Causal Local States: Achieving Scalable Simultaneous Causal Network Inference and Forecasting for Dynamical Systems

Master's Thesis at the Department of Physics
Ludwig-Maximilians-Universität München
Supervisor: Dr. Christoph Räth

April 1, 2026

submitted by

Jonas Braun

Causal Local States: Skalierbare simultane kausale Netzwerkinferenz und Vorhersage für Dynamische Systeme

Masterarbeit an der Fakultät für Physik
Ludwig-Maximilians-Universität München
Betreuer: Dr. Christoph Räth

April 1, 2026

vorgelegt von

Jonas Braun

ACKNOWLEDGEMENT

I would like to thank all the people who supported me throughout my thesis project.

First and foremost, Christoph for making this thesis possible. He provided me with the freedom to pursue my own ideas while also offering valuable guidance whenever I risked going too far off track.

Sebastian, Daniel, and Fabian for listening to my presentations, asking insightful questions, and discussing ideas with me.

My family for always supporting and encouraging me.

And last but not least, all the friends I met during my bachelor's and master's studies at LMU. Without them, the many hours of studying would have been far more difficult and much less enjoyable.

CONTENTS

LIST OF FIGURES

LIST OF ALGORITHMS

LIST OF TABLES

CHAPTER 1
INTRODUCTION

Countless decisions are made every day. Individuals make choices in their daily lives, businesses optimize strategies, and governments implement policies. In many of these cases, taking informed decisions relies on our ability to anticipate future developments.

However, many real-world systems are inherently high-dimensional, nonlinear, and governed by complex interactions. These properties make reliable prediction a challenging task. Recently, modern machine learning methods have demonstrated remarkable success on a wide range of problems. Yet, these approaches are frequently treated as black-box models, providing accurate forecasts without offering insight into the underlying system mechanisms. However, understanding the underlying system can be just as important as accurate predictions. Consequently, methods that achieve good results while retaining interpretability are desirable.

This thesis introduces the causal local states (CLS) algorithm, which, by incorporating ideas from causal discovery into a prediction task, tries to build a part toward more explainable machine learning. CLS builds upon the ideas of Pathak et al.[1], who introduced the concept of decomposing high-dimensional systems into smaller, more manageable subsystems to then generate a full system prediction. This idea was subsequently generalized by Baur et al. to account for non-local interactions with generalized local states (GLS) [2]. CLS extends these approaches by incorporating concepts from causal discovery, resulting in a fully data-driven framework that simultaneously infers the underlying network structure and performs forecasting.

While this thesis demonstrates the CLS framework using reservoir computing (RC) and next-generation reservoir computing (NGRC) as models and transfer entropy (TE) and convergent cross mapping (CCM) as causal measures, it is important to emphasize that the approach itself is both model- and causal-measure agnostic.

Even though there are a lot of causal discovery algorithms [3], only a limited number of studies that use RC (or variants) to directly connect network inference to predictive performance have been conducted. Li et al. introduced an RC-based framework that uses one-step prediction error to evaluate different neighborhood configurations and subsequently encode the local coupling structure directly into the reservoir [4]. Further, Srinivasan et al. provide a brief example of combining transfer entropy with their parallel RC scheme, where they

optimize the causal threshold as a global hyperparameter [5]. Similarly, Chu et al. incorporate the coupling structure into a parallel RC setup by calculating the transfer entropy matrix, fixing the maximum neighborhood size to the top q nodes and optimizing q as a global hyperparameter [6].

The thesis is split between a theory part, the introduction of the CLS method, and the results chapter. Chapters 2 and 3 establish the theoretical foundation of this work, introducing core concepts from graph theory, assumptions underlying causal inference, and bivariate causal measures. Chapter 4 reviews fundamental aspects of dynamical systems and introduces the benchmark datasets used throughout this thesis. Chapter 5 details the prediction framework, including performance metrics, employed models, and the GLS approach. The full CLS algorithm in its final form is presented in Chapter 6, while Chapter 7 shows the final results and the experiments that motivated key design decisions.

CHAPTER 2
INTERACTIONS, GRAPHS AND NETWORKS

Graphs arise naturally when studying dynamical systems with multiple variables interacting with each other. When interactions are pairwise, the underlying structure can be conveniently represented by a graph (or network), in which nodes correspond to dynamical variables and edges encode their interactions. In particular, when modeling causal relationships, these graphs are typically directed, allowing for the representation of asymmetric influences between variables [7, Chapter 1]. A more general framework to adequately capture higher-order dependencies (e.g. hypergraphs) is beyond the scope of this thesis.

2.1 Basics of graph theory

This section provides a concise overview of fundamental concepts in graph theory that are relevant for the construction and analysis of causal graphs. For comprehensive introductions, see [8], [9], and [10], upon which this section is based.

Simple graphs: The most basic object in graph theory is the simple graph. A simple graph is an ordered pair

$$G = (V, E), \tag{2.1}$$

where V is a finite set, and

$$E \subseteq \mathcal{P}_2(V). \tag{2.2}$$

$\mathcal{P}_2(V)$ is the set of all two-element subsets of V [9]. The set V is called the node or vertex set and E the edge set. Each element of E represents an undirected edge connecting two distinct nodes. Two nodes $u, v \in V$ are said to be adjacent if

$$\{u, v\} \in E, \tag{2.3}$$

i.e., if they are directly connected by an edge [9, 8, 10].

Directed graphs or digraphs: Simple graphs can be generalized to incorporate the directionality of interactions, which is important for causality. In a directed graph (or digraph), each edge is assigned an orientation. A digraph is defined as

$$D = (V, E), \tag{2.4}$$

where V is a finite set and

$$E \subseteq V \times V. \tag{2.5}$$

Each element $(u, v) \in E$ is called an arc, representing a directed connection from the source node u to the target node v [9]

Adjacency matrix representation: Both simple graphs and digraphs can be represented compactly using adjacency matrices. An adjacency matrix encodes the full graph structure in a $V \times V$ matrix. The adjacency matrix $A(G)$ of a graph (or digraph) G is defined as

$$A(G) = (a_{uv})_{u,v \in V}, \tag{2.6}$$

where

$$a_{uv} = \begin{cases} 1 & \text{if } \{u, v\} \in E \quad \text{(undirected graph)}, \\ 1 & \text{if } (u, v) \in E \quad \text{(directed graph)}, \\ 0 & \text{otherwise.} \end{cases} \tag{2.7}$$

For simple graphs, the adjacency matrix is symmetric, i.e., $a_{uv} = a_{vu}$. In contrast, this symmetry does not generally hold for directed graphs [9].

Basic graph properties: A fundamental local property of a graph is the degree of a node. For a node $v \in V$, the degree is defined as

$$\deg v = |\{e \in E \mid v \in e\}|.$$

In directed graphs, this notion splits into two quantities:

$$\deg^+ v = \text{(number of arcs with source v)}$$
$$\deg^- v = \text{(number of arcs with target v)},$$

where $\deg^+(v)$ is the outdegree and $\deg^-(v)$ the indegree of node v.

In some settings, it is suitable to think of graphs as networks where it is possible to move between nodes along the edges. A **walk** of length k in $G = (V, E)$ is a sequence of nodes

$$(v_0, v_1, \ldots, v_k), \tag{2.8}$$

such that (v_{i-1}, v_i) (or $\{v_{i-1}, v_i\}$ in the undirected case) is an edge for all $i = 1, \ldots, k$.

A path is a walk in which all vertices are distinct. A cycle is a path of length $k > 1$ in which the first and last vertices coincide, i.e., $v_0 = v_k$ [9].

2.2 Causal graphs

One of the goals of the thesis is to reconstruct an approximation of the underlying causal structure of a dynamical system in order to improve predictive performance. Causal graphs have several distinctive characteristics. Most importantly, causal relationships between variables are inherently directional, making directed graphs a natural representation. Additionally, the underlying graph must be acyclic. Cycles in a causal structure would create "feedback" loops, in which a variable could indirectly cause itself. By Pearl's definition of causality, such self-causation is explicitly excluded [7, p. 55]. Consequently, causal networks are commonly represented as directed acyclic graphs (DAGs).

When discussing causal graphs, it is common to adopt kinship terminology, such as "parents," "ancestors," and "descendants" [7, p. 13]. The meaning of these terms transfers naturally from their original interpretations. The parents $\mathrm{Pa}(v)$ of a node v are all nodes with edges pointing to v. The classic semantics by which a causal graph can be understood is that a directed edge $u \to v$ represents a causal influence from u to v [11]. However, a probabilistic DAG does not inherently encode causal information, but only conditional dependencies [11]. To interpret a DAG as a true causal model, additional assumptions about the underlying system are required, which will be discussed in Chapter 3.

Runge [12] expands on the idea of Eichler [13] to use time series graphs to represent causal interactions in time series data. A time series graph encodes the full interaction structure, specifying which variable causally influences which at specific time lags.

Definition 1 (Time series graph [12])**.** *Let $\mathbf{X} = (X^1, \dots, X^d)$ be a multivariate discrete-time stochastic process indexed by $t \in \mathbb{Z}$. The time series graph of $\mathbf{X}$ is a directed graph $G = (V \times \mathbb{Z}, E)$ with node set*

$$V \times \mathbb{Z} = \{X_t^p \mid p \in \{1, \dots, d\}, t \in \mathbb{Z}\}.$$

A directed edge

$$(X_{t-\tau}^p) \to (X_t^q)$$

is contained in E if and only if $X_{t-\tau}^p$ is a (direct) cause of X_t^q for some lag $\tau \in \mathbb{N}_0$. Thus, edges explicitly encode causal interactions at specific time delays.

For certain applications, including Causal Local States (Chapter 6), time series graphs are too detailed, as CLS focuses on spatial feature selection (which variables are important) rather than temporal feature selection (which time lags are important). Under the assumption of stationarity, the causal interactions are time-invariant, allowing a more compact representation called the summary graph.

Definition 2 (Summary graph [14])**.** *Let $\mathbf{X} = (X^1, \dots, X^d)$ be a multivariate discrete-time stochastic process, and let $G = (\{X^1, \dots, X^d\}, E)$ denote the associated summary causal graph. For $p, q \in \{1, \dots, d\}$, there exists a directed edge $X^p \to X^q$ in E if and only if there exists any time point $t \in \mathbb{Z}$ and a lag $i \in \mathbb{N}_0$ such that X_{t-i}^p causes X_t^q, where*

$$\begin{cases} 0 \le i & \text{if } p \ne q, \\ 0 < i & \text{if } p = q. \end{cases}$$

In other words, two nodes are connected in the summary graph whenever at least one causal interaction exists at some time lag.

A concept that will be used frequently throughout the series is a neighborhood. In this thesis, a neighborhood is defined around a core (one or multiple nodes) and includes all nodes and connections that end at the core nodes. While a neighborhood can be defined for multiple core nodes, this thesis only focuses on single nodes as cores of a neighborhood. Formally, this thesis defines a neighborhood as

$$\mathcal{N}(X^i) = \Big(\{X^i\} \cup \mathrm{Pa}(X^i), \{X^k \to X^i \ : \ X^k \in \mathrm{Pa}(X^i)\}\Big), \tag{2.9}$$

where $\mathrm{Pa}(X^i)$ denotes the parent set of X^i in the summary graph. While neighborhoods could also be defined for time series graphs, since CLS does not make use of the temporal feature selection, it is sufficient to just define the neighborhoods on the summary graph. Furthermore, this way only $|V|$ neighborhoods centered around different core nodes are necessary to cover the full system.

This definition is close to Pearl's concept of a family [7, p. 13] but distinguishes explicitly between the core node and its parents. The neighborhood concept is adapted from [2].

2.3 Graph reconstruction

One of the central objectives of this thesis is to reconstruct an approximation of the underlying causal graph of a dynamical system. To assess the quality of a reconstruction method, suitable evaluation metrics must be defined. This section is based on a paper of Delabays et al., who in applied standard classification metrics to quantify hypergraph reconstruction [15].

Given a reconstructed graph, its edge set can be compared to the ground-truth causal structure defined by the governing equations of the system. Each inferred edge can then be classified as either a true positive (TP), false positive (FP), true negative (TN), or false negative (FN). Based on these quantities, standard performance metrics can be defined.

The true positive rate (TPR) measures the proportion of correctly identified causal edges among all edges:

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}. \tag{2.10}$$

A high TPR indicates that most existing causal relationships are successfully recovered.

The false positive rate (FPR) quantifies the proportion of incorrectly inferred edges among all edges:

$$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}}. \tag{2.11}$$

A low FPR implies that only few spurious connections are introduced.

However, the FPR alone does not fully capture the impact of reconstruction errors in the context of prediction. In particular, it treats all false positive edges equally, regardless of their structural role within the network. While this is appropriate for pure graph reconstruction tasks (e.g., causal discovery), it may be insufficient when the reconstructed graph is subsequently used to support prediction. In such cases, a spurious edge between dynamically related variables may be less detrimental than a false long-range connection between otherwise unrelated parts of the system, as the former may still provide useful predictive information.

To the best of our knowledge, there are currently no standard evaluation measures that explicitly account for the predictive relevance of reconstruction errors. Therefore, despite its limitations, the FPR is retained as an evaluation metric in this thesis.

By varying a decision parameter of the reconstruction method, such as a threshold or neighborhood size, different pairs of TPR and FPR are obtained. Plotting the TPR against the FPR yields the receiver operating characteristic (ROC) curve, which characterizes the trade-off between correctly identifying edges and introducing false positives across different parameter settings.

A common summary statistic of the ROC curve is the area under the curve (AUC), defined as

$$\mathrm{AUC} = \int_0^1 \mathrm{TPR}(\mathrm{FPR})\, d(\mathrm{FPR}). \tag{2.12}$$

An AUC value of 1 corresponds to perfect reconstruction, whereas a value of 0.5 indicates performance equivalent to random guessing.

CHAPTER 3
CAUSALITY

Inferring causal relationships is a fundamental challenge across many scientific disciplines [12]. Establishing causality directly typically requires controlled experiments, which are often infeasible, costly, or ethically problematic [12]. Consequently, a wide range of methods has been developed to infer causal structure from observational data. Among these, bivariate approaches such as Granger causality (GC) [16], transfer entropy (TE) [17], and convergent cross mapping (CCM) [18] are used. In addition, more multivariate algorithms, such as the PC algorithm, have been proposed to address more complex causal discovery problems. A comprehensive overview of causal discovery methods is beyond the scope of this thesis. The reader is referred to [3] for an extensive survey of existing approaches.

The notion of causality employed in causal local states is twofold. First, causal measures are employed as scoring functions $S(X, Y)$ between pairs of variables in a preprocessing step. For this reason, this thesis focuses on bivariate causality measures such as GC, TE, and CCM. Second, CLS relies on an approximation of the underlying causal network structure for prediction.

To understand both roles of causality in CLS, we begin by first introducing the general ideas behind causality.

3.1 Basics of causal inference

To formalize causality, Pearl introduces the do(.) operator. While observing $X = x$ corresponds to conditioning on an observed value, $do(x)$ represents actively setting the variable X to x. The causal effect of X on Y can then be assessed by evaluating $P(Y \mid do(x))$ for all x and examining how this distribution changes with x [7]. If changes in x lead to changes in the distribution of Y, X is said to have a causal influence on Y.

Causal relationships are commonly represented using directed acyclic graphs (DAGs), referred to as causal graphs. Given a set of variables V, a causal graph is defined as a DAG $G = (V, E)$, where each vertex corresponds to a variable and a directed edge from X_i to X_j represents a causal influence of X_i on X_j [19].

In order to apply causal inference, certain assumptions about the data must be made. Runge [12] highlights three key assumptions: causal sufficiency, the causal Markov condition, and faithfulness.

Definition 3. *Causal Sufficiency*

A set of variables V is causally sufficient if it contains all common causes of possible pairs in V [12]

This assumption implies that there are no unobserved variables acting as direct or indirect causal drivers of the observed variables. In practice, however, causal sufficiency is rarely satisfied, as many relevant quantities are only indirectly observed through proxy measurements.

While causal discovery methods such as modified PC or FCI explicitly account for hidden variables [19], this thesis adopts a different perspective. Since CLS is ultimately concerned with prediction rather than exact recovery of the full causal graph, removing spurious links caused by hidden drivers may even be detrimental. Such links can still carry predictive information and may therefore improve forecasting performance (see Figure 3.1).

To illustrate this, consider the following example.

Example 1. Spurious links can improve prediction

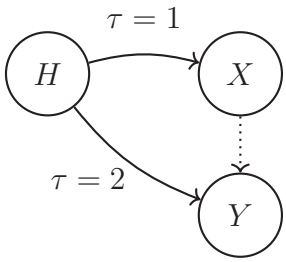


Figure 3.1: Hidden confounder inducing a spurious but predictive link

Consider a hidden stochastic process H_t evolving independently over time,

$$H_t \sim \mathcal{N}(0, 1).$$

Let the observed variables X_t and Y_t be generated as

$$X_t = H_{t-1} + \epsilon_t^X,$$
$$Y_t = H_{t-2} + \epsilon_t^Y,$$

where $\epsilon_t^X, \epsilon_t^Y \sim \mathcal{N}(0, \sigma^2)$ are independent noise terms. The true causal links are given by

$$H_{t-1} \to X_t, \quad H_{t-2} \to Y_t,$$

and there is no direct causal influence from X to Y. Although X_t does not causally influence Y_{t+1}, the two variables are statistically dependent through H:

$$\mathbb{E}[Y_{t+1} \mid X_t] \neq \mathbb{E}[Y_{t+1}].$$

This means that we expect predictive performance to increase when using X_{t-1} to predict Y even though there is no direct causal link $X \to Y$.

This example highlights that indirect causal relationships induced by hidden variables can be beneficial for prediction, even in the absence of direct causal relationships.

Definition 4. *Causal Markov Condition*

Let V be a set of variables with causal graph $G = (V, E)$. The graph satisfies the causal Markov condition if, for every variable $X \in V$, $\{X\}$ is conditionally independent of the set of all its nondescendants given the set of all its parents [20]. Denoting the set of non-descendants by $ND(X)$ and the parents by $Pa(X)$, this can be written as

$$X \perp\!\!\!\perp ND(X) \mid Pa(X). \tag{3.1}$$

The causal Markov condition provides the theoretical foundation for neighborhood-based prediction.It implies that the parents of a variable Y form a sufficient set for predicting Y. Consequently, a high-dimensional prediction problem can be decomposed into smaller local subproblems if the neighborhoods are chosen appropriately.

Importantly, the presence of additional (spurious) edges does not violate the Markov condition, as conditional independence with respect to non-descendants is preserved. However, from a practical perspective, minimizing neighborhood size remains desirable in order to speed up computation and improve interpretability of the neighborhood.

In order to construct an approximation of the underlying causal graph of a dynamical system, CLS uses ideas that build upon Granger's idea of causality.

3.2 Granger causality

Granger causality, introduced by Granger in 1969 [16], provides a framework for quantifying causal relationships between time series based on predictive performance. The central idea is that if the past of a variable X contains information that helps predict another variable Y, then X is said to Granger-cause Y.

Formally, let U denote the set of all relevant information for predicting Y. Then X Granger-causes Y if

$$\sigma^2(Y|U) < \sigma^2(Y|U/X), \tag{3.2}$$

where $\sigma^2(A \mid B)$ denotes the prediction error when predicting A using the information in B [16]. In practice, the full information set U is not accessible, and the test is typically performed using only the past values of the observed variables.

In its original formulation, Granger causality is implemented using a linear autoregressive model:

$$\hat{X}_t = P_t(X \mid X_{\text{past}}, Y_{\text{past}}) = \sum_{\tau=1}^{\tau_{\max}} a_\tau X_{t-\tau} + \sum_{\tau=1}^{\tau_{\max}} b_\tau Y_{t-\tau} + \epsilon_t, \tag{3.3}$$

where $\hat{X}_t$ is the prediction of X_t, and the coefficients a_τ and b_τ are chosen to minimize the prediction error $\sigma^2(X \mid X_{\text{past}}, Y_{\text{past}})$.

The interpretation of Granger causality as "true" causality remains debated. For instance, Pearl characterizes it as lying at the boundary between purely statistical and genuinely causal notions [7]. Moreover, the classical formulation has several limitations, including its reliance on linear models, fixed lag structures, and stationarity assumptions [21]. As a result, standard Granger causality is not well suited for causal discovery in complex nonlinear systems.

Nevertheless, within the context of this thesis, the objective is not to recover the exact causal structure, but rather to construct an approximate causal graph that is useful for prediction. From this perspective, Granger's notion of causality provides a practical and well-motivated framework.

3.3 Transfer entropy

A different approach to data-driven causality was introduced by Schreiber, who proposed an asymmetric, information-theoretic measure termed entropy (TE) [17]. TE quantifies the directed exchange of information between dynamical systems and thereby enables the inference of causal relationships from stationary time series.

3.3.1 An information-theoretic approach to causality

Since transfer entropy is rooted in information theory, we briefly review the necessary concepts. Let X be a discrete random variable that has possible values in $\mathcal{X} = \{x_1, x_2, \ldots, x_n\}$ with associated probabilities $p_1, p_2, \ldots, p_n$. Shannon postulated that a measure of uncertainty $H(p_1, \ldots, p_n)$ should satisfy the following properties [22]:

1. H is continuous in p_i
2. If all probabilities p_i are equal ($p_i = \frac{1}{n}$), H should be a monotonic increasing function in n.
3. If an event is split into multiple, the original H should be a weighted sum of the individual entropies.

The unique function satisfying these axioms is the Shannon entropy:

$$H(X) = -\sum_{i=1}^{n} p_i \log p_i, \tag{3.4}$$

where the logarithm base determines the unit of information.

This concept extends naturally to multiple variables. For two random variables X and Y, the joint entropy is defined as

$$H(X, Y) = -\sum_{x,y} p(x, y) \log p(x, y), \tag{3.5}$$

and the conditional entropy as

$$H(Y \mid X) = -\sum_{x,y} p(x, y) \log p(y \mid x). \tag{3.6}$$

These quantities are related via

$$H(X,Y) = H(X) + H(Y \mid X). \tag{3.7}$$

Entropy can be used to quantify the information overlap between X and Y by calculating the reduction of uncertainty about X if one had knowledge of a variable Y. This defines the mutual information [23]:

$$I(X;Y) = H(X) - H(X \mid Y) = \sum_{x,y} p(x,y) \log \frac{p(x,y)}{p(x)p(y)}. \tag{3.8}$$

However, mutual information is symmetric and therefore does not capture directional (causal) relationships.

To incorporate temporal structure and with it the directionality required for causality, consider a discrete-time stochastic process $\{X_n\}$. The process is said to be a Markov process of order k if

$$p(x_{n+1} \mid x_n, x_{n-1}, \dots) = p(x_{n+1} \mid x_n, \dots, x_{n-k+1}), \tag{3.9}$$

i.e., the future state depends only on the previous k states. The dynamics are thus fully characterized by the transition probabilities

$$p(x_{n+1} \mid x_n^{(k)}), \tag{3.10}$$

where $x_n^{(k)} = (x_n, x_{n-1}, \dots, x_{n-k+1})$ denotes the delay vector.

In the absence of information transfer from a process Y to a process X, the transition probabilities of X should be independent of the past of Y, such that

$$p(x_{n+1} \mid x_n^{(k)}) = p(x_{n+1} \mid x_n^{(k)}, y_n^{(l)}), \tag{3.11}$$

where $y_n^{(l)} = (y_n, \dots, y_{n-l+1})$ denotes the delay vector of Y [17]. Deviations from this condition can be quantified using the Kullback-Leibler divergence:

$$D(p\|q) = \sum_x p(x) \log \frac{p(x)}{q(x)}, \tag{3.12}$$

which quantifies the information-flow from Y to X. This defines the transfer entropy from Y to X,

$$T_{Y\to X} = \sum p(x_{n+1}, x_n^{(k)}, y_n^{(l)}) \log \frac{p(x_{n+1} \mid x_n^{(k)}, y_n^{(l)})}{p(x_{n+1} \mid x_n^{(k)})}, \tag{3.13}$$

which measures the directed information flow from Y to X [17]. Typical choices for the embedding dimensions are $l = k$ or $l = 1$.

An equivalent and more intuitive representation is obtained in terms of conditional entropies. Starting from

$$T_{Y\to X} = \sum_{x_{n+1},\, x_n^{(k)},\, y_n^{(l)}} p(x_{n+1}, x_n^{(k)}, y_n^{(l)}) \log \frac{p(x_{n+1} \mid x_n^{(k)}, y_n^{(l)})}{p(x_{n+1} \mid x_n^{(k)})}, \tag{3.14}$$

we first split the logarithm into a difference,

$$\begin{aligned} T_{Y\to X} &= \sum_{x_{n+1},x_n^{(k)},y_n^{(l)}} p(x_{n+1}, x_n^{(k)}, y_n^{(l)}) \Big[\log p(x_{n+1} \mid x_n^{(k)}, y_n^{(l)}) - \log p(x_{n+1} \mid x_n^{(k)}) \Big] \\ &= \sum p(x_{n+1}, x_n^{(k)}, y_n^{(l)}) \log p(x_{n+1} \mid x_n^{(k)}, y_n^{(l)}) \\ &\quad - \sum p(x_{n+1}, x_n^{(k)}, y_n^{(l)}) \log p(x_{n+1} \mid x_n^{(k)}). \end{aligned} \tag{3.15}$$

Using the definition of conditional entropy (Eq.3.6) and carrying out the sum over $y_n^{(l)}$ in the second term, we can rewrite the sums as entropies:

$$T_{Y\to X} = H(X_{n+1} \mid X_n^{(k)}) - H(X_{n+1} \mid X_n^{(k)}, Y_n^{(l)}). \tag{3.16}$$

This expression shows that transfer entropy quantifies the reduction in uncertainty of the future state X_{n+1} due to knowledge of the past of Y, beyond the information already contained in the past of X. This interpretation closely relates to Granger causality, and for Gaussian variables both measures are equivalent [24].

Transfer entropy is inherently asymmetric and model-free, making it particularly suitable for detecting directional interactions in nonlinear systems without requiring explicit assumptions about the underlying dynamics [25].

Since transfer entropy is defined via Shannon entropy, its extension to continuous variables is nontrivial. Although differential entropy provides a continuous analogue, it lacks important properties such as non-negativity, which complicates its interpretation in the context of transfer entropy. A detailed discussion of these issues is beyond the scope of this thesis and we refer the reader to [26].

In this thesis, transfer entropy is estimated using a histogram-based approach with uniformly spaced bins. While such estimators are known to introduce bias [26], this approach is sufficient here, as transfer entropy serves as a tool rather than the primary object of study.

3.4 Convergent cross mapping

Granger causality assumes that the underlying cause and effect can be separated. In order for Eq. 3.2 to be meaningful for causal inference, the influence of X on Y must be removable [27]. While this assumption may hold for simple or weakly coupled systems, it is generally violated in complex dynamical systems.

To address such cases, Sugihara et al. introduced Convergent Cross Mapping (CCM) in 2012, a method designed to detect causal relationships in systems where separability does not hold [18]. Since CCM is based on nonlinear state space reconstruction, we first review the relevant concepts from dynamical systems theory.

3.4.1 Embedding theory

In the context of embedding theory, a definition of dynamical systems needs some fundamental components [28]:

1. The phase space Γ
2. A notion of time, either continuous or discrete
3. A time evolution law governing the system's dynamic.

The phase space of a dynamical system can be modeled as a manifold M, i.e., an n-dimensional surface embedded in a higher-dimensional space [29]. Formally, a dynamical system in embedding theory is defined as follows:

Definition 5. *Dynamical system*

Let M be a compact manifold, then a dynamical system on M is defined by a diffeomorphism $\phi : M \to M$ (discrete time) or a vector field X (continuous time). [29], [30]

In an ideal setting, the full state $x \in M$ is observable. In practice, however, often only partial observations of the system are available. This corresponds to observing a projection of the underlying manifold, which may lead to overlaps of trajectories and loss of determinism, often referred to as singularities. An embedding is a transformation that preserves the topological and geometric structure of the manifold while avoiding such singularities [29]. If $\Phi : M \to \mathbb{R}^n$ is an embedding, then $\Phi(M)$ is diffeomorphic to M [31], allowing the reconstruction of dynamical properties from observed data. Consequently, constructing a valid embedding is of central importance to the study of dynamical systems.

A practical way to reconstruct a dynamical system from partial observations is with time-delay embeddings using Takens' embedding theorem. It was developed by Floris Takens in 1981 and provides the conditions for attractor reconstruction from a single observable function [30].

Definition 6. *Takens' Embedding Theorem*

Let M be a compact Manifold of dimension m, with a dynamical system $\phi : M \to M$ (or a vector field for continuous time). Given a smooth function $y : M \to \mathbb{R}$ (observable), for pairs (ϕ, y) the map $\Phi_{(\phi,y)} : M \to \mathbb{R}^{2m+1}$ defined as

$$\Phi_{(\phi,y)}(x) = (y(x), y(\phi(x)), \dots y(\phi^{2m}(x))), \tag{3.17}$$

defines an embedding [30].

The components of this embedding consist of time-delayed observations of a single scalar measurement. Hence, the theorem implies that the full dynamical structure of the system can, in principle, be reconstructed from a single observable. This result is fundamental to modern state space reconstruction methods.

3.4.2 Causality from state space reconstruction

A key implication of Takens' theorem is that the influence of one variable on another cannot, in general, be separated in reconstructed state spaces. This violates the separability assumption underlying Granger causality [18]. Sugihara et al. proposed Convergent Cross Mapping (CCM) as a complementary approach applicable to such inseparable systems [18]. The following part introducing CCM is largely based on Sugihara's original paper and the supplemental material [18].

Let $X = \{x_t\}_{t=1}^{L}$ and $Y = \{y_t\}_{t=1}^{L}$ denote two time series of length L (the library length). To reconstruct their respective state spaces, we construct delay embeddings with delay τ and embedding dimension E:

$$\mathbf{x}_t = (x_t, x_{t-\tau}, \dots, x_{t-(E-1)\tau}) \in \mathbb{R}^E,$$

where t runs from $t = 1 + (E-1)\tau$ to $t = L$. The sets of vectors $\{\mathbf{x}_t\}$ and $\{\mathbf{y}_t\}$ define the so-called shadow manifolds M_X and M_Y, respectively.

Since both embeddings are diffeomorphic to the original manifold M, neighborhoods on one manifold correspond to neighborhoods on the other if the variables are dynamically coupled.

CCM quantifies this correspondence via cross-mapping. To calculate the cross-map estimate at a time t from X to Y, one identifies the point $\mathbf{x}_t$ and its $E+1$ nearest neighbors on M_X. These neighbors define a local simplex in the embedding space[1]. The corresponding time indices are then used to obtain the associated values of Y. The estimate of y_t is computed as a locally weighted average:

$$\hat{y}_t \mid M_X = \sum_{i=1}^{E+1} w_i \, y_{t_i}. \tag{3.18}$$

The weights are defined as

$$w_i = \frac{u_i}{\sum_{j=1}^{E+1} u_j}, \quad u_i = \exp\left(-\frac{d(\mathbf{x}_t, \mathbf{x}_{t_i})}{d(\mathbf{x}_t, \mathbf{x}_{t_1})}\right), \tag{3.19}$$

where $d(\cdot, \cdot)$ denotes the Euclidean distance and $\mathbf{x}_{t_1}$ is the nearest neighbor.

The quality of the cross mapping from X to Y is assessed using the Pearson correlation between the observed values and their estimates:

$$\rho_{X \to Y}(L) = \mathrm{corr}(\{y_t\}_{t \in \mathcal{T}}, \{\hat{y}_t \mid M_X\}_{t \in \mathcal{T}}), \tag{3.20}$$

where $\mathcal{T}$ denotes the test indices.

If X and Y are causally coupled, increasing the library length L improves the sampling of the shadow manifolds, leading to more accurate nearest-neighbor approximations and, consequently, improved cross-mapping skill. In the limit of infinite data, the estimates will converge. In practice, however, measurement noise and finite sampling will limit this convergence. In this case, CCM is demonstrated by an increase in cross-map skill with increasing library length.

Once convergence is reached, CCM does not directly quantify the magnitude of causal strength. However, stronger causal interactions are reflected in faster convergence of the cross-mapping skill as a function of L. Thus, relative comparisons of causal strength between different links are possible when using consistent embedding parameters and library sizes.

3.4.3 Practical implementation and parameter selection

The performance of CCM depends sensitively on several hyperparameters: the embedding dimension E, the time delay τ, and the library size L.

Embedding dimension E: The embedding dimension E determines the number of delay coordinates used to reconstruct the shadow manifold. If E is chosen too small, the reconstructed attractor is not properly unfolded, leading to overlapping trajectories (singularities). Conversely, choosing E too large reduces the effective density of points in the reconstructed space, which slows convergence. [18, Suppl.]

[1] A minimum of $E+1$ points are needed to bound a point with a simplex [18, Suppl.]

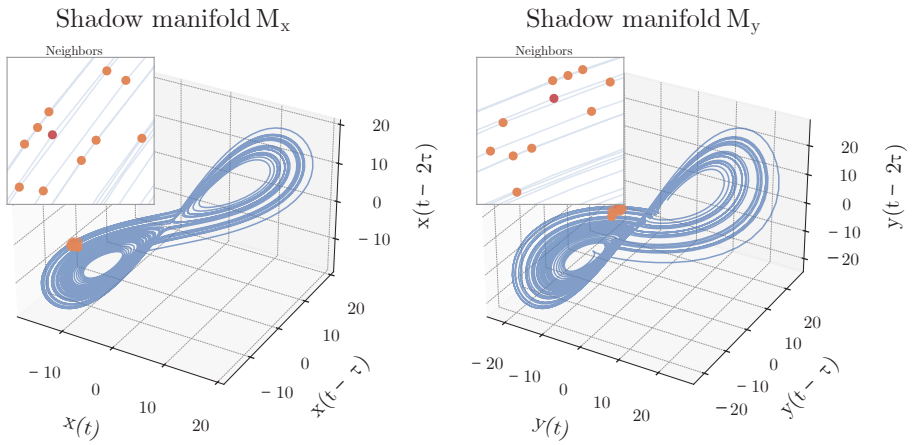


Figure 3.2: Shadow manifolds M_Y (left) and M_X (right) reconstructed from the Lorenz system using time-delay embedding with embedding dimension $E = 3$ and delay $\tau = 5$. Due to the dynamical coupling between variables X and Y, the neighborhood of a selected reference point on M_y remains spatially close when mapped onto M_x.

In practice, a commonly used method to estimate a suitable embedding dimension is the false nearest neighbors (FNN) algorithm. The FNN algorithm identifies spurious nearest neighbors that arise due to insufficient embedding dimension [32]:

1. For a given embedding dimension E, identify the nearest neighbors of each delay vector.
2. Increase the embedding dimension to $E + 1$ and recompute the distances between previously identified neighbors.
3. Classify neighbors as false if the relative increase in distance exceeds a predefined threshold.
4. Select the smallest embedding dimension for which the fraction of false neighbors approaches zero.

Time delay τ: The time delay τ controls the temporal spacing between components of the delay vectors. If τ is too small, successive components are highly correlated and provide redundant information. On the other hand, if τ is too large, the components become effectively independent and fail to capture the underlying dynamics. The objective is therefore to balance redundancy and independence.

Two standard heuristics are commonly used. The first is based on the autocorrelation function, selecting τ at the first zero crossing. The second approach relies on the self mutual information $I(x_t; x_{t+\tau})$, choosing τ at its first local minimum [33]. This criterion captures nonlinear dependencies and is therefore generally preferred in nonlinear settings.

Library size L: The library size L corresponds to the number of observations used to construct the shadow manifolds. In CCM, the cross-mapping skill is evaluated as a function of L. The causal inference relies on the convergence of the cross-map skill with increasing library size.

Algorithm 1 Convergent Cross Mapping (CCM)

Input: Time series $X = \{x_t\}_{t=1}^{L}$, $Y = \{y_t\}_{t=1}^{L}$, embedding dimension E, delay τ, library length L_{lib}

Output: Cross-mapping skill $\rho_{X\to Y}(L_{\text{lib}})$

The convergence is omitted, as CLS only needs to compare cross-mapping skills.

1: Construct delay embedding of X:

$$x_t = (x_t, x_{t-\tau}, \ldots, x_{t-(E-1)\tau}) \in \mathbb{R}^E, \quad t = 1 + (E-1)\tau, \ldots, L$$

2: Form shadow manifold $M_X = \{x_t\}$

3: **for** each time index t in testing set T **do**

4: Identify $E+1$ nearest neighbors of x_t in M_X

5: Compute weights:

$$u_i = \exp\Big(- d(x_t, x_{t_i})/d(x_t, x_{t_1})\Big), \quad w_i = \frac{u_i}{\sum_{j=1}^{E+1} u_j}$$

6: Estimate $\hat{y}_t \mid M_X = \sum_{i=1}^{E+1} w_i y_{t_i}$

7: Compute cross-mapping skill:

$$\rho_{X\to Y}(L_{\text{lib}}) = \operatorname{corr}\Big(\{y_t\}_{t\in T}, \{\hat{y}_t \mid M_X\}_{t\in T}\Big)$$

In this work, the primary objective is to compare causal influences across variables. Therefore, it is crucial to keep the library size fixed across all comparisons, ensuring that differences in cross-mapping performance reflect differences in causal structure rather than sample size effects. Under this constraint, relative convergence rates can be used as a proxy for causal strength.

CHAPTER 4
DYNAMICAL SYSTEMS

All systems used to develop and benchmark causal local states fall in the category of dynamical systems. This introduction is based on the material in [34], [35], and [28].

The main component of a dynamical system is its time evolution, which can be either stochastic or deterministic. In this thesis, only deterministic systems are considered, which means that the future trajectory is fully determined by past states. The time evolution can be either discrete or continuous, where we will focus on the latter. The state of a continuous-time dynamical system of dimension d at time t is represented by a d-dimensional vector $\boldsymbol{u}(t)$, and its evolution is defined as

$$\dot{\boldsymbol{u}}(t) = F(\boldsymbol{u}(t)), \tag{4.1}$$

where $F(.)$ is the flow. If the governing equations, and hence the system's flow, are nonlinear, the system exhibits nonlinear dynamics. Such a system might then exhibit complex trajectories or chaotic behavior.

There is no uniform definition of chaos. In his lecture about chaos, Richard Fitzpatrick classifies a chaotic system as a system that is aperiodic, bounded, and sensitive to its initial conditions. Aperiodicity means that each system state occurs only once over the full-time evolution. Boundedness requires that trajectories remain within a finite region of phase space. Lastly, sensitivity to initial conditions implies that trajectories starting from two nearby states $\boldsymbol{u}$ and $\boldsymbol{u} + \boldsymbol{\delta u}$ eventually diverge. [34]

4.1 Lyapunov exponents

A commonly used indicator of the chaotic behavior of a system are the Lyapunov exponents of the system. Lyapunov exponents quantify the rate at which nearby points in phase space diverge or converge during system evolution, which determines how fast a system becomes unpredictable [36]. Let $\boldsymbol{\delta_0}$ be the initial separation vector between two points in phase space. The Lyapunov exponent λ is then defined by the equation

$$|\boldsymbol{\delta}(t)| = |\boldsymbol{\delta_0}|e^{\lambda t}, \tag{4.2}$$

where $\boldsymbol{\delta}(t)$ is the separation vector at time t [36, Ch. 10.5]. There is a spectrum of Lyapunov exponents, one for each system dimension. However, when characterizing system behavior, it is usually sufficient to look at the maximal Lyapunov exponent (MLE) of the system [35, Ch. 5]. A positive MLE indicates chaos, whereas a negative MLE suggests that nearby trajectories converge. The MLE can be estimated using the Rosenstein algorithm [37].

4.2 Simulating continuous-time systems with RK4

Continuous-time dynamical systems are governed by ordinary differential equations, most of which do not have analytical solutions [38]. Consequently, numerical integration methods are required to approximate their behavior. A widely used family of integrators is the Runge-Kutta method.

In this thesis, the fourth-order Runge-Kutta method (RK4) is employed. RK4 discretizes time into steps of size Δt, approximating the continuous flow $F(\cdot)$ by a discrete-time map $M(\cdot)$ that propagates the state from time t to $t + \Delta t$:

$$\boldsymbol{u}_{t+1} = M(\boldsymbol{u}_t). \tag{4.3}$$

The RK4 map M is defined as

$$M(\boldsymbol{u}_t) = \boldsymbol{u}_t + \frac{\Delta t}{6}\left(\boldsymbol{k}_1 + 2\boldsymbol{k}_2 + 2\boldsymbol{k}_3 + \boldsymbol{k}_4\right), \tag{4.4}$$

where the intermediate slope evaluations are given by

$$\boldsymbol{k}_1 = F(\boldsymbol{u}_t), \tag{4.5}$$

$$\boldsymbol{k}_2 = F\left(\boldsymbol{u}_t + \frac{\Delta t}{2}\boldsymbol{k}_1\right), \tag{4.6}$$

$$\boldsymbol{k}_3 = F\left(\boldsymbol{u}_t + \frac{\Delta t}{2}\boldsymbol{k}_2\right), \tag{4.7}$$

$$\boldsymbol{k}_4 = F(\boldsymbol{u}_t + \Delta t\, \boldsymbol{k}_3). \tag{4.8}$$

This method allows for an approximation of the continuous-time flow, which is used to generate all datasets in this thesis. [39], [40], [41]

4.3 Low-dimensional chaotic systems

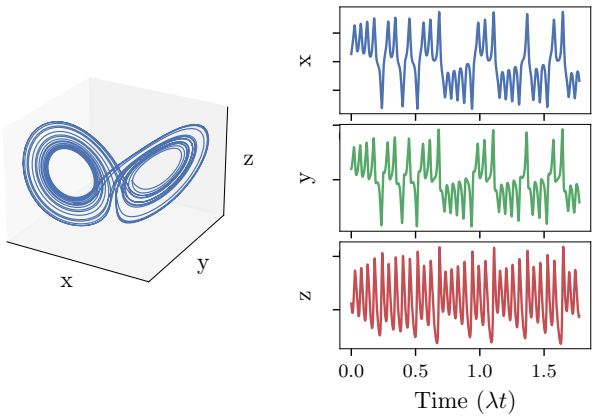
Although CLS is designed to be applicable to high-dimensional systems, low-dimensional chaotic systems provide a convenient starting point for algorithm development and testing. They are well-studied and allow for controlled experimentation.

One of the most studied chaotic systems is the Lorenz-63 system, introduced by Edward N. Lorenz to model atmospheric convection [42]. It is a three-dimensional continuous-time system defined by

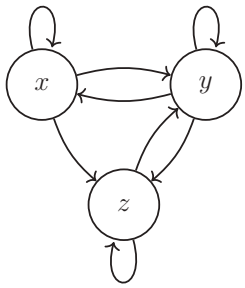
$$\dot{x}(t) = \sigma\Big(y(t) - x(t)\Big), \tag{4.9}$$

$$\dot{y}(t) = x(t)\Big(\rho - z(t)\Big) - y(t), \tag{4.10}$$

$$\dot{z}(t) = x(t)y(t) - \beta z(t), \tag{4.11}$$

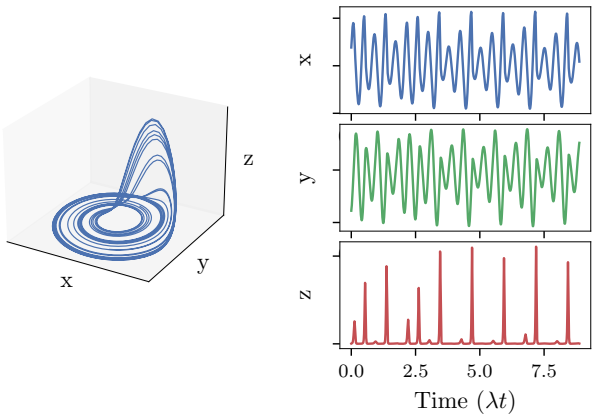


(a) Lorenz-63 attractor for $\sigma = 10$, $\rho = 28$, $\beta = 8/3$.

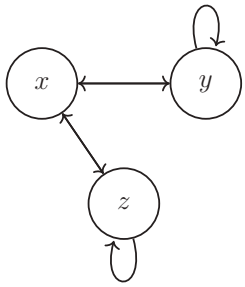


(b) Summary graph derived from the Lorenz equations.

Figure 4.1: Lorenz system overview. The chaotic attractor (left) and the corresponding summary graph encoding variable dependencies implied by the governing equations (right).



(a) Rössler attractor for $a = 0.2$, $b = 0.2$, $c = 5.7$.



(b) Summary graph derived from the Rössler equations.

Figure 4.2: Rössler system. The chaotic attractor (left) and the corresponding summary graph encoding variable dependencies implied by the governing equations (right).

where σ, ρ, and β are positive parameters. The standard parameter set $\sigma = 10$, $\rho = 28$, $\beta = 8/3$ produces the classic butterfly attractor (Figure 4.1).

Another canonical system is the Rössler system [43], a minimal three-dimensional system capable of exhibiting chaos. It is defined by

$$\dot{x}(t) = -y(t) - z(t), \tag{4.12}$$
$$\dot{y}(t) = x(t) + a\,y(t), \tag{4.13}$$
$$\dot{z}(t) = b + z(t)\Big(x(t) - c\Big), \tag{4.14}$$

where a, b, and c are real-valued system parameters.

With the standard choice $a = 0.2, b = 0.2, c = 5.7$, the system exhibits a chaotic attractor. The summary graph and attractor can be seen in Figure 4.2

The Rössler and Lorenz-63 systems were combined to create a 6-dimensional dataset used for benchmarking causal local states. There is no interaction between the two attractors, which makes it perfect to test the robustness for "false" data in the feature space.

4.4 Lorenz 96 system

The first higher-dimensional benchmark considered in this thesis is the Lorenz-96 (L96) system [44]. L96 is a high-dimensional continuous-time dynamical system capable of exhibiting complex dynamics. It consists of N interacting variables arranged on a one-dimensional periodic lattice, with each variable coupled to its nearest neighbors. The system dynamics are described by the Lorenz-96 equation

$$\dot{x}_i(t) = \Big(x_{i+1}(t) - x_{i-2}(t)\Big)x_{i-1}(t) - x_i(t) + F, \qquad i = 1, \ldots, N, \tag{4.15}$$

with periodic boundary conditions $x_{i+N} = x_i$. Here $x_i(t)$ denotes the state of variable i, and F is a constant forcing parameter controlling the degree of chaotic behavior. The maximal Lyapunov exponent (MLE) of the L96 system varies with system dimension, as shown in Figure 4.4. For the experiments in this thesis, a system dimension exhibiting chaotic dynamics was chosen ($N = 40$). L96 provides a moderately challenging benchmark: its coupling structure is simple and periodic (Figure 4.3), yet the system dynamics are complex. Previous work by Baur has demonstrated that the generalized local states approach can accurately predict the system evolution when the underlying network structure is known [2].

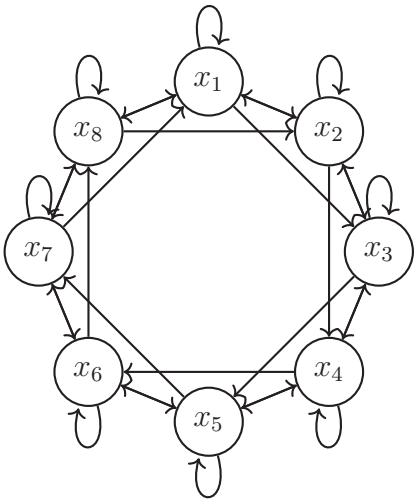


Figure 4.3: Summary graph of a 10-D Lorenz-96 system

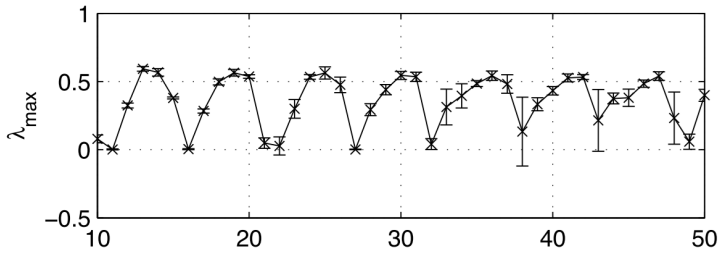


Figure 4.4: Maximal Lyapunov exponent λ_{max} for the Lorenz 96 system with different system sizes N and $F = 5$. [45]

4.5 Kuramoto oscillator networks

The primary datasets used in this thesis are oscillator networks. One of the most studied type is the Kuramoto system. [46]

On a graph with adjacency matrix A, the standard Kuramoto model is defined as

$$\dot{\theta}_i(t) = \omega_i + K \sum_{j=1}^{N} A_{ij} \sin\Big(\theta_j(t) - \theta_i(t)\Big), \qquad i = 1, \ldots, N, \tag{4.16}$$

where $\theta_i(t)$ is the phase of oscillator i, ω_i is its natural frequency, and K is the global coupling strength [46].

The degree of synchronization is quantified by the complex order parameter

$$re^{i\psi}(t) = \frac{1}{N} \sum_{j=1}^{N} e^{i\theta_j(t)}, \tag{4.17}$$

where $r(t) \in [0, 1]$ measures the phase coherence and $\psi(t)$ denotes the average phase. A fully incoherent system has $r \approx 0$, whereas $r \approx 1$ indicates complete synchronization [47].

Depending on the coupling structure, the behavior of a Kuramoto system can vary significantly. In this thesis, one specific network topology studied is the frequency-assortative

Kuramoto network, in which oscillators are more likely to connect if their natural frequencies are similar. Frequency-assortative networks have been shown to produce chaotic dynamics for certain parameter choices [48]..

A frequency-assortative network of size N is constructed as follows. Each oscillator is assigned a natural frequency ω_i drawn from a uniform distribution, and initially, all nodes are unconnected. A random node i that has not yet reached its maximum degree d is selected, followed by a second random node j that is neither connected to i nor at the maximum degree. Then with a probability p_{ij}, the two nodes are linked. This probability is calculated according to the natural frequencies of i and j:

$$p_{ij} \propto \frac{\delta^\gamma}{\delta^\gamma + |\omega_i - \omega_j|^\gamma}, \tag{4.18}$$

where δ and γ are tunable parameters controlling the frequency assortativity. [48][5]

4.6 UK power grid

The most complex benchmark considered in this thesis is a model of the UK power grid, featuring 120 nodes and 165 undirected edges. To describe the system dynamics, we consider the higher-order Kuramoto model defined as

$$\dot{\theta}_i(t) = \omega_i + \gamma_1 \sum_{j=1}^{N} A_{ij} \sin\Big(\theta_j(t) - \theta_i(t)\Big) + \gamma_2 \sum_{j=1}^{N}\sum_{k=1}^{N} B_{ijk} \sin\Big(\theta_j(t) + \theta_k(t) - 2\theta_i(t)\Big), \quad i = 1, \ldots, N, \tag{4.19}$$

where $\theta_i(t)$ and ω_i denote the phase and natural frequency of oscillator i. γ_1 and γ_2 control pairwise and three-node interactions, respectively, and B_{ijk} encodes higher-order couplings.

Two different data formats for the UK power grid are used throughout the experiments. One setup uses the raw data (raw angles θ). While the other follows Li et al., who in their analysis of the UKPG preprocessed the data by applying $(\sin(\theta), \cos(\theta))$ [4]. For the causal inference tasks, only the dimensions corresponding to $\sin(\theta)$ are used as the causal matrices should have the same dimensions as the original system.

For appropriate parameters, this system exhibits highly complex trajectories. The network and some sample trajectories are shown in Figure 4.5

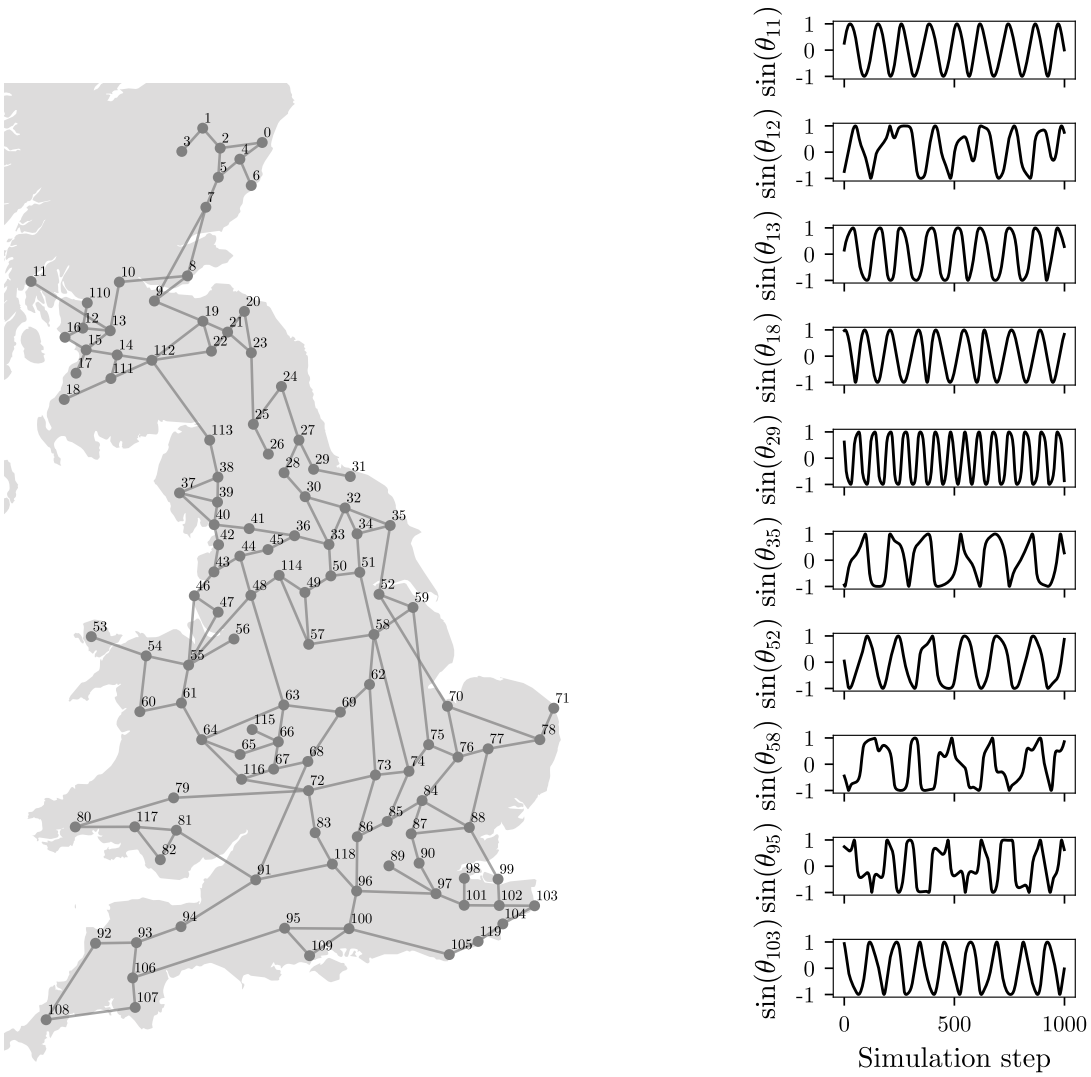


Figure 4.5: Overview over the UK power grid dataset. The left plot shows the network structure consisting of 120 nodes and 165 undirected edges. The right plot shows a few random sample trajectories. The geodata is taken from [49], whereas the locations are geocoded with QGIS from a reference image in [4].

CHAPTER 5
PREDICTION METHODS

Prediction forms the second core component of causal local states (CLS). It plays a central role at multiple stages of the methodology. In the final step of the algorithm, inferred causal information is utilized to generate predictions of the full system dynamics. In addition, prediction is already required during the causal inference stage itself, where a wrapper-based feature selection strategy is employed to evaluate and compare candidate feature subsets based on their predictive performance.

Accordingly, this chapter introduces all relevant aspects of generating predictions within the CLS framework. It first introduces the measures used to evaluate predictive performance. These evaluation criteria are essential both for the wrapper-based feature selection step and for assessing the overall performance of the CLS framework. Next, the chapter describes how predictions are generated across the different components of CLS. This includes the models employed as wrappers, reservoir computing (RC) and next generation reservoir computing (NGRC), as well as the framework used to forecast the full system, Local States (LS).

5.1 Prediction quality measures

To evaluate the accuracy of predictions for dynamical systems, two primary metrics are employed: the mean-squared error (MSE) and the valid prediction steps (VPS). Both measures capture short-term predictive performance.

5.1.1 Mean-squared error

The mean-squared error (MSE) quantifies the average deviation of predictions from the true trajectory over a given interval. Let $\mathbf{X}_t \in \mathbb{R}^d$ denote the true system state at time t and $\tilde{\mathbf{X}}_t \in \mathbb{R}^d$ the corresponding predicted state. Then, the mean-squared error (MSE) over a prediction horizon of length N_{pred} starting at time t_0 is defined as

$$\text{MSE} = \frac{1}{N_{\text{pred}}} \sum_{k=1}^{N_{\text{pred}}} \left\| \tilde{\mathbf{X}}_{t_0+k} - \mathbf{X}_{t_0+k} \right\|_2^2 . \tag{5.1}$$

5.1.2 Valid prediction steps

The valid prediction steps (VPS) quantify the number of discrete time steps over which a predicted trajectory remains sufficiently close to the true trajectory. Divergence is measured using the Euclidean distance between the predicted and measured states.

Let $\mathbf{X}_t \in \mathbb{R}^d$ denote the true system state at time step t and $\tilde{\mathbf{X}}_t \in \mathbb{R}^d$ the corresponding predicted state. The divergence at time step t is defined as

$$d(t) = \left\| \tilde{\mathbf{X}}_t - \mathbf{X}_t \right\|_2 . \tag{5.2}$$

Given a threshold $\epsilon > 0$, the valid prediction steps k_v are defined as the first time step at which the prediction error exceeds this threshold:

$$k_v = \min \left\{ k \in \{1, \ldots, N_{\text{pred}}\} \;\middle|\; \left\| \tilde{\mathbf{X}}_{t_0+k} - \mathbf{X}_{t_0+k} \right\|_2 > \epsilon \right\} . \tag{5.3}$$

If the prediction error remains below the threshold for the entire prediction horizon, the VPS is set to N_{pred}. The standard value used throughout the thesis is $\epsilon = 0.4$.

5.2 Reservoir computing

Reservoir computing (RC) is a machine learning framework that has shown strong performance in the forecasting of complex dynamical systems [50]. In this thesis, we focus on a specific class known as echo state networks (ESNs) [51], which will be used synonymously with RC. ESNs employ a recurrent neural network as the reservoir, providing a high-dimensional nonlinear transformation of the input signal. Crucially, the reservoir dynamics remain fixed during training, and only the readout layer is trained[52]. This significantly reduces training complexity, as the optimization problem is restricted to a linear regression in the output layer. For a more thorough introduction to RC, the reader is referred to [52] and [53].

An ESN consists of a reservoir of N recurrently connected nodes defined by an adjacency matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$. For this, sparse random networks with Erdős-Rényi connectivity have been shown to perform robustly in practice and are therefore adopted in this thesis [54]. After initialization, the weights of the reservoir are scaled such that the resulting adjacency matrix has a fixed spectral radius ρ. The spectral radius constitutes an important hyperparameter of the model.

The Input data $\boldsymbol{X} \in \mathbb{R}^d$ is coupled to the reservoir with an input matrix $\mathbf{W}_{\text{in}} \in \mathbb{R}^{N \times d}$. Following Herteux and Räth, the input is chosen to be sparse, with one nonzero element per row. Each nonzero entry is drawn uniformly from $[-1, 1]$ and rescaled with a factor $W_{\text{in,scale}}$

The internal reservoir state $\boldsymbol{r}_t \in \mathbb{R}^N$ is updated at each time step according to

$$\boldsymbol{r}_{t+1} = f(\mathbf{A}\boldsymbol{r}_t + \mathbf{W}_{\text{in}}\boldsymbol{X}_t) , \tag{5.4}$$

where $f(\cdot)$ is a nonlinear activation function applied element-wise. In this thesis, the tanh(.) will be used as the activation function.

To generate predictions with RC, a feature vector $\mathbb{O}_t$ is constructed from the reservoir state. In the most common case, the reservoir state itself is used as the feature vector: $\mathbb{O}_t = \boldsymbol{r}_t$.

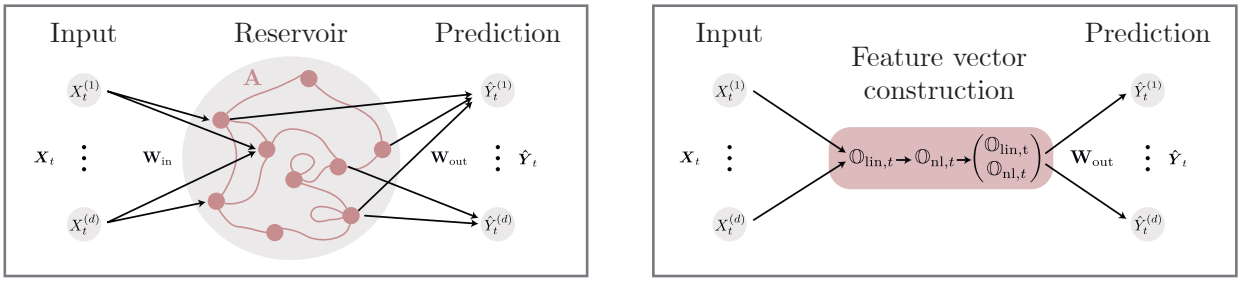


Figure 5.1: Architecture of traditional reservoir computing (left) and next generation reservoir computing (right)

In some experiments in this thesis, a quadratic readout is used, where the feature vector is augmented with element-wise squared terms of the reservoir state: $\mathbb{O}_t = (\boldsymbol{r}_t, \boldsymbol{r}_t^{\circ 2})$ [1].

Based on this feature vector, the predicted output is obtained via a linear coupling:

$$\tilde{\boldsymbol{Y}}_t = \mathbf{W}_{\text{out}} \mathbb{O}_t, \tag{5.5}$$

where $\mathbf{W}_{\text{out}} \in \mathbb{R}^{d \times N_{\text{feat}}}$ denotes the output weight matrix.

After initialization, the training phase of the reservoir computing model begins. During this phase, the reservoir is driven by the input sequence, while the internal weights $\mathbf{A}$ and $\mathbf{W}_{\text{in}}$ remain fixed. Only the readout matrix $\mathbf{W}_{\text{out}}$ is trained by minimizing a regularized least-squares loss,

$$\min_{\mathbf{W}_{\text{out}}} \sum_{t \in \mathcal{T}} \left\| \mathbf{W}_{\text{out}} \mathbb{O}_t - \boldsymbol{x}_t \right\|^2 + \beta \left\| \mathbf{W}_{\text{out}} \right\|_F^2, \tag{5.6}$$

where $\mathcal{T}$ denotes the set of training time indices and $\beta > 0$ is the Ridge regularization parameter.

To obtain reliable predictions, the reservoir must first be synchronized with the training data such that the influence of the initial reservoir state is negligible. This is achieved by applying a warm-up (or synchronization) phase of length N_{sync}, during which the reservoir is driven by the input sequence without recording outputs. The predicted values generated during this phase are discarded. If the prediction interval directly follows the training interval, an explicit resynchronization is typically unnecessary, as the reservoir state is already aligned with the final training state.

During the prediction phase, the model operates in a closed loop, where the predicted output is fed back as input to the reservoir. The dynamics assuming a linear readout of $\mathbb{O}_{t+1} = \boldsymbol{r}_{t+1}$ are given by

$$\tilde{\boldsymbol{Y}}_t = \mathbf{W}_{\text{out}} \mathbb{O}_t, \tag{5.7}$$

$$\boldsymbol{r}_{t+1} = f\left(\mathbf{A} \boldsymbol{r}_t + \mathbf{W}_{\text{in}} \tilde{\boldsymbol{Y}}_t \right) \tag{5.8}$$

This recursive prediction scheme can, in principle, be iterated for an arbitrary number of time steps. However, in practice, the prediction accuracy typically degrades over time due to the accumulation of errors. A schematic of the architecture of RC is shown in Figure 5.1.

5.3 Next generation reservoir computing

Next generation reservoir computing (NGRC) introduced by Gauthier et al. is an alternative to classical reservoir computing [55]. In contrast to classical reservoir computing, NGRC does not have a randomized reservoir. Instead, the feature vector is constructed explicitly from the input data using time-delayed coordinates and nonlinear transformations. In this thesis, NGRC serves as the primary predictive model.

The development of NGRC was motivated by results demonstrating the equivalence of reservoir computing and nonlinear vector autoregression (NVAR) [56][1]. Gauthier et al. show that an RC-like model based on NVAR performs strongly on standard RC benchmarks while requiring less training data and fewer hyperparameters [55].

Let $\mathbf{X}_t \in \mathbb{R}^d$ denote the input at discrete time t. At each time step, NGRC constructs the feature vector $\mathbb{O}_t$ by concatenating a constant bias term, a linear feature vector, and a nonlinear feature vector:

$$\mathbb{O}_t = c \oplus \mathbb{O}_{\text{lin},t} \oplus \mathbb{O}_{\text{nl},t}, \tag{5.9}$$

where $\oplus$ denotes vector concatenation. As in classical RC, predictions are obtained via a linear coupling,

$$\tilde{\mathbf{Y}}_t = \mathbf{W}_{\text{out}} \mathbb{O}_t, \tag{5.10}$$

where $\mathbf{W}_{\text{out}}$ is the only trainable part of the model.

Linear feature vector: The linear feature vector is constructed using a time-delay embedding of the input:

$$\mathbb{O}_{\text{lin},t} = \bigoplus_{i=1}^{k} \mathbf{X}_{t-(i-1)s}, \tag{5.11}$$

where k denotes the number of delays and s the delay spacing. Both parameters are hyperparameters of the model. Since $\mathbf{X}_t \in \mathbb{R}^d$, it follows that $\mathbb{O}_{\text{lin},t} \in \mathbb{R}^{dk}$.

Nonlinear feature vector: Nonlinearity in NGRC is introduced through polynomial expansions of the linear feature vector. All monomials of degree p can be generated via the outer product,

$$\bigotimes_{i=1}^{p} \mathbb{O}_{\text{lin},t}, \tag{5.12}$$

which is a rank-p tensor containing all degree-p monomials, including redundant permutations. To construct the feature vector, it is desirable to remove those redundancies, thereby reducing the feature space dimension. Following Gauthier et al., an operator $\lceil \otimes \rceil$ that collects the unique monomials of this tensor into a vector is defined [55].

The component of the nonlinear feature vector for order p is then defined as

$$\mathbb{O}^{(p)}_{\text{nonlin},t} = \left\lceil \bigotimes_{i=1}^{p} \mathbb{O}_{\text{lin},t} \right\rceil. \tag{5.13}$$

[1] For linear activation and quadratic readout

The full nonlinear feature vector is obtained by concatenating contributions from a set of polynomial orders $\mathcal{P}$:

$$\mathbb{O}_{\mathrm{nl},t} = \bigoplus_{p\in\mathcal{P}} \mathbb{O}_{\mathrm{nl},t}^{(p)}. \tag{5.14}$$

The set $\mathcal{P}$ constitutes an additional hyperparameter of the NGRC model.

5.3.1 Feature space size

NGRC shares conceptual similarities with statistical forecasting approaches such as nonlinear vector autoregression (NVAR) [57]. A key distinction, however, lies in the dimensionality of the feature space: NGRC typically employs substantially larger feature vectors due to the explicit construction of nonlinear terms [58]. The size of the feature vector grows rapidly with both the number of delays k and the polynomial orders included in $\mathcal{P}$.

To determine the size of the nonlinear feature space, consider a fixed polynomial order p. Given a state of dimension d, the linear feature vector $\mathbb{O}_{lin}$ has dk components. To create a feature vector entry of order p, p monomials are drawn from the pool of dk components. As each monomial can be "drawn" multiple times, this can be thought of as a sampling problem with replacement and without order. The number of ways one can sample n draws from a set of N components without order but with replacement is given by

$$|\Omega| = \binom{n+N-1}{n}, \tag{5.15}$$

where Ω is the sample space [59]. Thus, for p monomials drawn from dk linear components, this part of the feature vector has the magnitude

$$|\mathbb{O}_{\mathrm{nl},t}^{(p)}| = \binom{p+dk-1}{p}. \tag{5.16}$$

The full nonlinear feature vector is obtained by summing over all polynomial orders in $\mathcal{P}$, yielding

$$|\mathbb{O}_{\mathrm{nl},t}| = \sum_{p\in\mathcal{P}} |\mathbb{O}_{\mathrm{nl},t}^{(p)}| = \sum_{p\in\mathcal{P}} \binom{p+dk-1}{p}. \tag{5.17}$$

This combinatorial growth leads to a rapid increase in feature dimensionality as either k or the maximum polynomial order is increased. Consequently, even moderate parameter choices can result in very high-dimensional feature spaces.

From a practical perspective, this has significant computational implications. Assuming double-precision floating point representation (`float64`), each feature requires 8 bytes of memory [60]. As a result, both memory consumption and computational cost can quickly become prohibitive during training and inference. This is further amplified by the construction of the design matrix during training, which for a dataset of length T has dimension $T \times |\mathbb{O}_t|$. Figure 5.2 illustrates the growth of the feature space size and the corresponding memory requirements for different parameter configurations.

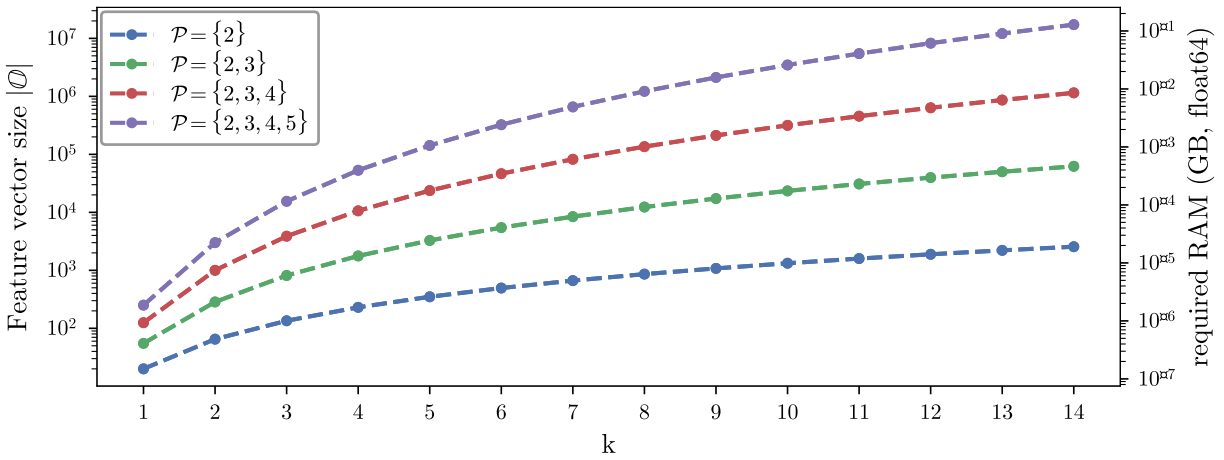


Figure 5.2: Size and memory requirements of a single NGRC feature vector assuming `float64` precision.

5.3.2 Inference mode

In addition to standard forecasting, NGRC supports an inference mode, in which the model learns a mapping between subsets of system variables. This capability is especially useful in scenarios where only partial observations are available during deployment while full system information is accessible during training.

Let $\mathbf{X}_t \in \mathbb{R}^q$ denote the input variables and $\mathbf{Y}_t \in \mathbb{R}^p$ the target variables. The model is trained to approximate the mapping

$$\mathbf{X}_t \mapsto \mathbf{Y}_t. \tag{5.18}$$

In this setting, the model is not operated in a closed loop. Instead, predictions at each time step depend only on the current input and its delayed coordinates, allowing each time step to be evaluated independently. This makes inference mode computationally efficient, particularly when a large number of one-step predictions is required, as they can be generated in parallel.

5.4 (Generalized) local states

The prediction of the full system in CLS builds upon local states (LS) by Pathak et al. [1] and generalized local states (GLS) by Baur et al. [2]. LS allows a parallel prediction of a system by fragmenting the system into multiple subsystems. Each of the fragments is called a neighborhood, which is a collection of nodes around a number of core nodes. For each neighborhood, a separate predictive model is trained using only the information available in the neighborhood. Each model generates forecasts for its respective core variables, and the individual predictions are concatenated to form a global prediction of the system state. In this thesis, we restrict attention to the case of a single-node core. The procedure is applied iteratively such that each system component serves as the core once.

Let $\mathbf{X}_t = (X_t^1, \ldots, X_t^d) \in \mathbb{R}^d$ denote the system state at time t. Assume that the interactions in the system can be represented by a summary graph $G = (V, E)$. For a core node X^i, the corresponding neighborhood is defined as (Eq. 2.9)

$$\mathcal{N}(X^i) = \Big(\{X^i\} \cup \mathrm{Pa}(X^i), \{X^k \to X^i \ : X^k \in \mathrm{Pa}(X^i)\}\Big),$$

where $\mathrm{Pa}(X^i)$ denotes the set of parent nodes of X^i in G.

Let V_i denote the vertex set of $\mathcal{N}(X^i)$ with $|V_i| = d_i$. Then we can define a projection operator $\pi_i : \mathbb{R}^d \to \mathbb{R}^{d_i}$ to be

$$\pi_i(\mathbf{X}) \ \coloneqq \ \Big(X^j : X^j \in V_i\Big). \tag{5.19}$$

Given the projection operator π_i, we define the neighborhood-restricted input at time t as

$$\mathbf{X}_t^{(i)} \ \coloneqq \ \pi_i(\mathbf{X}_t) \in \mathbb{R}^{d_i}. \tag{5.20}$$

We then train any suitable predictive model

$$g_i : \mathbb{R}^{d_i} \to \mathbb{R} \tag{5.21}$$

to predict the value of the core node X_t^i based solely on its neighborhood.

The individual prediction models $\{g_i\}_{i=1}^d$ can be combined into a global predictor of the full system state,

$$\mathbf{g}(\mathbf{X}_t) \ \coloneqq \ \Big(g_1(\mathbf{X}_t^{(1)}), \ldots, g_d(\mathbf{X}_t^{(d)})\Big) \ \in \ \mathbb{R}^d. \tag{5.22}$$

This allows for a closed-loop prediction via

$$\tilde{\mathbf{X}}_{t+1} = \mathbf{g}(\mathbf{X}_t), \tag{5.23}$$

where the predicted state is fed back as input to generate multi-step forecasts.

To efficiently store all neighborhoods, the neighborhood matrix representation introduced by Baur et al. is suitable [2]. The neighborhood matrix $A \in \{0, 1, 2\}^{m \times d}$ represents a collection of neighborhoods, where m denotes the number of neighborhoods. Each entry A_{ij} takes one of three possible values:

$$A_{ij} \ = \ \begin{cases} 2, & \text{if } j = i \quad \text{(core node)}, \\ 1, & \text{if } X^j \in \mathrm{Pa}(X^i) \quad \text{(directed edge } X^j \to X^i), \\ 0, & \text{otherwise.} \end{cases} \tag{5.24}$$

Applying the local states idea has clear advantages over a single model with one big feature vector. Assuming there is an underlying network governing the interaction structure, only a subset of dimensions are actually useful for prediction. Furthermore, LS provides a way to incorporate interaction knowledge to improve predictive performance, which is of central importance for an algorithm that tries to directly connect network inference with prediction.

CHAPTER 6

CAUSAL LOCAL STATES

This section introduces the causal local states (CLS) framework, which constitutes the central contribution of this thesis. The design choices underlying CLS are motivated by the experiments presented in the subsequent Chapter 7.

CLS is designed with two key challenges in mind. First, forecasting high dimensional systems requires models that remain computationally efficient despite the high dimensionality. Second, beyond predictive performance, it is desirable to obtain interpretable insights into the underlying interaction structure of the system. CLS addresses both objectives by combining efficient prediction with selected ideas from causal discovery.

The framework builds upon the idea of decomposing a system into core nodes and their associated neighborhoods, as introduced in Section 2.2. In contrast to local states (LS) [1], CLS does not assume that the neighborhoods are known a priori. Instead, the neighborhoods are inferred directly from data, making CLS applicable to systems with unknown and potentially complex interaction structures.

If one adapts Granger's view of causality, the task of identifying suitable neighborhoods can be interpreted as a variation of a feature selection problem. Accordingly, terminology and concepts commonly used in this field are adopted. In particular, two core paradigms are used: filter methods and wrapper methods. A filter method relies on a scoring function to rank candidate variables according to their relevance, independently of any predictive model [61]. In contrast, a wrapper method evaluates candidate feature subsets by training a predictive model, which is treated as a black box, and assessing its performance on the given subset [62].

The algorithm works in multiple steps. Steps 1-3 have to be repeated for every core node:

1. **Filter Step (Variable Ranking)**:

   (a) Sorting nodes with a bivariate causal measure.

   (b) Construction of neighborhood candidates of increasing size based on the ranked variables.

2. **Wrapper Neighborhood Selection**:
   (a) Evaluation of neighborhood candidates using a wrapper model.
   (b) Selection of the optimal neighborhood according to predictive performance.
3. **Backward Feature Elimination**:
   (a) Recursive removal of redundant or spurious features from the selected neighborhood.
   (b) Upon convergence, the remaining variables define the approximate causal neighborhood of the respective core node.
4. **Full System Prediction**:
   (a) Individual neighborhoods are combined into a neighborhood matrix.
   (b) The total system is predicted with a local states approach.

6.1 Filter step: variable ranking

For a d-dimensional system, the number of possible neighborhoods around a single core node grows exponentially. Each of the $d-1$ candidate variables can either be included or excluded, resulting in a total of 2^{d-1} possible neighborhood configurations for a fixed core node. For example, in the model system of the UK power grid, this would yield $2^{119} \approx 6.6 \cdot 10^{35}$ possible neighborhoods. Consequently, for high-dimensional systems, exhaustive search over all candidates is computationally infeasible.

To address this issue, the CLS framework introduces a filter step that drastically reduces the search space of neighborhood candidates. The key idea is to rank candidate variables using a scoring function and to restrict the search to the most promising candidates. This approach corresponds to a variable ranking method in the feature selection literature [61].

Consider a dataset generated by a dynamical system with an underlying interaction structure represented by a summary graph $G = (V, E)$. For each pair of variables, a bivariate causal measure is used as a scoring function. In contrast to purely statistical measures such as Pearson correlation or mutual information, causal measures aim to capture directed influence between variables. According to the causal Markov condition 4, knowing the causal parents of a node allows for optimal reconstruction of the node's trajectory.

Let $\{X_t\}_{t\in\mathbb{Z}}$ be a d-dimensional time series, where $X_t = (X_t^1, \ldots, X_t^d)$. For each $i \in \{1, \ldots, d\}$, let $X^i = \{X_t^i\}_{t\in\mathbb{Z}}$ denote the univariate time series associated with node i.

Importantly, the neighborhood inference process is completely independent for each core node. Let i denote the index of the core node and $j \neq i$ a neighbor candidate. The score of X^i relative to the core node X^j is defined as

$$C_{j\to i} := C(X^j \to X^i), \tag{6.1}$$

where $C(.)$ denotes a bivariate causal measure quantifying the directed influence of X^j on X^i.

Algorithm 2 Filter Step: Variable Ranking for Core Node X^i

Input: Multivariate time series $\mathbf{X} = (X^1, \ldots, X^d)$, Core node index i, Bivariate causal measure $C(\cdot \to \cdot)$, Maximum neighborhood size $d_{\max}$
Output: Ordered candidate set $\mathcal{C}_i = \{\mathcal{N}_k\}_{k=0}^{d_{\max}}$

1: **for** $j = 1$ to d, $j \neq i$ **do**
2: Compute score $C_{j\to i} := C(X^j \to X^i)$
3: Determine permutation $(1), \ldots, (d-1)$ such that

$$C_{(1)\to i} \geq C_{(2)\to i} \geq \cdots \geq C_{(d-1)\to i}$$

4: **for** $k = 1$ to $d_{\max}$ **do**
5: Construct neighborhood:

$$\mathcal{N}_k \leftarrow \left(\{X^i\} \cup \{X^{(1)}, \ldots, X^{(k)}\},\ \{(X^{(\ell)}, X^i)\}_{\ell=1}^{k}\right)$$

6: Create candidate set:

$$\mathcal{C}_i \leftarrow \{\mathcal{N}_k \mid k = 0, 1, \ldots, d_{\max}\}$$

7: **return** $\mathcal{C}_i$

The nodes are now reindexed such that

$$C_{(1)\to i} \geq C_{(2)\to i} \geq \cdots \geq C_{(N-1)\to i}, \tag{6.2}$$

with $C_{(k)\to i}$ denoting the k-th biggest causal score.

From this ranked list, neighborhood candidates can be defined. Starting from a base neighborhood $\mathcal{N}_0(X^i) = (\{X^i\}, \varnothing)$[1], candidate nodes are added incrementally by descending score, yielding a nested sequence of candidate neighborhoods. With $X^{(k)}$ as the node with the k-th highest causal score, the k-th neighborhood is defined as:

$$\mathcal{N}_k(X^i) = \mathcal{N}_k := \left(\{X^i\} \cup \{X^{(1)}, \ldots, X^{(k)}\},\ \{(X^{(\ell)},\ X^i)\}_{\ell=1}^{k}\right), \tag{6.3}$$

where k is an index that fulfills $1 \leq k \leq d_{\max}$. $d_{\max}$ is one hyperparameter introduced in this framework, which limits the maximum neighborhood size considered. The neighborhoods created from the ranked candidate variables constitute the main deliverable of the filter step in CLS:

$$\mathcal{C}_i := \{\mathcal{N}_k \ \mid \ k = 0, 1, \ldots, d_{\max}\}, \tag{6.4}$$

where i is the index of the core node.

This filter step reduces the search space size from $\mathcal{O}(2^d)$ to a sequence of at most $d_{\max} - 1$ candidates per core node, while favoring those variables that exhibit the strongest pairwise causal score with the core. The pseudocode implementation is provided in Algorithm 2. The next step is to evaluate the neighborhood candidates and pick the optimal one.

[1] Some wrappers need at least one feature besides the core node. Then $\mathcal{N}_0$ has to include $X^{(1)}$

6.2 Wrapper neighborhood selection

Following the filter step, which yields a ranked set of candidate neighborhoods $\mathcal{C}(X^i)$ for each core node X^i, the next task is to select the optimal neighborhood from this set. This is achieved using a wrapper-based evaluation strategy, originally introduced by Kohavi and John [62].

A wrapper is a black box predictive model that evaluates the quality of a feature set by training on it and assessing its predictive performance [61]. In the context of CLS, each neighborhood candidate $\mathcal{N}_k(X^i) \in \mathcal{C}(X^i)$ is interpreted as a feature subset whose relevance is determined through prediction of the core node.

To avoid bias in the final model evaluation, the available training data is further divided into a wrapper training set and a wrapper test set. The wrapper models are trained on the former and evaluated on the latter.

Let $\mathcal{N}_k$ be the neighborhood subgraph that needs to be evaluated. To evaluate this neighborhood with a wrapper model, the feature vector is first defined as

$$\mathbf{z}_k \coloneqq \left(X^j\right)_{X^j \in V(\mathcal{N}_k)}. \tag{6.5}$$

Depending on the chosen wrapper model, the core node X^i may be excluded from the feature vector (e.g., when using inference mode in NGRC).

For each candidate neighborhood, a predictive model $g_k : \mathbb{R}^{d_k} \to \mathbb{R}$ is trained to predict the next state of the core node based solely on its neighborhood,

$$\tilde{X}^i_{k,t+1} = g_k(\mathbf{z}_{k,t}), \tag{6.6}$$

where $d_k = |V_k|$ denotes the dimensionality of the feature vector.

The predictive performance of each wrapper model is evaluated on the wrapper test set using a loss function $\mathcal{L}$. Let $\mathcal{I}^{\text{w-test}}$ denote the corresponding set of time indices. The loss associated with the k-th neighborhood candidate is defined as

$$\mathcal{L}_k = \mathcal{L}\Big(\{\tilde{X}^i_{k,t}\}_{t\in\mathcal{I}^{\text{w-test}}},\ \{X^i_t\}_{t\in\mathcal{I}^{\text{w-test}}}\Big). \tag{6.7}$$

Once all neighborhood candidates in $\mathcal{C}(X^i)$ have been evaluated, the optimal candidate is selected using a relative improvement threshold $\alpha_1 \in (0, 1)$. The goal is to identify the smallest neighborhood that achieves near-optimal predictive performance.

$$\Delta(k, k') = \frac{\mathcal{L}_{k'} - \mathcal{L}_k}{\mathcal{L}_{k'}}. \tag{6.8}$$

A new candidate $\mathcal{N}_k$ is accepted as the updated baseline if it achieves a sufficient relative improvement, i.e.,

$$\Delta(k, k') \geq \alpha_1. \tag{6.9}$$

This procedure is applied sequentially across all candidates with increasing neighborhood size, always comparing the current candidate to the most recently accepted baseline.

The resulting neighborhood size k^* corresponds to the smallest candidate for which no subsequent neighborhood yields a significant improvement. A pseudocode implementation of this selection procedure is provided in Algorithm 3.

Algorithm 3 Neighborhood Selection by Relative Improvement

Input: Losses $\{\mathcal{L}_k\}_{k=0}^{d_{\max}}$, threshold α_1
Output: Optimal neighborhood size k^*

$k^* \leftarrow 0$ ▷ For wrappers that need a minimum size, one has to start with $k^* = 1$
for $k = 1$ to $d_{\max}$ **do**
    Compute relative improvement:

$$\Delta(k, k^*) = \frac{\mathcal{L}_{k^*} - \mathcal{L}_k}{\mathcal{L}_{k^*}}$$

    if $\Delta(k, k^*) \geq \alpha_1$ **then**
        $\mathcal{L}_{k^*} \leftarrow \mathcal{L}_k$
        $k^* \leftarrow k$
return k^*

6.3 Backward feature elimination

The neighborhood $\mathcal{N}_{k^*}$ obtained from the wrapper selection step may still contain redundant or non-informative variables. To maximize interpretability, compute speed, and prediction quality, it is desirable to remove those from the neighborhood. Consequently, the next part of the algorithm is a backward feature removal.

Starting from the selected neighborhood $\mathcal{N}_{k^*}(X^i)$, candidate variables are iteratively removed and the resulting neighborhoods are re-evaluated. The underlying assumption is that removing a truly relevant (causal) variable leads to a significant degradation in predictive performance, whereas removing an irrelevant variable results in only a minor or no change. This is motivated by the causal Markov condition (Def. 4), which ensures that the $\mathrm{ND}(X^i)$ are conditionally independent of the core node X^i, provided the causal parents are included in the neighborhood.

As this is an iterative procedure, the index k^* will be omitted. For each node $X^j \in \mathcal{N}$ with $j \neq i$, the reduced neighborhood is defined as

$$\mathcal{N}^{(-j)} = \Big(V \setminus \{X^j\},\ E \setminus \{(X^j, X^i)\}\Big), \tag{6.10}$$

where V, E are the vertex and edge sets of $\mathcal{N}$. Each reduced neighborhood is subsequently evaluated using the same wrapper approach. This yields a set of losses:

$$\left\{\mathcal{L}_{-j} \mid X^j \in \mathcal{N}\right\}. \tag{6.11}$$

Among all candidates, only the variable whose removal results in the smallest loss is considered for elimination. While this may be counterintuitive, a small loss corresponds to a small decrease in performance, which is evidence that the node does not contribute to prediction performance. Let X^m be such that $m = \mathrm{argmin}_j\ \mathcal{L}_{-j}$. X^m is removed only if the relative deterioration in performance does not exceed a predefined threshold $\alpha_2 \in (0, 1)$:

$$\frac{\mathcal{L}_{-m} - \mathcal{L}_0}{\mathcal{L}_0} \leq \alpha_2, \tag{6.12}$$

where $\mathcal{L}_0$ is the baseline calculated with an unreduced neighborhood.

Algorithm 4 Backward Feature Removal for a Neighborhood $\mathcal{N}$

Input: Initial neighborhood $\mathcal{N}$, loss function $\mathcal{L}(\cdot)$, relative threshold $\alpha_2 \in (0, 1)$
Output: Reduced neighborhood $\mathcal{N}^*$

Compute baseline loss $\mathcal{L}_0 \leftarrow \mathcal{L}(\mathcal{N})$
Initialize $\mathcal{N}^* \leftarrow \mathcal{N}$
repeat
  for all $X^j \in V(\mathcal{N})$ with $j \neq i$ **do**
    $\mathcal{N}^{(-j)} \leftarrow \Big(V \setminus \{X^j\},\ E \setminus \{(X^j, X^i)\}\Big)$
    $\mathcal{L}_{-j} \leftarrow \mathcal{L}(\mathcal{N}^{(-j)})$
  $m \leftarrow \arg\min_j (\mathcal{L}_{-j})$
  if $\dfrac{\mathcal{L}_{-m} - \mathcal{L}_0}{\mathcal{L}_0} \leq \alpha_2$ **then**
    $\mathcal{N}^* \leftarrow \mathcal{N}^{(-m)}$
    $\mathcal{L}_0 \leftarrow \mathcal{L}_{-m}$
  else
    break
until no candidate X^j satisfies the removal criterion
return $\mathcal{N}^*$

If this condition is satisfied, the neighborhood is updated as $\mathcal{N}(X^i) \leftarrow \mathcal{N}^{(-m)}(X^i)$, and the removal is repeated.

This iterative elimination continues until convergence, with the final resulting neighborhood representing the approximate causal neighborhood for the core node X^i.

A pseudocode implementation of this procedure is provided in Algorithm 4. The full neighborhood inference process is illustrated in Figure 6.1.

6.4 Predicting the full system with LS

The final step of the CLS framework is to leverage the inferred causal structure for forecasting the full system dynamics. By repeating the neighborhood inference procedure (Steps 1-3) for each variable X^i, a collection of estimated causal neighborhoods $\{\mathcal{N}^*(X^i)\}_{i=1}^d$ is obtained, which together approximate the underlying interaction structure of the system. To store all neighborhoods in a compact manner, they are combined in a neighborhood matrix.

$$A_{ij} = \begin{cases} 2, & \text{if } j = i, \\ 1, & \text{if } X^j \in V(\mathcal{N}^*(X^i)), \\ 0, & \text{otherwise.} \end{cases} \tag{6.13}$$

This neighborhood matrix is now used to forecast the full system with a local states approach as described in section 5.4. A schematic of the prediction step in CLS is shown in Figure 6.2

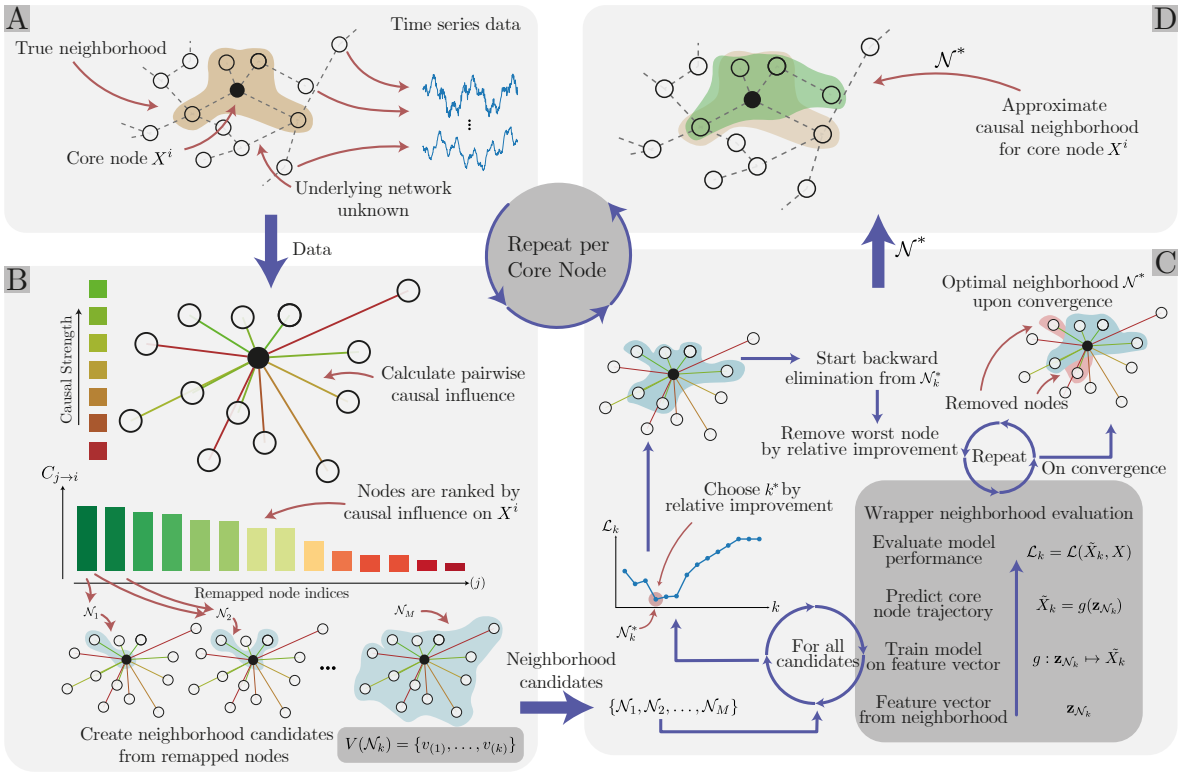


Figure 6.1: Schematic of the causal neighborhood inference process in CLS. **(A)** The input data is time series data with an unknown underlying network structure. **(B)** Sorting of the nodes by their bivariate causal strength and creation of neighborhood candidates. **(C)** Wrapper-based inference for approximate causal neighborhood. **(D)** Inference of an approximate causal neighborhood is the goal of this step in CLS.

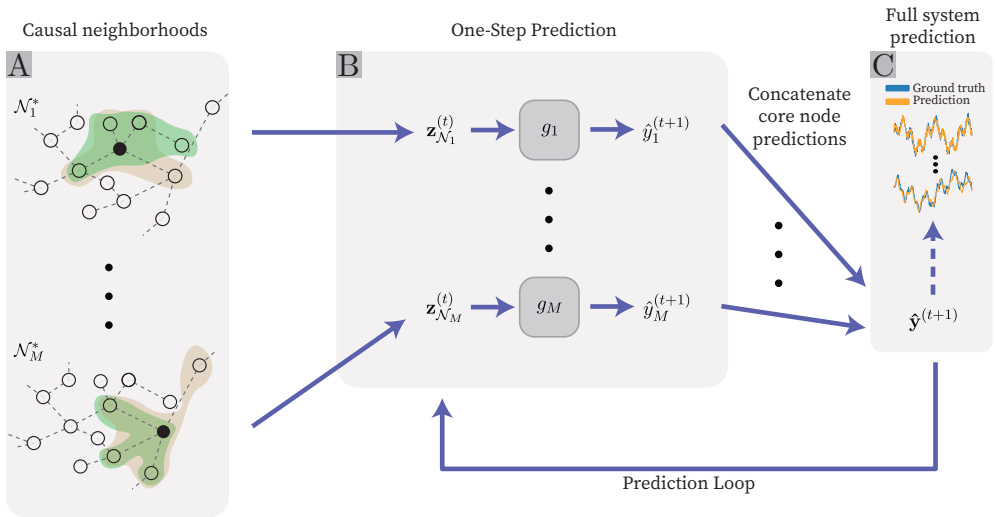


Figure 6.2: Schematic of the prediction step in causal local states. **(A)** All previously inferred causal neighborhoods are combined to a neighborhood matrix. **(B)** A model is trained separately for each of the individual core nodes to perform one-step prediction. **(C)** The one-step predictions are fed back as input data. All one-step predictions are concatenated to construct the final full system prediction.

CHAPTER 7
RESULTS AND EXPERIMENTS

The aim of this chapter is twofold. First, it documents the development process behind the causal local states (CLS) algorithm by presenting the sequence of experiments that guided its design. Second, and more importantly, it addresses many of the natural questions that arise when introducing a new algorithm: *Why was this part designed in this particular way?*

Throughout the experiments presented in this chapter, several potential failure points of network inference and prediction methods are identified. Highlighting these shortcomings provides the foundation for the final CLS algorithm, which was deliberately designed to avoid these pitfalls.

Before discussing the individual experiments, it is useful to briefly restate the objective of causal local states. The goal is to reconstruct an approximation of the causal structure of a dynamical system while simultaneously achieving accurate forecasting. In addition, the algorithm should remain interpretable and scale well to higher-dimensional systems.

7.1 Experiment 0: Is every edge equally important for prediction?

Causal discovery and forecasting are typically treated as two distinct components. Causal discovery methods aim to reconstruct the underlying causal structure of a system [3]. While this structure can serve as valuable system knowledge for downstream prediction tasks, it is not necessarily optimized for predictive performance. In particular, different elements of the causal graph may contribute unequally to forecasting accuracy. For instance, edges that are treated uniformly by causal discovery algorithms can vary substantially in their relevance for prediction. Consequently, assigning equal importance to all nodes may be suboptimal when the objective is accurate forecasting.

A simple illustrative example is synchronization: multiple synchronized nodes often provide redundant information and thus add little value to the feature space. In contrast, a desynchronized node may introduce complementary information that improves predictive performance. Nevertheless, in a purely causal discovery setting, both types of nodes are typically treated in a similar manner.

This first experiment is designed to motivate and illustrate possible connections between the inference and prediction tasks.

Setup: For all predictions in this experiment, a setup of reservoir computing (ESNs) and generalized local states is used. The true neighborhood matrix of the UK power grid is manipulated to test the effects of individual edges on the performance of the full system. To do so, an RC-LS is trained with a modified neighborhood matrix that has all connections except a single selected edge. In order to compare the node performance to a baseline, an RC-LS setup with the true neighborhood is trained and evaluated. By now calculating the valid prediction steps for each node and comparing them to the baseline, the influence of each connection on the prediction quality can be assessed.

The hyperparameters of RC and the other parameters for this experiment are provided in Table A.2.

Results & discussion: Three representative samples are seen in Figure 7.1. Nodes are classified as affected from an edge removal if their valid prediction steps change by more than 5. As evident from this figure, even removing a single node can decrease the prediction performance of the nearby nodes greatly. The blue edge (edge $(4, 5)$) for example, causes a drop in the prediction performance in all of the nearby nodes. The other two edges (edge $(37, 38)$ and edge $(73, 86)$) in the figure result in approximately the same mean VPS change, even though they affect vastly different amounts of nodes ($N = 7$ compared to $N = 20$). This illustrates that a global evaluation via the mean valid prediction steps can hide the true influence on performance.

The summary statistics showing the VPS distribution and also the number of affected nodes can be seen in Figure 7.2. From this figure it is evident that not all edges have the same importance for prediction quality.

The results in this experiment motivated two things. First, the network inference process by causal discovery might not be optimally constructed for prediction. All edges are treated equally, even though the effect on the forecast can vary greatly. Creating an algorithm that simultaneously infers the network and predicts the system trajectory can more easily find a more optimal network for prediction. Second, the influences from different edges can be hidden when only observed in the full system. Further, edges mostly affect local prediction quality and not the forecast of the total system. This motivates an algorithm that optimizes and evaluates the local network structures independently instead of optimizing the full system at once. This is also desirable from a computational standpoint, as an algorithm that separates parts of the problem into independent fractions is scalable along this dimension.

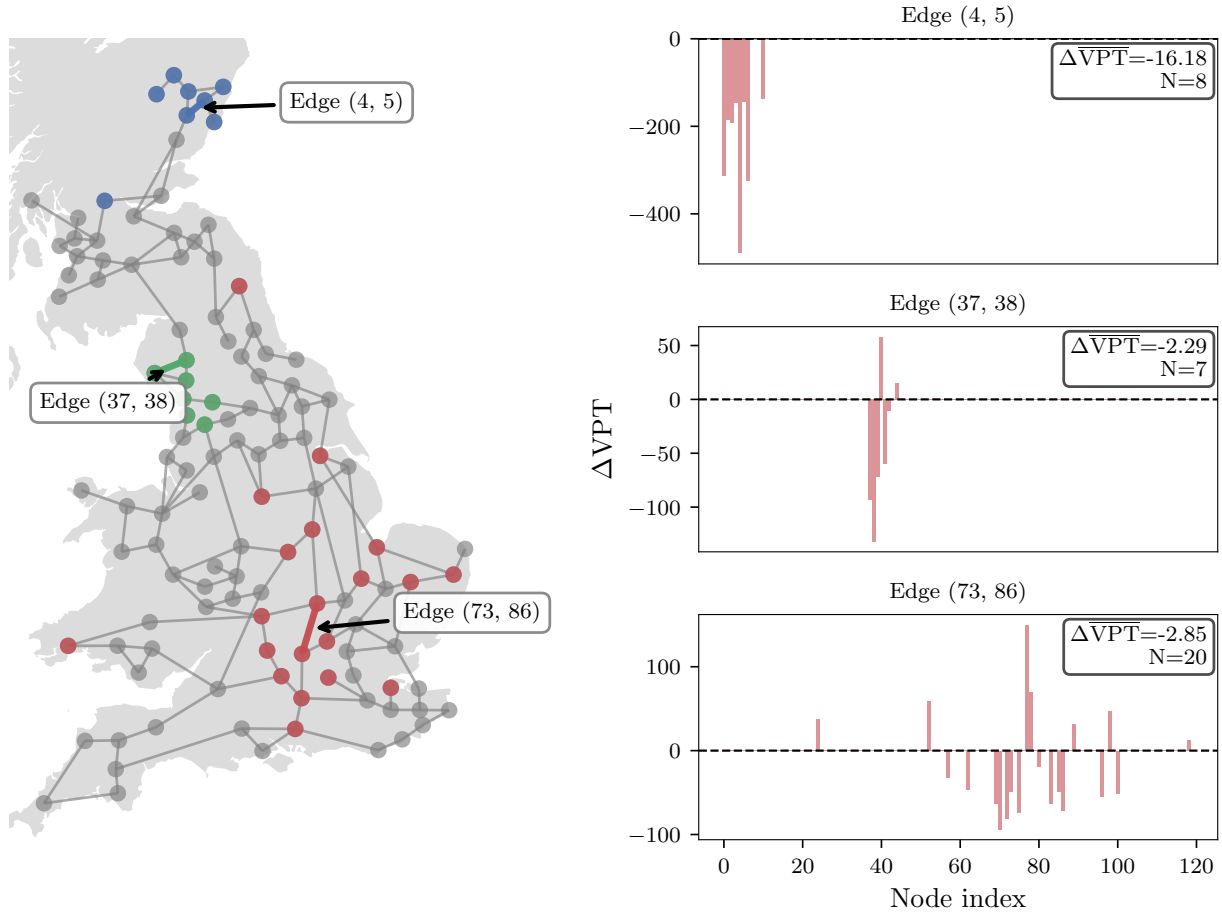


Figure 7.1: Three representative edge removal runs of Experiment 0. In each sample, an edge is selected, and two RC-LS models are trained. One predicts the UK-power grid provided with the true adjacency matrix and the other with the selected edge removed from the true adjacency. In order for a node to be affected by the edge removal, the VPT difference has to be greater than a threshold ($\mathrm{VPT}_{\mathrm{th}} = 5$). **(left)** The removed edges and the corresponding affected nodes are highlighted on the UKPG Network. **(right)** Barplots of the nodewise VPT effects for each of the edge removals

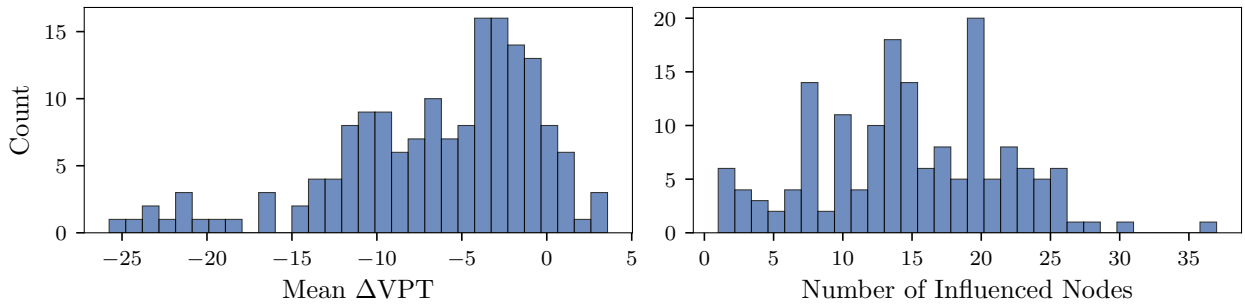


Figure 7.2: In each sample, an edge is selected, and two RC-LS models are trained. One predicts the UK power grid provided with the true adjacency matrix and the other with the selected edge removed from the true adjacency. In order for a node to be affected by the edge removal, the VPT difference has to be greater than a threshold ($\mathrm{VPT}_{\mathrm{th}} = 5$). **(left)** The removed edges and the corresponding affected nodes are highlighted on the UKPG Network. **(right)** Barplots of the nodewise VPT effects for each of the edge removals

7.2 Experiments 1 & 2: simple extensions of GLS

The starting point of this project was the method of generalized local states proposed by Baur et al. (see section 5.4). For a complex system with unknown network structure, they used cross-correlation and mutual information to reconstruct the neighborhoods. Those neighborhoods can then be used to predict the system with LS. [2]

Two experiments that extend their work were conducted. In the first experiment, the established method of neighborhood reconstruction with mutual information is applied to a more complex system. For the second experiment, the similarity measures used by Baur et al. [2] are replaced by causal measures, which are expected to better capture complex dynamics.

In combination with the insights gained from Experiment 0 (see Section 7.1), this will also motivate the usage of a wrapper-based approach in the causal local states framework.

7.2.1 Experiment 1: mutual information for the UK power grid

This experiment applies mutual information to the UK power grid dataset (see Section 4.6) to infer the neighborhood matrix. Both the raw and preprocessed data ($\sin(\theta)$) are compared. To construct the neighborhoods, Baur et al. used a constant neighborhood size q. Then for each core node the q highest NMI values are used to construct the corresponding neighborhood [2]. This constant neighborhood size method is similar to the method by Chu et al. who used a global neighborhood size q as a global hyperparameter to construct the adjacency matrix [6].

Setup: In order to forecast the UK power grid with LS, the neighborhood matrix needs to be constructed. As a first step, the mutual information between all nodes is calculated. To now generate a neighborhood matrix, a max neighborhood size method is used. The neighborhood size method constructs the neighborhood from the q nodes with the highest valued connections to the core. Repeating this procedure for all (reasonable) q-values yields a ROC curve, which quantifies all possible reconstructions based on the MI matrix (See Fig. 7.4). Selecting a specific value for q defines the neighborhoods for every core node (an example for $q = 10$ can be seen in Fig. 7.3).

The parameters for the NMI calculation and the other parameters for this experiment are provided in Table A.4.

Results & discussion: Figure 7.4 shows the results of the NMI calculations for the UK power grid. The ROC curve shows that the sin(.) data seems to be better to work with, but both NMI applied on raw and sin(.) data struggle to capture the structure of the network. As the network is quite sparse, the false positive rates are way too high for adequate predictions, as they result in a significant part of the neighborhood consisting of spurious neighbors.

The kernel density estimate plots (Fig. 7.4) show the distributions of the different NMI values. The max neighborhood size approach chooses a size q, which just separates all points in the KDE plot in two groups. In the given examples, the distributions in NMI values for true and false edges are overlapping a lot. Thus, by following the max neighborhood size approach, it is impossible to avoid a lot of spurious connections in each neighborhood.

There are two main takeaways from this experiment. First, correlation-based measures seem to struggle to infer the causal network of more complex systems. Further, a simple max neighborhood size approach does not work for a general network configuration.

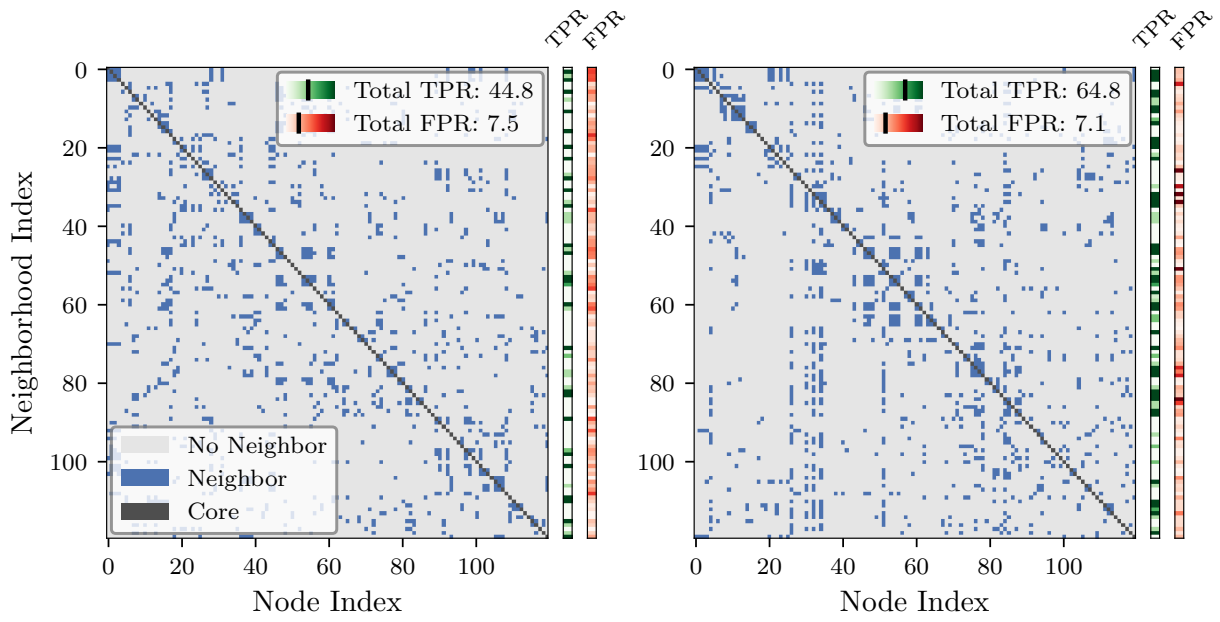


Figure 7.3: Neighborhood matrix reconstruction for the UKPG using normalized mutual information (NMI). The left matrix was inferred from raw data and the right from the sin(.) preprocessed data. The neighborhood matrices are reconstructed by selecting the top $q = 10$ connections for each neighborhood (rows correspond to neighborhoods). The inferred links are compared with the true network, yielding the true positive rate (TPR) and false positive rate (FPR) shown by the vertical bars. TPR runs from 0% to 100%, whereas FPR runs from 0% to 30%.

7.2.2 Experiment 2: causal measures as a similarity measure in GLS

As a possible improvement, this experiment tests bivariate causal measures for network inference. The experiment also aims to further showcase a systematic problem with the current approach of combining network inference with causal measures and forecasting in the literature. The two approaches used perform network inference using a global threshold ([5]) or a global maximum neighborhood size ([6]).

The experiment consists of two parts. The first extends experiment 1 by replacing NMI with the bivariate causal measures transfer entropy and convergent cross-mapping to compute a causal matrix. The second part conducts a more in-depth analysis of a transfer entropy analysis on a coupled L63-Rössler system.

Setup: The first part of the experiment mimics the standard approach for generalized local states proposed by Baur et al. [2] just replacing the similarity measure with TE and CCM. All matrices are calculated for the raw data and preprocessed ($\sin(\theta)$) data.

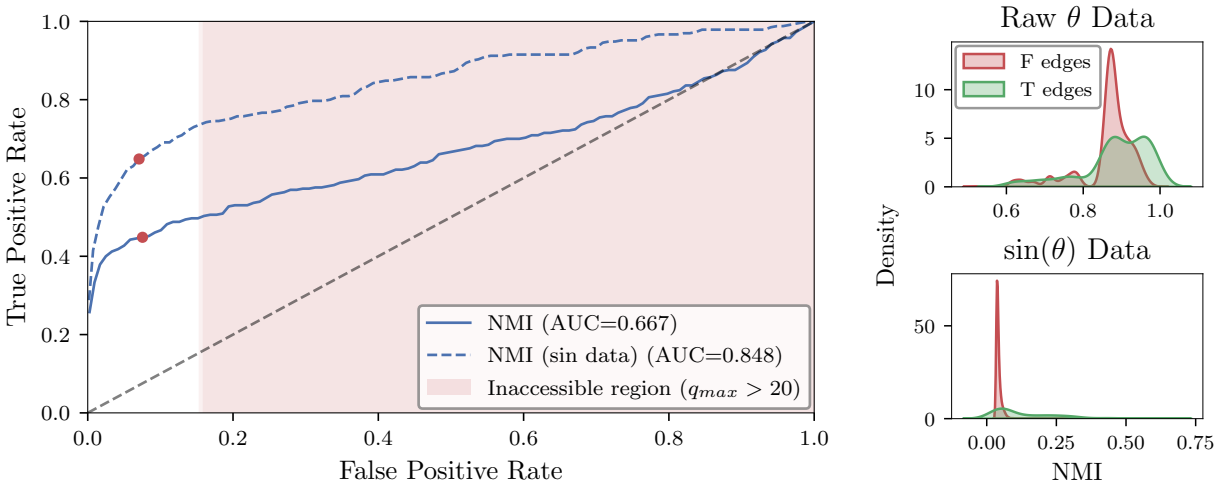


Figure 7.4: Results of the normalized mutual information (NMI) analysis of the UK power grid. Two NMI matrices have been calculated, one on the raw data and one with sin(.) applied prior to NMI calculation. **(left)** ROC curve generated by selecting the top q connections and calculating the TPR & FPR compared to the true adjacency matrix of the UKPG. The inaccessible region is highlighted because neighborhoods of increasing size become computationally infeasible (see subsection 5.3.1) and therefore this part of the ROC curve cannot be accessed. **(right)** Kernel density estimates (KDE) of the NMI distribution, separated according to whether the two nodes are connected (T edges) or not (F edges) in the true adjacency matrix.

Then an RC-LS model is trained using an inferred neighborhood matrix for each of the cases. The matrix is created with the max neighborhood size method and $q = 10$. All inferred neighborhood matrices used for the RC-LS models are shown in Figure 7.6.

For the second part of experiment 2, the bivariate causal measure used in this experiment is transfer entropy (see Section 3.3). The studied system is a composed system consisting of one L63 and one Rössler attractor, with no interaction between the two attractors (see Section 4.3. The binning for transfer entropy is a uniform binning into $n_{\text{bins}} = \sqrt{\frac{L}{4}}$ bins, which is a heuristic also used by [63] that proved sufficient for this experiment. A matrix of the calculated transfer entropy values is calculated (see Figure 7.8. From this transfer entropy matrix, possible neighborhoods can be inferred.

The corresponding parameters used for both parts of experiment 2 are summarized in Table A.5.

Results & discussion: Figure 7.5 shows the results of the TE and CCM calculations for the UK power grid. Similarly to experiment 1, the ROC curve again shows that the sin(.) data outperforms the raw data. Judged by the AUC, transfer entropy performs comparably to NMI on this task, whereas CCM on the sin(.) data captures the underlying structure by far the best.

The kernel density estimates shown in Figure 7.5) illustrate the distribution of inferred causal strengths. Among the evaluated configurations, only CCM applied to the sin(.) data shows a clear separation between true and spurious links.

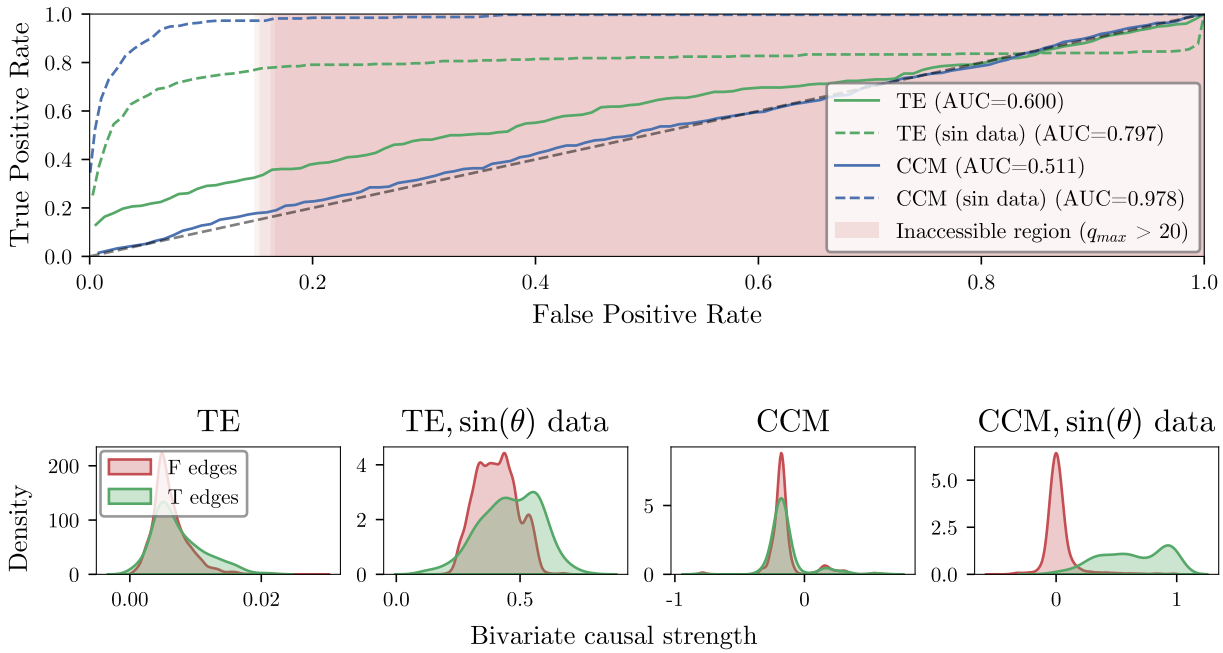


Figure 7.5: Results of the bivariate causal measure (CM) analysis of the UK power grid. For each data format (raw data and one with sin(.) applied to the values prior to CM calculation. (**Top**) ROC curve generated by selecting the top q connections and calculating the TPR & FPR compared to the true adjacency matrix of the UKPG. The inaccessible regions are highlighted because neighborhoods of increasing size become computationally infeasible (see subsection 5.3.1) and consequently, this region of the ROC curve is not accessible. (**Bottom**) KDE plots for the distribution of the calculated CM values, where each is split between whether the two nodes are connected (T edges) in the true adjacency matrix or not (F edges).

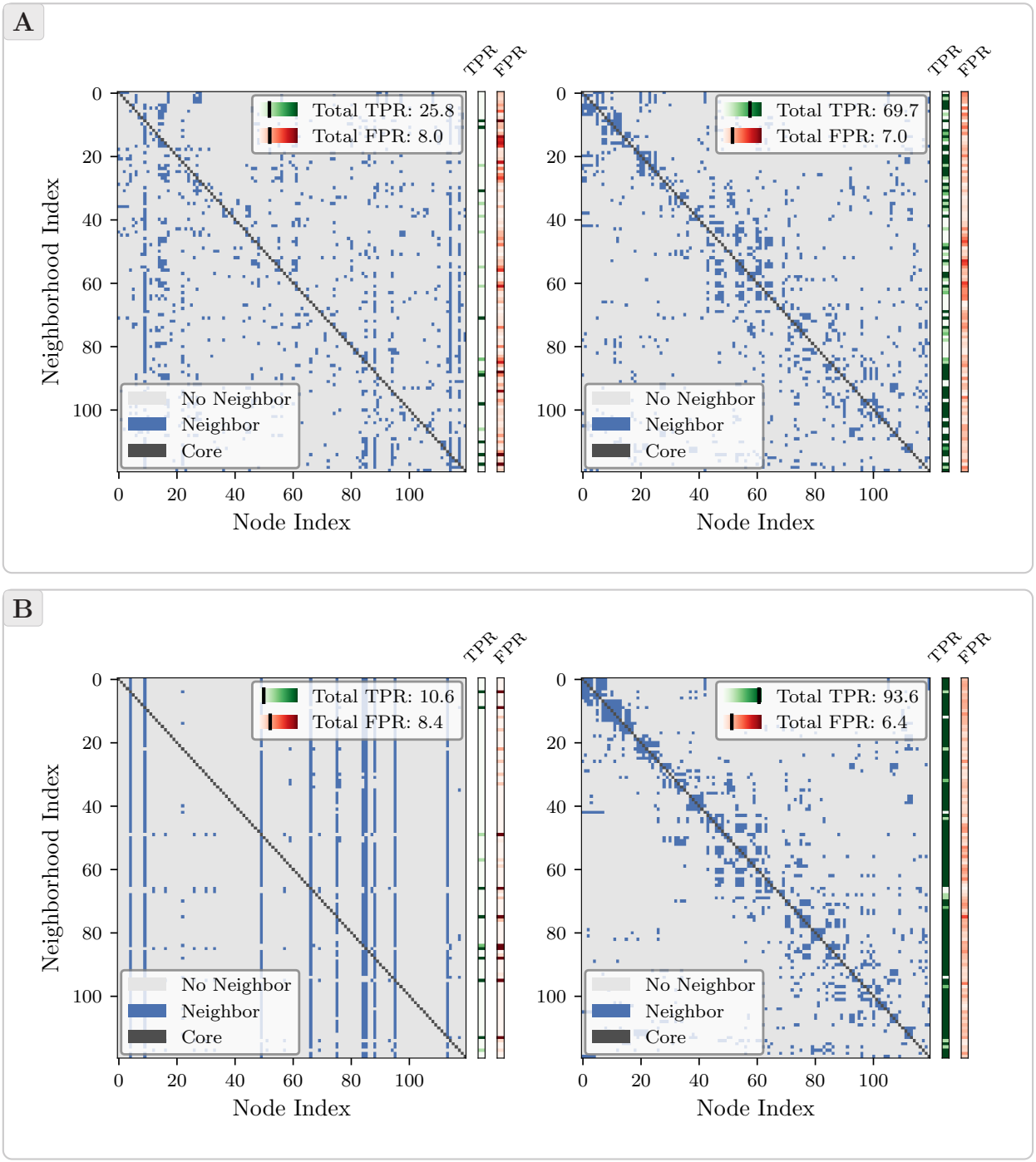


Figure 7.6: Neighborhood matrix reconstruction for the UKPG using transfer entropy (**A**) and convergent cross mapping (**B**). The left matrices show the inference for the raw data, whereas the right matrices show the results on preprocessed sin(.) data. The neighborhood matrix is reconstructed by selecting the top $q = 10$ connections for each neighborhood (rows correspond to neighborhoods). The inferred links are compared with the true network, yielding the true positive rate (TPR) and false positive rate (FPR) shown by the vertical bars. TPR runs from 0% to 100%, whereas FPR runs from 0% to 30%.

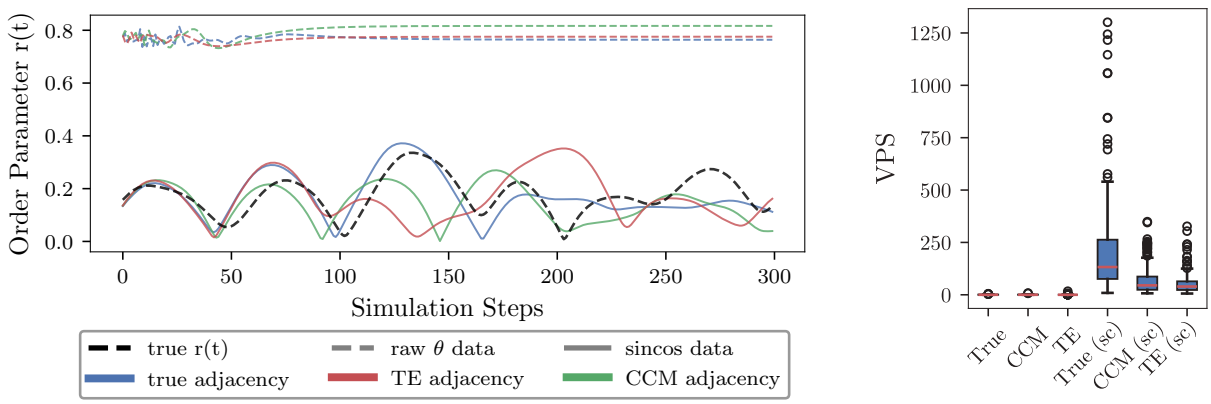


Figure 7.7: Predictions for the different adjacency matrices inferred in Experiment 1. Six RC-LS setups are trained. For each data format (raw and $\sin(\cdot)$-transformed), models are trained using either the true neighborhood matrix or matrices inferred via transfer entropy or convergent cross mapping ($q = 10$, see Figure 7.3). **(left)** Short-term predictions for the full system evaluated using the Kuramoto order parameter. **(right)** Valid prediction steps for all six models. Each boxplot aggregates the valid prediction steps across all individual nodes, with the median and outliers indicated.

The predictive performance of models trained on neighborhoods derived from the TE and CCM matrices is presented in Fig. 7.7. Short-term forecasting accuracy of the full system is quantified using the order parameter. Models trained on the raw θ data fail to produce meaningful predictions, even when provided with the true adjacency matrix. In contrast, transforming the data to $(\sin(\theta), \cos(\theta))$ leads to a substantial improvement in predictive performance. Nevertheless, the models based on inferred adjacency matrices remain clearly inferior to the RC-LS baseline using the true interaction structure, indicating that a neighborhood reconstruction based solely on CCM and TE is insufficient for accurate prediction.

The main results of the second part of this experiment are shown in Figure 7.8. This analysis of the combined L63-Rössler system aims to illustrate the drawbacks of the global threshold and global neighborhood size approaches. For example, according to the governing equations, the causal parents of the x-coordinate R_x of the Rössler attractor are R_y and R_z. However, TE does not find a higher causal influence of R_z compared to the completely independent Lorenz dimensions. Reconstruction of the neighborhood of R_x with the maximum neighborhood size method is therefore never able to correctly separate the L63 dimensions from R_z. Although this misclassification may appear negligible in a low-dimensional setting, its impact becomes increasingly problematic as the system dimension grows. In higher-dimensional systems, the accumulation of false positives may then occupy a substantial fraction of the resulting feature vector. This, in turn, can degrade prediction performance and obscure the true causal structure.

A limitation of the global threshold approach is also evident from the results shown in Figure 7.8. Because the two attractors are on different scales, the TE values of the different systems are not generally comparable. In this case, no single global threshold can reliably recover the complete system structure. An example of this is shown in Figure 7.8C, where the threshold was chosen such that the neighborhood inference of R_z yields the correct neighbors.

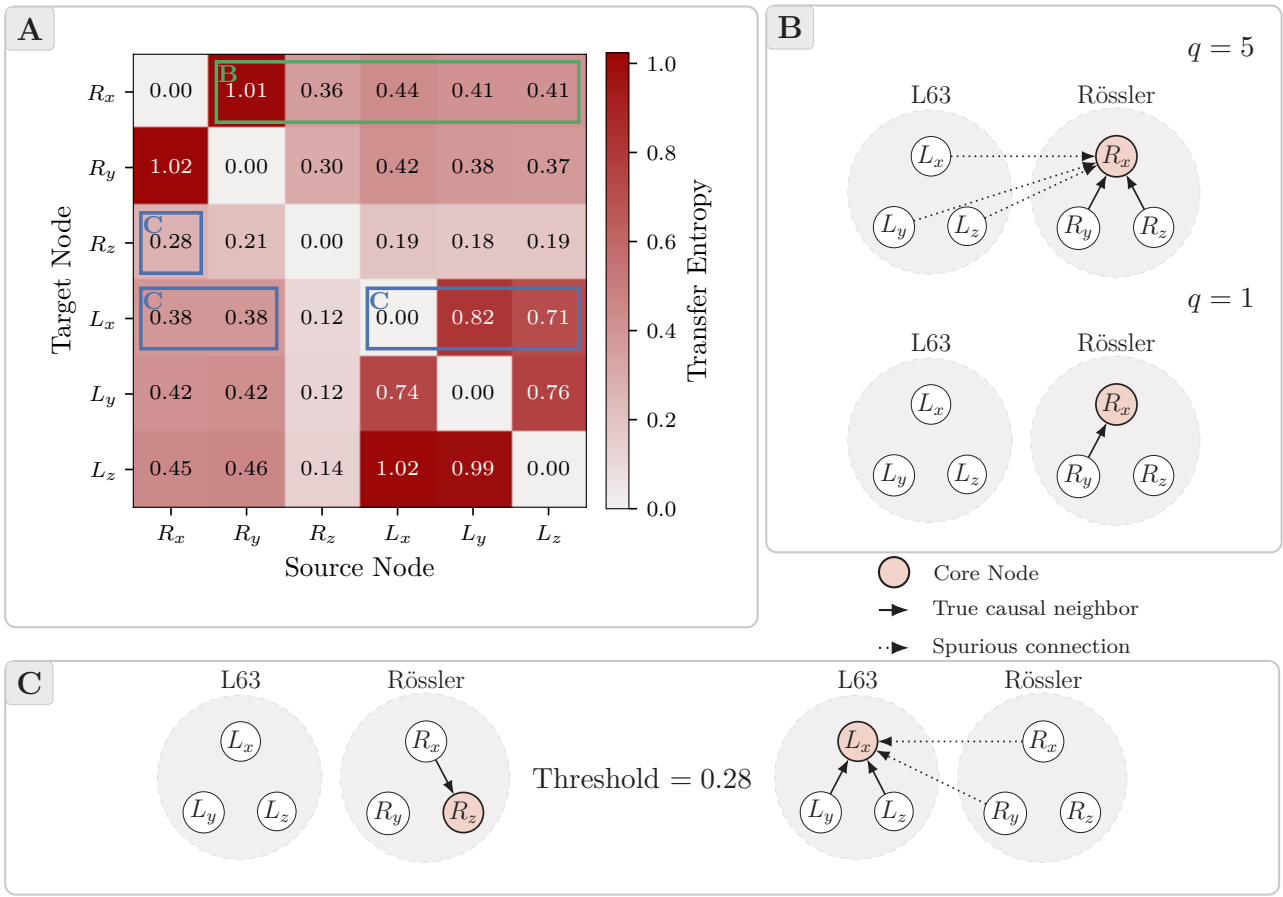


Figure 7.8: Exemplary causal analysis with transfer entropy on the concatenated L63-Rössler system. The figure illustrates the limitations of employing a global hyperparameter for neighborhood construction. (**A**) Transfer entropy heatmap of the combined system. The green box highlights the neighborhood analyzed in panel (B). The blue boxes correspond to the neighborhoods discussed in panel (C). (**B**) Examples for neighborhood reconstructions for R_x when fixing the global number of neighbors to $q = 1$ and $q = 5$. There is no neighborhood size q that could capture the true causal neighbors of R_x (R_y & R_z) without adding spurious connections to L_x, L_y, L_z. (**C**) Two different neighborhoods with R_z and L_x as respective core node showcasing the reconstruction with the global threshold method. The threshold is set to 0.28, which just captures the true neighbor of R_z. There is no global threshold that could correctly identify both neighborhoods without introducing spurious edges or false negatives in the other.

As the necessary threshold for that is quite low (0.28), this leads to the introduction of spurious connections in the neighborhood of L_x.

These observations indicate that, if causal local states should be applicable across different network structures, the algorithm cannot depend on a single global hyperparameter to determine all neighborhoods. Instead, the neighborhood inference must be separable at the level of individual core nodes. For this, a possible solution is a wrapper-based approach. A wrapper is a model acting as a black box, which can take a set of features and evaluate the feature set according to a criterion. Chapter 6 provides a more in-depth look into the usage of wrapper models in causal local states. By applying a wrapper, each neighborhood can be constructed independently for its respective core node. There are certain design choices one must make when employing a wrapper model. Guyon et al. claim that for a wrapper method to be successful, one needs to choose three things. The strategy by which the feature subset space is searched, how to assess the performance of the wrapper, and the wrapper itself [61]. The following experiments will assess precisely that.

7.3 Experiment 3: different wrapper strategies

This section focuses on the selection of the wrapper model and the associated prediction generation. In general, any prediction model can act as a "black box" wrapper. However, choosing a suitable model for the dataset can greatly improve the performance of the algorithm. Furthermore, to keep the computational requirements manageable, an efficient model is required. Two candidates that have proven successful in forecasting complex systems are echo state networks [51] and next generation reservoir computing [55]. Both models are explained in detail in sections 5.2 and 5.3.

For those models, there are different strategies to generate the predictions. All strategies have in common that the core node is fixed and the performance of the wrapper is evaluated by calculating the MSE between the true trajectory and the predicted core node trajectory. The prediction error of the other nodes in the neighborhood does not influence the wrapper score. This setup is very much similar to a Granger causal idea of testing causality, where one expects nodes that share a true edge to improve the prediction performance, while spurious nodes do not contribute to performance. One possible strategy used by Li *et al.* is a wrapper approach, where they generate a one-step prediction with a modified RC model and then evaluate the corresponding prediction error for subset selection [4]. This thesis compares multiple different strategies:

- Resetting the full neighborhood at each time step and performing one-step predictions: *One-step strategy*
- Running a free neighborhood strategy but resetting the full neighborhood at every N_{reset} steps: *Multi-step strategy*
- NGRC-Inference mode, with the core node at $t+1$ as target: *NGRC-Inference strategy*

All strategies are designed such that the neighborhood can be tested in isolation from the full system. This is very important as it allows scalability of the algorithm. There are several other possible strategies to generate the predictions. However, conducting a broad comparison of a lot of different wrapper strategies is beyond the scope of the thesis.

Setup: To illustrate the trade-offs between different wrapper strategies, a composite system consisting of one L63 attractor and one Rössler attractor that do not interact with each other is used. Since the concatenated system is $d = 6$ dimensional, for a given core node, there are only $2^{d-1} = 32$ possible neighborhood configurations. This makes it possible to evaluate all neighborhood configurations.

Because the two attractors are independent, the neighborhood sets can be grouped according to how many features in a given neighborhood originate from the same attractor as the core node. The resulting groups are: (i) no feature from the same attractor, (ii) one feature from the same attractor (distinguishing between the possible features), and (iii) both features from the same attractor. The expectation is that a well-defined wrapper should return a high error for neighborhoods without same attractor features. As features from the same attractor are included, the prediction error should decrease.

The parameters for this experiment are shown in table A.6.

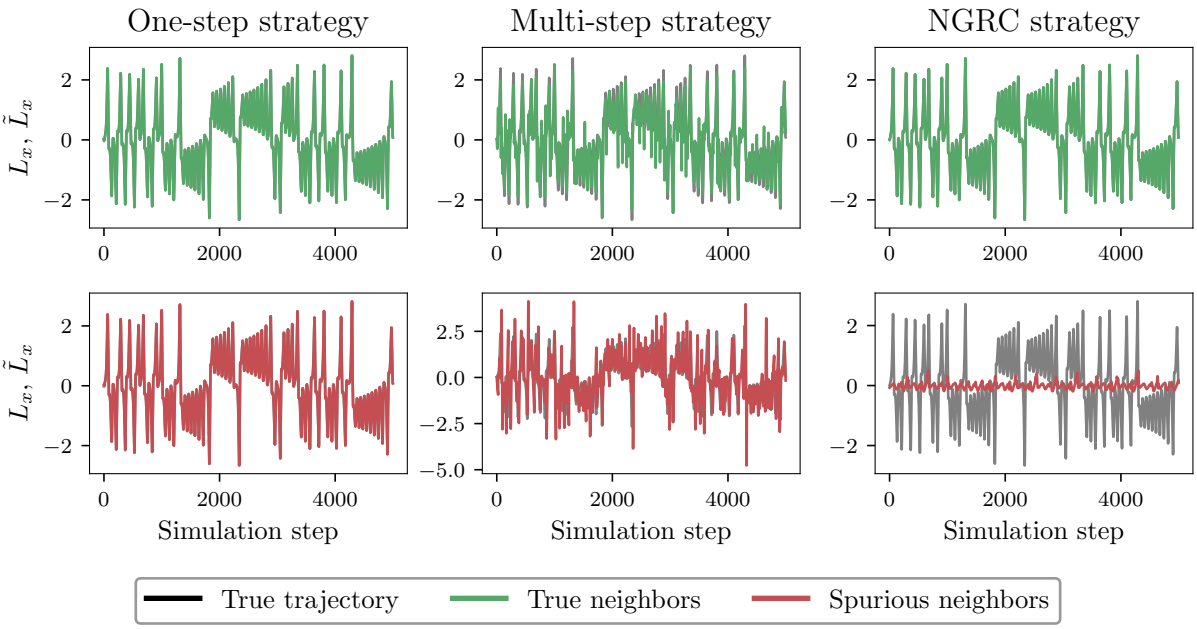


Figure 7.9: Sample predictions for a concatenated L63-Rössler system (see Section 4.3 for more details). Two different neighborhood sets for the core node L_x are used to train wrapper models across three wrapper strategies. The neighborhoods consist of the true neighbors L_y, L_z (green) and spurious neighbors R_x, R_y, R_z (red).

Results & discussion: The main results of this experiment can be extracted from Figure 7.10. As visible in this figure, there is a clear difference in the performance of the various strategies. A good strategy should show a clear distinction between the MSE of neighborhoods containing the true neighbors compared to spurious neighbors (even more so in this system, as the L63 and Rössler attractors share no coupling at all).

The One-Step Strategy (Figure 7.10 (A)) shows a slight trend where neighborhoods containing the true neighbors exhibit a slightly lower MSE. The overall accumulated error in the prediction is very low, independent of the neighborhood. The past states of the core node

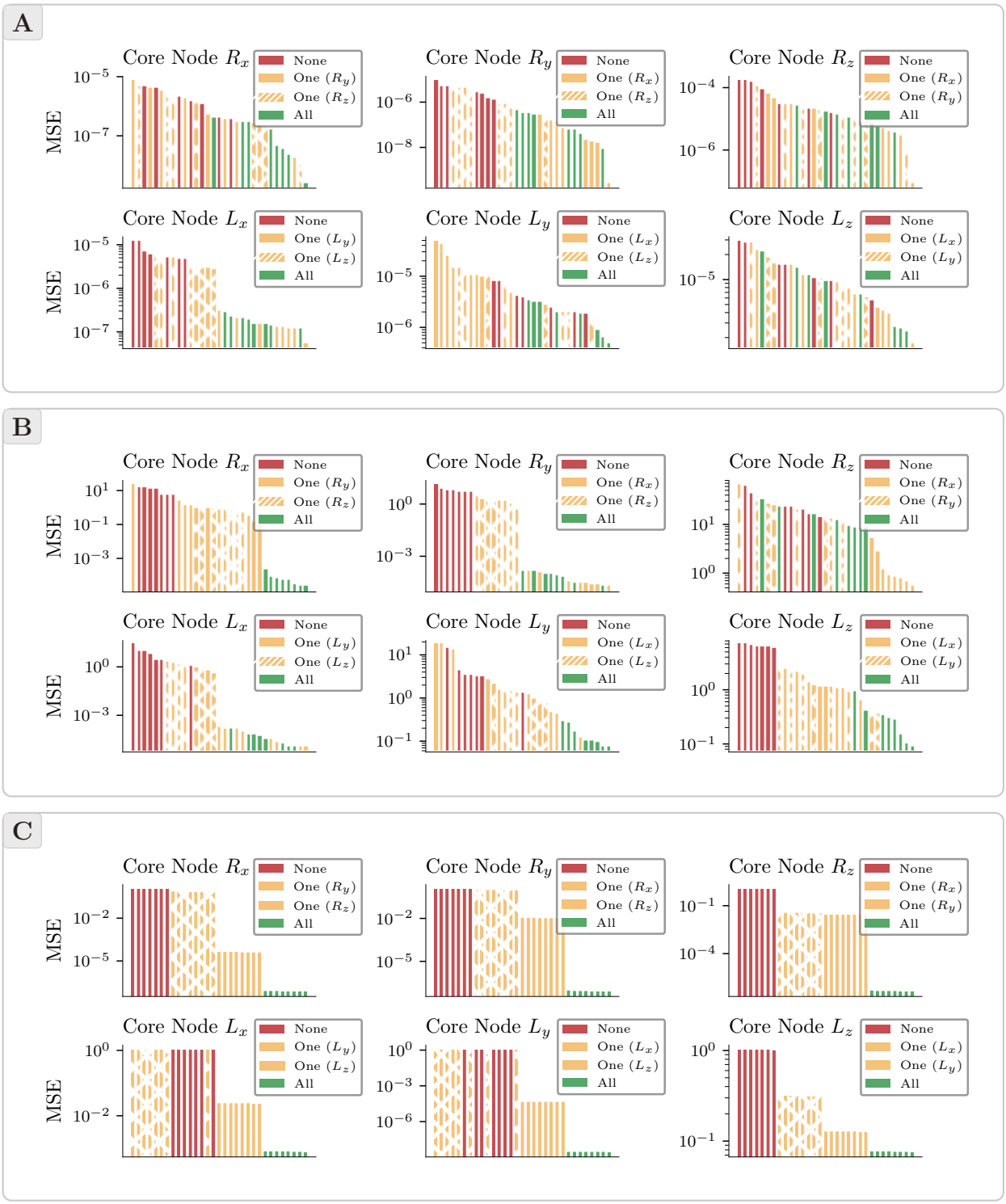


Figure 7.10: Comparison of MSE for core node predictions across different wrapper strategies. All possible neighborhood configurations in a concatenated L63-Rössler system (see Section 4.3) are evaluated for each core node. Neighborhoods are grouped by the number of features originating from the same attractor as the core node: (red) none, (orange) one (distinguishing the two possible cases), and (green) both. The subplots (**A-C**) correspond to the One-Step (**A**), Multi-Step (**B**), and NGRC-Inference strategies (**C**).

typically exhibit a strong causal influence on its future state [1]. Consequently, including the core node in the feature vector can therefore dominate the prediction task and mask the effect of the neighbors. This is evident from figure 7.9, where both wrappers generate an equally good prediction of the core node trajectory even though one has only spurious neighbors added to the neighborhood. The core node is the main driver of the prediction quality in this case. A further disadvantage of the One-Step Strategy is that since the neighbors are reset to the true values at every time step, the reservoir has to be re-synchronized for every prediction step. This introduces a lot of computational overhead, which can be problematic when dealing with higher-dimensional systems.

A possible improvement of this is the Multi-Step Strategy. The idea is that by allowing the system to evolve for t_{reset} steps, this allows the error to accumulate and therefore compensates for the effect of the core node. This in turn also requires fewer re-synchronizations, which reduces the computation. As evident from Figure (Figure 7.10 (b)), this clearly improves the separation between the different neighborhood classes. The prediction quality now deteriorates when no true neighbor is present, as the core node alone is not enough to drive a good prediction (see Fig. 7.9). However, even with this strategy, the distinction between the different regimes of the neighborhood candidates is not perfectly clear yet.

The last tested strategy makes use of the inference mode of NGRC. With this setup, it is possible to exclude the core node from the feature set and try to infer the core node trajectory solely from the neighbors tested. This completely eliminates the problem that the causal relation of the core node to its own past masks the influence of the other nodes. As evident from figure 7.9, the prediction for a neighborhood with only spurious neighbors has no resemblance to the true trajectory at all. Furthermore, as inference mode allows calculating all predictions at the same time, this strategy is also the most efficient. The separation between the different classes of neighborhoods is clearly visible in Figure 7.10 (C).

7.4 Different neighborhood set search methods

Experiments 4 & 5 focus on the selection of the search strategy by which the feature space is traversed. A wide range of methods exists for this, including best-first search, greedy search, sequential backward selection, and genetic algorithms [61]. Since the CLS framework aims to keep computational costs low, the obvious first candidates are simple one-directional search strategies: forward selection and backward elimination. Forward selection incrementally adds features, generating subsets of increasing size [64, p. 207]. In contrast, backward elimination begins with the full set of variables and removes features one by one, producing subsets of decreasing size [64, p. 209]. Both methods rely on a selection criterion that determines whether a node should be included in the neighborhood. In this case, the criterion will be the score of a wrapper model trained on the feature subset.

[1] This property depends on the specific form of the underlying system equations; In general this does not hold for every system

7.4.1 Experiment 4: forward selection & backward elimination

To reduce complexity even more, a variant of forward selection is used. Instead of evaluating all possible candidate neighbors at each step, the first neighbor that satisfies the selection criterion is immediately added to the neighborhood. Since checking all potential neighbors becomes computationally infeasible in high-dimensional systems, greedy forward selection provides a good alternative. Following a similar argument, the same variant of backward elimination is used in this thesis.

Setup: All different search strategies are studied on the UK power grid, as it is the most complex system used in this thesis and therefore best suited to highlight the limitations of the different search strategies. The wrapper method used in this experiment is next-generation reservoir computing, as it allows for fast computation of wrapper performances with the inference mode. This experiment consists of two different runs, testing the forward selection and backward elimination, respectively. The specific parameters for the NGRC model and the simulation can be taken from Table A.7.

To test forward selection, a maximum neighborhood size has been set to $q_{\max} = 20$. This is done to keep the size of the feature vector in NGRC manageable (see Subsection 5.3.1 for a more thorough discussion of this issue). Forward selection is done in isolation for each core node with the candidate neighbors tested in random order. This forward selection is run until the neighborhood size reaches $q_{\max}$ or convergence. A sample of a converged matrix is shown in 7.11(a).

For backward elimination, the initial feature set is a neighborhood of size $q_{\text{init}} = 25$ that consists of the true neighbors and random nodes that fill the remaining spots. At each step, single nodes are removed in random order until a node removal that improves MSE is found. A run continues until no further improvements can be made or the neighborhood size reaches $q = 1$. Subfigure 7.12(a) shows an example of such a run.

Results & discussion: The result of the forward selection run are shown in Figure 7.11. Subfigure 7.11(b) demonstrates that the search algorithm does not consistently recover the set of true neighbors, succeeding only in 35% of the runs. All converged neighborhood sets exhibit a higher MSE than the true neighborhood set, indicating that the algorithm becomes trapped in a local minimum. This tendency to get stuck in local minima is common for greedy search algorithms [65].

The inconsistency in the results suggests that the outcome depends strongly on the initial random ordering of candidate nodes. Since every node has a (possibly long) path to the core node, each node exerts at least a small influence on the core node. Consequently, many nodes provide a slight prediction improvement when evaluated early in the search. Depending on the order in which the neighbors are tested, this introduces a lot of nodes into the neighborhood that get redundant once a node closer to the core is tested. So even in cases where the algorithm correctly identifies the true neighbors, the resulting neighborhood is much larger than necessary. Greedy forward selection has no way of removing redundant features.

Moreover, adding a true neighbor to an already large neighborhood yields a reduced observable improvement. As a result, true neighbors are harder to discover when tested later in the feature selection, as they then occupy a smaller part of the feature vector.

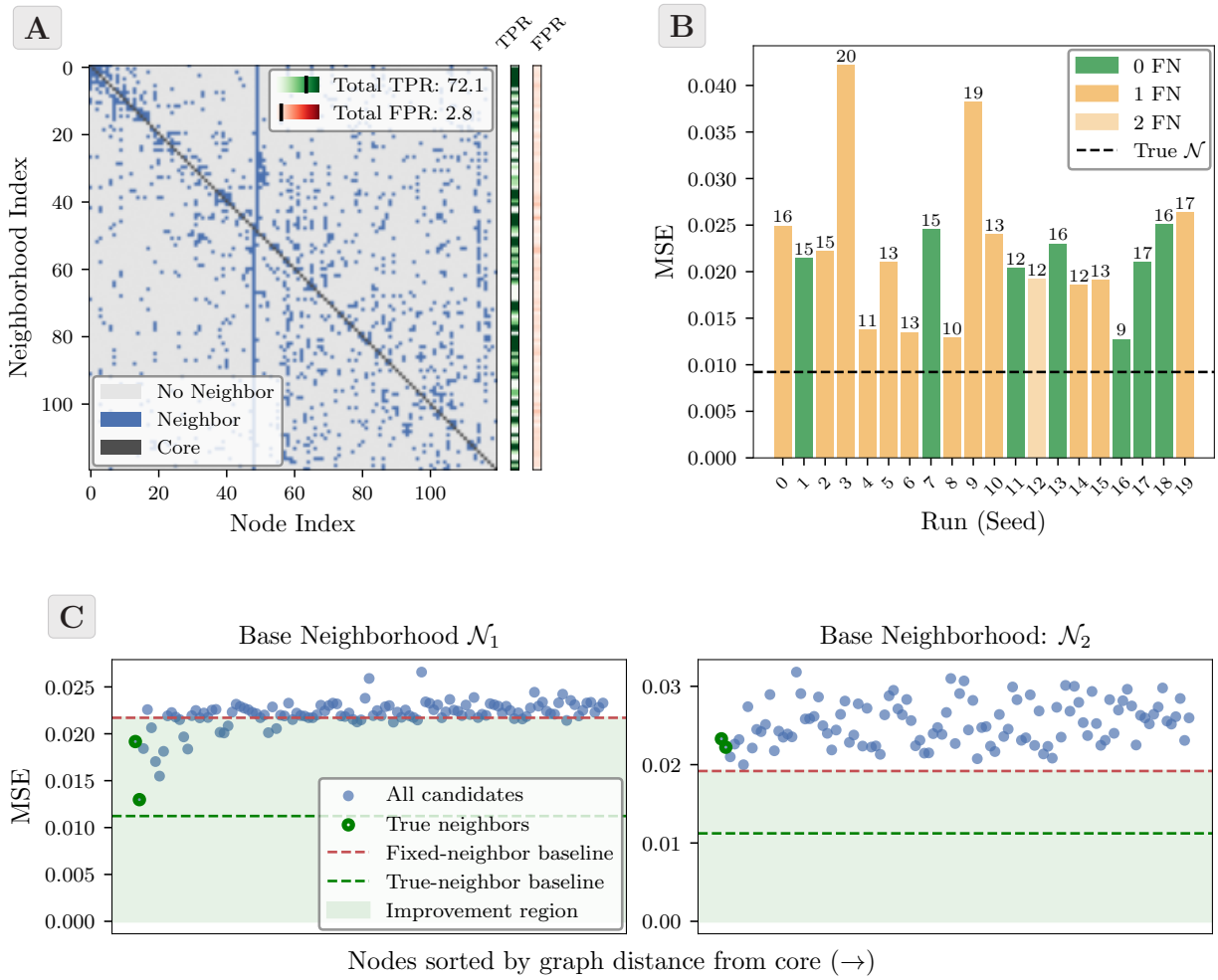


Figure 7.11: Greedy forward selection (FS) for adjacency reconstruction of the UK power grid (Experiment 4) with a maximum search size $q_{\max} = 20$. NGRC is used as a wrapper model to evaluate neighborhood candidates. **(A)** Reconstructed neighborhood matrix for the UKPG. Inferred links are compared to the true network, yielding the true positive rate (TPR; 0-100%) and false positive rate (FPR; 0-30%) shown by the vertical bars. Each row corresponds to the converged neighborhood from the greedy FS. **(B)** Repeated neighborhood inference runs for example node 32 (core node). Bars show the converged neighborhoods colored by the number of true neighbors missing (false negatives) and have the final size annotated. The true neighbors of node 32 are $\{30, 33, 34, 35\}$. **(C)** Node addition tests for the base neighborhood of node 32. Each marker represents a candidate node added to the base neighborhood and evaluated by training the wrapper model. Both $\mathcal{N}_1$ and $\mathcal{N}_2$ contain the true neighbors but $\mathcal{N}_2$ also contains 10 further random nodes.

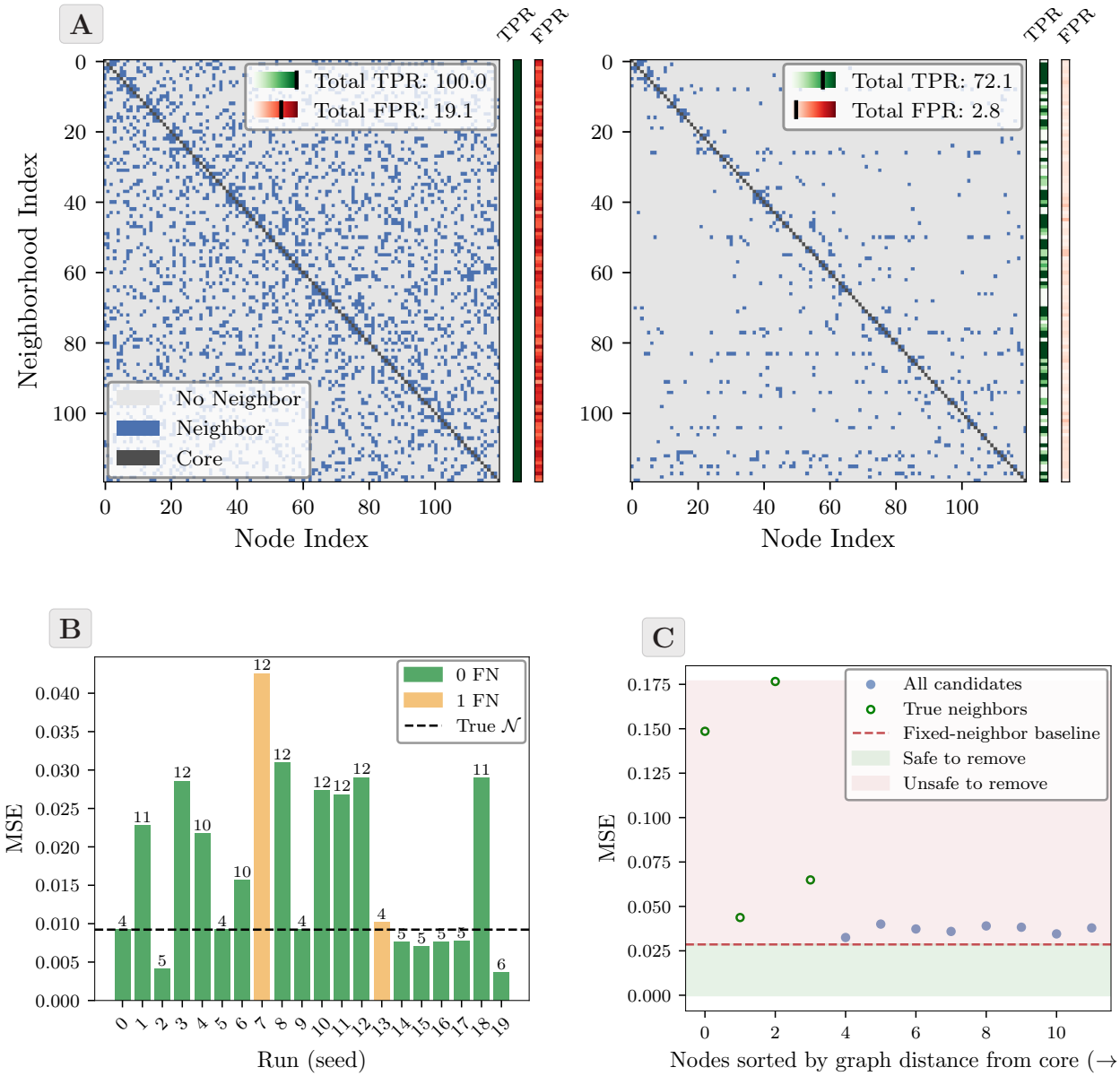


Figure 7.12: Backward elimination (BE) for adjacency reconstruction of the UK power grid (Experiment 4) with an initial neighborhood size of $q_{init} = 25$. NGRC is used as a wrapper model to evaluate neighborhood candidates. **(A)** Initial neighborhood matrix on the left, neighborhood matrix resulting from converged backward elimination on the right. All edges are compared to the true network, yielding the true positive rate (TPR; 0-100%) and false positive rate (FPR; 0-30%) shown by the vertical bars. Each row corresponds to the converged neighborhood from the greedy FS. **(B)** Repeated backward elimination runs for a representative core node (32) starting from different intial neighborhoods. Bars show the converged neighborhoods colored by the number of true neighbors missing (false negatives) and have the final size annotated. The true neighbors of node 32 are $\{30, 33, 34, 35\}$. **(C)** All possible node removals starting from a converged neighborhood ((**B**): seed = 7). Each marker represents a node removed from the base neighborhood. The new neighborhood is evaluated with a wrapper.

An example of this is shown in Figure 7.11(c). All possible additions in the next forward-selection step are evaluated for two neighborhoods, $\mathcal{N}_1$ and $\mathcal{N}_2$. Both contain two true neighbors, with $\mathcal{N}_2$ additionally containing ten random spurious nodes. In the larger neighborhood, adding the remaining true neighbors no longer leads to an improvement in the MSE.

The results of the backward elimination run are shown in Figure 7.12. A primary limitation of backward elimination is that the algorithm should ideally start from a feature set containing all candidate variables. As discussed in subsection 5.3.1, the feature space of NGRC and, consequently, the required computational resources grow very fast. Starting from the full system is, in general, not possible if an NGRC model with higher-order terms were used.

Subfigure 7.12(b) indicates that backward elimination recovers the true neighbors relatively consistently for the considered example. In several runs, the resulting MSE is even lower than the reference obtained with the true neighborhood. This observation either suggests that alternative neighborhood configurations may be better suited for predicting the core node or that the model begins to overfit the data. Either way, this shows that the common assumption of causal sufficiency (Definition 3) might not be a good constraint for an algorithm that tries to optimize the inferred neighborhood for prediction.

Two additional drawbacks of BE become evident from the experiment. First, removing almost any node initially yields a small improvement, as the signal-to-noise ratio again obscures the influence of the true neighbors (similar to the situation illustrated in Subfigure 7.11(c)). As a consequence, backward elimination may remove important nodes early in the search, even though other nodes would be more appropriate candidates for removal. While evaluating all possible removals would mitigate this issue, performing such a full search over all candidate neighbors is computationally expensive, particularly in the early stages when the neighborhood is still large.

Second, backward elimination can also become trapped in local minima. From Subfigure 7.12(b) it is evident that many runs still converge to neighborhoods with substantially higher MSE than the reference. A representative example of such a failure case is shown in Subfigure 7.12(c). In this situation, removing any of the true neighbors degrades the prediction performance more strongly than removing the remaining nodes. However, those remaining nodes still provide a small improvement and are therefore not removed by the greedy elimination rule. Ideally, nodes that contribute only marginally to prediction accuracy should also be removed. Their weak effect on prediction suggests only a weak Granger-causal influence on the core node.

The next experiment introduces an alternative search strategy that combines forward selection and backward elimination in order to mitigate several of the limitations observed for both methods.

7.4.2 Experiment 5: combining forward selection and backward elimination

This experiment introduces a search method that tries to mitigate all the previously discussed shortcomings of FS and BE.

To avoid the inconsistency of FS, the order in which the neighbors are tested can be optimized. As shown in Experiment 7.2.2, bivariate causal measures can be used to at least partially capture the underlying network structure. Calculating a causal matrix and sorting the possible neighbors according to the causal influence is a preprocessing step that improves the likelihood that nodes relevant for prediction are tested first. In this experiment, neighbor candidates are sorted by their causal influence onto the core node, and then incrementally increasing neighborhood candidates are built from this. The optimal neighborhood from those candidates is then selected and serves as the starting point for backward feature elimination. This mitigates the computation overhead introduced in backward elimination when the starting neighborhood is really big.

Furthermore, to mitigate the local minima, two relative improvement thresholds are introduced: α_1 and α_2. The idea is that during the forward selection part, as long as the neighborhood is still small [2], an inclusion of true neighbors should see a spike in the improvement of the prediction quality. So by introducing a threshold, this allows for better selection. For backward elimination the rationale is inverted. Removing a true neighbor is expected to deteriorate the performance. If the performance is similar after a removal, it is evidence that the removed node does not hold much valuable information.

The concrete implementation of the search method in this experiment is outlined in the algorithms: 2, 3, 4.

Setup: The CLS neighborhood inference strategy is tested on the UK power grid. The data is setup in the sin(.) format as experiments 1& 2 have proven this much easier to work with (see section 7.2). For the filter step, the causal matrix is calculated with convergent cross-mapping. Then three NGRC-LS models are trained using the true, initially inferred, and reduced adjacency matrices. The simulation parameters and model parameters for this experiment are shown in Table A.8

Results & discussion: Figure 7.14 shows the neighborhood matrices before and after applying the feature removal step. The initial reconstruction is already substantially improved compared to the forward selection results obtained in Experiment 4 (see Figure 7.11). Applying feature removal further reduces the FPR by 0.7% and the TPR by 6.1%. While the reduction in TPR might initially appear to be a drawback, the sparsity of the network means that a larger number of spurious connections are eliminated. Moreover, not all true connections might to be necessary for accurately predicting a core node[3].

To illustrate the necessity of the feature removal, an example neighborhood inference is shown in Figure 7.13. The initial neighborhood inference successfully identifies all true neighbors. However, CCM assigns a higher causal score to a group of nodes located in the southwest than to the actual neighbor, causing these spurious nodes to appear among

[2] Otherwise signal-to-noise ratio might pose a problem again

[3] For example, if multiple neighbors are synchronized.

the neighborhood candidates. As a consequence, a neighborhood selection procedure that recovers the true neighbors would also include these spurious nodes. Subfigure 7.13(b) now shows the converged backward removal step that started from the neighborhood seen in Subfigure 7.13(a). The cluster of spurious connections in the southwest is successfully eliminated by the feature removal step.

The predictive performance of models trained on the inferred neighborhoods (before and after reduction) is compared with a model trained on the true adjacency in Figure 7.15. The short-term predictions of the full system are summarized by the order parameter in Subfigure 7.15(a). Both the inferred and reduced adjacency-based models diverge from the reference trajectory, resulting in unusable forecasts after a short time. However, although the model trained on the true adjacency does not diverge, it also fails to reproduce the system dynamics accurately. This is further supported by Subfigure 7.15(b), which shows that the number of valid prediction steps is similar for all three models and occurs earlier than the onset of divergence.

A key limitation in this experiment was that the NGRC instances had to be run without including any higher-order terms. Including such terms led almost immediately to complete divergence in the predictions, independent of the chosen hyperparameters or adjacency matrix. An NGRC model without nonlinear terms likely lacks sufficient capacity to learn the higher-order Kuramoto dynamics, which are inherently nonlinear. This suggests that the poor prediction quality is not caused by the neighborhood inference process itself, but rather by limitations of the wrapper model or the generalized local states framework.

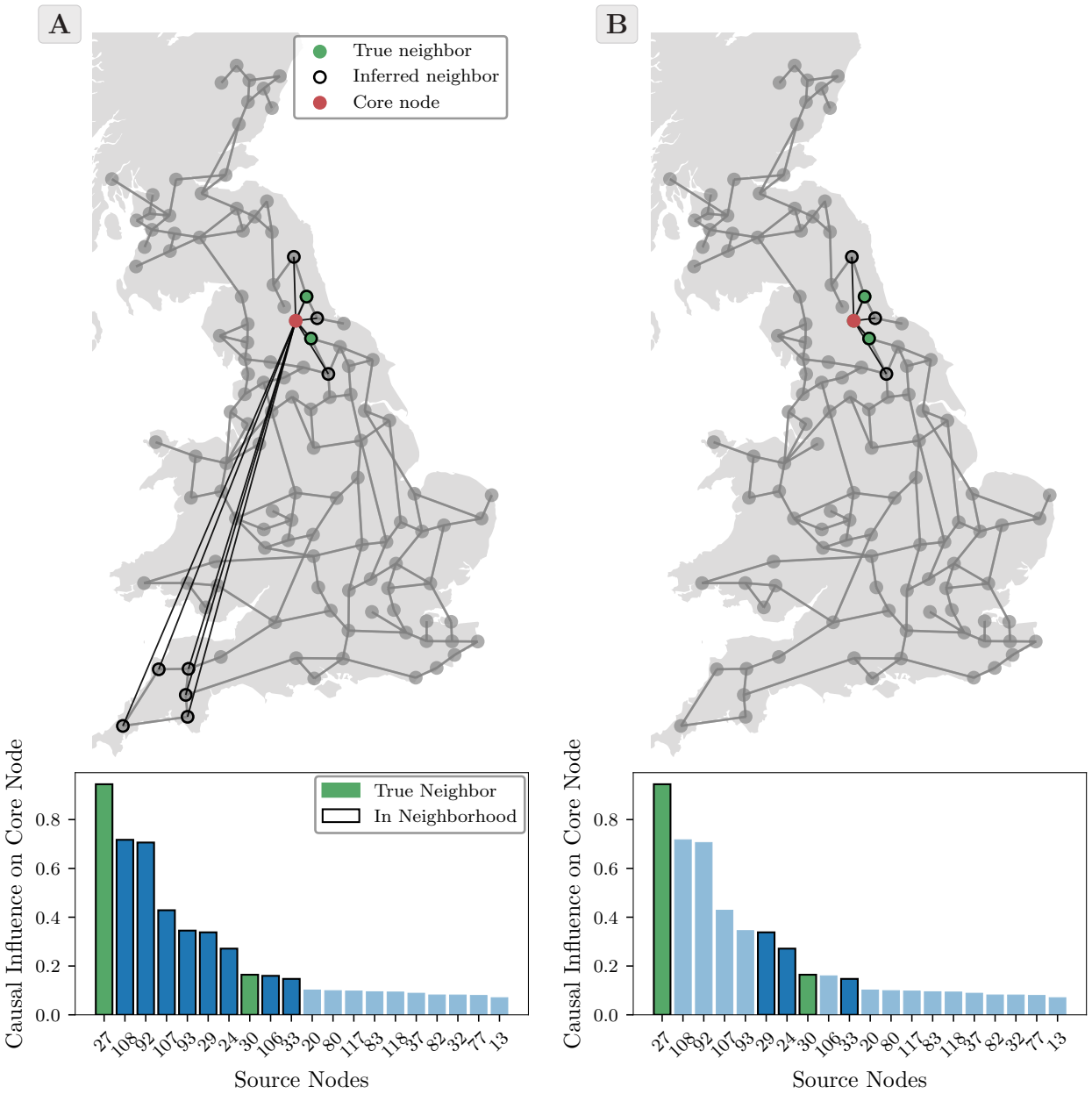


Figure 7.13: Causal local states neighborhood reconstruction example on the UK power grid using NGRC as a wrapper model. The core node, corresponding true neighbors and the inferred neighborhood found by CLS are highlighted together with the causal influence calculated with CCM. **(A)** The neighborhood found after the filter & selection steps of CLS. **(B)** Approximate causal neighborhood found after applying backward feature elimination.

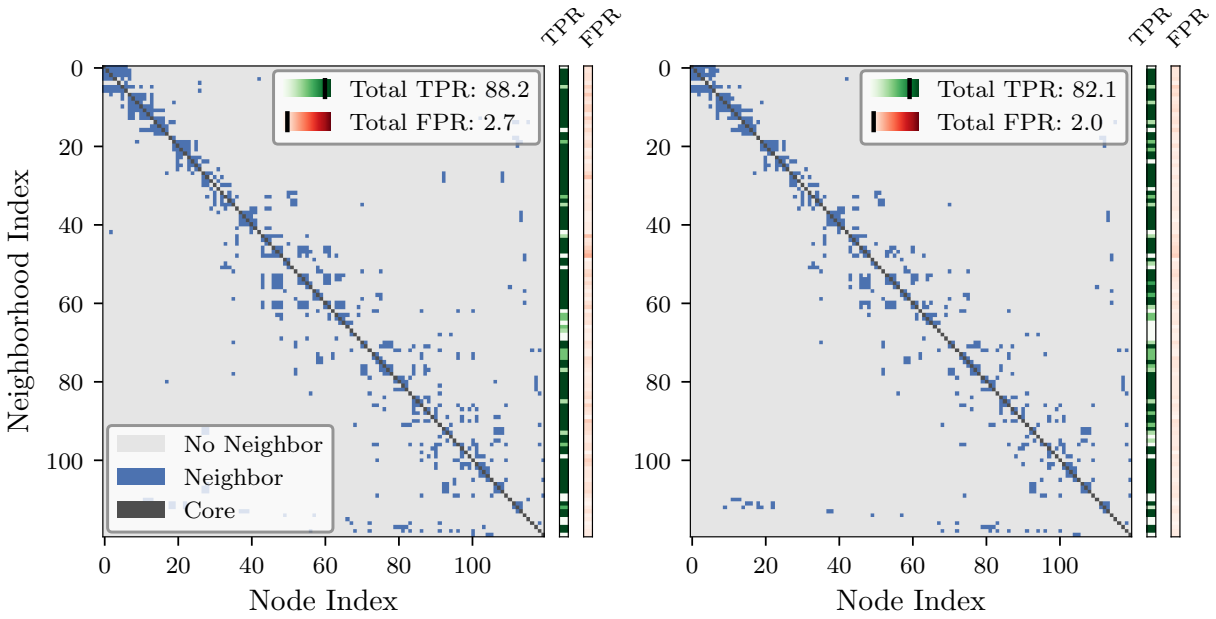


Figure 7.14: Neighborhood matrix reconstruction for the UKPG using a combination of forward selection and feature elimination. Each row corresponds to one neighborhood. The inferred links are compared with the true network, yielding the true positive rate (TPR) and false positive rate (FPR) shown by the vertical bars. TPR runs from 0% to 100%, whereas FPR runs from 0% to 30%. (**left**) Inferred neighborhood matrix after forward selection (Algorithms 2 3). (**right**) Final neighborhood after feature elimination (Algorithm 4)

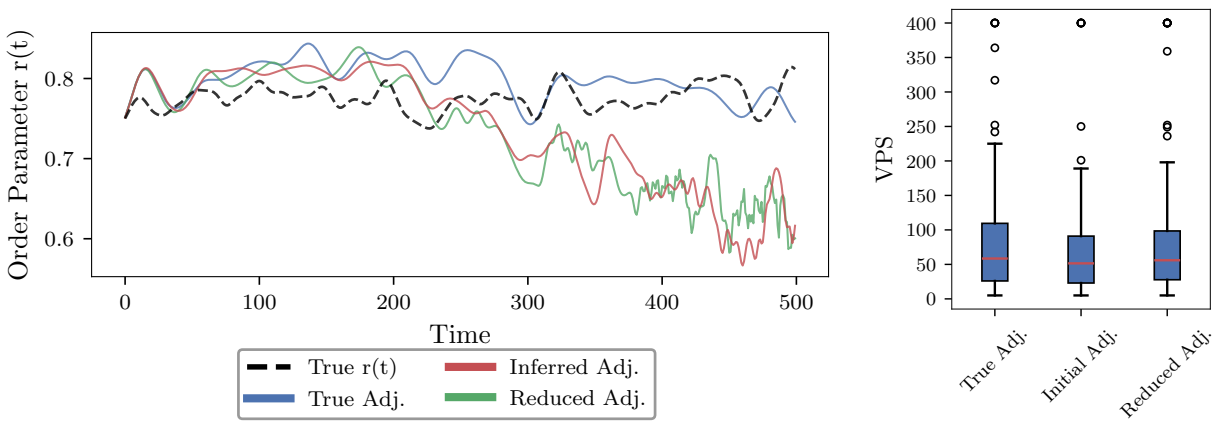


Figure 7.15: Predictions for the different adjacency matrices inferred in Experiment 5. Three NGRC-LS setups are trained using either the true neighborhood matrix or the matrices generated with causal local states (before and after backward elimination) (see Figure 7.3). **(left)** Short-term predictions for the full system evaluated using the Kuramoto order parameter. **(right)** Valid prediction steps for all three models. Each boxplot aggregates the valid prediction steps across all individual nodes, with the median and outliers indicated.

7.5 Improving prediction quality

Two possible reasons for the poor prediction quality on the UKPG are investigated in the next two experiments. Both experiments work with a frequency assortative Kuramoto system (see Section 4.5). As this system is not of a fixed size, the complexity of the problem can be scaled up gradually, which makes isolation of the issues easier.

7.5.1 Experiment 6: core node specific hyperparameters

Since not all neighborhoods are the same, the set of global hyperparameters used in LS is most likely not optimal for all neighborhoods. The system studied in this experiment is a frequency assortative Kuramoto network with $d = 10$ dimensions.

Especially in the Kuramoto system, as some nodes are synchronized, not all nodes are equally hard to forecast. Enabling nonlinearity only for certain, harder to predict nodes could improve the prediction quality a lot while still avoiding dynamic instability. To illustrate this point further, two NGRC-LS models based on a true and an inferred adjacency matrix are trained. To avoid divergence, the NGRC models only include linear terms (VAR mode). As evident from Figure 7.16, the prediction performance is not greatly influenced by whether the adjacency matrix was known or unknown prior to the prediction. More interestingly for this experiment, there seems to be a discrepancy in the prediction quality between different core nodes. Two examples of the predictions for an "easy" and "hard-to-predict" core node are also shown in 7.16.

Since including all model instances in a global hyperparameter search is computationally impossible, this experiment tests whether a hyperparameter search at the wrapper stage holds meaningful results for the hyperparameters of the same instance when combined in a LS setting. Ideally, it would be possible to optimize the complex neighborhoods while retaining an easy model for the easy neighborhoods.

Setup: To test whether an isolated hyperparameter search is possible, an NGRC model is trained in VAR mode as a reference. The examined core node 1 is the one that showed significantly worse performance during testing (see Figure 7.16). To try to improve the predictions, a full grid search is ran for the isolated neighborhood of core node 1. Then an NGRC model is retrained, which again uses the old hyperparameters but with a separate hyperparameter set for the instance prediction core node 1.

The concrete hyperparameters and simulation settings can be extracted from Table A.9.

Results & discussion: The main results of this experiment can be seen in Figure 7.17. The hyperparameter search resulted in a new optimal HP set that includes nonlinearity (orders: $[1] \rightarrow [1, 3]$). As evident from Subfigure 7.17(c), even with the nonlinearity, the prediction does not diverge. However, the prediction quality for the optimized core node did not improve. Interestingly though, the change in the specific model changed the prediction quality of other nodes in the system.

The key takeaway is that an isolated hyperparameter search seems to be inconclusive for how the model will perform if integrated into an ensemble of models in LS. Therefore, for

further experiments the HPs will be shared across all models, as there is no efficient way to optimize the individual hyperparameters in the wrapper step for full system prediction.

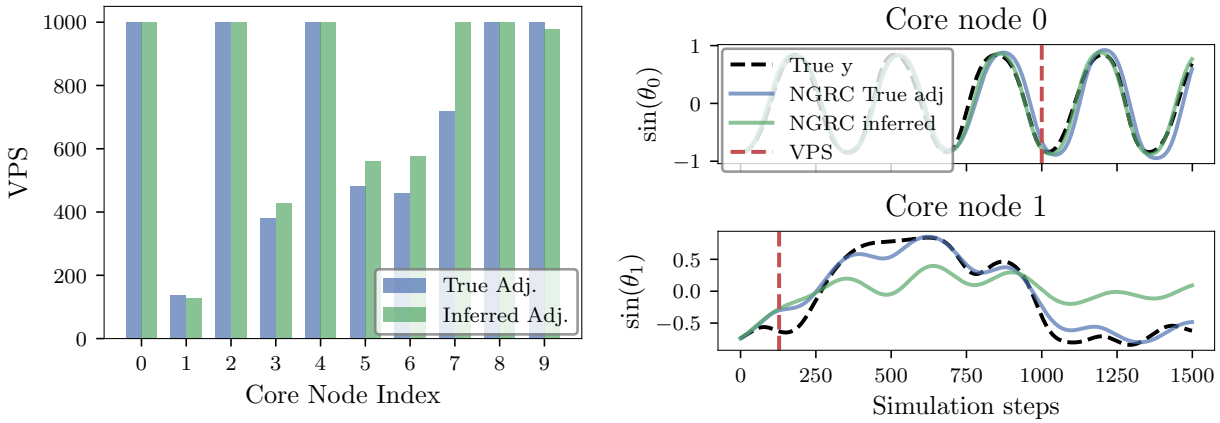


Figure 7.16: Predictions using LS-NGRC for a ten-dimensional, frequency-assortative Kuramoto system. The bar plot shows the number of valid prediction steps for each core node, comparing predictions obtained with the true adjacency matrix versus predictions using the adjacency matrix inferred via causal local states (see Chapter 6). Prediction quality varies considerably across nodes. On the right, example trajectories are shown for one "easy" and one "hard-to-predict" system dimension, illustrating the differences in predictive performance between core nodes.

7.5.2 Experiment 7: improving divergence in NGRC with sine-NGRC & Kuramoto-NGRC

A key limitation of using NGRC-LS is its tendency to produce divergent predictions. Due to this, when forecasting the UK power grid, it was not possible to include nonlinear expansions in the feature vector construction, as doing so consistently led to divergence. Similar behavior can already be observed in simple oscillator systems, such as the divergence illustrated in Figure 7.18, which shows this phenomenon for a three-dimensional coupled Kuramoto system.

Another observation, visible in Figure 7.19, is that prediction errors often originate at a single core node and subsequently propagate through the network. The figure shows the time steps immediately preceding divergence in an NGRC-LS prediction. At timestep 17, the prediction error is roughly uniform across all nodes. Shortly afterward, the prediction begins to deviate at a single core node (node 33), after which the error rapidly spreads throughout the system. NGRC-LS provides no mechanism to contain such instabilities. Furthermore, the construction of the nonlinear feature vector amplifies prediction errors, since small deviations are combined into higher-order polynomial terms.

This behavior of NGRC on oscillator systems is also reported by [57], where Zhang and Cornelius show that NGRC struggles to reconstruct the basins of attraction. In their work, the polynomial NGRC formulation is compared with two alternative variants that use different nonlinear feature compositions. In addition to the polynomial construction proposed in [55], alternative nonlinear feature maps can be obtained by modifying the construction of

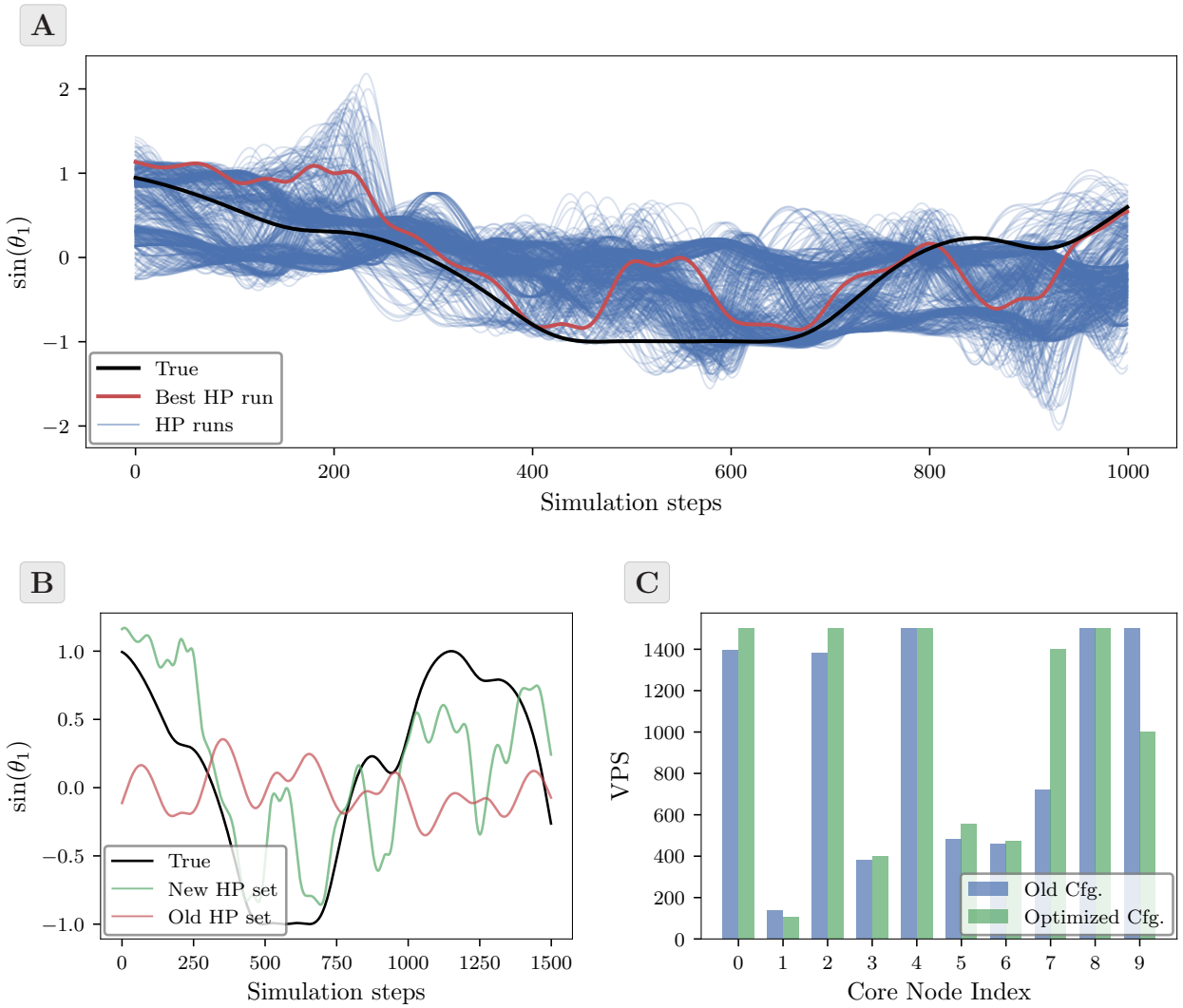


Figure 7.17: Hyperparameter search results for the "hard-to-predict" core node 1 of a 10-d frequency-assortative Kuramoto system. The HP search is conducted as a wrapper experiment with the true neighbors as a neighborhood. **(A)** NGRC predictions for all hyperparameters in a grid search: $k \in \{1, \ldots, 9\}, s \in \{1, 3, 6, 10\}, \mathcal{P} \in \{\{1\}, \{1, 2\}, \{1, 3\}\}, \alpha \in \{0.001, 0.005, 0.01, 0.1, 1\}$. **(B)** Prediction of best HP model compared to old HP model. Configuration details: $k : 4 \to 9$, $s : 5 \to 3$, $\alpha : 0.1 \to 0.001$, orders:[1]→[1,3]. **(C)** Valid prediction steps for each core node. Both models use the old configuration, with the one model using a separate HP set only for the instance predicting core node 1.

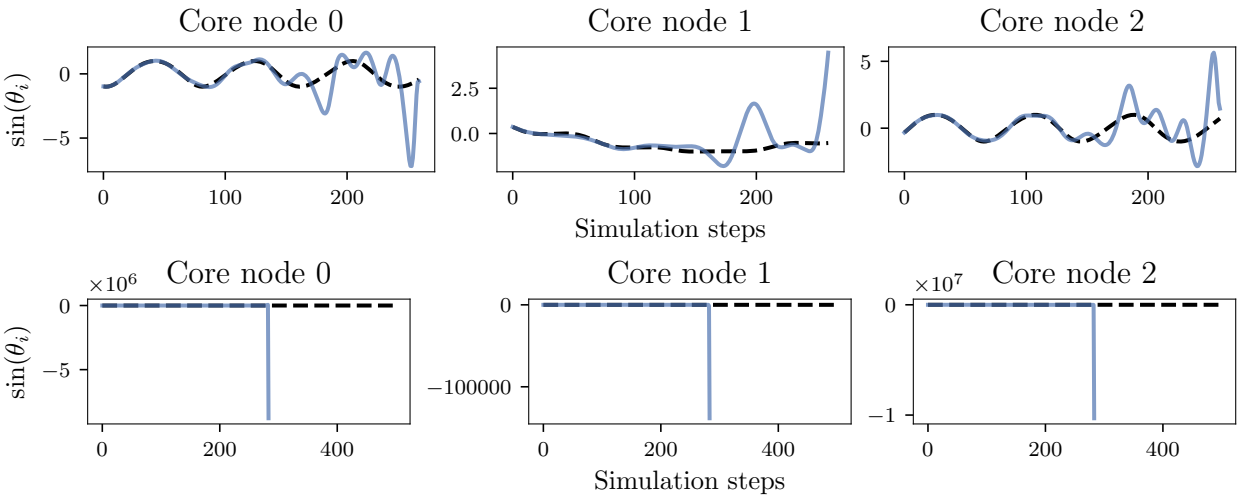


Figure 7.18: Sample diverging predictions generated with NGRC for a three-dimensional frequency-assortative Kuramoto system. The top row shows the time span before divergence, where the prediction is initially accurate but then gradually deteriorates. The bottom row shows the rapid divergence that occurs after a certain number of time steps.

$\mathbb{O}_{\text{nonlin},t}$ from the linear features $\mathbb{O}_{\text{lin},t}$. In particular, two variants are proposed to improve the standard NGRC formulation:

1. **Sine-NGRC**: The nonlinear feature vector is built from polynomials consisting of $\sin(\theta_i)$ and $\cos(\theta_i)$ for monomial θ_i.
2. **Kuramoto-NGRC**: The nonlinear expansion instead incorporates the coupling nonlinearity of the Kuramoto system by including interaction terms of the form $\sin(\theta_i - \theta_j)$ for connected node pairs (i, j).

Both NGRC variants have nonlinearities that are closer to the true equations of a Kuramoto system, which is expected to result in a better performance. Additionally, both provide a natural containment for divergence, as the nonlinear features are bounded within $(-1, 1)$. Consequently, a diverging core node trajectory would result only in faster oscillations of the features rather than unbounded growth of the predicted values.

Both variants were tested on the same three-dimensional oscillator system in order to evaluate their stability. As evident from Figure 7.20, the prediction is now stable even when higher-order terms or more delayed features are included in the feature vector. A detailed analysis of the mechanisms that lead to divergence in polynomial NGRC is beyond the scope of this thesis. However, a brief discussion is provided in Appendix A.2.

The parameters used in the simulations of this experiment are shown in Table A.10.

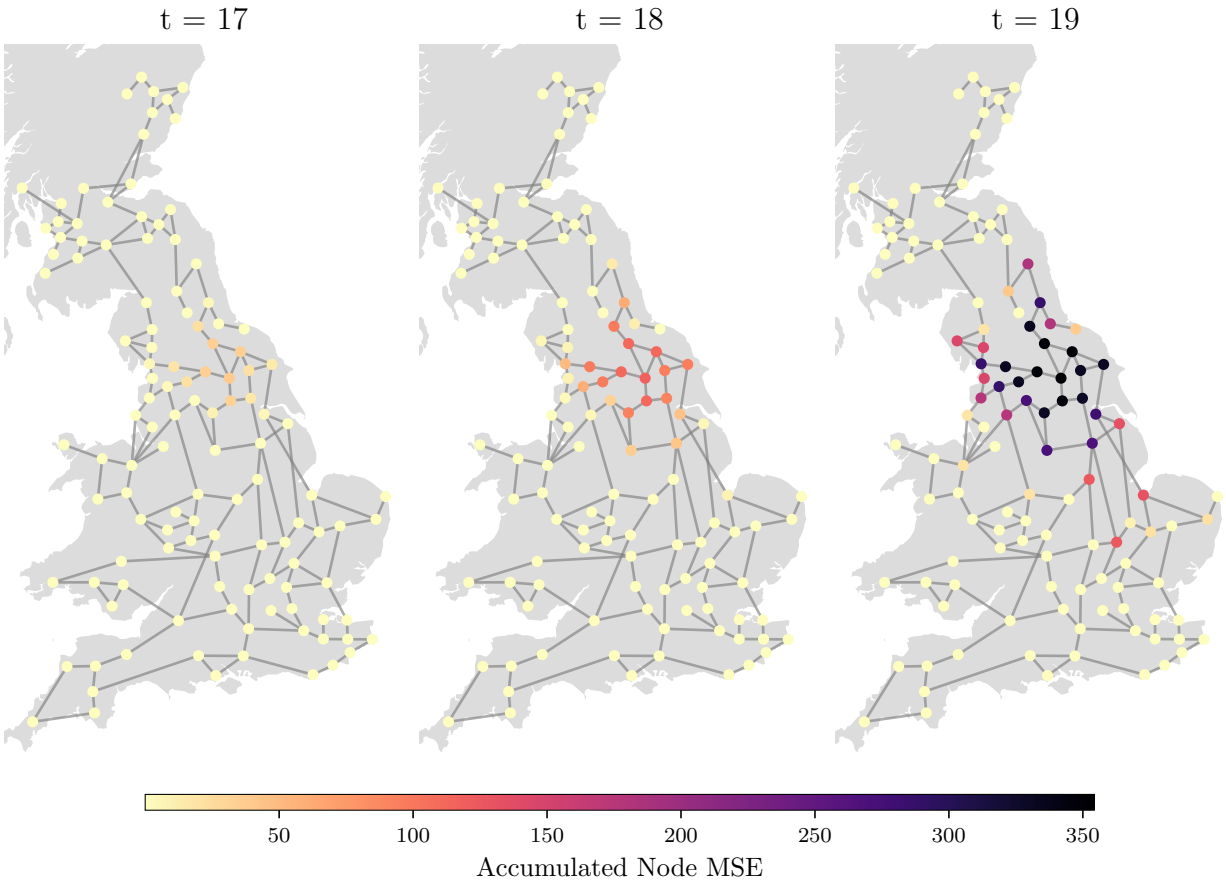


Figure 7.19: Accumulated MSE for each core node during a divergent prediction pf the UK power grid with NGRC. The plots show the timesteps ($t = 17, 18, 19$) immediately before the divergence occurs. The divergence starts from one "bad" core node and then rapidly spreads to the nearby nodes, propagating through the network.

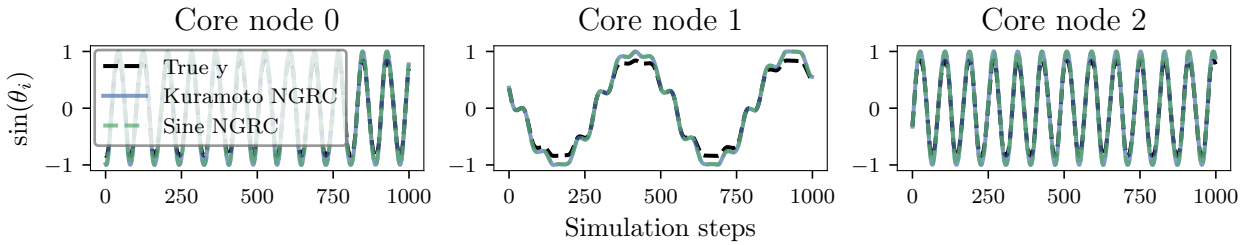


Figure 7.20: Sample predictions for a three-dimensional frequency-assortative Kuramoto system generated with two NGRC variants: Sine-NGRC and Kuramoto-NGRC. Both modify the nonlinear feature expansion of NGRC. Sine-NGRC applies $\sin(\cdot)$ and $\cos(\cdot)$ to the monomials, whereas Kuramoto-NGRC combines monomials in a form analogous to the Kuramoto interaction terms ($\sin(\theta_j - \theta_i)$). In contrast to standard NGRC (see Fig. 7.18), the predictions no longer diverge.

7.6 Results

With the final algorithm established, this section presents results obtained with the causal local states (CLS) implementation. A detailed description of the algorithm is provided in Chapter 6. The systems studied include a concatenated L63-Rössler system, the Lorenz 96 (L96) system, and the higher-order Kuramoto model of the UK power grid.

7.6.1 Concatenated L63-Rössler system

The coupled L63-Rössler system served as an easy model system used to illustrate shortcomings of some approaches to the network inference (section 7.2). The causal local states algorithm was developed specifically with those shortcomings in mind and should be able to correctly separate the two non-interacting attractors. To reproduce the situation described in experiment 2, transfer entropy is used to calculate the causal matrix. The resulting matrix is shown in 7.21(a). Subfigure 7.21(b) demonstrates that the spurious neighbors initially introduced in the neighborhoods of R_x and R_y are successfully removed during the feature removal step. The short-term prediction results are shown in Subfigure 7.21(c). The algorithm achieves good reproduction of the short-term behavior of both the Rössler and the L63 trajectories. Simulation and model parameters are summarized in Table A.11

7.6.2 Lorenz 96 system

Due to its simple coupling structure, the Lorenz 96 system (described in more detail in Section 4.4) provides a convenient test case for evaluating the algorithm in a higher-dimensional setting. The filter step was done with transfer entropy, and NGRC was used as the wrapper model. The reconstruction results are provided in Figure 7.22, where it is evident that the transfer entropy matrix captures the periodic coupling structure of the system well. The inferred and reduced matrices both consistently include more neighbors than the true system equations suggest. According to Takens' embedding theorem, information about the core node is also present in other nodes of the system. It is therefore not surprising that nodes that are not direct neighbors but still located near the core node can improve prediction performance. However, this can also be a sign of overfitting. The prediction results are shown in Figures 7.23 and 7.24. Notably, causal local states outperforms the NGRC-LS model that uses the true adjacency matrix. The simulation and model parameters are given in Table A.12.

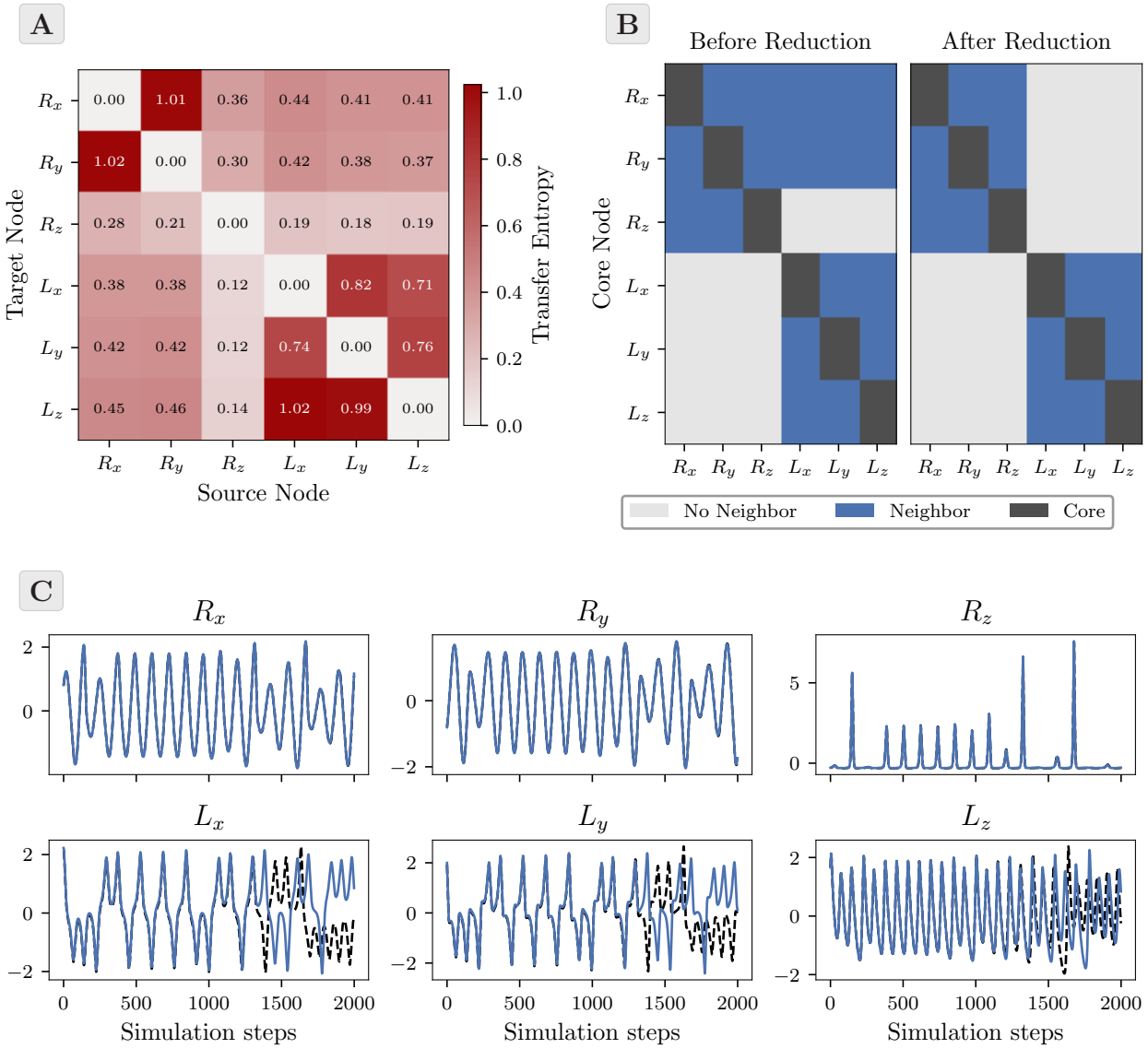


Figure 7.21: Causal local states results for a concatenated system of a Lorenz attractor and a Rössler attractor with no interaction between the attractors. **(A)** The transfer entropy (TE) matrix. **(B)** Inferred neighborhood matrices from the TE-Matrix before and after the dimensionality reduction step. **(C)**. Successful short-term predictions for each dimension of the system.

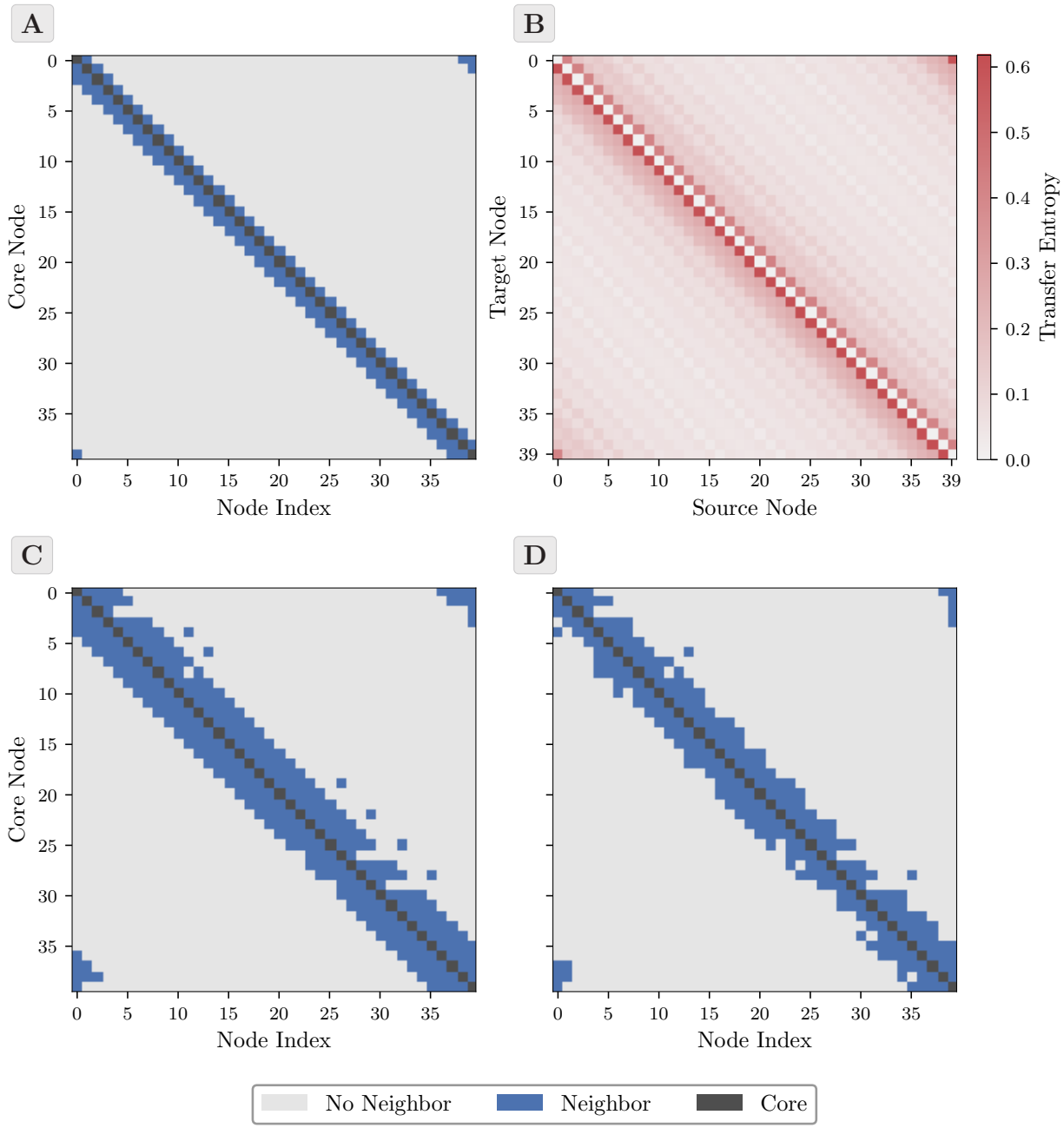


Figure 7.22: Reconstruction results for a 40-d Lorenz 96 with a forcing $F = 40$ achieved with causal local states using NGRC as a wrapper and transfer entropy as a filter function. **(A)** True adjacency (by system equations) of the L96 system as reference. **(B)** Transfer entropy matrix with $\tau = 1$, $n_{bins} = 30$. (C-D) Inferred neighborhood matrix from the TE-Matrix before **(C)** and after the dimensionality reduction step **(D)**.

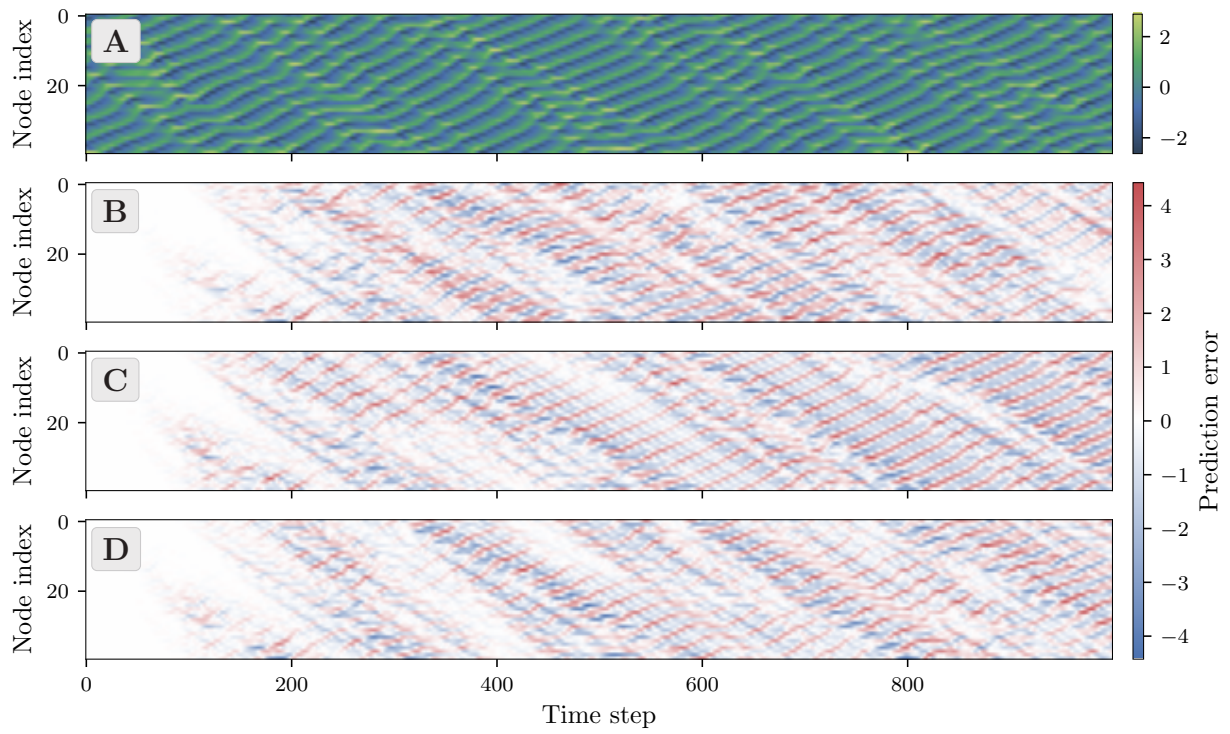


Figure 7.23: Prediction results for a 40-d Lorenz 96 with a forcing $F = 40$ achieved with causal local states using NGRC-LS with the adjacency matrices shown in Figure 7.22. The predictions are shown as the difference to the true trajectory. **(A)** Simulated system trajectory as reference **(B)** Using true adjacency (by system equations). **(B-C)** Using inferred neighborhood matrices before **(C)** and after the dimensionality reduction step **(D)**.

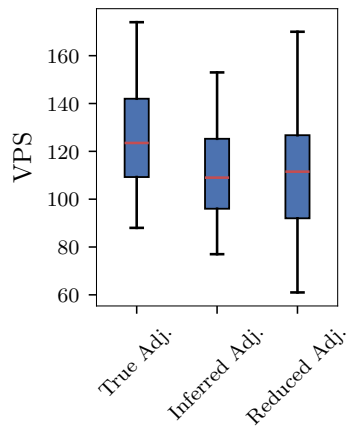


Figure 7.24: Valid prediction steps for three NGRC-LS models predicting an 40-d L96 system with $F = 5$. One model uses the true network, while the others use the matrices generated with causal local states (before and after the feature elimination step). Each boxplot aggregates the valid prediction steps across all individual nodes, with the median and outliers indicated.

7.6.3 UK power grid

The final dataset analyzed with CLS is the higher-order Kuramoto model of the UK power grid. Both NGRC variants (Sine-NGRC & Kuramoto-NGRC) were used. Since these models apply a trigonometric function as part of their nonlinear feature expansion step, it does not make sense to apply the sin(.) preprocessing to the data. Similar to Li et al., the model is trained to predict the difference $\Delta\theta = (\theta(t) - \theta(t-1))$ [4]. The simulation details and model parameters are listed in Table A.13.

An example of the neighborhood inference step for a single core node is seen in Figure 7.26. Remarkably, the algorithm perfectly identifies the true neighbors and removes all spurious connections in the process. The full system reconstruction for both Sine-NGRC and Kuramoto-NGRC is shown in Figure 7.27. Sine-NGRC does not outperform the reconstruction quality provided by normal NGRC (see Figure 7.14). For Kuramoto-NGRC, however, the initial neighborhood search finds a neighborhood configuration that includes all true neighbors of every core node. Although this initial step introduces a substantial number of false positives, most of them are removed during the subsequent backward elimination stage. This yields an almost perfect reconstruction of the true network.

The resulting prediction performance is evaluated in Figures 7.25 and 7.28. Consistent with the reconstruction results, Sine-NGRC yields poor predictive performance. This is surprising given that the polynomial monomials of $\sin(\theta_i)$ and $\cos(\theta_i)$ implicitly contain the Kuramoto coupling terms via the identity $\sin(\theta_i)\cos(\theta_j) - \cos(\theta_i)\sin(\theta_j) = \sin(\theta_i - \theta_j)$ [Th. 4.3.16][66], suggesting that the relevant nonlinearities are in principle representable by the Sine-NGRC feature library. In contrast, the Kuramoto-NGRC model trained on the CLS-inferred neighborhoods achieves prediction quality comparable to that obtained using the true network structure. This is a notable result, as both the inference and prediction processes are entirely data-driven and do not rely on prior knowledge of the underlying causal structure.

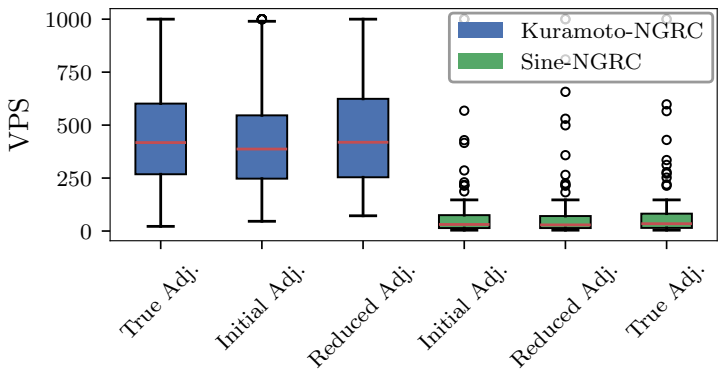


Figure 7.25: Valid prediction steps for sine and Kuramoto NGRC-LS models predicting the UK power grid. The models use the true network, and the matrices generated with causal local states (before and after the feature elimination step). Each boxplot aggregates the valid prediction steps across all individual nodes, with the median and outliers indicated.

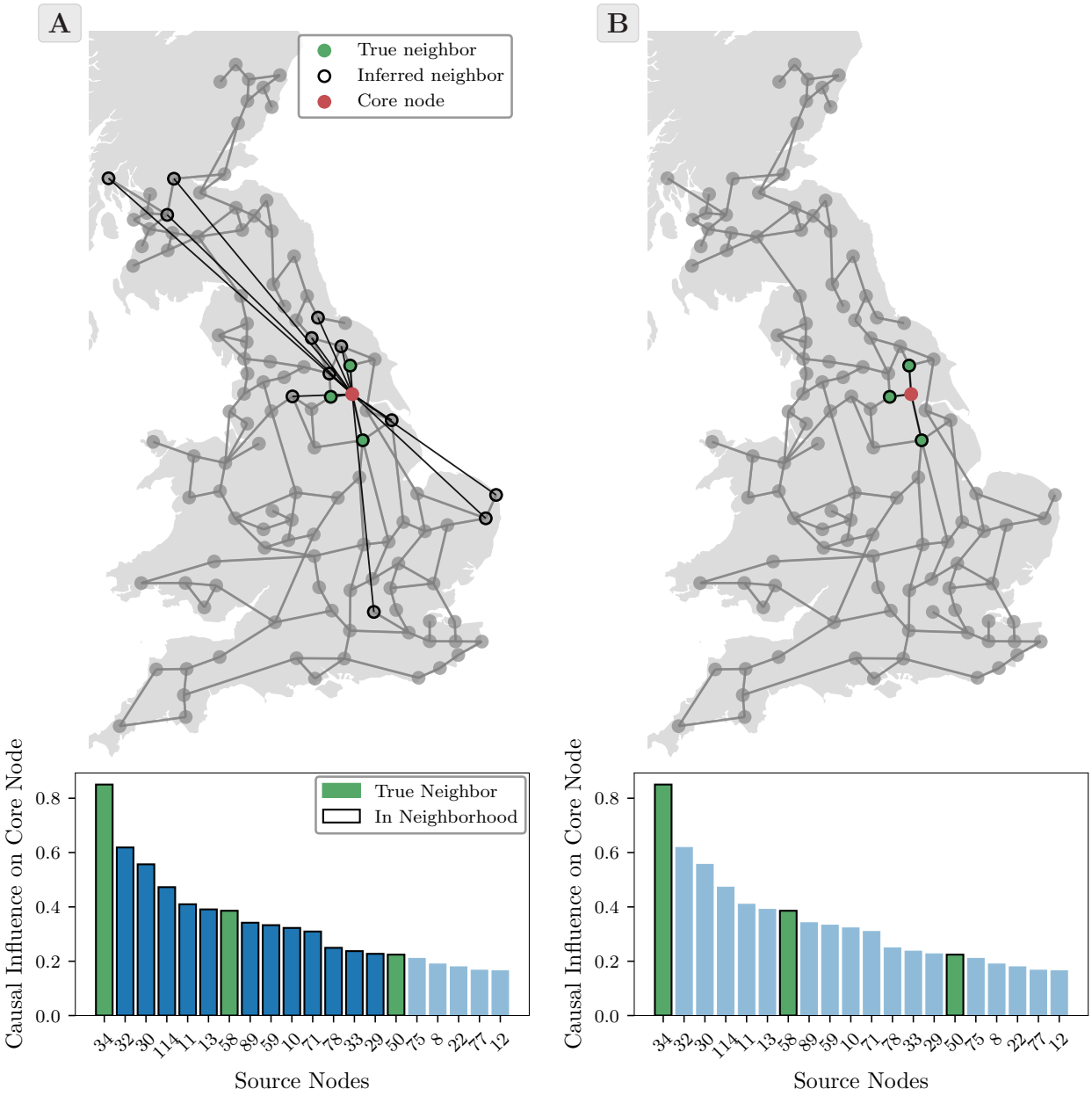


Figure 7.26: Causal local states neighborhood reconstruction example on the UK power grid using Kuramoto-NGRC as a wrapper model. The core node, corresponding true neighbors and the inferred neighborhood found by CLS are highlighted together with the causal influence calculated with CCM. **(A)** The neighborhood found after the filter & selection steps of CLS. **(B)** Approximate causal neighborhood found after applying backward feature elimination.

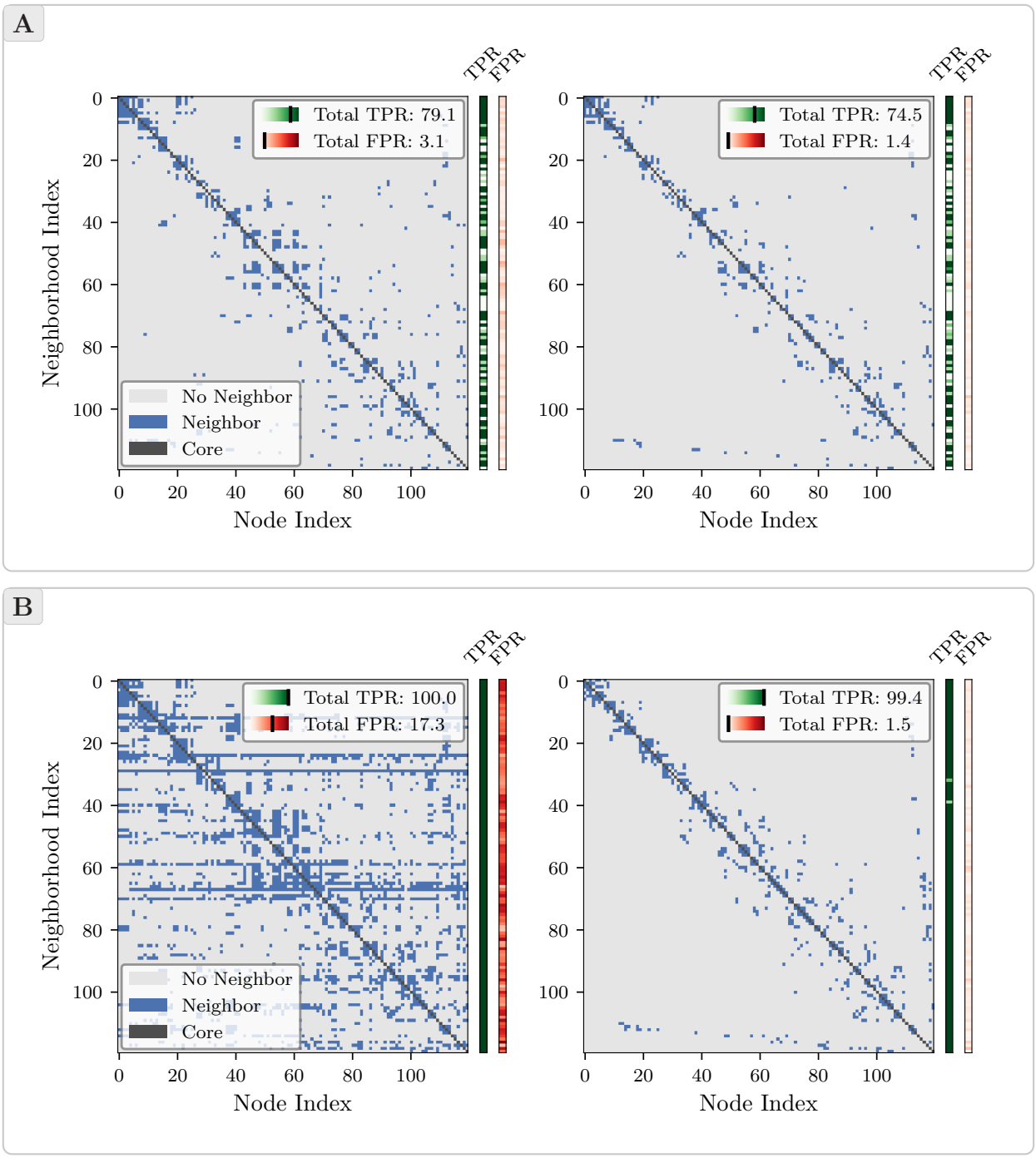


Figure 7.27: Neighborhood matrix reconstruction for the UKPG using a combination of forward selection and feature elimination. Sine-NGRC **(A)** and Kuramoto-NGRC **(B)** are used as wrapper models. The neighborhood matrix before (left) and after (right) feature removal is shown for both models, respectively. The inferred links are compared with the true network, yielding the true positive rate (TPR) and false positive rate (FPR) shown by the vertical bars. TPR runs from 0% to 100%, whereas FPR runs from 0% to 30%.

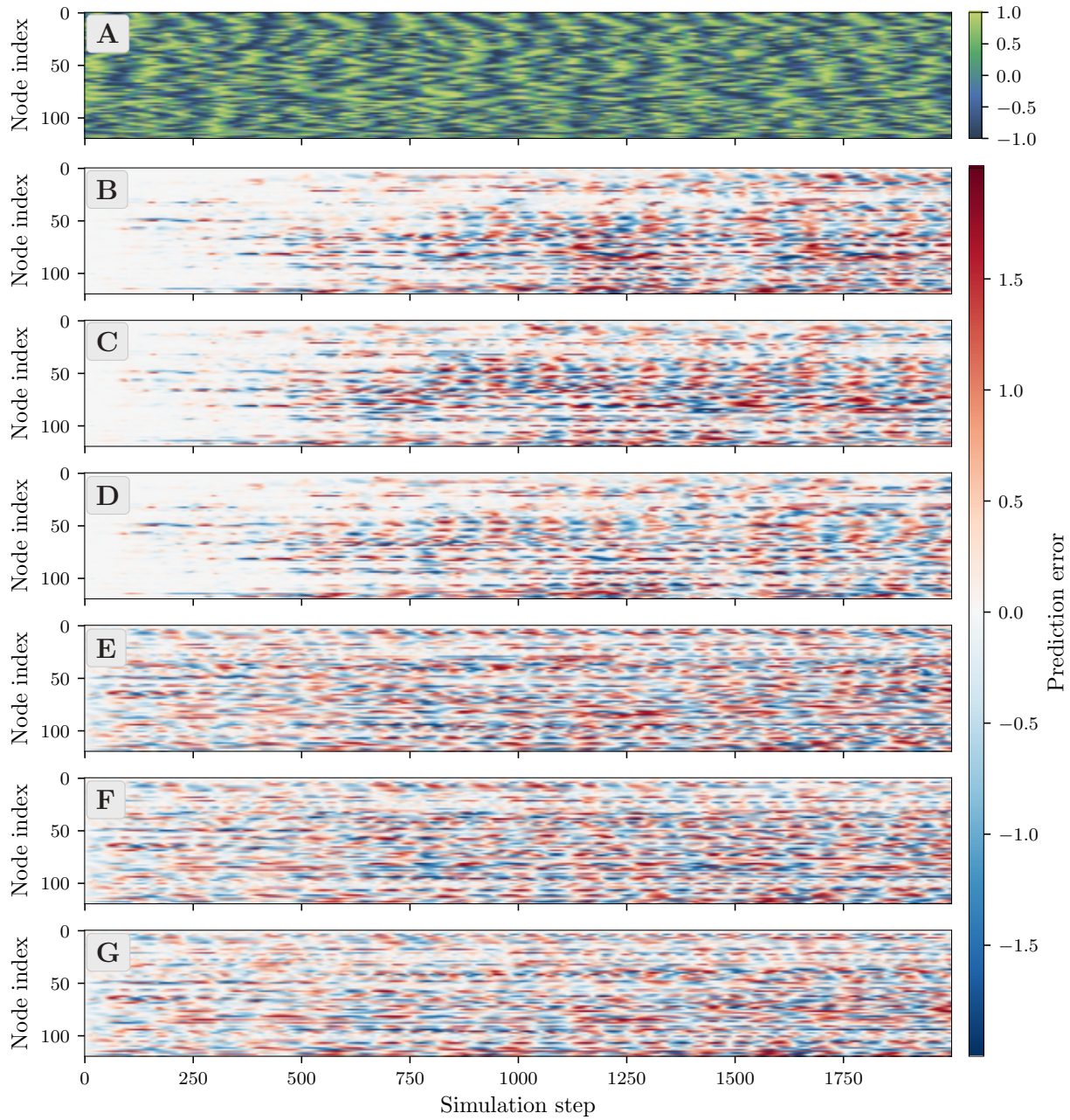


Figure 7.28: Prediction results for a higher-order Kuramoto model of the UK power grid using Sine-NGRC and Kuramoto-NGRC with the adjacency matrices shown in Figure 7.27. The predictions are shown as the difference from the true trajectory. **(A)** Simulated system trajectory as reference **(B-D)** Kuramoto-NGRC results using the true adjacency, initially inferred, and reduced neighborhood matrices. **(E-F)** Sine-NGRC results using the true adjacency, initially inferred, and reduced neighborhood matrices.

CHAPTER 8
SUMMARY AND OUTLOOK

This thesis introduced the method of causal local states (CLS). The CLS framework extends generalized local states (GLS) [2] and local states (LS) [1] by removing the requirement of prior knowledge about the underlying network structure. While GLS already introduced the idea of similarity measures to handle this, CLS introduces a wrapper based approach using causal measures to generalize this to more complex networks. By incorporating ideas from causal discovery into the prediction task, CLS represents a step toward more explainable approaches in machine learning.

The CLS framework was successfully applied to several benchmark systems. In the combined attractor example, CLS was able to separate the L63 and Rössler attractors and accurately predict both systems. Furthermore, CLS effectively captured the periodic network structure of the L96 system and achieved accurate forecasts. Notably, the prediction quality is on par with that of an NGRC-LS model provided with the true adjacency matrix.

For the higher-order Kuramoto model of the UK power grid, CLS in combination with a standard NGRC model did not yield accurate forecasts. However, when paired with a model specifically tailored to the task (Kuramoto-NGRC), CLS successfully inferred an almost exact reconstruction of the true adjacency matrix and achieved short-term forecasting performance comparable to a model with access to the true network structure. This result highlights an important lesson for CLS: as in any machine learning application, the choice of an appropriate predictive model remains crucial. Generally, model and causal measure selection must be made in accordance with the characteristics of the dataset, as the overall performance of the method depends on this choice.

A key strength of the CLS framework is that the inference of each local neighborhood is performed independently. This property ensures scalability to high-dimensional systems and makes CLS particularly attractive for large-scale applications.

Despite these promising results, several open challenges remain. Most notably, CLS has not yet been applied to real-world datasets. Potential application include financial time series, power grid dynamics, or weather forecasting, domains where both predictive performance and interpretability are of high importance.

In its current form, the inference procedure is restricted to identifying causal neighbors. Extending the framework to include the selection of relevant temporal features could provide

deeper insights into system dynamics and further improve predictive performance. Incorporating tree-based NGRC models with built-in feature selection capabilities could be a promising direction in this context [67].

Another challenge concerns hyperparameter optimization. The CLS framework involves multiple local models, each of which may require its own set of hyperparameters. The current approach is to use a shared hyperparameter configuration across all neighborhoods. However, this is unlikely to be optimal. As demonstrated in Experiment 6 (see Subsection 7.5.1), hyperparameter optimization at the wrapper level does not yield satisfactory results. Developing efficient strategies to optimize hyperparameters without exhaustive search remains an open challenge.

In conclusion, this thesis developed a novel framework that enables scalable prediction of high-dimensional dynamical systems while simultaneously inferring an approximation of the underlying causal structure. The results demonstrate the strong potential of CLS as a tool for combining prediction and interpretability in complex systems.

BIBLIOGRAPHY

[1] Jaideep Pathak et al. "Model-Free Prediction of Large Spatiotemporally Chaotic Systems from Data: A Reservoir Computing Approach". In: *Phys. Rev. Lett.* 120 (2 Jan. 2018), p. 024102. DOI: 10.1103/PhysRevLett.120.024102. URL: https://link.aps.org/doi/10.1103/PhysRevLett.120.024102.

[2] Sebastian Baur and Christoph Räth. "Predicting high-dimensional heterogeneous time series employing generalized local states". In: *Physical Review Research* 3.2 (June 2021). ISSN: 2643-1564. DOI: 10.1103/physrevresearch.3.023215. URL: http://dx.doi.org/10.1103/PhysRevResearch.3.023215.

[3] Alessio Zanga, Elif Ozkirimli, and Fabio Stella. "A Survey on Causal Discovery: Theory and Practice". In: *International Journal of Approximate Reasoning* 151 (Dec. 2022), pp. 101–129. ISSN: 0888-613X. DOI: 10.1016/j.ijar.2022.09.004. URL: http://dx.doi.org/10.1016/j.ijar.2022.09.004.

[4] Xin Li et al. "Higher-order Granger reservoir computing: simultaneously achieving scalable complex structures inference and accurate dynamics prediction". In: *Nature Communications* 15.1 (2024), p. 2506. DOI: 10.1038/s41467-024-46852-1. URL: https://doi.org/10.1038/s41467-024-46852-1.

[5] Keshav Srinivasan et al. "Parallel Machine Learning for Forecasting the Dynamics of Complex Networks". In: *Phys. Rev. Lett.* 128 (16 Apr. 2022), p. 164101. DOI: 10.1103/PhysRevLett.128.164101. URL: https://link.aps.org/doi/10.1103/PhysRevLett.128.164101.

[6] Kuei-Jan Chu, Nozomi Akashi, and Akihiro Yamamoto. "Incorporating coupling knowledge into echo state networks for learning spatiotemporally chaotic dynamics". In: *Chaos: An Interdisciplinary Journal of Nonlinear Science* 35.9 (Sept. 2025), p. 093138. ISSN: 1054-1500. DOI: 10.1063/5.0273343. eprint: https://pubs.aip.org/aip/cha/article-pdf/doi/10.1063/5.0273343/20702902/093138_1_5.0273343.pdf. URL: https://doi.org/10.1063/5.0273343.

[7] Judea Pearl. *Causality.* 2nd ed. Cambridge University Press, 2009.

[8] Robin J. Wilson. *Introduction to Graph Theory.* 4th ed. Addison-Wesley, 1998.

[9] Darij Grinberg. *An introduction to graph theory.* 2025. arXiv: 2308.04512 [math.HO]. URL: https://arxiv.org/abs/2308.04512.

[10] Gary Chartrand and Ping Zhang. *A FIRST COURSE IN GRAPH THEORY*. Dover publications, 2012.

[11] Philip Dawid. *What Is a Causal Graph?* 2024. arXiv: 2402.09429 [math.ST]. URL: https://arxiv.org/abs/2402.09429.

[12] J. Runge. "Causal network reconstruction from time series: From theoretical assumptions to practical estimation". In: *Chaos: An Interdisciplinary Journal of Nonlinear Science* 28.7 (July 2018), p. 075310. ISSN: 1054-1500. DOI: 10.1063/1.5025050. eprint: https://pubs.aip.org/aip/cha/article-pdf/doi/10.1063/1.5025050/13283543/075310_1_online.pdf. URL: https://doi.org/10.1063/1.5025050.

[13] Michael Eichler. "Graphical modelling of multivariate time series". In: *Probability Theory and Related Fields* 153.1–2 (Feb. 2011), pp. 233–268. ISSN: 1432-2064. DOI: 10.1007/s00440-011-0345-8. URL: http://dx.doi.org/10.1007/s00440-011-0345-8.

[14] Charles K. Assaad, Emilie Devijver, and Eric Gaussier. "Entropy-Based Discovery of Summary Causal Graphs in Time Series". In: *Entropy* 24.8 (2022), p. 1156. DOI: 10.3390/e24081156.

[15] Robin Delabays et al. "Hypergraph reconstruction from dynamics". In: *Nature Communications* 16.1 (Mar. 2025). ISSN: 2041-1723. DOI: 10.1038/s41467-025-57664-2. URL: http://dx.doi.org/10.1038/s41467-025-57664-2.

[16] C. W. J. Granger. "Investigating Causal Relations by Econometric Models and Cross-spectral Methods". In: *Econometrica* 37.3 (1969), pp. 424–438. ISSN: 00129682, 14680262. URL: http://www.jstor.org/stable/1912791 (visited on 12/29/2025).

[17] Thomas Schreiber. "Measuring Information Transfer". In: *Phys. Rev. Lett.* 85 (2 July 2000), pp. 461–464. DOI: 10.1103/PhysRevLett.85.461. URL: https://link.aps.org/doi/10.1103/PhysRevLett.85.461.

[18] George Sugihara et al. "Detecting Causality in Complex Ecosystems". In: *Science* 338.6106 (2012), pp. 496–500. DOI: 10.1126/science.1227079. eprint: https://www.science.org/doi/pdf/10.1126/science.1227079. URL: https://www.science.org/doi/abs/10.1126/science.1227079.

[19] Peter Spirtes, Clark Glymour, and Richard Scheines. *Causation, Prediction, and Search*. Vol. 81. Jan. 1993. ISBN: 978-1-4612-7650-0. DOI: 10.1007/978-1-4612-2748-9.

[20] Richard E. Neapolitan and Xia. Jiang. *Probabilistic methods for financial and marketing informatics*. eng. San Fransisco, CA: Morgan Kaufmann Publishers, 2007. ISBN: 9780123704771. URL: http://www.loc.gov/catdir/toc/ecip077/2006039447.html.

[21] Ali Shojaie and Emily B. Fox. "Granger Causality: A Review and Recent Advances". In: *Annual Review of Statistics and Its Application* 9.Volume 9, 2022 (2022), pp. 289–319. ISSN: 2326-831X. DOI: https://doi.org/10.1146/annurev-statistics-040120-010930. URL: https://www.annualreviews.org/content/journals/10.1146/annurev-statistics-0401

[22] Claude Elwood Shannon. "A Mathematical Theory of Communication". In: *The Bell System Technical Journal* 27 (1948), pp. 379–423. URL: http://plan9.bell-labs.com/cm/ms/what/shannonday/shannon1948.pdf (visited on 04/22/2003).

[23] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory 2nd Edition (Wiley Series in Telecommunications and Signal Processing)*. Wiley-Interscience, July 2006. ISBN: 0471241954.

[24] Lionel Barnett, Adam B. Barrett, and Anil K. Seth. "Granger Causality and Transfer Entropy Are Equivalent for Gaussian Variables". In: *Phys. Rev. Lett.* 103 (23 Dec. 2009), p. 238701. DOI: 10.1103/PhysRevLett.103.238701. URL: https://link.aps.org/doi/10.1103/PhysRevLett.103.238701.

[25] Raul Vicente et al. "Transfer entropy–a model-free measure of effective connectivity for the neurosciences". In: *J. Comput. Neurosci.* 30.1 (Feb. 2011), pp. 45–67. ISSN: 0929-5313. DOI: 10.1007/s10827-010-0262-3. URL: https://doi.org/10.1007/s10827-010-0262-3.

[26] M. Mynter. "Evaluation and Extension of the Transfer Entropy Calculus for the Measurement of Information Flows Between Futures Time Series During the COVID-19 Pandemic". Unpublished; Information about the project: https://github.com/maxmynter/transfe entropy-toolbox. Master's thesis. Ludwig-Maximilians-Universität München, 2021.

[27] Weng Siew Lam et al. "Bibliometric Analysis of Granger Causality Studies". In: *Entropy* 25.4 (2023). ISSN: 1099-4300. DOI: 10.3390/e25040632. URL: https://www.mdpi.com/1099-4300/25/4/632.

[28] Anatole Katok and Boris Hasselblatt. *Introduction to the Modern Theory of Dynamical Systems*. Encyclopedia of Mathematics and its Applications. Cambridge University Press, 1995.

[29] Ethan Deyle and George Sugihara. "Generalized Theorems for Nonlinear State Space Reconstruction". In: *PloS one* 6 (Mar. 2011), e18295. DOI: 10.1371/journal.pone.0018295.

[30] Floris Takens. "Detecting strange attractors in turbulence". In: *Dynamical Systems and Turbulence, Warwick 1980*. Ed. by David Rand and Lai-Sang Young. Berlin, Heidelberg: Springer Berlin Heidelberg, 1981, pp. 366–381. ISBN: 978-3-540-38945-3.

[31] Antoni A. Kosinski. *Differential manifolds*. eng. Pure and applied mathematics v. 138. Boston: Academic Press, 1993. ISBN: 0-12-421850-4. URL: http://www.sciencedirect.com/science/book/9780124218505.

[32] Matthew B. Kennel, Reggie Brown, and Henry D. I. Abarbanel. "Determining embedding dimension for phase-space reconstruction using a geometrical construction". In: *Phys. Rev. A* 45 (6 Mar. 1992), pp. 3403–3411. DOI: 10.1103/PhysRevA.45.3403. URL: https://link.aps.org/doi/10.1103/PhysRevA.45.3403.

[33] A. M. Fraser and H. D. Swinney. "Independent coordinates for strange attractors from mutual information". In: *Physical Review A* 33.2 (1986), pp. 1134–1140. DOI: 10.1103/PhysRevA.33.1134.

[34] Richard Fitzpatrick. *The definition of chaos*. https://farside.ph.utexas.edu/teaching/329/lectures/node57.html. Lecture notes, University of Texas at Austin. 2006.

[35] J. C. Sprott. *Chaos and Time-Series Analysis*. Oxford: Oxford University Press, 2003. ISBN: 9780198508403.

[36] Steven H. Strogatz. *Nonlinear Dynamics and Chaos: With Applications to Physics, Biology, Chemistry, and Engineering*. 2nd ed. Chapter 9: "Lyapunov Exponents". Boulder, CO: Westview Press, 2015. ISBN: 978-0-8133-4910-7.

[37] Michael T. Rosenstein, James J. Collins, and Carlo J. De Luca. "A practical method for calculating largest Lyapunov exponents from small data sets". In: *Physica D:*

Nonlinear Phenomena 65.1 (1993), pp. 117–134. ISSN: 0167-2789. DOI: https://doi.org/10.1016/0167-2789(93)90009-P. URL: https://www.sciencedirect.com/science/article/pii/016727899390009P.

[38] John K. Hunter. *Dynamical systems and ODEs*. MATH 207A course notes, Ch. 1, p. 10. 2023. URL: https://www.math.ucdavis.edu/~hunter/m207a/ch1.pdf.

[39] C. Runge. "Ueber die numerische Auflösung von Differentialgleichungen". In: *Mathematische Annalen* 46 (1895), pp. 167–178. DOI: 10.1007/BF01446807.

[40] W. Kutta. "Beitrag zur näherungsweisen Integration totaler Differentialgleichungen". In: *Zeitschrift für Mathematik und Physik* 46 (1901), pp. 435–453.

[41] Peter Albrecht. "The Runge–Kutta Theory in a Nutshell". In: *SIAM Journal on Numerical Analysis* 33.5 (Oct. 1996), pp. 1712–1735. ISSN: 0036-1429. DOI: 10.1137/S0036142995287847.

[42] Edward N. Lorenz. "Deterministic Nonperiodic Flow". In: *Journal of the Atmospheric Sciences* 20.2 (1963), pp. 130–141. DOI: 10.1175/1520-0469(1963)020<0130:DNF>2.0.CO;2.

[43] Otto E. Rössler. "An Equation for Continuous Chaos". In: *Physics Letters A* 57.5 (1976), pp. 397–398. DOI: 10.1016/0375-9601(76)90101-8.

[44] Edward N. Lorenz. "Predictability: A Problem Partly Solved". In: *Proceedings of the Seminar on Predictability*. Vol. 1. Reading, UK: European Centre for Medium-Range Weather Forecasts, 1996, pp. 1–18.

[45] Norbert Marwan, Jürgen Kurths, and Saskia Foerster. "Analysing spatially extended high-dimensional dynamics by recurrence plots". In: *Physics Letters A* 379.10 (2015), pp. 894–900. ISSN: 0375-9601. DOI: https://doi.org/10.1016/j.physleta.2015.01.013. URL: https://www.sciencedirect.com/science/article/pii/S0375960115000742.

[46] Yoshiki Kuramoto. *Chemical Oscillations, Waves, and Turbulence*. Vol. 19. Springer Series in Synergetics. Berlin: Springer, 1984. DOI: 10.1007/978-3-642-69689-3.

[47] Juan A. Acebrón et al. "The Kuramoto model: A simple paradigm for synchronization phenomena". In: *Rev. Mod. Phys.* 77 (1 Apr. 2005), pp. 137–185. DOI: 10.1103/RevModPhys.77.137. URL: https://link.aps.org/doi/10.1103/RevModPhys.77.137.

[48] Per Sebastian Skardal, Juan G. Restrepo, and Edward Ott. "Frequency assortativity can induce chaos in oscillator networks". In: *Phys. Rev. E* 91 (6 June 2015), p. 060902. DOI: 10.1103/PhysRevE.91.060902. URL: https://link.aps.org/doi/10.1103/PhysRevE.91.060902.

[49] Tom Patterson, Nathaniel Vaughn Kelso, et al. *Natural Earth Data*. https://www.naturalearthdata.com/. Free vector and raster map data at 1:10m, 1:50m, and 1:110m scales. 2009–2025.

[50] Jaideep Pathak et al. "Using machine learning to replicate chaotic attractors and calculate Lyapunov exponents from data". In: *Chaos: An Interdisciplinary Journal of Nonlinear Science* 27.12 (Dec. 2017). ISSN: 1089-7682. DOI: 10.1063/1.5010300. URL: http://dx.doi.org/10.1063/1.5010300.

[51] Herbert Jaeger and Harald Haas. "Harnessing Nonlinearity: Predicting Chaotic Systems and Saving Energy in Wireless Communication". In: *Science* 304.5667 (2004), pp. 78–80. DOI: 10.1126/science.1091277. eprint: https://www.science.org/

doi/pdf/10.1126/science.1091277. URL: https://www.science.org/doi/abs/10.1126/science.1091277.

[52] Mantas Lukoševičius, Herbert Jaeger, and Benjamin Schrauwen. "Reservoir Computing Trends". In: *KI - Künstliche Intelligenz* 26 (2012), pp. 365–371.

[53] Mantas Lukoševičius. "A Practical Guide to Applying Echo State Networks". In: *Neural Networks: Tricks of the Trade*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 659–686.

[54] Alexander Haluszczynski and Christoph Räth. "Good and bad predictions: Assessing and improving the replication of chaotic attractors by means of reservoir computing". In: *Chaos: An Interdisciplinary Journal of Nonlinear Science* 29.10 (Oct. 2019). ISSN: 1089-7682. DOI: 10.1063/1.5118725. URL: http://dx.doi.org/10.1063/1.5118725.

[55] Daniel J. Gauthier et al. "Next generation reservoir computing". In: *Nature Communications* 12.1 (Sept. 2021). ISSN: 2041-1723. DOI: 10.1038/s41467-021-25801-2. URL: http://dx.doi.org/10.1038/s41467-021-25801-2.

[56] 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". in: *Chaos: An Interdisciplinary Journal of Nonlinear Science* 31.1 (Jan. 2021). ISSN: 1089-7682. DOI: 10.1063/5.0024890. URL: http://dx.doi.org/10.1063/5.0024890.

[57] Yuanzhao Zhang and Sean P. Cornelius. "Catch-22s of reservoir computing". In: *Phys. Rev. Res.* 5 (3 Sept. 2023), p. 033213. DOI: 10.1103/PhysRevResearch.5.033213. URL: https://link.aps.org/doi/10.1103/PhysRevResearch.5.033213.

[58] Lina Jaurigue and Kathy Lüdge. "Connecting reservoir computing with statistical forecasting and deep neural networks". In: *Nature Communications* 13.1 (2022), p. 227. DOI: 10.1038/s41467-021-27715-5. URL: https://doi.org/10.1038/s41467-021-27715-5.

[59] ProbabilityCourse.com. *Unordered Sampling with Replacement*. Accessed: Feb 17, 2026. Section 2.1.4. 2024. URL: https://www.probabilitycourse.com/chapter2/2_1_4_unordered_with_replacement.php.

[60] NumPy Developers. *numpy.float64 — NumPy Documentation*. Accessed: 2026-01-05. URL: https://numpy.org/doc/stable/reference/arrays.scalars.html#numpy.float64.

[61] Isabelle Guyon and Andre Elisseeff. "An introduction to variable and feature selection". In: *J. Mach. Learn. Res.* 3 (2003), pp. 1157–1182. ISSN: 1533-7928. URL: http://portal.acm.org/citation.cfm?id=944919.944968.

[62] Ron Kohavi and George H. John. "Wrappers for feature subset selection". In: *Artificial Intelligence* 97.1 (1997). Relevance, pp. 273–324. ISSN: 0004-3702. DOI: https://doi.org/10.1016/S0004-3702(97)00043-X. URL: https://www.sciencedirect.com/science/article/pii/S000437029700043X.

[63] Haochun Ma et al. "Identifying causality drivers and deriving governing equations of nonlinear complex systems". In: *Chaos: An Interdisciplinary Journal of Nonlinear Science* 32.10 (Oct. 2022), p. 103128. ISSN: 1054-1500. DOI: 10.1063/5.0102250. eprint: https://pubs.aip.org/aip/cha/article-pdf/doi/10.1063/5.0102250/18196369/103128_1_5.0102250.pdf. URL: https://doi.org/10.1063/5.0102250.

[64] Gareth James et al. *An Introduction to Statistical Learning: with Applications in R*. Springer, 2013. URL: https://faculty.marshall.usc.edu/gareth-james/ISL/.

[65] Haiping Lu and Shuo Zhou. *An Introduction to Transparent Machine Learning*. https://alan-turing-institute.github.io/Intro-to-transparent-ML-course/index.html. Alan Turing Institute online course material, last accessed 2026-02-22. 2022.

[66] Milton Abramowitz and Irene A. Stegun. *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*. Vol. 55. National Bureau of Standards Applied Mathematics Series. p. 72, equation 4.3.16. Washington, DC: U.S. Government Printing Office, 1964.

[67] Daniel Köglmayr et al. *Two-shot learning of multiple strange attractors*. 2026. arXiv: 2601.21117 [nlin.CD]. URL: https://arxiv.org/abs/2601.21117.

[68] M. Bramson and D. Griffeath. "Asymptotics for interacting particle systems on Z^d". In: (1980). URL: https://doi.org/10.1007/BF01013315.

[69] Joschka Herteux and Christoph Räth. "Breaking symmetries of the reservoir equations in echo state networks". In: *Chaos: An Interdisciplinary Journal of Nonlinear Science* 30.12 (Dec. 2020). ISSN: 1089-7682. DOI: 10.1063/5.0028993. URL: http://dx.doi.org/10.1063/5.0028993.

[70] Edmilson Roque dos Santos and Erik Bollt. *On the emergence of numerical instabilities in Next Generation Reservoir Computing*. 2025. arXiv: 2505.00846 [stat.ML]. URL: https://arxiv.org/abs/2505.00846.

[71] Yuanzhao Zhang et al. "How more data can hurt: Instability and regularization in next-generation reservoir computing". In: *Chaos: An Interdisciplinary Journal of Nonlinear Science* 35.7 (July 2025). ISSN: 1089-7682. DOI: 10.1063/5.0262977. URL: http://dx.doi.org/10.1063/5.0262977.

A.1 Simulation Parameters

L63-Rössler system	
seed	42
Lorenz63	
dt	0.01
Starting point	(−14.030, −20.887, 25.535)
σ	10.0
ρ	28.0
β	8/3
Rössler	
dt	0.05
Starting point	(−0.753, 2.704, 1.392)
a	0.2
b	0.2
c	5.7

Table A.1: Standard simulation parameters for the L63-Rössler system

UK power grid	
Simulation parameters	standard: A.3
Train steps	4350
Pred steps	2000
Reservoir Computing	
n_dim	600
n_rad	0.9
n_avg_deg	3
regularization	0.01
w_in_scale	1.0
w_out_fit_flag	linear, square r
w_in_sparse	True
w_in_seed	0

Table A.2: Parameters for experiment 0

UK power grid	
α	0.02
β	0.06
γ_1	0.4
γ_2	0.4
$\omega_{\text{abs,low}}$	0.3
dt	8×10^{-2}
seed	0

Table A.3: Standard simulation parameters for the UK power grid

Mutual information	
Number of bins	50
UK power grid	
Simulation parameters	standard: A.3
Data length	10000

Table A.4: Parameters for experiment 1

UK power grid	
Simulation parameters	standard: A.3
Data length	10000
Transfer Entropy	
Number of bins	50
Time delay τ	1
Convergent Cross Mapping	
Lag	1
Embedding dimension	2
Training fraction	0.75
Library size	L_{train} (largest only)
Transfer Entropy (L63-Rössler)	
Number of bins	79
Time delay τ	1
L63-Rössler system	
Simulation parameters	standard: A.1
Transient steps (discarded)	30000
Data length	25000

Table A.5: Parameters for experiment 2

L63-Rössler system	
Simulation parameters	standard: A.1
Transient steps (discarded)	30000
Data length	25000
Train steps	20000
Test steps	5000
Reservoir Computing	
n_dim	400
n_rad	0.9
n_avg_deg	4
n_type_flag	0
regularization	10^{-5}
w_in_scale	1.0
w_in_sparse	True
w_in_seed	0
w_out_fit_flag	simple
save_r	False
NGRC	
k	5
s	1
orders	[1, 2]
regularization	10^{-5}
Bias	True

Table A.6: Parameters for experiment 3

UK power grid	
Simulation parameters	standard: A.3
Train steps	10000
Pred steps	2000
NGRC	
k	3
s	1
orders	[1, 2]
regularization	0.01
Bias	True
Search HPs	
q_max	20
q_init	25

Table A.7: Parameters for experiment 4

UK power grid	
Simulation parameters	standard: A.3
Train steps	10000
Pred steps	2000
NGRC	
k	3
s	1
orders	[1, 2]
regularization	0.01
Bias	True
Search HPs	
q_max	20
alpha_1	0.02
alpha_2	0.05
Convergent Cross Mapping	
Lag	1
Embedding dimension	2
Training fraction	0.75
Library size	L_{train} (largest only)
NGRC-GLS	
k	1
s	1
orders	[1]
regularization	0.01
Bias	True

Table A.8: Parameters for experiment 5

Kuramoto System	
K	0.15
degree	3
gamma	5
delta	0.8
dt	0.02
seed	42
Transient steps	30000
Train steps	7500
Pred steps	2000
NGRC	
k	3
s	1
orders	[1, 3]
regularization	0.01
Bias	True
NGRC-GLS	
k	4
s	5
orders	[1]
regularization	0.1
Bias	True
Optimized NGRC	
k	9
s	3
orders	[1, 3]
regularization	0.001
Bias	True

Table A.9: Parameters for experiment 6

Kuramoto System	
K	0.14
degree	3
gamma	5
delta	0.8
dt	0.08
seed	42
Transient steps (discarded)	10000
Train steps	10000
Pred steps	2000
UK power grid	
Simulation parameters	standard: A.3
Train steps	10000
Pred steps	2000
Divergent NGRC	
k	4
s	1
orders	[1, 2]
regularization	0.09
Bias	True
Divergent NGRC UKPG	
k	3
s	1
orders	[1, 3]
regularization	0.1
Bias	True
Kuramoto NGRC	
k	1
s	1
regularization	0.01
Bias	True
Sine NGRC	
k	3
s	1
orders	[1, 2, 3]
regularization	0.01
Bias	True

Table A.10: Parameters for experiment 7

L63-Rössler system	
Simulation parameters	standard: A.1
Transient steps (discarded)	30000
Data length	25000
Train steps	20000
Test steps	5000
Transfer Entropy	
Number of bins	79
Time delay τ	1
CLS Parameters	
alpha_1	0.05
alpha_2	0.05
NGRC (Wrapper & GLS)	
k	3
s	1
test_size	0.4
orders	[1, 2, 3]
regularization	10^{-3}
Bias	True

Table A.11: Parameters for L63-Rössler CLS results

Lorenz96 system	
Dimension	40
F	5
dt	0.05
Initial condition	$x_0 = F \cdot \mathbf{1} + 10^{-5} \cdot \mathcal{U}(0, 1)$
Seed	42
Train steps	4000
Prediction steps	1000
Transfer Entropy	
Number of bins	30
Time delay τ	1
NGRC (Wrapper)	
k	3
s	1
orders	[1, 2]
test_size	0.3
regularization	10^{-4}
NGRC (GLS)	
k	5
s	1
orders	[1, 2]
regularization	0.01
bias	True
CLS	
alpha_1	0.03
alpha_2	0.03

Table A.12: Parameters for L-96 CLS results

UK power grid	
Simulation parameters	standard: A.3
Train steps	10000
Pred steps	2000
Convergent Cross Mapping	
Lag	1
Embedding dimension	2
Training fraction	0.75
Library size	L_{train} (largest only)
Sine-NGRC	
k	3
s	1
orders	[1, 2]
regularization	0.001
bias	True
Kuramoto-NGRC	
k	1
s	1
orders	[1]
regularization	0.01
bias	True
CLS	
alpha_1	0.05
alpha_2	0.05

Table A.13: Parameters for UKPG CLS results

A.2 Divergence in NGRC investigated

As discussed in Section 7.5, NGRC exhibits sensitivity to hyperparameter choices that can lead to divergent predictions. In particular, the choice of the delay spacing parameter s has a significant impact on the stability of the learned dynamics [1]. The divergence in NGRC is investigated on the UKPG with standard simulation parameters and $N_{\text{train}} = 5000, N_{\text{pred}} = 2000$. The other NGRC parameters are $k : 3\,, \alpha = 0.01\,,$ orders $= [1, 2]$.

As illustrated in Figure A.1, selecting an appropriate value of s can yield near-perfect agreement with the reference trajectory, whereas other choices result in divergence (see Figure 7.18). A comprehensive grid search over hyperparameters is presented in Figure A.2, where distinct regions of stable and unstable behavior can be observed. Furthermore, Figure A.3 indicates that the predictive performance in those regions is highly bimodal: the model either achieves perfectly accurate forecasts or rapidly diverges, with no intermediate regime of moderate accuracy. This indicates that the time delay plays a critical role in the dynamical stability of NGRC predictions.

Common heuristics for selecting s are the first minimum of the mutual information or the zero-crossings of the autocorrelation function. These criteria aim to reduce redundancy in the feature vector by selecting delays that provide maximally informative embeddings. A comparison of these heuristics with the observed divergence regions is shown in Figures A.5 and A.4. While the autocorrelation-based heuristic appears to align reasonably well with stable regions in this case, it is unclear how to apply autocorrelation to more complex systems, as the locations of zero-crossings may vary between the different system dimensions. It is therefore unclear which of the system dimensions should be used to choose s. The mutual information test provides no insights into the cause of the divergence regions.

Despite the empirical relevance of these heuristics, the underlying causes of divergence in NGRC are not yet fully understood. Only few existing studies have investigated this issue. Santos et al. [70] hypothesize that ill-conditioning of the feature (design) matrix is a primary contributor to dynamical instability. In this setting, small perturbations in the data can lead to large variations in the learned readout weights, which may destabilize recursive prediction. Similarly, Zhang et al. [71] observe that increasing the amount and thereby the redundancy of training data can lead to an increase in the condition number of the readout matrix, which correlates with reduced stability. To investigate those findings on the Kuramoto system, the condition numbers of the readout matrix W_{out} and the feature matrix W_{in} are compared to the observed divergence regions in Figures A.7 and A.6, respectively. There seems to be no relationship between the feature matrix and the divergence regions. However, the condition number of W_{out} exhibits spikes near the onset of divergence regions. Nonetheless, throughout the total unstable region, no consistent relationship is observed. Moreover, a direct comparison between matrices obtained from stable and unstable configurations (Figure A.8) does not reveal a clear structural distinction.

Overall, these observations suggest that while ill-conditioning may contribute to instability in certain regimes, it does not fully explain the divergence behavior of NGRC. A systematic understanding of the factors governing dynamical stability remains an open research question that is beyond the scope of this thesis but of great importance to NGRC research.

[1] The polynomial order set also has a significant impact

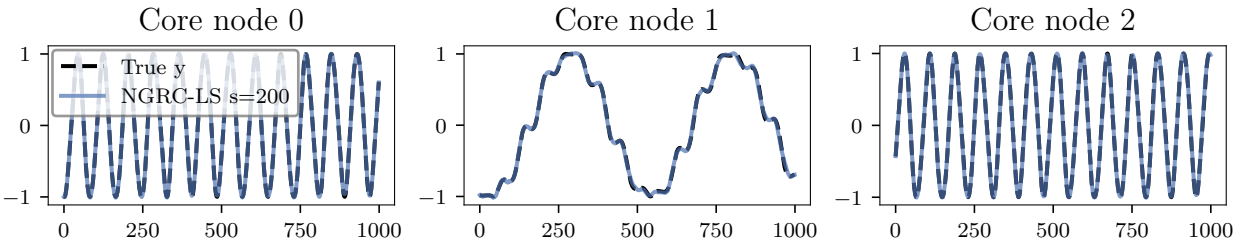


Figure A.1: NGRC-LS achieves a perfect reconstruction of the 3-d Kuramoto system for $s = 200$

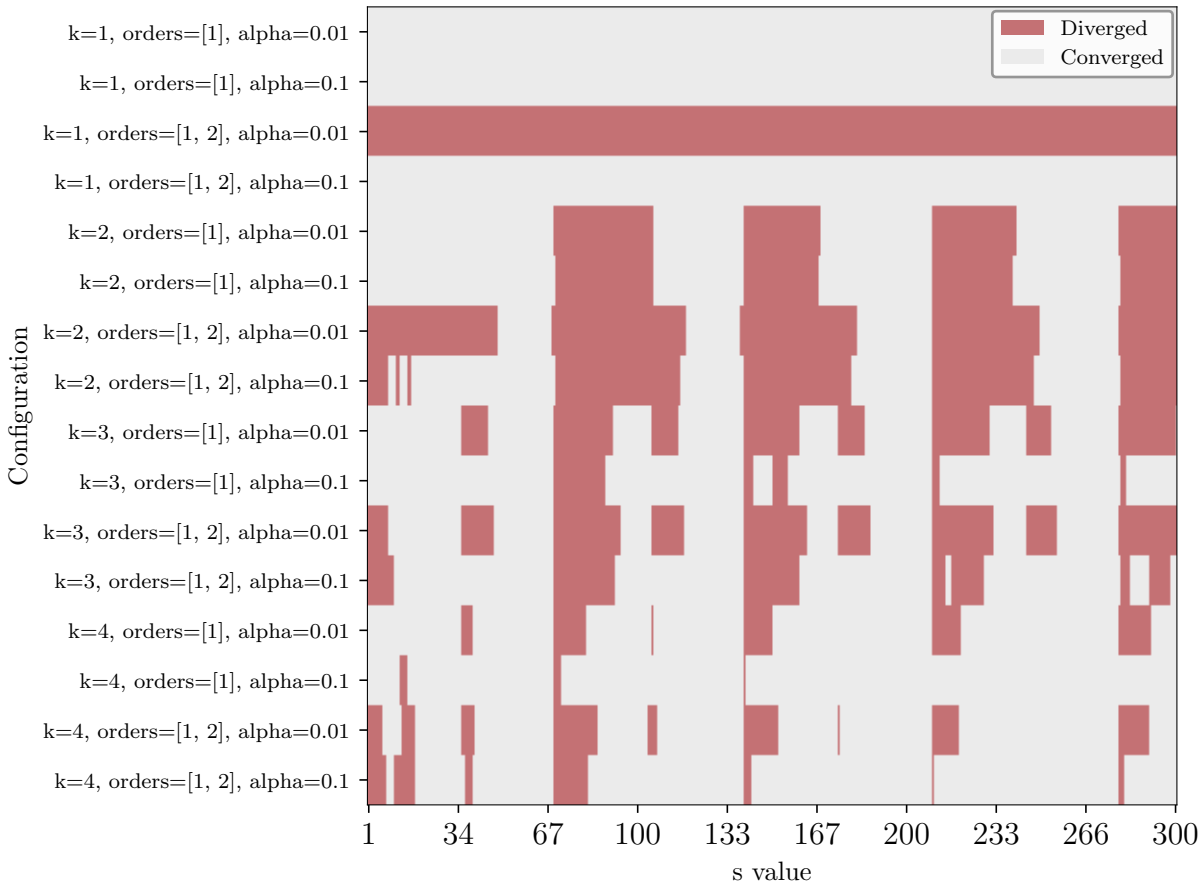


Figure A.2: Divergence regions across different hyperparameter configurations. Despite variation in hyperparameter values, certain divergence regions are shared across configurations, with the onset of divergence occurring at consistent values of s, suggesting a common underlying mechanism governing the transition to instability.

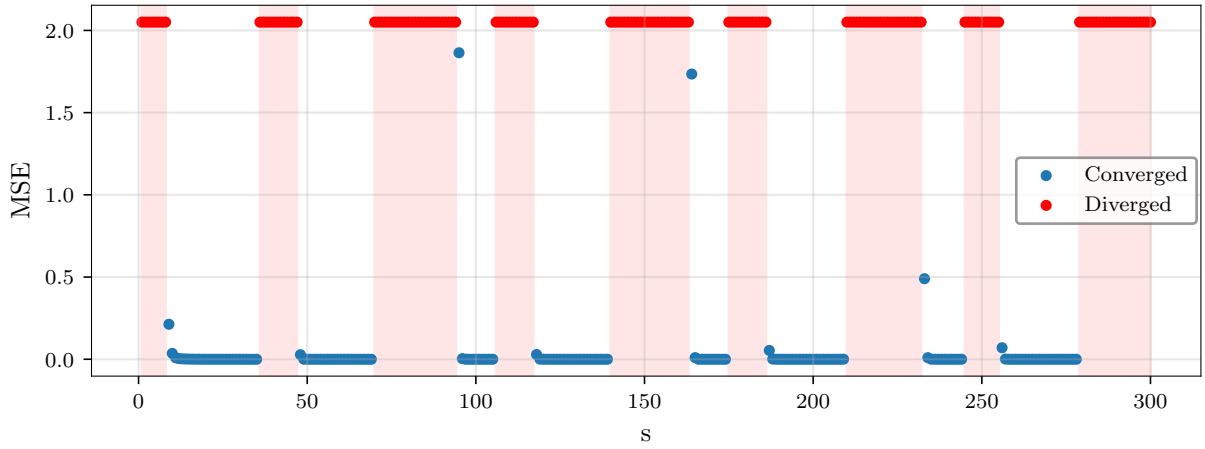


Figure A.3: Mean squared error (MSE) of the NGRC model as a function of delay hyperparameter s. Divergent and non-divergent regions are indicated. Within non-divergent regions the model achieves near-perfect reconstruction, followed by a sharp phase transition into divergent regimes.

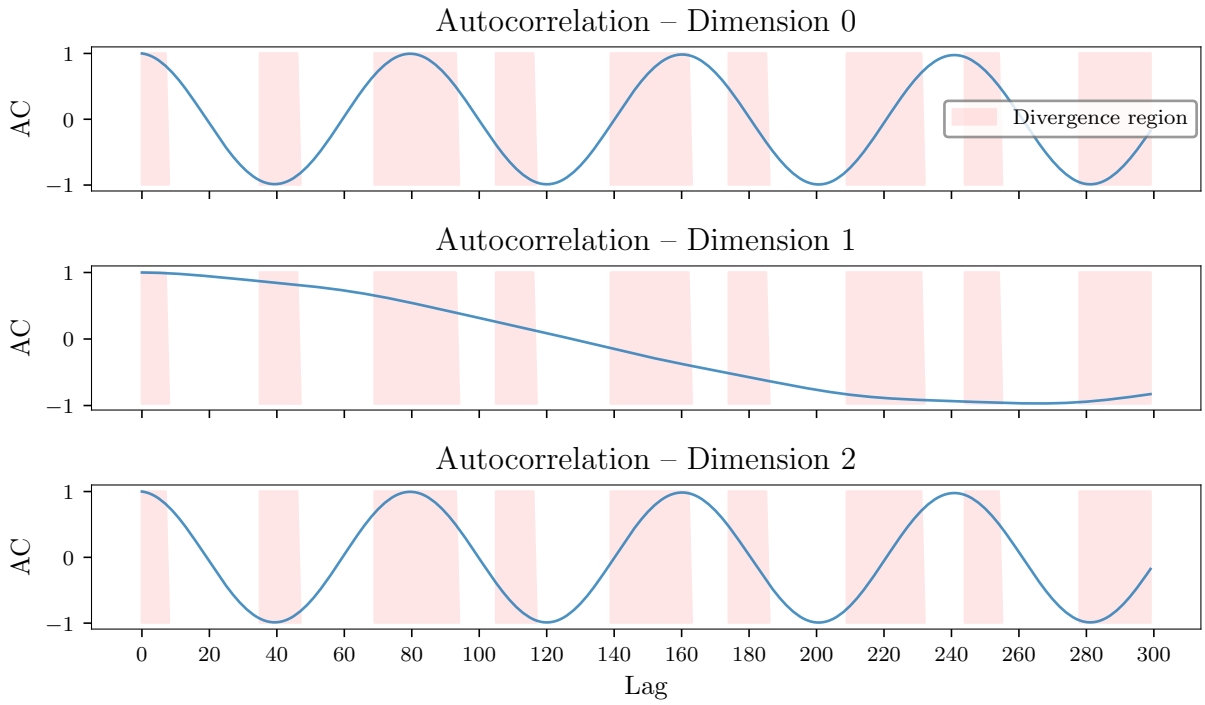


Figure A.4: Autocorrelation of the training time series plotted alongside the divergence regions. The alignment between zeros of the autocorrelation function and the convergent regions is examined as a potential heuristic for identifying stable time delay choices.

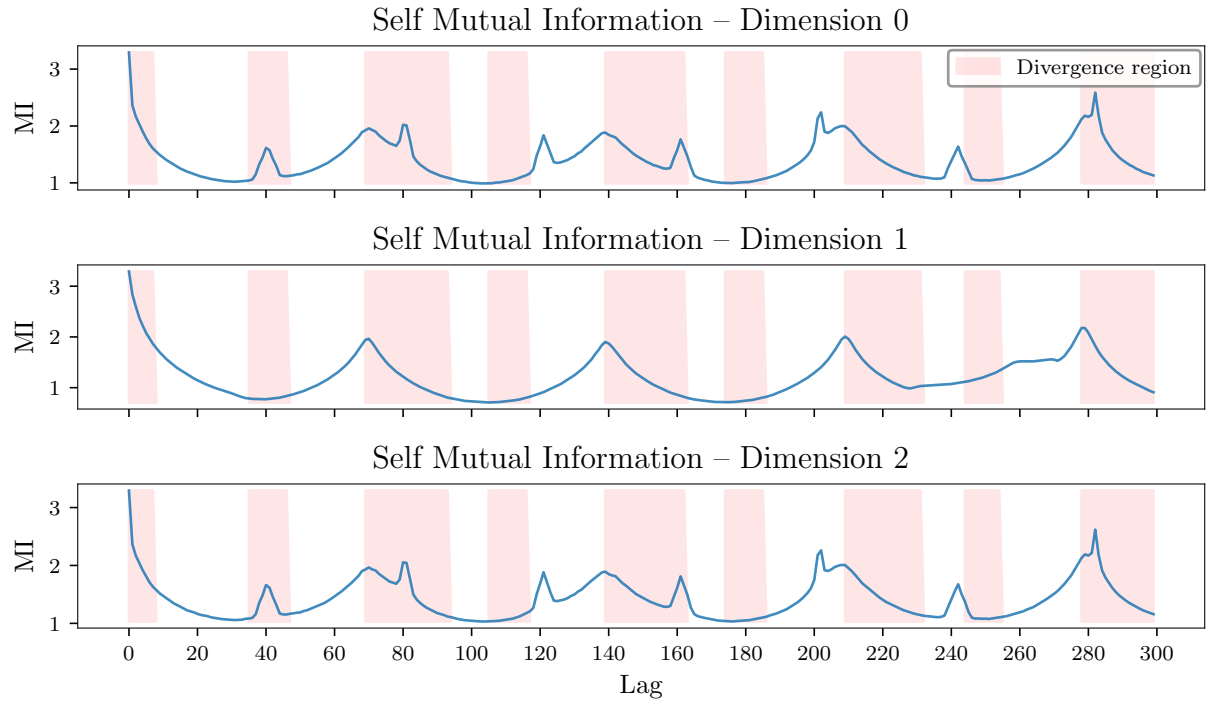


Figure A.5: Self mutual information of the training time series plotted alongside the divergence regions, analogous to Fig. A.4. The first minimum of the mutual information is examined as an alternative heuristic for identifying time delay values associated with stable NGRC dynamics.

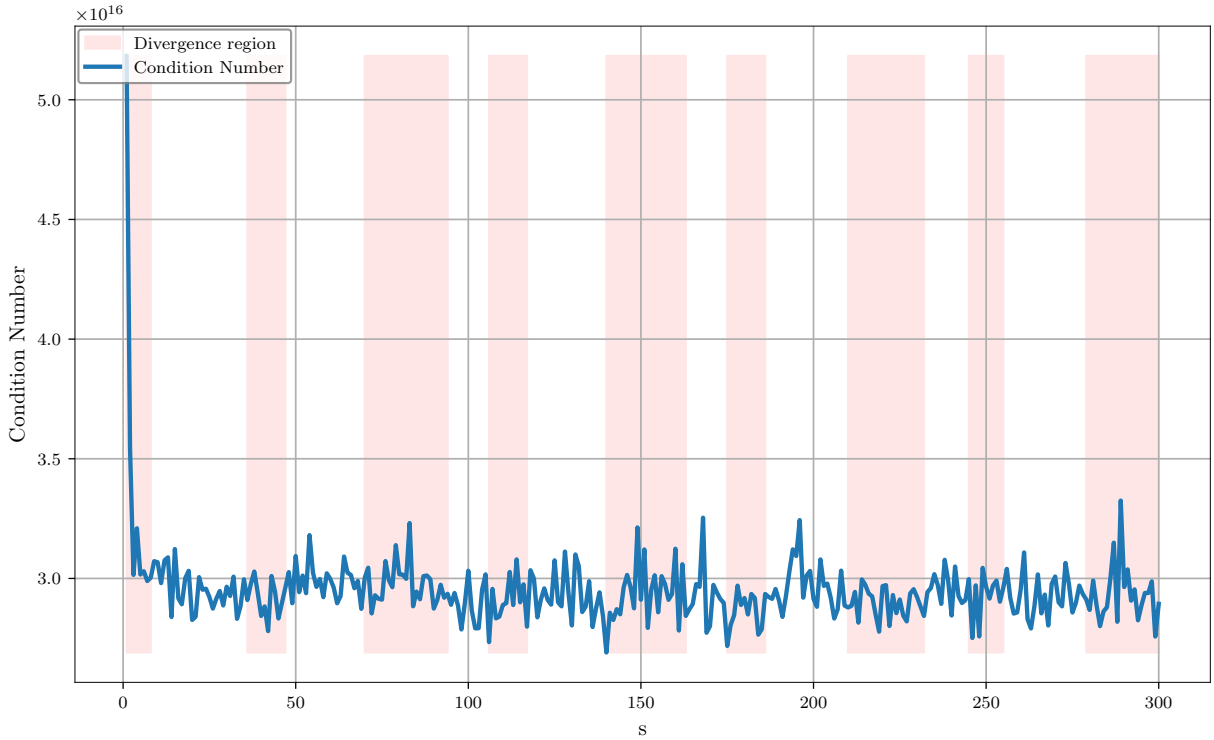


Figure A.6: Condition number of the feature matrix as a function of delay s. The condition number is initially large for very small s, consistent with the near-linear dependence between feature columns evaluated at consecutive time points, before decaying to a noisy baseline with no distinguished structure at larger s.

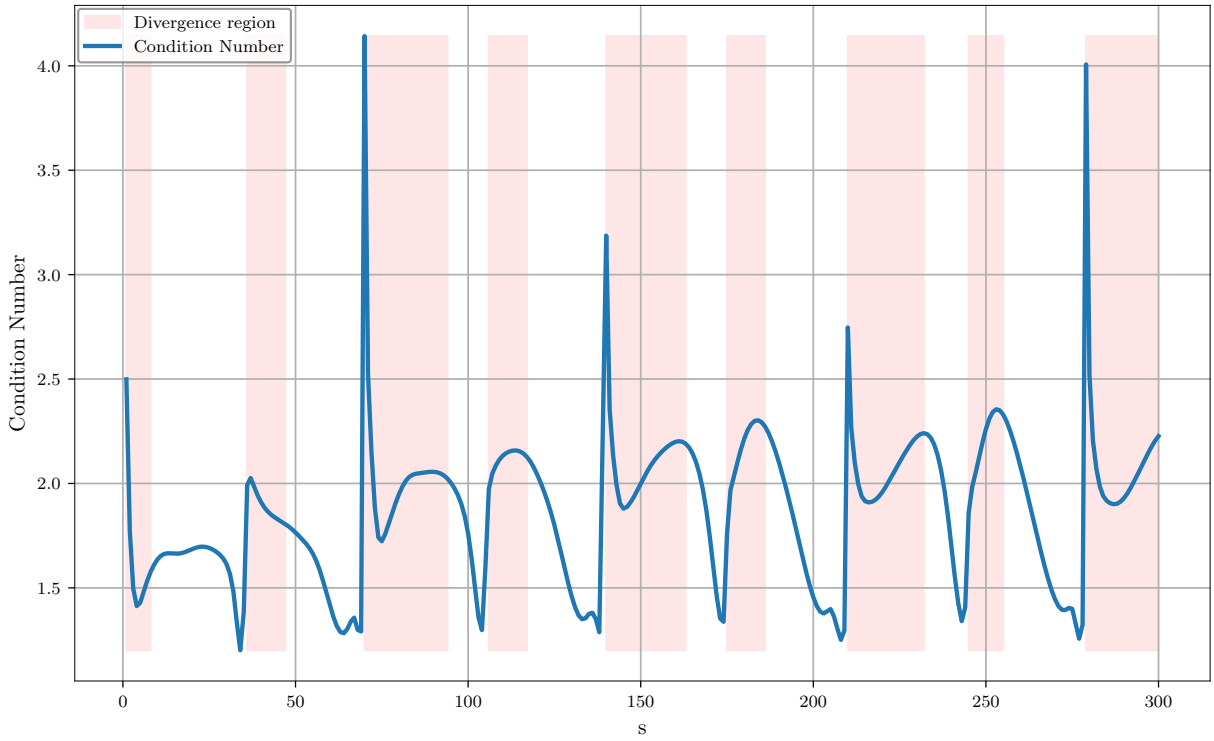


Figure A.7: Condition number of the output weight matrix W_{out} plotted against the divergence regions. Sharp spikes in the condition number coincide with the start of divergent regions.

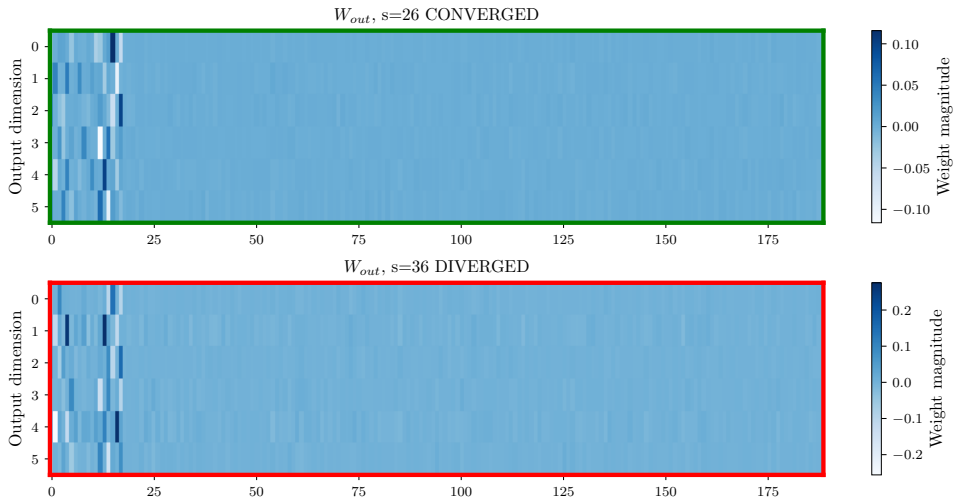


Figure A.8: Visualisation of two representative W_{out} matrices, one drawn from a convergent configuration (top) and one from a divergent configuration (bottom). No qualitative difference in the weight structure is apparent between the two cases, indicating that divergence cannot be diagnosed directly from inspection of the learned weight values.

Declaration of Authorship

I hereby declare that this thesis titled

Causal Local States: Achieving Scalable Simultaneous Causal Network Inference and Forecasting for Dynamical Systems

is my own work, and that I have not used any sources and aids other than those stated in the thesis.

I have used generative artificial intelligence tools throughout this thesis for various purposes. Perplexity AI was used to identify relevant literature, GitHub Copilot to support code development, and ChatGPT to refine language and grammar in this manuscript. All content generated by AI tools has been critically evaluated, edited, and integrated by me. Proper source attribution has been provided where applicable. This way, I have ensured that the use of AI tools has been in line with the academic standards of originality and integrity, while also leveraging their capabilities to enhance the quality and efficiency of my work.

München, der 01.04.2026

Jonas Braun