t such that $\Lambda_{max} t > 6$), the RC predicted time-series qualitatively looks similar to the original time-series. This is confirmed by comparing long-term statistical properties of the dynamics, like power spectrum (Fig. 1.2(c)) and Lyapunov spectrum (which is the set of Lyapunov exponents of a chaotic systems, ordered according to their numerical values, shown in Fig. 1.2(d)) for the original time-series and closed-loop the RC predicted time-series. Figs. 1.2(c,d) show that long-term statistical properties of the RC prediction indeed matches with the corresponding properties of the original time-series data, thus fulfilling the 'time-series prediction and generalization' condition (ii) of Sec. 1.3.1.

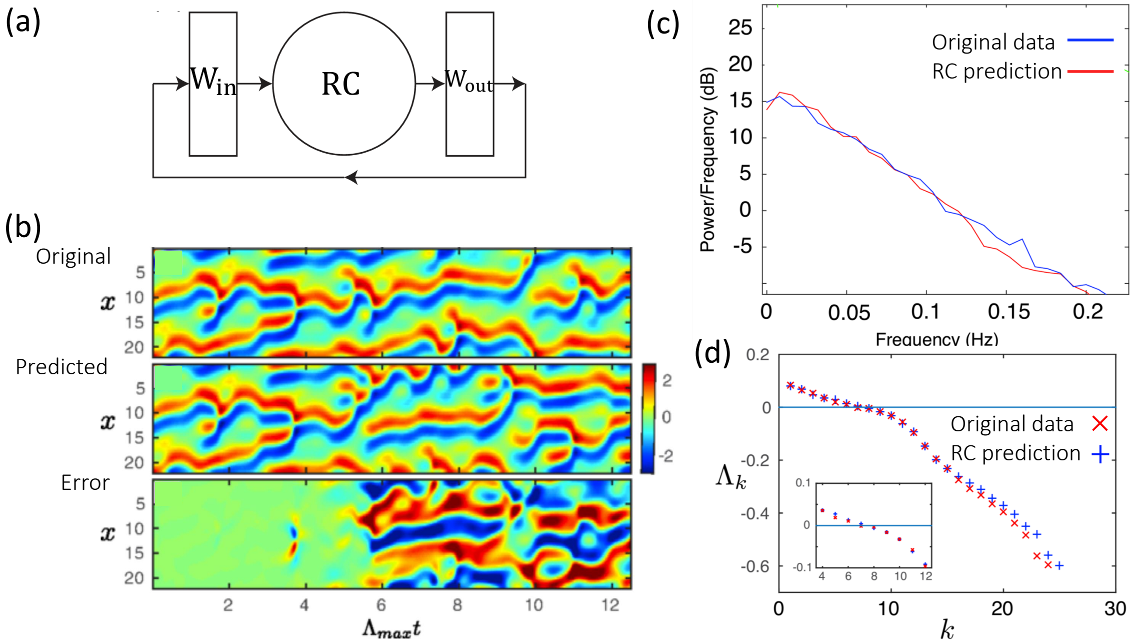

Figure 1.2: Properties of time-series prediction by reservoir computers: (a) a schematic of closed-loop configuration of a trained RC used for autonomous time-series prediction, (b) original and RC-predicted time-series from a large, spatiotemporally chaotic system (Kuramoto-Sivashinsky model) and the prediction error over time, with time measured in terms of the maximum Lyapunov exponent Λ_{max}, (c) comparison of the power spectrum, and (d) Lyapunov spectrum of the original time-series data and RC prediction for the Kuramoto-Sivashinsky model. (a) and (b) are adapted from Ref. [54] and (c) and (d) are adapted from Ref. [56].

Since the RC has been demonstrated in the literature to fulfill conditions (i) and (ii) of Sec. 1.3.1, we wish to know if it can capture the causal interactions between the dynamics of different agents in the complex system (condition (iii) of Sec. 1.3.1). In our formalism, this is represented by causal interaction among the components of the vector $\mathbf{u}(t)$. Mathematically, we suppose that, if a component i of the vector $\mathbf{u}(t)$ is causally affecting another component j of $\mathbf{u}(t)$, then, the derivative $\partial u_j(t+\tau)/\partial u_i(t)$ would be non-zero for some $\tau > 0$. For example, if the system does not have any time-delay, then over a very short period of time $\tau = \delta t$, the derivative would acquire a non-zero value if the dynamics of component i is causally affecting the dynamics of component j. For time-delayed systems, we can set τ to be the time-delay of causal interactions to have a similar effect. Note that, our notion of causality, as defined with the partial derivative of a component of the state vector $\mathbf{u}(t)$ with respect to another component at a past time, is close to the concept of causality in terms of interventions. That is, for example, if we changed a component u_i at time t by putting an external perturbation, that would cause a corresponding change to a component u_j a time τ later if and only if the component j is causally connected to the component i over this time. As such, our task of causal inference in complex systems amounts to estimating the derivatives $\partial u_j(t+\tau)/\partial u_i(t)$ from the sampled observations of the time-series of $\mathbf{u}(t)$.

Partial derivatives like $\partial u_j(t+\tau)/\partial u_i(t)$ cannot be directly calculated from the time-series of $\mathbf{u}(t)$ directly in a straightforward way. This is because, when we test causal dependence between a pair of variables, we wish to have the other variables unchanged, so that their effects do not interfere with the testing pair of variables. In the partial derivative $\partial u_j(t+\tau)/\partial u_i(t)$, all components of the vector $\mathbf{u}(t)$ other than the i-th and j-th components are assumed to be held fixed. In the original time-series, this rarely happens, since, at every sampling time point, generally there are changes to all components of the state vector $\mathbf{u}(t)$ together. Thus, the original time series of $\mathbf{u}(t)$ itself cannot be used to calculate the partial derivatives $\partial u_j(t+\tau)/\partial u_i(t)$ indicating causal interactions.

However, if we have a model for the observed dynamics of the state $\mathbf{u}(t)$, of the form $\mathbf{u}(t+\tau) = \mathbf{M}\left(\mathbf{u}(t)\right)$, then we can use that model to calculate the partial derivatives: $\partial u_j(t+\tau)/\partial u_i(t) = \partial M_j/\partial u_i$ and test the causal dependence among the components of $\mathbf{u}(t)$. Since RCs are able to give us an excellent model of an observed time-series that not only fits to the sampled data accurately, but can also both predict and generalize the sampled data, we use the RC model of an observed time-series to calculate the partial derivatives. The model is obtained by combining Eqs. 1.1 and 1.2 into a single equation, assuming a perfect fit to the training data by the RC (i.e., $\mathbf{u}(t) \approx \tilde{\mathbf{u}}(t)$),

$$\mathbf{u}(t+\tau) = \mathbf{W}_{\text{out}} \sigma \left(\mathbf{W}_{\text{in}} \mathbf{u}(t), \mathbf{W} \mathbf{R}(t) \right), \tag{1.3}$$

which we use as the model of the observed time-series to calculate the partial derivatives $\partial u_j(t+\tau)/\partial u_i(t) = \partial M_j/\partial u_i$. If the RC model of the time-series, given by Eq. 1.3 indeed captures the true causal dependencies among the components of the time-series, the resulting derivatives should indicate such. Throughout the rest of this thesis, we shall use equations like Eq. 1.3 to test whether RC models are able to actually predict the true causal dependencies among the components of a complex system, which would be known in all cases that we shall consider. We shall also investigate how properties of the time-series, like amount of correlations among their components, and sampling limitations, like finite sampling time, observational and dynamical noise systematically affect the causal learning performance of the RC.

1.4 Organization of this thesis

The next three chapters of the thesis are about three applications of causal inference by the RC.

1.4.1 RC inference of short-term causal dependence

In chapter 2, based on Ref. [59], we test whether RCs can capture short term causal dependencies in the data, i.e., predict how one component of a time-series affects another over a very short time. To test this, we shall use simulated data from complex networks of coupled chaotic Lorenz oscillators. Varying the size of the network, as well as observational and dynamical noise strengths in the time-series data, we show that the RC is able to correctly infer the causal interactions networks if the degree of correlation among the time-series from different Lorenz oscillators and the observational noise strength are sufficiently small. Interestingly, in contrast to what we generally described so far in this chapter, presence of dynamical noise is shown to aid RCs to identify causal interactions. We hypothesize that the reason of this is two-fold: (i) dynamical noise breaks down correlations and synchrony among the time-series from different Lorenz oscillators, and (ii) over a short time period, dynamical noise at one Lorenz oscillator only propagates to those particular oscillators which are causally connected to that oscillator. As such, dynamical noise in the network generates perturbations of the Lorenz oscillator nodal states, which propagate in time to only causally connected nodal states, revealing the causal connectivity structure in the process. Even for relatively small dynamical noise strengths, when these changes are infinitesimal, the RC is shown to be able to successfully use their information to correctly predict short-term causal connections among the networked Lorenz oscillators.

1.4.2 RC inference of time-delayed causal dependence

In contrast to the short-term causal inference in chapter 2, chapter 3, based on Ref. [60], discusses inference of time-delayed causal interaction by the RC. In this chapter, we use experimental and simulated voltage data from networks of delayed-feedback-coupled nonlinear optoelectronic oscillator (OEO) networks - an excellent testbed for studying nonlinear dynamics of delay-coupled networks. The results are similar to the ones in chapter 2 in that the RC is shown to be able to capture the causal interaction network correctly, provided the amount of synchronization among the dynamical variables are small - something that can be achieved by tuning the coupling strengths, increasing the dynamical noise strength, having different time-delays along different feedback network links, or by rewiring the OEO networks. Moreover, the RCs identifying causal interactions are here shown to perform equally well on experimental data. This observation inspires us to apply the RC causal inference technique to more real-world time-series datasets from less well-controlled experiments. This is the subject of chapter 4.

1.4.3 Effects of sampling conditions on RC causal inference and statistical analysis

In chapter 4, we apply RC technique for short-term causal dependence, but on publicly available experimental data on calcium fluorescence dynamics from the neural system of live *C. elegans*. Such data suffers from low sampling temporal resolution, high noise level, large amount of correlation among different variables, and only a partial measurement of the full biophysical neural state of the organism. However, the *C. elegans* neuronal connectome is completely mapped and thus serves as the ground truth for benchmarking the network inference performance of the RC. To investigate the effect of data limitations on network inference in this system, we also use time-series data from a synthetic model of coupled Lorenz oscillators on the same network and systematically changed the different sampling parameters. In all such cases, when the data limitations restrict the complete identification of the causal network, we apply a previously-developed statistical technique with surrogate time-series data to assign statistical confidence to each inferred causal link. The RC network inference technique, together with the statistical methodology, is shown to yield useful information about the *C. elegans* neural connectome, when compared with the ground truth connectome.

Finally, the next section summarizes how RC can be used for inference of causal connections in complex systems.

Chapter 2: Inference of short term causal interactions: applications in coupled Lorenz oscillators

This chapter is based on the publication "Using machine learning to assess short term causal dependence and infer network links" by Amitava Banerjee, Jaideep Pathak, Rajarshi Roy, Juan G Restrepo, Edward Ott, published as Chaos 29, 121104 (2019). The code to generate some of the results can be found in the GitHub repository banerjeeamitava/Reservoir-Computer-Network-Inference.

2.1 Overview

This chapter demonstrates the first concrete application of our network inference procedure, using simulated time-series data from networks of coupled chaotic Lorenz oscillators. The problem of determining causal dependencies among various agents, in an unknown time-evolving system, from its time series observations, has wide-ranging and diverse applications. These include inferring neuronal connections from spiking data, deducing causal dependencies between genes from expression data, discovering long spatial range influences in climate variations, etc. Previous work has often addressed such problems by considering correlations, prediction impact, or information transfer metrics. In this chapter we propose a new method for causal inference in complex systems that leverages the potential ability of a specific type of machine learning, called ‘reservoir computing’ (RC), to perform predictive and generative tasks involving complex time-series.

Using RCs, We introduce and test a general computational technique for the inference of short term causal dependence among the measured time-dependent state variables of an unknown dynamical system. The basic method involves training the RC with the observed time-series data of the state variables of the unknown dynamical system to construct a computer model for the unknown dynamics, and, subsequently, estimating the causal dependencies of different state variables on each other in that computer model.

We test our method on model complex systems consisting of networks of many interconnected chaotic Lorenz oscillators. Comparing the causal networks inferred by the RC to the ground truth, we find that dynamical noise can greatly enhance the effectiveness of our technique, while observational noise and presence of synchronization degrades the effectiveness. We believe that the competition between these two opposing types of noise, as well as the degree of synchrony in the system will be the key factor determining the success of causal inference in many of the most important application situations. These tests show that machine learning offers a unique and potentially highly effective approach to the general problem of causal inference.

2.2 Introduction

The core goal of science is often described to be generalization from observations to understanding [61], commonly embodied in predictive theories. Related to this is the desire to use measured data to infer necessary properties and structure of any description consistent with a given class of observations. On the other hand, it has recently emerged that machine learning

(ML) is capable of effectively performing a wide range of interpretive and predictive tasks on data [62]. Thus it is natural to ask whether machine learning might be useful for the common scientific goal of discovering structural properties of a system from data generated by that system. In this chapter we consider an important, widely applicable class of such tasks. Specifically, we consider the use of machine learning to address two goals.

Goal (i): Determine whether or not a state variable of a time evolving system causally influences another state variable.

Goal (ii): Determine the 'strength' of such causal influences.

In the terminology of ML, Goal (i) is referred to as "classification ML," and Goal (ii) is referred to as "regression ML." These goals have previously been of great interest in many applications (e.g., economics [33], neuroscience [63], genomics [21], climate [64], etc.). Many past approaches have, for example, been based upon the concepts of prediction impact, [33, 63] correlation, [65–68] information transfer, [31, 69] and direct physical perturbations [11, 70]. Other previous works have investigated the inference of network links from time series of node states assuming some prior knowledge of the form of the network system and using that knowledge in a fitting procedure to determine links [66, 71–74]. In addition, some recent papers address network link inference from data via techniques based on delay coordinate embedding, [72] random forest methods, [75] network embedding algorithms [76] and feature ranking [77]. In this chapter, we introduce a technique that makes the use of an ML training process in performing predictive and interpretive tasks and attempts to use it to extract information about causal dependencies. In particular, here we use a particular type of machine learning (ML) called reservoir computing, an efficient method of time series analysis which has previously been successfully used for different tasks, e.g., prediction of chaotic dynamics [54, 78, 79] and speech recognition [44, 47] to mention a few. In our case, a “reservoir” dynamical system is trained such that it becomes synchronized to a training time series data set from the unknown system of interest. The trained reservoir system is then able to provide an estimation of the response to perturbations in different parts of the original system, thus yielding information about causal dependencies in the actual system. We will show that this ML-based technique offers a unique and potentially highly effective approach to determining causal dependencies. Furthermore, the presence of dynamical noise (either naturally present or intentionally injected) can very greatly improve the ability to infer causality, [71, 72] while, in contrast, observational noise degrades inference.

This rest of this chapter is organized as follows: Sec. 2.3 mathematically defines our notion of short term causal dependence (STCD) in the context of dynamical systems and argues how it captures the effects of small external interventions into the system, which are often employed in experiments to identify causal connections. Section 2.4 introduces the method of inferring STCD using reservoir computer models of the dynamics. The next section, Sec. 2.5, illustrates the performance of the RC-based STCD inference technique in the system of coupled Lorenz oscillators, and identifies conditions which are helpful and detrimental for STCD inference.

2.3 Short Term Causal Dependence (STCD)

We begin by considering the very general case of an evolving, deterministic, dynamical system whose state at time t is represented by the M-dimensional vector $\mathbf{z}(t) = [z_1(t), z_2(t), \ldots, z_M(t)]^T$, where $\mathbf{z}(t)$ evolves via a system of M differential equations, $d\mathbf{z}(t)/dt = \mathbf{F}(\mathbf{z}(t))$, and has reached a statistically steady dynamical state (perhaps chaotic). In this context, we frame the issue of causality as follows: Will a perturbation at time t applied to a component z_i of the state vector $\mathbf{z}(t)$ (i.e., $z_i(t) \to z_i(t) + \delta z_i(t)$) lead to a subsequent change at a slightly later time, $t+\tau$, of another scalar component z_j (i.e., $z_j(t+\tau) \to z_j(t+\tau) + \delta z_j(t+\tau)$); and how can we quantify the strength of this dependence? This formulation might suggest comparison of the evolutions of $\mathbf{z}(t)$ that result from two identical systems, one with, and the other without, application of the perturbation. However, we will be interested in the typical situation in which such a comparison is not possible, and one can only passively observe (measure) the state $\mathbf{z}(t)$ of the (single) system of interest. Aside from knowing that the dynamics of interest evolves according to a system of the form $d\mathbf{z}/dt = \mathbf{F}(\mathbf{z})$, we assume little or no additional knowledge of the system, and that the available information is a limited-duration past time series of the state evolution $\mathbf{z}(t)$. Nevertheless, we still desire to deduce causal dependencies, where the meaning of causal is in terms of responses to perturbations as defined above. Since, as we will see, accomplishment of this task, in principle, is not always possible, our approach will be to first propose a heuristic solution, and then numerically test its validity. The main message of this chapter is that our proposed procedure can be extremely effective for a very large class of important problems. We will also delineate situations where our procedure is expected to fail. We emphasize that, as our method is conceptually based on consideration of responses to perturbations, in our opinion, it provides a more direct test of what is commonly of interest when determining causality than do tests based on prediction impact, correlation, or entropy metrics.

Furthermore, although the setting motivating our procedure is for deterministic systems, $d\mathbf{z}/dt = \mathbf{F}(\mathbf{z})$, we will also investigate performance of our procedure in the presence of both dynamical noise (i.e., noise added to the state evolution equation, $d\mathbf{z}/dt = \mathbf{F}(\mathbf{z})$) and observational noise (i.e., noise added to observations of $\mathbf{z}(t)$ used as training data for the machine learning). Both types of noise are, in practice, invariably present. An important result from our study is that the presence of *dynamical* noise can very greatly enhance the accuracy and applicability of our method (a similar point has been made in Ref. [71] and Ref. [72]), while observational noise degrades the ability to infer causal dependence.

To more precisely define causal dependence, we consider the effect of a perturbation on one variable on the other variables as follows. Taking the j^{th} component of $d\mathbf{z}/dt = \mathbf{F}(\mathbf{z})$, we have

$$dz_j(t)/dt = F_j(z_1(t), \ldots, z_2(t), \ldots, z_M(t)),$$

for $j = 1, 2, \ldots, M$. Perturbing $z_i(t)$ by $\delta z_i(t)$, we obtain for small τ, that the component of the orbit perturbation of z_j at time $(t + \tau)$ due to δz_i is

$$\delta z_j(t + \tau) = \tau \left\{ \frac{\partial F_j(\mathbf{z})}{\partial z_i} |_{\mathbf{z}=z(t)} \right\} \delta z_i(t) + \mathcal{O}(\tau^2).$$

We define the *Short Term Causal Dependence* (STCD) metric, f_{ji}, of z_j on z_i by

$$f_{ji} = \left\langle G\left(\frac{\partial F_j(\mathbf{z})}{\partial z_i}\right)\right\rangle, \tag{2.1}$$

where $\langle (\ldots) \rangle$ denotes a long time average of the quantity $(\ldots)$ over an orbit, and the function G is to be chosen in a situation-dependent manner. For example, later in this chapter, we consider examples addressing Goal (i) (where we want to distinguish whether or not $\partial F_j(\mathbf{z})/\partial z_i$ is always zero) for which we use $G(q) = |q|$, while, when we consider an example addressing Goal (ii) and are concerned with the time-averaged signed value of the interaction strength, we then use $G(q) = q$. In either case, we view f_{ji} as quantifying the causal dependence of z_j on z_i, and the key goal of this chapter will be to obtain and test a machine learning procedure for estimating f_{ji} from observations of the state evolution $\mathbf{z}(t)$. For future reference, we will henceforth denote our machine learning estimate of f_{ji} by $\hat{f}_{ji}$. In the case of our Goal (i) experiments, where $G(q) = |q|$, we note that f_{ji} defined by (1) is an average of a non-negative quantity and thus $f_{ji} \geq 0$, as will be our estimate, $\hat{f}_{ji} \geq 0$. Furthermore, for that case we will define STCD of z_i on z_j by the condition, $f_{ji} > 0$, and, when using our machine learning estimate $\hat{f}_{ji}$, we shall judge STCD to likely apply when $\hat{f}_{ji} > \epsilon$ where we call $\epsilon > 0$ the *discrimination threshold*. In the ideal case $(\hat{f}_{ji} = f_{ji})$, the discrimination threshold ϵ can be set to zero, but, in practice, due to error in our estimate, we consider ϵ to be a suitably chosen positive number. We note that, in the ideal case, $\epsilon = 0$ can be regarded as a test for whether or not $F_j(\mathbf{z})$ is independent of z_i.

As a demonstration of a situation for which the determination of STCD from observations of the motion of $\mathbf{z}(t)$ on its attractor is not possible, we note the case where the attractor is a fixed point (a zero-dimensional attractor). Here, the measured available information is the M numbers that are the coordinates of the fixed point, and this information is clearly insufficient for determining STCD. As another problematic example, we note that in certain cases one is interested in a dynamical system that is a connected network of identical dynamical subsystems, and that such a network system can exhibit exact synchronization of its component subsystems [80](including cases where the subsystem orbits are chaotic). In the case where such a synchronized state is stable, observations of the individual subsystems are indistinguishable, and it is then impossible, in principle, for one to infer causal relationships between state variables belonging to different subsystems. More generally, in addition to the above fixed point and synchronization examples, we note that the dimension of the tangent space at a given point $\mathbf{z}^*$ on the attractor is, at most, the smallest embedding dimension of the part of the attractor in a small neighborhood of $\mathbf{z}^*$. Thus the full $M \times M$ Jacobian of $\mathbf{F}(\mathbf{z})$ at $\mathbf{z}^*$ cannot be precisely determined from data on the attractor when the local attractor embedding dimension at $\mathbf{z}^*$ is less than M, which is commonly the case. Thus these examples motivate the conjecture that to efficiently and accurately infer STCD, the orbital complexity of the dynamics should be large enough so as to encode the information that we seek. Note that these considerations of cases where inference of STCD is problematic do not apply to situations with dynamical noise, e.g., $d\mathbf{z}/dt = \mathbf{F}(\mathbf{z}) + \text{(noise)}$, as the addition of noise may be roughly thought of as introducing an infinite amount of orbital complexity. Alternatively, the addition of noise increases the embedding dimension of the data to that of the full state space, i.e., M.

2.4 Using Reservoir Computer to Determine Short Term Causal Dependence

We base our considerations on a type of machine learning called reservoir computing, originally put forward in Refs. [81] and [82] (for a review, see Ref. [83]). We assume that we can sample the time-series data $\mathbf{z}(t)$ from our system at regular time intervals of length τ, so that we have a discrete set of observations $\{\mathbf{z}(0), \mathbf{z}(\tau), \mathbf{z}(2\tau), ...\}$. To begin, we first describe a reservoir-computer-based machine learning procedure in which the reservoir computer is trained to give an output $\hat{\mathbf{z}}(t+\tau)$ in response to an M-dimensional input $\mathbf{z}(t)$ as illustrated in Fig. 2.1.

For our numerical tests we consider a specific reservoir computer implementation (Fig. 2.1) in which the reservoir consists of a network of $R \gg M$ nodes whose scalar states, $r_1(n\tau), r_2(n\tau), \ldots, r_R(n\tau)$, are the components of the R-dimensional vector $\mathbf{r}(n\tau)$.

The nodes interact dynamically with each other through an $R \times R$ network adjacency matrix $\mathbf{A}$, and their evolution is also influenced by coupling of the M-dimensional input $\mathbf{z}(n\tau)$ to the individual nodes of the reservoir network by the $M \times R$ input coupling matrix $\mathbf{W}_{in}$ according to the neural-network-type of evolution equation (e.g., Refs. [54, 78, 79, 83–85])

$$\mathbf{r}((n+1)\tau) = \tanh(\mathbf{A}\mathbf{r}(n\tau) + \mathbf{W}_{in}\mathbf{z}(n\tau)), \tag{2.2}$$

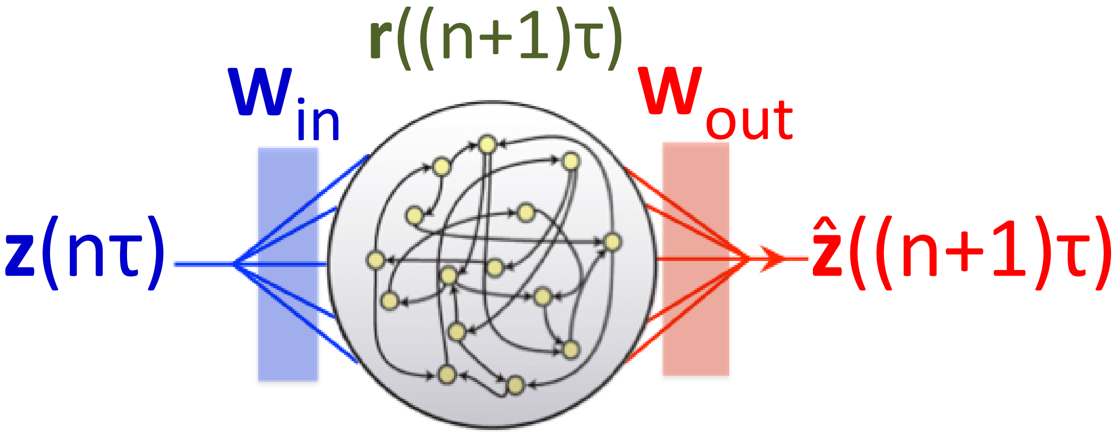

Figure 2.1: Schematic of the reservoir computing architecture used in this work. The input-to-reservoir coupling matrix $\mathbf{W}_{in}$ couples the input time series for the vector $\mathbf{z}$ to the reservoir state vector $\mathbf{r}$. The reservoir-to-output coupling matrix $\mathbf{W}_{out}$ generates the output vector $\hat{\mathbf{z}}$ from the reservoir. $\hat{\mathbf{z}}$ is found to be a good estimate of $\mathbf{z}$ after training.

where $\tanh(\mathbf{v})$ for a vector $\mathbf{v} = (v_1, v_2, v_3, \ldots)^T$ is defined as $(\tanh v_1, \tanh v_2, \tanh v_3, \ldots)^T$. For proper operation of the reservoir computer, it is important that Eq.(2.2) satisfy the 'echo state property' [78, 81, 83] (in nonlinear dynamics this condition is also know as 'generalized synchronization' [86–88]): given two different initial reservoir states, $\mathbf{r}_{1*}$ and $\mathbf{r}_{2*}$, for the same input time series of $\mathbf{z}$, the difference between the two corresponding reservoir states converges to zero as they evolve in time (that is, $|\mathbf{r}_1(t) - \mathbf{r}_2(t)| \to 0$ as $t \to \infty$, implying that, after a transient initial period, $\mathbf{r}(t)$ essentially depends only on the past history of $\mathbf{z}$, $\mathbf{z}(t')$ for $t' \leq t$, and not on the initial condition for $\mathbf{r}$).

Using measured input training data over a training interval of length $T\tau$, which begins after the initial transient period mentioned above, we use Eq.(2.2) to generate $\mathbf{r}(\tau), \mathbf{r}(2\tau), \ldots, \mathbf{r}(T\tau)$. We also record and store these determined values $\mathbf{r}(n\tau)$ along with the corresponding inputs, $\mathbf{z}(n\tau)$ that created them. The matrices $\mathbf{A}$ and $\mathbf{W}_{in}$ are regarded as fixed and are typically chosen randomly. In contrast, the $R \times M$ output coupling matrix $\mathbf{W}_{out}$, shown in Fig. 2.1, is regarded as an adjustable linear mapping from the reservoir states $\mathbf{r}$ to an M-dimensional output vector $\hat{\mathbf{z}}$,

$$\hat{\mathbf{z}}((n+1)\tau) = \mathbf{W}_{out}\mathbf{r}((n+1)\tau). \tag{2.3}$$

'Training' of the machine learning reservoir computer then consists of choosing the RM adjustable matrix elements ('weights') of $\mathbf{W}_{out}$ so as to make $\hat{\mathbf{z}}(n\tau)$ a very good approximation to $\mathbf{z}(n\tau)$ over the time duration $(\tau, 2\tau, \dots, T\tau)$ of the training data. This is done by minimization with respect to $\mathbf{W}_{out}$ of the quantity, $\left\{\sum_{n=1}^{T} \parallel \mathbf{z}(n\tau) - \mathbf{W}_{out}\mathbf{r}(n\tau) \parallel^2\right\} + \beta \parallel \mathbf{W}_{out} \parallel^2$. Here $\beta \parallel \mathbf{W}_{out} \parallel^2$, with β small, is a 'ridge' regularization term [89] added to prevent overfitting and $(\mathbf{r}(n\tau), \mathbf{z}(n\tau))$ are the previously recorded and stored training data. In general, $R \gg M$ is required in order to obtain a good fit of $\hat{\mathbf{z}}$ to $\mathbf{z}(t)$. For illustrative purposes we now consider the ideal case where $\hat{\mathbf{z}} = \mathbf{z}$ (i.e., the training perfectly achieves its goal).

For the purpose of estimating STCD, we now wish to eliminate the quantity $\mathbf{r}$ from the basic reservoir computer system (Eqs.(2.2) and (2.3)) to obtain an evolution equation solely for the state variable $\mathbf{z}$. To do this, we would like to solve Eq. (2.3) for $\mathbf{r}$ in terms of $\mathbf{z}$. However, since R, the dimension of $\mathbf{r}$, is much larger than M, the dimension of $\mathbf{z}$, there are typically an infinite number of solutions of Eq.(2.3) for $\mathbf{r}$. To proceed, we hypothesize that it may be useful to eliminate $\mathbf{r}$ by choosing it to be the solution of Eq.(2.3) with the smallest L_2 norm. This condition defines the so-called Moore-Penrose inverse [90] of $\mathbf{W}_{out}$, which we denote $\hat{\mathbf{W}}_{out}^{-1}$; i.e., the minimum L_2 norm solution for $\mathbf{r}$ is written $\mathbf{r} = \hat{\mathbf{W}}_{out}^{-1}\mathbf{z}$. We emphasize that $\hat{\mathbf{W}}_{out}^{-1}\mathbf{z}$ is not necessarily expected to give the correct $\mathbf{r}$ obtained by solving the system, Eq. (2.2) and Eq. (2.3). However, from numerical results to follow, our choice will be supported by the fact that it often yields very useful estimates of f_{ji}.

Now applying $\mathbf{W}_{out}$ to both sides of Eq. (2.2) and, employing $\mathbf{r} = \hat{\mathbf{W}}_{out}^{-1}\mathbf{z}$ to eliminate $\mathbf{r}(n\tau)$ from the argument of the $\tanh$ function in Eq. (2.2), we obtain a surrogate time -τ map for the evolution of $\mathbf{z}$, $\mathbf{z}((n+1)\tau) = \mathbf{H}[\mathbf{z}(n\tau)]$, where $\mathbf{H}(\mathbf{z}) = \mathbf{W}_{out}\tanh[(\mathbf{A}\hat{\mathbf{W}}_{out}^{-1} + \mathbf{W}_{in})\mathbf{z}]$. Here we note that we do not claim that this map in itself can be used for time-series prediction in place of Eqs. (2.2) and (2.3), which were commonly used in previous works (e.g., Refs. [54, 78, 79, 84, 85]) . Rather, we use it as a symbolic represention of the result obtained after eliminating the reservoir state vector $\mathbf{r}$ from Eqs. (2.2) and (2.3). In particular, the prediction recipe using Eqs. (2.2) and (2.3) is always unique and well-defined, in contrast to the above map, where $\mathbf{W}_{out}^{-1}$ is clearly non-unique. So, we use this map only for causality estimation purposes, as described below. Differentiating $\mathbf{H}(\mathbf{z})$ with respect to z_i, we have

$$\frac{\partial F_j(\mathbf{z})}{\partial z_i} = \tau^{-1}\left[\frac{\partial H_j(\mathbf{z})}{\partial z_i} - \delta_{ij}\right], \tag{2.4}$$

where δ_{ij} is the Kronecker delta, and we propose to use Eqs. (2.1) and (2.4) to determine STCD.

In our numerical experiments, the number of training time steps is $T = 6 \times 10^4$ for Figs. 2.2, 2.3 and $T = 2 \times 10^4$ for Fig. 2.4. In each case, the actual training data is obtained after discarding a transient part of 2×10^4 time steps and the reservoir system sampling time is $\tau = 0.02$. The elements of the input matrix $\mathbf{W}_{in}$ are randomly chosen in the interval $[-0.1, 0.1]$. The reservoir is a sparse random network of $R = 5000$ nodes for Figs. 2.2, 2.3 and of $R = 1000$ nodes for Fig. 2.4. In each case the average number of incoming links per node is 3. Each nonzero element of the reservoir adjacency matric $\mathbf{A}$ is randomly chosen from the interval $[-a, a]$, and $a > 0$ is then adjusted so that the maximum magnitude eigenvalue of $\mathbf{A}$ is 0.9. The regularization parameter is $\beta = 10^{-4}$. These parameters are adapted from Ref. [79]. The average indicated in Eq. (2.1) is over 1000 time steps. The chosen time step τ is sufficiently small compared to the timescale over which $\mathbf{z}(t)$ evolves that the discrete time series $\mathbf{z}(n\tau)$ is a good representation of the continuous variation of $\mathbf{z}(t)$.

Although we use a specific reservoir computing implementation, we expect that, with suitable modifications, our approach can be adapted to 'deep' types of machine learning [62], as well as to other implementations of reservoir computing [44, 47, 48, 91], (notably implementations involving photonics [44], electronics [48] and field programmable gate arrays(FPGAs) [47]).

2.5 Tests of Machine Learning Inference of STCD

In order to evaluate the effectiveness of our proposed method, we introduce mathematical model test systems that we use as proxies for the unknown system of interest for whose state variables we wish to determine STCD. We next use the test systems to generate simulated training data from which we determine STCD by our ML technique. We then assess the performance of the technique by the correctness of its results determined from the known properties of the test systems.

We first consider examples addressing our Goal (i) ($G(q) = |q|$ in Eq. (2.1)), and for our simulation test systems, we consider the case of a network of N nodes and L links, where each node is a classical Lorenz system [92] with heterogeneity from node to node, additive dynamical noise, and internode coupling,

$$dx_k/dt = -10[x_k - y_k + c\sum_{l=1}^{N} a_{kl}^{(x,y)}(y_l - y_k)] + \sigma_{\text{Dyn}} n_{kx}(t), \tag{2.5}$$

$$dy_k/dt = 28(1 + h_k)x_k - y_k - x_k z_k + \sigma_{\text{Dyn}} n_{ky}(t), \tag{2.6}$$

$$dz_k/dt = -(8/3)z_k + x_k y_k + \sigma_{\text{Dyn}} n_{kz}(t). \tag{2.7}$$

The state space dimension of this system is $M = 3N$. The coupling of the N nodes is taken to be only from the y variable of one node to the x variable of another node with coupling constant c, and $a_{kl}^{(x,y)}$ is either 1 or 0 depending on whether or not there is a link from l to k. The adjacency matrix $a_{kl}^{(x,y)}$ of our Lorenz network (not to be confused with the adjacency matrix $\mathbf{A}$ of the reservoir) is constructed by placing directed links between L distinct randomly chosen node pairs. For each node k, h_k is randomly chosen in the interval $[-h, +h]$, and we call h the heterogeneity parameter. Independent white noise terms of equal variance σ_{Dyn}^2 are added to the left-hand sides of the equations for dx/dt, dy/dt and dz/dt, where, for example, $\langle n_{kx}(t) n_{k'x}(t') \rangle = 2\delta_{kk'}\delta(t - t')$. For $\sigma = c = h = 0$, each node obeys the classical chaotic Lorenz equation with the parameter values originally studied by Lorenz [92]. Furthermore, denoting the right-hand side of Eq. (2.5) by F_{xk}, we have $\partial F_{xk}/\partial y_l = 10c$ or 0, depending on whether there is, or is not, a link from y_l to x_k.

Since in this case, the derivative $\partial F_{xk}/\partial y_l$ is time independent, $\langle |\partial F_{xk}/\partial y_l| \rangle$ is also either $10c$ or 0, and, adopting the notation $f_{kl}^{(x,y)} = \langle |\partial F_{xk}/\partial y_l| \rangle$, we denote its machine learning estimate by our previously described procedure by $\hat{f}_{kl}^{(x,y)}$. For a reasonably large network, the number $N^2 - N$ of ordered node pairs (k, l) of distinct nodes is large, and we consequently have many values of $\hat{f}_{kl}^{(x,y)}$. Bayesian techniques (see Ref. [93] and references therein) can be applied to such data to obtain an estimate $\hat{L}$ for the total number of links L, and one can then set the value of ϵ so that there are $\hat{L}$ values of $\hat{f}_{kl}^{(x,y)}$ that are greater than ϵ. Less formally, we find that making a histogram of the values of $\hat{f}_{kl}^{(x,y)}$ often reveals a peak at zero and another peak at a higher positive value with a large gap or discernible minimum in between. One can then estimate ϵ by a value in the gap or by the location of the minimum between the peaks, respectively. For simplicity, in our illustrative numerical simulations to follow we assume that L is known (approximately equivalent to the case that L is unknown but that a very good estimate ($\hat{L}$) has been obtained).

Example 1: A heterogeneous noiseless case. We consider the parameter set $c = 0.3$, $h = 0.06$, $\sigma_{\text{Dyn}} = \sigma_{\text{Obs}} = 0$, $N = 20$, and we vary the number of links L. Figure 2.2(a) (for $L = 50$) and (b) (for $L = 100$) each show an array of 20×20 boxes where each of the boxes represents an ordered node pair (k, l) of the 20-node network, and the boxes have been colored (see Table 1) according to whether the results for our procedure predict a link from l to k ("positive") or not ("negative"), and whether the prediction is correct ("true") or wrong ("false").

TP (True Positive)	Black Square
TN (True Negative)	White Square
FP (False Positive)	Blue Square
FN (False Negative)	Red Square

Table 2.1: Color-coding scheme for Figs. 2.2 and 2.3.

We see that for a typical case with $L = 50$ (Fig. 2.2(a)) *all* the boxes have been correctly labeled, corresponding to all boxes being either black or white. In contrast to this perfect result at $L = 50$, at $L = 100$ (Fig. 2.2(b)) the method fails terribly, and the fraction of correct inferences is small. In fact, we find excellent performance for $L \leq 50$, but that, as L increases past 50, the performance of our method degrades markedly. This is shown in Fig. 2.2(c) where we give plots of the number of false positives (FP) normalized to the expected value of FP that would result if L links were randomly assigned to the $N^2 - N = 380$ node pairs (k, l). (We denote this normalization $\langle \text{FP} \rangle_R$; it is given by $\langle \text{FP} \rangle_R = L(380 - L)/380$.) Note that, with this normalization, for the different heterogeneities plotted in Fig. 2.2(c), the curves are similar, and that they all begin increasing at around $L = 60$ and FP/ $\langle \text{FP} \rangle_R$ becomes nearly 1 (i.e., inference no better than random) past $L \sim 100$. In our earlier discussion we have conjectured that, for inference of STCD to be possible, the orbital complexity should not be too small. To test this conjecture we have calculated the information dimension D_{INFO} of the network system attractor corresponding to the parameters, $c = 0.3$, $h = 0$, $\sigma = 0$, $N = 20$, as a function of L. We do this by calculating the Lyapunov exponents of the system Eqs.(2.5,2.6,2.7), and then applying the Kaplan-Yorke formula for D_{INFO} in terms of the calculated Lyapunov exponents. [94, 95] The result is shown in Fig. 2.2(d), where we see that D_{INFO} decreases with increasing L. Regarding D_{INFO} as a measure of the orbital complexity, this is consistent with our expectation that the ability to infer STCD will be lost if the orbital complexity of the dynamics is too small. As we next show, the above negative result for L increasing past about 60 does not apply when even small dynamical noise is present.

Example 2: The effects of dynamical and observational noise. We first consider the effect of dynamical noise of variance σ^2_{Dyn} for the parameters $h = 0$ (homogeneous), $c = 0.3$, $N = 20$, $L = 200$. Results (similar in style to Figs. 2.2(a) and 2.2(b)) are shown in Figs. 2.3(a), 2.3(b), and 2.3(c). For extremely low dynamical noise variance, $\sigma^2_{\text{Dyn}} = 10^{-9}$ (Fig. 2.3(a)), the result is essentially the same as for zero noise, and about one quarter of the boxes are classified TP, TN, FP, and FN each (since there are 200 links and 400 boxes, this is no better than random assignment). As the noise variance is increased to $\sigma^2_{\text{Dyn}} = 10^{-7.5}$ (Fig. 2.3(b)), the results become better, with a fraction 0.75 of the boxes either TP or TF (as opposed to 0.52 for Fig. 2.3(a)). Upon further increase of the dynamical noise variance to the still small value of $\sigma^2_{\text{Dyn}} = 10^{-6}$ (Fig. 2.3(c)), the results typically become perfect or nearly perfect. Furthermore, excellent results, similar to those for $\sigma^2_{\text{Dyn}} = 10^{-6}$, continue to apply for larger σ^2_{Dyn}. This is shown by the red curve in

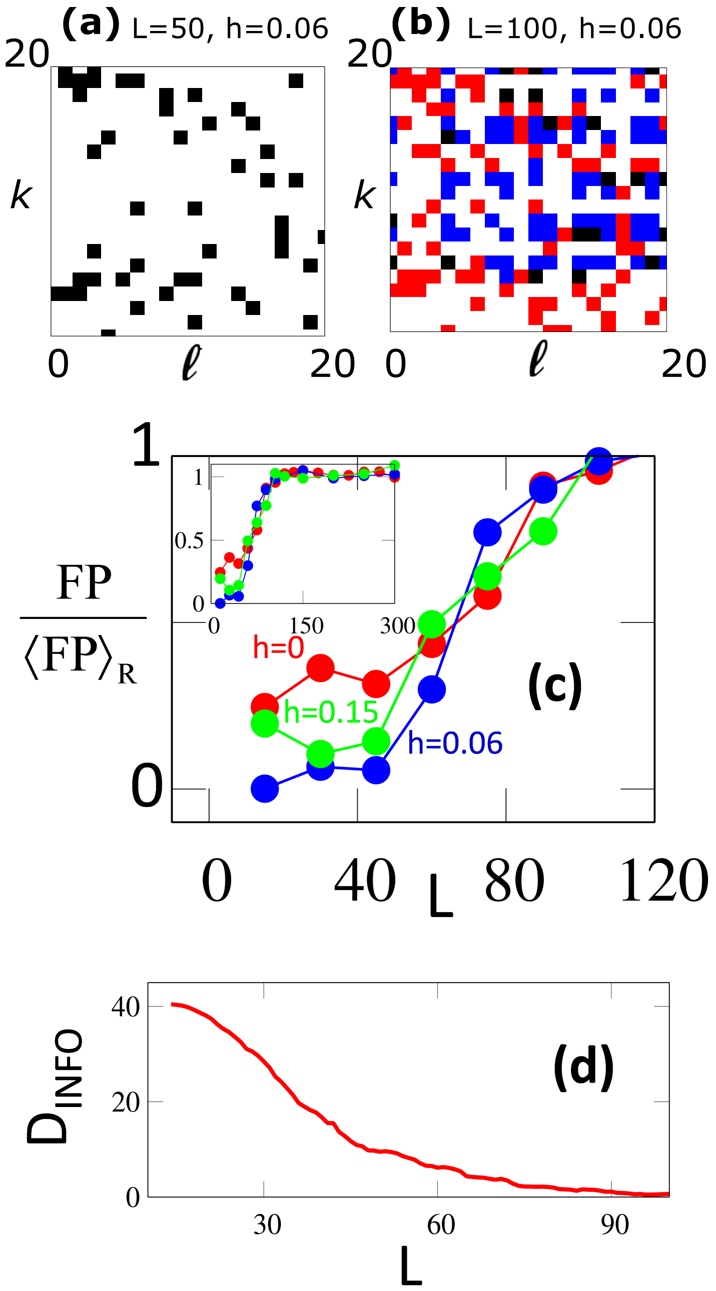

Figure 2.2: Results of Experiment 1 (noiseless case). Panels (a) and (b) show the results of link inferences for two noiseless cases for $L = 50$ links and $L = 100$ links. The inference is perfect in (a), but is very bad in (b). (c) FP/ $\langle \text{FP} \rangle_R$ versus L for $h = 0, 0.06, 0.15$ averaged over 100 random realizations of the system and the reservoir adjacency matrix. (d) The orbital complexity as measured by the attractor information dimension D_{INFO} decreases with increasing L. Note that at each value of L, we compute the D_{INFO} for 10 random realizations of a network with L links with $h = 0$. The Kaplan-Yorke dimension is then averaged over all network realizations and the resulting plot is further smoothed by applying a moving average filter.

Fig. 2.3(f) which shows FP/ $\langle \text{FP} \rangle_R$ versus σ^2_{Dyn} $(N = 20; L = 200)$. Importantly, we also note that our normalization of FP by $\langle \text{FP} \rangle_R$ essentially makes the red curve L-independent over the range we have tested, $50 \leq L \leq 200$. Our interpretation of this dynamical-noise-mediated strong enhancement of our ability to correctly infer links is that the dynamical noise allows the orbit to explore the state space dynamics off the orbit's attractor and that the machine learning is able to make appropriate good use of the information it thus gains.

We now turn to the effect of observational noise by replacing the machine learning time series training data formerly used, $[x_k(n\tau), y_k(n\tau), z_k(n\tau)]$, by $[x_k(n\tau) + \hat{\sigma}_{\text{Obs}}\hat{n}_{kx}(n\tau), y_k(n\tau) + \hat{\sigma}_{\text{Obs}}\hat{n}_{ky}(n\tau), z_k(n\tau) + \hat{\sigma}_{\text{Obs}}\hat{n}_k(n\tau)]$, where the parameter σ^2_{Obs} is the observational noise variance and the $\hat{n}_{kx}, \hat{n}_{ky}, \hat{n}_{kz}$ are independent Gaussian random variables with, e.g., $\langle \hat{n}_{kx}(n\tau)\hat{n}_{k'x}(n'\tau)\rangle = 2\delta_{kk'}\delta_{nn'}$. The blue curve in Fig. 2.3(f) shows the effect of adding observational noise of variance σ^2_{Obs} on top of dynamical noise for the situation $\sigma^2_{\text{Dyn}} = 10^{-5}$ of Fig. 2.3(c). We see from Figs. 2.3(d)-(f) that, when σ^2_{Obs} is below about 10^{-5}, it is too small to have much effect, but, as σ^2_{Obs} is increased above 10^{-5}, the observational noise has an increasing deleterious effect on link inference. This negative effect of observational noise is to be expected, since inference of characteristics of the unknown system is necessarily based on the part of the signal that is influenced by the dynamics of the unknown system, which the observational noise tends to obscure.

Example 3: Inferring Continuous Valued Dependence Strengths. We now wish to address Goal (ii) (for which we take $G(q) = q$ in Eq. (2.1)) and we, accordingly, consider the case where $f^{(x,y)}_{kl}$ for each (k, l) takes on a value in a continuous range (rather than the case of *Examples 1 and 2* where $f^{(x,y)}_{kl}$ is either $10c$ or zero for all (k, l)). For this purpose we replace Eq. (2.5) by

$$dx_k/dt = -10(x_k - y_k) + \sum_l f^{(x,y)}_{kl} y_l, \tag{2.8}$$

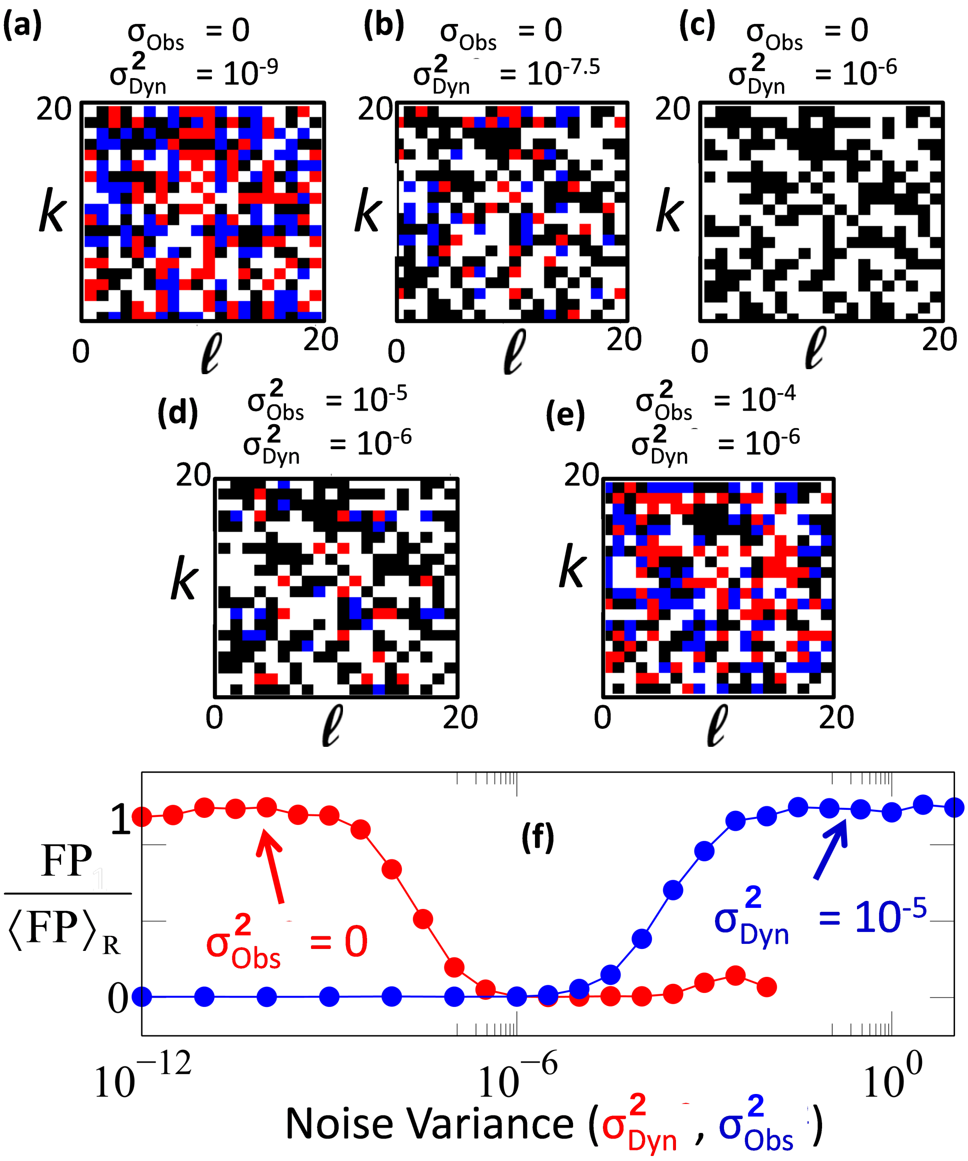

Figure 2.3: The effect of noise on STCD inference. Panels (a), (b), and (c) shows the effect of increasing the dynamical noise variance σ^2_{Dyn} is to greatly enhance the effectiveness of link identification even at the rather low noise level of $\sigma^2_{\text{Dyn}} = 10^{-6}$. In contrast, as shown in panels (d), (e), and (f), starting with the situation (c) and increasing the observational noise variance σ^2_{Obs} degrades link identification. $L = 200, h = 0$ for all the subfigures here.

and consider Eqs. (2.6,2.7, 2.8) as our new test system, with $h = 0.9$, $\sigma^2_{\text{Dyn}} = \sigma^2_{\text{Obs}} = 0$, and $N = 100$ nodes (corresponding to $100 \times 100 = 10^4$ possible connection strength values). We choose the connection strength values as follows. A photographic portrait of Edward N. Lorenz is divided up into $100 \times 100 = 10^4$ boxes and, by using a shading scale from dark (coded as +10) to light (coded as -5), Fig. 2.4(a) is obtained, with the shading scale given to the right of Fig. 2.4(b). Setting $f_{kl}^{(x,y)}$ equal to the color scale value of box (k, l), we next numerically solve Eqs. (2.6,2.7, 2.8). We then use this orbit as the training data for input to our ML determination of causal strength dependence, $\hat{f}_{kl}^{(x,y)}$, and employing the same shading scale, use the thus determined values of $\hat{f}_{kl}^{(x,y)}$ to reconstruct the original portrait, as shown in Fig. 2.4(b). We see that, although the reproduction is not exact, the overall picture is still clearly recognizable, indicating the effectiveness of the method for Goal (ii). For a more quantitative comparison of the actual and the estimated Jacobian elements, we calculate the normalized Frobenius norm of their difference matrix $f^{(x,y)} - \hat{f}^{(x,y)}$. We first apply upper and lower cut-offs equal to 10 and -5.5 respectively to $\hat{f}^{(x,y)}$, in order to eliminate some extreme values. Then we calculate the ratio

$$\delta = \frac{\left|\left| f^{(x,y)} - \hat{f}^{(x,y)} \right|\right|_F}{\left\langle \left|\left| f^{(x,y)} - \tilde{f}^{(x,y)} \right|\right|_F \right\rangle}, \tag{2.9}$$

where $||M||_F = \sqrt{\text{Trace}(M^\dagger M)} = \sqrt{\sum_{i,j} |M_{ij}|^2}$ is the Frobenius norm of the matrix M. Here $\tilde{f}^{(x,y)}$ denotes a matrix constructed by randomly permuting the elements of the matrix $f^{(x,y)}$, and the angled brackets denote an average over such random permutations. So this ratio compares the total error in the inferred Jacobian with the variation in the original matrix elements of $f^{(x,y)}$. For example, for a perfect estimation of $f^{(x,y)}$, we will have $\delta = 0$. In contrast, $\delta = 1$ means that the prediction error is equal to the average error when the elements of $f^{(x,y)}$ are randomized. For the example shown in Fig. 2.4, we find that δ is approximately equal to 0.37.

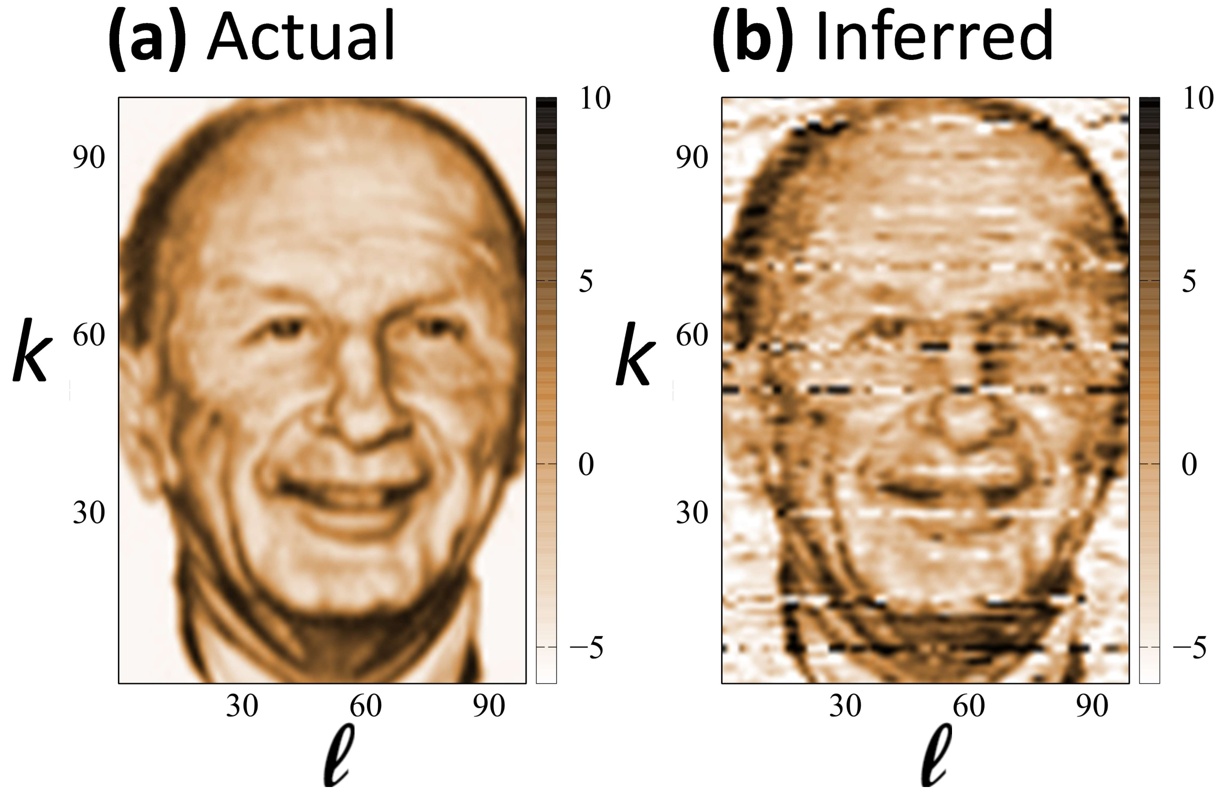

Figure 2.4: Results of Experiment 3. Panel (a) shows a 100×100 pixelated, shade-coded portrait of Edward N. Lorenz; (b) reconstruction of (a) by our ML link inference technique. Note that, in (b), we plot all the values greater than or equal to 10 as black and all the values less than or equal to -5.5 as white.

2.6 Discussion

In this chapter, we have formulated and tested a new, highly effective, machine-learning-based approach for inferring causal dependencies of state variables of an unknown system from time series observations of these state variables. A key finding is that the effectiveness of our approach is greatly enhanced in the presence of sufficient dynamical noise, provided that the deleterious effect of observational noise is not too great. *The competition between the opposing effects of these two types of noise will likely be the essential key factor determining the success* *or failure of causality inference in many of the most important situations of interest (e.g., in neuroscience and genomics).* Much work remains to be done to more fully address the utility of our method. In particular, further numerical tests on diverse systems, and, especially, experimental studies in real world applications, will ultimately determine the circumstances under which the method developed here will be useful.

2.7 Acknowledgements

This work was supported by the U. S. National Science Foundation (Grant DMS 1813027). The authors acknowledge useful discussion with Sarthak Chandra, Amitabha Sen, Woodrow Shew, Nuno Martins, Adrian Papamarcou, Erik Bollt, and especially Brian Hunt.

Chapter 3: Inference of time-delayed causal interactions: applications in optoelectronic oscillator networks

This chapter is based on the publication "Machine Learning Link Inference of Noisy Delay-Coupled Networks with Optoelectronic Experimental Tests" by Amitava Banerjee, Joseph D. Hart, Rajarshi Roy, and Edward Ott, published as Phys. Rev. X 11, 031014 (2021). The code to generate some of the results can be found in the GitHub repository banerjeeamitava/Delay-Network-Inference.

3.1 Overview

In the previous chapter, we employed reservoir computers (RCs) to infer short term causal interactions among the dynamical agents in a complex system. In this chapter, we formulate and test a slightly modified approach of inferring an interaction network in the common situation where the dynamics is noisy and the cause-and-effect interactions among units are delayed.

Building on our RC-based method developed in chapter 2, we propose a two-step method for noninvasively inferring the network of time-delayed connections, using solely the measured time-series data from the dynamics of the network-connected units. In the first step, we train the RC to predict the state of the system after a certain delay time. In the second step, we track the spread of small perturbations in that trained RC model, and use that information to infer the time-delayed networked interactions of the original system. Similar to chapter 2, our technique, by its nature, is noninvasive but is motivated by the widely used invasive network inference method, whereby the responses to active perturbations applied to the network are observed and employed to infer network links (e.g., knocking down genes to infer gene regulatory networks).

We test this technique on experimental and simulated voltage time-series data from delay-coupled optoelectronic oscillator networks, an excellent test bed for complex dynamics, with both identical and heterogeneous time delays along the links. Similar to our results for short term causal interaction inference in chapter 2, we show here that the RC-based technique is also able to infer time-delayed network connectivity, particularly if the system does not exhibit strong synchrony among the voltage dynamics of the individual oscillators. Similar to the results in chapter 2, we also find that the presence of dynamical noise can strikingly enhance the accuracy and ability of our technique, especially in networks that exhibit synchrony. Furthermore, in the optoelectronic oscillator networks, synchrony among the voltage dynamics of the individual oscillators can be broken by changing the coupling strength and network topology, and having heterogeneous delays along the network links - all of which are shown to be helpful for RC-based network inference.

Based on the performance of the RC in finding the time-delayed causal interactions from both experimental and simulated time-series data, we anticipate that this new method for inferring delay-coupled interaction networks offers the promise of widespread future impact for the study of dynamics on such networks.

3.2 Introduction

Dynamically evolving complex networks are ubiquitous in natural and technological systems [13]. Examples include food webs [96], biochemical [97, 98] and gene interaction [99, 100] networks, neural networks [14], human interaction networks [16], and the internet, to mention a few. Inference of the structure of such networks from observation of their dynamics is a key issue with applications such as determination of the connectivity in nervous systems [2, 19, 20], mapping of interactions between genes [21] and proteins in biochemical networks [22], distinguishing relationships between species in ecological networks [23, 24], understanding the causal dependencies between elements of the global climate [25], and charting of the invisible dark web of the internet [101]. In many of these problems, we can only passively observe time series data for the states of the network nodes and cannot actively perturb the systems in any way. Network inference for these cases has led to several different computational and statistical approaches, including Granger causality [102, 103], transfer entropy [31], causation entropy [104], event timing analysis [105], Bayesian techniques [21, 24, 39, 106], inversion of response functions [66, 107], random forest methods [75], and feature ranking methods [77], among others.

In this work, we are interested in the common situation of dynamics that evolves through interactions mediated by the network links along which information transfer is subject to time delay and dynamical noise. We propose and test, both experimentally and computationally, a machine learning methodology to infer these time-delayed network interactions. In doing this, we use only the sampled time series data of the network nodal states. We find that our method is successful in both experimental and computational tests, for a wide variety of network topologies, and for networks with either identical or heterogeneous delay times along their links, provided the time series we use contains sufficient information for the networks to be inferred.

Applications of machine learning techniques for network inference have recently begun to be explored [59, 75, 76, 108, 109]. However, all of these treat networks without link delay. For example, Ref. [109] uses machine learning, but considers inference of generalized synchronization, rather than inference of links, between two systems. In particular, to our knowledge, there is no paper in which the common general situation considered in this chapter is treated by machine learning, namely, link inference in delay-coupled networks with arbitrary topology and noise. Furthermore, a key feature of our work is that, in contrast with Refs. [59, 75, 76, 108, 109], we present an experimental validation with *known ground truth*. Based on the surprising success of machine learning across a wide variety of data-based tasks [62, 110], we believe machine learning is a particularly promising approach to the general network inference problem that we address.

Our approach is based on the demonstrated ability of machine learning for the prediction and analysis of dynamical time series data. In particular, we shall use a specific neural network architecture called Reservoir Computing (RC) [41, 78], which has previously been used to analyze time series data from complex and chaotic systems for such tasks as forecasting spatiotemporally chaotic evolution [54, 111–113], determination of Lyapunov exponents and replication of chaotic attractors [56], chaotic source separation [53, 58], and inference of networks (without link time-delays) [59]. Reservoir computers have been implemented in a variety of platforms [41–43], e.g., in photonic [44–46], electronic and opto-electronic [47, 48] systems. In our technique, we first use time series data from an unknown delay-coupled network to train an RC to predict the future evolution of the network. We then employ this trained network to predict how the effect of imagined applied perturbations would spread through the network, thus enabling us to deduce the network structure. This approach allows us to retain the non-invasive nature of computational tools like the transfer entropy, while also retaining the conceptual advantage of invasive methods [11, 114, 115].

We will test our network inference method on both simulated and experimental time series data from delay-coupled opto-electronic oscillator networks. An opto-electronic oscillator with time-delayed feedback is a dynamical system that can display a wide variety of complex behaviors, including periodic dynamics [116], breathers [117], and broadband chaos [118]. Opto-electronic oscillators have found applications in highly stable microwave generation for frequency references [116], neuromorphic computing [119, 120], chaotic communications [121], and sensing [122]. The nonlinear dynamics of individual [123–125] and coupled opto-electronic oscillators [126–129] are well-understood, making networks of opto-electronic oscillators an excellent test bed for network inference techniques.

We find that our method accurately reconstructs the network from experimentally measured time series data, as long as the coupling is sufficiently strong and the network does not display strong global synchronization. We also find that the presence of dynamical noise, and heterogeneity of delays along network links, may have a significant positive effect on the ability to infer links. Our results provide a clear demonstration that reservoir computing, and possibly other related machine learning methods, can provide accurate network inference for real networks, including situations where complications like noise and time delays in the coupling are present.

This chapter is organized as follows. In Sec. 3.3, we introduce our network inference method for a general delay-coupled network dynamical system. In Sec. 3.4, we present the opto-electronic oscillator networks that we use for testing our method, along with a brief description of the collective dynamics of these networks in different parameter regimes. Section 3.5 presents results of our tests of the effectiveness of our method for both simulated and experimental time series data. Finally, we conclude in Sec. 3.6 with further discussion, suggested future directions, and possible generalizations of our method.

3.3 Reservoir Computing Methodology for Network Inference

3.3.1 The General Delay-coupled Network

In this section, we present the principles of our RC-based network link inference method. We consider a system of D_n nodes, with the interactions among them mediated by a network of time-delayed links. For simplicity of presentation, in this section, we restrict ourselves to networks with identical delays along different links. Later in this chapter (Sec. 3.5.3), we consider application of our framework to networks with heterogeneous link delays. For the present purpose, we assume that the state of the i-th node in the network is given by a time dependent vector $\mathbf{X}_i[t]$ of dimension D_s, with $i = 1, 2, 3, ..., D_n$. We denote the components of this vector by X_i^μ with $\mu = 1, 2, 3, ..., D_s$. The coupled dynamics of the full system is governed by a general delay differential equation of the form

$$\frac{dX_i^\mu(t)}{dt} = F_i^\mu \left[\mathbf{X}_i(t); \mathbf{X}_1(t-\tau), \mathbf{X}_2(t-\tau), ..., \mathbf{X}_{D_n}(t-\tau); t\right]. \tag{3.1}$$

Here F_i^μ is the function governing the dynamics of the μ-th component of the state vector of the i-th node. F_i^μ is a function of X_j^ν if and only if there is a network link (connection) from the ν-th component of the state vector of the j-th node to the μ-th component of the state vector of the i-th node. Note that, as previously noted, for simplicity, in the above equation, we assume that the couplings have only a single time delay τ, which is the time it takes a signal to propagate from one component of the system to another. In the experiments we shall consider here, the time series data from the above dynamics is sampled at a time interval $\Delta t = \tau/k$ (with k being an integer) and are denoted by $\{\mathbf{X}_i [\Delta t]\}, \{\mathbf{X}_i [2\Delta t]\}, \{\mathbf{X}_i [3\Delta t]\}$, ... and so on.

The problem we wish to address can be formalized as follows: *If the observed time series data $\{\mathbf{X}_i[t]\}$ is the only information from the system we have, can we infer the connections of the network assuming that the underlying dynamical equations are of the general form as in Eq.* (3.1)*?* We note that we lack any explicit knowledge of the functions $\{F_i^{\mu}\}$. However, we shall henceforth assume that we know the delay $\tau = k\Delta t$, which, as shown in Appendix A, can, in the case of our opto-electronic test system, be inferred from the available time series data. Furthermore, we note that the performance of our method is not strongly dependent on the accuracy with which we infer the delay time k. For example, in our cases where we typically had $k = 34$ (corresponding to a delay time $\tau = 1.4$ms) in the simulations and experiments, setting the inferred value of k anywhere between 34 and 37 (delay time of 1.4ms and 1.5ms, respectively) gave us essentially the same link inference results. (We chose the delay time in order to work in a regime where the dynamics for our particular experimental test network is well-characterized [127–130]). While the dynamics of the network depend on the delay time, we do not expect any change in the efficacy of our link inference technique for different delay times. In those cases, we only need to change the value of k (as in Eqs. (3.4)-(3.9)) in our inference procedures. Finally, we note that in general situation, it might not be feasible to sample all the components of the state vectors. So, henceforth we will assume that we may sample only a subset of the components of each of the nodal state vectors $\{\mathbf{X}_i[t]\}$, which, without loss of generality, we designate as the first $D'_s(\leq D_s)$ components. We now turn to a description of RC machine learning.

3.3.2 Time Series Prediction with a Reservoir Computer

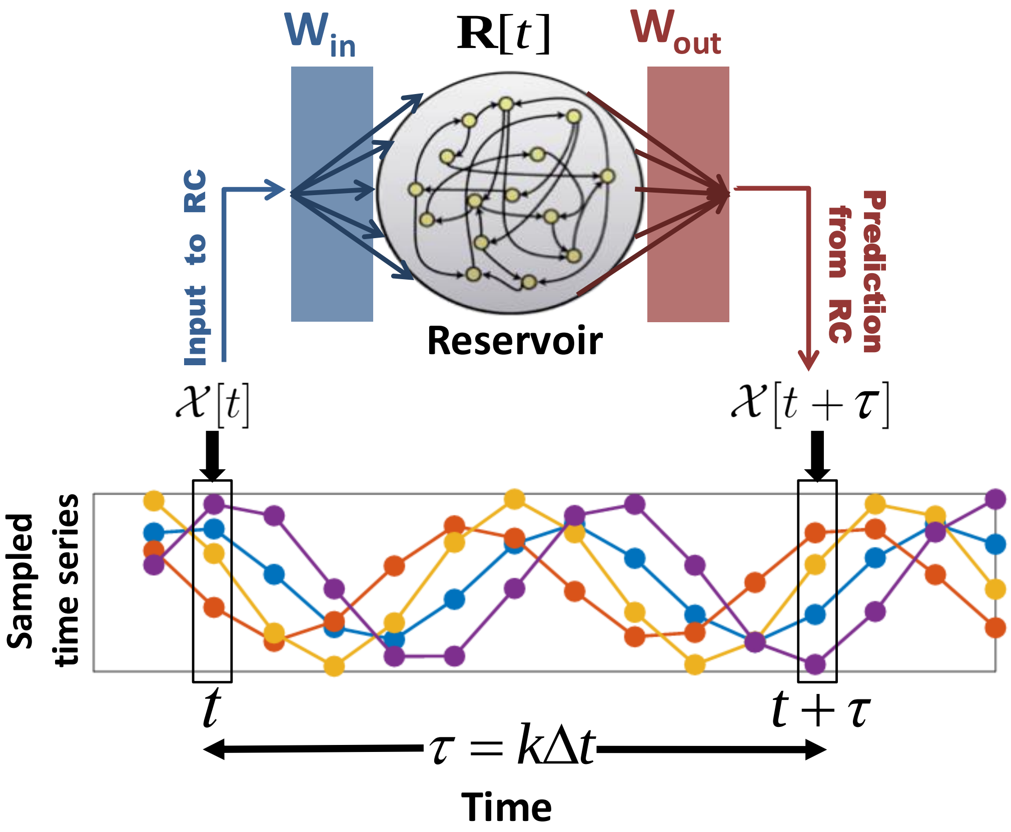

Figure 3.1: Schematics of the reservoir computer (RC) trained for predicting the time series k time steps ahead. In the lower panel, the four time series represent scalar components of $\mathcal{X}[t]$.

In this work, our first step is to train an RC to predict the time evolution of the node states one delay time $\tau = k\Delta t$ into the future, following which we will use that training information to extract the network structure of the system. A schematic of our RC implementation ([54, 59, 78]) is shown in Fig. 3.1. We consider an RC network consisting of a large number of nodes D_r (such that $D_r >> D_n \times D'_s \equiv D_X$). Each of the nodes has a time dependent scalar state, all of which are collected in a column vector $\mathbf{R}$ of length D_r. These reservoir nodes receive measured inputs of the unknown network system states $\{\mathbf{X}_i[t]\}$. We concatenate the sampled input measurements of the time-dependent node state vectors $\{\mathbf{X}_i[t]\}$ and place them in a single time-dependent column vector $\mathcal{X}[t]$ of length D_X, such that the components of $\mathcal{X}[t]$ are arranged as follows;

$$\mathcal{X}[t] = \left(X_1^1[t], X_1^2[t], ..., X_1^{D_s'}[t], X_2^1[t], X_2^2[t], ..., X_2^{D_s'}[t], ..., X_{D_n}^{D_s'}[t]\right)^T. \tag{3.2}$$

This vector is fed into the reservoir via a $D_r \times D_X$ input-to-reservoir coupling matrix $\mathbf{W}_{\text{in}}$ (Fig. 3.1). Furthermore, the reservoir nodes are coupled among themselves with a $D_r \times D_r$ adjacency matrix $\mathbf{H}$. The time evolution of the reservoir node states $\mathbf{R}$ are given by the equation,

$$\mathbf{R}\left[n\Delta t\right] = \sigma\left(\mathbf{H}\mathbf{R}\left[\left(n-1\right)\Delta t\right] + \mathbf{W}_{\text{in}}\mathcal{X}\left[n\Delta t\right]\right), \tag{3.3}$$

which maps the reservoir state at time $(n-1)\Delta t$ to the reservoir state at the next time step, $n\Delta t$, where n is a positive integer, and σ is a sigmoidal activation function acting componentwise on its vector argument (which has the same dimension, D_r, as $\mathbf{R}$).

Keeping in mind the form of Eq. (3.1), our first step is to predict the future values of the sampled components of $\{\mathbf{X}_i\left[\left(n+k\right)\Delta t\right]\}$ in the concatenated form $\mathcal{X}\left[\left(n+k\right)\Delta t\right]$ (Eq. (3.2)) from their current observed values $\mathcal{X}\left[n\Delta t\right]$ using the reservoir state vector $\mathbf{R}\left[n\Delta t\right]$. In our case, this is done by using a suitable linear combination of the reservoir node states with a $D_X \times D_r$ reservoir-to-output coupling matrix $\mathbf{W}_{\text{out}}$ (Fig. 3.1) according to the equation,

$$\mathcal{X}^P\left[\left(n+k\right)\Delta t\right] = \mathbf{W}_{\text{out}}\mathbf{R}\left[n\Delta t\right], \tag{3.4}$$

where the superscript P indicates that the vector is a prediction from the RC, as opposed to being sampled from the actual system. During the training time, we measure the system training time series data $\{\mathcal{X}\left[t\right]\}$ from the unknown system of interest for a large number of time steps. We use this data along with Eq. (3.3) to generate the time series data for the RC nodal states $\{\mathbf{R}\left[n\Delta t\right]\}$, which we store. For the training of the RC, we then find the elements of the matrix $\mathbf{W}_{\text{out}}$ by doing a linear regression from these stored reservoir states $\mathbf{R}\left[n\Delta t\right]$ to the measured time-advanced system states $\left\{\mathcal{X}\left[(n+k)\,\Delta t\right]\right\}$, such that Eq. (3.4) provides a best mean squared fit of the prediction $\mathcal{X}^{P}\left[(n+k)\,\Delta t\right]$ to the measured state $\mathcal{X}\left[(n+k)\,\Delta t\right]$. This amounts to the minimization of a cost function $\mathcal{C}$ given by

$$\mathcal{C} = \sum_{\text{Training Steps n}} \left|\left|\mathcal{X}\left[(n+k)\,\Delta t\right] - \mathbf{W}_{\text{out}}\mathbf{R}\left[n\Delta t\right]\right|\right|^{2} + \lambda\left|\left|\mathbf{W}_{\text{out}}\right|\right|^{2} \tag{3.5}$$

where the last term ($\lambda\left|\left|\mathbf{W}_{\text{out}}\right|\right|^{2}$) is a "ridge" regularization term [89] used to prevent overfitting to the training data and λ is typically a small number.

To completely specify the training procedure that we use, we now specify the structures of the different associated matrices. The elements of the input matrix $\mathbf{W}_{\text{in}}$ are chosen randomly from a uniform distribution in the interval $[-w, w]$. The reservoir connectivity matrix $\mathbf{H}$ is a sparse random matrix, corresponding to an average in-degree d_{av} of the reservoir nodes. The non-zero elements of $\mathbf{H}$ are chosen randomly from an uniform distribution $[-h, h]$ and h is chosen such that the spectral radius of $\mathbf{H}$ (i.e., the maximum magnitude of the eigenvalues of $\mathbf{H}$) is equal to some predefined value ρ. The hyperparameters w and d_{av} are chosen using a Nelder-Mead optimization procedure where we minimize $\mathcal{C}$ for a representative training data and the corresponding output matrix $\mathbf{W}_{\text{out}}$ found from the training data. [We use $w = 1.17$ and $d_{\text{av}} = 2.38$ for tests on simulated data and $w = 1.19$ and $d_{\text{av}} = 2.56$ for tests on experimental data. We typically use values $\lambda = 10^{-4}$ and $\rho = 0.9$ for the other two hyperparameters, 3×10^{4} steps (about 880 delay times) for training, and the sigmoidal activation function σ is taken to be the hyperbolic tangent function. The reservoirs we used typically had 3000 nodes]. After a successful training of the RC with these specifications, Eq. (3.4) can be seen as an *in silico* model for the dynamics of the actual system. Explanations for the special properties of trained RCs, which allow us to use RCs for our purpose, can be found in Refs. [56,57,131,132].

3.3.3 Our Network Inference Procedure

We now describe how we use the training results of the previous subsection to obtain the network structure of the unknown system. We first briefly discuss how the form of Eq. (3.1) allows us to relate the network structure to the spread of the effect of small perturbations to the system. Suppose that, at a time point $t = n\Delta t$, we perturb the ν-th component of the state of node j by an infinitesimal amount $\delta X_j^\nu\,[n\Delta t]$. Differentiating both sides of Eq. (3.1), we see that this perturbation changes the μ-th component of the state of node $i(\neq j)$ at a later time $(n+k)\Delta t = t+\tau$ via the corresponding change in the time derivative,

$$\delta\left(\left.\frac{dX_i^\mu}{dt}\right|_{t+\tau}\right) = \frac{\partial F_i^\mu}{\partial X_j^\nu}\delta X_j^\nu\,[t] + \mathcal{O}\left(\left(\delta X_j^\nu\,[t]\right)^2\right). \tag{3.6}$$

This equation shows that, to lowest order, the effect of the small perturbation on component ν of the state of node j is propagated to the component μ of the state of node i with a delay of $\tau = k\Delta t$ provided that there is a corresponding network link between them, i.e., if $\partial F_i^\mu/\partial X_j^\nu \neq 0$. In particular, propagation of a perturbation from component ν of the state of node j to component μ of the state of node i with a delay of k time steps implies that a directional network link, $(j,\nu) \to (i,\mu)$, exists between them.

While the above discussion is predicated on application of an active perturbation, we see that the result is essentially determined by the partial derivative $\partial F_i^\mu/\partial X_j^\nu$. Thus we wish to determine whether this partial derivative is zero (corresponding to the absence of a link) or not (corresponding to the presence of a link). We attempt to do this by use of the trained RC (which, as we emphasize, was obtained solely from observations, i.e., non-invasively). Indeed, when Eq.

(3.4) is approximately true for a well-trained RC with $\mathcal{X}^P\left[(n+k)\,\Delta t\right] \approx \mathcal{X}\left[(n+k)\,\Delta t\right]$, we can use that equation to consider the RC-predicted dynamics as a proxy for the dynamics of the actual system. In that case, within this assumed RC proxy model, we can analytically assess the effects of small perturbations, and compare them to Eq. (3.6). To do so, we first combine Eqs. (3.3) and (3.4) for the RC, and assume the relation $\mathcal{X}^P\left[(n+k)\,\Delta t\right] = \mathcal{X}\left[(n+k)\,\Delta t\right]$ for the training data to obtain the equation,

$$\mathcal{X}\left[(n+k)\,\Delta t\right] = \mathbf{W}_{\text{out}}\sigma\left(\mathbf{H}\mathbf{R}\left[(n-1)\,\Delta t\right] + \mathbf{W}_{\text{in}}\mathcal{X}\left[n\Delta t\right]\right), \tag{3.7}$$

where the time points belong to the training time series. In order to evaluate the effect of a perturbation to one node on another, we desire to eliminate reservoir variables $\mathbf{R}$ from this equation. Naively, this could be done by solving Eq. (3.4) for $\mathbf{R}\left[(n-1)\,\Delta t\right]$ in terms of $\mathcal{X}\left[(n+k-1)\,\Delta t\right]$. However, the number of components of $\mathbf{R}$ is large compared to the number of components of $\mathcal{X}$, and so there are many solutions of Eq. (3.4) for $\mathbf{R}$. As in our previous work [59], we hypothesize (and subsequently test) that, for our purpose, the Moore-Penrose pseudoinverse [90] (symbolically denoted by $\hat{\mathbf{W}}_{\text{out}}^{-1}$) provides a useful solution of the equation $\mathcal{X}\left[(n+k)\,\Delta t\right] = \mathbf{W}_{\text{out}}\mathbf{R}\left[(n-1)\,\Delta t\right]$ for $\mathbf{R}\left[(n-1)\,\Delta t\right]$. With this, Eq. (3.7) becomes

$$\mathcal{X}\left[(n+k)\,\Delta t\right] = \mathbf{W}_{\text{out}}\sigma\left(\mathbf{H}\hat{\mathbf{W}}_{\text{out}}{}^{-1}\mathcal{X}\left[(n+k-1)\,\Delta t\right] + \mathbf{W}_{\text{in}}\mathcal{X}\left[n\Delta t\right]\right) \tag{3.8}$$

yielding a putative dynamical model for the system from which we will now study the effect of small perturbations. Thus, an infinitesimal amount of change in the network node states at time step $n\Delta t$, written as $\delta\mathcal{X}\left[n\Delta t\right]$, propagates to a change at time $(n+k)\Delta t$ as described by differentiating Eq. (3.8),

$$\delta\mathcal{X}\left[(n+k)\,\Delta t\right]=\left(\mathbb{1}-\mathbf{W}_{\text{out}}\hat{\mathbf{H}}\left[n\Delta t\right]\hat{\mathbf{W}}_{\text{out}}^{-1}\right)^{-1}\mathbf{W}_{\text{out}}\hat{\mathbf{W}}_{\text{in}}\left[n\Delta t\right]\delta\mathcal{X}\left[n\Delta t\right]\;\equiv\mathbf{M}\left[n\Delta t\right]\delta\mathcal{X}\left[n\Delta t\right], \tag{3.9}$$

where we have used Eq. (3.3), assumed that Δt is sufficiently small that $\delta\mathcal{X}\left[(n+k)\Delta t\right]\approx\delta\mathcal{X}\left[(n+k-1)\Delta t\right]$, and defined the new matrix elements $\hat{H}_{ij}\left[n\Delta t\right]=H_{ij}\sigma'\left(R_i[n\Delta t]\right)$ and $\left(\hat{W}_{\text{in}}\right)_{ij}\left[n\Delta t\right]=\left(W_{\text{in}}\right)_{ij}\sigma'\left(R_i[n\Delta t]\right)$ where $\sigma'(u)=d\sigma(u)/du$. We now employ this equation in component form in the place of Eq. (3.6) as a proxy approximating how small perturbations spread across the network. In particular, just like the partial derivative $\partial F_i^{\mu}/\partial X_j^{\nu}$ determines whether a change at the ν-th component of state of node j results in a change of the μ-th component of the state of node i after k time steps in Eq. (3.6), $M_{i,j}[n\Delta t]$ determines the same in Eq. (3.9), when used in conjunction with our definition Eq. (3.2) for $\mathcal{X}\left[t\right]$.

We now describe how we use our determination of $\mathbf{M}$ in Eq. (3.9) to recover the network structure. For this purpose, based on our knowledge of $M_{i,j}[n\Delta t]$, we are interested only in determining whether $\partial F_i^{\mu}/\partial X_j^{\nu}$ is zero (no link $(j,\nu)\to(i,\mu)$) or not. In the true Jacobian, the absence of link would imply $\partial F_i^{\mu}/\partial X_j^{\nu}=0$ exactly. However, in our procedure, there are errors and thus, the elements of $\mathbf{M}\left[n\Delta t\right]$ are never zero. These errors are due to finite reservoir size, finite training data length, noise in the training data, and the Moore-Penrose inversion which, as we hypothesized, is useful but not exact.

Generally speaking, in past link inference methods, the common approach is to somehow obtain a time-independent, continuous-valued score S_{ij} hopefully accurately reflecting the likelihood that node i is linked to j. Once the score is found, as is explained below, one can then choose an appropriate statistical technique for translating the score into a good binary choice of whether or not $i \to j$ corresponds to an actual link. In essence, the key goal of our chapter is to use machine learning to obtain good scores, and for that purpose we will use our machine learning determination of $\mathbf{M}$.

To form an appropriate score corresponding to each of the time-dependent elements M_{ij}, we use $S_{ij} \equiv \langle |M_{ij}| \rangle_t$ where $\langle \rangle_t$ denotes time-averaging over a time sufficiently long that the scores S_{ij} do not change appreciably upon, e.g., doubling the averaging time. In our tests (Sec. 3.5), the averaging time is $1000\Delta t$. Here the absolute value of M_{ij} is to be taken so that the positive and negative values do not cancel each other while doing the time averaging. If this assigned score S_{ij} is above a threshold for a particular element M_{ij}, we assume that there is a network link corresponding to that element, and if the score is less than a threshold, we assume that the corresponding network link is absent.

With the scores S_{ij} defined and calculated, choosing the threshold is a well-known problem of binary categorization of a collection of continuous numerical scores. Since obtaining a useful score for the network link inference is the goal of our machine-learning-based methodology, we regard a detailed discussion of thresholding or other follow-up statistical analysis of the obtained scores to be beyond the scope of this work. We comment only that once the scores S_{ij} are found, one can then choose an appropriate statistical technique for translating the score into a good binary choice. This choice will depend on circumstances that are specific to the situation at hand (e.g., the cost of false positive link choice versus the cost of a false negative link choice). This is a basic problem in statistics and addressed extensively in earlier works with methods such as Receiver Operating Characteristic curves [133, 134], Precision-Recall curves [133, 134], fitting to mixtures of statistical distributions [133], Bayesian techniques [133], etc. A recent paper [135] has proposed a binary classification technique specifically designed for network inference purposes.

To avoid the details of the statistical methodologies, we adopt a procedure which is simple but sufficient for the purpose of evaluating the goodness of the scores of candidate links resulting from our link inference method. Thus we shall henceforth assume that we know the total number of links (denoted by L) in the unknown network and shall designate the largest L scores S_{ij} as corresponding to inferred network links, while those M_{ij} with scores below the largest L scores will be inferred to not correspond to network links. The performance of this link inference technique will be measured by the corresponding number of falsely inferred links ("false positives"). Since we assume that we know the total number of links, the number of falsely inferred links is also equal to the number of missed links ("false negatives"). As we will show below, this method, applied to our machine-learning-determined scores, can produce excellent results in link inference tasks over a wide range of coupling strengths, network topologies, and noise levels. While in practice, L may be unknown, and, depending on the situation, a user may wish to employ an appropriate statistical technique for making the above binary choice from our score, we claim that the results we obtained with L assumed known indicate the value of our technique for score determination, without having to introduce and discuss more involved methods and their appropriateness in different situations (e.g., precision-recall is more appropriate for sparse networks than Receiver Operating Characteristic [134]). Later in this chapter, at the end of Sec. 3.5.1, we shall present results that support this claim.

3.4 Opto-electronic Oscillator Networks

3.4.1 Description of the experiment

In this section, we introduce our opto-electronic network used as a platform to test our network inference procedure. An individual opto-electronic oscillator is a nonlinear, time-delayed feedback loop. Our network consists of four nominally identical opto-electronic oscillators with time-delayed optical coupling between them. The individual and coupled dynamics of opto-electronic oscillators have been studied extensively [118, 124, 125, 127–129, 136, 137].

An opto-electronic oscillator essentially consists of a nonlinearity whose output is fed back into its input with some feedback time delay τ^f. This feedback delay τ^f is inherent to the oscillator, and without it, the system would have no dynamical behavior. A description of one of our opto-electronic oscillators follows:

A fiber-coupled continuous-wave laser emits light of constant intensity. The light passes through an intensity modulator, which serves as the nonlinearity in the feedback loop and can be modeled by $\cos^2(\pi v/2V_\pi)$ where v is the voltage applied to the modulator. For our modulators $V_\pi = 3.4$V. The feedback optical signal is converted to an electrical signal by a photodiode, which is then delayed and filtered by a digital signal processing (DSP) board (Texas Instruments TMS320C6713). This signal is output by the DSP board, amplified, and fed back to drive the modulator. The normalized voltage $x(t) \equiv \pi v(t)/2V_\pi$ applied to the intensity modulator is measured and is our dynamical variable. If there were no DSP board, the delay would be controlled by the optical fiber length and the filtering would be done by the analog electronic components, such as the amplifier. The DSP board simply provides enhanced control over the delay and filter

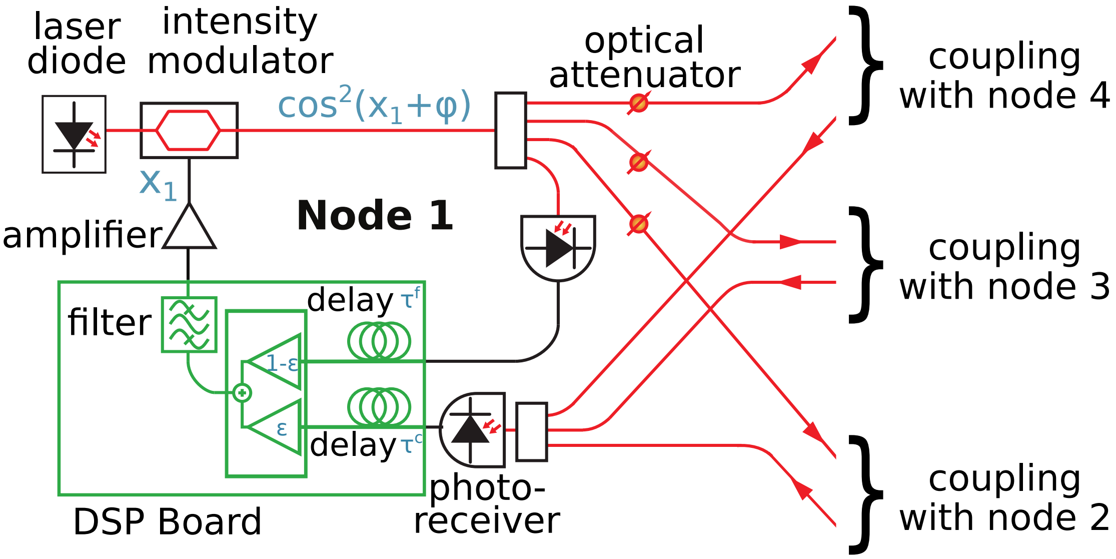

Figure 3.2: Illustration of opto-electronic oscillator and coupling scheme. Red lines indicate signal paths in optical fibers. Black lines are used to indicate electronic signals, and green indicates the digital signal processing (DSP) board.

parameters.

In general, when one couples two oscillators together, a coupling delay will be induced by the finite propagation time of the signal. That coupling delay becomes important when it is not too much shorter than the fastest system time scale. This is the case for our network of coupled opto-electronic oscillators. In general, for a network of oscillators with L links, there will be L different coupling delays. We use the notation τ_{ij}^c to refer to the coupling delay in the link from node j to node i. In our experiments, we choose all the coupling delays to be identical, such that $\tau_{ij}^c = \tau^c$ (heterogeneous delays are considered in Sec. 3.5.3).

An illustration of a single networked opto-electronic oscillator is shown in Fig. 3.2. In order to couple the opto-electronic oscillator to other nodes in the network, a 1×4 fiber optic splitter is introduced after the intensity modulator. One of the optical outputs of the splitter is fed back into that node's DSP board, creating a self-feedback electrical signal. The other three splitter outputs are sent to the other three nodes in the network. Incoming optical signals from the other nodes pass through variable optical attenuators, which control the link strengths, then are combined by a 3x1 fiber optic combiner. This combined optical signal is converted into a coupling electrical signal by a second photodiode.

The feedback and coupling signals are delayed independently in the DSP board such that τ^f and τ^c represent the feedback and coupling delay times, respectively. The outputs of the two delay lines are weighted and combined in the DSP board. This combined signal is output by the DSP, amplified, and used to drive the modulator. The ratio of amplification factors of the coupling signal and feedback signal is given by the coupling strength ϵ.

The amplifier gain is set such that each feedback loop has identical round-trip gain $\beta = 3.8$, phase bias $\phi_0 = \pi/4$, and feedback time delay τ^f = 1.4 ms such that a single uncoupled node behaves chaotically. The digital filter implemented by the DSP board is a two-pole Butterworth bandpass filter with cutoff frequencies $\omega_H/2\pi$ = 100 Hz and $\omega_L/2\pi$ = 2.5 kHz and a sampling rate of 24 kSamples/s. These parameters were chosen because the experimental system with these parameters has been well-characterized [127–130].

For each set of measurements, the nodes are initialized from noise from the electrical signal into the digital signal processing (DSP) board. Then feedback is turned on without coupling, and the opto-electronic oscillators are allowed to oscillate independently until transients die out. At the end of this period, the coupling to the other nodes is turned on and the voltage reading $x(t)$ of each opto-electronic oscillator is recorded on an oscilloscope.

3.4.2 Mathematical Model and Numerical Simulations of the Opto-Electronic Network

The equations governing the dynamics of our network of opto-electronic oscillators are derived in Ref. [136] and are given by

$$\begin{aligned}\frac{d\mathbf{X}_i(t)}{dt} =& \mathbf{E}\mathbf{X}_i(t) - \beta\mathbf{G}\cos^2\left(X_i^1(t-\tau^f)+\phi_0\right) \\ &- \epsilon\beta\mathbf{G}\sum_j A_{ij}[\cos^2\left(X_j^1(t-\tau_{ij}^c)+\phi_0\right) - \cos^2\left(X_i^1(t-\tau^f)+\phi_0\right)] + \xi_i(t)\end{aligned} \tag{3.10}$$

where

$$\mathbf{E} = \begin{bmatrix} -(\omega_L+\omega_H) & -\omega_L \\ \omega_L & 0 \end{bmatrix}, \qquad \mathbf{G} = \begin{bmatrix} \omega_L \\ 0 \end{bmatrix}, \tag{3.11}$$

Here $\mathbf{X}_i = [X_i^1(t), X_i^2(t)]^T$ (corresponding to $D_s = 2$ in Eq. (3.2)) is the state of the digital filter of node i (with $i, j \in \{1, 2, 3, 4\}$, corresponding to $D_n = 4$). By virtue of the second component of $\mathbf{G}$ being zero, coupling between nodes occurs only between X_i^1 and $X_j^1 (i \neq j)$, where $X_i^1(t)$, the normalized voltage of the electrical input to the intensity modulator and is also the only observed variable (i.e., $X_i^1(t) = x(t)$, corresponding to $D_s' = 1$). The nodes are coupled via the adjacency matrix A_{ij}, such that $A_{ij} = 1$ if there is a link to the first component of the state vector of node i from the first component of the state vector of node j, and $A_{ij} = 0$ otherwise. Since the coupling is between only the first components of the vectors $\mathbf{X}_i$, we have dropped the component indices in the adjacency matrix in Eq. (3.10). The coupling strength is given by ϵ, and $\mathbf{E}$ and $\mathbf{G}$ are matrices that describe the filter. Finally, $\xi_i(t) = [\xi_i^1(t), \xi_i^2(t)]^T$ is a vector corresponding to white noise acting independently at each oscillator, and its components have the property that $\left\langle \xi_i^\mu[s]\xi_j^\nu[t] \right\rangle = 2\zeta\delta(s-t)\delta_{ij}\delta_{\mu\nu}$ with ζ denoting the strength of the noise.

In our experiments, we choose all the feedback delays and coupling delays to be nominally equal (i.e., $\tau^f = \tau_{ij}^c = \tau$). In this case, Eq. (3.10) describes a network with Laplacian coupling:

$$\frac{d\mathbf{X}_i(t)}{dt} = \mathbf{E}\mathbf{X}_i(t) - \beta\mathbf{G}\cos^2\left(X_i^1(t-\tau)+\phi_0\right) - \epsilon\beta\mathbf{G}\sum_j L_{ij}\cos^2\left(X_j^1(t-\tau)+\phi_0\right) + \xi_i(t) \tag{3.12}$$

In this case the nodes are coupled via the Laplacian connectivity matrix L_{ij}, defined so that $L_{ij} = 1$ if there is a link to the first component of the state vector of node i from the first component of the state vector of node j, $L_{ij} = 0$ if there is no such link, and $L_{ii} = -\sum_{j\neq i} L_{ij}$. Laplacian coupling tends to lead to global synchronization, which is a particularly challenging case for link inference, as we show in the following sections.

Since the coupling is between only the first components of the vectors $\mathbf{X}_i$, we have dropped the component indices in the Laplacian adjacency matrix in Eq. (3.10). Comparison with Eq. (3.1) shows that in our example, $\boldsymbol{F}_i = \mathbf{E}\mathbf{X}_i(t) - \beta\mathbf{G}\cos^2\left(X_i^1(t-\tau)+\phi_0\right) - \epsilon\beta\mathbf{G}\sum_j L_{ij}\cos^2\left(X_j^1(t-\tau)+\phi_0\right) + \xi_i(t)$, where we have dropped the component superscripts. The oscillators are identical, so these functions are independent of i, except for the noise term. The relevant partial derivative controlling the propagation of perturbation is $\partial F_i^\mu/\partial X_j^\nu \propto L_{ij}\delta_{\mu 1}\delta_{\nu 1}$, for $i \neq j$.

While Eq. (3.10) accurately describes the behavior of our network of opto-electronic oscillators, numerical simulations are inherently discrete in time. Instead of discretizing Eq. (3.10) directly, our simulations use a discrete-time model based on the discrete-time filter equations implemented on the DSP board, which can be found in Ref. [138] and is explained in Appendix B. In particular, for this case, we characterize the noise strength by the variable κ, so that $\kappa = \zeta\Delta t$. For the discrete equation that we simulate, the time step is 0.04ms, which corresponds to the

2.4×10^4 samples/second sampling rate used by the digital filter in our experiment.

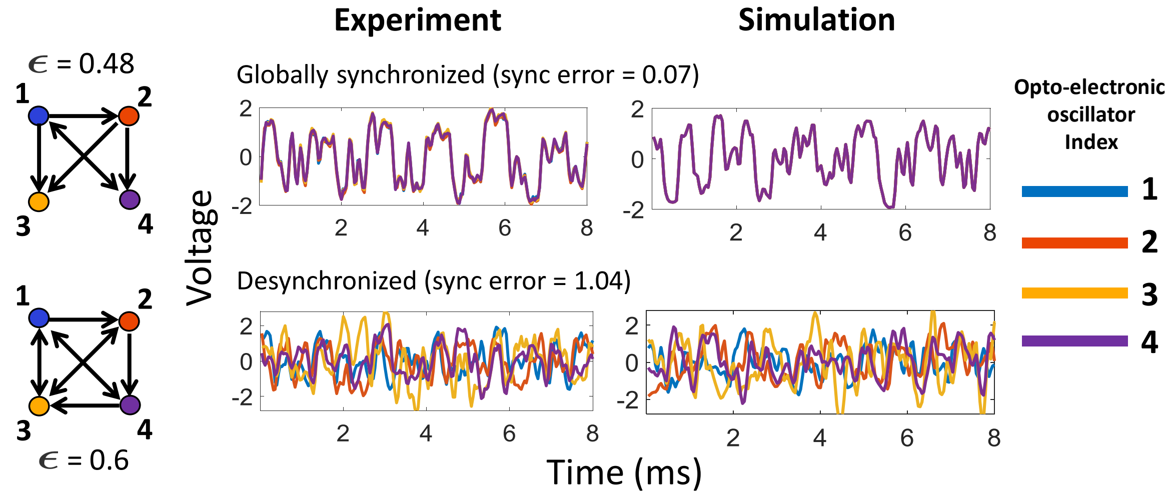

Figure 3.3: Examples of experimental and simulated time series from two opto-electronic oscillator networks showing globally synchronous (upper panels) and completely desynchronized (lower panels) behavior respectively. For the simulations, we have used the noise strength $\kappa = 10^{-6}$. In the plots, purple overlays orange, which overlays red, which overlays blue.

Our model is verified by comparison with the experiment, as shown in Fig. 3.3 for two sets of examples. The upper panel shows experimentally measured (left) and simulated (right) time series of the four opto-electronic oscillators arranged in the six-link network shown. The dynamics are synchronized and are also the dynamics of an individual uncoupled opto-electronic oscillator, since the effect of the Laplacian coupling vanishes for global synchronization. The lower panel shows a measurement (left) and simulation (right) of the four opto-electronic oscillators coupled in the nine-link network shown. In this case, the opto-electronic oscillators do not synchronize even though the coupling is strong ($\epsilon = 0.6$). In both cases, the simulations are in good agreement with the experiment

As we shall see, the degree of synchronization of the oscillators in the network is an important factor in the success of our method to infer the network topology. In order to quantify the degree of global synchrony, we define synchronization error as

$$\text{Synchronization Error} = \frac{1}{D_n(D_n-1)}\left\langle \sum_{i,j} |x_i(t) - x_j(t)| \right\rangle_t \tag{3.13}$$

where $\langle\rangle_t$ means time average over a sufficiently long time. This non-negative measure decreases with the amount of synchronization in the system and is zero for perfect global synchrony. For example, in Fig. 3.3, the synchronized examples (upper panel) have synchronization error ≈ 0.07, whereas the desynchronized examples (lower panel) have synchronization error ≈ 1.04.

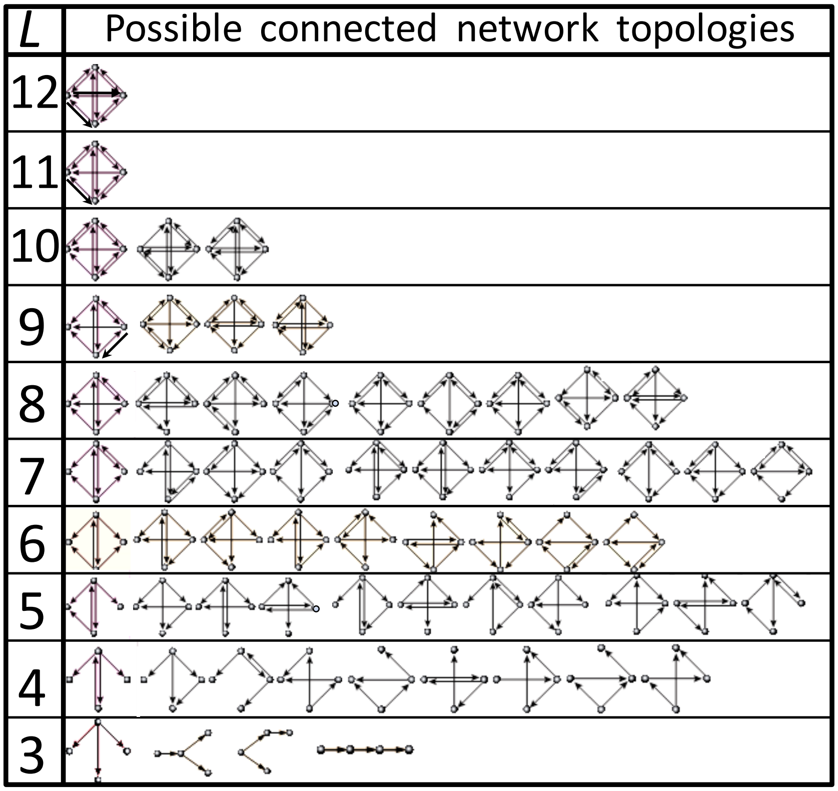

Figure 3.4: List of possible connected directed 4-node networks [127] with different numbers of links (L).

Using computer simulations, we have studied the dependence of the synchronization error on the network coupling strength ϵ and the number L of network links for all possible directed and connected networks with 4 opto-electronic oscillator nodes. The list of the 62 possible networks is shown in Fig. 3.4, and is adapted from Ref. [127]. Figure 3.5 shows the synchronization behavior of these networks as a function of the coupling strength ϵ for fixed $\beta = 3.8$ and $\phi_0 = \pi/4$. In Fig. 3.5, the color coded synchronization error for each of the 62 networks in Fig. 3.4 is shown as one of the 62 horizontal bars for each value of the coupling strength. Here, the convention we follow is that, for fixed number of links (L), moving upwards, the horizontal bars correspond to the networks listed in Fig. 3.4 left to right. The same convention is followed in Figs. 3.6, 3.7 and 3.10. The results in Fig. 3.5 were obtained from numerical simulations without noise ($\kappa = 0$). We see that for intermediate coupling strengths, the networks synchronize, but for small and large coupling strengths, the networks do not synchronize. The seemingly counterintuitive behavior that large coupling strengths lead to desynchronization has been studied for our network of opto-electronic oscillators [130] and is characteristic of delay coupled systems in general [139]. Furthermore, for coupling strengths in the range $\epsilon > 0.5$, for a fixed value of coupling strength, sparser networks are seen to synchronize more readily than densely connected networks. This behavior is also studied and explained in earlier works [130, 140].

3.5 Results of Link Inference Tests

In this section, we present tests of the efficacy of our machine learning technique. These include numerical simulation tests for simulations with homogeneous link delays (Sec. 3.5.1), opto-electronic experimental tests with nominally homogeneous link delays (Sec. 3.5.2), and numerical simulation tests with inhomogeneous link delays (Sec. 3.5.3).

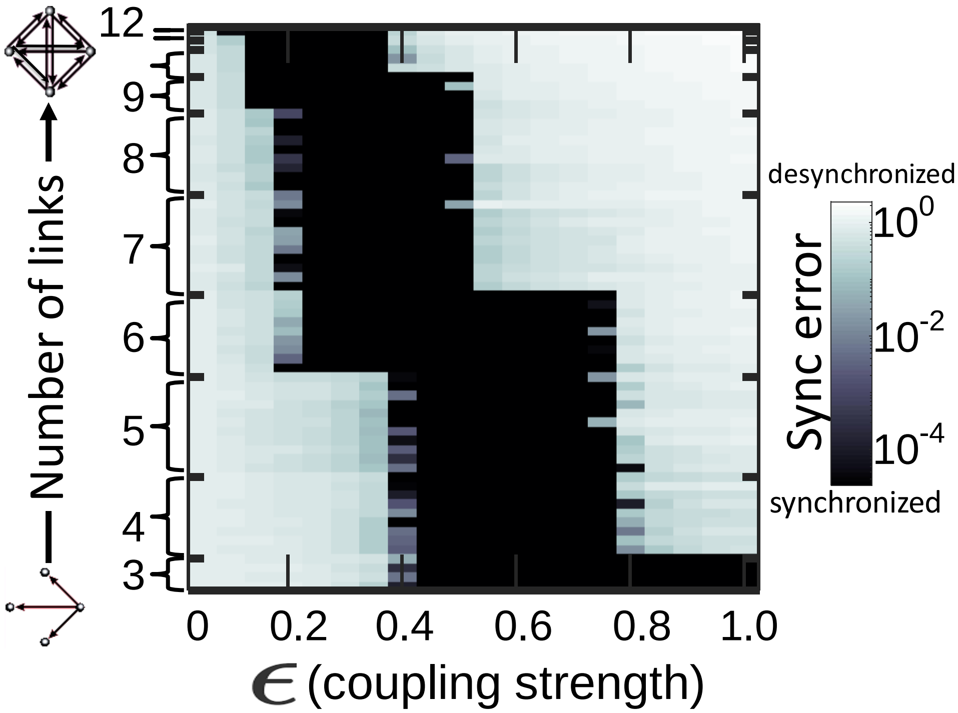

Figure 3.5: Synchronization error for simulated time series of the networks in Fig. 3.4 realized with opto-electronic oscillator nodes with different coupling strengths for random initial conditions.The color coded synchronization error for each of the 62 networks in Fig. 3.4 is shown as one of the 62 horizontal bars for each value of the coupling strength. Here, the convention we follow is that, for fixed number of links (L), moving upwards, the horizontal bars correspond to the networks listed in Fig. 3.4 left to right. The same convention is followed in Figs. 3.6, 3.7 and 3.10.

3.5.1 Performance on Simulated Data - Homogeneous Delays

In this section, we test our methodology on simulated time series for our coupled opto-electronic oscillator networks where links have identical delays. We will use these simulation tests to study the effects of noise and coupling strength on the amount of synchrony in the system, and their effect on the performance of link inference tasks. In particular, in Sec. 3.4.2 we showed that our opto-electronic oscillator networks show synchronized dynamics for certain ranges of the coupling strength ϵ. As we will now show, our method works excellently when the system dynamics does not show pronounced global synchrony, while it does not work well when there

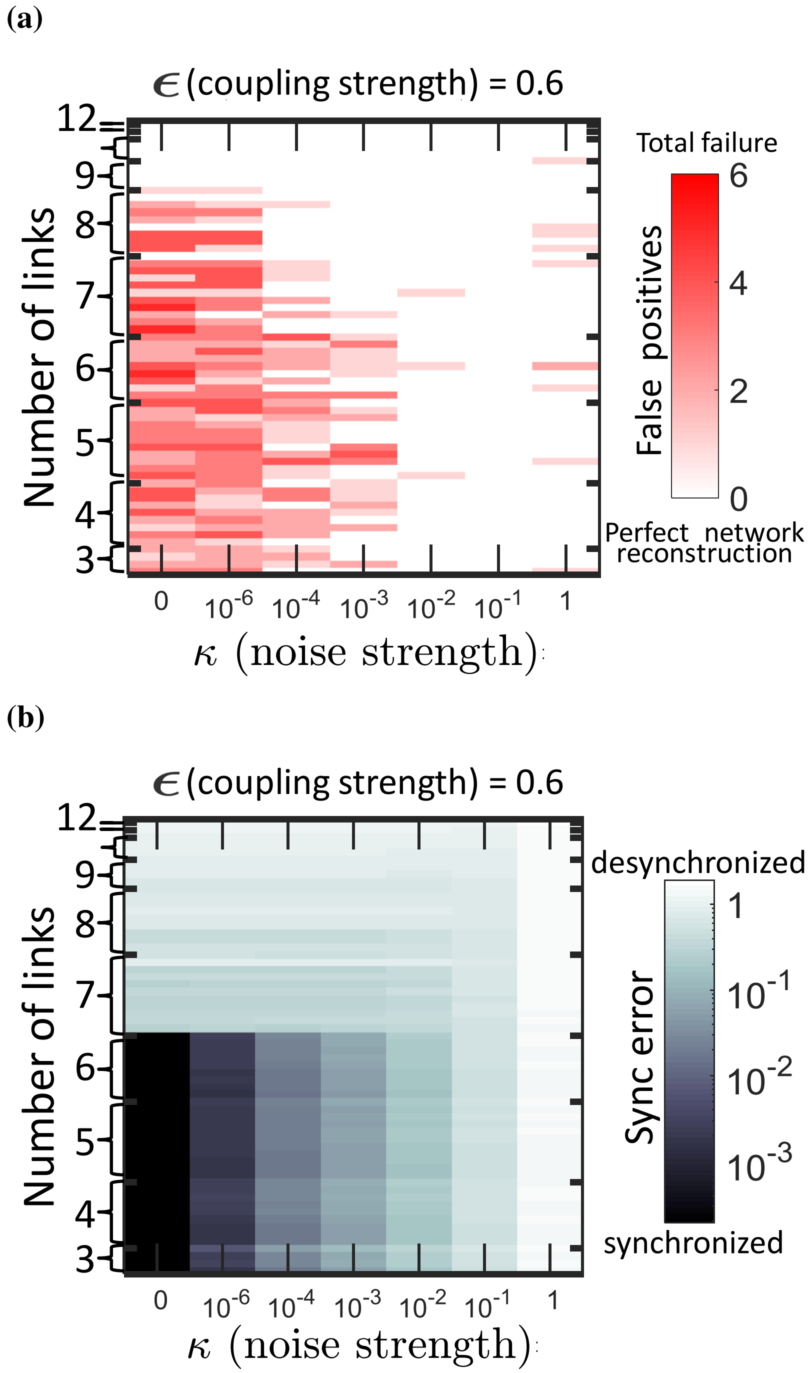

Figure 3.6: Simulation test results with varying noise strength κ. (a) Number of false positives and (b) synchronization error for simulated time series from different networks with progressively increasing noise. As described in the text, each horizontal cut of the plots represents a single trajectory of the system, starting from a random initial condition. The convention for sequence of the networks is the same as in Fig. 3.5.

is pronounced global synchrony. Furthermore, we will show that our technique gives excellent results when there is a small amount of dynamical noise present so that the global synchrony among the opto-electronic oscillators is appreciably broken.

In order to directly demonstrate the effect of loss of synchrony on link inference, we vary the noise level and coupling strength in the following two sets of examples. In the first example set, the system starts with a random initial condition with no noise for 5×10^4 time steps (about 1470 delay times) and is allowed to settle down to an attractor. Then we continue the simulation, but with the noise strength κ set to 10^{-6} for the next 5×10^4 steps, then with the noise strength set to $\kappa = 10^{-4}$ for the next 5×10^4 steps, and so on, keeping the coupling strength fixed at $\epsilon = 0.6$ (Figs. 3.6(a) and 6(b)). As shown in Fig. 3.6(b), as the noise strength increases, it drives the system away from the attractor and disrupts its global synchrony, resulting in larger synchronization error, which allows better link inference performance, as shown in Fig. 3.6(a). In these figures, for each of the time series segments with a fixed noise strength, we use the first 3×10^4 time steps (about 880 delay times) to train our RC and infer links using our procedure described in Sec. 3.3. We repeat this process for each of the 62 possible connected networks of 4 opto-electronic oscillators [127], each one with a different random initial condition.

We use the same procedure in the next set of examples (Figs. 3.7(a) and 3.7(b)), but this time we keep the noise level fixed at its nominal experimental value of $\kappa = 10^{-6}$ and vary the coupling strength ϵ stepwise. The results are summarized in Fig. 3.7. For both Figs. 3.6 and 3.7, we simultaneously plot the number of false positives and synchronization error and follow the same convention as in Fig. 3.5.

As we see from Fig. 3.6, a greater degree of global synchronization generally corresponds to a larger number of false positives, consistent with the hypothesis that global synchrony is

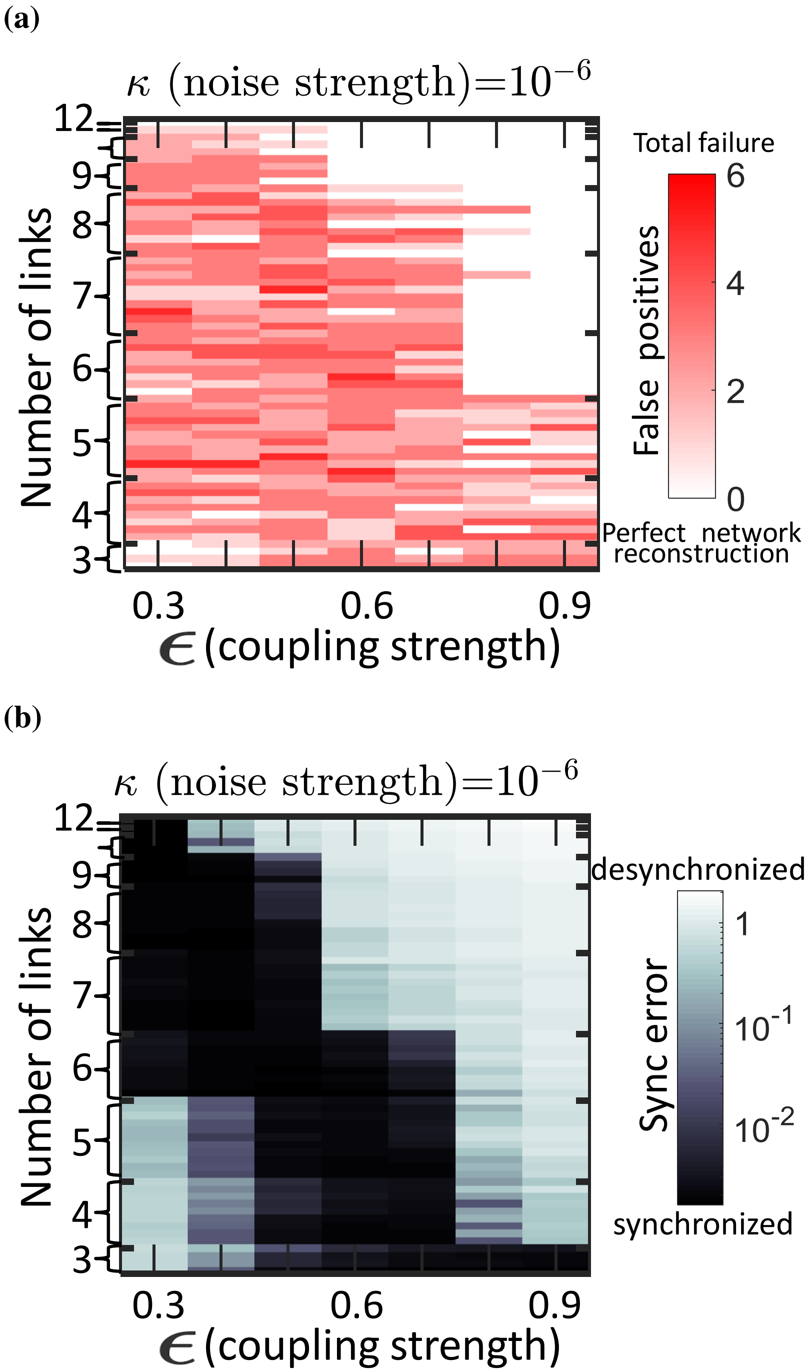

Figure 3.7: Simulation test results with varying coupling strength ϵ. (a) Number of false positives and (b) synchronization error for simulated time series from different networks with progressively increasing coupling strength. As described in the text, each horizontal cut of the plots represents a single trajectory of the system, starting from a random initial condition. The convention for sequence of the networks is the same as in Fig. 3.5.

detrimental to the performance of link inference. This is expected because exact synchronization makes the time series from the four opto-electronic oscillators indistinguishable and hence the observed dynamics yields no information about their underlying causal interactions. In particular, we see that networks with eight or more links do not synchronize sufficiently even in absence of the noise, and we are indeed able to infer the links well, with, at most, only one false positive. In contrast, for networks with smaller numbers of links, which are strongly synchronized for noise levels $\kappa \lesssim 10^{-3}$, we have many false positive link inferences. Again, as we see from Fig. 3.6, all the networks show a loss of global synchrony for sufficiently strong noise levels ($\kappa \gtrsim 10^{-2}$) and this results in almost perfect link inference until the noise strength becomes significant compared to the noiseless opto-electronic oscillator signal amplitudes ($\kappa \gtrsim 10^{0}$). Other examples of similar beneficial role of noise in link inference can be found at earlier works as well, e.g., in [59, 66, 71, 72, 141–143]. Of these, Ref. [143] describes a network inference technique based solely on noise correlations. However, Ref. [143] proposes a technique applicable only in the case of Laplacian coupling, unlike our work which does not employ this model-specific restriction. Furthermore, unlike our method, their method does not work in the absence of dynamical noise, and was not validated using experimental data.

In the second set of examples (Figs. 3.7(a) and 3.7(b)), we fix the noise at a particular strength $\kappa = 10^{-6}$, and progressively increase the coupling strength ϵ. We estimate that this noise level approximates that for the experimental tests reported in the next subsection (Sec. 3.5.2). As in the previous examples, Fig. 3.7 shows that our link inference method performs well when the global synchrony is not too strong. A difference in this set of results from the previous ones is the non-trivial relationship between the coupling strength ϵ and global synchrony, which we have already discussed in the last section (Fig. 3.5). Furthermore, we notice that, even in the absence of global synchrony, the coupling strength needs to exceed a minimum value (about 0.1) for successful link inference. For smaller coupling, the off-diagonal elements of the matrix $\mathbf{M}[n\Delta t]$ could be so small in magnitude that the values corresponding to actual links are of the same order as those corresponding to absent links. Thus, sufficiently large coupling strength ϵ is beneficial for our link inference technique because of the better contrast among the elements of $\mathbf{M}[n\Delta t]$ and diminished global synchrony.

When the number of links is unknown. Finally, to show the effectiveness of our procedure in situations where the number of links L is unknown, in Fig. 3.8 we plot the distribution of the scores S_{ij} calculated from the matrix $\mathbf{M}[n\Delta t]$, for all the 62 networks listed in Fig. 3.4 with coupling strength $\epsilon = 0.6$ and noise strength $\kappa = 10^{-3}$, where, in determining the statistics, for each of the networks, we use a single random realization of initial condition and reservoir couplings. Fig. 3.8 also shows the numerical values and properties of the scores we typically get in our method. In Fig. 3.8(a), we have labeled individual scores into two types [those corresponding to actual network links (colored blue), and those corresponding to absence of links (colored red)] and plotted the scores with the respective synchronization errors in the network. We see that, in the cases with complete desynchronization, where we obtain perfect network inference with known L (as seen in Fig. 3.6(a)), it is also easy to predict a binary decision threshold on our scores (the top histogram, Fig. 3.8(b)). These scores separate into two distinct populations according to their magnitudes, with a gap in between them (Fig. 3.8(b)), and the populations correspond to actual links or absence of links. In the cases for which Fig. 3.6(a) shows finite but small number of false positives with known L, the two populations overlap, but there are still two distinct histogram peaks (the middle histogram, Fig. 3.8(c)), so that one can expect good inference results in cases where L is not known by choosing a suitable threshold based on the shape of the histogram.

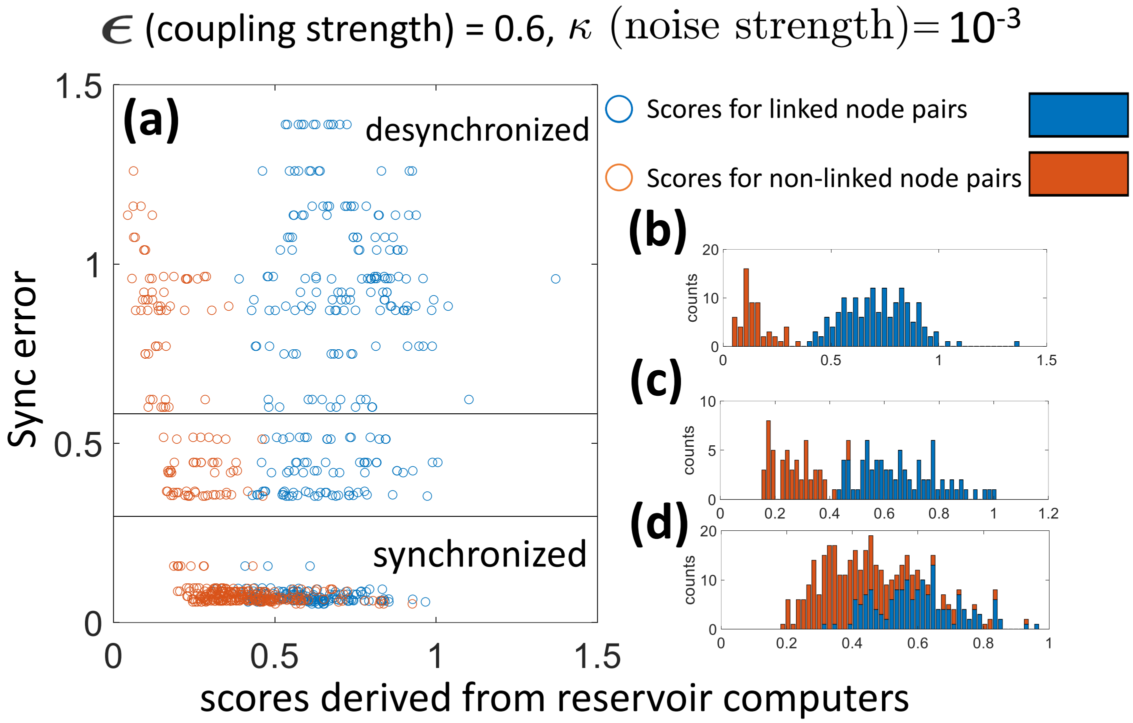

Figure 3.8: This figure shows statistics of our results for the link scores $S_{i,j}$ of all the possible networks (listed in Fig. 3.4) with $\epsilon = 0.6$ and $\kappa = 10^{-3}$, where, in determining the statistics, for each of the networks, we use a single random realization of the initial condition and reservoir couplings. In panel (a), for each individual directed node pair (i, j), i.e., each candidate link, a point is plotted in the sync-error/score plane with true links plotted in blue, and link absences denoted in red, with red overlaying blue. The two black horizontal lines divide the sync-error/score plane into three regions corresponding to networks that are highly desynchronized, moderately synchronized, and strongly synchronized. Panels (b), (c), and (d) show histograms of the node scores for each of the three levels of network synchronization demarcated in panel (a). Bins with scores that all correspond to true links (absence of links) are colored blue (red). Bins with scores corresponding to both true links and link absence are vertically stacked into upper and lower pieces where the lower piece (blue) corresponds to the number of true links in the bin, and the upper piece (red) corresponds to the number of missing links.

However, when the histogram overlap is so great that two peaks are not discernible, we expect that the ability to infer links no longer exists (the bottom histogram, Fig. 3.8(d)). In Fig. 3.6(a), this scenario corresponds to cases with a large number of false positives, even with a known L, because the network exhibits global synchrony. We shall develop an effective statistical methodology to extract useful link information in such cases in the next chapter (Chapter 4).

3.5.2 Performance on Experimental Data

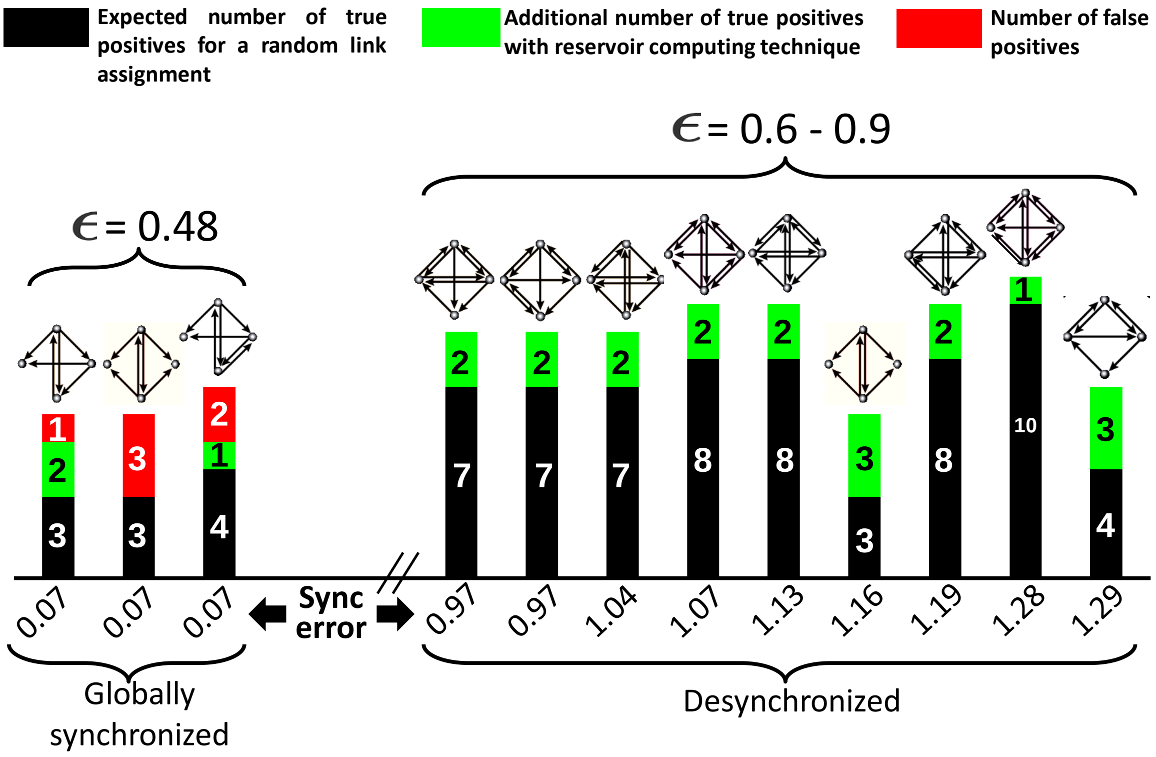

Figure 3.9: Typical performance of our network inference method on experimental data. Each of the vertical bars correspond to the experimental realization of a distinct opto-electronic oscillator network, with the height indicating the corresponding number of links. Each bar is separated into 3 segments as shown, but in many of them only two are seen if there is a perfect link inference. The numbers inside the segments indicate the corresponding heights of the segments, rounded to the nearest integers. Ranges of coupling strength (ϵ) and synchronization error for all the examples are indicated as well

.

Having established the usefulness of our network inference method on simulated time series, we now report our experimental tests on the opto-electronic oscillator networks described in Sec. 3.4.1. In Fig. 3.9, we show some representative examples of the performance of our method on experimental time series. Each column in the figure corresponds to a time-series from a distinct network indicated above the column, with the respective global synchronization error indicated on the horizontal axis. The height of the columns gives the total number of links in the corresponding network. The columns are each divided into three parts (colored in red, green, and black in the plot). The height of the red portion indicates the number of falsely inferred links ("false positives", FP). This portion is absent in the many cases where we have perfect network inference. The number of correctly inferred links ("true positives", TP) is indicated by the total height of the green plus black portions of a column. The height of the black portion indicates the expected number of true positives on average (to nearest integer) that would be obtained if all L links were to be guessed randomly, while the height of the green portion indicates the increase of true positives over what would result from random selection. To evaluate the expected number of randomly selected true positives, we note that, for L links randomly and uniformly assigned among the $D_n\left(D_n-1\right)$ ordered pairs of D_n nodes, the expected number of false positive links is $L\left(1-\frac{L}{D_n(D_n-1)}\right)$ and the expected number of true positive links is thus $L^2/D_n\left(D_n-1\right)$. If our method yields more true positives than $L^2/D_n\left(D_n-1\right)$, then we consider our method to be successful, even if it gives some false positives.

To summarize our experimental results, consistent with the simulation results of Figs. 3.5 and 3.7, the time-series from the experimental opto-electronic oscillator networks (Fig. 3.9) were either globally well synchronized or else were strongly desynchronized, and, when strong desynchronization applied, our method correctly identified all of the links.

3.5.3 Performance on Simulated Data - Heterogeneous Delays

So far we have considered cases where the link time delays τ_{ij}^c in Eq. (3.1) are the same along all links (i.e.,$\tau_{ij}^c=\tau$). We now present results on simulated data for which the link delays τ_{ij}^c are chosen randomly from a uniform distribution of mean τ_0 and width $2\Delta\tau$, where the mean link delay τ_0 is assumed known. We then apply our previously described method treating all links as if they had the same delay τ_0, and assess the results as a function of the link delay heterogeneity, as characterized by the fractional spread $2\Delta\tau/\tau_0$ of the link delays. We do this for all networks listed in Fig. 3.4. For purposes of discussion, we divide these networks into two categories: (1) networks for which we obtained perfect inference with homogeneous delays, and (2) networks for which synchronization hindered link inference performance with homogeneous delays (Fig. 3.6(a)). For a fixed mean delay time τ_0 (corresponding to $k = 34$ with $\tau_0 = k\Delta t$), the results for different amounts of spread of the delay times $\Delta\tau$ are plotted in Fig. 3.10. The results indicate that in case (1), we continue obtaining good results, with the average of maximum number of false positives around 1, if the heterogeneity of the spread in delays is not too large (Fig. 3.10(a)). In case (2), we obtain better, and, in some cases perfect (i.e., zero false positives), results with moderate delay heterogeneity. This improvement can be attributed to a change in network dynamics: The heterogeneity of the link delays inhibits global synchronization, as is evident from the corresponding synchronization error plot of Fig. 3.10(b). Thus, moderate delay heterogeneity can be beneficial to the working of our method. This study confirms that our formalism can be applied to realistic networks with distributed link delay time, broadening the scope of our methodology. Further investigations in which we include the delay heterogeneity in the reservoir computer formalism itself are reserved for the future.

3.6 Discussion

In this work, we developed a reservoir-computer-based technique for the general problem of link inference of noisy delay-coupled networks from their nodal time series data and demonstrated

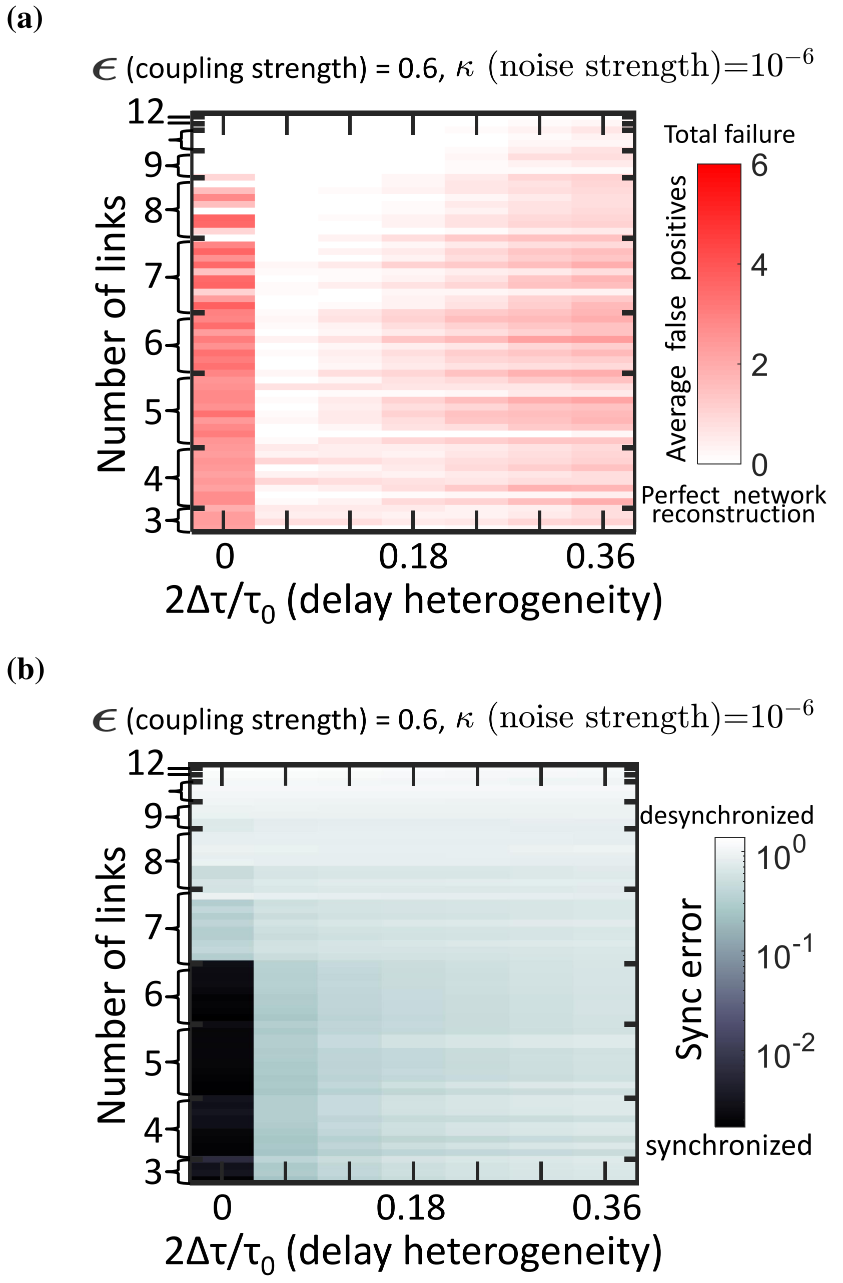

Figure 3.10: Tests on networks with heterogeneous link delays. (a) Average number of false positives, and (b) the corresponding synchronization error for different networks listed in Fig. 3.4, with $\epsilon = 0.6$ and $\kappa = 10^{-6}$, and different amounts of link delay heterogeneity, presented as the ratio of link delay variation range $2\Delta\tau$ and mean link delay τ_0. The results are based on simulated data with different network configurations, and different random link delays along the network links. In each case, the averaging is done over 100 different random realizations of reservoir connections, initial conditions and assignments of interaction delays to different links.

the success of our method on simulated and experimental data from opto-electronic oscillator networks with identical and distributed link delay times. Our main findings are as follows:

- Testing on experimental and simulated time-series datasets from networks, we found that, in the absence of dynamical noise, our method yields extremely good results, as long as there is no synchrony in the system.
- We found that dynamical noise and/or a moderate amount of link time delay heterogeneity can greatly enhance the performance of our method when synchrony is present provided that the noise amplitude or link time delay heterogeneity is large enough to perturb the synchrony.
- Since dynamical noise is ubiquitous in natural and experimental situations, we anticipate that this technique may be useful in network inference tasks relevant to fields like biochemistry, neuroscience, ecology, and economics.

Among the important issues for future investigation, our work in this chapter could be extended to cases when the dynamics of the network nodal states are partially synchronized (e.g., cluster synchronization of nodes [144]) or display generalized synchronization [86, 87, 145, 146]. Effects of network symmetries [144, 147], non-uniform coupling [148], and non-identical nodes [149] —all of which can affect the synchrony of nodal states —would also be very interesting to study. Another important issue that awaits study is the effects of incomplete [150], or erroneous nodal state data [151, 152] on link inference , e.g., a case of particular interest is that where measured nodal time series is only available from a subset of $N' < N$ of the N network nodes, and the value of N itself is unknown.

3.7 Acknowledgements

The authors would like to thank Thomas E. Murphy for his efforts to enable the experiments to be performed at the University of Maryland, College Park in the time of COVID-19. This work was supported by the U. S. National Science Foundation Grant DMS 1813027 (AB and EO) and the Office of Naval Research Grant N000142012139 (RR).

3.8 Appendices

3.8.1 Determination of the time-delay from cross-correlation

In this Appendix, we demonstrate that the duration of the delay along a link in our network can be accurately estimated from the cross-correlation between the measured time series of the two nodes connected by that link. In particular, we show that the location of the peak of the cross-correlation between the two nodes provides a good estimate of the delay time. We also show that the cross-correlation cannot determine causality, because it cannot determine the direction of a given putative link, or if the link even exists at all.

Consider the network depicted in Fig. 3.11a, where each node is an opto-electronic oscillator as described in Sec. 3.4. The delay in each link is $\tau = 1.44$ms. We define the cross-correlation between the sampled time series of two nodes i and j as

$$\rho_{ij}(\mathrm{lag}) = \frac{1}{\sigma_i \sigma_j} \sum_k x_i[k + \mathrm{lag}] x_j[k], \tag{3.14}$$

where $x_i[k]$ is the measured time series of node i at discrete time k and σ_i is the RMS value of x_i. The time series should be mean-subtracted so that $\langle x_i \rangle = 0$. The time series used here were

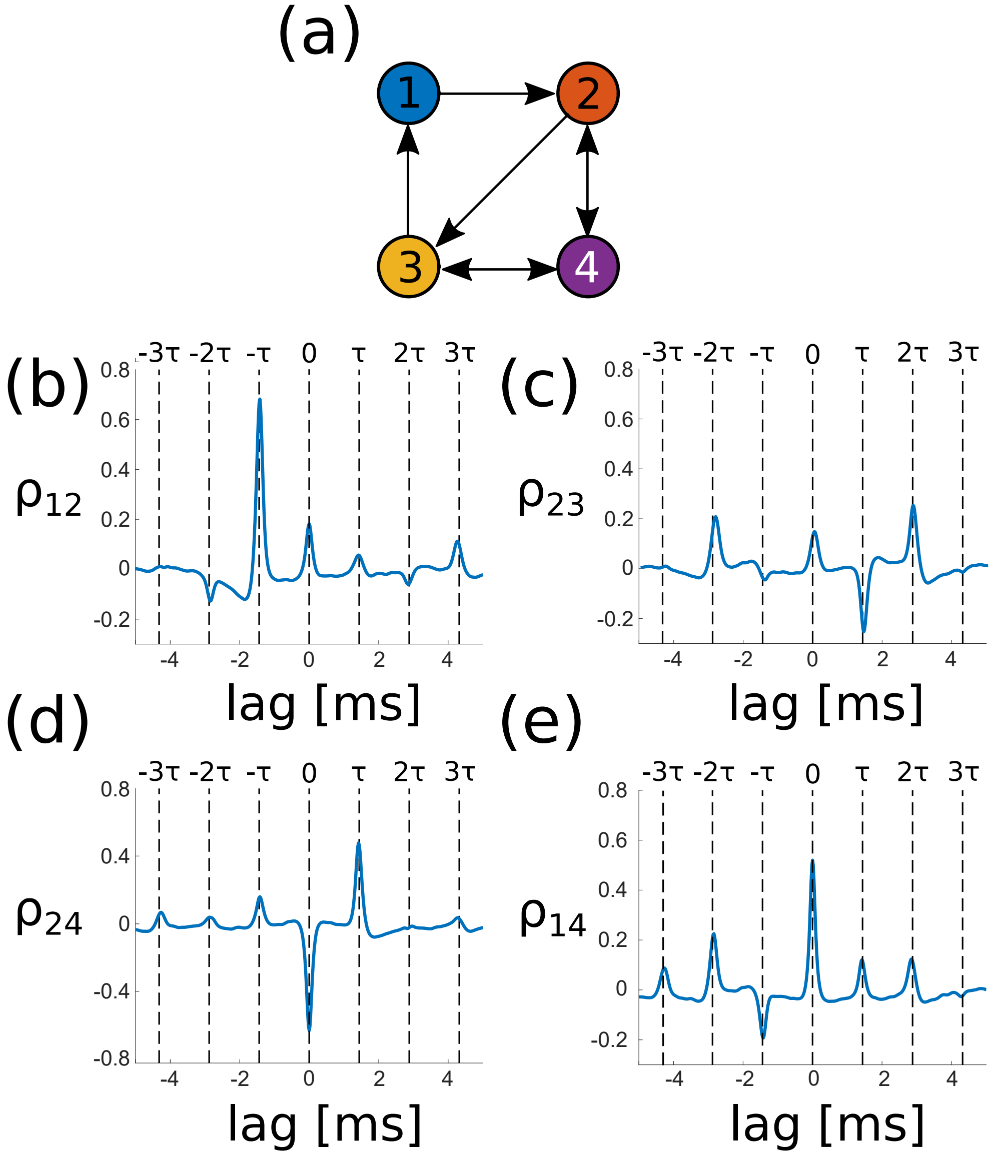

Figure 3.11: Time delay determination by cross-correlation of measured time series. In all cases, the coupling delay time is 1.44ms. (a) The network to be inferred from time series measurements. (b) The cross-correlation between the time series of nodes 1 and 2. A strong peak is observed near -1.44ms. (c) The cross-correlation between the time series of nodes 2 and 3. A strong negative peak is observed near 1.44ms and a strong positive peak near -1.44ms. (d) The cross-correlation between the time series of nodes 2 and 4. A strong peak is observed near 1.44ms. (e) The cross-correlation between the time series of nodes 1 and 4. Peaks are observed at 1.44ms and -1.44ms, even though there is no link from node 1 to node 4.

obtained from experimental measurements of our opto-electronic oscillator network.

First, we compute ρ_{12}, the cross-correlation between node 1 and node 2, shown in Fig.

3.11b. A peak is located at -1.44ms. This corresponds to the delay in the link from node 1 to node 2. This suggests that the dynamics of node 2 lag behind the dynamics of node 1, as one might expect since the delayed link is from node 1 to node 2.

Next, we compute ρ_{23}, the cross-correlation between node 2 and node 3, shown in Fig. 3.11c. The largest (negative, in this case) peak is located at 1.44ms, correctly identifying the absolute value of the delay time of the link from node 2 to node 3. However, in contrast to ρ_{12}, the peak location in ρ_{23} shows that node 3 leads node 2. There is no peak at a lag of -1.44ms. This shows that the cross-correlation can identify the delay time, but not the link direction.

We now consider ρ_{24}. Nodes 2 and 4 have a bidirectional link; however, the cross-correlation ρ_{24} shown in Fig. 3.11d has a prominent peak at 1.44ms but not at -1.44ms. There is no indication that the link is bidirectional.

Finally, we consider ρ_{14}. There is no direct link between nodes 1 and 4. Still, the cross-correlation ρ_{24} shown in Fig. 3.11e has peaks at both 1.44ms but at -1.44ms.

This example demonstrates that the cross-correlation can provide an accurate estimate of the duration of the delay in the coupling between two nodes, but that it does not provide sufficient information to determine the existence or directionality of a link. We find similar results in all the networks of opto-electronic oscillators we tested.

3.8.2 Derivation of the discrete-time equation for simulating the opto-electronic system

In this appendix, we derive the discrete-time equations implemented by the DSP board in our experimental setup and used in our simulations. These discrete-time equations are derived using standard techniques for approximating an analog filter as a digital filter and are essentially a trapezoid rule approximation to Eqs. (3.10) and (3.11).

The derivation here closely follows that presented in Ref. [136]. The missing details from Ref. [136] are filled in here, drawing from Ref. [153] for the details of the z-transform and bilinear transform.

The continuous-time filter equations that describe a two-pole bandpass filter are

$$\frac{d\mathbf{u}(t)}{dt} = \mathbf{E}\mathbf{u}(t) - \mathbf{F}r(t) \tag{3.15}$$

$$x(t) = \mathbf{G}\mathbf{u}(t) \tag{3.16}$$

where

$$\mathbf{E} = \begin{bmatrix} -(\omega_L + \omega_H) & -\omega_L \\ \omega_L & 0 \end{bmatrix}, \qquad \mathbf{F} = \begin{bmatrix} \omega_L \\ 0 \end{bmatrix}, \qquad \text{and} \ \ \mathbf{G} = \begin{bmatrix} 1 & 0 \end{bmatrix}. \tag{3.17}$$

Here, $\mathbf{u}(t)$ is a 2-vector that describes the state of the filter, $r(t)$ is the filter input, and $x(t)$ is the filter output. In the case of one of our opto-electronic oscillators $r(t) = \beta\cos^2(x(t-\tau) + \phi_0)$. In order to implement this filter digitally, one derives the digital filter equations by computing the transfer function of the analog filter, then applying the bilinear transform with frequency pre-warping to the continuous-time transfer function to obtain the discrete-time transfer function. From there, the discrete-time digital filter equations can be written down.

The transfer function $H(s)$ of the analog filter can be found by $H(s) \equiv X(s)/R(s)$, where the capital letters X and R indicate the Laplace transform of x and r, respectively. We compute the Laplace transform of Eq. (3.15):

$$s\mathbf{U}(s) = \mathbf{E}\mathbf{U}(s) + \mathbf{F}R(s) \tag{3.18}$$

Then, performing the Laplace transform of Eq. (3.16) and inserting Eq. (3.18), we have:

$$H(s) \equiv \frac{X(s)}{R(s)} = \mathbf{G}(s\mathbf{I} - \mathbf{E})^{-1}\mathbf{F} = \frac{s\tau_H}{(1+\tau_L s)(1+\tau_H s)}, \tag{3.19}$$

where $\tau_H = 1/\omega_H$ and $\tau_L = 1/\omega_L$. Equation (3.19) is the continuous time filter transfer function for the filter described by Eqs. (3.15)-(3.17).

Two standard tools used in the design and analysis of digital filters are the z-transform and the bilinear transform. The z-transform is the discrete time analog of the Laplace transform. The bilinear transform is a tool used to turn a continuous-time filter into a discrete-time filter. It can be shown that the result obtained by the bilinear transform method we use here is equivalent to applying the trapezoidal integration rule to Eqs. (3.15)-(3.17) [153].

The z-transform is defined as

$$Z\{x[n]\} \equiv \sum_{n=-\infty}^{\infty} x[n]z^{-n}, \tag{3.20}$$

where z is a continuous complex variable, and n is discrete time. One important z-transform relations is that a delay by m time steps in the discrete-time domain is equivalent to multiplication by z^{-m} in the z-domain.

The bilinear transform is used to convert our continuous-time filter transfer function (Eq. (3.19)) into a discrete-time filter transfer function. An exact conversion is done by discretizing with a time-step of T and equating $z = e^{sT}$. Since T is small, we can approximate

$$s = \frac{1}{T}\ln(z) = \frac{2}{T}\frac{1-z^{-1}}{1+z^{-1}}. \tag{3.21}$$

Equation (3.21) is the bilinear transform. This approximation is equivalent to applying the trapezoid rule to the continuous-time filter equations [153]. When Eq. (3.21) is substituted into

Eq. (3.19), we obtain the transfer function for a discrete-time filter with similar characteristics to the desired analog filter:

$$H(z) = \frac{1}{4}(1 - z_L)(1 + z_H)\frac{1 - z^{-2}}{(1 - z_L z^{-1})(1 - z_H z^{-1})}. \tag{3.22}$$

This change of variables is a nonlinear mapping, so frequency warping occurs. This effect is minimal when the frequencies are significantly less than the Nyquist frequency (in this case $f_L = 2.5$kHz and the Nyquist frequency is 12kHz) and can be further mitigated by pre-warping the frequencies of the continuous-time filter by $\Omega = \frac{2}{T}\tan(\frac{\omega}{2})$, where Ω is the discrete-time frequency and ω is the continuous-time frequency [153]. Therefore, we find that

$$z_H = \frac{1 - \tan(T/2\tau_H)}{1 + \tan(T/2\tau_H)}$$

and

$$z_L = \frac{1 - \tan(T/2\tau_L)}{1 + \tan(T/2\tau_L)}.$$

Now, one can use the definition of the transfer function $H(z) \equiv X(z)/R(z)$ to find

$$(1 - (z_L + z_H)z^{-1} + z_L z_H z^{-2})X(z) = \frac{1}{4}(1 - z_L)(1 + z_H)(1 - z^{-2})R(z). \tag{3.23}$$

We arrive at the discrete-time filter equation by performing the inverse z-transform on Eq. (3.23):

$$x[n] = (z_L + z_H)x[n - 1] - z_L z_H x[n - 2] + \frac{1}{4}(1 - z_L)(1 + z_H)\left(r[n] - r[n - 2]\right). \tag{3.24}$$

For the filter used in this work, $z_L + z_H$=1.4845, $z_L z_H$=0.4968, and $\frac{1}{4}(1 - z_L)(1 + z_H)$=0.242.

Chapter 4: Network inference application for *C. elegans* neuronal calcium fluorescence dynamics

This chapter is based on the work "Network inference from short, noisy, low time resolution, partial state measurement time series: with application to C. elegans neuronal calcium fluorescence dynamics" by Amitava Banerjee, Sarthak Chandra, and Edward Ott, submitted for publication.

4.1 Overview

In the previous chapters, we saw that reservoir computer (RC) models of experimental and simulated time-series data can be used to infer both short term and time-delayed interactions in complex systems. In typical cases, however, such time-series data may be subject to limitations, including limited duration, low sampling rate, observational noise, and partial nodal state measurement. While previous chapters showed that network inference benefits from dynamical noise and suffers from observational noise, it is generally unknown how the performance of link inference techniques on such datasets depend on different experimental limitations of data acquisition. In this chapter, we utilize both synthetic data generated from coupled chaotic systems, as well as experimental data obtained from *C. elegans* neural activity to assess the influence of data limitations on the effectiveness of three network inference techniques: Granger causality, transfer entropy, and, RC-based method.

Network inference techniques typically assign scores to each potential link in a network. Under *ideal* circumstances these scores directly lead to clear discrimination between linked and unlinked nodes in the network (for example, as was the case in Fig. 3.8b). However, the measured data generally available from *non-ideal*, real-world systems typically consists of short noisy time-series with poor temporal resolution and may have other limitations, all of which may be expected to lead to poor link discriminability (for example, as was the case in Fig. 3.8c, where the score distributions corresponding to linked node pairs and non-linked node pairs overlapped). Focusing on Granger causality, transfer entropy, and the newly developed RC-based link inference technique, we characterize scenarios that lead to poor link discrimination through tests on synthetic data generated from coupled chaotic systems, as well as experimental neuronal activity data obtained from *C. elegans* for which the ground truth neuronal synaptic connectivity network is known.

Furthermore, in cases when score distributions corresponding to linked node pairs and non-linked node pairs overlap, we demonstrate that appropriate surrogate data can be generated to ascertain statistical confidence levels associated with the results of link inference techniques, thus, potentially allowing impactful application of these techniques on real-world datasets.

4.2 Introduction

The task of reconstructing complex networks solely from observations of their nodal state time-series dynamics has applications in a wide range of problems. Examples include inferring neuronal networks from neural recordings (Fig. 4.1(a-c)) [19, 156], predator-prey interaction networks from population data of different species in an ecosystem [23], gene and protein

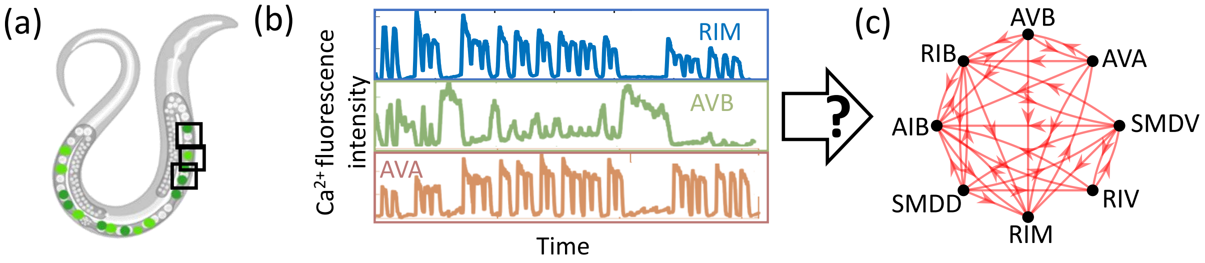

Figure 4.1: Schematic of the network inference task from calcium fluorescence dynamics of individual *C. elegans* neurons showing (a) identified neurons in a live worm (b) representative full time series available from three identified neurons from [154] (c) The 8-neuron ground truth synaptic network of *C. elegans* from [155].

interaction networks from protein concentration and gene expression data of biochemical systems [157], climatic interaction networks from global temperature and ocean circulation data [158, 159], etc. A number of computational techniques have been proposed for this purpose, e.g., Granger causality [35, 36, 160], transfer entropy [31, 161–163], Bayesian techniques [39, 164], machine-learning-based techniques [59, 60, 75, 165, 166], among others [21, 167, 168]. Typically these techniques take the time-series of the nodal states of the unknown network as input, and use it to assign to each potential directed link between a pair of nodes (i, j) of the network a link score $S_{i\to j}$ that is supposedly reflective of the probability of existence of an actual network connection from node i to node j. Ideally these network inference techniques would yield a bimodal distribution of the link scores $S_{i\to j}$, with linked node pairs having relatively larger scores, and unlinked node pairs having relatively smaller scores, with a *clear gap* in between these two types of scores (Fig. 4.2(a)), so that linked and unlinked node pairs are easy to separate based solely on their respective link scores. We refer to the distribution of scores between linked node pairs as the link-score distribution, and the distribution of scores between unlinked node pairs as the non-link-score distribution. In practice, however, it is often the case that the link-score distribution and the non-link-score distributions overlap (Fig. 4.2(b)). In such cases, it becomes useful to (i) understand the relationships between experimental factors characterizing the measurements of the network nodal state time-series, link inference techniques, and the distribution of link scores, and (ii) to make better inference from the results of network inference link scoring. To these ends we conduct tests on a newly-developed machine learning link inference technique [59], the Granger causality technique [35, 36], and an information theoretic technique, Transfer Entropy [31, 163]. The first method, developed in Refs. [59, 60], involves training an artificial neural network called a "reservoir computer" with the measured time-series and then using the trained reservoir as an *in silico* model of the experimental system to evaluate the network structure between the nodes of the experimental system. For more details on these three methods, see the Methods Section 4.8.

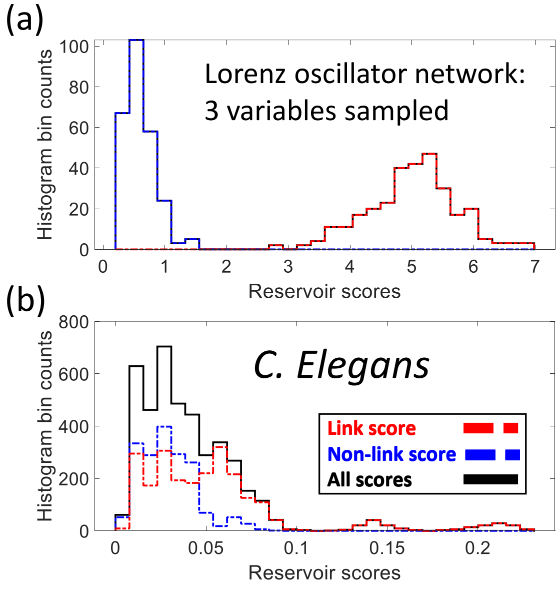

Figure 4.2: For the 8-neuron network in Fig. 4.1(c), two examples of link score distributions with (a) separated scores from Lorenz oscillator time-series with with $T = 3000, \sigma_{obs} = 0.01, \Delta t = 0.02, \delta t = 0.02$ and complete nodal states and full network sampled and (b) non-separated scores from *C. elegans* calcium fluorescence dynamics from [154].

To test the validity and efficacy of these techniques it is essential to apply them in a scenario where the ground truth of the underlying network is known. Thus, we focus our attention on two types of network dynamical system. One is the neural system of the *C. elegans* worm, a model organism that we choose due to the availability of its complete neural connectome as obtained from previous microscopy studies [155, 169], in addition to whole-brain calcium fluorescence imaging of individual neurons [154, 170, 171]. The other dataset is obtained by numerical simulations in which we take each node to be a Lorenz oscillator [92]. For the Lorenz oscillator case we simulate and compare results from two different kinds of nodal state time series measurements data: (i) the entire three-component vector state of each node is assumed to be measured, and (ii) we take as the nodal observations a single composite variable per node (representative of the case of partial state measurements of nodal states).

As we will show, the overlap between the link-score distribution and the non-link-score distribution can have various causes, including: (1) low temporal resolution of sampling (e.g., $\sim$3 samples/second for the *C. elegans* time-series that we use, examples shown in Fig. 4.1(b)); (2) the presence of observational noise; (3) short length of the available time series data (e.g., ~ 0.3 second × 3000 time points for the *C. elegans* time-series in [154], Fig. 4.1(b)); (4) availability of only partial, indirect measurements (e.g., instead of accessing the full biophysical state of each neuron, we might only have measurements of the fluorescence intensity of the Calcium ions involved in the neural activity); among others (e.g., the presence of strong synchrony or correlations among the activities of different neurons, making connectivity inference prone to spurious false positive connections between neurons with correlated activities [59, 60]). In Sec. 4.4, we illustrate the effect of such factors on the separability of the distribution function of link and non-link scores in the network Lorenz model with chaotic dynamics when examined using the example of the machine learning network inference method of Ref. [59]. In Sec. 4.5, we consider a method of generating surrogate time-series data, in which causal relationships in the experimental data between a given pair of nodes are destroyed, and we demonstrate that such surrogate data can be used to estimate null-hypothesis score distributions, which mimic the distributions of the non-link scores of the original time-series data [172–179]. We find that (Sec. 4.6), in the situation where the link-score and non-link-score distributions overlap, the surrogate data is able to give an estimate of the p-value associated with any particular node pair being linked in the underlying network. Choosing an appropriate p-value as a link-score cutoff can then be used to partially reconstruct an estimate of the unknown network at a confidence level corresponding to the chosen p-value cutoff.

The main results/contributions of this chapter are as follows:

1. We show that network link inference techniques can give useful prediction with statistical confidence estimates based on appropriately constructed surrogate data, when they are applied on short, low sampling resolution, noisy time-series data from partial nodal state measurements.

2. We systematically study the effects of training time length, observational noise variance, sampling resolution, and incomplete nodal state measurements on the performance of network inference.

3. A key feature enabling use of *C. elegans* data for our study is the availability of its known ground-truth synaptic network.

4.3 Physical systems and datasets used

4.3.1 Calcium imaging time-series data from *C. elegans*

To test the performance of techniques for inferring neural connectomes from neural recordings, we apply them to the publicly available [154] dataset of whole-brain calcium-imaging time-series of individual neurons of a freely-moving *C. elegans* worm. We then compare our inferred connectivity among the identified neurons to the well-established ground-truth connectome for *C. elegans* available from [155, 169]. While data is available for all neurons, a large fraction of the neurons have low magnitudes of activity over the duration of neural recording. Thus, we focus on the subnetwork consisting of the 16 most active motor neurons and the synaptic connections between them. Furthermore, due to the left-right symmetry of the *C. elegans* worm, the network of neuronal connections of these 16 neurons is left-right symmetric [155, 169]. Thus, we consider a "folded-over" network of connections between the corresponding 8 left-right neuron pairs. These pairs are conventionally labled AIB, AVA, AVB, RIB, RIM, RIV, SMDD, and SMDV. We take the link score of each potential connection in the folded network to be the average over the corresponding left and right connections in the original 16 node network. Considering the folded-over network, our ground-truth network has 8 nodes, with 30 directed pairs of linked nodes and 26 pairs of unlinked nodes (Fig. 4.1(c)). We henceforth refer to this 8-node network as the Folded *C. elegans* network.

4.3.2 Coupled Lorenz oscillator network simulations

In order to test the effect of different experimental and sampling conditions, such as sampling frequency, observational noise, etc., we also generate time-series data from a set of 8 Lorenz oscillators, coupled using the same network topology as the Folded *C. elegans* network. The equations describing our example network of Lorenz oscillators reads:

$$\frac{dx_i}{dt} = -\alpha x + (\alpha/2)\sum_j A_{ij}y_j + \sigma_{Dyn}\xi_i^x(t), \tag{4.1}$$

$$\frac{dy_i}{dt} = x_i(\rho_i - z_i) - y_i + \sigma_{Dyn}\xi_i^y(t), \tag{4.2}$$

$$\frac{dz_i}{dt} = x_i y_i - \beta z_i + \sigma_{Dyn}\xi_i^z(t), \tag{4.3}$$

for $i = 1, 2, ..., 8$, where (x_i, y_i, z_i) constitute the nodal state of the i-th node, and, A_{ij} is a binary adjacency matrix corresponding to the Folded *C. elegans* network (network in Fig. 4.1(c)). The coupling, represented by the second term on the right side of Eq. (4.1), is from the y variable of node j to the x variable of node i. The parameters α and β are chosen to have the same value used by Lorenz, $\alpha = 10$ and $\beta = 8/3$, and for each node i, the parameter ρ_i is sampled uniformly from the range [30,70] to increase heterogeneity across nodes. The dynamical noise strength parameter is $\sigma_{Dyn} = 10^{-3}$, and the noise is assumed to be white, $\langle \xi_i^p(t)\xi_j^q(t')\rangle = 2\delta_{pq}\delta_{ij}\delta(t-t')$. Thus, in this setup, the y-variable of node j directly affects the x-variable of node i if $A_{ij} = 1$. Starting from a random initial condition, we integrate the system using a 4-th order Runge-Kutta method with step size $\delta t = 0.02$ and discard the initial transients. In addition, to mimic realistic sampling, we add white Gaussian observational noise of mean zero and standard deviation σ_{Obs} independently at each sampled data point, and sample the time-series of the system with a time-step Δt (which is chosen to be a multiple of the Runge-Kutta simulation time-step δt). We consider two modes of measurements of the dynamics: (i) the case of full nodal state measurements, where the full state (x_i, y_i, z_i) is sampled from each of the nodes. In this case, we calculate link scores from y_i to x_j as the relevant quantity of interest that measures the strength of the connection from node i to node j. (ii) A single composite variable, $s_i = x_i^2/\langle x_i^2 \rangle_t + y_i^2/\langle y_i^2 \rangle_t + z_i^2/\langle z_i^2 \rangle_t$, is measured, where $\langle . \rangle_t$ denotes the time average of the respective variable.

4.4 Link-score distributions for the coupled Lorenz Oscillator Network

As shown in Fig. 4.2(b) and the next section, the *C. elegans* data that we use yields link-score distributions that strongly differ from what we call the ideal case, shown in Fig. 4.2(a), and instead are more like Fig. 4.2(b). Thus we were motivated to ask why this happens. This section addresses this question through an investigation of the Lorenz network model system of Sec. 4.3.2.

We assume that the states of the dynamical systems are sampled at a time interval Δt at times $n\Delta t$ with $n = 1, 2, \dots, T$. In particular, we investigate the effects of sampling time Δt, length of time series $T\Delta t$, and number of sampled variables on the distribution of link inference scores. (For *C. elegans* analogous parameters are fixed by the dataset we considered, and are largely determined by the experimental setup used for measuring the calcium fluorescence data). Our main results are summarized in Figs. 4.3(a-o) which plot histograms of scores obtained using the machine learning technique. In each panel of Fig. 4.3, the histogram plotted in black is for the scores of all directed node pairs, the histogram plotted in red is for the scores of directed node pairs that are linked, and the histogram plotted in blue is for directed node pairs that are not linked. (The histograms plotted in green will not be discussed in this section, but will be discussed in Sec. 4.5).

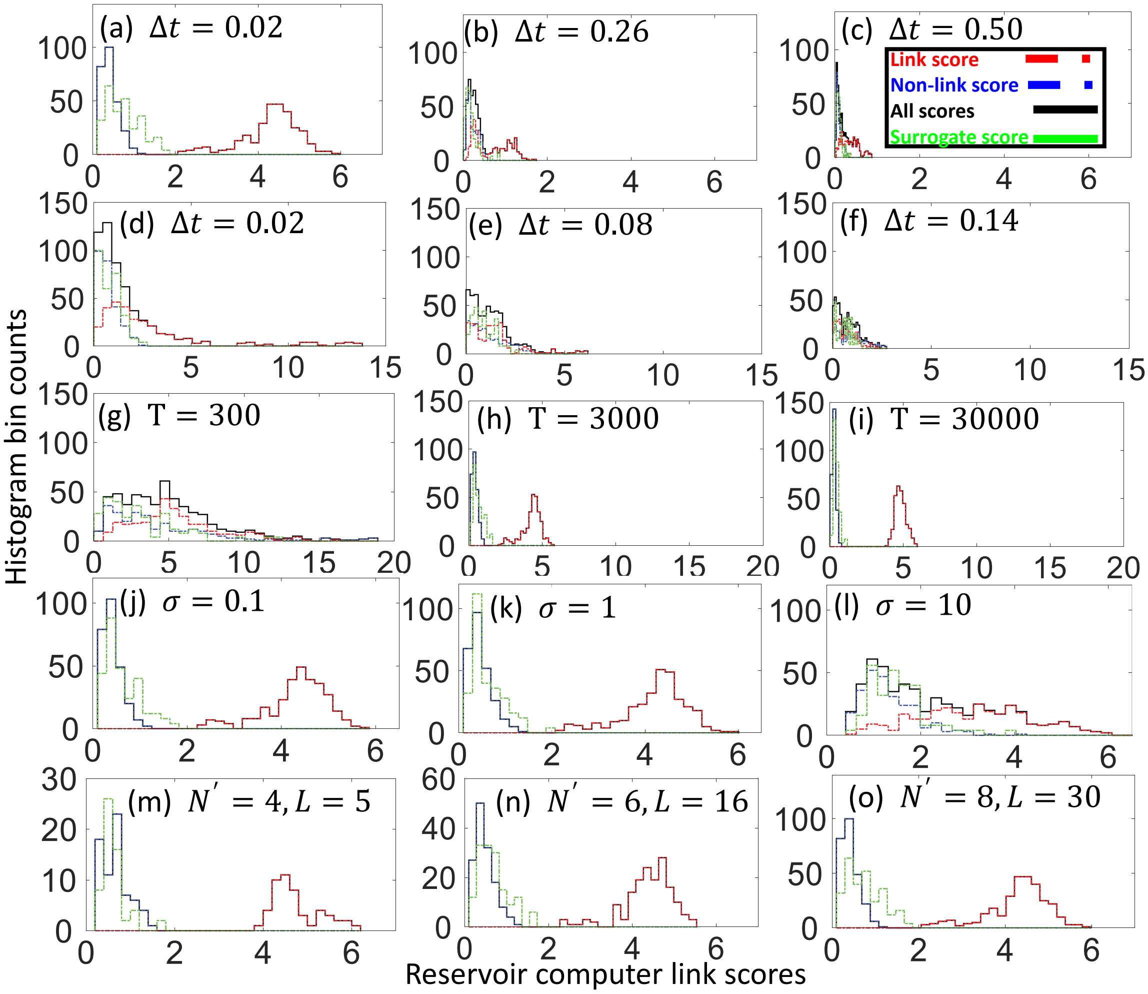

Figure 4.3: Number of directed network node pairs vs their scores for all such pairs (plotted in black), for pairs corresponding to actual links (plotted in red), for pairs not corresponding to links (plotted in blue), and for the surrogate data (plotted in green) from the coupled Lorenz oscillators on the network shown in Fig. 4.1(c), with (a-c) 3 variables and (d-f) 1 variable per node sampled, with $T = 3000, \delta t = 0.02, \sigma_{obs} = 0.01$; and 3 variables per node sampled, with (g-i) $\Delta t = 0.02, \delta t = 0.02, \sigma_{obs} = 0.01$, (j-l)$\Delta t = 0.02, \delta t = 0.02, T = 3000$, and (m-o) $\Delta t = 0.02, \delta t = 0.02, T = 3000, \sigma_{obs} = 0.01$.

We start with the parameters $T = 3000$, $\sigma_{Obs} = 0.01$,with the data sampling time interval Δt initially chosen to be the same as the numerical time-step $\delta t = 0.02$ and with all the 3 variables of each Lorenz oscillator are assumed to be measured. Figure 4.3(a) shows a typical histogram approximation of the link-score distribution for this example network obtained by the application of the machine learning technique described in the Methods section [59]. In constructing the histogram, we used 10 random realizations of the reservoir computer (see Methods Sec. 4.8). We observe from Fig. 4.3(a) that we get an effectively "perfect" separation between the link-score and non-link score distributions (i.e., a distribution with a clear gap between the low-score component and the high-score component), which yields perfect network inference. We will henceforth refer to this case as the 'canonical parameter case'.

To see how the network inference performance varies with change of parameters away from our canonical parameter case, we first decrease the sampling rate by increasing the sampling time Δt for the system, keeping the number of sampled points T fixed. We find that with increasing sampling times, the magnitude of the scores diminish, and the separation between the link-score and non-link-score distributions is reduced, resulting in progressive deterioration of the performance of the link inference technique (Figs. 4.3(b) and 4.3(c)). We interpret this reduction in performance with increasing Δt to arise due to the causal effect of state changes at a node spreading to other nodes to which it is not directly connected by a single link. That is, with larger Δt the causal effect of a state change at one node can propagate to another node via multiple-link paths. This effectively yields a larger-than-expected score for disconnected node-pairs, leading to a merger of the non-link-score distribution with the link-score distribution. In contrast at the lower Δt value (Fig. 4.3(a)), we postulate that such a causal effect only has time to travel one link, and, thus, a causal effect of a state change at time t at node i on the state of node j at time $t + \Delta t$

implies a network link $i \rightarrow j$.

We next study the case where we sample only a single variable (as described in Sec. 4.3.2) from each Lorenz oscillator. In this case, we do not find any parameter regime where we obtain a score distribution (black curves in Fig. 4.3(d-f)) where the distribution is bimodal. Since, in applications, the black curve is all that is observed (the link-score [red] and non-link-score [blue] distributions are unknown to an experimenter), based on only the black curve of Fig. 4.3(d-f), link inference is not possible. On the other hand, from Fig. 4.3(d), taking into account the red and blue plots, we see that the scores in the large-score tail of the black-plotted distribution predominantly correspond to actual links. However, as Δt is increased (Figs. 4.3(e) and 4.3(f)), this feature becomes weaker. This behavior suggests that, as we will soon discuss, there may be potential for extracting useful information from measurements like those resulting in Fig. 4.3(d).

We also vary the length T of the training time series, and the standard deviation of observational noise σ_{Obs} for the case where all 3 variables are sampled. As might be intuitively expected, the link inference performance progressively improves with increase of T (Figs. 4.3(g-i)); and deteriorates with increasing σ_{Obs} (Figs. 4.3(j-l)) due to increased spurious features in the training data arising form the observational noise.

Our previous results [59, 60], as well as results of others [66, 71, 141–143], show that the strength of dynamical noise (σ_{Dyn}) present in the system is another important factor in determining the nature of the link score distribution. A main result of the previous work is this previous work is that, in contrast to observational noise, dynamical noise can have a positive effect on network inference. This happens because the dynamical noise itself generates perturbations of the nodal states of the system which propagate to other linked nodes to aid link inference (e.g., see Refs. [66, 143]). Because the effect of varying dynamical noise has been previously considered [59], we will not investigate dependence of our score distributions on σ_{Dyn}.

Finally, we test the performance of our link inference technique in cases where we sample only a subset of the nodes in the full 8-node network of Fig. 4.1(c). To do this, for the canonical parameter case, we consider two cases, one in which we consider a subnetwork where we sample $N' = 4$ nodes and one in which we sample $N' = 6$ nodes. The sampled subnetworks in these two cases have $L = 5$, and $L = 16$ links respectively. In both these cases, we see from Fig. 4.3(m-o) that we get a perfect separation between link scores and non-link scores, showing that the method may be used even when we do not have a complete sampling of all the nodes as is often the case.

To summarize this section, we note that, in general, time series data available for use in link inference will typically be subject to limitations. In many such cases, application of a link scoring technique may confront us with score distributions, which, *by themselves*, cannot be used to infer links. However, as our examples from this section show, even in cases where score distributions are not bimodally separated (e.g., Fig. 4.2(b)), there may be valuable information in the score distribution that could help us obtain useful partial information about the network structure. This is because, in many cases where the score distributions of links and non-links are not well-separated, the tail of the total distribution may contain scores predominantly from actually linked nodes (e.g., Figs. 4.2(b), 4.3(d), 4.3(l)). Although, in practice, we only have access to the total distribution (black curves in Fig. 4.3), if we knew that the true links were concentrated sufficiently far into the tail of the distribution, we could use that to identify some fraction of the true links that occur in the tail. However, if the only information we have is the score histogram (black), we do not know that this is in fact the case. We thus desire some method of obtaining information regarding the true link score distribution, which would then determine how large a score cutoff one should choose before judging that a score in the tail most likely indicates a true link. In the next section, we address this crucial issue by use of surrogate data.

4.5 Statistical analysis of candidate link scores for link inference

4.5.1 Brief review of surrogate data generation techniques for network link inference

In order to estimate the statistical significance of inferred network links, a common method is to generate a surrogate time-series data. Surrogate time-series data is data that is synthetically generated to have, as much as possible, the same statistical properties as the original data but with any causal dependence of the truly linked directed node pair whose score is to be determined removed. Ideally, the surrogate data would then yield a set of scores whose distribution is similar to the distribution of the scores of non-linked variables in the original time-series data. Without prior knowledge of the underlying ground-truth network we cannot reconstruct the distribution of non-link scores directly from the original data; however, the distribution of scores calculated from an appropriate "Causality-destroyed surrogate data" (CDSD) can serve as a proxy for the non-link score distribution. Using this proxy distribution, statistical tools can then be applied to obtain the statistical significance of inferred network links. In particular, we can estimate a true-link p-value for any particular score s found from the causal time-series data, by calculating the fraction of the CDSD scores that are larger than s.

There exist several distinct methods for generating CDSD. A key early paper on surrogate data methods applied to analysis of time-series from dynamical systems is that of Theiler et al. [180], while (to our knowledge) the first application to neuronal systems was in the paper of Kamiński et al. [35]. Refs. [172–179] provide reviews on several such CDSD generation methods, and Refs. [173, 174, 179] compare their performances on causality detection and network inference in different settings involving multivariate time-series data. Some common techniques for generating CDSDs involve performing one of the following operations on time-series of individual variables: (i) time-shifting the time series of individual variables so as to destroy their causal relationships with time-series from other variables [173, 181], (ii) randomizing phases and amplitudes of the Fourier transform of the original time-series and then inverse Fourier transforming to generate surrogate data that preserves the individual variables' power spectra [180, 182], and (iii) drawing segments of the original time-series, each starting at a random time-point and having a random length, and joining them contiguously [173, 174, 183]. Some examples using surrogate data for inferring causal interaction and network connectivity from multivariate neural time-series data, are Refs. [184–189]. Refs. [161, 162, 172, 179, 190] use surrogate data to estimate statistical significance of their link scores and also evaluate the performance of their network inference technique using a ground truth network and simulated and experimental data from large-scale networks. In what follows we use the time-shifting method.

4.5.2 Method of surrogate time-series generation and statistical significance analysis of inferred connections

In order to generate the surrogate time series data for links incoming to a given node, we shift the time-series of that node, by an amount equal to half the total length of the time-series, and keep the time-series from all the other nodes in the network intact. That is, for an original time-series of length $2T$ from the selected node, $\{\mathbf{X}_1, \mathbf{X}_2, \mathbf{X}_3, ..., \mathbf{X}_{2T}\}$, with $\mathbf{X}_i$ denoting the nodal state vector $\mathbf{X}$ at the i-th time-point (e.g., $\mathbf{X} = [x, y, z]$ for a Lorenz oscillator node), the corresponding surrogate time-series will be $\{\mathbf{Y}_1, \mathbf{Y}_2, \mathbf{Y}_3, ..., \mathbf{Y}_{2T}\}$, where $\mathbf{Y}_i = \mathbf{X}_{(i+T)\mathrm{mod}2T}$ for $i \neq T$ and $\mathbf{Y}_T = \mathbf{X}_{2T}$. We then use the reservoir computer with the same setup and hyperparameters as used for analysis of the original time-series to calculate, from the surrogate data, scores corresponding to all candidate links incoming to the shifted node. We repeat this procedure for all nodes in the network. (Additionally, when using the machine learning technique (see Sec. 4.8.1), we also repeat this procedure ten times for each parameter value using different random reservoir computer initializations). Through this procedure of shifting the nodal time-series, the distribution of all scores generated from the surrogate data is expected to be a useful proxy for the non-link-score distribution from the original, unshifted time-series data. Fig. 4.3 show comparisons of the histograms of the two distributions.

4.6 Tests on Example Systems

4.6.1 Application to Lorenz network model

In this subsection we apply the above surrogate data procedure to the Lorenz model. The histograms plotted in green in Fig. 4.3 correspond to the distributions of surrogate scores for the various cases (panels (a-o)) of Fig. 4.3. (Note that in a real network inference situation, the ground truth network is unknown and the blue and red plotted curves are consequently also unknown.) These figures demonstrate that in cases where there is a partial overlap between the link-score and non-link-score distribution calculated from the causal data, the surrogate score distribution is in reasonable agreement with the non-link score distribution calculated from the causal time-series data.

We note however that this agreement between the non-link-score distribution and the surrogate score distribution appears to depend strongly on the choice of the regularization parameter of the reservoir computing technique. In the methods section, we describe a heuristic to choose an appropriate regularization parameter in a fashion that is agnostic to the underlying ground truth network.

In Fig. 4.3(b-c), we see that for the Lorenz case with all three variables sampled, for larger sampling time Δt, the scores calculated from the causal time-series and the surrogate scores decrease. In fact, the distributions of the surrogate data scores match the distribution of the non-link scores better as we increase Δt, and the link and non-link scores overlap. This shows that the surrogate scores are able to give a proxy of the non-link score distribution in situations where, due to lack of clear separation between the link and non-link scores, such proxies are needed the most. In each case, a significant fraction of scores in the tail of the total score distribution (black histograms in Fig. 4.3(a-d)) correspond to true links in the network, and appropriate score thresholds based on the surrogate data distribution can successfully identify a large fraction of them. For the case of $\Delta t = 0.02$ in Fig. 4.3(a), where there is a clear separation, the exact distributions of the surrogate scores and the non-link scores do not match very well. However, the link and non-link scores are already well-separated, diminishing the requirement for surrogate data in this case. Despite this, a score threshold based on the surrogate score distribution is indeed smaller than link scores and falls within the range of scores separating the link and non-link score distributions. Thus, in practice, the surrogate data continues to perform their intended function well.

Similar results are seen in Figs. 4.3(g-i) and 4.3(j-l), where we vary the observational noise strength (σ_{Obs}) and the training length (T) for the Lorenz network with three variables sampled. In Figs. 4.3(g) and 4.3(l), we observe overlapping link-score and non-link-score distributions.

However, in both cases, there is an enrichment of the density of link scores at the tails of the total score distributions (black histograms in Figs. 4.3(g), 4.3(l)). In both of these cases, comparisons between the surrogate and non-link score distributions show that the surrogate scores are able to generate a good proxy for the non-link score distribution. Thus score thresholds based on the surrogate distributions in each of these cases are able to identify link scores in the tail of the full distribution. Furthermore, as Figs. 4.3(h-i) and 4.3(j-k) show, in situations with relatively smaller observational noise, and relatively larger training length, when the scores separate, a cutoff based on the surrogate data falls within the separation range. Thus, a cutoff based on the surrogate data works well in all such cases as well. The same behavior occurs in Figs. 4.3(m-o), where the surrogate score distribution is shown to give a cutoff within the range separating the link and non-link scores.

For the cases where only one variable is sampled per Lorenz oscillator node (Figs. 4.3(d-f)), we again observe (See Fig. 4.3(d) with $\Delta t = 0.02$) that in the case where link and non-link score distributions overlap, the surrogate distribution matches well with the non-link score distribution, and the surrogate distribution can be used to choose a score threshold that will allow the link scores at the tail of the score distribution to be identified. For the cases with larger Δt, the link inference procedure itself is unable to give us any information about the network since the tail of the score distribution does not contain scores predominantly from links. However, the surrogate score distribution may nonetheless be a good proxy of the non-link score distribution, as is evident from the plots of Figs. 4.3(e-f).

4.6.2 Applications to *C. elegans* data

Next, we test the performance of our surrogate data for the time-series of *C. elegans* neurons that we described in Section 4.3.1. For the cases where we use our reservoir computing technique as the network link inference method, we also consider two cases: (i) where a large number of neurons (127 in the example we used, which is the total number of neurons from which time-series data was available in the dataset in [154]) are measured for the data used for training the reservoir computer (Figs. 4.4(a-b)), and (ii) when only the 8 neuron pairs (listed in Fig 4.1(c)) are assumed to be measured in the data to be used for training the reservoir computer (Figs. 4.4(c-d)).

The score results (black) are plotted in Fig. 4.4 where we simultaneously plot the link-score (red), non-link-score (blue) distributions obtained from the original data, and the surrogate score (green) distribution obtained from the corresponding CDSD. The surrogate score distribution can be used to estimate the p-value associated with each score s obtained from the original data, which would be the fraction of surrogate scores large than s. This is useful in practical scenarios where users of a link inference technique might have a pre-defined statistical confidence threshold for accepting inferred links, which could be quantified in terms of a p-value cutoff (e.g., selecting only those inferred links for which $p < 0.02$). In such cases, the correspondence between link scores and estimated p-values could be used to convert a p-value cutoff to a link score threshold, and a desired confidence level (e.g., $p < 0.02$) can be achieved by selecting only the inferred links whose link scores lie above that threshold. Following this procedure, for each such confidence level represented by a p-value cutoff, we get a set of acceptable inferred links having link scores lying above the score threshold corresponding to the p-value cutoff. For each such set of accepted links, we use the ground truth connectome data of *C. elegans*, shown in Fig. 4.1(c), to calculate

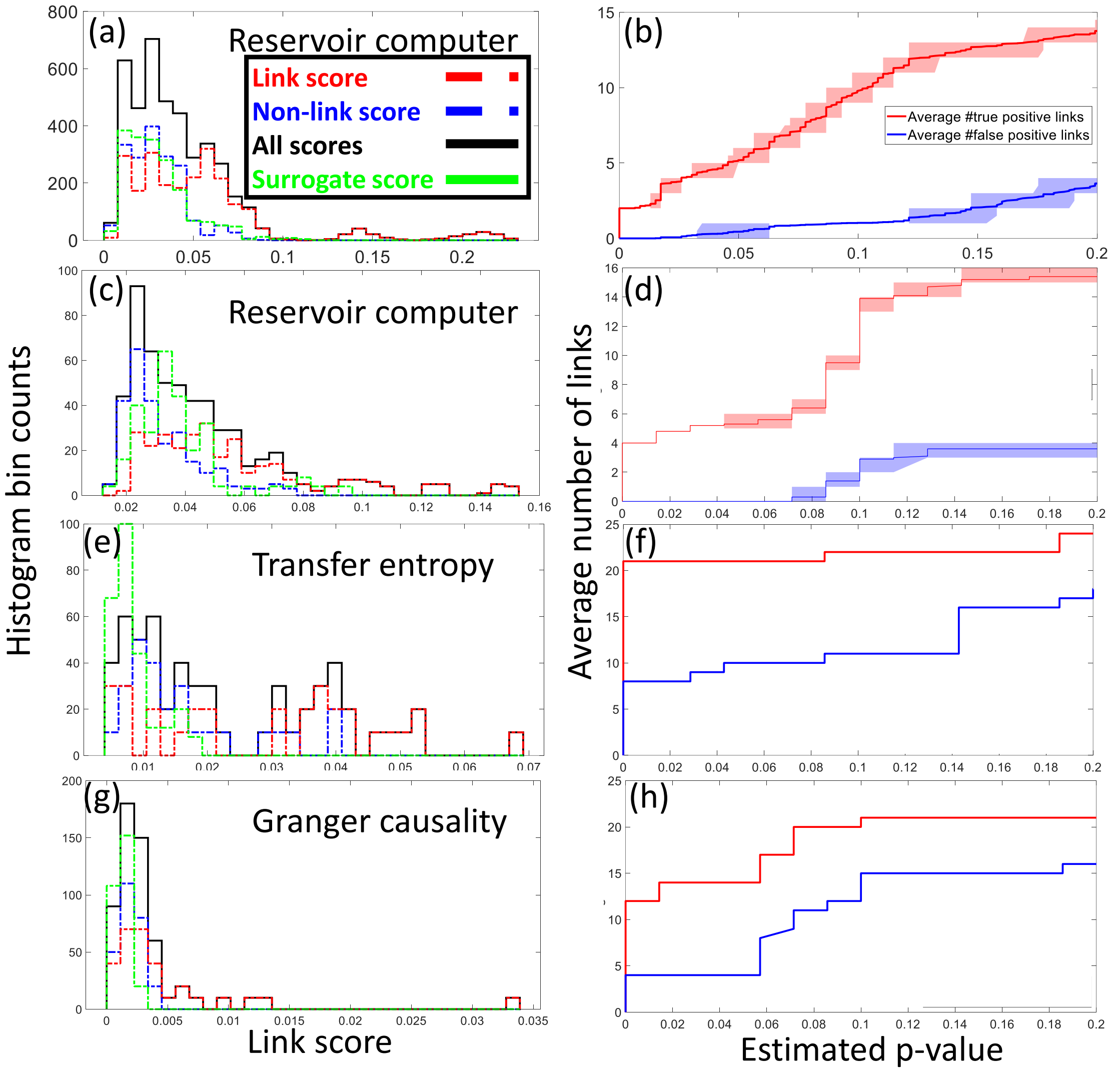

Figure 4.4: Distributions of scores for links, non-links, and surrogate data and the corresponding average number of true and false links at different p-value cut-offs for the network shown in Fig. 4.1(c), as inferred from the *C. elegans* calcium fluorescence time series of [154] with (a-b) all sampled neurons used and (c-h) with only 8 pair of neurons used, for different link inference techniques. In (b) and (d), the shaded region indicates the region between first and third quartiles for the distribution of average number of inferred true and false links for different random configurations of the reservoir computer.

the number of true positive link inferences and false positive link inferences. We plot these two numbers with the corresponding cutoff p-values (Figs. 4.4(b), 4.4(d)) for the two cases introduced earlier in this subsection.

From both Figs. 4.4(a) and 4.4(c), we see that actual links predominate in the tail of the high-score distribution. We also note, that surrogate score distributions, in both cases, provide us with a good estimate of the non-link score distribution obtained from the causal data. Furthermore, score cut-offs based on the surrogate data distribution successfully yield a significant number of true links and relatively few false positives (Fig. 4.4(b) and 4.4(d)). For example, Figs. 4.4(b) and 4.4(d) show that at the threshold p-value of 0.05, in the first case, we obtain 5 true links and less than one false link on average, while, in the second case, we obtain the same number of true links with no false links. As we increase the p-value cutoff threshold, both of Figs. 4.4(b) and 4.4(d) show that the number of correctly inferred links increases rapidly, while the number of false links increases at a slower rate. For example, even at a cut-off at $p = 0.2$, we obtain 14 true links and 4 false links in the first case, and 15 true link and 4 false links in the second case. Note that, in both the cases, the ratio of the number of true positives to the number of false positives (TP/FP), which is about 4 in both cases, is significantly higher than what would have been obtained through random assignment of links and non-links to candidate links: Given that the subnetwork of Fig. 4.1(c) has 30 links, and 26 non-links, such random assignments would correspond to the ratio (TP/FP) being approximately 1 on average. Furthermore, we note that, in both cases (Figs. 4.4(b) and 4.4(d)), the ratio of the number of false positives to the total number of links above the chosen score threshold is approximately equal to the estimated p-value. This is yet another consequence of the fact that the surrogate score distribution matches well with the non-link score distribution.

4.6.3 Transfer entropy and Granger causality

Next, we test the performance of transfer entropy and Granger causality as link inference techniques for the *C. elegans* time-series. In both cases, we use the same surrogate data generation technique as described in Section 4.5.2. The results are plotted in Figs. 4.4(e-h). Unlike the reservoir computing technique which yields an ensemble of scores for each candidate link with each score in the ensemble predicted by a different realization of the random reservoir connectivity matrix and input-to-reservoir coupling matrix, transfer entropy and Granger causality predict unique scores for each candidate link. Figs. 4.4(e) and 4.4(g) show that, in the case of transfer entropy and Granger causality, the surrogate distribution is narrower than the distribution of the scores for non-links. This is also reflected in Figs. 4.4(f) and 4.4(h), where we see that at a very small p-value cutoff, we obtain a large number of false positive links in both cases (9 for transfer entropy, and 4 for Granger causality). This suggests that the surrogate data generation technique that we used for the reservoir computer link inference may not be universally applicable to other link inference techniques.

4.6.4 Another Surrogate Data Generation Technique

Finally, we test a different surrogate data generation technique with all three of the link inference techniques considered in this chapter, namely, reservoir computing, transfer entropy, and Granger causality. The tests are done on the same *C. elegans* dataset as the one we used in our previous results and on the Lorenz network with one variable per node sampled with $\delta t = 0.02$ (same parameter regime as Figs. 4.3(d)). This surrogate data generation technique, known as amplitude adjusted Fourier transform (AAFT), was introduced in [180] and improved in [182], and involves Fourier transforms of the original time series data, followed by phase randomization, and finally an inverse Fourier transform. In our case, we apply AAFT to the time-series of one Lorenz network node (or one pair of neurons for *C. elegans*) at a time, keeping the time-series of other nodes intact. We then apply the link inference techniques, and, for each technique, we collect the link scores corresponding to links incoming to the node on which we applied the AAFT to generate our pool of surrogate scores. For the reservoir computing method, we chose the value of the regularization parameter according to our methodology described in the Methods section (Sec. 4.8.1). The results plotted in Fig. 4.5 show that, for the case of both Lorenz and *C. elegans* networks, the AAFT surrogate score distribution is able to give a good estimate of the non-link score distribution for the reservoir computer technique. In these cases, a cutoff based on the extent of the surrogate data is able to pick up the scores in the tail of the score distribution which are predominantly seen to come from true links. For the other two link inference techniques, the surrogate score distribution is also able to approximately match the non-link score distribution. However, unlike the reservoir computer results, for both transfer entropy and Granger causality, there appear to be a few non-link scores extended beyond the score value where the AAFT surrogate score distribution ends. Thus, a score cut-off based on the AAFT surrogate score distribution will give a larger number of false positives than the reservoir computer technique. Compared with Fig. 4.3(d) and Fig. 4.4(a), this example shows that our reservoir computer link inference technique is suitable to be used robustly with multiple surrogate time-series generation methods, and it is possible to obtain consistent performance across multiple methods of generating surrogate time-series, on different networked systems.

4.7 Discussion

In this work, we tested the performance of three network inference techniques - Granger causality, transfer entropy, and a reservoir-computer-based method, on time-series data from

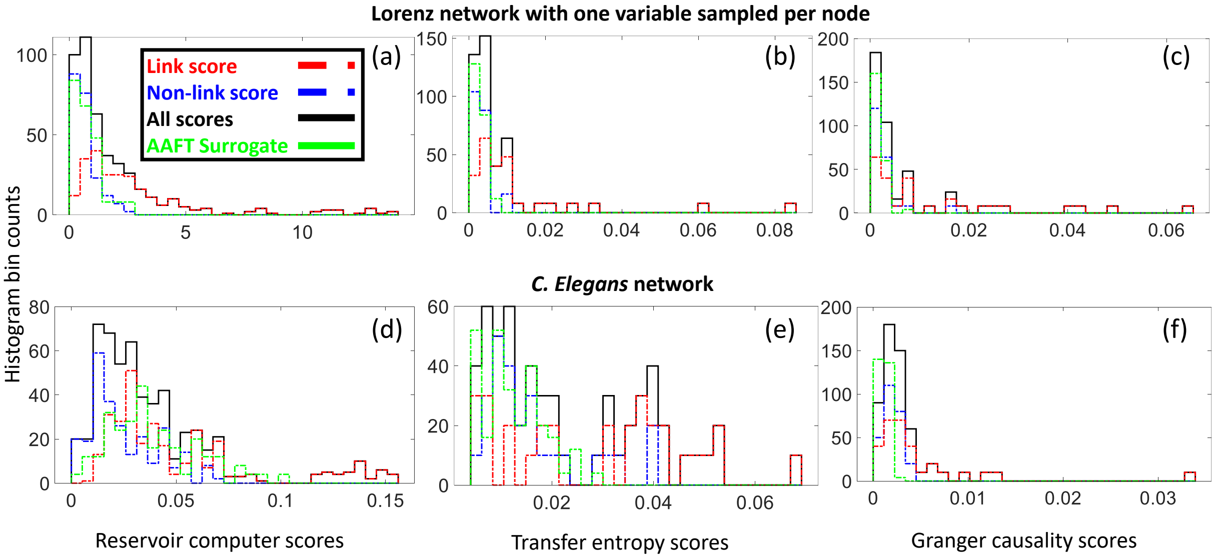

Figure 4.5: Distributions of scores for links, non-links, and amplitude-adjusted Fourier transform (AAFT) surrogate data with three different link inference techniques (reservoir computer, transfer entropy and Granger causality) from (a-c) data from Lorenz oscillator network with one variable sampled per node and (d-f) *C. elegans* neuronal fluorescence time-series.

calcium fluorescence images of *C. elegans* neurons, and the Lorenz model, with the ground truth networks known in all cases. Focusing on the important common situation in which the available time-series data is short, noisy, has low sampling time resolution, and is limited to partial state measurements, we show that, by use of a surrogate data technique, it is possible to obtain useful estimates of the statistical confidence of the potential existence of a significant fraction of the network links for all methods.

Since the RC method is a new technique that we have developed and used, we were more comfortable in confirming that we were working with the most optimized set of hyperparameters known to the best of our knowledge. However, the two other conventional network inference techniques (Granger causality, and transfer entropy) are much more widely used, and, as such, there exist numerous modifications to suite them for specialized types of time-series data, e.g., neural data of the kind used in this work. There are also a wide range of surrogate data generation techniques available for them as well. Since it is beyond the scope of this work to test all the different versions of Granger causality, transfer entropy, and surrogate data generation techniques, we decided to use the most basic versions of these techniques, as proposed and implemented by authors who first introduced the methods and the associated software packages. Furthermore, there are differences between the amount of input data that each method takes: RC and Granger causality techniques use the time-series data from all variables simultaneously, while transfer entropy is calculated pairwise between different variables. For the case of statistical analysis, RC method has a large number of fixed parameters (the elements of the input-to-reservoir and reservoir coupling matrices) which can be randomized to obtain an ensemble of RCs working on the same time-series data. This generates many samples for score distributions for statistical analysis of results for the RC, but this is absent for the other two methods, which do not usually have such random parameters. As such, we have avoided strong claims of comparison among the performance of different techniques in our work.

We also showed that the relatively new reservoir computer link inference techniques developed recently works better if the sampling time is shorter keeping the number of training time steps fixed, and there are only moderate amounts of observational noise. The network inference tests done on experimental data from *C. elegans* neurons with known ground truth connectivity show the importance and validity of network inference techniques on real-world neuronal calcium fluorescence datasets. As such, we hope that the results of this chapter provide information that will be useful for assessing the effectiveness of network inference techniques to time-series data from unknown networks with similar time-series data limitations.

4.8 Methods

4.8.1 Reservoir-computer-based connection inference technique

In this section, we summarize the reservoir-computer-based link inference technique [59], that we employ in our results.

Consider a system of D_n nodes connected by a directed network that we wish to infer. We assume that the full state of the i^{th} node in the network is given by a time-dependent vector $\mathbf{X}_i[t]$ of dimension D_s, with $i = 1, 2, 3, ..., D_n$. Denoting the components of this vector by X_i^μ with $\mu = 1, 2, 3, ..., D_s$, we suppose that the dynamics of the full system is governed by a general differential equation of the form,

$$\frac{dX_i^\mu(t)}{dt} = F_i^\mu \left[\mathbf{X}_1(t), \mathbf{X}_2(t), ..., \mathbf{X}_{D_n}(t)\right]. \tag{4.4}$$

Here, F_i^μ is the function dictating how the dynamics of the μ^{th} component of the state vector of the i^{th} node is governed by the states of all other nodes. A network link is said to exist from node i to node j if and only if F_i^μ is a function of $\mathbf{X}_j$ for some μ. Moreover, we define a network link to exist from the μ^{th} component of node i to the ν^{th} component of node j if F_i^μ is a function of X_j^ν. Thus, the presence (absence) of such a link implies $\partial F_i^\mu / \partial X_j^\nu$ is nonzero (zero). If we knew the function F_i^μ, then we could calculate these derivatives. However, as we consider situations in the absence of such knowledge, we are therefore tasked with estimating the derivative solely from the observed time-series data.

We assume that observations of the dynamical system Eq. (4.4) are sampled at a time interval Δt at times $n\Delta t$ for $n = \{1 \dots T\}$. We further assume that these observations are carried out via some measurement apparatus that measures $\mathbf{Y_i} = \mathbf{M}(\mathbf{X_i})$ for each node i. In cases where the full nodal state is available at each time, $\mathbf{Y}$ is identical to $\mathbf{X}$; however in general this may not be the case, and $\mathbf{Y}$ may be a vector with dimensionality $D'_s \leq D_s$.

We concatenate the sampled input measurements of the time-dependent node state vectors $\{\mathbf{Y}_i[t]\}$ and place them in a single time-dependent column vector $\mathcal{X}[t]$ of length $D_{\mathcal{X}} = D'_s D_n$,

$$\mathcal{X}[t] = \left(Y_1^1[t], Y_1^2[t], ..., Y_1^{D'_s}[t], Y_2^1[t], Y_2^2[t], \ldots, Y_2^{D'_s}[t], \ldots, Y_{D_n}^{D'_s}[t]\right)^T. \tag{4.5}$$

Considering the case where full state measurements are taken, we now briefly describe the machine learning technique for link inference developed in Refs. [59], which the reader can consult for a more detailed derivation. We train an artificial neural network called a reservoir computer (RC) to predict the time evolution of the nodal states one sampling time step Δt into the future. While RCs can be in general constructed through arbitrary high-dimensional dynamical systems [41], we implement the RC *in silico* as a dynamical system on a network of nodes (Fig. 4.6). (Note that this network of nodes is unrelated of the underlying network of the dynamical system being measured). We assume that the number of nodes D_r is large (such that $D_r \gg D_n \times D'_s \equiv D_{\mathcal{X}}$). The nodal states are stored in a vector $\mathbf{R}$ of length D_r. The sampled time-series vector is fed into the reservoir via a D_r-by-$D_{\mathcal{X}}$ input-to-reservoir coupling matrix $\mathbf{W}_{\text{in}}$ (Fig. 4.6). Furthermore, the reservoir nodes affect the dynamics of each other according to a D_r-by-D_r asymmetric adjacency matrix $\mathbf{H}$. The time evolution of the reservoir node states $\mathbf{R}$ are given by the equation,

$$\mathbf{R}\left[n\Delta t\right] = \sigma\left(\mathbf{H}\mathbf{R}\left[\left(n-1\right)\Delta t\right] + \mathbf{W}_{\text{in}}\mathcal{X}\left[n\Delta t\right]\right), \tag{4.6}$$

where n is a positive integer, and σ is a sigmoidal activation function acting componentwise on its vector argument (which has the same dimension, D_r, as $\mathbf{R}$).

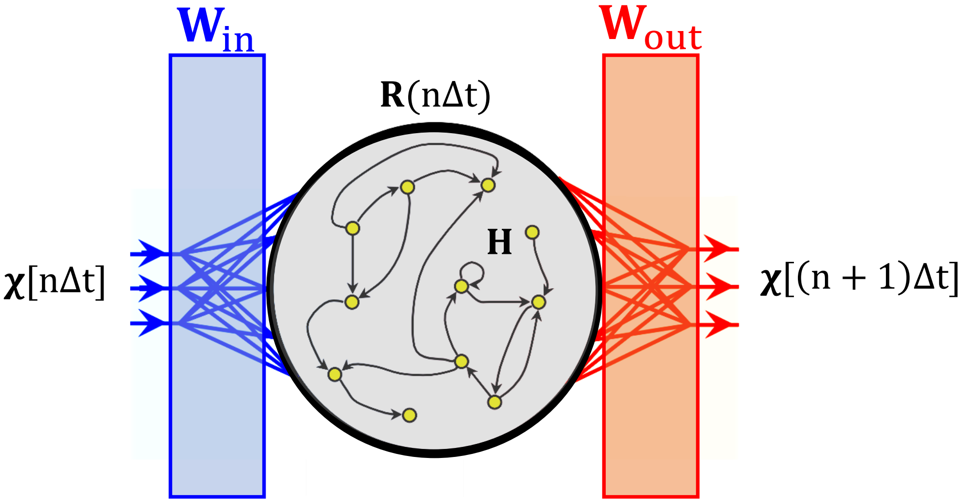

Figure 4.6: Schematic of the reservoir computer.

The goal of the reservoir computer training is to predict the one-time-step future values of the sampled components of $\{\mathbf{Y}_i\left[(n+1)\,\Delta t\right]\}$ in the concatenated form $\mathcal{X}\left[(n+1)\,\Delta t\right]$ (Eq. (4.5)) from their current values $\mathcal{X}\left[n\Delta t\right]$ using the reservoir state vector $\mathbf{R}\left[n\Delta t\right]$. In our case, this is done with a regularized linear regression determining a $D_{\mathcal{X}}$-by-D_r reservoir-to-output coupling matrix $\mathbf{W}_{\text{out}}$ by best-fitting $\mathbf{W}_{\text{out}}\mathbf{R}\left[n\Delta t\right]$ to the data for $\mathcal{X}$ one time-step Δt in the future $\mathcal{X}\left[(n+1)\Delta t\right]$, i.e., minimizing the cost function $\mathcal{C}$ given by

$$\mathcal{C} = \left\{\frac{1}{T}\sum_{n=1}^{T}\left\|\mathcal{X}\left[(n+1)\,\Delta t\right] - \mathbf{W}_{\text{out}}\mathbf{R}\left[n\Delta t\right]\right\|^2\right\} + \lambda\left\|\mathbf{W}_{\text{out}}\right\|^2 \tag{4.7}$$

where T is the number of training steps and the last term ($\lambda\left\|\mathbf{W}_{\text{out}}\right\|^2$) is a “ridge” regularization term used to prevent overfitting to the training data and λ is typically a small number.

The input matrix $\mathbf{W}_{\text{in}}$ is chosen such that each of the $D_{\mathcal{X}}$ components of the input vector goes to $R/D_{\mathcal{X}}$ distinct nodes of the reservoir. The non-zero elements of the input matrix $\mathbf{W}_{\text{in}}$ are chosen randomly from a uniform distribution in the interval $[-w, w]$. $\mathbf{H}$ is a sparse random matrix, corresponding to an average in-degree d_{av} of the reservoir nodes. The non-zero elements of $\mathbf{H}$ are chosen randomly from a uniform distribution such that the spectral radius of $\mathbf{H}$ is equal to some predefined value ρ, which we choose to be equal to 0.9. The hyperparameters w and d_{av} are chosen using a Nelder-Mead optimization procedure [191]. In our work, we used, $d_{\text{av}} = 2.7, w = 0.105$ for the Lorenz networks, and $d_{\text{av}} = 5, w = 0.42$ for the *C. elegans* network. In all network inference works with the Lorenz (*C. elegans*) model, we typically use values $\lambda T = 10^3(10^{-3})$, unless stated otherwise, a choice that we shall examine in the next section. The sigmoidal activation function σ is taken to be the hyperbolic tangent function. The reservoirs we used typically had $D_r = 3000$ nodes.

As derived in [59] and chapter 2, we estimate the Jacobian matrix of partial derivatives $M_{ij}(t) = \partial \mathcal{X}_i[(n+1)\Delta t]/\partial \mathcal{X}_j[n\Delta t]$ by

$$\frac{\partial \mathcal{X}_i[(n+1)\Delta t]}{\partial \mathcal{X}_j[n\Delta t]} = \sum_{k=1}^{D_R} (\mathbf{W}_{\text{out}})_{ik}\, \sigma'\left(\left(\mathbf{W}_{\text{in}} + \mathbf{H}\mathbf{W}_{\text{out}}^{-1}\right) \mathcal{X}[n\Delta t]\right)_k \left(\mathbf{W}_{\text{in}} + \mathbf{H}\mathbf{W}_{\text{out}}^{-1}\right)_{kj} \tag{4.8}$$

, and construct link scores as $S_{ij} = \langle |M_{ij}| \rangle_t$ where $\langle \rangle_t$ denotes time-averaging over a sufficiently long time so that the averages do not change significantly after doubling the averaging time.

Finally, we comment of the choice of the regularization parameter λ for the reservoir computer. When considering which λ value to choose, we first, for several different λ values, examine versions of our comparison of the score PDF distribution for all scores (e.g., plotted in black in Fig. 4.3) with the score distribution from the acausal surrogate (e.g., plotted in green in Fig. 4.2). We then select λ values so that, according to our comparison, there is a large-score threshold such that there are a substantial number of scores in the black distribution of Fig. 4.3 that are larger than that threshold score and no, or very few, surrogate distribution (green curve in Fig. 4.3) scores lie beyond that score. We then choose, from among these, the smallest λ value. Once a

λ value is chosen, we have a surrogate score distribution that we use further to assign a p-value to a given score and thus, obtain a score cut-off corresponding to a pre-defined p-value cut-off. This gave us $\lambda T = 10^3$ in all of the results that we show in figures 4.3-4.4. For the reservoir computer, we used $\lambda T = 10^3$ for the Lorenz data, and $\lambda T = 10^0$ for the *C. elegans* data in Fig. 4.5.

4.8.2 Transfer Entropy

For time series from two processes I and J, the transfer entropy from J to I is defined by [31]

$$T_{J \to I} = \sum_n p(i_{n+1}, i_n^{(k)}, j_n^{(l)}) \log \frac{p(i_{n+1} | i_n^{(k)}, j_n^{(l)})}{p(i_{n+1} | i_n^{(k)})} \tag{4.9}$$

where p denotes joint or conditional probability, i_n is the element of the time-series of I sampled at the n-th time-point and $i_n^{(k)} = (i_n, \ldots, i_{n-k+1})$ is the delay-embedding vector. In this equation, k and l are two time series "block-length" parameters which are chosen accordingly. There are multiple computational techniques [192] to obtain an estimate of transfer entropy expressed with this equation. Among them, in this work, we used the MATLAB code [192] for the rank-based technique to estimate transfer entropy, with block length and time-delay parameters set to 1 for both variables, and the number of quantization levels set to 3. For more details of the method and the definition of these parameters, please see the MATLAB code associated with [192].

4.8.3 Granger Causality

For two time series from processes I and J, we say that J does not Granger cause I if and only if I, conditional on its own past, is independent of the past of J [34–36]. In other words, if we can have information on the past of I from the past of J, then J would Granger cause I. The typical way to test this dependency of two time-series involves fitting a vector autoregressive model for I, and measuring whether inclusion of J in that model makes the fitting error significantly lower. For network link inference scoring purposes, we use the logarithm of the ratio of the two fitting errors as the network link score. For more details, as well as for the MATLAB toolbox that we have used in this work, please see [34].

4.9 Acknowledgements

This work was supported by NSF grant DMS1813027.

Chapter 5: Summary

In this final chapter, we summarize the material presented in this thesis. In this work, we wished to have a data-driven framework that could extract causal interactions among different agents in a dynamical system, solely from their observed time-series data. Such a framework is essential in integrating causal processes into numerical modeling of a multi-agent dynamical system. In seeking for causal inference, we took inspiration from standard experimental methodologies for obtaining causal networks, namely, the process of controlled interventions into the system and tracking the downstream effects of such interventions on different parts of the system over time. However, our method, in its nature, was purely computational and non-invasive. So, in a sense, we were able to build machine-learning-based computational models of observed complex dynamics which were not only able to predict and generalize the recorded time-series data, but were also able to reproduce the effects of small, controlled interventions that a human experimentalist might have performed on the system.

To demonstrate that our methodology could indeed detect both short-term and time-delayed causal interaction networks correctly, we tested the methodology on simulated time-series datasets from coupled Lorenz models and optoelectronic oscillator networks, as well as experimental time-series datasets from the optoelectronic oscillator networks and the *C. elegans* neural system. Since the ground truth causal connectome was completely known in all cases, we were able to quantify the performance of our method in network inference tasks, and identify situations and conditions which aided or limited causal inference.

Comparing our results in different systems we studied, we saw that presence of strong correlation and synchronization among the different time-series was detrimental to network link inference, as was observational noise, finite sampling time, and limited nodal state measurements. On the other hand, presence of dynamical noise, heterogeneity of parameters of individual dynamical agents, and large coupling strengths helped network inference by breaking strong synchrony or creating perturbations in the system.

We were not able to perfectly capture the complete causal network structure correctly in all cases, particularly when the above-mentioned limiting factors were present. Even in those cases, we were able to assign statistical confidence in the inferred causal links, by using properly constructed surrogate time-series data.

The results present in this thesis inspire one to use machine-learning-based techniques for data-driven study, modeling, prediction, and control of complex, nonlinear, networked dynamical systems.

Bibliography

[1] Judea Pearl and Dana Mackenzie. *The book of why: the new science of cause and effect*. Basic books, 2018.

[2] Mary-Ellen Lynall, Danielle S Bassett, Robert Kerwin, Peter J McKenna, Manfred Kitzbichler, Ulrich Muller, and Ed Bullmore. Functional connectivity and brain networks in schizophrenia. *J. Neurosci.*, 30(28):9477–9487, 2010.

[3] Danielle S Bassett, Nicholas F Wymbs, Mason A Porter, Peter J Mucha, Jean M Carlson, and Scott T Grafton. Dynamic reconfiguration of human brain networks during learning. *Proceedings of the National Academy of Sciences*, 108(18):7641–7646, 2011.

[4] David Adam. The promise and peril of the new science of social genomics. *Nature*, 574(7779):618–621, 2019.

[5] Kevin Hartnett. To build truly intelligent machines, teach them cause and effect. 2018.

[6] Cathy O'neil. *Weapons of math destruction: How big data increases inequality and threatens democracy*. Broadway books, 2016.

[7] Joy Buolamwini and Timnit Gebru. Gender shades: Intersectional accuracy disparities in commercial gender classification. In *Conference on fairness, accountability and transparency*, pages 77–91. PMLR, 2018.

[8] Emily M Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. On the dangers of stochastic parrots: Can language models be too big?. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pages 610–623, 2021.

[9] Alesia Chernikova, Alina Oprea, Cristina Nita-Rotaru, and BaekGyu Kim. Are self-driving cars secure? evasion attacks against deep neural networks for steering angle prediction. In *2019 IEEE Security and Privacy Workshops (SPW)*, pages 132–137. IEEE, 2019.

[10] Julia K Winkler, Christine Fink, Ferdinand Toberer, Alexander Enk, Teresa Deinlein, Rainer Hofmann-Wellenhof, Luc Thomas, Aimilios Lallas, Andreas Blum, Wilhelm Stolz, et al. Association between surgical skin markings in dermoscopic images and diagnostic performance of a deep learning convolutional neural network for melanoma recognition. *JAMA dermatology*, 155(10):1135–1141, 2019.

[11] Evan J Molinelli, Anil Korkut, Weiqing Wang, Martin L Miller, Nicholas P Gauthier, Xiaohong Jing, Poorvi Kaushik, Qin He, Gordon Mills, David B Solit, et al. Perturbation biology: inferring signaling networks in cellular systems. *PLoS Comput. Biol.*, 9(12):e1003290, 2013.

[12] Mark Ed Newman, Albert-László Ed Barabási, and Duncan J Watts. *The structure and dynamics of networks.* Princeton university press, 2006.

[13] Reka Albert and Albert-Laszlo Barabasi. Statistical mechanics of complex networks. *Rev. Mod. Phys.*, 74(1):47, 2002.

[14] Danielle S Bassett and Olaf Sporns. Network neuroscience. *Nat. Neurosci.*, 20(3):353–364, 2017.

[15] Nabil Guelzim, Samuele Bottani, Paul Bourgine, and François Képès. Topological and causal structure of the yeast transcriptional regulatory network. *Nature genetics*, 31(1):60–63, 2002.

[16] Romualdo Pastor-Satorras, Claudio Castellano, Piet Van Mieghem, and Alessandro Vespignani. Epidemic processes in complex networks. *Rev. Mod. Phys.*, 87(3):925, 2015.

[17] Alex Arenas, Albert Díaz-Guilera, Jurgen Kurths, Yamir Moreno, and Changsong Zhou. Synchronization in complex networks. *Physics reports*, 469(3):93–153, 2008.

[18] Michael Conover, Jacob Ratkiewicz, Matthew Francisco, Bruno Gonçalves, Filippo Menczer, and Alessandro Flammini. Political polarization on twitter. In *Proceedings of the international aaai conference on web and social media*, volume 5, pages 89–96, 2011.

[19] Ildefons Magrans de Abril, Junichiro Yoshimoto, and Kenji Doya. Connectivity inference from neural recording data: Challenges, mathematical bases and research directions. *Neural Networks*, 102:120–137, 2018.

[20] Mario Chavez, Miguel Valencia, Vincent Navarro, Vito Latora, and Jacques Martinerie. Functional modularity of background activities in normal and epileptic brain networks. *Phys. Rev. Lett.*, 104(11):118701, 2010.

[21] Chao Sima, Jianping Hua, and Sungwon Jung. Inference of gene regulatory networks using time-series data: a survey. *Current genomics*, 10(6):416–429, 2009.

[22] Reka Albert. Network inference, analysis, and modeling in systems biology. *The Plant Cell*, 19(11):3327–3338, 2007.

[23] Elizabeth L Sander, J Timothy Wootton, and Stefano Allesina. Ecological network inference from long-term presence-absence data. *Scientific reports*, 7(1):1–12, 2017.

[24] Isobel Milns, Colin M Beale, and V Anne Smith. Revealing ecological networks using bayesian network inference algorithms. *Ecology*, 91(7):1892–1899, 2010.

[25] Jakob Runge, Vladimir Petoukhov, Jonathan F Donges, Jaroslav Hlinka, Nikola Jajcay, Martin Vejmelka, David Hartman, Norbert Marwan, Milan Paluš, and Jürgen Kurths. Identifying causal gateways and mediators in complex spatio-temporal systems. *Nat. Commun.*, 6(1):1–10, 2015.

[26] Bin Zhang and Steve Horvath. A general framework for weighted gene co-expression network analysis. *Statistical applications in genetics and molecular biology*, 4(1), 2005.

[27] Karoline Faust, J Fah Sathirapongsasuti, Jacques Izard, Nicola Segata, Dirk Gevers, Jeroen Raes, and Curtis Huttenhower. Microbial co-occurrence relationships in the human microbiome. *PLoS computational biology*, 8(7):e1002606, 2012.

[28] CJ Stam, W De Haan, ABFJ Daffertshofer, BF Jones, I Manshanden, Anne-Marie van Cappellen van Walsum, Teresa Montez, JPA Verbunt, JC De Munck, BW Van Dijk, et al. Graph theoretical analysis of magnetoencephalographic functional connectivity in alzheimer's disease. *Brain*, 132(1):213–224, 2009.

[29] Carl D Hacker, Joel S Perlmutter, Susan R Criswell, Beau M Ances, and Abraham Z Snyder. Resting state functional connectivity of the striatum in parkinson's disease. *Brain*, 135(12):3699–3711, 2012.

[30] Steven N Baldassano and Danielle S Bassett. Topological distortion and reorganized modular structure of gut microbial co-occurrence networks in inflammatory bowel disease. *Scientific reports*, 6(1):1–14, 2016.

[31] Thomas Schreiber. Measuring information transfer. *Physical review letters*, 85(2):461, 2000.

[32] Vivek Veeriah, Rohit Durvasula, and Guo-Jun Qi. Deep learning architecture with dynamically programmed layers for brain connectome prediction. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1205–1214, 2015.

[33] Clive WJ Granger. Investigating causal relations by econometric models and cross-spectral methods. *Econometrica: journal of the Econometric Society*, pages 424–438, 1969.

[34] Lionel Barnett and Anil K Seth. The MVGC multivariate Granger causality toolbox: a new approach to Granger-causal inference. *Journal of neuroscience methods*, 223:50–68, 2014.

[35] Maciej Kamiński, Mingzhou Ding, Wilson A Truccolo, and Steven L Bressler. Evaluating causal relations in neural systems: Granger causality, directed transfer function and statistical assessment of significance. *Biological cybernetics*, 85(2):145–157, 2001.

[36] Karl Friston, Rosalyn Moran, and Anil K Seth. Analysing connectivity with Granger causality and dynamic causal modelling. *Current opinion in neurobiology*, 23(2):172–178, 2013.

[37] Patrick A Stokes and Patrick L Purdon. A study of problems encountered in Granger causality analysis from a neuroscience perspective. *Proceedings of the national academy of sciences*, 114(34):E7063–E7072, 2017.

[38] Lionel Barnett, Adam B Barrett, and Anil K Seth. Misunderstandings regarding the application of Granger causality in neuroscience. *Proceedings of the National Academy of Sciences*, 115(29):E6676–E6677, 2018.

[39] Cunlu Zou and Jianfeng Feng. Granger causality vs. dynamic Bayesian network inference: a comparative study. *BMC bioinformatics*, 10(1):1–17, 2009.

[40] Pantelis R Vlachas, Jaideep Pathak, Brian R Hunt, Themistoklis P Sapsis, Michelle Girvan, Edward Ott, and Petros Koumoutsakos. Backpropagation algorithms and reservoir computing in recurrent neural networks for the forecasting of complex spatiotemporal dynamics. *Neural Networks*, 126:191–217, 2020.

[41] Gouhei Tanaka, Toshiyuki Yamane, Jean Benoit Héroux, Ryosho Nakane, Naoki Kanazawa, Seiji Takeda, Hidetoshi Numata, Daiju Nakano, and Akira Hirose. Recent advances in physical reservoir computing: A review. *Neural Networks*, 115:100–123, 2019.

[42] Yanne K Chembo. Machine learning based on reservoir computing with time-delayed optoelectronic and photonic systems. *Chaos*, 30(1):013111, 2020.

[43] Danijela Marković, Alice Mizrahi, Damien Querlioz, and Julie Grollier. Physics for neuromorphic computing. *Nature Reviews Physics*, 2(9):499–510, 2020.

[44] Laurent Larger, Antonio Baylon-Fuentes, Romain Martinenghi, Vladimir S Udaltsov, Yanne K Chembo, and Maxime Jacquot. High-speed photonic reservoir computing using a time-delay-based architecture: Million words per second classification. *Phys. Rev. X*, 7(1):011015, 2017.

[45] Quentin Vinckier, François Duport, Anteo Smerieri, Kristof Vandoorne, Peter Bienstman, Marc Haelterman, and Serge Massar. High-performance photonic reservoir computer based on a coherently driven passive cavity. *Optica*, 2(5):438–446, 2015.

[46] François Duport, Anteo Smerieri, Akram Akrout, Marc Haelterman, and Serge Massar. Fully analogue photonic reservoir computer. *Sci. Rep.*, 6:22381, 2016.

[47] Benjamin Schrauwen, Michiel D'Haene, David Verstraeten, and Jan Van Campenhout. Compact hardware liquid state machines on FPGA for real-time speech recognition. *Neural Netw.*, 21(2-3):511–523, 2008.

[48] Lennert Appeltant, Miguel Cornelles Soriano, Guy Van der Sande, Jan Danckaert, Serge Massar, Joni Dambre, Benjamin Schrauwen, Claudio R Mirasso, and Ingo Fischer. Information processing using a single dynamical node as complex system. *Nat. Commun.*, 2(1):1–6, 2011.

[49] Pierre Enel, Emmanuel Procyk, Rene Quilodran, and Peter Ford Dominey. Reservoir computing properties of neural dynamics in prefrontal cortex. *PLoS Comput. Biol.*, 12(6):e1004967, 2016.

[50] Alireza Goudarzi, Matthew R Lakin, and Darko Stefanovic. Dna reservoir computing: a novel molecular computing approach. In *International Workshop on DNA-Based Computers*, pages 76–89. Springer, 2013.

[51] Hoang Nguyen, Peter Banda, Darko Stefanovic, and Christof Teuscher. Reservoir computing with random chemical systems. In *Artificial Life Conference Proceedings*, pages 491–499. MIT Press, 2020.

[52] Daniele Pinna, George Bourianoff, and Karin Everschor-Sitte. Reservoir computing with random skyrmion textures. *Physical Review Applied*, 14(5):054020, 2020.

[53] Zhixin Lu, Jason Z Kim, and Danielle S Bassett. Supervised chaotic source separation by a tank of water. *Chaos*, 30(2):021101, 2020.

[54] Jaideep Pathak, Brian Hunt, Michelle Girvan, Zhixin Lu, and Edward Ott. Model-free prediction of large spatiotemporally chaotic systems from data: A reservoir computing approach. *Phys. Rev. Lett.*, 120(2):024102, 2018.

[55] Keshav Srinivasan, Nolan Coble, Joy Hamlin, Thomas Antonsen, Edward Ott, and Michelle Girvan. Parallel machine learning for forecasting the dynamics of complex networks. *Physical Review Letters*, 128(16):164101, 2022.

[56] Jaideep Pathak, Zhixin Lu, Brian R Hunt, Michelle Girvan, and Edward Ott. Using machine learning to replicate chaotic attractors and calculate lyapunov exponents from data. *Chaos*, 27(12):121102, 2017.

[57] Zhixin Lu, Brian R Hunt, and Edward Ott. Attractor reconstruction by machine learning. *Chaos*, 28(6):061104, 2018.

[58] Sanjukta Krishnagopal, Michelle Girvan, Edward Ott, and Brian R Hunt. Separation of chaotic signals by reservoir computing. *Chaos*, 30(2):023123, 2020.

[59] Amitava Banerjee, Jaideep Pathak, Rajarshi Roy, Juan G Restrepo, and Edward Ott. Using machine learning to assess short term causal dependence and infer network links. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 29(12):121104, 2019.

[60] Amitava Banerjee, Joseph D Hart, Rajarshi Roy, and Edward Ott. Machine learning link inference of noisy delay-coupled networks with optoelectronic experimental tests. *Physical Review X*, 11(3):031014, 2021.

[61] Richard Feynman. *The Character of Physical Law, with new foreword.* MIT press, 2017.

[62] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT press Cambridge, 2016.

[63] Mingzhou Ding, Yonghong Chen, and Steven L Bressler. Granger causality: basic theory and application to neuroscience. *Handbook of time series analysis: recent theoretical developments and applications*, pages 437–460, 2006.

[64] Jonathan F Donges, Yong Zou, Norbert Marwan, and Jürgen Kurths. Complex networks in climate dynamics. *The European Physical Journal Special Topics*, 174(1):157–179, 2009.

[65] Wai Lim Ku, Geet Duggal, Yuan Li, Michelle Girvan, and Edward Ott. Interpreting patterns of gene expression: Signatures of coregulation, the data processing inequality, and triplet motifs. *PloS one*, 7(2):e31969, 2012.

[66] Jie Ren, Wen-Xu Wang, Baowen Li, and Ying-Cheng Lai. Noise bridges dynamical correlation and topology in coupled oscillator networks. *Physical review letters*, 104(5):058701, 2010.

[67] Zoran Levnajić and Arkady Pikovsky. Untangling complex dynamical systems via derivative-variable correlations. *Scientific reports*, 4(1):1–7, 2014.

[68] Marc G Leguia, Ralph G Andrzejak, and Zoran Levnajić. Evolutionary optimization of network reconstruction from derivative-variable correlations. *Journal of Physics A: Mathematical and Theoretical*, 50(33):334001, 2017.

[69] Jie Sun and Erik M Bollt. Causation entropy identifies indirect influences, dominance of neighbors and anticipatory couplings. *Physica D: Nonlinear Phenomena*, 267:49–57, 2014.

[70] Marc Timme. Revealing network connectivity from response dynamics. *Physical review letters*, 98(22):224101, 2007.

[71] Mark J Panaggio, Maria-Veronica Ciocanel, Lauren Lazarus, Chad M Topaz, and Bin Xu. Model reconstruction from temporal data for coupled oscillator networks. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 29(10):103116, 2019.

[72] Marc G Leguia, Cristina GB Martinez, Irene Malvestio, Adrià Tauste Campo, Rodrigo Rocamora, Zoran Levnajic, and Ralph G Andrzejak. Inferring directed networks using a rank-based connectivity measure. *Phys. Rev. E*, 99(1):012319, 2019.

[73] Tomislav Stankovski, Tiago Pereira, Peter VE McClintock, and Aneta Stefanovska. Coupling functions: universal insights into dynamical interaction mechanisms. *Reviews of Modern Physics*, 89(4):045001, 2017.

[74] Srinivas Gorur Shandilya and Marc Timme. Inferring network topology from complex dynamics. *New Journal of Physics*, 13(1):013004, 2011.

[75] Siyang Leng, Ziwei Xu, and Huanfei Ma. Reconstructing directional causal networks with random forest: Causality meeting machine learning. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 29(9):093130, 2019.

[76] Ren-Meng Cao, Si-Yuan Liu, and Xiao-Ke Xu. Network embedding for link prediction: The pitfall and improvement. *Chaos*, 29(10):103102, 2019.

[77] Marc G Leguia, Zoran Levnajic, Ljupčo Todorovski, and Bernard Ženko. Reconstructing dynamical networks via feature ranking. *Chaos*, 29(9):093107, 2019.

[78] Herbert Jaeger and Harald Haas. Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication. *Science*, 304(5667):78–80, 2004.

[79] Zhixin Lu, Jaideep Pathak, Brian Hunt, Michelle Girvan, Roger Brockett, and Edward Ott. Reservoir observers: Model-free inference of unmeasured variables in chaotic systems. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 27(4):041102, 2017.

[80] Louis M Pecora and Thomas L Carroll. Master stability functions for synchronized coupled systems. *Physical review letters*, 80(10):2109, 1998.

[81] Herbert Jaeger. The "echo state" approach to analysing and training recurrent neural networks-with an erratum note. *Bonn, Germany: German National Research Center for Information Technology GMD Technical Report*, 148(34):13, 2001.

[82] Wolfgang Maass, Thomas Natschläger, and Henry Markram. Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural computation*, 14(11):2531–2560, 2002.

[83] Mantas Lukoševičius and Herbert Jaeger. Reservoir computing approaches to recurrent neural network training. *Computer Science Review*, 3(3):127–149, 2009.

[84] Piotr Antonik, Marvyn Gulina, Jaël Pauwels, and Serge Massar. Using a reservoir computer to learn chaotic attractors, with applications to chaos synchronization and cryptography. *Physical Review E*, 98(1):012215, 2018.

[85] Piotr Antonik, Marc Haelterman, and Serge Massar. Brain-inspired photonic signal processor for generating periodic patterns and emulating chaotic systems. *Physical Review Applied*, 7(5):054014, 2017.

[86] Nikolai F Rulkov, Mikhail M Sushchik, Lev S Tsimring, and Henry DI Abarbanel. Generalized synchronization of chaos in directionally coupled chaotic systems. *Phys. Rev. E*, 51(2):980, 1995.

[87] Ljupco Kocarev and Ulrich Parlitz. Generalized synchronization, predictability, and equivalence of unidirectionally coupled dynamical systems. *Phys. Rev. Lett.*, 76(11):1816, 1996.

[88] Brian R Hunt, Edward Ott, and James A Yorke. Differentiable generalized synchronization of chaos. *Physical Review E*, 55(4):4029, 1997.

[89] Arthur E Hoerl and Robert W Kennard. Ridge regression: applications to nonorthogonal problems. *Technometrics*, 12(1):69–82, 1970.

[90] Roger Penrose. A generalized inverse for matrices. In *Mathematical proceedings of the Cambridge philosophical society*, volume 51, pages 406–413. Cambridge University Press, 1955.

[91] Lukas Gonon and Juan-Pablo Ortega. Reservoir computing universality with stochastic inputs. *IEEE transactions on neural networks and learning systems*, 31(1):100–112, 2019.

[92] Edward N Lorenz. Deterministic nonperiodic flow. *Journal of atmospheric sciences*, 20(2):130–141, 1963.

[93] Hien D Nguyen and Geoffrey J McLachlan. Maximum likelihood estimation of gaussian mixture models without matrix operations. *Advances in Data Analysis and Classification*, 9(4):371–394, 2015.

[94] James L Kaplan and James A Yorke. Chaotic behavior of multidimensional difference equations. In *Functional differential equations and approximation of fixed points*, pages 204–227. Springer, 1979.

[95] J Doyne Farmer, Edward Ott, and James A Yorke. The dimension of chaotic attractors. *Physica D: Nonlinear Phenomena*, 7(1-3):153–180, 1983.

[96] Jennifer A Dunne, Richard J Williams, and Neo D Martinez. Food-web structure and network theory: the role of connectance and size. *Proc. Natl. Acad. Sci. USA*, 99(20):12917–12922, 2002.

[97] Nathan D Price and Ilya Shmulevich. Biochemical and statistical network models for systems biology. *Curr. Opin. Biotech.*, 18(4):365–370, 2007.

[98] Marc Vidal, Michael E Cusick, and Albert-Laszlo Barabasi. Interactome networks and human disease. *Cell*, 144(6):986–998, 2011.

[99] Anastasia Baryshnikova, Michael Costanzo, Chad L Myers, Brenda Andrews, and Charles Boone. Genetic interaction networks: toward an understanding of heritability. *Annu. Rev. Genomics Hum. Genet.*, 14:111–133, 2013.

[100] Michael Costanzo, Benjamin VanderSluis, Elizabeth N Koch, Anastasia Baryshnikova, Carles Pons, Guihong Tan, Wen Wang, Matej Usaj, Julia Hanchard, Susan D Lee, et al. A global genetic interaction network maps a wiring diagram of cellular function. *Science*, 353(6306), 2016.

[101] Percy Venegas. Tracing the untraceable: Ai network inference for the dark web and crypto privacy coins. 2018.

[102] Mingzhou Ding, Yonghong Chen, and Steven L. Bressler. *Granger Causality: Basic Theory and Application to Neuroscience*, chapter 17, pages 437–460. John Wiley & Sons, Ltd, 2006.

[103] Douglas Zhou, Yanyang Xiao, Yaoyu Zhang, Zhiqin Xu, and David Cai. Causal and structural connectivity of pulse-coupled nonlinear networks. *Phys. Rev. Lett.*, 111(5):054102, 2013.

[104] Jie Sun, Dane Taylor, and Erik M Bollt. Causal network inference by optimal causation entropy. *SIAM J. on Appl. Dyn. Syst.*, 14(1):73–106, 2015.

[105] Jose Casadiego, Dimitra Maoutsa, and Marc Timme. Inferring network connectivity from event timing patterns. *Phys. Rev. Lett.*, 121(5):054101, 2018.

[106] Tiago P Peixoto. Network reconstruction and community detection from dynamics. *Phys. Rev. Lett.*, 123(12):128301, 2019.

[107] David Dahmen, Hannah Bos, and Moritz Helias. Correlated fluctuations in strongly coupled binary networks beyond equilibrium. *Phys. Rev. X*, 6(3):031024, 2016.

[108] Jinyin Chen, Xiang Lin, Chenyu Jia, Yuwei Li, Yangyang Wu, Haibin Zheng, and Yi Liu. Generative dynamic link prediction. *Chaos*, 29(12):123111, 2019.

[109] Nikita Frolov, Vladimir Maksimenko, Annika Lüttjohann, Alexey Koronovskii, and Alexander Hramov. Feed-forward artificial neural network provides data-driven inference of functional connectivity. *Chaos*, 29(9):091101, 2019.

[110] Giuseppe Carleo, Ignacio Cirac, Kyle Cranmer, Laurent Daudet, Maria Schuld, Naftali Tishby, Leslie Vogt-Maranto, and Lenka Zdeborova. Machine learning and the physical sciences. *Reviews of Modern Physics*, 91(4):045002, 2019.

[111] Jaideep Pathak, Alexander Wikner, Rebeckah Fussell, Sarthak Chandra, Brian R Hunt, Michelle Girvan, and Edward Ott. Hybrid forecasting of chaotic processes: Using machine learning in conjunction with a knowledge-based model. *Chaos*, 28(4):041101, 2018.

[112] George Neofotistos, Marios Mattheakis, Georgios D Barmparis, Johanne Hizanidis, Giorgos P Tsironis, and Efthimios Kaxiras. Machine learning with observers predicts complex spatiotemporal behavior. *Front. Phys.*, 7:24, 2019.

[113] Roland S Zimmermann and Ulrich Parlitz. Observing spatio-temporal dynamics of excitable media using reservoir computing. *Chaos*, 28(4):043118, 2018.

[114] Abhranil Das and Ila R Fiete. Systematic errors in connectivity inferred from activity in strongly recurrent networks. *Nat. Neurosci.*, 23:1286–1296, 2020.

[115] Catharina Olsen, Kathleen Fleming, Niall Prendergast, Renee Rubio, Frank Emmert-Streib, Gianluca Bontempi, Benjamin Haibe-Kains, and John Quackenbush. Inference and validation of predictive gene networks from biomedical literature and gene expression data. *Genomics*, 103(5-6):329–336, 2014.

[116] X Steve Yao and Lute Maleki. Optoelectronic microwave oscillator. *J. Opt. Soc. Am. B*, 13(8):1725–1735, 1996.

[117] Y Chembo Kouomou, Pere Colet, Laurent Larger, and Nicolas Gastaud. Chaotic breathers in delayed electro-optical systems. *Phys. Rev. Lett.*, 95(20):203903, 2005.

[118] Kristine E Callan, Lucas Illing, Zheng Gao, Daniel J Gauthier, and Eckehard Scholl. Broadband chaos generated by an optoelectronic oscillator. *Phys. Rev. Lett.*, 104(11):113901, 2010.

[119] Laurent Larger, Miguel C Soriano, Daniel Brunner, Lennert Appeltant, Jose M Gutierrez, Luis Pesquera, Claudio R Mirasso, and Ingo Fischer. Photonic information processing beyond turing: an optoelectronic implementation of reservoir computing. *Opt. Express*, 20(3):3241–3249, 2012.

[120] Yvan Paquot, Francois Duport, Antoneo Smerieri, Joni Dambre, Benjamin Schrauwen, Marc Haelterman, and Serge Massar. Optoelectronic reservoir computing. *Sci. Rep.*, 2:287, 2012.

[121] Apostolos Argyris, Dimitris Syvridis, Laurent Larger, Valerio Annovazzi-Lodi, Pere Colet, Ingo Fischer, Jordi Garcia-Ojalvo, Claudio R Mirasso, Luis Pesquera, and K Alan Shore. Chaos-based communications at high bit rates using commercial fibre-optic links. *Nature*, 438(7066):343–346, 2005.

[122] Jianping Yao. Optoelectronic oscillators for high speed and high resolution optical sensing. *Journal of Lightwave Technology*, 35(16):3489–3497, 2017.

[123] Laurent Larger and John M Dudley. Optoelectronic chaos. *Nature*, 465(7294):41–42, 2010.

[124] Laurent Larger. Complexity in electro-optic delay dynamics: modelling, design and applications. *Phil. Trans. R. Soc. A*, 371(1999):20120464, 2013.

[125] Yanne K Chembo, Daniel Brunner, Maxime Jacquot, and Laurent Larger. Optoelectronic oscillators with time-delayed feedback. *Rev. Mod. Phys.*, 91(3):035006, 2019.

[126] Lucas Illing, Cristian D Panda, and Lauren Shareshian. Isochronal chaos synchronization of delay-coupled optoelectronic oscillators. *Phys. Rev. E*, 84(1):016213, 2011.

[127] Bhargava Ravoori, Adam B Cohen, Jie Sun, Adilson E Motter, Thomas E Murphy, and Rajarshi Roy. Robustness of optimal synchronization in real networks. *Phys. Rev. Lett.*, 107(3):034102, 2011.

[128] Caitlin RS Williams, Thomas E Murphy, Rajarshi Roy, Francesco Sorrentino, Thomas Dahms, and Eckehard Scholl. Experimental observations of group synchrony in a system of chaotic optoelectronic oscillators. *Phys. Rev. Lett.*, 110(6):064104, 2013.

[129] Joseph D Hart, Kanika Bansal, Thomas E Murphy, and Rajarshi Roy. Experimental observation of chimera and cluster states in a minimal globally coupled network. *Chaos*, 26(9):094801, 2016.

[130] Joseph D Hart, Jan Philipp Pade, Tiago Pereira, Thomas E Murphy, and Rajarshi Roy. Adding connections can hinder network synchronization of time-delayed oscillators. *Phys. Rev. E*, 92(2):022804, 2015.

[131] Thomas Lymburn, David M Walker, Michael Small, and Thomas Jüngling. The reservoir's perspective on generalized synchronization. *Chaos*, 29(9):093133, 2019.

[132] Erik Bollt. On explaining the surprising success of reservoir computing forecaster of chaos? the universal machine learning dynamical system with contrast to var and dmd. *Chaos*, 31(1):013108, 2021.

[133] Tom Fawcett. An introduction to roc analysis. *Pattern recognition letters*, 27(8):861–874, 2006.

[134] Yang Yang, Ryan N Lichtenwalter, and Nitesh V Chawla. Evaluating link prediction methods. *Knowledge and Information Systems*, 45(3):751–782, 2015.

[135] Gloria Cecchini, Arkady Pikovsky, et al. Impact of local network characteristics on network reconstruction. *Physical Review E*, 103(2):022305, 2021.

[136] Thomas E Murphy, Adam B Cohen, Bhargava Ravoori, Karl RB Schmitt, Anurag V Setty, Francesco Sorrentino, Caitlin RS Williams, Edward Ott, and Rajarshi Roy. Complex dynamics and synchronization of delayed-feedback nonlinear oscillators. *Phil. Trans. R. Soc. A*, 368(1911):343–366, 2010.

[137] Joseph D Hart, Laurent Larger, Thomas E Murphy, and Rajarshi Roy. Delayed dynamical systems: networks, chimeras and reservoir computing. *Philosophical Transactions of the Royal Society A*, 377(2153):20180123, 2019.

[138] Joseph David Hart. *Experiments on networks of coupled opto-electronic oscillators and Phys. random number generators*. PhD thesis, 2018.

[139] Valentin Flunkert, Serhiy Yanchuk, Thomas Dahms, and Eckehard Scholl. Synchronizing distant nodes: a universal classification of networks. *Phys. Rev. Lett.*, 105(25):254101, 2010.

[140] Alex Townsend, Michael Stillman, and Steven H Strogatz. Dense networks that do not synchronize and sparse ones that do. *Chaos*, 30(8):083142, 2020.

[141] Joanna Lipinski-Kruszka, Jacob Stewart-Ornstein, Michael W Chevalier, and Hana El-Samad. Using dynamic noise propagation to infer causal regulatory relationships in biochemical networks. *ACS synthetic biology*, 4(3):258–264, 2015.

[142] Robert J Prill, Robert Vogel, Guillermo A Cecchi, Grégoire Altan-Bonnet, and Gustavo Stolovitzky. Noise-driven causal inference in biomolecular networks. *PloS one*, 10(6):e0125777, 2015.

[143] Wen-Xu Wang, Jie Ren, Ying-Cheng Lai, and Baowen Li. Reverse engineering of complex dynamical networks in the presence of time-delayed interactions based on noisy time series. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 22(3):033131, 2012.

[144] Louis M Pecora, Francesco Sorrentino, Aaron M Hagerstrom, Thomas E Murphy, and Rajarshi Roy. Cluster synchronization and isolated desynchronization in complex networks with symmetries. *Nat. Commun.*, 5(1):1–8, 2014.

[145] DV Senthilkumar, M Lakshmanan, and J Kurths. Transition from phase to generalized synchronization in time-delay systems. *Chaos*, 18(2):023118, 2008.

[146] DV Senthilkumar, R Suresh, M Lakshmanan, and J Kurths. Global generalized synchronization in networks of different time-delay systems. *Europhys. Lett.*, 103(5):50010, 2013.

[147] Andrew J Whalen, Sean N Brennan, Timothy D Sauer, and Steven J Schiff. Observability and controllability of nonlinear networks: The role of symmetry. *Phys. Rev. X*, 5(1):011005, 2015.

[148] Michael Denker, Marc Timme, Markus Diesmann, Fred Wolf, and Theo Geisel. Breaking synchrony by heterogeneity in complex networks. *Phys. Rev. Lett.*, 92(7):074103, 2004.

[149] Takashi Nishikawa, Adilson E Motter, Ying-Cheng Lai, and Frank C Hoppensteadt. Heterogeneity in oscillator networks: Are smaller worlds easier to synchronize? *Phys. Rev. Lett.*, 91(1):014101, 2003.

[150] Xiao Han, Zhesi Shen, Wen-Xu Wang, and Zengru Di. Robust reconstruction of complex networks from sparse data. *Phys. Rev. Lett.*, 114(2):028701, 2015.

[151] Tiago P Peixoto. Reconstructing networks with unknown and heterogeneous errors. *Phys. Rev. X*, 8(4):041011, 2018.

[152] Domenico Napoletani and Timothy D Sauer. Reconstructing the topology of sparsely connected dynamical networks. *Phys. Rev. E*, 77(2):026103, 2008.

[153] Alan V Oppenheim, John R Buck, and Ronald W Schafer. *Discrete-time signal processing. Vol. 2*. Upper Saddle River, NJ: Prentice Hall, 2001.

[154] Saul Kato, Harris S Kaplan, Tina Schrödel, Susanne Skora, Theodore H Lindsay, Eviatar Yemini, Shawn Lockery, and Manuel Zimmer. Global brain dynamics embed the motor command sequence of Caenorhabditis elegans. *Cell*, 163(3):656–669, 2015.

[155] Lav R Varshney, Beth L Chen, Eric Paniagua, David H Hall, and Dmitri B Chklovskii. Structural properties of the Caenorhabditis elegans neuronal network. *PLoS computational biology*, 7(2):e1001066, 2011.

[156] Ann S Blevins, Dani S Bassett, Ethan K Scott, and Gilles C Vanwalleghem. From calcium imaging to graph topology. *Network Neuroscience*, pages 1–42, 2022.

[157] Xiaojie Qiu, Arman Rahimzamani, Li Wang, Bingcheng Ren, Qi Mao, Timothy Durham, José L McFaline-Figueroa, Lauren Saunders, Cole Trapnell, and Sreeram Kannan. Inferring causal gene regulatory networks from coupled single-cell expression dynamics using scribe. *Cell systems*, 10(3):265–274, 2020.

[158] Josef Ludescher, Maria Martin, Niklas Boers, Armin Bunde, Catrin Ciemer, Jingfang Fan, Shlomo Havlin, Marlene Kretschmer, Jürgen Kurths, Jakob Runge, et al. Network-based forecasting of climate phenomena. *Proceedings of the National Academy of Sciences*, 118(47):e1922872118, 2021.

[159] Jingfang Fan, Jun Meng, Yosef Ashkenazy, Shlomo Havlin, and Hans Joachim Schellnhuber. Network analysis reveals strongly localized impacts of El Niño. *Proceedings of the National Academy of Sciences*, 114(29):7543–7548, 2017.

[160] Aditya Pratapa, Amogh P Jalihal, Jeffrey N Law, Aditya Bharadwaj, and TM Murali. Benchmarking algorithms for gene regulatory network inference from single-cell transcriptomic data. *Nature methods*, 17(2):147–154, 2020.

[161] Leonardo Novelli, Patricia Wollstadt, Pedro Mediano, Michael Wibral, and Joseph T Lizier. Large-scale directed network inference with multivariate transfer entropy and hierarchical statistical testing. *Network Neuroscience*, 3(3):827–847, 2019.

[162] David P Shorten, Richard E Spinney, and Joseph T Lizier. Estimating transfer entropy in continuous time between neural spike trains or other event-based data. *PLoS computational biology*, 17(4):e1008054, 2021.

[163] Raul Vicente, Michael Wibral, Michael Lindner, and Gordon Pipa. Transfer entropy—a model-free measure of effective connectivity for the neurosciences. *Journal of computational neuroscience*, 30(1):45–67, 2011.

[164] Jing Yu, V Anne Smith, Paul P Wang, Alexander J Hartemink, and Erich D Jarvis. Advances to Bayesian network inference for generating causal networks from observational biological data. *Bioinformatics*, 20(18):3594–3603, 2004.

[165] Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the national academy of sciences*, 113(15):3932–3937, 2016.

[166] Eugene Tan, Débora Corrêa, Thomas Stemler, and Michael Small. Backpropagation on dynamical networks. *arXiv preprint arXiv:2207.03093*, 2022.

[167] Jose Casadiego, Mor Nitzan, Sarah Hallerberg, and Marc Timme. Model-free inference of direct network interactions from nonlinear collective dynamics. *Nature communications*, 8(1):1–10, 2017.

[168] Wen-Xu Wang, Ying-Cheng Lai, and Celso Grebogi. Data based identification and prediction of nonlinear and complex dynamical systems. *Physics Reports*, 644:1–76, 2016.

[169] John G White, Eileen Southgate, J Nichol Thomson, Sydney Brenner, et al. The structure of the nervous system of the nematode Caenorhabditis elegans. *Philos Trans R Soc Lond B Biol Sci*, 314(1165):1–340, 1986.

[170] Jeffrey P Nguyen, Frederick B Shipley, Ashley N Linder, George S Plummer, Mochi Liu, Sagar U Setru, Joshua W Shaevitz, and Andrew M Leifer. Whole-brain calcium imaging with cellular resolution in freely behaving Caenorhabditis elegans. *Proceedings of the National Academy of Sciences*, 113(8):E1074–E1081, 2016.

[171] Annika LA Nichols, Tomáš Eichler, Richard Latham, and Manuel Zimmer. A global brain state underlies C. elegans sleep behavior. *Science*, 356(6344):eaam6851, 2017.

[172] Jakob Runge. Causal network reconstruction from time series: From theoretical assumptions to practical estimation. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 28(7):075310, 2018.

[173] Cees Diks and Hao Fang. Transfer entropy for nonparametric Granger causality detection: an evaluation of different resampling methods. *Entropy*, 19(7):372, 2017.

[174] Angeliki Papana, Catherine Kyrtsou, Dimitris Kugiumtzis, and Cees Diks. Assessment of resampling methods for causality testing: A note on the us inflation behavior. *PloS one*, 12(7):e0180852, 2017.

[175] Phillip Good. *Permutation tests: a practical guide to resampling methods for testing hypotheses*. Springer Science & Business Media, 2013.

[176] Phillip I Good. *Resampling methods*. Springer, 2006.

[177] Gemma Lancaster, Dmytro Iatsenko, Aleksandra Pidde, Valentina Ticcinelli, and Aneta Stefanovska. Surrogate data for hypothesis testing of physical systems. *Physics Reports*, 748:1–60, 2018.

[178] Milan Paluš. From nonlinearity to causality: statistical testing and inference of physical mechanisms underlying complex dynamics. *Contemporary physics*, 48(6):307–348, 2007.

[179] Oliver M Cliff, Leonardo Novelli, Ben D Fulcher, James M Shine, and Joseph T Lizier. Assessing the significance of directed and multivariate measures of linear dependence between time series. *Physical Review Research*, 3(1):013145, 2021.

[180] James Theiler, Stephen Eubank, André Longtin, Bryan Galdrikian, and J Doyne Farmer. Testing for nonlinearity in time series: the method of surrogate data. *Physica D: Nonlinear Phenomena*, 58(1-4):77–94, 1992.

[181] R Quian Quiroga, A Kraskov, T Kreuz, and Peter Grassberger. Performance of different synchronization measures in real data: a case study on electroencephalographic signals. *Physical Review E*, 65(4):041903, 2002.

[182] D Kugiumtzis. Surrogate data test for nonlinearity including nonmonotonic transforms. *Physical Review E*, 62(1):R25, 2000.

[183] Dimitris N Politis and Joseph P Romano. The stationary bootstrap. *Journal of the American Statistical association*, 89(428):1303–1313, 1994.

[184] Xue Wang, Yonghong Chen, and Mingzhou Ding. Testing for statistical significance in bispectra: a surrogate data approach and application to neuroscience. *IEEE Transactions on Biomedical engineering*, 54(11):1974–1982, 2007.

[185] Shigeyoshi Fujisawa, Asohan Amarasingham, Matthew T Harrison, and György Buzsáki. Behavior-dependent short-term assembly dynamics in the medial prefrontal cortex. *Nature neuroscience*, 11(7):823–833, 2008.

[186] Joseph T Lizier, Jakob Heinzle, Annette Horstmann, John-Dylan Haynes, and Mikhail Prokopenko. Multivariate information-theoretic measures reveal directed information structure and task relevant changes in fmri connectivity. *Journal of computational neuroscience*, 30(1):85–107, 2011.

[187] Masanori Shimono and John M Beggs. Functional clusters, hubs, and communities in the cortical microconnectome. *Cerebral Cortex*, 25(10):3743–3757, 2015.

[188] Stavros I Dimitriadis, George Zouridakis, Roozbeh Rezaie, Abbas Babajani-Feremi, and Andrew C Papanicolaou. Functional connectivity changes detected with magnetoencephalography after mild traumatic brain injury. *NeuroImage: Clinical*, 9:519–531, 2015.

[189] Elzbieta Olejarczyk, Laura Marzetti, Vittorio Pizzella, and Filippo Zappasodi. Comparison of connectivity analyses for resting state eeg data. *Journal of neural engineering*, 14(3):036017, 2017.

[190] Matthieu Gilson, Adrià Tauste Campo, Xing Chen, Alexander Thiele, and Gustavo Deco. Nonparametric test for connectivity detection in multivariate autoregressive networks and application to multiunit activity data. *Network Neuroscience*, 1(4):357–380, 2017.

[191] Saša Singer and John Nelder. Nelder-Mead algorithm. *Scholarpedia*, 4(7):2928, 2009.

[192] Joon Lee, Shamim Nemati, Ikaro Silva, Bradley A Edwards, James P Butler, and Atul Malhotra. Transfer entropy estimation and directional coupling change detection in biomedical time series. *Biomedical engineering online*, 11(1):1–17, 2012.

Curriculum Vitae

Updated: 21 July, 2022

Amitava Banerjee

Personal Information

Pronouns	he/him, or they/them
Date of Birth	January 8, 1996
Nationality	Indian
Office Address	3353 A.V. Williams Bldg. 8223 Paint Branch Dr. College Park, 20740 Maryland, USA
Phone	+1 240 413 3686
Designation	Graduate Research Assistant (Fall 2018 - Present) Department of Physics Institute for Research in Electronics and Applied Physics (IREAP) University of Maryland, College Park, USA 20742
Email	amitavab@umd.edu, amitava8196@gmail.com
Linkedin	amitava-banerjee-physics
🐦	amitava_physics
	github.com/banerjeeamitava
Website	banerjeeamitava.github.io

Academic Qualifications

Degree	**Institute**	**Subjects**	**Year**
Bachelor of Science	Presidency University	Physics	2016
Master of Science	Presidency University	Physics	2018
Doctor of Philosophy	University of Maryland	Physics	2018-present

Current Research Interests

In my doctoral work, I am broadly interested in understanding what artificial neural networks (ANNs) can discover by observing natural dynamics of complex systems - from the climate to the brain. My work shows that ANNs trained to mimic the dynamical behaviors of networks can be applied to infer causal interactions in the network. In cases where the actual network is unknown, ANNs trained on their dynamics can be even used to reconstruct the network and predict its future dynamics. Some specific applications of my work are listed.

- Development of machine learning techniques for analysis of networked dynamical systems
- Causal inference from time-series data in noisy and partially sampled systems with time-delayed interactions and applications in earth science and neuroscience.
- Inferring higher-order causal inference and applications in ecology and neuroscience.

Publications

ORCID ID 0000-0001-9241-3555
ArXiv Publication Link arxiv.org/a/banerjee_a_2
Google Scholar scholar.google.com\citations?user=JgqGlpyLGyUC (citations: 46, h-index: 4)

1. ‘Machine Learning for causal interaction estimation from near-whole-brain calcium imaging data’, **A. Banerjee**, S. Chandra, E. Ott (in preparation)

2. ‘Quantifying Interdependencies in Geyser Eruptions at the Upper Geyser Basin, Yellowstone National Park’, W. F. Fagan, A. Swain , **A. Banerjee**, H. Ranade, P. Thompson, P. P.A. Staniczenko, B. Flynn, J. Hungerford, S. Hurwitz, Journal of Geophysical Research: Solid Earth, e2021JB023749 (2022).

3. ‘Machine Learning Link Inference of Noisy Delay-coupled Networks with Opto-Electronic Experimental Tests ’, **A. Banerjee**, J. D. Hart, R. Roy, E. Ott, Preprint: Physical Review X, 11, 031014 (2021).

4. ‘Using machine learning to assess short term causal dependence and infer network links’, **A. Banerjee**, J. Pathak, R. Roy, J. G. Restrepo, E. Ott, Chaos 29, 121104 (2019).

5. ‘An interplay of topology and quantized geometric phase for two different symmetry-class Hamiltonians’, S. Rahul, Y. R. Kartik, **A. Banerjee**, S. Sarkar, Physica Scripta 94 115803 (2019).

6. ‘Transient Chaos Generates Small Chimeras’, **A. Banerjee** and D. Sikder, Physical Review E 98, 032220 (2018). Article figure was featured in Phys. Rev. E Kaleidoscope for September, 2018..

7. ‘Bacteria Around an Acoustic Black Hole: Trapping and Frame-Dragging’, **A. Banerjee**, R. Koley, P. Majumdar, Preprint: arXiv:1808.01828 (2018).

8. ‘Dynamical Phase Transitions in Generalized Kuramoto Model with Distributed Sakaguchi phase’, **A. Banerjee**, Journal of Statistical Mechanics: Theory and Experiment 113402 (2017).

9. ‘Spatiotemporal Dynamics of the Kuramoto-Sakaguchi Model with Time-dependent Connectivity’, **A. Banerjee** and M. Acharyya, Phys. Rev. E 94, 022213 (2016). Article figure was featured in Phys. Rev. E Kaleidoscope for August, 2016.

- **Popular articles on my research**

‘Artificial intelligence discovers long-term influencers hiding in noisy systems’, Science X Dialog, July 26, 2021.

‘Neuromorphics for Network Discovery’, University of Maryland Physics Research News, July 21, 2021.

‘Neural Networks and Hidden Figures’, University of Maryland Physics graduate student spotlight, September 4, 2020.

‘An Unexpected Collaboration’, Letter to the editor of *Physics* magazine, April 13, 2020.

Awards and Fellowships

1. Travel Award, Society for Industrial and Applied Mathematics (SIAM) Conference on Dynamical Systems 2021
2. Honorable Mention, Ralph Myers and Friends of Physics Award for exceptional graduate teaching assistantship, Spring 2019 and Spring 2020
3. Graduate school offers from physics departments of University of Maryland - College Park (with Dean’s Fellowship), University of Colorado - Boulder, State University of New York - Buffalo (with Presidential Fellowship), University of Rochester, Georgia Institute of Technology, University of Waterloo (with Mike and Ophelia Lazaridis Fellowship)
4. American Physical Society (APS) Distinguished Student Fellowship 2017 and invitation for oral presentation at the APS March Meeting 2018.
5. Kishore Vaigyanik Protsahan Yojana scholarship, Indian Institute of Science, 2013 (awarded to promising undergraduates who have the potential to become future scientist).
6. INSPIRE Fellowship, Department of Science and Technology, India, 2013, (awarded to the top 1% students out of approximately 300,000 students at the higher secondary (+2) level, who are pursuing bachelors degrees in the sciences).

Presentations and Conferences

1. Contributed Talk, “Using Machine Learning for Inference of *C. elegans* Neural Network Structure from Calcium Fluorescence Time Series”, Annual Meeting of the International Physics of Living Systems (iPoLS) Network, June 3, 2022.
2. Contributed Talk, “Inference of *C. elegans* neural network structure from calcium flurescence time series data with reservoir computing, American Physical Society March Meeting, March 14, 2022.
3. Contributed Talk, “Neural connectivity inference from *C. elegans* whole-brain calcium-imaging data with reservoir computing”, Satellite Meeting of the Conference on Complex Systems: Machine Learning Perspectives of Complex Networks, October 28, 2021.

4. Contributed Talk, "Reservoir Computing for Link Inference of Noisy Networks from Time Series Data: Theory and Experiments", Society for Industrial and Applied Mathematics (SIAM) Conference on Dynamical Systems, May 23, 2021.
5. Contributed Talk, "Neuromorphics for network inference: new techniques and validation in opto-electronic experiments", American Physical Society March Meeting, March 19, 2021.
6. Invited Talk (co-presented with Joseph Hart), "Machine Learning for Non-invasive Network Link Inference", Complex Systems Society (CCS) Satellite Conference: Machine Learning Prospective for Complex Networks, held virtually, December 9, 2020.
7. Invited Talk, "Using machine learning to assess short term causal dependence and infer network links", Applied Dynamics Seminar, Department of Physics and IREAP, University of Maryland - College Park, October 17, 2019.
8. Invited Pedagogical Lecture, "Machine Learning in Biology", Summer Internship Program, Center for Bioinformatics and Computational Biology (CBCB), University of Maryland - College Park, July 17, 2019.
9. Invited Talk, "Transiently Chaotic behavior in Superconducting Metamaterials", Applied Dynamics Seminar, Department of Physics and IREAP, University of Maryland - College Park, November 15, 2018.
10. Contributed Talk, "Spatiotemporal Dynamics of the Kuramoto-Sakaguchi Model with Time-dependent Connectivity", American Physical Society March Meeting, March 7, 2018.

Professional Activities

1. **Reviewer**: Physical Review Letters (1 review, ongoing), Physical Review E (4 reviews), Physical Review Research (2 reviews), Physical Review Applied (2 reviews), Chaos: An Interdisciplinary Journal of Nonlinear Science (10 reviews)
2. **Co-organizer**, Conference on Complex Systems Satellite: Artificial Intelligence in Complex Networks, Palma de Mallorca+online, 19 October, 2022.
3. **Co-organizer**, Conference on Complex Systems Satellite: Machine Learning Perspectives of Complex Networks II, Lyon+online, 28 October, 2021.
4. **Volunteer and Co-lead**, Graduate Resources for Advancing Diversity with Maryland Astronomy and Physics (GRAD-MAP) (2020-Present).
5. Member, IREAP-ROLE task force on Diversity, Equity, and Inclusion (2020-Present).
6. Member, Physics of Living Systems - Student Research Network (2020-Present).
7. Mentor, Toolkit for Success (2022 Summer and 2021 Summer).
8. Member of the planning committee, University of Maryland Multicultural Involvement Community Advocacy events for Asian American Pacific Islander and Desi American Heritage Month 2022 (Spring 2022).

9. Teaching assistant and curriculum developer, The Computational Research Access Network (CRANE) (Spring 2022).
10. Curriculum developer, Girls Talk Math at UMD (Summer 2021).
11. Social Media and Website Management Officer,Women in Physics at the University of Maryland (2019-2020).
12. Student member of the American Physical Society and member of APS Forums On Graduate Student Affairs (FGSA) and Outreach and Engaging the Public (FOEP) (2017-Present).
13. Chair of Scientific Organizing Committee Undergraduate Physics Symposium, Presidency University, April 2017.

References

1. **Prof. Edward Ott**, edott@umd.edu
2. **Prof. Rajarshi Roy**, rroy@umd.edu
3. **Prof. Michelle Girvan**, girvan@umd.edu