Modelling Observations of Dynamical Systems with Memory

Dissertation
zur Erlangung des
akademischen Grades eines
Doktors der Naturwissenschaften

vorgelegt von
Niklas Wulkow

am Fachbereich
Mathematik und Informatik
der Freien Universität Berlin

Berlin, März 2022

Betreuer:	Prof. Dr. Christof Schütte Zuse Institute Berlin Takustraße 7, 14195 Berlin
Gutachter:	Prof. Dr. Christof Schütte Prof. Dr. Illia Horenko (USI Lugano)

Tag der Disputation: 30.06.2022

Eidesstattliche Erklärung

Ich erkläre gegenüber der Freien Universität Berlin, dass ich die vorliegende Dissertation selbstständig und ohne Benutzung anderer als der angegebenen Quellen und Hilfsmittel angefertigt habe. Die vorliegende Arbeit ist frei von Plagiaten. Alle Ausführungen, die wörtlich oder inhaltlich aus anderen Schriften entnommen sind, habe ich als solche kenntlich gemacht. Diese Arbeit wurde in gleicher oder ähnlicher Form noch bei keiner anderen Universität als Prüfungsleistung eingereicht.

Mit einer Prüfung meiner Arbeit durch ein Plagiatsprüfungsprogramm erkläre ich mich einverstanden.

Berlin, März 2022

Niklas Wulkow

Figures

Figures 1.1–1.5, 2.1–2.4, 3.1, 3.3–3.8, 4.5– 4.14, 5.1–5.3, 5.6–5.10 have been designed by the author for this thesis.

Figures 3.2, 4.1–4.3 have been designed by author for this thesis, too, but are conceptually identical to ones the author has created for [WKS21]. Figures 5.4 and 5.5 are conceptually identical to ones the author has created for [WKSS21].

Figures 1.6 and 4.4 have been designed by the author for [WKS21].

Acknowledgements

First and foremost, I would like to sincerely thank my supervisors Christof Schütte and Péter Koltai, from whom I learnt more than they can probably imagine, for their invaluable support, help and trust. I hope I could pay it back.

I would also like to genuinely thank my colleagues who have become friends Alexander Sikorski, Enric Ribera-Borrell, Jan-Hendrik Niemann, Johannes Zonker, Luzie Helfmann, Renata Secchi, Robert Polzin, Robert Rabben and Vikram Sunkara for all the fun we have had with mathematics and many other things.

A special thanks goes to my office mate Luzie Helfmann for our always helpful discussions, Vikram Sunkara for our very enjoyable collaboration and the many things I've learnt during it and Illia Horenko for introducing me to his field of work.

Furthermore, I would never take for granted the help I received in various forms from Ralf Banisch, Natasa Conrad, Raphael Gerlach, Jannes Quer and Marcus Weber.

Finally, I could not have written a single line of this thesis without the support from my family, my friends and Lisa. You know that I cannot thank you enough.

This work was funded by Freie Universität Berlin, Zuse Institute Berlin and the Einstein Center for Mathematics Berlin.

Contents

Introduction

Over the last decades, learning the governing equations of a dynamical system from data has been crucial in gaining insight into many different scientific fields and applications such as molecular dynamics [HHSN07], climatology [WDKW21], chemical reaction kinetics [SBSR19] or fluid dynamics [BNK20]. Once a mathematical description of the system is derived, one has the chance to understand relations between different variables, explain which influences drove the system up to a certain point and make predictions about its future evolution.

Deriving a mathematical model, however, is a challenge whose degree of difficulty depends on the complexity of the system, the existence of prior intuition about the dynamics and the available data. These characteristics also determine how to best approach the modelling process. It has been shown in innumerably many publications that many real-world dynamical systems admit a representation where future states of the system depend solely on its current state and not on its past memory. Such *memoryless* models occupy a broad range of model classes, containing, among others, simple Markov State Models [Sar11,HP18], differential equations [Chi17, WTH$^+$] and their time-discretizations [BD91, BPK16], Kernel methods [MOW15,SSAB20] and neural networks [BK19, KLL$^+$21, PD18].

However, an accurate model should incorporate all or at least the dominant relations between variables of the system. Often times, not all these variables can be physically measured. If no data about some of these variables are available, one generally cannot derive a model which includes these inaccessible variables, so that the task of modelling the dynamics becomes significantly more difficult. Nevertheless, it has been discussed from different perspectives how in these cases, while the full dynamics might admit a memoryless representation, one can still construct a model for the evolution of the observed variables by utilizing their past, or memory, terms.

One of these perspectives was established by Floris Takens in 1981 [Tak06]. He proved that under certain, fairly general conditions, by merging a fixed number of past observations into a new state, called a *delay embedding*, one can construct dynamics along the delay-embedded states that are topologically equivalent to the underlying full dynamics. While Takens gives an upper bound for the required number of past terms, his result does not inspire a specific model form for the observed dynamics.

Another perspective is the one of the Mori–Zwanzig formalism (MZ) [Zwa01]. It is more constructive than Takens and explains how the evolution of an observable of the system can be modelled as a projection of the full dynamics using past values of the observable. Under certain assumptions, the Mori–Zwanzig formalism even delivers an explicit model form for these observed dynamics, so-called *autoregressive models* [BD91, Bil13]. In contrast to Takens, it technically requires an infinite number of past terms.

From a numerical point of view, on the basis of the Takens and MZ approaches, there have been significant advances in defining methods to model observed dynamics from data, extending and even generalizing the previously mentioned memoryless model forms, e.g., in [LL21, BBP+16, HDPCBS18, LLC15, CMCW+20]. Furthermore, over the last ten years several regression methods have been introduced [TRL+14, WKR14, BPK16] that estimate the *Koopman operator*, which formalizes the propagation of observables over time, from data. These methods do not use memory but function by transforming observations to a suitable higher-dimensional space with the hope of determining a simple mapping between points in this space and future states of the observable. Especially interesting in this regard are methods which generate a sparse and therefore more interpretable model, e.g., Sparse Identification of Nonlinear Dynamics (SINDy) [BPK16]. This is in contrast to most memory-based models which require many parameters to describe the influence of all the memory terms and are typically hardly interpretable.

In this thesis, the task of modelling observed dynamics using memory is approached from a theoretical, a numerical and an application-centred perspective. In particular, different numerical methods for the modelling of memory-exhibiting dynamics are constructed by extending known methods and leveraging memory. These methods are theoretically and numerically compared to existing methods and applied to different examples that come from very different areas.

One such new method combines SINDy with memory-based autoregressive (AR) models. It generates sparse models for memory-exhibiting and potentially nonlinear dynamics and is therefore called Sparse Identification of Nonlinear Autoregressive Models (SINAR). Furthermore, it is shown that if certain conditions are fulfilled, SINAR, AR models and the Koopman-based methods are in fact mathematically strongly related and admit equivalent formulations. All these methods are tested on numerical examples and compared to several other already existing methods, including neural networks and a Kernel method.

A second new method is especially suited for the modelling of high-dimensional dynamics. Even if a dynamical system is fully accessible, a high dimension often makes a direct model identification infeasible. This is a more and more frequent case in the age of Big Data, when complex systems can be measured and large datasets are assembled, e.g., in molecular dynamics [RRW20] or climate science [FGL+21]. In such cases, one is usually well-advised to perform a dimension reduction of the system first, i.e., finding a best-approximative representation of system states in a lower dimension [CLKB19, JC16, CLL+05]. In general, this entails a loss of information since states do not have to permit such a representation in reduced coordinates. Again, the low-dimensional representation of a state can be interpreted as an observable of the full system state. Then in order to model the evolution along these reduced coordinates, the need for memory terms ensues according to Takens and MZ. This concept is employed using the low-cost discretization method Scalable Probabilistic Approximation (SPA) [GPNH20]. SPA can be used to simultaneously find optimal probabilistic representations of data points while constructing a linear model between such representations of different variables. When SPA is leveraged in a certain way, it can be seen as a projection of points to a low-dimensional convex polytope, thus leading to a representation of points in reduced coordinates that are given with respect to the polytope. A new method called *memory SPA* (mSPA) is introduced which estimates the dynamics in this polytope using memory. It deploys a specific nonlinear transformation on points in the polytope and constructs a mapping forward in time while keeping predicted states inside the polytope. This guarantees *stability* of long-term forecasts, distinguishing the method from most of the previously introduced numerical methods. It is shown that mSPA is capable to model nonlinear dynamical systems of varying complexity and high dimension.

Most of the examples that these methods are tested on come from mathematics and natural sciences, where the underlying family of governing equations is known. However, as a central contribution of this thesis, especially the SINAR method is deployed in connection to dynamics from social science for which merely an intuition of the governing equations is available. For such cases, agent-based models (ABMs) have become increasingly prominent [LJMR12, JSW98]. ABMs model each element, such as a single person within a society, individually by equipping it with specific, often stochastic, rules that govern its behaviour. From the individual behaviour of these so-called *agents* emerges a collective behaviour of the whole society. Usually, this collective behaviour, measured in the form of aggregated variables or statistics, is of the main interest in ABMs. In some cases, the ABM serves simply to develop a suitable equation-based model structure for the dynamics of these statistics in the real-world setting [WCDC$^+$21]. Interpreting the evolution of the statistics of the ABM as observed dynamics, this procedure then falls directly into the context of the results from Takens and MZ which suggest the use of memory for the equation-based model. This aspect has so far mostly been neglected in research on ABMs and especially so in the field of opinion dynamics, an emerging field which investigates the spreading of opinions across a population. In this thesis, it is shown in detail that in fact the inclusion of memory using SINAR improves the model identification for the observed dynamics of ABMs. To this end, two new ABMs for opinion dynamics are defined. This should serve as incentive for the use of memory in modelling observed dynamics in real-world problems that come from outside of the natural sciences.

The main contributions of this thesis are the following:

- Defining two new numerical methods for the modelling of nonlinear dynamical systems: (1) SINAR, which is suited for memory-exhibiting dynamics and can produce sparse models and (2) mSPA, which is suited for high-dimensional systems and produces stable models.

- Bridging the gap between (1) practical numerical methods to model observed dynamics using memory and (2) the field of agent-based models. This is done by formalizing two newly defined ABMs for opinion spreading in the context of the Mori–Zwanzig formalism and deploying the memory-based SINAR method to model observed dynamics of them.

The first two chapters of this thesis are a comprehensive review of relevant concepts for the modelling of observed dynamics: in **Chapter 1**, the theoretical concepts of Takens' Theorem, including variants proved by other researchers, and the Mori–Zwanzig formalism are explained in detail. In **Chapter 2**, several numerical methods to approximate the observed dynamics from data are discussed, highlighting the connections between them.

Chapters 3–5 contain work that is either novel or was previously published by the author of this thesis in articles [WKS21] and [WKSS21] together with co-authors: **Chapter 3** contains the introduction of SINAR, results on the connections between the Koopman-based methods introduced in Chapter 2 and numerical comparisons between several numerical methods described in this thesis. **Chapter 4** contains a detailed formalization and analysis of the use of memory to model observed dynamics from two newly defined ABMs for opinion spreading. In **Chapter 5**, mSPA, the novel method for the modelling of nonlinear dynamics, is formally introduced and applied to several examples.

List of essential symbols

Symbol	Meaning
General	
F	Dynamics
ϕ	Observable
p	Memory depth
Φ	Delay-coordinate map
$\mathbb{X}$	Full state space
$\mathbb{Y}$	State space of observable
$\mathcal{H}_\phi$	Set of functions from $\mathbb{X}$ to $\mathbb{Y}$ needing only $\phi(X)$
X_t	Full system state
x_t	Observed system state
$\hat{x}_t$	Delay-coordinate map applied to X_t
ω_t	Stochastic influence in system
$\mathcal{K}$	Koopman operator
ψ	Basis functions
ε_t	Noise term
H	Hankel matrix
Chapter 1	
h_k	Mori–Zwanzig coefficients
H_k	Reformulated Mori–Zwanzig coefficients
Chapter 4	
α	Agent-based model coefficients
P^t	Opinion change probability in ABM
Chapter 5	
Σ	Matrix of vertices in SPA
Γ	Matrix of barycentric coordinates
σ	Vertex in SPA
γ_t	Barycentric coordinates of X_t
K	Number of vertices in SPA
Λ	Transition matrix in SPA
Ψ	Path affiliation function
ψ	Path affiliation vector

Chapter 1

Memory in Observed Dynamical Systems

Throughout most of this thesis, we will consider the following setting: let a deterministic discrete-time dynamical system be given by $F : \mathbb{R}^d \to \mathbb{R}^d$ while we only see an *observable* $\phi : \mathbb{R}^d \to \mathbb{R}^m$ for which in general $m < d$ or even $m = 1$. Questions arise such as *Which properties of the full system can we recover from* ϕ*?* or *Can we predict the evolution of* ϕ*?* As it turns out, by not only using the current state of ϕ but also including past information, one is often able to give satisfying answers to these questions.

An intuitive reasoning for this is the following: let the full dynamics F produce states X_t with $t = 0, 1, \ldots$ while we denote $\phi(X_t) = x_t$. What does x_t tell us about x_{t+1}? Usually not sufficiently much since x_{t+1} is a direct result of X_{t+1} which depends on the full state X_t. For most observables, x_t does not allow us to uniquely determine X_t, since the preimage of x_t under ϕ consists of more than one point. If we now also include x_{t-1} into our considerations, we find possible candidates of X_{t-1} to be points in $\phi^{-1}(x_{t-1})$. Thus, X_t has to lie inside the image of the set of candidates of X_{t-1} under F, so that $X_t \in \phi^{-1}(x_t) \cap F(\phi^{-1}(x_{t-1}))$. This allows us to narrow down the set of candidates for X_t subsequently by including more and more past terms. Assuming that we know F we could now reduce the set of possible values for X_{t+1} and thus x_{t+1}. Even assuming that we do not know F, this at least illustrates the vital point: information about the full state is encoded in past states, the *memory terms*, of the observed states.

Simple example

To build intuition, let us consider a simple example. Let a particle slowly move inside a one-dimensional interval and let us assume we know both the rule under which it moves and the distribution of positions inside the interval that it attains. Suppose we have discretized the interval into several boxes and can only observe in which box the particle is at any point in time. Our observable then simply takes any point x inside the interval and maps it onto a natural number denoting the number of the box it is in. Knowing that the particle is currently in box i then only allows us to shrink down the distribution from the full interval onto the ith box. But if the particle was in box $i - 1$ recently we can suspect that now it should be close to the border between these two boxes since it could not have moved far inside box i yet. Hence, in the near future it could well transition back to box $i - 1$ (see Figure 1.1). If, however, the particle had been in box i for a long time, it becomes more likely that currently it is well inside box i and will therefore remain inside the box for some time. In this way, past information about the observable changes the probabilities which we have to assign to each exact position of the particle and the future values of the observable, in this case, the boxes.

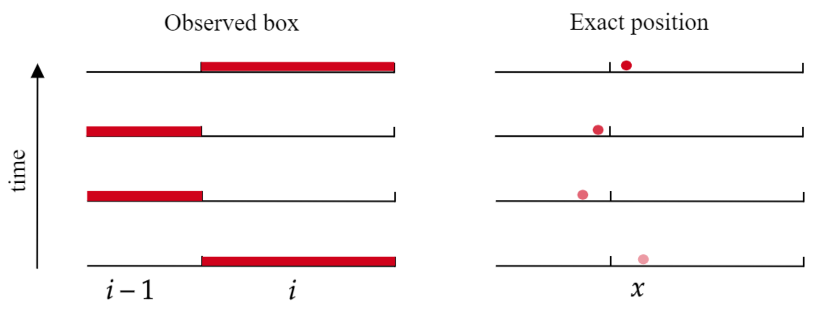


Figure 1.1: Example of how past observations (the box the particle is in at respective points in time) provide additional information on the current exact state. The fact that the particle switched boxes indicates that currently it should be close to the border between boxes i and $i-1$, making transitions to box $i-1$ more likely compared to being positioned well inside box i.

To place this on a mathematically sound foundation, this chapter will summarize the two aforementioned perspectives on the emergence of memory in dynamical systems, Takens and Mori–Zwanzig, starting with the Theorem of Takens and its implications.

We will at times use results from theory about Markov processes and memoryless dynamical systems. Giving a full introduction to these broad areas would go beyond the scope of this thesis. For a comprehensive overview, please refer to [Sar11] and [KR99].

1.1 Takens' Theorem

The work of Floris Takens discusses under which conditions certain properties of a dynamical system F can be reconstructed from a scalar-valued observable. The system has to evolve on a compact smooth manifold $\mathcal{M}$, an invariant set of the dynamics, so that $F: \mathcal{M} \to \mathcal{M}$. $\mathcal{M}$ is often called *attractor* if one additionally assumes that trajectories starting outside of $\mathcal{M}$ converge to it, enabling one to consider the dynamics only on $\mathcal{M}$. An attractor is defined as follows.

Definition 1.1 (Attractor)**.** For a function $F: \mathbb{R}^d \to \mathbb{R}^d$, a set $A \subset \mathbb{R}^d$ is an attractor if it fulfils (1) $F(X) \in A$ if $X \in A$, (2) there is a neighbourhood U of A so that for all $X \in U$, $\lim_{t\to\infty} F^t(X) \in A$ and (3) there is no subset of A that fulfils (1) and (2).

Before we can truly understand the full width of both its prerequisites and implications, we need to clarify what the term "reconstructed" could mean: the goal is to create *equivalent* dynamics where equivalent means preservation of certain properties. The existing literature on this topic typically speaks of three different types of these properties:

- Topological: these are, by definition, all properties that are preserved under homeomorphisms. They include the dimension, compactness and connectedness of the attractor and the number and types of fixed points of the dynamics.
- Dynamical: these pertain the evolution of the constructed dynamics, such as smoothness or the Lyapunov exponents, which quantify how fast two trajectories that start at slightly perturbed initial values deviate from each other. Of special interest is typically the *predictability* of the dynamics. Although not being a strictly defined mathematical property, it is often the focal point of the analysis: how precisely can we formulate the dynamics so that we can predict both its short- and long-term behaviour well?
- Geometric: such properties are in regard to the shape of the attractor. They include distances between points and its volume. As we will see, preservation of geometric properties is intimately connected with predictability.

1.1.1 The Theorem and its Variants

Now, with only an observable $\phi : \mathcal{M} \to \mathbb{R}$ (Takens explicitly considers the case of scalar-valued observables but this can be generalized) at hand, we strive to formulate dynamics in ϕ in such a way that as many of the introduced properties as possible are preserved. To this end, we will map $\mathcal{M}$ to a new manifold using the image of the observable and define dynamics on it.

As will become clear, a mapping that we should seek for this task is an *embedding*. In order to define an embedding, we first need the term *immersion*:

Definition 1.2 (Immersion)**.** An immersion is a differentiable function between two differentiable manifolds whose derivative is injective at every point in the manifold.

An embedding then is an injective immersion. This means that an embedding neither collapses points nor tangent directions. As a consequence, one obtains dynamics in the embedded manifold that are equivalent to the full dynamics up to a mere coordinate change.

An embedding can also be called a smooth *diffeomorphism* onto its image since a diffeomorphism is defined as a function between manifolds that is smoothly differentiable, invertible and its inverse is also smoothly differentiable. It preserves all topological properties – this, in fact, is by definition, since a diffeomorphism is a homeomorphism between manifolds – and is, as it turns out, well-suited to preserve dynamical and geometric properties to a strong degree.

Takens' Theorem shows that by leveraging the information in ϕ in a specific way, one can in fact create an embedding. Before being able to understand what it is motivated from, we first need to turn ourselves to a much older theorem proved by Hassler Whitney in 1936. It ensures that, given a d-dimensional smooth and compact manifold $\mathcal{M}$, typically an arbitrarily chosen smooth $(2d+1)$-dimensional mapping will be an embedding. To be precise, the theorem states

Theorem 1.3 (Whitney, 1936, [Whi36])**.** Let $\mathcal{M}$ be a d-dimensional smooth and compact manifold. Then the set of smooth functions $f : \mathcal{M} \to \mathbb{R}^{2d+1}$ that are an embedding is generic.

The term generic means open and dense inside the $\mathcal{C}^1$-topology of maps, implying that around every function that is an embedding, there is a ball with positive radius whose elements are also embeddings (this pertains to openness) and for every function that is not an embedding it only takes an infinitesimal perturbation to arrive at an embedding (the meaning of denseness).

Remark. Please note that all theorems, lemmas and propositions that were not proved by the author of this thesis are marked as such by the literature reference where also the proof can be found. Later, when new results from the author are introduced, instead of a reference there will be a proof and theorems will be distinguished by shaded boxes.

Whitney's Theorem shows that generically smooth mappings, if they are sufficiently high-dimensional, are in fact embeddings of a manifold. He went on to prove in 1944 that for every smooth, compact manifold of dimension $d \geq 2$ there exists a mapping into $\mathbb{R}^{2d}$ that is an embedding which is known under the Strong Theorem of Whitney.

Delay embedding

Takens managed to build on the foundation laid by Theorem 1.3 and relate it to the setting that this thesis is concerned with: dynamics of which only some variables are observed. Although we will mostly consider discrete-time dynamics throughout this thesis, Takens' Theorem can be placed into the context of time-continuous systems of which we take measurements at discrete time steps. This perspective should allow for a better understanding of certain aspects of the topic. F then is understood to transport the dynamics by a time step of length τ while the time-discrete perspective can straightforwardly be taken by setting $\tau = 1$. In both cases, we write

$$X_{t+1} = F(X_t) \tag{1.1}$$

while X_t denotes the state of the system at the tth time step.

We use sequences of states of the observable ϕ to construct a topologically equivalent system, as the following steps assert.

Definition 1.4 (Delay-coordinate map)**.** Let $\phi : \mathcal{M} \to \mathbb{R}$ be a smooth, scalar-valued observable. Then let us define the delay-coordinate map of ϕ under a time-τ-map F with memory depth, or, as we call it here, *embedding dimension* p by

$$\Phi_{\phi,F,p,\tau}(X) = (\phi(X), \phi(F^{-1}(X)), \dots, \phi(F^{-(p-1)}(X)))^T \in \mathbb{R}^p. \tag{1.2}$$

Takens' Theorem reads:

Theorem 1.5 (Takens, 1981, [Tak06]). Let $\mathcal{M}$ be a smooth, compact, d-dimensional manifold. The set of pairs (F, ϕ) for which $\Phi_{\phi,F,p,\tau}$ is an embedding is generic in $\mathcal{D}^r(\mathcal{M}) \times \mathcal{C}^r(\mathcal{M}, \mathbb{R})$ if $p > 2d$ for $r \geq 2$.

$\mathcal{D}^r(\mathcal{M})$ denotes the set of r-times continuously differentiable diffeomorphisms mapping from $\mathcal{M}$ to $\mathcal{M}$. Takens originally showed this for $r = 2$ but it was shown by J. P. Huke [Huk93] that it holds for $r = 1$. Huke also demonstrated the proof of Theorem 1.5 in great detail. Takens originally defined the delay-coordinate map forward in time, i.e., as $\Phi^+_{\phi,F,p,\tau}(X) = (\phi(X), \phi(F(X)), \dots, \phi(F^{p-1}(X)))^T$, but the formulation of Theorem 1.5 is equivalent since $\Phi^+_{\phi,F,p,\tau}$ applied to the observable $\phi \circ F^{-(p-1)} \in \mathcal{C}^r(\mathcal{M}, \mathbb{R})$ gives the delay-coordinate map defined in Definition 1.4.

Takens' Theorem connects the differential-geometric result of Whitney to dynamics. It guarantees a topological equivalence between the original system and one that is constructed from the memory of an observable. The theorem allows us to formulate the dynamics on the set $\Phi_{\phi,F,p,\tau}(\mathcal{M})$. Denote by $\hat{x}_t = \Phi_{\phi,F,p,\tau}(X_t)$ a point on $\Phi_{\phi,F,p,\tau}(\mathcal{M})$. We further write Φ for $\Phi_{\phi,F,p,\tau}$ when the subscripts are not decisive. Then the dynamics, which we call the *embedded dynamics*, on these coordinates are given by

$$\hat{x}_{t+1} = \hat{f}(\hat{x}_t) := \Phi \circ F \circ \Phi^{-1}(\hat{x}_t). \tag{1.3}$$

An illustration is shown in Figure 1.2. Clearly, the first coordinate of $\hat{f}(\hat{x}_t)$ is equal to

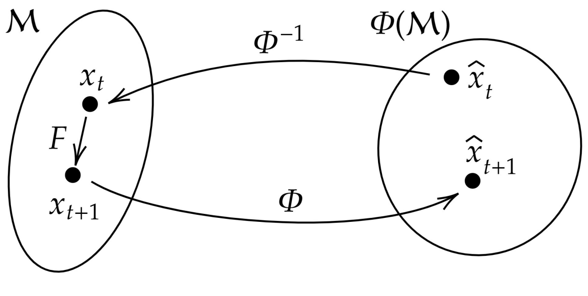


Figure 1.2: Dynamics on $\Phi(\mathcal{M})$.

x_{t+1} by definition of Φ. We therefore define the function $\hat{f}_1 : \Phi(\mathcal{M}) \to \mathbb{R}$ as $\hat{f}_1(\hat{x}_t) = \hat{f}(\hat{x}_t)_1$ which directly constructs the next value of the observable. If $\phi \in \mathbb{R}^m$ is multivariate, we define $\hat{f}_1$ as the first m coordinates of $\hat{f}$. Note that the multivariate case is not covered by Takens' Theorem but will be briefly discussed later.

Since both F and Φ are diffeomorphisms, the same holds for $\hat{f}$. This implies that from one observable alone one is generically able to reconstruct essential properties of the dynamics, such as its Lyapunov exponents, number and nature of its fixed points, and the dimension of the manifold. Note that if F is unknown – which should generally be assumed, since otherwise one could typically draw most desired conclusions from F directly – Takens' Theorem does not permit a precise model form. We will discuss various ways to estimate it in Chapter 2.

In his paper, Takens refers to two other publications that were made slightly earlier. One is by Packard et. al. [PCFS80], where it is discovered from an empirical point of view that the geometry of an attractor can be recreated from the delay embedding of an observation or by using derivatives of the observation of different degrees. The authors demonstrate this for an example system but do not prove any theoretical statements. The second publication is by Aeyels [Aey81] and already discusses the embedding dimension of $2d + 1$. It, however, is concerned with *observability* of a system, meaning that a delay-coordinate map is injective, in contrast to an embedding as Takens showed. Aeyels further discusses the topic from a control-theoretic point of view. These results point into the direction into which Takens advanced, but as Takens politely points out, his result is more – he writes "in some sense somewhat" – general, since it gives precise statements on the conditions under which a delay-coordinate map is not just injective but an embedding.

Note that through Takens' Theorem, we can install dependence of other variables on observations. Assume there exists a relation $Y = G(X)$, then, similarly to Eq. (1.3), we find that

$$Y_t = G(X_t) = G \circ \Phi^{-1}(\hat{x}_t). \tag{1.4}$$

Remember that generally neither of the functions might be known so that the full function $G \circ \Phi^{-1}$ must be estimated. The topological properties of this function, however, are guaranteed to be the same as those of G, if Φ is an embedding, implying that if G is "nice" in a certain sense, than x_t can be mapped to Y_t in a similarly nice way. We will exploit this later on in Chapter 5.

Periodic points

In his proof, Takens specifies the conditions under which F can allow $\Phi_{\phi,F,p,\tau}$ to be an embedding: there must be no periodic orbits with period equal to τ. The reason for this is that it would imply

$$F(X_t) = F(X_{t+\tau}) \tag{1.5}$$

so that

$$\Phi_{\phi,F,p,\tau}(X_t) = (\phi(X_t), \ldots, \phi(X_t))^T. \tag{1.6}$$

With this, an orbit would be mapped to a segment of the diagonal line in $\mathbb{R}^p$. As a consequence, Φ would not be injective and hence not an embedding. Similarly, although following from a slightly more complicated argumentation, there must be no periodic orbits of period 2τ. For periodic orbits with period $k\tau$ for $k \geq 3$ it suffices if their number is finite. Takens showed that almost all, not just generic, choices for F that fulfil these conditions allow for an embedding under the delay-coordinate map. Still, the theorem cannot be formulated without using the term generic since the choice of the observable makes a difference. For example, constant functions do usually not give an embedding since the manifold would be mapped to a single point.

Note that Takens' Theorem can be seen as an extension of Whitney's Theorem in the sense that Whitney merely ensures that generic choices of $(2d+1)$-dimensional functions are embeddings but not which. Takens derived that the specific choice of $\Phi_{\phi,F,2d+1,\tau}$ is an embedding (although this is itself only a generic property). A result that forms a hybrid between both theorems was proved in [DS11] in 2011. It states that by using multiple different scalar-valued observation functions instead of one and applying the delay-coordinate map to each one with potentially pairwise different memory depths also generically gives an embedding. The sum of these individual memory depths then has to be bigger than $2d$. Further, the observables must show a suitable degree of independence. A similar result was stated as a remark already by Sauer et. al. in [SYC91], a publication that makes several seminal contributions to the topic. Their observation directly points towards multidimensional observables, since these can be interpreted as a collection of several scalar-valued observables.

What Takens' Theorem gives us is that typically the delay-coordinate map of already a scalar-valued observation of a dynamical system is enough to create topologically equivalent dynamics when the embedding dimension is bigger than twice the dimension of the manifold that the dynamics are on. Still, there are several questions it does not answer, e.g., the form of the dynamics on $\Phi(\mathcal{M})$ or a lower instead of an upper bound on the required embedding dimension.

Nor does it imply geometric properties of $\Phi(\mathcal{M})$ or the embedded dynamics $\hat{f}$. Indeed, since Φ is a diffeomorphism and hence differentiable, we know that neighbourhoods on $\mathcal{M}$ must map to neighbourhoods on $\Phi(\mathcal{M})$ so that

$$\frac{\|\Phi(X) - \Phi(X')\|}{\|X - X'\|} < \infty, \tag{1.7}$$

but bounds for the amount of perturbation of distances are desirable, e.g., a statement of the form

$$l \leq \frac{\|\Phi(X) - \Phi(X')\|}{\|X - X'\|} \leq u, \tag{1.8}$$

especially in regard to the dynamics $\hat{f}$ as in

$$l \leq \frac{\|\Phi(X) - \hat{f}(\Phi(X))\|}{\|X - F(X)\|} \leq u. \tag{1.9}$$

Eq. (1.8) would shed light on the amount to which, e.g., observations that are contaminated with noise would affect the reconstruction. (1.9) refers to the speed of the embedded compared to the full dynamics, e.g., can the embedded take a long step while the full dynamics only take a small step and vice versa.

Strange attractors

In some cases, the necessary embedding dimension can be reduced. Work in [SYC91] yields that for *strange attractors*, i.e., those with *fractal*, or non-integer, dimension, an equivalent statement as Takens' Theorem can be made. This is important if the dynamics evolve not in a manifold but in a set that lies inside only high-dimensional manifolds while the set itself is much simpler. Then, as shown in [SYC91], the upper bound for the embedding dimension is determined by the geometry of the set and not the manifold.

For this, we use the so-called *box-counting dimension* as a measure of the dimensionality of sets.

Definition 1.6 (Box-counting dimension)**.** The box-counting dimension of a compact set A is defined as

$$boxdim(A) = \lim_{\varepsilon \to 0} \frac{-\log(N_A(\varepsilon))}{\log(\varepsilon)}, \tag{1.10}$$

where $N_A(\varepsilon)$ is the minimal number of cubes with edge length ε needed to cover A.

The box-counting dimension measures the rate in which the minimal number of boxes needed to cover a set increases when decreasing their size. If A is in fact a d-dimensional manifold, then $N_A(\varepsilon)$ is proportional to ε^{-d} so that $boxdim(A) = d$. For sets that are not a manifold, this number is generally non-integer.

The authors of [SYC91] additionally use the notion of *prevalence*, which in finite-dimensional spaces means that the complement of a set has zero measure. With this, they managed to prove a variant of Takens' Theorem with slightly stronger conditions on the periodic orbits of the full dynamics F but with potentially lower embedding dimension:

Theorem 1.7 (Fractal Delay Embedding Prevalence Theorem, 1991, [SYC91])**.** Let F be a flow on an open subset $U \subset \mathbb{R}^d$ and let A be a compact subset of U of box-counting dimension d_{boxdim}. Let $p > 2d_{boxdim}$ be an integer, and let $\tau > 0$. Assume that A contains at most a finite number of equilibria, no periodic orbits of F of period τ or 2τ, at most finitely many periodic orbits of period $3\tau, 4\tau, \ldots.p\tau$, and that the linearizations of those periodic orbits have distinct eigenvalues. Then for almost every smooth function ϕ on U, the delay coordinate map $\Phi_{\phi,F,p,\tau} : U \to \mathbb{R}^p$ is injective on A and an immersion on each compact subset C of a smooth manifold contained in A.

This theorem, although seeming like only a slight modification of Takens' original theorem, provides strong practical benefits. Firstly, it replaces the term generic with almost every. Moreover, instead of being only valid for dynamics on compact, smooth manifolds, this theorem holds for sets in $\mathbb{R}^d$ with a box-counting dimension that might be much lower than d. For example, let F drive a dynamical system on a set A inside $\mathbb{R}^d$ while $boxdim(A) \ll d$. Then Takens' Theorem forces us to choose the embedding dimension as $2d + 1$ while Theorem 1.7 allows us to use the much smaller embedding dimension of $2boxdim(A) + 1$. As pointed out in [SYC91], the result of the delay-coordinate map being injective and an immersion on each compact subset is only slightly weaker than the embedding property.

This perspective becomes immediately helpful when considering one of the most prominent examples for the implications of Takens' Theorem: the Lorenz-63 system. The Lorenz-63 system describes three-dimensional dynamics, whose trajectories form a butterfly-shaped attractor. It has several properties that showcase the usefulness and shortcomings of Takens' Theorem. Introduced by and named after the American mathematician and meteorologist Edward Lorenz, the Lorenz-63 system is a model for atmospheric convection. Depending on initial conditions and parameters, it can show strongly chaotic behaviour (meaning that trajectories from two different initial values diverge form each other exponentially quickly) and is thus a frequently chosen example for such systems. It is given by

$$\frac{\mathrm{d}x}{\mathrm{d}t} = \sigma(x - y), \quad \frac{\mathrm{d}y}{\mathrm{d}t} = x(\rho - z) - y, \quad \frac{\mathrm{d}z}{\mathrm{d}t} = xy - \beta z. \tag{1.11}$$

A trajectory is depicted in Figure 1.3. From Takens' Theorem, an embedding dimension of 7 should be chosen so that a delay-coordinate map of observations of the system are an embedding. The Lorenz attractor is a strange attractor with a box-counting dimension of approximately 2.06, so that Theorem 1.7 implies that already an embedding dimension of 5 should suffice. For this system, it can be shown that already an embedding dimension of 3 is enough for the delay-coordinate maps of the x- and y-coordinates to give embeddings (remember that $2d + 1$ is only an upper bound). This does not hold for the z-coordinate: as can straightforwardly be computed, the Lorenz system has fixed points inside the two lobes of the butterfly-shaped attractor. When using a delay-coordinate map of the z-coordinate, both fixed points are collapsed into one, yielding that $\Phi_{z,F,p,\tau}$ is no embedding for any p. By choosing as observation only a slight perturbation of the z-coordinate – this is guaranteed by both the genericity condition in Takens' Theorem and the almost every

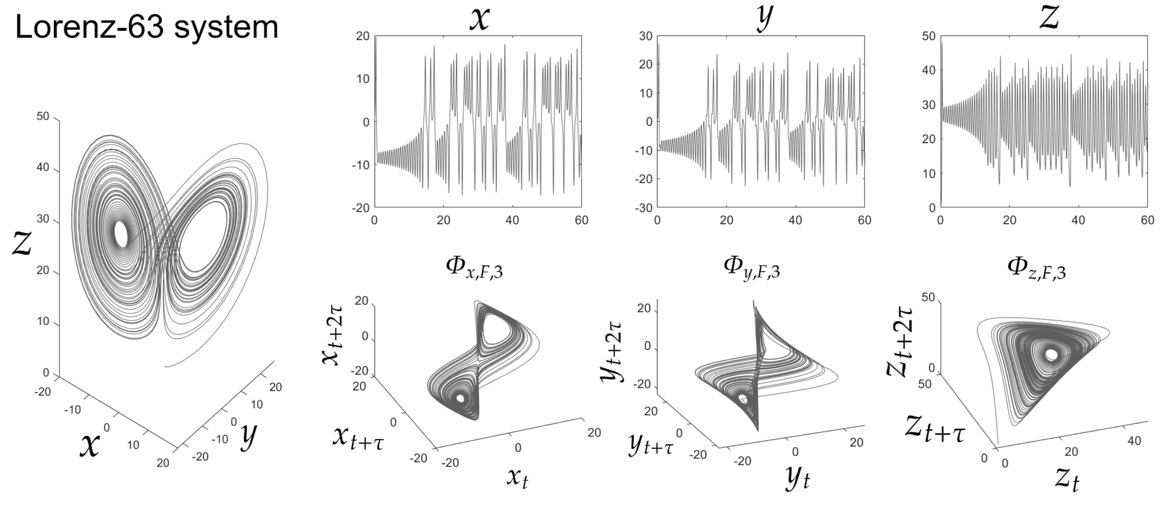


Figure 1.3: The Lorenz-63 system. Full system (left) and individual coordinates along with three-dimensional delay embeddings with $\tau = 0.1$.

statement in Theorem 1.7 – one can obtain an embedding again. Note that the delay embeddings of x and y with sampling rate $\tau = 0.1$ are similar in shape to the original attractor. This is a fortunate artefact of the Lorenz-63 system but, as mentioned before, this does not have to be the case. To see this, one may choose as observation $\phi(x, y, z) = z + \frac{x}{2}$ or the x-coordinate with the sampling rate $\tau = 0.01$ or $\tau = 0.2$, which all give an embedding but very different shapes.

We have now seen that the upper bound in the embedding dimension from Takens' Theorem is optimal neither in theory nor in practice. However, the simple example shown in Figure 1.4 illustrates that even Theorem 1.7 cannot be made with a lower condition for the embedding dimension. The figure shows dynamics on an at least one-dimensional set. Using the x-coordinate as observation yields an overlap in the trajectory of the delay-coordinate map with embedding dimension of 2. As a consequence, the dynamics under the delay-coordinate map are not injective and not even well-defined at the point of the intersection. Small perturbations to the observable would not remedy this.

Filtered delay embeddings

The authors of [SYC91] further prove results on *filtered delay embeddings*. They argue that using an embedding dimension w that is much higher than theoretically necessary can have the benefit of reducing the influence of noise in observed data. However, it also yields additional cost in most ensuing computations. The authors of [SYC91] show that for $p > 2boxdim(A)$ and a rank-p-matrix $B \in \mathbb{R}^{p \times w}$, the filtered delay embedding

$$\Phi_{\phi,F,p,\tau,B}(X) = B\Phi_{\phi,F,w,\tau}(X) \in \mathbb{R}^p \tag{1.12}$$

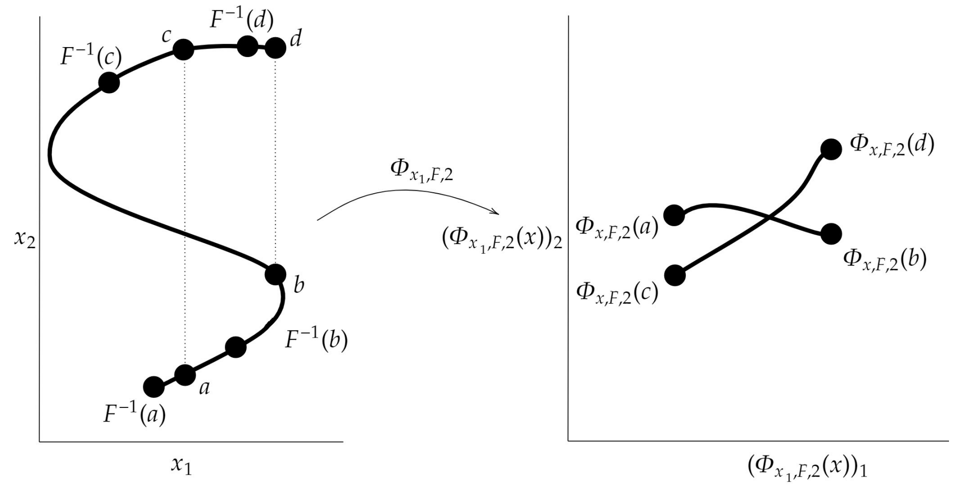


Figure 1.4: Taking the x_1-coordinate as the observation from the two-dimensional dynamics on the left results in an overlap between two segments of the trajectory of the delay-coordinate map. The trajectory on the left has box-counting dimension 1 but an embedding dimension of 2 is apparently not sufficient. 3 should suffice. Example taken from [SYC91].

still is an embedding as long as stricter conditions on the periodic orbits of F are fulfilled.

Theorem 1.8 (Filtered Delay Embedding Prevalence Theorem, 1991, [SYC91]). Let U be an open subset of $\mathbb{R}^k$, F be a smooth diffeomorphism on U, and let A be a compact subset of U, $boxdim(A) = d_{boxdim}$. For a positive integer $p > 2d_{boxdim}$, let B be a $(p \times w)-$matrix of rank p. Assume F has no periodic points of period less than or equal to w. Then for almost every smooth function ϕ, the delay coordinate map $\Phi_{\phi,F,p,\tau,B} : U \to \mathbb{R}^p$ is injective on A and an immersion on each closed subset C of a smooth manifold contained in A.

This theorem allows us to use a large embedding dimension and compress the delay embedding back to a lower-dimensional space.

Systems with control and stochastic systems

Although Theorems 1.5 and 1.7 imply that delay-coordinate maps of most – meaning generic, respectively almost all – choices for F and ϕ give embeddings, in general this statement has to be proved for specific sets of functions. Let us consider forced systems, i.e., dynamical systems of the form

$$\begin{aligned} X_{t+1} &= F(X_t, Y_t) \in \mathcal{M} \\ Y_{t+1} &= G(X_t, Y_t) \in \mathcal{N}. \end{aligned} \tag{1.13}$$

As an example why the theorems that were stated until now do not have satisfying implications for such systems, let us consider the following example (taken from

[Sta99]). Let

$$\frac{\mathrm{d}X_t}{\mathrm{d}t} = \Psi(X_t, t), \quad \Psi(\cdot, t) = \Psi(\cdot, t+T) \tag{1.14}$$

describe a non-autonomous dynamical system with a period of T for each fixed state. We observe the state of the system after time intervals of length τ, so that we can write this with respect to the time-τ-flow F as

$$X_{t+1} = F(X_t, t), \quad F(\cdot, \frac{t}{\tau}) = F(\cdot, \frac{t+T}{\tau}). \tag{1.15}$$

This can be written as a forced system by

$$\begin{aligned} X_{t+1} &= F(X_t, \theta_t T) \\ \theta_{t+1} &= (\theta_t + \frac{\tau}{T}) \mod 1 \end{aligned} \tag{1.16}$$

When asking whether an observable of the system generates a delay embedding, the answer from Theorems 1.5 and 1.7 is yes – for generic, respectively almost all, functions that observe both X and the time parameter θ. However, in practice, an observation generally only measures a quantity of the state and is independent of the time. This case is not covered by Theorems 1.5 and 1.7 and, therefore, it is unclear whether it is possible to generate an embedding by applying the delay-coordinate map to an observation of the state.

In more formal terms: the set of $((F, G), \phi)$ for which the $\Phi_{\phi,(F,G),p,\tau}$ are embeddings is generic in $\mathcal{D}(\mathcal{M} \times \mathcal{N}) \times \mathcal{C}(\mathcal{M} \times \mathcal{N}, \mathbb{R})$. What we seek are conditions under which the set of (F, ϕ) is generic in $\mathcal{D}(\mathcal{M} \times \mathcal{N}, M) \times \mathcal{C}(\mathcal{M}, \mathbb{R})$.

An answer was given in [Sta99] in the form of the following theorem:

Theorem 1.9 (Takens' Theorem for Forced Systems, 1999, [Sta99]). Let $\mathcal{M}$ and $\mathcal{N}$ be compact manifolds of dimensions d and e, respectively. Suppose that the periodic orbits of period smaller than $2p$ of $G \in \mathcal{D}^r(\mathcal{N})$ are isolated and have distinct eigenvalues, where $p > 2(d+e)$. Then for $r \geq 1$, there exists a generic set of $(F, \phi) \in \mathcal{D}^r(\mathcal{M} \times \mathcal{N}, \mathcal{M}) \times \mathcal{C}^r(\mathcal{M}, \mathbb{R})$ for which the map $\Phi_{\phi,F,p}$ is an embedding.

Thus, under certain conditions on the forcing G, we are in fact generically able to construct an embedding of the forced system with a delay-coordinate map. For the example above, this means: if the period T is long enough, Theorem 1.9 implies that generic delay-coordinate maps of an observation of the state X_t will be an embedding. What if T is too short? In this case, if possible, one could decrease the length of τ, i.e., take observations more frequently. This directly connects the theoretical perspective, dominated by the theorems, with the more practical view for which we may ask how different choices for parameters, e.g., the embedding dimension and the sampling rate, influence the practicality of delay-coordinate maps. We will discuss this shortly.

Theorem 1.9 is of immediate help when stating a similar result for stochastic systems. As such a system we denote

$$X_{t+1} = F(X_t, \omega_t) \tag{1.17}$$

where $\omega_t \in \mathcal{N}$ is randomly drawn under a law μ from the set $\mathcal{N}$. Clearly, this is a forced system so Theorem 1.9 could apply. However, the ω_t are not generated under a diffeomorphism G as above but stochastically drawn from $\mathcal{N}$. As such, one could write $\omega_{t+1} = G(\omega_t)$ but G does not have to be injective, since $\omega_t = \omega_{t'}$ might occur. To remedy this problem, we may use a *shift space*, meaning that we define

$$\bar{\omega} = [\dots, \omega_{-1}, \omega_0, \omega_1, \dots] \in \mathcal{N}^{\mathbb{Z}}, \tag{1.18}$$

and equip it with the shift map

$$\sigma(\bar{\omega}_t)_i = (\bar{\omega}_t)_{i+1}. \tag{1.19}$$

We can then write Eq. (1.17) as

$$\begin{aligned} X_{t+1} &= F(X_t, (\bar{\omega}_t)_0) \\ \bar{\omega}_{t+1} &= \sigma(\bar{\omega}_t). \end{aligned} \tag{1.20}$$

where now $\bar{\omega}_0$ is drawn from $\Sigma := \mathcal{N}^{\mathbb{Z}}$ according to a law μ_Σ. In the case of independence of the ω_t, this would be the product measure.

With this, we have drawn all stochastic terms in advance and transformed the stochastic system into a deterministic system. The manifold Σ that contains the $\bar{\omega}_t$ is compact if $\mathcal{N}$ is compact by the so-called Tychonoff's Theorem. Therefore, we could apply Theorem 1.9 here and derive that for generic choices of F and ϕ, we obtain an embedding. However, Σ is infinite-dimensional, making the theorem as it is unusable.

Nevertheless, the infinite dimension of Σ is not transported further, since the delay-coordinate map $\Phi_{\phi,F,p}$ depends on only finitely many components of $\bar{\omega}$. This enabled the authors of [SBDH03] to modify Theorem 1.9 and derive the following result.

Theorem 1.10 (Taken's Theorem for Stochastic Systems, 2003, [SBDH03]). Let $\mathcal{M}$ and $\mathcal{N}$ be compact manifolds of dimension $d \geq 1$ and e respectively and let μ_Σ be an invariant measure on $\Sigma = \mathcal{N}^{\mathbb{Z}}$ that is absolutely continuous with respect to Lebesgue measure on $\mathcal{N}^{p-1}$. Suppose that $p > 2d$. Then for $r \geq 1$, there exists a residual set of $(F, \phi) \in \mathcal{D}^r(\mathcal{M} \times \mathcal{N}, \mathcal{M}) \times \mathcal{C}^r(\mathcal{M}, \mathbb{R})$ such that for any (F, ϕ) in this set $\Phi_{\phi,F,p}$ is an embedding for μ_Σ almost every $\bar{\omega}$.

A residual set is the complement of a countable union of nowhere dense sets. Note that the embedding dimension does not depend on the dimension of $\mathcal{N}$. With this, Theorem 1.10 provides a strong result on the applicability of delay-coordinate maps on stochastic systems.

If $\mathcal{N}$ consists of finitely many points, this theorem has immediate implications. In this case "almost every" is equivalent to "every". Further, the authors of [SBDH03] show that in this case the term "residual set" can be replaced by "generic" which is a stronger statement.

Theorem 1.11 (Taken's Theorem for Iterated Function Systems, 2003, [SBDH03])**.** Let $\mathcal{M}$ and $\mathcal{N}$ be compact manifolds of dimension $d \geq 1$ and $e = 0$ respectively. Suppose that $p > 2d$ and $r \geq 1$. Then there exists an open and dense set of $(F, \phi) \in \mathcal{D}^r(\mathcal{M} \times \mathcal{N}, \mathcal{M}) \times \mathcal{C}^r(\mathcal{M}, \mathbb{R})$ such that for any (F, ϕ) in this set $\Phi_{\phi,F,p}$ is an embedding for every $\bar{\omega}$.

Note that there exist various other variants of Takens' Theorem that loosen some of the restrictions. We have seen already which conditions we have to enforce if we want to apply the result of the theorem to specific dynamics, e.g., on compact sets instead of manifolds, time-dependent or stochastic systems. It was also mentioned that the theorem can be modified to permit multivariate observables [DS11]. In [Rob05] and [Gut16], conditions are shown for which the delay-coordinate map of a non-differentiable but Lipschitz continuous observable is injective (it can naturally not be smooth if the observable is not smooth but injective is a strong property in its own right – it yields that we can at least invertably map between the original system and the delay embedding).

This being said, in future years there is a solid amount of work left in proving theorems which cover all different scenarios that might exist in regard to F and ϕ. As of now, one is likely to face a lack of "combinations" between the different extensions of Takens' Theorem: there do exist statements on smooth multivariate observables and on Lipschitz continuous scalar-valued ones, but what about multivariate, Lipschitz continuous observables? Although some of these "combinations" seem straightforward to prove if understanding of the individual proofs exists, it would be commendable if in the near future more theorems of this style could be proved.

1.1.2 Finding Embedding Parameters

Up to this point, we have summarized results which allow generation of topologically equivalent dynamics using the delay-coordinate map of an observable. We also discussed under which conditions on the dynamics and the observable the delay-coordinate map is an embedding. These conditions, to a good part, refer to the embedding dimension and sampling rate. Naturally the question arises: how do we find these parameters if we do not assume them to be known? There exist practical methods which take as input a *time series* of the system, i.e. states $\phi(X_0), \ldots, \phi(X_N)$

from a trajectory of $\phi(X_t)$, representing either $N+1$ subsequent observations of a time-discrete system or observations sampled at $N+1$ time steps of a fixed length τ over a time span of length $N\tau$. The case of time series with non-uniform sampling rate is discussed in [JM98].

As it turns out, geometric properties such as the bounds in Eq. (1.8) are indeed affected by the choices of p and especially τ.

The embedding dimension

So far, we have argued that we either know the dimension of the manifold of the dynamics or we can compute the box-counting dimension of their domain. However, if we can only access an observable of the dynamics, it is generally non-trivial to find out the dimension of the manifold of the true dynamics. Plus, computation of the box-counting dimension is very expensive for high-dimensional dynamics. Therefore, the need for a simpler method to compute the embedding dimension arises.

False Nearest Neighbours For this reason is now explained the *false nearest neighbour method* (FNN) [KBA92]. It is based on the assumption that embeddings generally preserve neighbourhoods, i.e., points that are close in one set should be close in the embedded set.

As a consequence, if the embedding dimension is large enough for Φ to be an embedding, a point $X_t \in \mathcal{M}$ should have the same neighbours in $\Phi(\mathcal{M})$ for increasing embedding dimension. If, however, the embedding dimension is lower than required for an embedding, this does not have to be the case. Then the geometric structure of the manifold is not preserved and points that are close to each other in $\Phi(\mathcal{M})$ might not have been neighbours in $\mathcal{M}$. This is utilized in the following way:

Define the squared Euclidean distance between two points in $\Phi_p(\mathcal{M})$ by

$$D_p(\Phi_p(X_t), \Phi_p(X_t)') := \|\Phi_p(X_t) - \Phi_p(X_t)'\|_2^2. \tag{1.21}$$

Let $\Phi_p(X_t)'$ be the nearest neighbour of $\Phi_p(X_t)$, i.e., the minimizer of $D_p(\Phi_p(X_t), \cdot)$. Then compare this term with the distance in the delay embedding for the embedding dimension increased by 1 and compute

$$\frac{|D_{p+1}(\Phi_{p+1}(X_t), \Phi_{p+1}(X_t)') - D_p(\Phi_p(X_t), \Phi_p(X_t)')|}{D_p(\Phi_p(X_t), \Phi_p(X_t)')} = \frac{\|\phi(X_{t-p}) - \phi(X_{t'-p})\|_2^2}{D_p(\Phi_p(X_t), \Phi_p(X_t)')}. \tag{1.22}$$

If this ratio exceeds a certain tolerance threshold R_{tol}, then apparently these neighbours in $\Phi_p(\mathcal{M})$ are not neighbours in $\Phi_{p+1}(\mathcal{M})$ and are therefore deemed false nearest neighbours. Once the necessary embedding dimension is reached, neighbours should remain neighbours so that the number of points which violate (1.22) – the false nearest neighbours – should plateau at a small value.

Using the box-counting dimension Another way to estimate a sufficient embedding dimension is to use the fact that embeddings preserve the dimension of manifolds (while this is not true for the box-counting dimension of sets). If a delay-coordinate map is in fact an embedding then increasing the embedding dimension will still lead to an embedding. Thus, the box-counting dimension of the delay embedding will plateau, once an embedding is generated. These same should approximately happen when using the box-counting dimension. As illustrated in [GWSP82], it can be both difficult and expensive to compute for high-dimensional sets.

The sampling rate

We have seen in the conditions on the periodic orbits that the sampling rate τ has to avoid certain values, namely certain multiples of period of periodic orbits of the dynamics. But the sampling rate has a practical effect on the delay embedding even apart from these periods as we will observe shortly. The aim in choosing τ is to make the different memory terms contain as much independent information as possible. For a trajectory a simple method to achieve this is by minimising the *autocorrelation function*. Let us assume here, for notational purposes, that we possess a time series $x_0, x_v, x_{2v}, \ldots, x_T \in R$ and we seek to choose τ as a prefactor for the fine sampling rate v (this is a slight change in notation: we seek $\tau \in \mathbb{N}$ and not from $\mathbb{R}^+$ as in the theoretical setting but assume we have access to measurements of the system, separated by the time v). Any sampling rate should then be a multiple of v. The autocorrelation function – for scalar-valued observables – is given by

$$A(\tau) = \frac{1}{N-\tau} \sum_{i=1}^{N-\tau} x_i x_{i+\tau v}. \tag{1.23}$$

The value for τ that minimizes this term minimizes the linear dependence between time-shifted states of X. For multivariate observables one would receive an autocorrelation matrix and would optimize the sampling rate according to a suitable metric of this matrix.

An alternative is given by minimizing the *average mutual information* (AMI) [FS86, AMRT01]. The AMI is a nonlinear function that measures the amount of information between time-shifted states of a trajectory by

$$AMI(\tau) = \sum_{i=0}^{N-\tau} \hat{p}(x_i, x_{i+\tau v}) \log_2 \Big(\frac{\hat{p}(x_i, x_{i+\tau v})}{\hat{p}(x_i)\hat{p}(x_{i+\tau v})} \Big). \tag{1.24}$$

$\hat{p}(x_t)$ is the empirically estimated probability of the occurrence of x_t, e.g., as the relative frequency of x_t being inside a certain bin of a histogram. If x_t and $x_{i+\tau v}$ are independent of each other, $AMI(\tau)$ will be 0, which is its minimal value.

Since the AMI is nonlinear, it should generally be a stronger method to find τ that gives maximal independence than the autocorrelation function which only detect linear independence. However, unlike the autocorrelation, the computation of the AMI is subject to how one estimates the probability density $\hat{p}$ and should therefore only be employed if a sufficiently long time series is provided. For multivariate observables, it can only entail infeasible computational cost.

Although it might seem intuitive to choose the sampling rate so that subsequent observations are independent, there exists surprisingly little information in the literature as to whether the minimizers of the autocorrelation function and the AMI *guarantee* any properties, not even the necessary one regarding the periodic orbits. In the absence of theoretical answers to such questions until recently [EYWR18], there does exist a strong body of experimental research that sheds light on them, e.g., [PD20, DBGM20] to only name a few. This research indicates that the geometry of the image of the delay-coordinate map strongly depends on τ (which will have practical implications later on). In essence, we can already observe in Figure 1.5 the effect of different sampling rates on delay embeddings of the Lorenz-63 system.

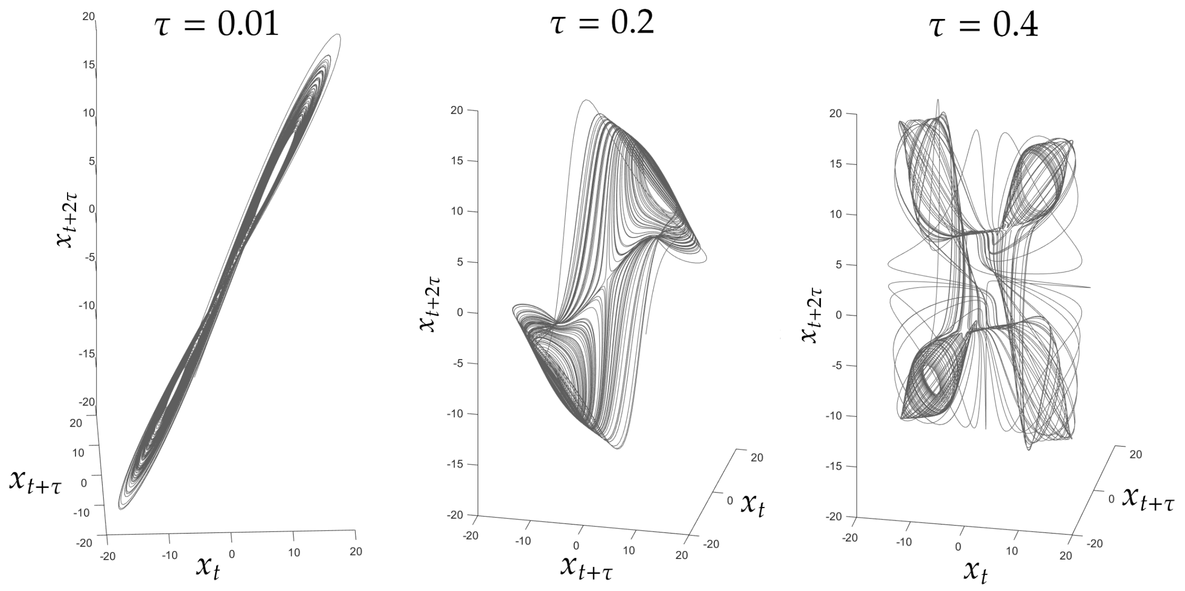


Figure 1.5: Delay embeddings of the x-coordinate of the Lorenz-63 system with three different sampling rates. Although the attractors are topologically equivalent, their geometries are very different from each other.

If the geometry of a manifold is not preserved under the delay embedding, then close points in the original system could be mapped far apart in the delay embedding and vice versa. As a consequence, small perturbations, e.g., if the observation is subject to noise, may significantly affect the delay embedding. This, of course, would be a contradiction to the assumption made in the FNN method, which relies on neighbourhoods to be preserved under a delay embedding. Thus, – in extreme cases – the FNN method is not practical (although for most dynamical systems investigated in the literature, it is). In this light, it immediately becomes desirable to possess statements about metric properties, i.e., under which conditions distances or at least the ratios between them are (approximately) preserved. Note that an embedding is bi-Lipschitz continuous, meaning that Eq. (1.8) holds and distances cannot be arbitrarily strongly perturbed. An exact quantification, however, e.g., by specifying the bounds u and l, is desirable.

With these objections in mind, the presented methods to find embedding parameters do not truly give reliable quantitative statements but may appear rather vague, although being common practice. The following results will lead to more precise statements about good choices for τ and p.

1.1.3 Metric Properties of Delay Embeddings

In [EYWR18], a theorem is proven that gives for each u and l the probability with which an observation function that is drawn under a specific randomness gives an embedding that fulfils Eq. (1.8). Such a delay embedding is then called a *stable embedding* if $u \approx l$. From this, the authors derive a way to determine embedding dimension and sampling rate needed for a stable embedding.

The authors construct an observation ϕ as a linear combination of basis functions $h_1, \dots, h_L$ with coefficients from a vector $\alpha \in \mathbb{R}^L$ that is drawn from a certain subgaussian distribution, i.e.,

$$\phi(X) = \alpha[h_1, \dots, h_L]^T(X). \tag{1.25}$$

For an embedding dimension p, they introduce the matrix

$$A_{h,\tau,p}(X) = \left[\Phi_{h_1,F,p,\tau}, \dots, \Phi_{h_L,F,p,\tau}.\right] \in \mathbb{R}^{p \times L} \tag{1.26}$$

and define the *stable rank* of a matrix as

$$R(A) = \frac{\|A\|_F^2}{\|A\|^2} = \frac{\sum_{i=1}^{p} \sigma_i^2}{\sigma_1^2} \tag{1.27}$$

where σ_i is the ith biggest singular value [GHS87] of A. They then define the stable rank of the manifold $\mathcal{M}$ as

$$R_{h,\tau,p}(\mathcal{M}) = \inf_{X,Y \in \mathcal{M}, X \neq Y} R(A_{h,\tau,p}(X) - A_{h,\tau,p}(Y)). \tag{1.28}$$

From this, they prove a theorem that – roughly – states that if

$$R_{h,\tau,p}(\mathcal{M}) \gtrsim \dim(\mathcal{M}) \log\left(\frac{vol(\mathcal{M})^{\frac{1}{\dim(\mathcal{M})}}}{rch(\mathcal{M})}\right) \tag{1.29}$$

they can state a probability with which u and l can be set to certain - close - values. rch is the *reach* of the manifold which can be understood as the inverse of its maximal curvature. The full result is difficult to comprehend without first being acquainted to some differential-geometric terms, but in essence it says: the richer the observation, i.e., the more evenly it captures different variables of the dynamics, the closer are u and l to each other because in this case the stable rank is high. A good choice for the sampling rate τ also contributes to this effect: if it is too small, the rows of $A_{h,\tau,p}(X) - A_{h,\tau,p}(y)$ will be too similar to each other and the stable rank small, too. If τ is similar to the Lyapunov exponents of the dynamics, then, as the authors point out, the rows of $A_{h,\tau,p}(X) - A_{h,\tau,p}(y)$ have different lengths and again the stable rank is small. It is therefore vital to choose a good sampling rate. This will be discussed shortly.

Moreover, the manifold should be of simple form, including that it has a high reach, which means that it is not close to overlapping itself.

Further, the authors present an algorithmic way to determine p and τ so that the delay embedding is in fact stable. They observe that under Eq. (1.8), it holds that

$$(pl)^{\frac{\dim(\mathcal{M})}{2}} \leq vol(\Phi_{\phi,F,p,\tau}(\mathcal{M})) \leq (pu)^{\frac{\dim(\mathcal{M})}{2}}. \tag{1.30}$$

It follows that if $l \approx u$ for a certain range of values for p and τ, then the term $\frac{vol(\Phi_{\phi,F,p,\tau}(\mathcal{M}))}{p^{\frac{\dim(\mathcal{M})}{2}}}$ should remain almost constant over this range. They deduce that vice versa if over a certain range of p and τ this term does stay almost constant, it could indicate that for this range in fact $l \approx u$ and those values for p and τ generate a stable embedding.

Results such as those demonstrated in [EYWR18] help to place methods such as FNN or the AMI minimization on a more solid theoretical basis and even find improvements of them. Further research in this direction should be expected and could be of practical help.

We have now discussed the implications that Takens' Theorem and its variants yield and the formal conditions they demand. They allow us to infer various properties of a dynamical system from only a scalar-valued observation. Plus, we have seen how we can influence the geometric and dynamical properties of delay embeddings. As might have become clear by now – also by the extensive use of the term "properties" –, through the view point taken by Takens we rather gain insight into the structure of the full dynamics and the manifold they are on from the delay-embedded dynamics, than we are able to predict the delay embedded dynamics (we merely know they are given by $\hat{f}$ in Eq. (1.3) but a precise formulation of the dynamics eludes us).

In Chapter 2, we will see how delay embeddings can be put into practice in order to draw a benefit in the understanding of a given dynamical system. For this, the geometric and dynamical properties will play a crucial role. For now, however, we will take a different view point on memory effects in partially observed dynamical systems.

1.2 The Mori–Zwanzig Formalism

In this section, we will approach the modelling of observables of dynamical systems from a different perspective. One the one hand it is, in some sense, more practical then the view point taken by Takens, since it permits an approximative model form of the dynamics. On the other hand, it is not as strongly focused on preserving properties of the dynamics.

The Mori–Zwanzig formalism (MZ) describes a mathematical framework that stems from statistical physics. It was originally introduced in order to separate slow from fast variables in dynamical systems with different time scales but generalizes to a wide range of systems. As we will see, the main benefit it generates is a closed formulation of the dynamics of a multivariate observable. There exists a large body of literature on the topic, including, obviously, the initial source [Zwa01]. Typically, the literature deals exclusively with either the time-continuous setting, e.g., [CHK00, CHK02, LTLA21, HEVEDB10] or the time-discrete one [LL21, CL15, LLC15]. We will focus on the latter.

Again, we consider the scenario of

$$\begin{aligned} \text{dynamics } X_{t+1} &= F(X_t) \in \mathbb{X} \\ \text{with an observable } x_t :&= \phi(X_t) \in \mathbb{Y} \end{aligned} \tag{1.31}$$

with sets $\mathbb{Y} \subset \mathbb{R}^m$ and $\mathbb{X} \subset \mathbb{R}^d$. MZ aims at first constructing an operator that approximates the evaluation of an observable g (that can be different from ϕ) at X with only the information in $\phi(X)$ and second using this operator to find a representation of the dynamics of the observable.

1.2.1 Derivation of the Dynamics of an Observable Using Mori–Zwanzig

Technical preparations for the Mori–Zwanzig formalism

In order to introduce the ideas from the Mori–Zwanzig formalism, we need to make a few technical definitions: the set of functions which map states from $\mathbb{X}$ into $\mathbb{Y}$ –

which ϕ belongs to – is denoted by

$$\mathcal{G} := \{g : \mathbb{X} \to \mathbb{Y}\}. \tag{1.32}$$

Further, we define as $\mathcal{H}_\phi \subset \mathcal{G}$ the subset of observations that depend only on these relevant variables and map to $\mathbb{Y}$, i.e.,

$$\mathcal{H}_\phi := \{h \in \mathcal{G} | \exists \tilde{h} : \phi(\mathbb{X}) \to \mathbb{Y} : h = \tilde{h} \circ \phi\}. \tag{1.33}$$

Functions in $\mathcal{H}_\phi$ still depend on $X \in \mathbb{X}$ but it suffices to know $x \in \mathbb{Y}$ to evaluate them. In other words, for every $X \in \phi^{-1}(x)$, $h(X)$ is identical so that x can be seen as representing an equivalence class. We will therefore write $h(x)$ for $x \in \mathbb{Y}$ to denote evaluating $h(X)$ for an arbitrary $X \in \phi^{-1}(x)$, in other words, evaluating only $\tilde{h}$ at $\phi(X)$. An example is

$$\mathbb{X} = \mathbb{R}^2, \quad \phi(X) = X_1 + X_2, \quad h(X_1, X_2) = (X_1 + X_2)^2 = \phi(X)^2.$$

It is easy to see that $\mathcal{H}_\phi$ is only a small subset of $\mathcal{G}$. For example, consider the setting

$$\mathbb{X} = \mathbb{R}^2, \quad \phi(X) = X_1 + X_2, \quad g(X) = X_1^2 + X_2.$$

Here knowledge only of $\phi(X)$ does not permit a direct computation of $g(X)$. For functions $g \in \mathcal{G} \setminus \mathcal{H}_\phi$, the term $g(x)$ is generally not defined.

For this reason, we define a projection operator $P : \mathcal{G} \to \mathcal{H}_\phi$ which maps functions that depend on X to functions depending on x. Its aim is to approximate $g \in \mathcal{G}$ by $h \in \mathcal{H}_\phi$ so that h is close to g according to a given metric. In addition to P, we define its complement $Q := Id - P$.

An intuitive choice for P is the conditional expectation

$$(Pg)(x) = \mathbb{E}[g(X) \mid \phi(X) = x]. \tag{1.34}$$

This definition of P gives a reasonable answer to our question: it says what we *expect* $g(X)$ to be if we know that $\phi(X) = x$. In order to define the conditional expectation, we assume that states in $\mathbb{X}$ obey an F-invariant probability distribution μ.

An alternative choice for P is the orthogonal projection onto the span of linearly independent functions $\psi := [\psi_1, \dots, \psi_L]$ where $\psi_i \in \mathcal{H}_\phi$ for all i. Then P is given by

$$(Pg)(x) := \psi(x)\langle\psi, \psi\rangle^{-1}\langle\psi, g\rangle \tag{1.35}$$

where the scalar product $\langle\cdot, \cdot\rangle$ is defined for matrix-valued functions $f_1 : \mathbb{X} \to \mathbb{R}^{m \times a}$

and $f_2 : \mathbb{X} \to \mathbb{R}^{m\times b}$ as

$$\langle f_1, f_2 \rangle := \int_{\mathbb{X}} \underbrace{f_1(X)^T}_{\in\mathbb{R}^{a\times m}} \underbrace{f_2(X)}_{\in\mathbb{R}^{m\times b}} \mathrm{d}\mu(X) \in \mathbb{R}^{a\times b}, \tag{1.36}$$

which itself is matrix-valued. The term $\langle \psi, \psi \rangle^{-1}$ is called a mass matrix and guarantees that P is in fact an orthogonal projection. This projection has the property that Pg is the closest function in $span(\psi)$ to g with respect to $\langle \cdot, \cdot \rangle$, i.e., Pg minimizes for all $h \in span(\psi)$,

$$\langle g-h, g-h \rangle = \int_{\mathbb{X}} (g-h)^T(X)(g-h)(X)\mathrm{d}\mu(X). \tag{1.37}$$

which is the *expected squared difference between* g and h.

Note that the conditional expectation minimizes the expression in Eq. (1.37) for all functions in $\mathcal{H}_\phi$ while the orthogonal projection does so for all function in $span(\psi)$. The latter can therefore be seen as a more practical approximation of the former (see [CHK02] for further investigations of this relation). If $span(\psi) = \mathcal{H}_\phi$, both projections are equivalent. In practice, this is unrealistic if $\mathcal{H}_\phi$ is infinite-dimensional since one would need an infinite number of functions. In this case one would choose a sufficiently rich finite set of functions so that $span(\psi) \approx \mathcal{H}_\phi$ or it at least covers the parts of $\mathcal{H}_\phi$ that are of interest. The conditional expectation is often expensive to compute as illustrated in [GGH21]. As we will see, the orthogonal projection leads to a formulation of the dynamics of x_t that can be estimated in a much simpler way.

Representation of the dynamics

The function that we are interested in evaluating is one that transports x_t to x_{t+1}. It is, however, unclear whether such a function actually exists. A function that comes close to this wish is $\phi \circ F$ since it transports X_t to x_{t+1}. However, we assumed not to possess knowledge of X_t and generally $\phi \circ F \notin \mathcal{H}_\phi$. We therefore will need to apply a projection operator P to $\phi \circ F$ to generate a function which transports observations over time. In the form of the diagram below: instead of transporting the dynamics on the full state X by F and then evaluating ϕ at $F(X)$, we only know $\phi(X)$ and want to find $\phi(F(X))$.

$$\begin{tikzcd} X \arrow[r, "F"] \arrow[d, "\phi"'] & F(X) \arrow[d, "\phi"] \\ \phi(X) = x \arrow[r, "?"'] & \phi(F(X)) \end{tikzcd} \tag{1.38}$$

In order to make the framework of projection operators compatible with dynamics, let us introduce the Koopman operator, which will be more extensively discussed in Chapter 2, and which plays a prominent role whenever observations of dynamical systems are investigated:

Definition 1.12 (Koopman operator)**.** The Koopman operator $\mathcal{K} : L^\infty(\mathbb{X}) \to L^\infty(\mathbb{X})$ corresponding to a discrete-time dynamical system $F : \mathbb{X} \to \mathbb{X}$ maps an observable $g : \mathbb{X} \to \mathbb{Y}$ to the function $\mathcal{K} \circ g := g \circ F$.

$L^r(\mathbb{X})$ is the space of r-Lebesgue integrable functions on the bounded space $\mathbb{X}$ for $1 \leq r \leq \infty$. The Koopman operator fulfils

$$(\mathcal{K}^t \phi)(X_0) = (\phi \circ F^t)(X_0) = \phi(X_t) = x_t. \tag{1.39}$$

With these definitions, we consider the so-called Dyson formula

$$\mathcal{K}^{t+1} = \sum_{k=0}^{t} \mathcal{K}^{t-k} P\mathcal{K}(Q\mathcal{K})^k + (Q\mathcal{K})^{k+1}. \tag{1.40}$$

The Dyson formula denotes an iterative splitting of the Koopman operator into different parts $P\mathcal{K}$ and $Q\mathcal{K}$. We now follow the procedure from [WKS21] and [LL21] and apply both sides of Eq. (1.40) to ϕ. Evaluating them at the initial value of the full dynamics X_0 yields

$$\begin{aligned}
(\mathcal{K}^{t+1}\phi)(X_0) &= \sum_{k=0}^{t} \mathcal{K}^{t-k}[P\mathcal{K}(Q\mathcal{K})^k\phi](X_0) + (Q\mathcal{K})^{t+1}\phi(X_0) \\
\text{so that } x_{t+1} &= \sum_{k=0}^{t} [P\mathcal{K}(Q\mathcal{K})^k\phi](x_{t-k}) + (Q\mathcal{K})^{t+1}\phi(X_0) \\
\text{Setting } \rho^k := (Q\mathcal{K})^k\phi \text{ gives: } x_{t+1} &= \sum_{k=0}^{t} [P\mathcal{K}\rho^k](x_{t-k}) + \rho^{t+1}(X_0) \\
&= \sum_{k=0}^{t} [P(\rho^k \circ F)](x_{t-k}) + \rho^{t+1}(X_0).
\end{aligned} \tag{1.41}$$

We could replace X_{t-k} by x_{t-k} since $P\mathcal{K}(Q\mathcal{K})^k\phi \in \mathcal{H}_\phi$.

$\rho^0 = \phi$ gives that $P(\rho^0 \circ F) = P(\phi \circ F)$. This term is typically called the *optimal prediction* term since it is the best memoryless approximation of x_{t+1} because it minimizes Eq. (1.37) for $g = \phi \circ F$. The terms of the form $[P\mathcal{K}\rho^k](X_{t-k})$ for $k \geq 1$ are called *memory terms*, since they use information from previous values of x.

If P is given by the orthogonal projection onto basis functions $\psi_1, \dots, \psi_L$, then we find

$$\begin{aligned}
[P(\rho^k \circ F)(x_{t-k})] &= \psi(x_{t-k})\langle\psi,\psi\rangle^{-1}\langle\psi,\rho^k \circ F\rangle \\
&= \underbrace{\psi(x_{t-k})}_{\in\mathbb{R}^{m\times L}}\underbrace{\langle\psi,\psi\rangle^{-1}}_{\in\mathbb{R}^{L\times L}}\underbrace{\int_{\mathbb{X}}\underbrace{\psi(\phi(X))^T}_{\in\mathbb{R}^{L\times m}}\underbrace{\rho^k(F(X))}_{\in\mathbb{Y}\subset\mathbb{R}^m}\,\mathrm{d}\mu(X)}_{\in\mathbb{R}^L} \qquad (1.42)\\
&=: \psi(x_{t-k})h_k \in \mathbb{R}^m
\end{aligned}$$

with vector-valued coefficients

$$h_k = \langle\psi,\psi\rangle^{-1}\int_{\mathbb{X}}\psi(\phi(X))^T\rho^k(F(X))\mathrm{d}\mu(X). \qquad (1.43)$$

In the absence of information of μ, the integral in h_k is generally not accessible, either. In Chapter 3, we will discuss how it can be approximated.

The term $\rho^{t+1}(X_0)$ in Eq. (1.41) depends on the full state X_0 itself and is often called *noise*, because one does not have explicit access to it and can often only treat it as a stochastic influence. It is generally non-accessible so that we have not yet reached a closed formulation of the dynamics of x_t, i.e., one which only depends on x_t but not on X_t. Approximating the noise term generally provides a challenge. As the authors of [LTLA21] put it, ″identifying the [...] noise model is often from educated guesses supported by domain-specific knowledge″. The degree of difficulty typically depends on properties of F, as is discussed in [LC17, HEVEDB10, KDG15]. Often, $\rho^{t+1}(X_0)$ is replaced by a stochastic noise term $\varepsilon_{t+1} \in \mathbb{Y}$ (as we will do from here onwards), especially by a zero-mean Gaussian random variable as, e.g., in [LL21, LBL16].

The last row of Eq. (1.41) is called the *Mori–Zwanzig equation*. Taking as projection P the orthogonal projection onto basis functions in ψ, we can reformulate it as

$$x_{t+1} = \sum_{k=0}^{t}\psi(x_{t-k})h_k + \varepsilon_{t+1}. \qquad (1.44)$$

Through the Mori–Zwanzig formalism, we have now derived a closed form of the dynamics of the observation x_t. As we can see in Eq. (1.44), future states depend on memory terms. MZ therefore formalises how we can replace the missing information about the full state by memory terms of the observations, similarly to what is suggested by Takens' Theorem and its variants, but derive a more concrete model formulation.

Strength of the influence of memory terms

So far we have seen how through MZ we obtain the structure of the dynamics of the observable but we have not yet discussed the amount of influence that memory terms have on the dynamics compared to the optimal prediction term. More generally, we should ask the question how the influences of the individual memory terms differ from each other. As we will see now, the answer to this question is intimately connected to the richness and relevance of information contained in the observable ϕ.

Case 1: Knowledge of $\phi(X)$ is enough to evaluate $\phi \circ F(X)$ If $\phi(X_t)$ suffices to evaluate $(\phi \circ F)(X_t)$, this simply means that $[P(\phi \circ F)](X_t) = (\phi \circ F)(X_t)$. Thus, for the complement projection it holds that $Q(\phi \circ F) = (Id - P)(\phi \circ F) \equiv 0$. This in turn yields for the Mori–Zwanzig equation that all terms involving $k \geq 1$ vanish, including $\rho^{t+1}(X_0)$, since they contain $Q\mathcal{K}\phi = Q(\phi \circ F) = (Id - P)(\phi \circ F)$. We are then left with

$$x_{t+1} = \phi(F(X_t)) \tag{1.45}$$

which we assumed to be a function depending on x_t.

Example 1.1. Let

$$\phi(X_t) = (X_t)_1 + (X_t)_2, \quad F(X_t) = (-\frac{(X_t)_1 + (X_t)_2}{2}, -\frac{(X_t)_1 + (X_t)_2}{2})^T, \quad X_0 = (1,1)^T. \tag{1.46}$$

Then $F(X_t) = (-\frac{\phi(X_t)}{2}, -\frac{\phi(X_t)}{2})^T$ so that $x_{t+1} = \phi(F(X_t)) = -x_t$. Let us choose as basis functions for the orthogonal projection simply $\psi(x) = x$.

Of course, this example is trivial, but for completion, let us compute $[P(\phi \circ F)(x_t)]$ as in Eq. (1.42): since the full dynamics oscillate between the states $(1,1)^T$ and $-(1,1)^T$, μ simply is given by

$$\mu((1,1)^T) = \mu(-(1,1)^T) = 0.5. \tag{1.47}$$

Observe that

$$\phi((1,1)^T) = \psi(\phi((1,1)^T)) = 2, \quad \phi(-(1,1)^T) = \psi(\phi(-(1,1)^T)) = -2. \tag{1.48}$$

Thus, with $\psi(x) = x$,

$$\begin{aligned}\langle \psi, \phi \circ F \rangle &= \int_{\mathbb{X}} \psi(\phi(X))^T \phi(F(X)) \mathrm{d}\mu(X) \\ &= 0.5 \cdot \psi(\phi((1,1)^T))^T \cdot \phi(-(1,1)^T) + 0.5 \cdot \psi(\phi(-(1,1)^T))^T \cdot \phi((1,1)^T) \\ &= 0.5 \cdot 2 \cdot (-2) + 0.5 \cdot (-2) \cdot 2 = -4.\end{aligned} \tag{1.49}$$

Analogously, it can be shown that $\langle \psi, \psi \rangle = \frac{1}{4}$. Therefore,

$$[P(\phi \circ F)](x_t) = \psi(x_t) \langle \psi, \psi \rangle^{-1} \langle \psi, \phi \circ F \rangle = -x_t. \tag{1.50}$$

This should not come as a big surprise. It simply formalizes that in case our observation already contains all necessary information to describe its evolution, we do not require memory terms.

In this example, the relation between observable and full state are such that knowing the state of the observable directly gives us the full state, making this example especially simple. This does not even have to be the case in order to describe the dynamics of the observable without memory, as the following example shows.

Example 1.2. Consider dynamics given by F and as observable simply $\phi(X) \equiv 0$. Then if the image of the projection operator P contains 0, we find that $[P(\phi \circ F)] = \phi \circ F \equiv 0$. Thus, all memory terms vanish (since $Q(\phi \circ F) = 0$) and it can easily be checked that $h_0 = 0$ so that correctly the Mori–Zwanzig equation reads $x_{t+1} = 0$. Apparently, we cannot reconstruct the full state from the observable but still do not need memory. The point of this example, together with the previous one, is to show that the observable must contain the *relevant* information for its *own* evolution in order to not impose the need for memory terms. In this example, one does not require any information about the full state since the observable maps any state to 0.

In Chapter 4, we will consider a similar example of much higher complexity. However, the two examples considered here are rather extreme cases and chosen in order to build intuition. In most scenarios of given dynamics F and observable ϕ, the MZ equation does not simplify so easily, including the following example.

Case 2: Knowledge of x_t is not enough to evaluate $\phi \circ F(X_t)$

Example 1.3. Consider the setting

$$F(X_t) = \lambda X_t, \lambda \in [0,1), \phi(X) = \frac{1}{10}\lfloor 10X \rfloor, X_0 = 1. \tag{1.51}$$

These dynamics describe a simple exponential decay while the observation simply rounds down the system state to the next-lowest multiple of 0.1. From $\phi(X)$ alone we cannot say whether X is close to a multiple of 0.1 and thus will enter the next "box" in the next time step. Essentially, this example is analogous to the one given at the beginning of this chapter. It then holds that $[P(\phi \circ F)](X_t) \neq (\phi \circ F)(X_t)$ so that the terms ρ^k do not vanish, regardless of the choice of the projection. Then the memory terms do play a role in the dynamics. As can be observed, if $Q(\phi \circ F)$ has a large norm, i.e., if typically the best approximation in $\mathcal{H}_\phi$ for a function $\phi \circ F \in \mathcal{G}$ is far away from it, then the influence of the memory terms is big, too. This also holds for the noise term $\rho^{t+1}(X_0)$. Although the noise term might seem like only a minor addition to the dynamics described by the sum in the MZ equation, if Q is large then this term actually dominates. Intuitively speaking, if ϕ provides us with little information about the state of the full dynamics, we can only model its evolution as stochastic. However, if Q is small then $(Q\mathcal{K})^k$ should decay fast with increasing k, so that only the first few memory terms truly impact the dynamics.

Summary

This analysis should clarify two things:

1. We can approximate the dynamics of an observable by using transformations of memory terms of it.
2. The memory depth that is required for a good approximation of the observed dynamics depends on the observable and the dynamics.

However, it must be stressed that the memory is introduced here in a different way than in Takens' Theorem: through delay embedding, we construct dynamics that (typically) are topologically *equivalent* to the full dynamics if sufficiently many memory terms of the observation are used. In MZ, we have performed two essential steps. The first step was to split up the Koopman operator and arrive at the Mori–Zwanzig equation, i.e., the last row of Eq. (1.41). There already, memory terms of x play a role. But we have merely derived an alternative way to phrase the action of the Koopman operator. After all, we apply an operator – $\mathcal{K}^{t+1}$ – to a function – ϕ – and apply this to a state X_0. To illustrate this: we could write $F(X_t) = (\mathcal{K}^{t+1} \circ F)(X_0)$ with the Koopman operator now defined as applicable to functions mapping from $\mathbb{X}$ to $\mathbb{X}$. Would the dynamics now have memory since X_0 occurs? No, we simply have rewritten the memoryless dynamics of F with an operator that encapsulates its evolution across t time steps, making it dependent on X_0. In the MZ equation, since the noise term is not accessible we cannot gain a closed formulation of the dynamics of x_t. Therefore, a second step was needed in which we *approximated* the MZ equation and obtained Eq. (1.44). This equation now does describe closed dynamics with memory – but they are only an approximation of the true evolution of x_t. Observe also how the two perspectives complement each other and their downsides: Takens gives us that with a finite amount of memory we can construct topologically equivalent dynamics but do not know their structure. MZ yields that, requiring infinite memory, we can specify the structure of the dynamics.

Building on this observation, in practice, the infinite memory depth quickly becomes inapplicable because one would arrive at a large, thus computationally infeasible and usually non-interpretable, representation. Moreover, in every time step one would have to compute a new coefficient. For this reason, one typically seeks to truncate terms of the dynamics starting at a suitably chosen memory depth p and obtain

$$x_{t+1} = \sum_{k=0}^{p-1} \psi(x_{t-k})h_k + \varepsilon_{t+1}. \tag{1.52}$$

This formulation is called *nonlinear autoregressive model*. Future terms of x depend on a fixed number p of memory terms with fixed coefficients $h_0, \dots, h_{p-1}$. It will play a prominent role in the remainder of this thesis.

1.2.2 Extensions and Technical Details of Mori–Zwanzig

We have now discussed and analysed the main steps of the Mori–Zwanzig formalism. For the topics that will be considered later in this thesis, it will be helpful to consider several special cases and reformulations, as we will do now. Now follow several rather technical extensions and modifications of MZ, which can mostly be found in [WKS21], whose results will be deployed later on.

Stochastic systems

Similarly to when we investigated Takens' Theorem, in case the dynamics underlie stochastic influences, in MZ it is sufficient to make several adjustments to the objects involved to use the same argumentation as in the non-stochastic case.

Let us consider stochastic dynamics

$$X_{t+1} = F(X_t, \omega_t) \tag{1.53}$$

where the ω_t are drawn randomly – we assume independent and identically distributed – from Σ according to a law μ_Σ. F is now defined as $F : \mathbb{X} \times \Sigma \to \mathbb{X}$. In this case we attempt to predict the *expected* evolution of ϕ. For this, we define the Koopman operator for stochastic systems as

$$(\mathcal{K} \circ g)(X) = \mathbb{E}_{\mu_\Sigma}[g(F(X, \omega))]. \tag{1.54}$$

Both the sets $\mathcal{G}$ and $\mathcal{H}_\phi$ and the projection P are unchanged. Now the last step in (1.41) has to be modified to

$$[P\mathcal{K}\rho^k](x_{t-k}) = \psi(x_{t-k})\langle \psi, \psi \rangle^{-1} \int_\Sigma \int_X \psi(\phi(X))^T \rho^k(F(X, \omega)) \mathrm{d}\mu(X) \mathrm{d}\mu_\Sigma(\omega). \tag{1.55}$$

Observe that through simple modifications to the definition of F as mapping from $\mathbb{X} \times \Sigma$ to $\mathbb{X}$ we can quite naturally obtain the same structure of the observed dynamics as in Eq. (1.44). For the computation of the coefficients h_k in Eq. (1.43) the expectation with respect to μ_Σ had to be added.

Dynamics of other observables

Although most of the literature on MZ is concerned with deriving the dynamics for values of an observation function ϕ which one assumes to be able to access, the formalism, in fact, leads to more general results. Note that two different observables play a role in MZ: firstly, ϕ, which is the observation of the system that we see. Secondly, we derived the Mori–Zwanzig equation by applying both sides of the Dyson formula to ϕ. But we could as well apply the Dyson formula to a different observation function $\eta : \mathbb{X} \to \mathbb{Y}$. The ensuing equation then answers the question: if $\phi(X_t), \phi(X_{t-1}), \dots$ are given, how do we compute $\eta(X_{t+1})$?

Indeed, we can quite straightforwardly perform the same steps as in Eq. (1.41) and arrive at

$$\begin{aligned}\eta(X_{t+1}) &= \sum_{k=0}^{t} [P((Q\mathcal{K})^k \eta \circ F)](x_{t-k}) + (Q\mathcal{K})^{t+1}\eta(X_0) \\ &= \sum_{k=0}^{t} \psi(x_{t-k}) h_k(\eta) + (Q\mathcal{K})^{t+1}\eta(X_0)\end{aligned} \tag{1.56}$$

with $h_k(\eta) := \langle \psi, \psi \rangle^{-1} \langle \psi, (Q\mathcal{K})^k \eta \circ F \rangle$.

Reformulation of the observed dynamics to matrix-vector product form

In Chapter 2, we will investigate in detail how the dynamics in Eq. (1.44), or rather the truncated dynamics in Eq. (1.52), can be estimated from data. There we will use techniques that estimate systems which come in the form of coefficient-matrix times vector, instead of coefficient-vector times matrix as is the case in Eq. (1.44). For this reason, let us reformulate the dynamics: selecting scalar-valued basis functions $\tilde{\psi}_1, \dots, \tilde{\psi}_K \in \mathcal{H}_\phi$ and denoting $\tilde{\psi} = [\tilde{\psi}_1, \dots, \tilde{\psi}_K]^T : \mathbb{X} \to \mathbb{R}^K$, we define dynamics given by

$$x_{t+1} = \sum_{k=0}^{t} H_k \tilde{\psi}(x_{t-k}) + \varepsilon_{t+1}, \tag{1.57}$$

with $H_k \in \mathbb{R}^{m \times K}$.

The difference between the two formulations for the dynamics is the following: in Eq. (1.44) the dynamics are expressed with different basis functions and the same coefficients across all coordinates (one chooses L m-dimensional basis functions and finds L-dimensional coefficients). In Eq. (1.57), scalar-valued basis functions $\tilde{\psi}_1, \dots, \tilde{\psi}_L$ are used for each coordinate while the coefficients for all coordinates can be different, given by the different rows of the H_k (one chooses K 1-dimensional basis functions and finds $(m \times K)$-dimensional coefficients). An illustration is given in Figure 1.6.

Of course, one might, and should, ask whether Eq. (1.57) is still consistent with the Mori–Zwanzig formalism. In fact, it can be directly derived through MZ by choosing the basis functions ψ in a suitable way depending on $\tilde{\psi}$. Then Eq. (1.57) is equivalent to Eq. (1.44). For the backward direction, one can define the matrices H_k suitably to have the desired effect of the scalar-valued h_k. In [WKS21] Appendix A2, a technical proof for these results is given. Note that this does not mean that both

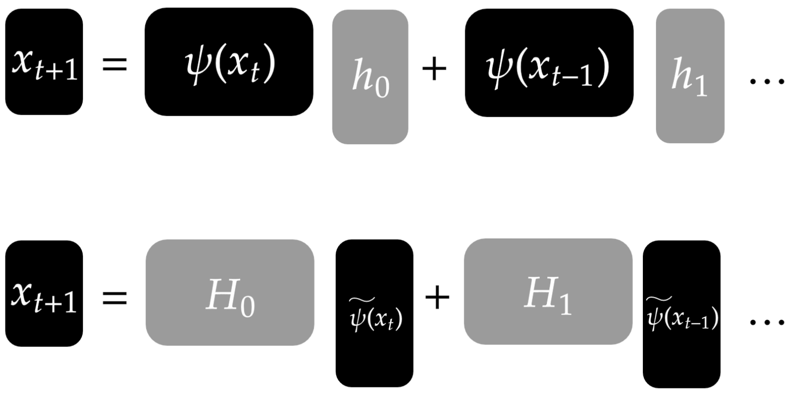

$x_{t+1} = \psi(x_t)\, h_0 + \psi(x_{t-1})\, h_1 \cdots$
$x_{t+1} = H_0\, \tilde{\psi}(x_t) + H_1\, \tilde{\psi}(x_{t-1}) \cdots$

Figure 1.6: Illustration of different formulations for the dynamics derived from the MZ equation. Top: form of Eq. (1.44) with matrix-valued basis functions and vector-valued coefficients. Bottom: form of Eq. (1.57) with matrix-valued coefficients and vector-valued basis-functions.

model formulations are always equivalent. One can merely always choose ψ and h_k in dependence of $\tilde{\psi}$ and H_k, respectively vice versa, in a way that makes the dynamics equivalent.

Ensuing Research Questions

Takens and MZ formalize and prove that generally memory terms are required to formulate the dynamics of an observable. However, in the future it would be desirable if research could generate answers to the following questions: (1) Can generalizations of Takens' Theorem be proved so that for more settings, i.e., type of dynamics, state space and observable, similar statements can be readily made? (2) Can the Takens and MZ perspectives be incorporated into a unified theory? Potentially this could help to develop a structure for the observed dynamics (as in MZ) that at the same time preserves topological properties (as in Takens' Theorem).

CHAPTER 2

Numerical Methods to Model a Dynamical System from Data

In the previous chapter we have analysed under which conditions we can formulate the dynamics of an observable of a dynamical system (Takens) and investigated the structure of these dynamics (Mori–Zwanzig). We can view this as a *forward direction*, i.e., as defining a dynamical system. Now we will consider the *backward direction* of learning several characteristics of the dynamics from data. For this aim, we will discuss several approaches to this learning process in detail. Many of these approaches are not necessarily inspired by memory but are intimately connected to approaches which take the need for memory terms into account.

The aim of the learning process – *what* exactly is learned from the data – can have very different faces. A predominant goal is to acquire the ability to make *predictions* of future states of the system. Another typical goal is to find an *understanding* of the dynamics, e.g., which parts of it influence each other, what happens in the long-term or which characteristics are preserved over time. For both goals there exists an abundance of approaches which are mostly based on constructing a *model* of the dynamics, meaning that one approximates a system function F of a dynamical system by a function, the model, $\mathcal{W}$. Typically, one chooses the structure of $\mathcal{W}$ first, i.e., the class of functions it comes from, and then determines parameters θ which exactly specify $\mathcal{W}$, called $\mathcal{W}_\theta$. The choice of the function class is the main difference between the different modelling approaches.

The demands set on such a model differ depending on the overall goal the modeller has. In many cases, one is interested in plain *accuracy* of the model, meaning that the distance between $\mathcal{W}_\theta(\cdot)$ and $F(\cdot)$ should be as small as possible, often ignoring any costs by building more and more precise but also more and more complex models.

In order to generate understanding of the true dynamics, one often seeks a model that is *interpretable*. This means that from the model and its parameters the modeller can learn something about the relations between the different variables which play a role in the system. Sometimes the values of the parameters directly send a message, e.g. by suggesting a strong relationship between variables of the model. Often times, however, one has to work harder, e.g., using specific algorithms to extract certain dominant characteristics of the model and, thereby, the dynamics, assuming that the model describes them well.

Another desired property of a model is *scalability*. This means that the computational cost of a model does not increase too quickly given an increase of fixed values such as the dimension of the system or the number of data points used to construct the model. Scalability is important to make a model applicable even to complex and high-dimensional problems.

Depending on the desired benefit that the model should yield in practice in a specific real-world problem, different model forms are suitable. There are models which generate tremendous accuracy but are non-interpretable and scale badly. Others yield immediate interpretation but might not represent the dynamics accurately. Some are simple and scale well but give a mediocre prediction accuracy and admit only a superficial level of interpretation. Typically, one needs to specify the desired balance between these demands and then select a model.

As we have seen in Chapter 1, if we can only make certain observations from the system from which we cannot directly reconstruct the true system state, the dynamics which push the state of the observable forward in time require memory terms. When constructing a model, one has to take that into account and make it dependent on memory terms, too. Thus, in this setting one is focused on finding a model, including its parameters, which fulfils $x_{t+1} \approx \mathcal{W}_\theta(x_t, \dots, x_{t-p+1})$.

One could write thousands of pages on different perspectives, approaches and techniques of the modelling of dynamical systems. The aim of this chapter and the next one is to illustrate methods which identify memory-exhibiting dynamics which fulfil the different demands mentioned. Further, we will see how they are inspired from Takens' Theorem or the Mori–Zwanzig formalism. In the following chapter, we will formally establish relations between some of these methods, introduce a novel method and compare their applicability on examples.

This chapter is divided into three parts: at first, an overview of different state-of-the-art models and related research is presented in the context of memory. Secondly, we will discuss methods which directly emerge from theory on Takens' Theorem to model dynamics of observables and understand the relation between them. Third, we will in detail analyse methods which are based on projecting the dynamics into the span of suitably chosen basis functions to discover the best approximation of the dynamics in this space. The latter resemble simple regression problems but can be leveraged in effective ways to gain understanding of dynamics. As we will see, these methods are closely related to methods inspired from the Mori–Zwanzig formalism which will help us to connect theoretic to practical approaches on modelling dynamical systems with memory.

Note that $x_t = f(x_{t-1}, \dots, x_{t-p})$ is not truly a dynamical system because input - the tuple $(x_{t-1}, \dots, x_{t-p})$ - and output - the state x_t - come from different sets but we can simply augment the system to $[x_t, \dots, x_{t-p+1}] = \hat{f}(x_{t-1}, \dots, x_{t-p}) = [f(x_{t-1}, \dots, x_{t-p}), x_{t-1}, \dots, x_{t-p+1}]$. We will therefore still use the term "dynamics" or "dynamical system" if multiple past states are mapped to a future one.

2.1 A Collection of Methods to Model Dynamics with Memory

Firstly, a brief summary of research on numerical methods for the investigation of memory-exhibiting dynamics is given, simply to illustrate the broad width of possible research questions.

For example, in [KHN19, KDB$^+$18, LKN20], the *mean first passage times* of barrier crossing in Physical processes is estimated using the assumption of a specific formulation of time-continuous memory-exhibiting dynamics, the Generalized Langevin Equation (GLE). A time-approximation of the GLE is sought in [HHSN07] with the aim of quantifying the memory effects in an example of molecular dynamics. The effects of memory terms are further estimated for biological systems in [GHS08] and in the context of climate change in [MGO10]. Attempts to model complex systems with memory have been conducted with different approaches, such as Memory Markov State Models to find the master equation of discretized dynamics of a protein network [CMCW$^+$20]. Methods for modelling memory-exhibiting dynamics by separating between different types of behaviour, so-called regimes, are introduced in [Hor10, PGSH18]. Moreover, discrete-space modelling methods with memory have been developed such as Cellular Automata [ASM03] or discrete autoregressive models [DHH03].

One could extend this list endlessly with specific, seemingly unconnected examples. In contrast, we will now start discussing and comparing three different specific classes of models which are dominant in the field of estimating memory-exhibiting dynamics in discrete time and continuous state space.

2.1.1 Neural Networks

Many real-world problems nowadays are tackled with so-called neural networks (NNs). Although they are not meant to be an integral part of this thesis, it is worth getting acquainted with them since they denote state-of-the-art techniques directed towards many problems. It can be said already, that considering the trade-off between accuracy, interpretability and scalability, most versions of them can be placed far in the accuracy-corner with shortcomings in the other two categories. Knowing this should help us gain perspective on strengths and weaknesses of methods that are of more interest to this thesis. In this light, the next few pages can be seen as an argumentation as to why neural networks do not automatically solve all problems.

Neural networks are inspired by the human brain, which functions by sending signals from neuron to neuron to process information and enable action of the body. After first attempts at utilizing such processes mathematically in the 1940s [MP43], NNs have seen first incremental advances towards the 1990s [Ros58, Fuk80, Hop82] and over the last 25 years have surged to being a crucial class of methods to solve problems of different types.

In contrast to the popular perception of NNs as emulating brain processes, from a mathematical point of view they can simply be perceived as a concatenation of functions. An input x is pushed through a concatenation of affine transformations and scalar-valued nonlinear *activation functions* until it is mapped to an output y. Their general form is as follows:

$$\begin{aligned} h^{(0)} &:= x \\ \text{for } k = 1, \dots, L: \ h^{(k)} &:= f_{k-1}(W^{(k-1)} h^{(k-1)} + b^{(k-1)}) \\ y &:= f_L(W^{(L)} h^{(L)} + b^{(L)}) \end{aligned} \tag{2.1}$$

Layers $h^{(1)}, \dots, h^{(L)}$ are called *hidden layers*, while $h^{(0)} = x$ is called the *input layer*. The entries of the layers are called *neurons*. The matrices $W^{(k-1)}$ are called *weights* and the vectors $b^{(k-1)}$ are called *biases*. θ denotes all $W^{(k)}$ and $b^{(k)}$ in the network. The $f_k : \mathbb{R} \to \mathbb{R}$ denote the activation functions, which we define as being applied to each coordinate individually. In the end, $g(x, \theta)$ denotes the output of the network with input x and parameters θ.

The number of hidden layers L and the dimensions of every layer have to be specified in the beginning. Moreover, the activation functions have to be chosen. The weights and biases however, are estimated on the basis of a data set with which the network is trained. Figure 2.1 sketches the typical structure of an NN.

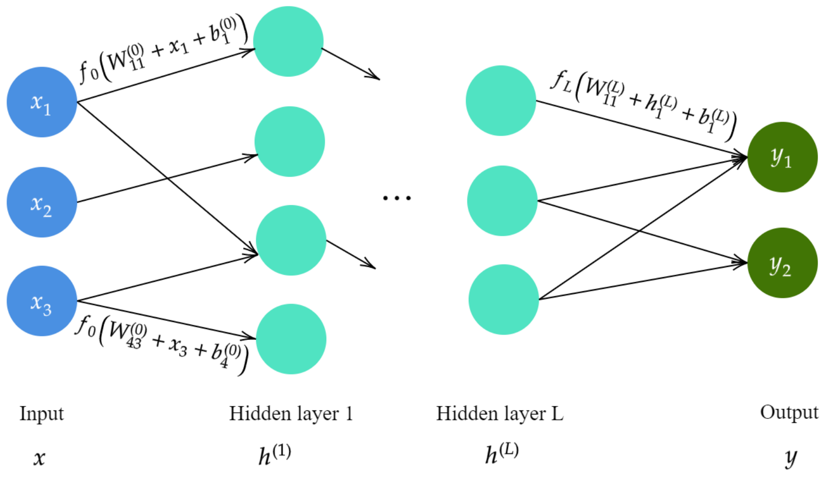

$f_0(W^{(0)}_{11} + x_1 + b^{(0)}_1)$
x_1
x_2
x_3
$f_0(W^{(0)}_{43} + x_3 + b^{(0)}_4)$
$\dots$
$f_L(W^{(L)}_{11} + h^{(L)}_1 + b^{(L)}_1)$
y_1
y_2
Input
Hidden layer 1
Hidden layer L
Output
x
$h^{(1)}$
$h^{(L)}$
y

Figure 2.1: Top: structure of a neural network. An input vector x is mapped to the entries of a hidden layer $h^{(0)}$ by affine transformations (W, b), followed by nonlinear activation functions f. Arbitrarily many hidden layers can follow, again generated by the application of affine transformation and activation function to the previous layer. In the end, an output vector y is constructed. Bottom: three typical activation functions. Green: sigmoid. Purple: tanh. Orange: ReLU.

In contrast to many more classical techniques in modelling, the form of a neural network does usually not use any a priori knowledge about the problem, although research in this direction exists in the field of *Physics-inspired NNs* (see, e.g., [HSN20]). It rather relies on a brute-force approach of using a sufficient number of neurons and layers to capture the connection between input to output accurately.

A very helpful result which connects NNs with more classical techniques is the Universal Approximation Theorem [Cyb89]. It states that any smooth function can be arbitrarily well approximated with an NN with only one hidden layer. However, in theory, the required number of neurons in the hidden layer can become infeasibly high. It actually seems advisable to use *deep* networks, i.e., with many layers, instead of *shallow* networks with few but large layers [LTR17, RT17]. Still, the theorem asserts that, while many complex problems exist in which NNs are superior, even if the problem is intrinsically simple, there still exists an NN which is suitable for it. In other words, NNs are not restricted to complex problems but the technique in general is highly applicable.

That said, one has to determine its parameters, i.e., the weights and biases. This is typically done by minimization of an objective function $\mathcal{L}$ which measures the distance between network output with given input and parameters and the desired output. The minimization is typically done by one of various forms of the gradient descent method [Rud16], yielding the need to compute the gradient of $\mathcal{L}$. Although often an analytical formulation of the gradient can be derived, the search for the global minimum of $\mathcal{L}$ poses many difficulties, since it is usually not convex but rather has many local minima, yielding the typical problems one faces in parameter estimation. Further, the number of parameters is high for many networks – from multiple thousands to several billions in commercially used NNs – making the minimization process very cost-intensive. Plus, in order to determine parameters in a robust way, one usually requires a large amount of data.

Neural networks for dynamics

NNs can be used for different tasks, such as image or speech recognition, e.g., [KSH12], translation, e.g., [Sta20] or complex Physics problems such as Schrödinger equations [HSN20] or protein folding [SEJ+20], to only name a few. These problems are all concerned with dynamics. To solve such problems, there are special classes of NNs.

Recurrent neural networks The most fundamental class is called *Recurrent neural networks* (RNNs) and was introduced first as early as the mid-1980s in [RHW86, Jor97]. An RNN constructs a sequence of outputs $y_{t-1}, y_t, \dots$, depending on inputs $x_{t-1}, x_t, \dots$ at each time step. To generate sequential dependencies, there exists a hidden state h_t at each time step which depends on the input and the hidden state from the previous time step.

It can, however, be analytically shown that due to their structure typical RNNs are generally unable to install long-term dependencies, i.e., between a future state and one that lies far in the past. The reason for that is that the gradient of the loss function in dependence on past states can in- or decrease exponentially quickly which is known as the *exploding/vanishing gradient problem*. This diminishes the usefulness of this generic form of RNNs in many practical applications, e.g., translation, where the meaning of a word can depend on the context given by another word much earlier. In the context of dynamical systems with memory, this means that RNNs will generally not be a good choice to model those systems.

We will now consider a special class of RNNs which better captures the influence of states from the past.

Long short-term memory networks *Long short-term memory* networks (LSTMs) were introduced in 1997 in [HS97] but more thoroughly put into practice only in the 2010s. An up-to-date review of the method including applications can be found in [VHMN20].

LSTMs circumvent the exploding/vanishing gradient problem by storing information in a hidden layer, called *memory cell*, for multiple time steps. Multiple parts of the LSTMs, all networks on their own, decide whether new input is included into the memory cell and if previous information is discarded. From the new input and the information in the memory cell, an output is constructed. An example is, again, text translation. For a new word, the input, the LSTM has to decide whether the word is important for the translation and if it makes the previous information obsolete, e.g., by creating a new context. On this basis it is decided which translation of the word best captures its meaning at the current position. A study of the application of LSTMs on a more theoretic example can be found in [MWE18].

As can analytically by shown, by storing information, LSTMs can be trained to uncover long-term dependencies between states of a dynamical system. This makes them a powerful class of methods in various applications.

Although the quality of many variants of neural networks is assessed on the basis of practical real-world examples, there exists literature on benchmarking of different NNs on well-known problems. For example, a combination of an LSTM and a so-called Convolutional NN (see [LBBH98]) is used in [TZ19] to predict states of the Lorenz-63 system introduced in Chapter 1 with strong accuracy.

An NN variant that is competing with LSTMs is theso-called *transformer* [VSP$^+$17] which, unlike LSTMs, does not take into account the time-order of inputs. At least on the basis of many experiments (see, e.g., [ZBI$^+$19]), the transformer seems to be an often superior alternative in terms of accuracy and scalability. It does have weaknesses, such as that the maximal memory depth has to be fixed (unlike in LSTMs). For definitive statements on comparisons to LSTMs, however, it seems that more theoretical research is required.

Summary of neural networks

In summary, neural networks have been used to solve problems where many rather classical methods failed, generating a high level of accuracy in countless examples. Their architecture allows for a modelling of complex phenomena, using a very high number of parameters, harnessed in defining affine transformations and nonlinear activation functions so that the relevant features are extracted from an input. This structure can successfully be utilized for the modelling of dynamical systems. However, their architecture yields significant drawbacks: firstly, the parameter estimation requires large, sometimes unrealistic amounts of data. Secondly, given their high number of parameters, they are hardly interpretable, therefore functioning as *black box* models, so that they are often of little help in understanding dominant relations in a system. For these reasons, despite the quick progress NNs have made over the last years, the need for different techniques to model dynamical systems exists.

2.2 Methods Based on Takens' Theorem

In the literature on memory-based data-driven methods, often times Takens' Theorem is used as a justification for the use of memory by the mere mention of the name. Its implications in practice, however, do not far exceed the *existence* of memory-exhibiting dynamics of the observable. As illustrated in this section, the smoothness of the delay embedding does allow for modelling of dynamics. One does not, however, obtain an analyzable or interpretable operator. In this section we will briefly review three different numerical methods which are based on the idea of Takens' Theorem in order to point out which numerical methods it induces but also its shortcomings in practice.

Recall that according to Takens, there exists a diffeomorphism between states X in the original state space and points of the form $\hat{x} := \Phi_{\phi,F,p}(X)$. Therefore, the embedded dynamics $\hat{f} = \Phi_{\phi,F,p} \circ F \circ \Phi_{\phi,F,p}^{-1}$ are diffeomorphic, too. A diffeomorphism is smooth by definition, yielding that neighbourhoods map to neighbourhoods. More precisely, using that any differentiable function on a compact space is Lipschitz continuous, we find

$$\|\hat{f}(\hat{x}) - \hat{f}(\hat{x}')\|_2 \leq u_{\hat{f}} \|\hat{x} - \hat{x}'\|_2 \quad \text{for a } u_{\hat{f}} < \infty. \tag{2.2}$$

If $u_{\hat{f}}$ is small, this helps to learn about both future states of observed dynamics and detect causalities between different variables of a system, as we will see next.

2.2.1 Nearest Neighbours Regression

For a dynamical system on a compact space $\mathbb{X}$ with dynamics given by a smooth function $F : \mathbb{X} \to \mathbb{X}$ and a smooth observable $\phi : \mathbb{X} \to \mathbb{R}^m$, a simple method to predict future states of the form $\phi(X_t)$ is the nearest neighbours regression method, see e.g., [CS18, ZLY+13, AQC13]. It is not to be confused with the k-nearest neighbours method introduced in Chapter 1, nearest neighbour classification [ANT19] or graph-related nearest neighbour methods although the general approach, also suggested by the title of the method, is similar.

The intuition behind nearest neighbours prediction is as follows: in order to predict what the next observed state, x_{t+1}, should be, we consider states in the delay-embedded trajectory that are similar to $\hat{x}_t$, denoted by $\hat{x}_{t_1}, \dots, \hat{x}_{t_L}$ (in ascending order by their distance to $\hat{x}_t$). Since the delay-embedded dynamics $\hat{f}$ are assumed to be smooth, $x_{t+1} = \hat{f}_1(\hat{x}_t)$ should also be close to $\hat{f}_1(\hat{x}_{t_1}), \dots, \hat{f}_1(\hat{x}_{t_L})$. Formally, if $\hat{f}_1$ is Lipschitz continuous with upper Lipschitz constant $u_{\hat{f}_1}$, it holds

$$\|\hat{x}_t - \hat{x}_{t_i}\|_2 \leq \varepsilon \Rightarrow \|x_{t+1} - x_{t_i+1}\|_2 = \|\hat{f}_1(x_t) - \hat{f}_1(\hat{x}_{t_i})\|_2 \leq u_{\hat{f}_1}\varepsilon. \tag{2.3}$$

Based on this, the method works as follows: choose a number $L \in \mathbb{N}$ of nearest neighbours. Then given a trajectory $x_1, \dots, x_t$, consider $\hat{x}_p, \dots, \hat{x}_t$ and determine the L nearest neighbours of $\hat{x}_t$, i.e. the points closest to $\hat{x}_t$ according to a given distance metric, e.g., the Euclidean norm. Then approximate x_{t+1} by

$$\tilde{x}_{t+1} = \frac{1}{L}\sum_{i=1}^{L} x_{t_i+1}. \tag{2.4}$$

The approximation of the future observed state is thus constructed as an average of subsequent states of points similar to $\hat{x}_t$, using the smoothness of the delay-embedded dynamics. Alternatively to using L nearest neighbours one could define a threshold c and use all points – denoted by L_t – whose distance to $\hat{x}_t$ is not bigger than c or even make the number of nearest neighbours itself depend on $\hat{x}_t$. For this reason, from now on we denote this number by L_t. $L_t \equiv L$ is then a special case.

Takens-based nearest neighbours regression [MLC17, OAH20, SM90] then functions in the following way: Let $\hat{x}_{t_1}, \dots, \hat{x}_{t_{L_t}}$ be the L_t nearest neighbours of $\hat{x}_t$ according to a given metric. Then the Takens-based nearest neighbours prediction of x_{t+1} is given by

$$\tilde{x}_{t+1} = \frac{1}{L_t}\sum_{i=1}^{L_t} x_{t_i+1}. \tag{2.5}$$

An illustration is given in Figure 2.2. For the prediction error we can derive the following result:

Proposition 2.1. Let $\mathcal{M}$ be a smooth, compact, d-dimensional manifold. Let $\hat{x}_{L_t}$ be the L_tth nearest neighbour of $\hat{x}_t$. Then for a generic set of pairs $(F,\phi) \in \mathcal{D}^1(\mathcal{M}) \times \mathcal{C}^1(\mathcal{M},\mathbb{R})$, it holds that the prediction error of Takens-based nearest neighbours regression with memory depth $p > 2d$ and the Euclidean distance as metric is bounded by

$$\|x_{t+1} - \tilde{x}_{t+1}\|_2 \leq u_{\hat{f}_1} \|\hat{x}_t - \hat{x}_{t_{L_t}}\|_2 \tag{2.6}$$

for a constant $u_{\hat{f}_1}$ that depends on F, ϕ, $\mathcal{M}$ and the memory depth p.

Proof. The assumptions in the proposition are exactly the ones from Takens' Theorem (see Theorem 1.5). They imply the generic existence of a smooth map $\hat{f}$ as defined in Eq. (1.3). Over a compact finite-dimensional manifold, $\hat{f}$ is Lipschitz continuous. Clearly, this also holds for its first m coordinates, $\hat{f}_1$, having Lipschitz constant $u_{\hat{f}_1}$. $u_{\hat{f}_1}$ clearly depends on F, ϕ and p since $\hat{f}$ is constructed with them, and further on its domain $\mathcal{M}$. It then holds,

$$\begin{aligned}
\|x_{t+1} - \tilde{x}_{t+1}\|_2 &\leq \|x_{t+1} - \frac{1}{L_t}\sum_{i=1}^{L_t} x_{t_i+1}\|_2 \\
&\leq \frac{1}{L_t}\sum_{i=1}^{L_t} \|x_{t+1} - x_{t_i+1}\|_2 \\
&= \frac{1}{L_t}\sum_{i=1}^{L_t} \|\hat{f}_1(\hat{x}_t) - \hat{f}_1(\hat{x}_{t_i})\|_2 \\
&\leq u_{\hat{f}_1}\frac{1}{L_t}\sum_{i=1}^{L_t} \|\hat{x}_t - \hat{x}_{t_i}\|_2 \\
&\leq u_{\hat{f}_1}\|\hat{x}_t - \hat{x}_{t_{L_t}}\|_2.
\end{aligned} \tag{2.7}$$

□

Proposition 2.1 establishes a direct connection from demanded properties of dynamics, observable, memory and state space to an error-bound of the Takens-based nearest neighbours regression. We have used the assumptions from the original theorem of Takens. Of course, one could replace these with any suitable variant of the theorem presented in Chapter 1 and derive an analogous result for the method.

Observe that the Lipschitz constant of the delay-coordinate map Φ, and subsequently the embedded dynamics $\hat{f}$, determines the quality of the Takens-based nearest neighbours regression. The Lipschitz constant of Φ directly regards the preservation of the geometry of the state space under Φ. As explained in Chapter 1, an embedding guarantees the preservation of topological properties. We can see here that for practical purposes, the geometric ones are of highest relevance, too.

Note that there exist variants, e.g., in [MLC17], in which the neighbours $\hat{x}_{t_i}$ are given a weighting $w_{t,i}$ according to their distance to $\hat{x}_t$. The weighting coefficients are bigger for close distances, e.g., given by $w_{t,i} = \exp(\|\hat{x}_t - \hat{x}_{t_i}\|_2)$ and the predic-

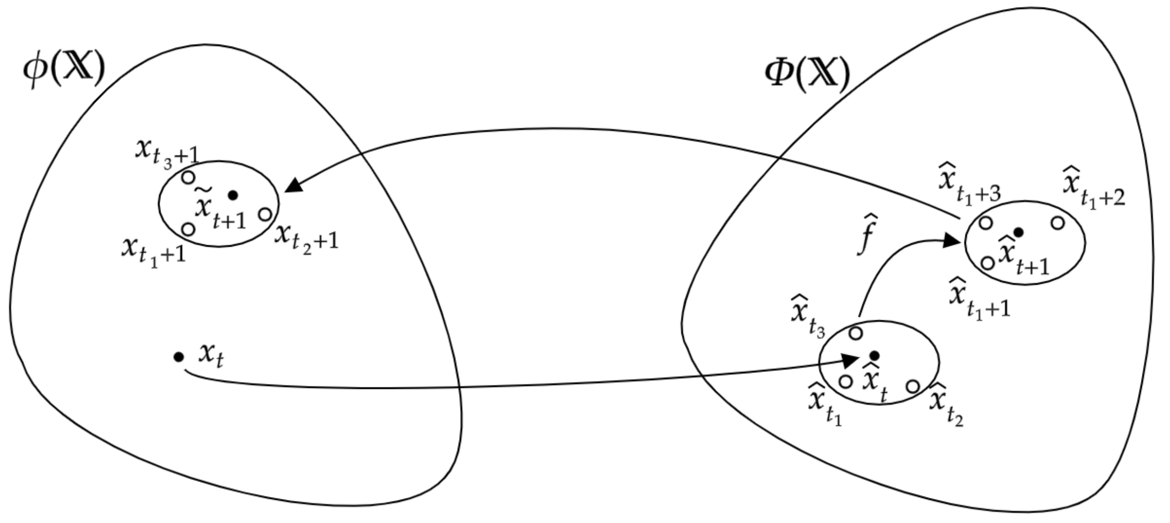


Figure 2.2: Illustration of the Takens-based nearest neighbours regression. Neighbours of $\hat{x}_t$ are identified in the delay embedding of $\mathbb{X}$. One considers the time-propagated counterparts of these neighbours and identifies the pre-images of the delay-coordinate map in $\mathbb{X}$. One then computes their average to obtain a prediction of x_{t+1}.

tion scheme reads

$$\tilde{x}_{t+1} = \frac{\sum_{i=1}^{L_t} w_{t,i} x_{t_i+1}}{\sum_{i=1}^{L} w_{t,i}}, \tag{2.8}$$

resulting in the computation of $\hat{x}_{t+1}$ as a weighted average of other points.

2.2.2 Kernel Regression

A prediction method which is a close variant to nearest neighbours regression is Kernel regression, also called Kernel smoothing [HTF09,Gho18]. In order to predict subsequent states of the observable, it does not truncate the sum in Eq. (2.8) after the L nearest neighbours of a point, but takes all points in the data into account. In order to control the influence of different points one selects a *Kernel* function $K : \mathbb{R}^{mp} \times \mathbb{R}^{mp} \to \mathbb{R}$ – non-negative and symmetric – which measures closeness between two points in the image of the delay-coordinate map. The motivation is that in nearest neighbours regression, by truncating the sum after L neighbours, one might lose potentially vital influence of the truncated terms. Instead, in Kernel regression, one includes all terms but accounts for the different distances to the point of interest by the Kernel as a weighting function. As is argued in [GGH21], a smooth function $\hat{f}_1 : \mathbb{R}^{mp} \to \mathbb{R}^m$ can be written as

$$\hat{f}_1(\hat{x}) = \int_{\mathbb{R}^{mp}} \delta(\hat{x} - \hat{x}') \hat{f}_1(\hat{x}') \mathrm{d}\mu(\hat{x}'), \tag{2.9}$$

assuming the states $\hat{x}$ are distributed by μ. δ denotes the Dirac-delta distribution. One can then try to approximate this integral by replacing the Dirac-delta distribution by a Kernel function and the integral by a normalized sum. Thus, one makes a prediction using

$$x_{t+1} = \int_{\mathbb{R}^{mp}} \delta(\hat{x}_t - \hat{x}) \hat{f}_1(\hat{x}) \mathrm{d}\mu(\hat{x}) \approx \int_{\mathbb{R}^{mp}} K(\hat{x}_t, \hat{x}) \hat{f}_1(\hat{x}) \mathrm{d}\mu(\hat{x}) \approx \frac{\sum_{i=1}^{t-1} K(\hat{x}_t, \hat{x}_i) \hat{f}_1(\hat{x}_i)}{\sum_{i=1}^{t} K(\hat{x}_t, \hat{x}_i)}. \tag{2.10}$$

Typically, one chooses a family of Kernels and parametrises them using a so-called *bandwidth* parameter h. For example, let

$$\tilde{K}(\hat{x}) = \exp(\|\hat{x}\|_2) \text{ and set } K(\hat{x}, \hat{x}') = \tilde{K}\left(\frac{\hat{x} - \hat{x}'}{h}\right) = \exp\left(\frac{\|\hat{x} - \hat{x}'\|_2}{h}\right). \tag{2.11}$$

h should be selected so that Kernel regression generates the optimal prediction accuracy. A higher value for h makes the weights of all points in the data more similar regardless of their distance to $\hat{x}_t$ while a small value for h makes the prediction strongly dependent on the closest points. The latter is advantageous if data points are sufficiently close to each other but if this is not the case, predictions might be too sensitive to the positions of the data points (as is the problem with the truncation in nearest neighbours regression).

Kernel regression can be used not only in the context of the prediction of observations of a dynamical system, but more generally to approximate the value of a function g evaluated at x if information of the form $g(x_1), \ldots, g(x_N)$ is available. A typical approach to obtain an optimal value for h is by cross-validation, i.e., by tuning h on the basis of the data at hand. For more information, see [SSMZK11].

Observe that by choosing the Kernel function as

$$K(z, z') = \begin{cases} 0 & \text{if } \|\hat{x} - \hat{x}'\|_2 > c \\ \tilde{K}(\|\hat{x} - \hat{x}'\|_2) & \text{otherwise,} \end{cases} \tag{2.12}$$

for a suitably chosen $\tilde{K}$, Kernel regression is equivalent to weighted nearest neighbours regression in which one truncates all points further away from z than c.

An advantage of both nearest neighbours and Kernel regression is that, despite being conceptually simple, they can produce a low prediction error, if the underlying dynamics are sufficiently smooth and one possesses sufficiently much data. Further, they demand almost no prior intuition about the dynamics, since one needs to specify only the number of nearest neighbours, respectively the distance threshold and a Kernel, including the bandwidth parameter, for the weighting coefficients. Good choices for these parameters can be estimated by fitting them to data at hand.

Theoretical results on the accuracy of Kernel regression in the limit of memory depth and data can be found in [Gia21].

However, these two methods do not come without significant drawbacks. Firstly, one can hardly deduce interpretations about the dynamics from the application of the method, since there exists no closed model formulation. Rather, the data are directly used to make predictions. We will see later that this forces these methods to usually require many data points to function well when other methods survive with much less data points. Moreover, the methods can only interpolate since every prediction must lie *between* the data points used to construct it. This makes the method unsuitable for cases in which the data at hand come from one part of the state space but would admit identification of dominant structures that hold up in other parts of the state space. While on this basis other methods might be able to predict the dynamics across the whole state space, the two methods discussed here will always predict into the domain that the data came from.

2.2.3 Convergent Cross Mapping

We have now seen two related methods to make predictions of observables of a dynamical system, directly supported by the neighbourhoods-map-to-neighbourhoods property of delay-coordinate maps. As discussed, the degree to which this property is true determines the quality of nearest neighbours and Kernel regression. In [SMY$^+$12], a method is presented which directly quantifies this quality with the aim of detecting strength of causation between different variables of a system.

The argumentation of the authors of [SMY$^+$12] is as follows: let a smooth dynamical system F work on states of the form (X, Y) in a d-dimensional manifold $\mathcal{M}$. Let two observables be given by

$$\phi_X(X, Y) = X, \quad \phi_Y(X, Y) = Y. \tag{2.13}$$

For a memory depth $p > 2d$, we consider their delay-coordinate maps. Now, If X and Y influence each other – the authors of [SMY$^+$12] say they are *dynamically coupled* –, e.g., as parts of the same dynamical system, then similar to the nearest neighbours regression, neighbourhoods in the image of Φ_{ϕ_X} should map to neighbourhoods in the image of Φ_{ϕ_Y} and vice versa. If X and Y are not coupled but are variables of separate dynamical systems, the delay-coordinate maps generally are not diffeomorphisms so that this does not hold. The method *Convergent Cross Mapping* (CCM) introduced in [SMY$^+$12] performs a quantitative test whether this property holds to see if two variables are coupled. Since we do not further use it in this thesis, we refer to [SMY$^+$12] for its precise formulation.

If there is directional coupling, i.e., X forces Y but not vice versa, then the states in $\Phi_{\phi_X}(\mathcal{M})$ encode no information about Y. The map Φ_X is therefore no diffeomorphism while Φ_Y is one. This generally yields that the prediction of Y_t on the basis of nearest neighbours on $\Phi_{\phi_X}(\mathcal{M})$ is not precise while the prediction of X_t on the basis of nearest neighbours on $\Phi_{\phi_Y}(\mathcal{M})$ is. Potentially contrary to intuition, according to CCM, if X can be predicted from Y this indicates that X causes Y, not vice versa.

With this, CCM can serve as a method to detect which variables influence which. For example, in [SMY$^+$12] it is tested whether presence of fish of different species in an ecosystem impact each other strongly (the authors find that rather sea surface temperature is the main driver of populations). Further examples can be found in [SBK20, PKJC18, BZF$^+$18].

However, it must be said that when deducing interpretations from results of the method, one has to be cautious. If one variable is too strongly forced by the other, so that it is close to being a direct copy, then by the construction of the method the directionality of the coupling will not be uncovered. More details can be found in [YDGS15] where the authors acknowledge this.

Instead of detecting causalities between different variables of a system, one could alternatively try to find memory effects in the dynamics of one variable. Note that, as we have learned, since the influence of other variables is stored in the memory of one variable, one should be able to detect dependencies between memory terms. In the future, it could be a worthwhile endeavour to conduct research in this direction and test whether one can determine an optimal memory depth for nearest neighbours- or Kernel regression or the methods which are to follow in this chapter.

2.2.4 Summary of the Takens-based Methods

In this section, we have seen three data-driven methods which are based on Takens' Theorem. The first two, nearest neighbours and Kernel regression provide a non-parametric way to predict states of observables at future time steps. The latter, Convergent Cross Mapping, aims at identifying causation between the variables of a dynamical system. All methods use the property of smooth diffeomorphisms that neighbourhoods map to neighbourhoods. They exploit this property by considering delay-embeddings of observables which, firstly, represent information worth the one from the original system states and, secondly, generate smooth mappings both along the delay-embedded states and to the original state space. This allows to approximate unknown states, such as x_{t+1} or Y_t, on the basis of known ones, such as neighbours $\hat{x}_{t_i}$ together with x_{t_i+1} (in regression) or Y_{t_i} (in CCM).

In the next section, we will discuss in detail a branch of modelling techniques which is based on finding a suitable projection space from where estimation of the dynamics becomes simpler. They denote parametric models for which data is required to estimate their parameters rather than functioning directly on the data as in the Takens-based methods. As we will see, they complement the previously introduced methods nicely and strongly contribute to the range of techniques of dynamical systems. We will in fact focus on them in the following two chapters.

2.3 Koopman-based Methods

In this section we will introduce several related methods, which are based on transforming states into a different, typically higher-dimensional, space from which one can determine a linear mapping to the state one time step in the future. Again, we consider dynamics $F : \mathbb{X} \to \mathbb{X}$ and a uniformly bounded observable $\phi : \mathbb{X} \to \mathbb{R}^m$. We will first introduce the general concept, building on the so-called *Koopman operator*, and then discuss several methods. Afterwards, in the following chapter, we will define a novel extension and prove relations between them.

Remark. Please note that for the next few pages we assume that ϕ is scalar-valued, i.e., $m = 1$, since this makes the derivation of results notationally easier. Afterwards it is shown how to extend them to multidimensional functions in a natural way.

2.3.1 The Koopman Operator

Recall the Koopman operator, defined in Definition 1.12 as $\mathcal{K} : L^\infty(\mathbb{X}) \to L^\infty(\mathbb{X})$, depending on the dynamics F, with

$$\mathcal{K} \circ \phi = \phi \circ F \quad \forall \phi \in L^\infty(\mathbb{X}). \tag{2.14}$$

The Koopman operator was introduced in 1931 by B. O. Koopman in [Koo31] (its name was, of course, only later attached to Koopman). Although seeming like only a formalization of the propagation of an observable through time, it provides a crucial shift of perspective: it does not map a *value* of the observable ϕ to its value at a future time step, but the *function* ϕ to the function $\phi \circ F$. This has many practical implications, as we will learn from the following considerations.

It can be quickly observed that the Koopman operator is linear,

$$\mathcal{K} \circ (a_1\phi_1 + a_2\phi_2) = a_1\phi_1 \circ F + a_2\phi_2 \circ F = a_1\mathcal{K} \circ \phi_1 + a_2\mathcal{K} \circ \phi_2. \tag{2.15}$$

Further, the operators $\mathcal{K}^k, k = 0, \ldots,$ form a semi-group, since

$$\mathcal{K}^k \circ \mathcal{K}^l \circ \phi = \mathcal{K}^k \circ \phi \circ F^l = \phi \circ F^l \circ F^k = \phi \circ F^{l+k} = \mathcal{K}^{l+k} \circ \phi. \tag{2.16}$$

While the Koopman operator is commonly defined for scalar-valued functions, we define for a multidimensional function $\psi = [\psi_1, \dots, \psi_N]^T : \mathbb{X} \to \mathbb{R}^N$,

$$\mathcal{K} \circ \psi = \begin{bmatrix} \mathcal{K} \circ \psi_1 \\ \vdots \\ \mathcal{K} \circ \psi_N \end{bmatrix}. \tag{2.17}$$

Note that the Koopman operator is linear regardless of any nonlinearity of the dynamics. However, even if the dynamics are finite-dimensional, the Koopman operator is not. By adapting the perspective of the Koopman operator, one replaces a finite-dimensional but potentially nonlinear system by an infinite-dimensional but linear system. The decisive element of the applicability of the Koopman perspective thus is whether one can find an accurate finite-dimensional representation of it.

Propagation of observables through matrix representation of the Koopman operator

Let us make the usually unrealistic but for now convenient assumption that there exists a Koopman-invariant subspace with scalar-valued basis functions $\psi_1, \dots, \psi_N : \mathbb{X} \to \mathbb{R}^m$, corresponding to which we denote $\psi = [\psi_1, \dots, \psi_N]^T$, meaning that if $\phi \in span\{\psi_1, \dots, \psi_N\}$ then also $\mathcal{K} \circ \phi \in span\{\psi_1, \dots, \psi_N\}$. Therefore, there exist coefficients $b_{ij} \in \mathbb{R}$ so that

$$\mathcal{K} \circ \psi_i = \sum_{j=1}^{N} b_{ij} \psi_j. \tag{2.18}$$

Writing $B = (b_{ij}) \in \mathbb{R}^{N \times N}$, this implies

$$\begin{bmatrix} \mathcal{K} \circ \psi_1 \\ \vdots \\ \mathcal{K} \circ \psi_N \end{bmatrix} = B\psi, \quad \begin{bmatrix} \mathcal{K}^k \circ \psi_1 \\ \vdots \\ \mathcal{K}^k \circ \psi_N \end{bmatrix} = B^k \psi. \tag{2.19}$$

We see that for finite-dimensional invariant spaces, the action of the Koopman operator can be reduced to a matrix multiplication. Let us assume for now that our observable ϕ can be constructed using the basis provided by these functions, i.e., there are coefficients $a = [a_1, \dots, a_N]^T \in \mathbb{R}^N$ so that

$$\phi = \sum_{i=1}^{N} a_i \psi_i = a^T \psi. \tag{2.20}$$

Then Eq. (2.18) yields

$$\mathcal{K} \circ \phi = \mathcal{K} \circ \sum_{i=1}^{N} a_i \psi_i = \sum_{i=1}^{N} a_i \mathcal{K} \circ \psi_i = \sum_{i=1}^{N} a_i \sum_{j=1}^{N} b_{ij} \psi_j = a^T B \psi. \tag{2.21}$$

This observation should give hope that, even if one does not have a Koopman-invariant subspace at hand, one can at least approximate $\mathcal{K}$ by a matrix. The task is to make a good choice of basis functions $\psi_1, \ldots, \psi_N$ for which, on the one hand, the action of $\mathcal{K}$ in their span can be approximated by a matrix, B, and, on the other hand, the desired observable ϕ can be constructed from these basis functions with coefficients in the vector a. Figure 2.3 illustrates the propagation of ϕ by transforming it into $span\{\psi_1, \ldots, \psi_N\}$, transporting these forward in time by the matrix B and recomposing ϕ by multiplication with a.

Remark. For multidimensional observables $\phi : \mathbb{X} \to \mathbb{R}^m$, we can follow the same argumentation using basis functions $\psi_i : \mathbb{X} \to \mathbb{R}^m$ with $\psi = [\psi_1, \ldots, \psi_N] : \mathbb{X} \to \mathbb{R}^{m \times N}$. Modifying Eq. (2.20) with $a \in \mathbb{R}^N$ to

$$\phi = (a^T \psi^T)^T \tag{2.22}$$

and assuming Eq. (2.18), we obtain in Eq. (2.19)

$$[\mathcal{K}^k \circ \psi_1, \ldots, \mathcal{K}^k \circ \psi_N]^T = B^k \psi^T \in \mathbb{R}^{N \times m} \tag{2.23}$$

and in Eq. (2.21)

$$\mathcal{K} \circ \phi = (a^T B \psi^T)^T. \tag{2.24}$$

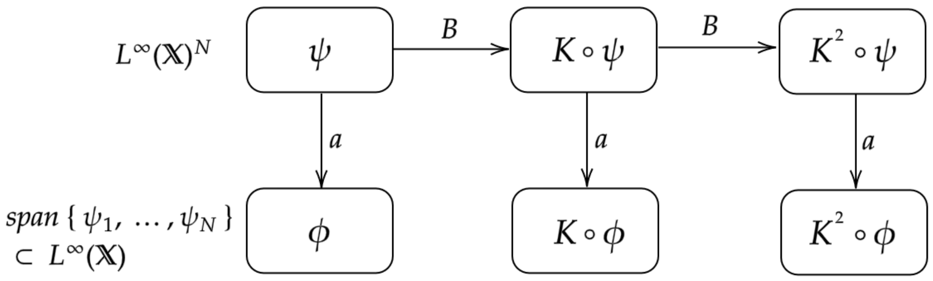


Figure 2.3: Illustration of propagation of observables with the Koopman operator in a Koopman-invariant subspace $span\{\psi_1, \ldots, \psi_N\}$.

Eigenvalues of the Koopman operator

Since the Koopman operator is linear, it has eigenvalues λ_i and corresponding eigenfunctions φ_i, $i = 1, 2, \ldots$ which span $L^\infty(\mathbb{X})$. Representing an observable in this basis as

$$\phi = \sum_{i=1}^{\infty} a_i \varphi_i, \quad a_i \in \mathbb{R}, \tag{2.25}$$

we find that we can easily obtain an alternative representation for the propagation of observables as

$$\begin{aligned} \phi \circ F &= \mathcal{K} \circ \phi = \sum_{i=1}^{\infty} a_i \mathcal{K} \circ \varphi_i = \sum_{i=1}^{\infty} a_i \lambda_i \varphi_i, \\ \phi \circ F^k &= \mathcal{K}^k \circ \phi = \sum_{i=1}^{\infty} a_i \lambda_i^k \varphi_i. \end{aligned} \tag{2.26}$$

If ϕ can be represented by only N of these eigenfunctions, the propagation of observables becomes even simpler compared to Eq. (2.21):

$$\text{Let } \phi = \sum_{i=1}^{N} a_i \varphi_i \Rightarrow \mathcal{K}^k \circ \phi = \sum_{i=1}^{N} a_i \lambda_i^k \varphi_i. \tag{2.27}$$

For this, one does not even have to perform a matrix multiplication, since merely multiplication with scalars, the eigenvalues λ_i, is sufficient.

In this section, we will discuss several different methods which all aim at finding a suitable finite-dimensional approximation of the action of $\mathcal{K}$ onto functions. This is done either by estimating the matrix B, or directly estimating $a^T B$ or by detecting eigenvalues and eigenfunctions of $\mathcal{K}$ to utilize the observation from Eq. (2.27). Some of these methods are directly suitable without memory – note that the above considerations did not involve memory either – but some do require memory terms. Although some of these methods come from very distinct fields of research, we will see that they are intimately connected and can in fact be seen as special cases of the same problem. We will also see how the Mori–Zwanzig formalism directly inspires some of the methods.

Remark. On the previous pages we defined the basis functions ψ_i as maps from $\mathbb{X}$ to $\mathbb{R}^m$. When introducing practical methods in the next few pages, we will define them as coming from the set $\mathcal{H}_\phi$ from Eq. (1.33). This is no contradiction, since functions in $\mathcal{H}_\phi$ also map from $\mathbb{X}$ to $\mathbb{R}^m$ but with the property that information of $\phi(X) = x$ is enough to evaluate them. For this reason, we can also apply them to values of the observable ϕ and write $\psi_i(x)$, as we did when introducing the Mori–Zwanzig formalism. Then this simply means $\psi_i(\phi(X))$. For the practical methods introduced in this chapter and the next, one can therefore perceive these basis functions as mapping from $\mathbb{R}^m$ to $\mathbb{R}^m$ without danger and in Chapters 3 and 4, we will use them as such. For some theoretical results that follow the introduction of the methods, we will utilize that the functions actually map from $\mathbb{X}$ to $\mathbb{R}^m$.

2.3.2 Dynamic Mode Decomposition (DMD)

Dynamic Mode Decomposition (DMD) is a method that was developed in the context of fluid dynamics in 2010 [Sch10]. It aims at identifying a linear mapping which propagates states of an observable without further basis functions. The observable can also be the identity so that the method is applicable – and actually originates from – outside of the context of the propagation of observables. Please note that in this section we will always assume to be given data of the observable ϕ evaluated at states X, giving $\phi(X) =: x \in \mathbb{R}^m$, even when a method, such as DMD, is often used on the full states. In this case simply assume $\phi(X) = X$.

Given data

$$\mathbf{X} = \begin{bmatrix} x_0 & \dots & x_{T-1} \end{bmatrix} \in \mathbb{R}^{m \times T}, \quad \mathbf{X}' = \begin{bmatrix} x_1 & \dots & x_T \end{bmatrix} \in \mathbb{R}^{m \times T}, \tag{2.28}$$

one tries to find a matrix $\mathbf{K}_{\mathrm{DMD}} \in \mathbb{R}^{m \times m}$ which fulfils

$$\mathbf{K}_{\mathrm{DMD}} = \underset{\mathbf{K} \in \mathbb{R}^{m \times m}}{\arg\min} \|\mathbf{X}' - \mathbf{K}\mathbf{X}\|_F, \tag{2.29}$$

where $\|\cdot\|_F$ is the Frobenius norm. For linear dynamics driven by a function $F(X) = AX$ with a matrix A, one finds that A is a minimizer of Eq. (2.29) with a loss of 0. For nonlinear dynamics, a linear mapping will usually not suffice, so that there remains an error between $\mathbf{X}'$ and $\mathbf{K}\mathbf{X}$. Nevertheless, the minimizer is given by

$$\mathbf{K}_{\mathrm{DMD}} = \mathbf{X}'\mathbf{X}^{+} \tag{2.30}$$

where $\mathbf{X}^{+}$ is the Moore-Penrose pseudo-inverse of $\mathbf{X}$ [Pen55] (it can directly be computed using the *Singular Value Decomposition* of $\mathbf{X}$). If $m, rank(\mathbf{X}) \leq T$, the matrix $\mathbf{X}\mathbf{X}^T$ is invertible and the then unique solution can equivalently be written as

$$\mathbf{K}_{\mathrm{DMD}} = \mathbf{X}'\mathbf{X}^T(\mathbf{X}\mathbf{X}^T)^{-1} \tag{2.31}$$

by constructing the so-called *normal equation*.

One then defines dynamics

$$\tilde{x}_{t+1} = \mathbf{K}_{\mathrm{DMD}} x_t. \tag{2.32}$$

At this point, DMD is not more than a multidimensional linear regression for sequential data points. The eventual aim of DMD, however, goes beyond that. It was mentioned that DMD stems from analysis of fluid dynamics. In this field, the dimension d of the states can be very high and in fact exceed the number of available data points T. As a result, the minimizer of the problem in Eq. (2.29) is not unique. One can, however, determine T pairs of eigenvalues λ_i and eigenvectors v_i of all minimizers with the help of specifically designed algorithms. This helps to, firstly, uncover dominant structures of the dynamics through interpretation of the eigenvectors and, secondly, allows to compute the evolution of the dynamics starting at initial states which can be written in the basis of the available eigenvectors.

Let $v_1, \dots, v_m$ be right eigenvectors of $\mathbf{K}_{\text{DMD}}$ and let

$$x_1 = \sum_{i=1}^{m} a_i v_i, \quad \text{then} \quad \mathbf{K}_{\text{DMD}}^k x_1 = \sum_{i=1}^{m} a_i \lambda_i^k v_i. \tag{2.33}$$

Note that through a linear modelling of dynamics, there are only three possibilities for the long-term behaviour of predictions. Let x_1 be given as in Eq. (2.33) with $a_i \neq 0$ for all $i = 1, \dots, m$. Then we must distinguish the following three cases whose results are directly implied by Eq. (2.33).

1. There exists an eigenvalue λ_i with $|Re(\lambda_i)| > 1$ with $a_i \neq 0$. Then $\lim\limits_{k\to\infty} \mathbf{K}_{\text{DMD}}^k x_1 = \pm\infty$.
2. $|Re(\lambda_i)| < 1$ for all $a_i \neq 0$. Then $\lim\limits_{k\to\infty} \mathbf{K}_{\text{DMD}}^k x_1 = 0$.
3. $|Re(\lambda_i)| \leq 1$ for all i and equal to 1 for $\lambda_{i_1}, \dots, \lambda_{i_l}$. Depending on the complex part of the eigenvalues, this prediction then either converges to a fixed point or oscillates.

These observations imply that for nonlinear dynamics a linear model is typically ill-suited for long-term predictions. Still, for short-term predictions, they can provide a sufficiently accurate approximation.

Let us turn back to algorithmic aspects of prediction with linear models: Even if we can find a unique minimizer in Eq. (2.29), prediction as in Eq. (2.32) yields the multiplication with a large matrix, which can be computationally time-consuming. The representation of a state through eigenvectors is hence valuable even if $T \geq m$ because the computation becomes much simpler through Eq. (2.33).

For the determination of eigenpairs of minimizers of Eq. (2.29), there exist several algorithms. Although these algorithms represent a vital addition to the method and in the case $T < m$ are even its focal point, we will not use them in the continuation of this thesis and therefore refer to [WKR14, TRL$^+$14, KNK$^+$18].

2.3.3 Extended Dynamic Mode Decomposition (EDMD)

In the previous subsection we have seen that through DMD one can find eigenpairs of the best linear approximation of the dynamics F. As previously explained, the Koopman operator provides a way to transform a nonlinear dynamical system into an infinite-dimensional linear one. It would be tempting to expect the existence of a method similar to DMD which discovers not eigenvalues and -vectors of a linear approximation of the dynamics but instead eigenvalues and -functions of the Koopman operator. This would be valuable to be able to employ Eq. (2.26) and propagate observables using these eigenpairs. There exists an extension of DMD, simply called Extended Dynamic Mode Decomposition (EDMD) [WKR14], which aims at this.

Let us remember the function space $\mathcal{H}_\phi$ from Eq. (1.33) where here $\mathbb{Y} = \mathbb{R}^m$. For EDMD one chooses a set of N scalar-valued basis functions $\psi_1, \dots, \psi_N \in \mathcal{H}_\phi$, denoted by $\psi = [\psi_1, \dots, \psi_N]^T$, and replaces the matrices $\mathbf{X}$ and $\mathbf{X}'$ by

$$\begin{aligned} \psi(\mathbf{X}) &= \begin{bmatrix} \psi(x_0) & \dots & \psi(x_{T-1}) \end{bmatrix} \in \mathbb{R}^{N \times T}, \\ \psi(\mathbf{X}') &= \begin{bmatrix} (\mathcal{K} \circ \psi_1)(x_0) & \dots & (\mathcal{K} \circ \psi_1)(x_{T-1}) \\ \vdots & & \vdots \\ (\mathcal{K} \circ \psi_N)(x_0) & \dots & (\mathcal{K} \circ \psi_N)(x_{T-1}) \end{bmatrix} = \begin{bmatrix} \psi(x_1) & \dots & \psi(x_T) \end{bmatrix} \in \mathbb{R}^{N \times T}. \end{aligned} \tag{2.34}$$

Note that this should remind us of Eq. (2.19) where we could connect both these matrices by multiplication with a matrix B under the assumption that the basis functions form a Koopman-invariant subspace.

In fact, in EDMD, one strives to solve

$$\mathbf{K}_{\text{EDMD}} = \underset{\mathbf{K} \in \mathbb{R}^{N \times N}}{\arg\min} \|\psi(\mathbf{X}') - \mathbf{K}\psi(\mathbf{X})\|_F. \tag{2.35}$$

In the rare case that these basis functions actually form a Koopman–invariant subspace, the loss function in Eq. (2.35) can be set to 0 due to the relation in Eq. (2.19). Generally, $\mathbf{K}_{\text{EDMD}}$ is the *matrix approximation* to the action of the Koopman operator on $span\{\psi_1, \dots, \psi_N\}$. Analogously to DMD, EDMD goes beyond that. As the following proposition shows, with the left eigenvectors of $\mathbf{K}_{\text{EDMD}}$ we can approximate eigenfunctions of the Koopman operator:

Proposition 2.2. Let $\mathbf{K}_{\text{EDMD}}\psi(\cdot) = \mathcal{K} \circ \psi$ where $\mathcal{K}$ is the Koopman operator associated with F. Let u be a left eigenvector of $\mathbf{K}_{\text{EDMD}}$ with eigenvalue λ. Then the function $u^T\psi$ is an eigenfunction of $\mathcal{K}$ with the eigenvalue λ.

Proof. As demonstrated in [KKS16], it holds,

$$\mathcal{K} \circ (u^T\psi) = u^T(\mathcal{K} \circ \psi) = u^T\mathbf{K}_{\text{EDMD}}\psi = \lambda u^T\psi. \tag{2.36}$$

□

Note that in the proposition we assumed exactness of the linear mapping $\mathbf{K}_{\text{EDMD}}$ to $\mathcal{K}$ in the space spanned by the basis functions $\psi_1, \dots, \psi_N$. Normally, this will be an approximation so that the functions $u^T\psi$ are only approximations of Koopman eigenfunctions, too.

Still, utilizing the following observation, the possibility to approximate some Koopman eigenfunctions leads to further opportunities:

Lemma 2.3. Let $\varphi_1, \dots, \varphi_n$ be eigenfunctions of $\mathcal{K}$ to the eigenvalues $\lambda_1, \dots, \lambda_n$. Then the function $\tilde{\varphi} := \prod_{i=1}^n \varphi_i$ is an eigenfunction of $\mathcal{K}$ to the eigenvalue $\prod_{i=1}^n \lambda_i$.

Proof. As done in [KKS16], we can assert,

$$\mathcal{K} \circ \tilde{\varphi} = \mathcal{K} \circ \prod_{i=1}^{n} \varphi_i = \prod_{i=1}^{n} \varphi_i \circ F = \prod_{i=1}^{n} \mathcal{K} \circ \varphi_i = \prod_{i=1}^{n} \lambda_i \varphi_i = \tilde{\varphi} \prod_{i=1}^{n} \lambda_i. \tag{2.37}$$

□

These two results imply that from the eigenvectors of the matrix $\mathbf{K}_{\text{EDMD}}$ we can construct a toolbox of eigenfunctions of $\mathcal{K}$ through the products of the ensuing eigenfunctions. This helps to model the action of the Koopman operator in the economic format using the representation of observables in the basis of these eigenfunctions as in Eq. (2.27). Using $\varphi_i = u_i^T \psi$ and $\mathcal{K} \circ \varphi_i = \lambda_i u_i^T \psi$, Eq. (2.27) gives

$$\mathcal{K}^k \circ \phi = \sum_{i=1}^{N} \lambda_i^k a_i u_i^T \psi. \tag{2.38}$$

In this light, let us reconsider the non-extended DMD method: although here being motivated as a method to find eigenvectors of the best linear approximation to a dynamical system, we can simply view it as EDMD with $d = m$ and the d-dimensional identity function as the vector of basis functions ψ. We then find *linear* eigenfunctions of the form $\varphi(X) = u^T X$. Using Lemma 2.3, this is especially helpful since from this we can find polynomial eigenfunctions in all orders, allowing us to build a rich toolbox of usable eigenfunctions. The question now might arise: why then even use nonlinear basis functions since they do not easily allow construction of simple lower-order functions, e.g., linear ones. The answer is that the quality of approximation of the true eigenfunctions of the Koopman operator hinges on the quality of the approximation of its action on $span\{\psi_1, \dots, \psi_N\}$ with $\mathbf{K}_{\text{EDMD}}$. For a linear basis and a nonlinear system, this approximation is prone to be poor. Therefore, one is tasked with selecting a suitable set of basis functions so that the loss function in Eq. (2.35) is small. Then the approximations of Koopman eigenfunctions can be good.

For the coefficients $a = [a_1, \dots, a_N]^T$ that are used in order to project the basis functions onto the observable of interest $\phi : \mathbb{X} \to \mathbb{R}^m$, one should additionally solve the following optimization problem,

$$a_{\text{EDMD}} = \underset{a \in \mathbb{R}^{m \times N}}{\arg\min} \|\mathbf{X} - a\psi(\mathbf{X})\|_F. \tag{2.39}$$

Note that contrary to the motivation of the Koopman framework previously, for $m > 1$, a_{EDMD} is a matrix and not a vector. The reason is that in EDMD, the basis functions are $N \times 1$-dimensional while the observable ϕ is m-dimensional. Still, a few reformulations can align EDMD precisely with the Koopman framework.

2.3.4 Sparse Identification of Nonlinear Dynamics (SINDy)

In EDMD, we tried to map the value of a vector-valued basis function ψ evaluated at a point x_t to its subsequent value $\psi(x_{t+1})$ through a matrix $\mathbf{K}_{\text{EDMD}}$. From there, $\psi(x_{t+1})$ is mapped to x_{t+1} by multiplication with a coefficient matrix a_{EDMD}. The main point of EDMD was to extract eigenpairs of the Koopman operator from the matrix $\mathbf{K}_{\text{EDMD}}$, offering a way to construct observables whose evolution can be computed in a simple way. If one is rather interested in predicting a specific observable – again, this can include the original state – then this two-step procedure might not be optimal since one requires an accurate mapping from ψ to $\mathcal{K} \circ \psi$ and then one from $\mathcal{K} \circ \psi$ to $\mathcal{K} \circ \phi$. The set of those mappings is thus only a subset of the set of all mappings from ψ to $\mathcal{K} \circ \phi$ while the benefit of this self-constraint is the identification of eigenpairs of $\mathcal{K}$. A similar method which determines a direct linear mapping from ψ to $\mathcal{K} \circ \phi$ is Sparse Identification of Nonlinear Dynamics (SINDy) [BPK16, BLK16, KKB18]. It works in the following way:

Given data from an m-dimensional observable and N scalar-valued basis functions,

$$\psi(\mathbf{X}) = \begin{bmatrix} \psi(x_0) & \dots & \psi(x_{T-1}) \end{bmatrix} \in \mathbb{R}^{N \times T}, \quad \mathbf{X}' = \begin{bmatrix} x_1 & \dots & x_T \end{bmatrix} \in \mathbb{R}^{m \times T}, \tag{2.40}$$

one solves

$$A_{\text{SINDy}} = \underset{A \in \mathbb{R}^{m \times N}}{\arg\min} \|\mathbf{X}' - A\psi(\mathbf{X})\|_F + c\|A\|_1. \tag{2.41}$$

The first term represents a nonlinear dynamic least squares regression since coefficients are found which map $\psi(x_t)$ to x_{t+1} in an optimal way regarding the data, according to the Frobenius norm. The second term denotes a sparsity constraint since it rewards choosing matrices with few non-zero entries. Sparse models typically admit a readier interpretation of the model. $c > 0$ is a sparsity regularization parameter which has to be specified. We further denote $\|A\|_1 = \sum\limits_{i=1}^{m} \sum\limits_{j=1}^{N} |A_{ij}|$. The optimization problem is solved using the LASSO algorithm [Tib96].

With A_{SINDy} determined, one can approximate the evolution of x by

$$\tilde{x}_{t+1} = A_{\text{SINDy}} \psi(\tilde{x}_t). \tag{2.42}$$

While in EDMD we are forced to select basis functions which give a closed linear system between them, in SINDy we can choose any set of basis functions. By including more functions into the basis, the loss of the optimal solution in Eq. (2.41) cannot get bigger. In order to still obtain an economic and interpretable model, one enforces the sparsity constraint. However, there is a danger of overfitting: the optimal coefficients in the matrix A_{SINDy} might not be unique and put emphasis on basis functions which are only suitable in the region of the state space where the training data lie. For other regions, they might not be applicable and yield nonsensical forecasts, often diverging ones. As we will see later on, this is one of the weakest points of SINDy.

The following results assert the sensitivity of SINDy on its basis functions and the available data:

Theorem 2.4. Let A be the solution of Eq. (2.41) with $c = 0$. Let $\Delta\mathbf{X} \in \mathbb{R}^{m\times N}$ and let $A + \Delta A$ be the solution of the perturbed Eq. (2.41) with $\mathbf{X}'$ replaced by $\mathbf{X}' + \Delta\mathbf{X}'$ and $\psi(\mathbf{X})$ replaced by $\psi(\mathbf{X} + \Delta\mathbf{X})$. Define

$$\|\mathbf{X}\|_{k\Sigma} := \sqrt{\sum_{t=0}^{T-1} \|x_t\|_F^k} \text{ and } \|\mathbf{X},\mathbf{X}'\|_{k\Sigma} := \sqrt{\sum_{t=0}^{T-1} \|x_t\|_F^k \|x_{t+1}\|_F^k}.$$

Further let the following assumptions hold

(A1) There exist $\psi_{max}, \psi_{min} > 0$ with

$$\psi_{min} \leq \frac{|\psi_i(x) - \psi_i(y)|}{\|x-y\|_2} \leq \psi_{max} \text{ for all } i = 1,\dots,N \text{ and all } x \neq y,$$
$$\psi_{min} \leq \frac{|\psi_i(x)|}{\|x\|_2} \leq \psi_{max} \text{ for all } i = 1,\dots,N \text{ and all } x \neq 0 \in \mathbb{R}^m.$$

(A2) $\psi(\mathbf{X})$ has full row-rank.

(A3) $\|\Delta\mathbf{X}\|_{4\Sigma} < \frac{1}{\|(\psi(\mathbf{X})\psi(\mathbf{X})^T)^{-1}\|_F \psi_{max}^2 N}$.

Then it holds,

$$\frac{\|\Delta A\|_F}{\|A\|_F} \leq \frac{\|(\psi(\mathbf{X})\psi(\mathbf{X})^T)^{-1}\|_F \psi_{max}^2 N \|\mathbf{X}\|_{4\Sigma}}{1 - \|(\psi(\mathbf{X})\psi(\mathbf{X})^T)^{-1}\|_F \psi_{max}^2 N \|\Delta\mathbf{X}\|_{4\Sigma}}$$
$$\left(\frac{\psi_{max}(\|\mathbf{X}'\|_F\|\Delta\mathbf{X}\|_F + \|\Delta\mathbf{X}'\|_F\|\mathbf{X}\|_F + \|\Delta\mathbf{X}'\|_F\|\Delta\mathbf{X}\|_F)}{\psi_{min}\|\mathbf{X},\mathbf{X}'\|_{2\Sigma}} + \frac{\psi_{max}^2\|\Delta\mathbf{X}\|_{4\Sigma}}{\psi_{min}^2\|\mathbf{X}\|_{4\Sigma}}\right). \tag{2.43}$$

Proof. The proof can be found at the end of this thesis in the Appendix. □

All technicalities of Theorem 2.4 aside, it shows that the maximal relative perturbation of the SINDy result under perturbations of the data points in $\mathbf{X}$, which are then translated to values in $\psi(\mathbf{X})$, is generally large if the ratio ψ_{max}/ψ_{min} is large. Therefore, one should ideally choose well-balanced basis functions, i.e., ones that scale similarly. This makes intuitive sense: if $k < N$ basis functions dominate in magnitude, then $\psi(\mathbf{X})$ will be close to a rank-k matrix. From standard perturbation analysis (see, e.g., [GVL96]), the result of the regression problem will be more sensitive to perturbation.

Remark. Note that in Eq. (2.43) there occur three different metrics and we divide $\|\mathbf{X}'\|_F \|\Delta \mathbf{X}\|_F$ by $\|\mathbf{X}, \mathbf{X}'\|_{2\Sigma}$ (Σ is chosen as symbol since one *sums* up the norms of the columns of a matrix). The orders of magnitude are comparable, since in both terms we take the square root across a sum of degree-4 monomials. The ratio between the two then roughly scales with the average ratio between $\|\Delta x_t\|_2$ and $\|x_t\|_2$. Regarding the assumptions, (A1) seems to be quite strong and is not true for many families of basis functions. However, it does hold for all monomials if restricted to a bounded domain. Important for the proof is that we can upper- and lower-bound the action of each ψ_i on a point x. If only one of the two inequalities in (A1) is fulfilled, additional affine terms are introduced in the result but the general theme is preserved.

We can further observe:

Proposition 2.5. Let the basis functions $\psi_1, \ldots, \psi_N$ be linearly dependent. Then the solution to Eq. (2.41) is not unique.

Proof. Let without loss of generality $\psi_1 = \sum_{i=2}^{N} b_i \psi_i$ and let A be the minimizer of Eq. (2.41). Then one can subtract an arbitrary real value d from each entry in the first column of A and for each i add $b_i d$ to each value in the ith column of A and the product with $\psi(\mathbf{X})$ remains unchanged. □

In Figure 2.4 is depicted a conceptual comparison of the way in which models determined by DMD, EDMD and SINDy propagate observables over time. Note that there exists recent literature on the application of SINDy on low-dimensional representations of points. The low-dimensional representation is obtained separately through a neural network in [FMZF21] and found simultaneously with the model for the propagation in a single optimization problem in [CLKB19]. In Chapter 5, we will introduce a novel method which follows a similar intuition.

Note that all methods discussed in this section so far are Markovian. Since in Chapter 1 we concluded that generally we require memory to propagate observables over time, do they actually fit into the setting? Although not using memory, in these three methods the hope is that not through *additional* (by memory) but through *transformed* (by basis functions) information we can model the dynamics. We will now discuss similar alternative methods that do use memory and later on show that the inclusion of memory can be seen as a transformation with basis functions, bridging the gap to the so-far introduced methods.

2.3.5 Autoregressive Moving Average Models (ARMA)

In this section, we have so far discussed multiple least squares based regression methods which model a dynamical system without memory, regardless of whether

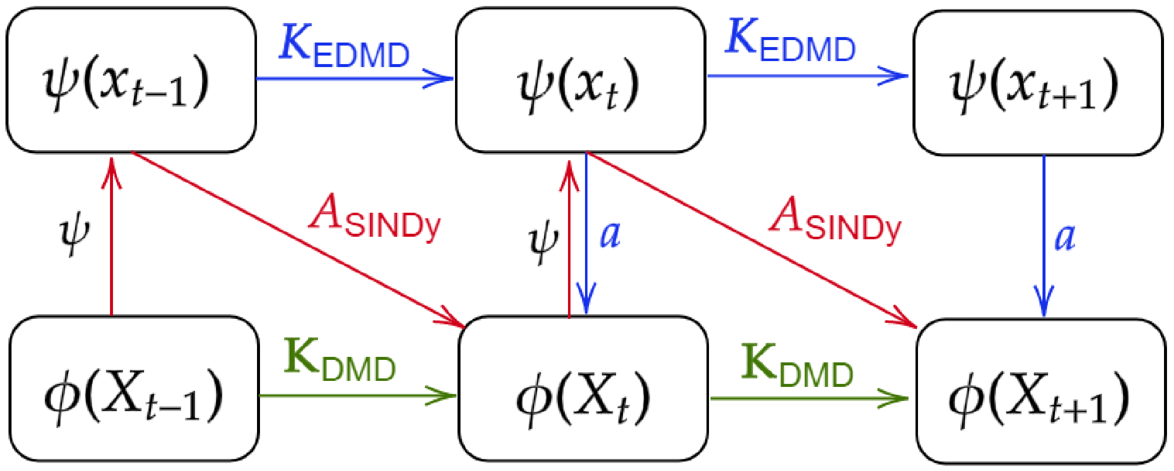


Figure 2.4: Illustration of the ways in which DMD, EDMD and SINDy propagate observables. It shows how the algorithms approximate the action of the Koopman operator and the multiplication with the coefficient vector a shown in Figure 2.3. Note that in this figure a is a matrix and not a vector. DMD propagates states directly with a linear mapping $\mathbf{K}_{\mathrm{DMD}}$. Often times, in DMD ϕ is the identity. EDMD seeks a matrix $\mathbf{K}_{\mathrm{EDMD}}$ which progresses basis functions over time. From there, one can construct observables in the span of the basis functions through multiplication with a coefficient matrix. In SINDy, one seeks a direct mapping from the basis functions to a desired observable at the future time step and transforms back by application of ψ.

the system exhibits memory effects or not. Next, we will discuss a class of methods which is designed specifically for uncovering memory-exhibiting dynamics that are predominantly linear. These are the so-called Autoregressive Moving Average (ARMA) models, consisting of two essential subclasses, the *autoregressive (AR)* and *moving-average (MA)* models. They read

$$\text{ARMA:} \quad x_{t+1} = \theta_1 x_t + \cdots + \theta_p x_{t-p+1} + \beta_1 \varepsilon_t + \cdots + \beta_q \varepsilon_{t-q+1} \tag{2.44}$$

$$\text{AR:} \quad x_{t+1} = \theta_1 x_t + \cdots + \theta_p x_{t-p+1} + \varepsilon_{t+1} \tag{2.45}$$

$$\text{MA:} \quad x_{t+1} = \beta_1 \varepsilon_{t+1} + \cdots + \beta_q \varepsilon_{t-q+2} \tag{2.46}$$

with $\theta_i, \beta \in \mathbb{R}^{m\times m}, \varepsilon_t \sim \mathcal{N}(\mu, \Sigma^T\Sigma), \mu \in \mathbb{R}^m, \Sigma \in \mathbb{R}^{m\times m}$.

In AR models, past states influence future ones in the short term linearly according to the θ_i for small i while long-term influences are characterised by the θ_i for large i. Note that if $\mu \neq 0$, we can always add the term μ to the model formulations and define ε_t as zero-mean noise terms. MA models produce entirely stochastic processes by using a fixed number q of random noise terms to construct subsequent states.

One could easily devote a book on its own to ARMA methods and go into more detail (as done in [BD91]). In this thesis it is explicitly chosen to explain AR models as one class of Koopman-based data-driven modelling methods to show the relations between them and others and since their partly-deterministic structure better positions them in the context of dynamical systems modelling compared to MA models, which are entirely stochastic. For more information on both AR, MA and ARMA models, together with the relation between AR and MA models – including the option to transform one into the other – consider, e.g., [BD91, Bil13, Neu16]. Moreover, there exist model families which are extensions of ARMA models, such as ARIMA [Hyn18], GARCH [Bol86] or ARMAX [CVM04] models or combinations of these. Note that all these are linear models while later on we will present nonlinear extensions.

AR models

As a seemingly technical observation, let us record that an m-dimensional AR(p) model can be reformulated as an mp-dimensional AR(1) model,

$$x_{t+1} = \theta_1 x_t + \cdots + \theta_p x_{t-p+1} + \varepsilon_{t+1}$$

$$\Leftrightarrow \begin{bmatrix} x_{t+1} \\ \vdots \\ x_{t-p+2} \end{bmatrix} = \underbrace{\begin{bmatrix} \theta_1 & & \dots & \theta_p \\ 1 & 0 & \dots & 0 \\ & \ddots & & \\ 0 & \dots & 1 & 0 \end{bmatrix}}_{=:C} \begin{bmatrix} x_t \\ \vdots \\ x_{t-p+1} \end{bmatrix} + \begin{bmatrix} \varepsilon_{t+1} \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{2.47}$$

resulting in $\hat{x}_{t+1} = C\hat{x}_t + \hat{\varepsilon}_{t+1}$.

The matrix C is called a *companion matrix*. Eq. (2.47) implies that we can formulate an AR(p) model as a memoryless linear dynamical system. Denote by $v_1, \dots, v_{pm}$ the right eigenvectors of C. If $\mathbb{E}[\varepsilon_{t+1}] = 0$, then in expectation we find analogously to linear models without memory,

$$\hat{x}_t = \sum_{i=1}^{pm} a_i v_i \Rightarrow \mathbb{E}[\hat{x}_{t+k}] = C^k \hat{x}_t = \sum_{i=1}^{pm} a_i \lambda_i^k v_i. \tag{2.48}$$

In the way illustrated in Subsection 2.3.2 on DMD, from the eigenvalues of C we can observe the expected long-term behaviour of predictions of the AR(1) model above and thus of the AR(p) model in Eq. (2.46).

If $\mathbb{E}[\varepsilon_{t+1}] \neq 0$, the evolution becomes slightly more complex. Let $\mathbb{E}[\varepsilon_t] = \mu$ for all $t > 0$ and let $\mu = \sum_{i=1}^{pm} b_i v_i$. Then

$$\mathbb{E}[\hat{x}_{t+k}] = C^k \hat{x}_t + \sum_{j=0}^{k-1} C^j \mu = \sum_{i=1}^{pm} a_i \lambda_i^k v_i + \sum_{j=0}^{k-1} \sum_{i=1}^{pm} \lambda_i^j b_i v_i. \tag{2.49}$$

If the terms $C^j \mu$ in the above equation decay fast, i.e., the eigenvalues of C are sufficiently smaller than 1, it becomes clear that after many iterations the ensuing long-term behaviour is in expectation dominated by the most recent noise terms and not by the starting value. More precisely, with $l < k$,

$$\mathbb{E}[\hat{x}_{t+k}] = \sum_{j=0}^{l} \sum_{i=1}^{pm} \lambda_i^j b_i v_i + o(\lambda_1^{l+1}), \tag{2.50}$$

where λ_1 denotes the largest eigenvalue of C. If any eigenvalue with non-zero coefficients is bigger than 1, we can see that the process will diverge in expectation.

Parameter estimation of AR models

Let us now investigate how to estimate AR models which best approximate the dynamics behind a given time series. Let data points $x_0, \ldots, x_T$ be given and define

$$\begin{aligned} \mathbf{H} &= \begin{bmatrix} x_{p-1} & \cdots & x_{T-1} \\ \vdots & & \vdots \\ x_0 & \cdots & x_{T-p} \end{bmatrix} = \begin{bmatrix} \hat{x}_{p-1} & \cdots & \hat{x}_{T-1} \end{bmatrix} \in \mathbb{R}^{mp \times T-p+1}, \\ \mathbf{X}' &= \begin{bmatrix} x_p & \cdots & x_T \end{bmatrix} \in \mathbb{R}^{m \times T-p+1}. \end{aligned} \tag{2.51}$$

H is called a *Hankel matrix*.

To construct an AR(p) model, we determine coefficients $\theta = [\theta_1, \ldots, \theta_p] \in \mathbb{R}^{m \times mp}$ by solving

$$\theta_{\text{AR}} = \underset{\theta \in \mathbb{R}^{m \times pm}}{\arg\min} \|\mathbf{X}' - \theta \mathbf{H}\|_F. \tag{2.52}$$

The solution to this is

$$\theta_{\text{AR}} = \mathbf{X}'\mathbf{H}^+. \tag{2.53}$$

If the matrix $\mathbf{H}\mathbf{H}^T$ is invertible, the solution can equivalently be written as

$$\theta_{\text{AR}} = \mathbf{X}'\mathbf{H}^T(\mathbf{H}\mathbf{H}^T)^{-1}. \tag{2.54}$$

Clearly, the minimization problem in Eq. (2.52) seems not to take the noise terms into account. However, if the data actually come from an AR(p) model with coefficients $\theta^\star = [\theta_1^\star, \ldots, \theta_p^\star]$ and noise terms distributed by $\mathcal{N}(0, \Sigma^T\Sigma)$, the solution given in Eq. (2.53) is the *best linear unbiased estimator* (BLUE) of the coefficients by the Gauss-Markov-Theorem [BH05]. This means that it is the best estimator of the true underlying coefficients $\theta^\star$ whose expectation is equal to $\theta^\star$ and that can be constructed using a linear combination of the data. If the noise terms have a non-zero mean μ, one can always augment the Hankel matrix by a row of ones to estimate an affine term as the estimate of μ.

For convergence results of the estimator θ_{AR}, please see [Lie03, Yao00]. The results therein assert that with enough data, a simple least squares estimate generally produces a good estimate of the coefficients of an AR process.

Taking the reformulation of an AR model as a memoryless linear system into account, we can view the AR regression problem in an alternative way by defining

$$\mathbf{H}' = \begin{bmatrix} x_p & \dots & x_T \\ \vdots & & \vdots \\ x_1 & \dots & x_{T-p+1} \end{bmatrix} = \begin{bmatrix} \hat{x}_p & \dots & \hat{x}_T \end{bmatrix} \in \mathbb{R}^{mp \times T-p+1}. \tag{2.55}$$

Then the matrix

$$C_{\text{AR}} = \underset{C \in \mathbb{R}^{pm \times pm}}{\arg\min} \|\mathbf{H}' - C\mathbf{H}\|_F. \tag{2.56}$$

connects columns of $\mathbf{H}$ and $\mathbf{H}'$ in a linear way. We are thus in the same setting as in DMD where we seek a linear mapping between points in $\mathbb{R}^m$. In this case, we view *sequences* of the form $x_{t-1}, \dots, x_{t-p} \in \mathbb{R}^m$ as *points* of the form $[x_{t-1}^T, \dots, x_{t-p}^T]^T \in \mathbb{R}^{pm}$ to arrive at a memoryless linear regression problem. C_{AR} is itself a companion matrix with the estimate θ_{AR} from Eq. (2.53) in the first m rows, as the following theorem states.

Theorem 2.6. Let $x_0, \dots, x_T \in \mathbb{R}^m$ and let $\mathbf{H}$ and $\mathbf{H}'$ be defined as before. Then C_{AR} from Eq. (2.56) is a companion matrix whose first m rows are given by θ_{AR} from Eq. (2.53).

Proof. Define a matrix $C \in \mathbb{R}^{pm \times pm}$ and denote its first m rows by θ and the remaining rows by E. Denote the rows $m+1, \dots, pm$ of $\mathbf{H}'$ by $\mathbf{H}'_{>m}$.

It trivially holds that $\underset{\theta \in \mathbb{R}^{pm \times pm}}{\arg\min} \|\mathbf{H}' - C\mathbf{H}\|_F = \underset{\theta \in \mathbb{R}^{pm \times pm}}{\arg\min} \|\mathbf{H}' - C\mathbf{H}\|_F^2$. The latter can be written as

$$\begin{aligned} \|\mathbf{H}' - C\mathbf{H}\|_F^2 &= \sum_{j=1}^{T-p+1} \sum_{i=1}^{pm} (\mathbf{H}'_{ij} - (C\mathbf{H})_{ij})^2 \\ &= \sum_{j=1}^{T-p+1} \sum_{i=1}^{m} (\mathbf{H}'_{ij} - (C\mathbf{H})_{ij})^2 + \sum_{j=1}^{T-p+1} \sum_{i=m+1}^{pm} (\mathbf{H}'_{ij} - (C\mathbf{H})_{ij})^2 \\ &= \|\mathbf{X}' - \theta\mathbf{H}\|_F^2 + \|\mathbf{H}'_{>m} - E\mathbf{H}\|_F^2. \end{aligned} \tag{2.57}$$

We have thus split Eq. (2.56) into two separate minimization problems. Clearly, the first of these terms is the minimization problem from Eq. (2.53) while in the latter E should simply copy the first $pm - m$ entries of a column of $\mathbf{H}$ into a column of $\mathbf{H}'_{>m}$. In total, we derive

$$C_{\text{AR}} = \begin{bmatrix} & & \theta_{\text{AR}} & & \\ 1 & 0 & \dots & & 0 \\ 0 & 1 & 0 & \dots & 0 \\ & & \ddots & & \\ 0 & 0 & \dots & 1 & 0 \end{bmatrix}. \tag{2.58}$$

□

With Theorem 2.6, we have established an equivalence between DMD an AR models. AR models are essentially linear memoryless models, as determined by DMD, only in an augmented space. This yields the opportunity to exploit the DMD algorithms used to determine eigenvalues of the companion matrix C_{AR} in order to simulate AR models more efficiently in the eigenvector basis.

As stated in [AM17], the eigenvalues of C_{AR} converge to true eigenvalues of the Koopman operator for the dynamical system in the data limit. The authors denote DMD applied to the Hankel matrix by *Hankel-DMD* and derive further theoretical results.

Note that the covariance of the noise terms can be estimated by the statistical covariance

$$\tilde{\Sigma} = \frac{1}{T-p+1} \sum_{i=p}^{T-1} (x_i - \theta_{\text{AR}} \hat{x}_i)(x_i - \theta_{\text{AR}} \hat{x}_i)^T. \tag{2.59}$$

If one does not assume that the expectation of the noise terms is 0 one should include an affine term into the parameter estimation, i.e., add a row of ones to the Hankel matrix. The ensuing minimization problem is then not equivalent to the DMD problem any more since we try to map $[1, \hat{x}_t^T]$ to $[\hat{x}_{t+1}^T]$. This is no contradiction to the arguments made before since DMD estimates linear, not affine, models. To reestablish equivalence in order to exploit the eigenvalues of the system matrix, one could also append a row of ones to $\mathbf{H}'$ and obtain a quadratic matrix again.

There are alternative methods to estimate the model coefficients, such as the Yule-Walker method [SS11], whose solution converges to θ_{AR} in the data limit. There also exist algorithms for the numerically efficient computation of θ_{AR} such as Burg's method [RKEV03] or the Levinson-Durbin method [Fra85].

The condition of parameter estimation and memory estimation

Until here, we have always assumed a fixed memory depth p. Generally, when tasked with estimating a suitable AR model one does not know an appropriate memory depth a priori. A bad choice can yield significant problems: if we choose the memory depth too low, we will be unable to derive a good fit between data points with any model coefficients. The error in Eq. (2.52) will indicate this and we would be urged to increase the memory depth. However, as it turns out, choosing the memory depth too high can also be problematic:

Proposition 2.7. Let $x_0, \dots, x_T$ come from an AR(p) model without noise. If the memory depth used in the optimization problem in Eq. (2.52) is bigger than p, then its solution is not unique.

Proof. If the memory depth is given by $p + k$ for a $k > 0$, then in the ensuing Hankel matrix $\mathbf{H}$, rows will be linearly dependent, since x_{t+p+k} depends linearly one $x_{t+p+k-1}, \dots, x_{t+k}$, bounding the rank of $\mathbf{H}$ from above by p. Then analogously to the proof for Proposition 2.5, we can subtract an arbitrary real value y from each entry in the first column of the coefficient matrix θ, which is multiplied with the first row of $\mathbf{H}'$, so with the terms $x_{p+k-1}, \dots, x_{T-1}$, and find suitable coefficients for the other rows so that the product between θ and $\mathbf{H}$ does not change. □

Let us assume the less extreme case that the data do not actually come from a noise-free AR model and that a memory depth of p does produce a good fit. Then with higher memory depths, while $\mathbf{H}\mathbf{H}^T$ might not be of rank p, its numerical rank will be close to it, meaning that it has only p singular values which are not almost 0. As a consequence, its condition number, given by the ratio between its biggest and smallest singular values, will be high, yielding high sensitivity to the data and thus an ill-conditioned parameter estimation problem with many very different but similarly good solutions. This is problematic and would require further uncertainty quantification, e.g., in the form of a Bayesian view on the parameter estimation. For an overview of the interplay between parameter estimation based on single optimization problems and uncertainty quantification consider, e.g., [WTH$^+$].

At this point there arises a similarity to the previously presented method SINDy. For SINDy, we noted the danger of overfitting when using too many different basis functions and asserted non-uniqueness of the optimal coefficient matrix in Proposition 2.5. Viewing the determination of AR coefficients as SINDy without sparsity constraint and with linear time-delayed basis functions, more precisely, $\psi(X) = [\phi(X), (\mathcal{K}^{-1} \circ \phi)(X), \dots, (\mathcal{K}^{-p+1} \circ \phi)(X)]$, the same danger lures in this setting: too large a memory depth essentially means using too many basis functions and can yield non-unique solutions.

It is therefore desirable to estimate the minimal required memory depth for an acceptable model accuracy. For this aim, common ways are the Akaike Information Criterion (AIC) and the Bayesian Information Criterion (BIC) for which we refer to [Aka74, WHR12]. Additionally, it would be advantageous if methods used to determine the memory depth in the context of Takens' Theorem could be leveraged to find a suitable memory depth for AR methods. The synchronisation of these two methods is left for future research.

The last few pages have provided an introduction of existing methods for the estimation of dynamical systems, especially those that focus on estimating the action of the Koopman operator. Most of it has been a summary of previous literature. In the next chapter, we will state novel results that show how they can easily be augmented to nonlinear forms and estimated using previously introduced techniques. Afterwards, we are prepared for a summarizing comparison and application of all presented Koopman-based methods.

Ensuing Research Questions

In the future, there seems to be large potential to combine the different model classes to produce more powerful methods. This is already happening, as, e.g., in [LKB18] where a neural network is used to determine eigenfunctions of the Koopman operator, in [MFM$^+$20, DR20] where the expensive training process of a neural network is interpreted as a dynamical system and approximated through Koopman eigenfunctions, in autoregressive neural networks [BNNB20] or in Kernel-based EDMD [MOW15]. For this, it seems important that the respective communities are in lively exchange.

Extension, Connection and Application of the Koopman Methods

In the previous chapter, we have become acquainted with different classes of numerical methods which can be used to model memory-exhibiting dynamics. Special focus was placed on the Koopman-based methods and the Koopman operator perspective. In this chapter, we will present a novel extension to these methods, formally establish connections between the Koopman-based methods and apply them and others of the previously presented ones to two examples. While in the first two chapters most of the results stemmed from prior research, most of the results over the next three chapters have been derived by the author.

3.1 Sparse Identification of Nonlinear Autoregressive Models (SINAR)

In this section, we will introduce a novel method (see also [WKS21] by the author of this thesis and co-authors) in order to estimate nonlinear variants of AR models. While AR models are linear models, let us analogously define a *nonlinear autoregressive model* (NAR) by specifying basis functions $\hat{\psi} = [\psi_1, \dots, \psi_N]^T : \mathbb{R}^{pm} \to \mathbb{R}^N$ and defining

$$x_{t+1} = \theta\hat{\psi}(x_t, \dots, x_{t-p+1}) + \varepsilon_{t+1} = \theta\hat{\psi}(\hat{x}_t) + \varepsilon_{t+1} \in \mathbb{R}^m, \tag{3.1}$$

where $\theta \in \mathbb{R}^{m\times N}$. Please find more information on NAR models in [Bil13, LLC15, LL21].

With data matrices

$$\begin{aligned} \hat{\psi}(\mathbf{H}) &= \begin{bmatrix}\hat{\psi}(\hat{x}_{p-1}) & \dots & \hat{\psi}(\hat{x}_{T-1})\end{bmatrix} \in \mathbb{R}^{N\times T-p+1} \\ \mathbf{X}' &= \begin{bmatrix}x_p & \dots & x_T\end{bmatrix} \in \mathbb{R}^{m\times T-p+1} \end{aligned} \tag{3.2}$$

the typical least squares way to estimate model parameters is by solving

$$A_{\text{NAR}} = \underset{A\in\mathbb{R}^{m\times N}}{\arg\min} \|\mathbf{X}' - A\hat{\psi}(\mathbf{H})\|_F. \tag{3.3}$$

As in SINDy, let us add a sparsity constraint to the model to obtain the dominant basis functions - which could hint at characteristics of the dynamics - and solve

$$A_{\text{SINAR}} = \underset{A\in\mathbb{R}^{m\times N}}{\arg\min} \|\mathbf{X}' - A\hat{\psi}(\mathbf{H})\|_F + c\|A\|_1. \tag{3.4}$$

Note that the minimization of Eq. (3.4) is equivalent to SINDy with memory terms to which the basis functions are applied (more details on this relation will be shown later on). We therefore denote the process of solving Eq. (3.4) by *Sparse Identification of Nonlinear Autoregressive Models* (SINAR) [WKS21]. The name is chosen explicitly to highlight the close connections to SINDy on the one hand and AR models on the other hand (it must be said that there seems to exist a competitive spirit among researchers in the dynamical systems field to showcase creativity with method names, as also in SINDy, MANDy [GKES19] or VAMP [MPWN18]).

A similar method to SINAR was previously introduced in [BBP$^+$16] in which SINDy is applied to nonlinear transformations of the Singular Value Decomposition (SVD) of the Hankel matrix of a scalar-valued observable ϕ. This procedure was called Hankel-alternative View of Koopman (HAVOK). Note that this procedure is in line with Theorem 1.8 about filtered delay embeddings which says that applying a linear mapping to the delay-coordinate map of a scalar-valued observable gives an immersion of the original dynamics under certain conditions. The case that the linear mapping represents finding the SVD of the Hankel matrix – as in HAVOK – is actually already referred to in the publication by Sauer et. al. in 1991 [SYC91], page 21. SINAR is a generalization of HAVOK.

Note that while the formulation of NAR models in Eq. (3.1) is very general, a typical structure is

$$x_{t+1} = \theta_1 \psi(x_t) + \cdots + \theta_p \psi(x_{t-p+1}) + \varepsilon_{t+1} \in \mathbb{R}^m, \tag{3.5}$$

with $\psi : \mathbb{R}^m \to \mathbb{R}^N$, $\theta_i \in \mathbb{R}^{m\times N}$. Remember that the term "nonlinear autoregressive model" was previously introduced in this thesis in Section 1.2.1, Eq. (1.52). There an NAR model emerged naturally from the Mori–Zwanzig formalism by choosing as projection the orthogonal projection onto the span of m-dimensional basis functions $\psi_1, \ldots, \psi_L \in \mathcal{H}_\phi$. It was shown later on in Section 1.2.1 how to replace these by scalar-valued basis functions and obtain an equivalent model (Eq. (1.57)) with the additive structure of Eq. (3.5). For this special case, SINAR can be formulated accordingly by denoting

$$\hat{\psi}(\mathbf{H}) = \begin{bmatrix} \psi(x_{p-1}) & \ldots & \psi(x_{T-1}) \\ \vdots & & \vdots \\ \psi(x_0) & \ldots & \psi(x_{T-p+1}) \end{bmatrix} \in \mathbb{R}^{pN\times T-p+1} \text{ and } A = [\theta_1, \ldots, \theta_p] \in \mathbb{R}^{m\times pL}. \tag{3.6}$$

Therefore, while the MZ formalism explains how to construct dynamics of observables of a dynamical system, with SINAR we can estimate a sparse approximation of the coefficients of the ensuing MZ equation.

3.1.1 Relating SINAR to Mori–Zwanzig

One might notice a slight inconsistency between MZ and SINAR at this point: in MZ, each coefficient, h_k, of the NAR model Eq. (3.5) is computed separately, i.e., without respect to the others by calculating $\langle \psi, \psi \rangle^{-1} \langle \psi, \rho^k \circ F \rangle$. In SINAR, all coefficients are computed at once. In fact, the results of the two computations are generally not equal to each other. From this perspective, MZ merely gives the structure of the dynamics whose coefficients are then estimated with SINAR. This is acknowledged in the recent publications [LL21] and [LTLA21]. In the latter an iterative scheme to estimate the h_k from data is presented. It also references [LL21] in which, similarly to but more complex than SINAR, a NARMAX model is constructed on the basis of a *delay-embedded MZ*: the authors assume invertible dynamics and use augmented basis functions given by

$$\hat{\psi} = [\psi^T, \mathcal{K}^{-1} \circ \psi^T, \dots, \mathcal{K}^{-p+1} \circ \psi^T]^T \text{ so that } \hat{\psi}(x_t) = [\psi(x_t)^T, \psi(x_{t-1})^T, \dots, \psi(x_{t-p+1})^T]^T \tag{3.7}$$

on which they define the orthogonal projection

$$(\hat{P}g)(x) = \hat{\psi}(x) \langle \hat{\psi}, \hat{\psi} \rangle^{-1} \langle \hat{\phi}, g \rangle \tag{3.8}$$

to derive the Mori–Zwanzig equation Eq. (1.41). For the MZ equation, this means that the optimal prediction term is given by

$$\hat{P}(\phi \circ F)(x_{t-1}) = \hat{\psi}(x_{t-1}) \langle \hat{\psi}, \hat{\psi} \rangle^{-1} \langle \hat{\psi}, \phi \circ F \rangle = \hat{\psi}(x_{t-1}) \hat{h}_0. \tag{3.9}$$

Remembering that the basis functions ψ_i that compose ψ lie in $\mathcal{H}_\phi$, we can use that they map from $\mathbb{X}$ to $\mathbb{R}^m$ and apply the inverse of the Koopman operator to them.

The authors of [LL21] use Wiener filters which can be seen as a generalization of the NAR model (3.5) and line out arguments why delay-embedded MZ is a special case of MZ. They further mention orthogonality properties between observed and unresolved terms in delay-embedded MZ which are advantageous over those of the traditional MZ. We will now make the relation between SINAR and delay-embedded MZ more precise and show that the SINAR solution without sparsity constraint converges to the first coefficient $\hat{h}_0 = \langle \hat{\psi}, \hat{\psi} \rangle^{-1} \langle \hat{\psi}, \phi \circ F \rangle$ of the delay-embedded MZ equation if the system is invertible and ergodic (ergodicity denotes a usually very desirable property of dynamics [CFS82]). Recall the reformulation of MZ from matrix-valued basis functions and vector-valued coefficients to vector-valued basis functions and matrix-valued coefficients from Chapter 1. We will now formulate a theorem that establishes an equivalence between SINAR and MZ but uses the former formulation. It can thus be directly related to the latter formulation used in SINAR.

Theorem 3.1. Let $F : \mathbb{X} \to \mathbb{X}$, $\phi : \mathbb{X} \to \mathbb{R}^m$ and define $x_t = \phi(X_{t-1})$ for all $t > 0$. Let $\psi_1, \dots, \psi_L \in \mathcal{H}_\phi$, let $\psi : \mathbb{X} \to \mathbb{R}^{m\times L} = [\psi_1, \dots, \psi_L]$ be a bounded function and let $\mathcal{K}$ be the Koopman operator of ergodic and invertible dynamics F. Define $\hat{\psi} = [\psi, \mathcal{K}^{-1} \circ \psi, \dots, \mathcal{K}^{-p+1} \circ \psi] : \mathbb{X} \to \mathbb{R}^{m\times pL}$. Let a scalar product $\langle \cdot, \cdot \rangle$ be defined as in Eq. (1.36) and $\| \cdot \|_2$ be the Euclidean norm. Then for a time series $x_0, \dots, x_T$, it holds that

$$\lim_{T\to\infty} \operatorname*{arg\,min}_{\hat{A}\in\mathbb{R}^{pL\times 1}} \frac{1}{T} \sum_{t=p}^{T} \|x_t - [\psi(x_{t-1}), \dots, \psi(x_{t-p})]\hat{A}\|_2^2 = \langle \hat{\psi}, \hat{\psi}\rangle^{-1} \langle \hat{\psi}, \phi \circ F\rangle. \tag{3.10}$$

Proof. As a preparatory step, let data $x_1, \dots, x_T$ come from an ergodic system $x_t = f(x_{t-1}) \in \mathbb{X}$ with stationary distribution μ and let $\psi : \mathbb{X} \to \mathbb{R}^{N\times L}$ be a bounded function. Consider the term

$$\frac{1}{T} \sum_{t=1}^{T} \|x_{t+1} - \psi(x_t)A\|_2^2, \quad A \in \mathbb{R}^L. \tag{3.11}$$

By the Birkhoff Ergodic Theorem [CFS82], it holds

$$\begin{aligned} \lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} \|x_{t+1} - \psi(x_t)A\|_2^2 &= \int_{\mathbb{X}} \|f(x) - \psi(x)A\|_2^2 \mathrm{d}\mu(x) \\ &= \int_{\mathbb{X}} (f(x) - \psi(x)A)^T (f(x) - \psi(x)A) \mathrm{d}\mu(x). \end{aligned} \tag{3.12}$$

It is well known that this is minimized across all choices for A by choosing $A = \langle \psi, \psi \rangle^{-1} \langle \psi, f \rangle$ so that $\psi(x)A$ is the orthogonal projection of $f(x)$ onto the span of the basis ψ.

Now, let F, ϕ and ψ be given as in the theorem. For a given matrix $\hat{A}$, the minimization function on the left-hand side of Eq. (3.10) then converges to

$$\begin{aligned} &\lim_{T\to\infty} \frac{1}{T} \sum_{t=p}^{T} \|x_t - [\psi(x_{t-1}), \dots, \psi(x_{t-p})]\hat{A}\|_2^2 \\ &= \lim_{T\to\infty} \frac{1}{T} \sum_{t=p}^{T} \|\phi(F(X_{t-1})) - \hat{\psi}(X_{t-1})\hat{A}\|_2^2 \\ &= \int_{\mathbb{X}} \|\phi(F(X_{t-1})) - \hat{\psi}(X_{t-1})\hat{A}\|_2^2 \mathrm{d}\mu(X). \end{aligned} \tag{3.13}$$

Analogously to above, this is minimized if we choose $\hat{A} = \langle \hat{\psi}, \hat{\psi} \rangle^{-1} \langle \hat{\psi}, \phi \circ F \rangle$. □

Naturally the question arises, what the influence of the terms $\hat{\psi}(x_{t-k})$ is for $k = 2, \dots, p$. In [LL21], it is shown that for invertible ergodic dynamical systems and using the projection from Eq. (3.8) with an infinite memory depth, these terms vanish and what remains is the same structure as in Eq. (1.44) but with favourable orthogonality properties. It is at this point unclear what the influence of the terms $\hat{\psi}(x_{t-k})$ is if the chosen memory depth is finite but it can be suspected that it is negligible for sufficiently high memory depth. Similar observations are also made in [GGH21].

The relation of the left-hand side of Eq. (3.10) to the SINAR problem is directly given by applying the straightforward reformulations detailed at the end of Chapter 1. With this we have derived consistency between the data-driven method SINAR (without sparsity constraint) and the theoretical basis of MZ.

3.2 Connection Between the Koopman-based Methods

In the previous chapter, we have seen several data-driven methods which aim to estimate the dynamical equations for the evolution of an observable ϕ over time: DMD, EDMD, SINDy, AR and NAR (with possible, so far partly mentioned, extensions Hankel-DMD, Tensor-based DMD [KGPS16], Reactive SINDy [HFN19], ARX, ARMA, GARCH, ARIMA or NARMAX [LL21]). Although neither their names suggest it nor is it frequently emphasised in the majority of the literature, all of the presented methods are intimately connected and solve variants of the same problem: one seeks a mapping between ϕ and $\phi \circ F$. Such a mapping exists and it is given by the Koopman operator $\mathcal{K}$. On the downside, $\mathcal{K}$ is infinite-dimensional. Therefore, one transforms points of the form $x = \phi(X)$ into a new space by the use of basis functions ψ and hopes that there exists a finite-dimensional linear mapping which relates points of the form $\psi(x_t)$ to points of the form x_{t+1} (see Figure 3.1). This linear mapping is chosen to be optimal with respect to the Euclidean distance of points in the data at hand. Since all these methods are built around solving a least squares problem, Section 2.3 could have been called *Least squares based methods*. However, the name is meant to explicitly emphasise the concept of transforming into a new space with the use of basis functions with the aim of approximating the Koopman operator.

In its purest form, this is done in SINDy, although with an additional sparsity constraint. As the following theorem shows, by a specific choice of the basis functions, SINDy can be made equivalent to DMD, AR models and SINAR. The condition on the dynamics F is that they are invertible. This is naturally the case for diffeomorphisms as in Takens' Theorem. Intuitively, it simply means that for each element of the state space there is exactly one that is mapped onto it by F. In particular, this avoids two branches of the dynamics merging together so that each initial state induces a unique trajectory (although trajectories can approach each other arbitrarily closely). We formalize the relations in the following theorem.

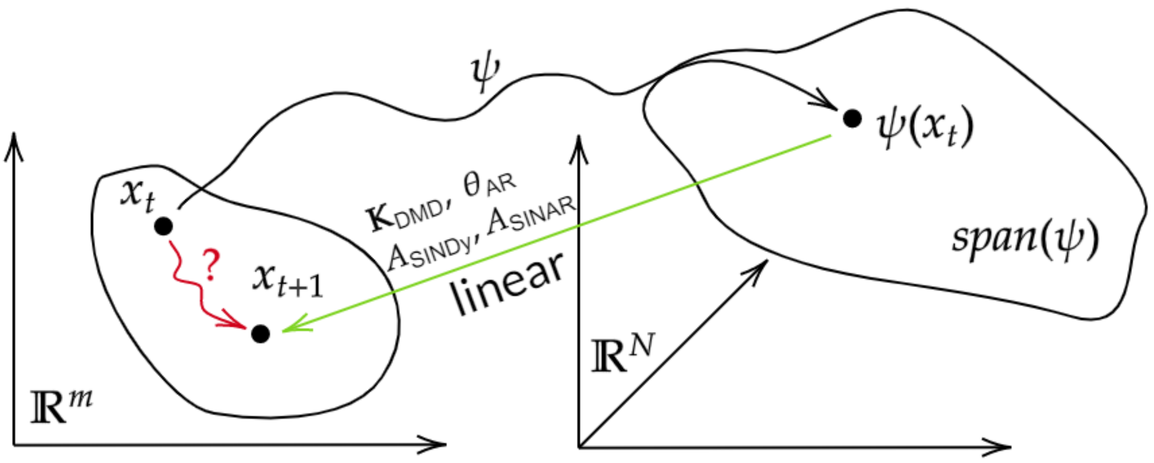


Figure 3.1: Illustration of general approach of the Koopman-based methods: in order to approximate the typically complex direct mapping from x_t to x_{t+1} one transforms points into the space *span*$\{\psi\}$ and determines a *linear* mapping to points in the original space.

Theorem 3.2. Let $F : \mathbb{X} \to \mathbb{X}$ denote invertible dynamics and let $\mathbb{X} \to \mathbb{R}^m$ and denote $x_t = \phi(X_t)$. Let $\mathcal{K}$ be the Koopman operator associated to F. Then a minimizer of the SINDy problem Eq. (2.41) is a minimizer of

(a) Eq. (2.29) (DMD) if $\psi(X) = \phi(X)$ and $c = 0$

(b) Eq. (2.52) (AR) if $\psi(X) = \Phi_{\phi,F,p}(X)$, $c = 0$ and omitting the first $p - 1$ columns

(c) Eq. (3.4) (SINAR) if $\psi(X) = \hat{\psi}(\Phi_{\phi,F,p}(X))$ and omitting the first $p - 1$ columns.

Proof. Firstly, while ψ was applied to observed states x in the introduction of the methods, applying it to full states X is no contradiction by definition of the basis functions as lying in $\mathcal{H}_\phi$. Please refer to the remark on page 59.

(a): Eq. (2.41) with $c = 0$ and basis $\psi(X) = \phi(X)$ reads

$$\underset{A \in \mathbb{R}^{m \times N}}{\arg\min} \|\mathbf{X}' - A\mathbf{X}\|_F$$

which is the problem in Eq. (2.29).

(b): With basis $\psi(X) = \Phi_{\phi,F,p}(X)$, $c = 0$ and omitting the first $p - 1$ columns, the

Name	Minimization problem	Model
DMD	$\mathbf{K}_{\text{DMD}} = \underset{\mathbf{K}\in\mathbb{R}^{m\times m}}{\arg\min} \lVert \mathbf{X}' - \mathbf{K}\mathbf{X} \rVert_F,$	$x_{t+1} = \mathbf{K}_{\text{DMD}} x_t,$
EDMD	$\mathbf{K}_{\text{EDMD}} = \underset{\mathbf{K}\in\mathbb{R}^{N\times N}}{\arg\min} \lVert \psi(\mathbf{X}') - \mathbf{K}\psi(\mathbf{X}) \rVert_F,$	$\psi(x_{t+1}) = \mathbf{K}_{\text{EDMD}} \psi(x_t),$
SINDy	$A_{\text{SINDy}} = \underset{A\in\mathbb{R}^{m\times N}}{\arg\min} \lVert \mathbf{X}' - A\psi(\mathbf{X}) \rVert_F + c\lVert A \rVert_1$	$x_{t+1} = A_{\text{SINDy}} \psi(x_t)$
AR	$\theta_{\text{AR}} = \underset{\theta\in\mathbb{R}^{m\times pm}}{\arg\min} \lVert \mathbf{X}' - \theta\mathbf{H} \rVert_F,$	$x_{t+1} = \theta_{\text{AR}} \hat{x}_t$
SINAR	$A_{\text{SINAR}} = \underset{A\in\mathbb{R}^{m\times N}}{\arg\min} \lVert \mathbf{X}' - A\hat{\psi}(\mathbf{H}) \rVert_F + c\lVert A \rVert_1,$	$x_{t+1} = A_{\text{SINAR}} \hat{\psi}(\hat{x}_t)$

Table 3.1: Summary of the Koopman-based methods

minimization problem in Eq. (2.41) reads

$$\begin{aligned}
&\underset{A\in\mathbb{R}^{m\times pm}}{\arg\min} \lVert \begin{bmatrix} x_p & \dots & x_T \end{bmatrix} - A \begin{bmatrix} \Phi_{\phi,F,p}(X_{p-1}) & \dots & \Phi_{\phi,F,p}(X_{T-1}) \end{bmatrix} \rVert_F \\
= &\underset{A\in\mathbb{R}^{m\times pm}}{\arg\min} \lVert \begin{bmatrix} x_p & \dots & x_T \end{bmatrix} - A \begin{bmatrix} x_{p-1} & \dots & x_{T-1} \\ \vdots & & \vdots \\ x_0 & \dots & x_{T-p} \end{bmatrix} \rVert_F
\end{aligned}$$

which is the problem in Eq. (2.52).

(c): With basis $\hat{\psi}(X) = \psi(\Phi_{\phi,F,p}(X))$ and omitting the first $p-1$ columns, the minimization problem in Eq. (2.41) reads

$$\begin{aligned}
&\underset{A\in\mathbb{R}^{m\times N}}{\arg\min} \lVert \begin{bmatrix} x_p & \dots & x_T \end{bmatrix} - A \begin{bmatrix} \hat{\psi}(\Phi_{\phi,F,p}(X_{p-1}) & \dots & \Phi_{\phi,F,p}(X_{T-1}) \end{bmatrix} \rVert_F + c\lVert A \rVert_1 \\
= &\underset{A\in\mathbb{R}^{m\times N}}{\arg\min} \lVert \begin{bmatrix} x_p & \dots & x_T \end{bmatrix} - A \begin{bmatrix} \hat{\psi}(\hat{x}_{p-1}) & \dots & \hat{\psi}(\hat{x}_{T-1}) \end{bmatrix} \rVert_F + c\lVert A \rVert_1
\end{aligned}$$

which is the problem in Eq. (3.4). □

In Table 3.1 the minimization problems and ensuing forward models of the five methods presented in this section are summarized. An illustration of the connections between the methods can be found in Figure 3.2.

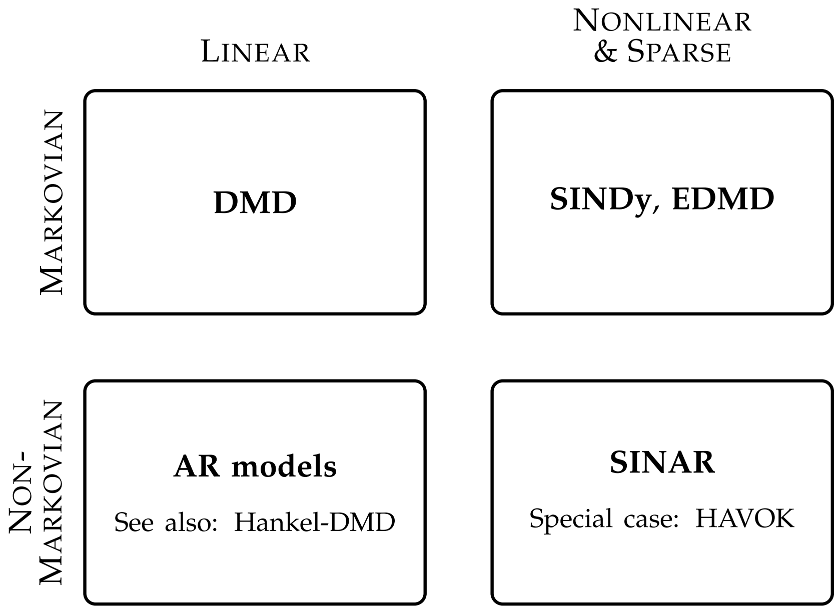


Figure 3.2: Illustration of relation between Koopman-based methods as explained in Chapter 2.

3.3 Numerical Experiments

We will now apply several of the so far presented methods to two different dynamical systems to see how they compare in different scenarios and for different purposes. In each example, we will use different sets of coordinates as the observables and investigate how the performances of the individual methods vary dependent on the observable. Further, we will vary the memory depth for the memory-dependent methods. The methods we will apply in the examples are Kernel regression, RNNs and LSTMs, DMD, SINDy and AR models and SINAR.

We will compare them based on their forecasting capacity using (1) the error between predicted and true future values and (2), for the first example, the sets of points in the predicted long-term trajectory and the true trajectory. For the former, we consider the *average Euclidean k-step forecasting error*

$$\mathcal{E}_k(\mathcal{W}) := \frac{1}{L}\sum_{i=1}^{L} \|x_{t_i} - \tilde{x}_{t_i}\|_2 \tag{3.14}$$

where time points $t_1, \ldots, t_L$ are selected for comparison and $\tilde{x}_{t_i}$ was predicted by a model $\mathcal{W}$ with a trajectory starting at x_{t_i-k}.

For the latter, we use the *Hausdorff distance* which measures the maximal minimal distance between non-empty compact sets. It is defined in the following way: from two trajectories $\mathbf{X} = [x_0, \ldots, x_T]$ and $\mathbf{Y} = [y_0, \ldots, y_T]$, we construct the delay embeddings with embedding dimension p as

$$\Phi_p(\mathbf{X}) = \left\{ \begin{bmatrix} x_{p-1} \\ \vdots \\ x_0 \end{bmatrix}, \begin{bmatrix} x_p \\ \vdots \\ x_1 \end{bmatrix}, \dots \right\}, \quad \Phi_p(\mathbf{Y}) = \left\{ \begin{bmatrix} y_{p-1} \\ \vdots \\ y_0 \end{bmatrix}, \begin{bmatrix} y_p \\ \vdots \\ y_1 \end{bmatrix}, \dots \right\}. \tag{3.15}$$

We then calculate their Hausdorff distance as

$$\mathcal{H}_p(\mathbf{X},\mathbf{Y}) := \max\Big(\max_{\hat{x}\in\Phi_p(\mathbf{X})} \min_{\hat{y}\in\Phi_p(\mathbf{Y})} \|\hat{x}-\hat{y}\|_2, \max_{\hat{y}\in\Phi_p(\mathbf{Y})} \min_{\hat{x}\in\Phi_p(\mathbf{X})} \|\hat{x}-\hat{y}\|_2 \Big). \tag{3.16}$$

Comparing the images of the delay-coordinate map between two trajectories gives additional insight into the underlying dynamics of the trajectories: consider two different one-dimensional trajectories which move inside the interval $[-1,1]$ but with a very different distribution of visited states. If both cover the entire interval their Hausdorff distance will be 0 although the trajectories are different. By using a delay embedding, we compare whether pairs of subsequent points in both trajectories match. If the dynamics underlying each trajectory are very different then a point x is followed by a different point in both trajectories. Therefore, the Hausdorff distance of the delay embeddings should uncover these differences.

3.3.1 An Extended Hénon System

We consider the Hénon system [Hé76], a two-dimensional chaotic dynamical system. It is given by the equations

$$\begin{aligned} x_{t+1} &= 1 - a x_t^2 + y_t \\ y_{t+1} &= b x_t, \end{aligned} \tag{3.17}$$

This system can serve as a simple example to see how memory of one variable can substitute missing information of other variables. Suppose we can only observe the x-coordinate, then

$$x_{t+1} = 1 - a x_t^2 + b x_{t-1}. \tag{3.18}$$

Since y is merely a delayed and scaled copy of x, one can straightforwardly replace y_t in the equation for x_{t+1}. However, to make matters less trivial, let us now consider an extended version of the Hénon system, defined by the author in [WKS21]

$$\begin{aligned} x_{t+1} &= 1 - a x_t^2 + y_t \\ y_{t+1} &= b x_t + c y_t. \end{aligned} \tag{3.19}$$

Then we can see,

$$\begin{aligned} x_{t+1} &= 1 - ax_t^2 + bx_{t-1} + cy_{t-1} \\ &= 1 - ax_t^2 + bx_{t-1} + cbx_{t-2} + c^2 y_{t-2} \\ &= 1 - ax_t^2 + bx_{t-1} + cbx_{t-2} + c^2 b_{t-3} + c^3 y_{t-3} \\ &= 1 - ax_t^2 + \sum_{j=1}^{t} c^{j-1} bx_{t-j} + c^{t+1} y_0. \end{aligned} \tag{3.20}$$

We have therefore derived a Mori–Zwanzig-like formulation for the dynamics of x: $1 - ax_t^2$ is the Markovian term, the sum $\sum_{j=1}^{t} c^{j-1} bx_{t-j}$ contains the memory terms of x and the term $c^{t+1}y_0$ is unobservable and denotes the "noise" in the MZ equation.

For certain parameters, both the classical and the extended Hénon systems converge to an *attractor*, which denotes a set which is invariant under the dynamics and to which nearby trajectories must converge. We consider the extended Hénon system with parameters $a = 1.3, b = 0.3, c = 0.3$. See Figure 3.3 for images of both the classical and the extended Hénon system. Note that for $c < 1$, memory terms decay exponentially quickly towards 0 with increasing memory.

We will aim to predict subsequent values of the observables given by (1) the full system state and (2) only the x-coordinate. We will use different amounts of data to determine parameters of the methods and compare them both in short-term prediction using the average Euclidean prediction error and their ability to reconstruct the attractor of the extended Hénon system in the long-term with the Hausdorff distance.

Let us turn towards the specifications of the used methods to reconstruct the dynamics of the extended Hénon system from data: for Kernel regression, we use a Gaussian Kernel and a bandwidth $h = 0.01$. For SINDy, we use as basis function

$$\psi_{\text{SINDy}}(x) = [1, x, x^2, x^3, \underbrace{y, xy, y^2, y^3}_{\text{in case (1)}}]. \tag{3.21}$$

For SINAR we use $\psi_{\text{SINAR}}(\hat{x}_{t-1}) = [\psi_{\text{SINDy}}(x_{t-1}), \ldots, \psi_{\text{SINDy}}(x_{t-p})]$. We omit the sparsity constraints by setting $c = 0$. One must note that this choice is well suited for the given system since it contains the right basis functions already and not many more. As memory depths we use $p = 1, \ldots, 10$. For the RNN, we use a hidden state net without control input and consisting of a layer with 100 neurons, a Rectified Linear Unit activation function and another layer with the number of neurons equal to the output dimension, i.e., 2 for both coordinates and 1 for the x coordinate. For the LSTM, each sub-network (the different parts of an LSTM) consists of a layer with 100 neurons and a regression layer. For both neural networks we use an implementation provided by Matlab.

The results concerning the short-term forecasting are shown in Figure 3.4 and the results regarding the long-term forecast in Figure 3.5.

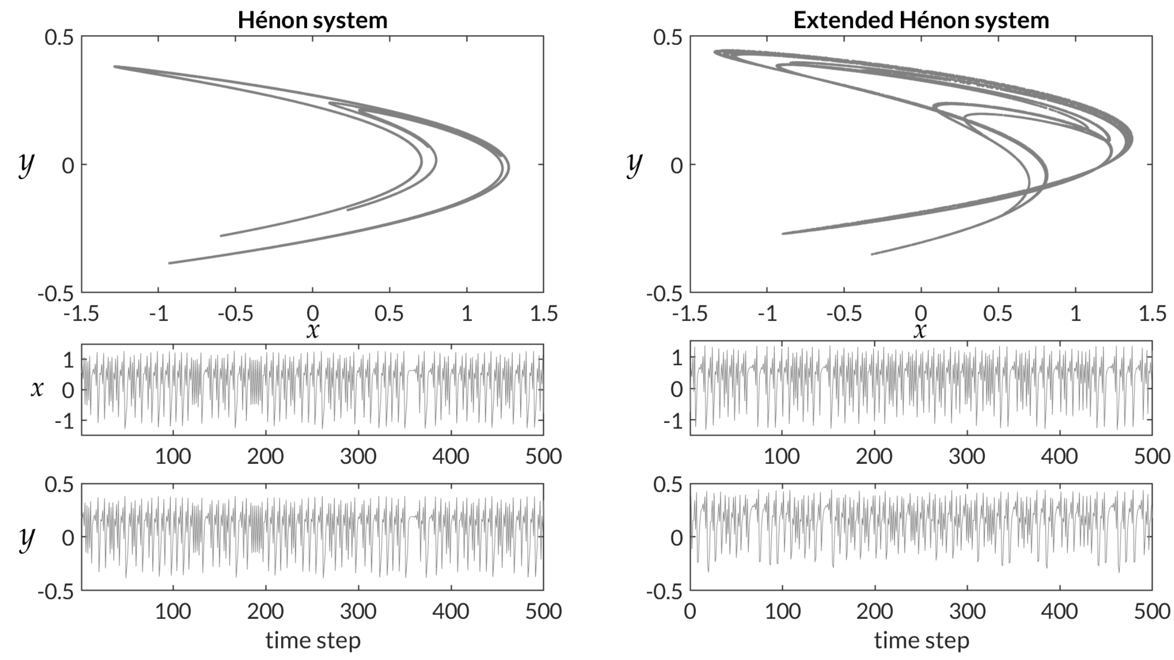


Figure 3.3: Left: Hénon system with $a = 1.4$, $b = 0.3$. Right: extended Hénon system with $a = 1.3$, $b = 0.3$, $c = 0.3$. Trajectories in the two-dimensional plot consist of 5000 points each.

Results

DMD and AR We can see that with information about the full system state and for only the x-coordinate, DMD and AR models do not produce accurate predictions even for only one time-step ahead (Figure 3.4, left, **I–IV**), since they are linear models while the system is nonlinear to a degree that makes linear approximations immediately imprecise. For AR models, a higher memory depth at least gives better forecasts (see Figure 3.4, right, **I–IV**). Long-term predictions are very inaccurate, too, as shown in Figure 3.5, **VI** and **VIII**. The predictions of both models naturally converge to a fixed point since the eigenvalues of their system matrices, $\mathbf{K}_{\text{DMD}}$ for DMD and the companion matrix C_{AR} for AR, are smaller than 1 in absolute value.

SINDy and SINAR Introducing nonlinearity with SINDy and SINAR however yields much better accuracy. SINDy, equipped with the right basis functions is very precise when considering the full state both in the short- and in the long-term (see Figure 3.4, **I,II** and Figure 3.5, **V**, **VI**). With only the x-coordinate, however, the loss of information of the y-coordinate makes SINDy imprecise (see Figure 3.4, **III,IV** and Figure 3.5, **VII**, **VIII**). SINAR generates the best results for all settings. For little training data, the estimated model yields predictions with a high error but with even 50 data points, the forecasts become very precise. We can further see how an increase in memory depth improves the prediction accuracy for both short- and long-term forecasts (see Figure 3.4, right, **I–IV** and Figure 3.5, right, **V–VIII**). This is because the model coefficients can gradually approximate Eq. (3.20) increasingly well.

Kernel regression The accuracy of Kernel regression strongly depends on the amount of training data. This should not be surprising since its precision depends on the closeness of points in the training data to a point from which one wants to predict future states as shown in Section 2.2. If the attractor is filled more densely by the training data, the accuracy should thus increase. It further constructs a reasonable approximation of the attractor for the case $\phi(X) = x$ while seems to visit only a small set of points for the full-state observable.

Neural networks The RNN produces good accuracy for the full system with sufficient training data and manages to reconstruct the attractor well. However, similar to SINDy, for only the x-coordinate it cannot overcome the loss of information about the y-coordinate and the results are comparable to the one of SINDy. The LSTM, surprisingly, produces imprecise short-term forecasts which are worse than those of the RNN even with only the x-coordinate. It, however, manages to reconstruct the attractors in both settings well if sufficient data is provided.

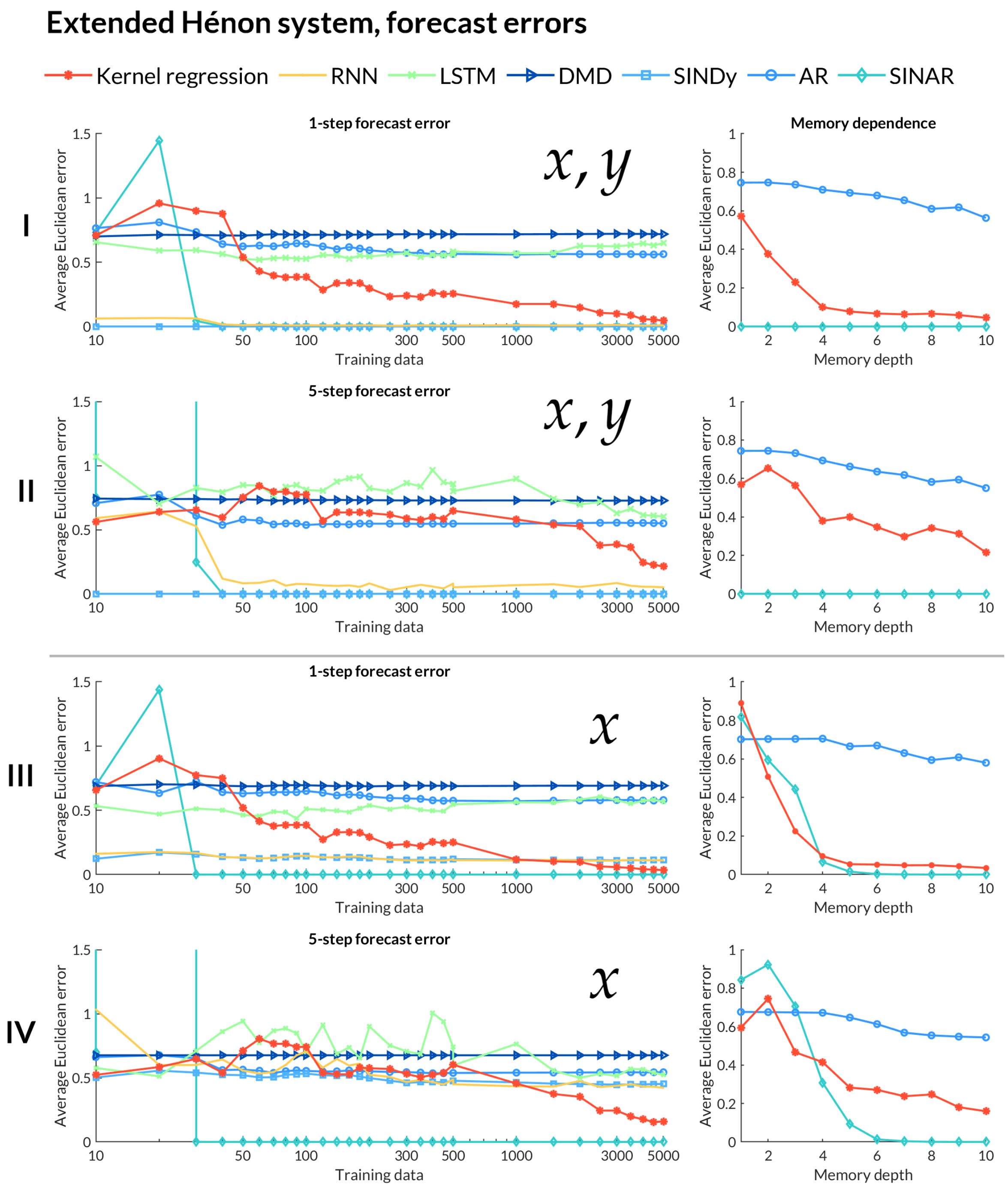


Figure 3.4: Performance results of different methods on the extended Hénon System. Left: 1- and 5-step forecasting error over number of training data points for both coordinates (**I** and **II**) and x-coordinate only (**III** and **IV**). Kernel regression, AR and SINAR with memory $p = 10$. Right: error over memory depth for 5000 training data points.

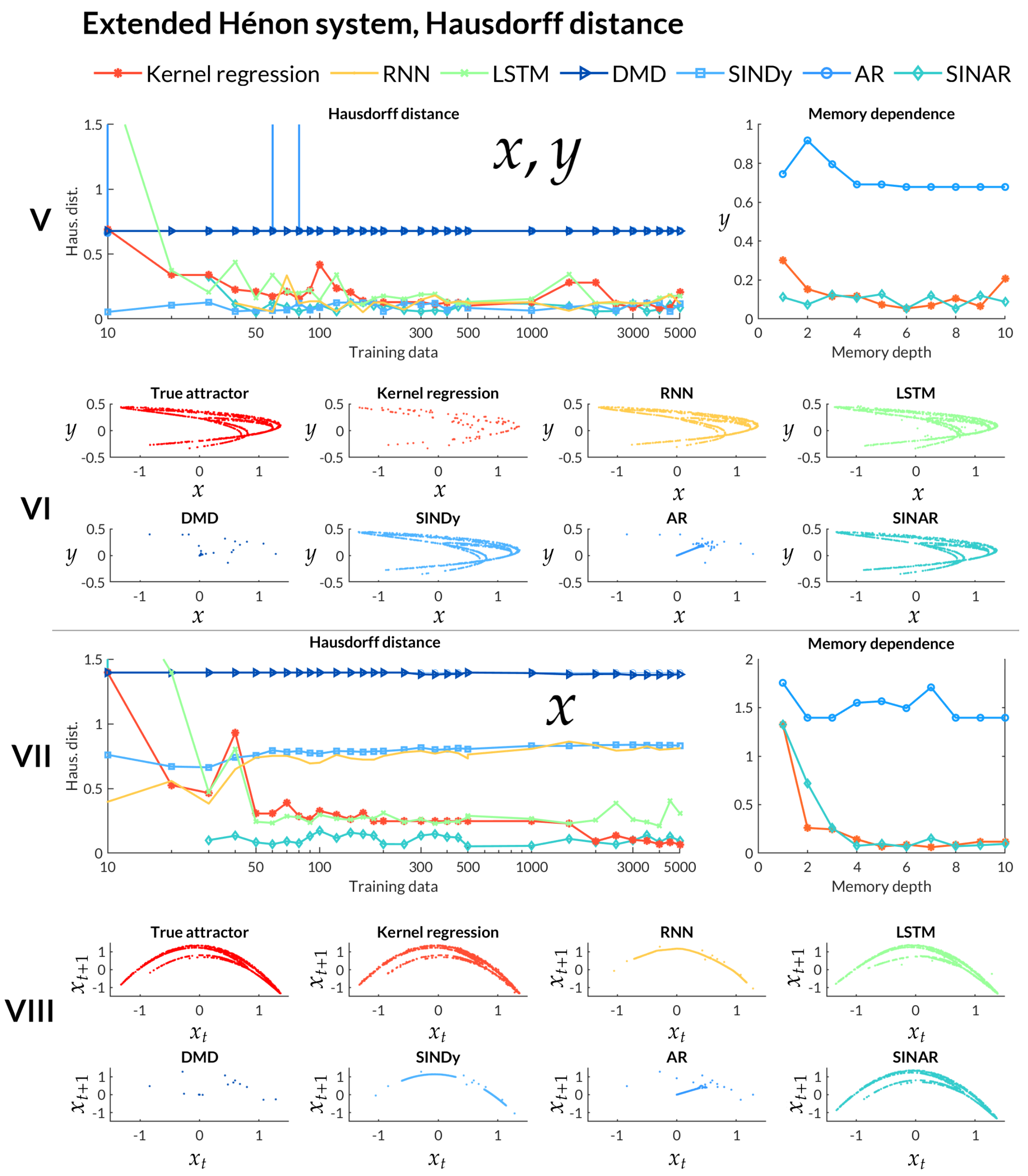


Figure 3.5: **V** and **VII**: Hausdorff distance (3.16) of attractors from long-term prediction over training data (left) and dependence on memory (right; for Kernel regression, AR and SINAR). $\mathcal{H}_1$ for both coordinates and $\mathcal{H}_2$ for only the x-coordinate. **VI** and **VIII**: predicted attractors with each method for both coordinates and x-coordinate only (delay embedding).

3.3.2 Simulation of a Gene Regulatory Network

Next, we will consider simulated dynamics of the concentrations of substances from a Gene Regulatory Network (GRN). The GRN describes the following processes: DNA from the nucleus of a cell is transcribed into messenger RNA (mRNA) which is then translated to a protein outside of the nucleus. The expressed proteins can influence – *regulate* – the transcription of the DNA of other proteins so that commonly the term GRN is used to describe the set of relations between genes of an organism. In summary, the reactions which we consider are:

$$\text{DNA} \overset{\text{trancribed to}}{\rightarrow} \text{mRNA} \overset{\text{translated to}}{\rightarrow} \text{Protein} \;\text{ and } \text{mRNA}_i \overset{\text{regulates}}{\rightsquigarrow} \text{DNA}_j.$$

Further, typically mRNA and proteins decay over time due to control mechanisms of the organism. We assume the concentration of DNA to stay constant since it generally does not decay by transcription.

We assume that transcription, translation, regulation and decay occur at constant rates. All reactions and their rates are given in Table 3.2. We assume three proteins, A, B, and C, and use the same rate coefficients as in [HFN19], except for the ones regarding regulation. In contrast to the procedure in [HFN19], we assume the ensuing dynamics of concentrations to be deterministic.

With the constant reaction rates and making convenient assumptions on the mixing of the substances, we can derive the expression for the evolution of the concentrations by the nonlinear Ordinary Differential Equations in Eq. (3.22).

Reactant		Product	Rate	Description
DNA_A	$\rightarrow$	mRNA_A	1.8	Transcription of DNA_A
mRNA_A	$\rightarrow$	A	2.1	Translation to protein A
mRNA_A	$\rightarrow$	$\emptyset$	1.3	Decay of mRNA_A
A	$\rightarrow$	$\emptyset$	1.5	Decay of protein A
DNA_B	$\rightarrow$	mRNA_B	2.2	Transcription of DNA_B
mRNA_B	$\rightarrow$	B	2.0	Translation to protein B
mRNA_B	$\rightarrow$	$\emptyset$	2.0	Decay of mRNA_B
B	$\rightarrow$	$\emptyset$	2.5	Decay of protein B
DNA_C	$\rightarrow$	mRNA_C	3.2	Transcription of DNA_C
mRNA_C	$\rightarrow$	C	3.0	Translation to protein C
mRNA_C	$\rightarrow$	$\emptyset$	2.3	Decay of mRNA_C
C	$\rightarrow$	$\emptyset$	2.5	Decay of protein C
$\text{mRNA}_A + B$	$\rightarrow$	$\emptyset$	1.0	Regulation of mRNA_A by B
$\text{mRNA}_B + C$	$\rightarrow$	$\emptyset$	2.0	Regulation of mRNA_B by C
$\text{mRNA}_C + A$	$\rightarrow$	$\emptyset$	1.0	Regulation of mRNA_C by A

Table 3.2: Rate equations for the reactions we assume in the GRN.

$$\begin{aligned}
\frac{d\,\text{mRNA}_A}{dt} &= 1.8\ \text{DNA}_A - 1.3\ \text{mRNA}_A - 1.0\ \text{mRNA}_A B \\
\frac{dA}{dt} &= 2.1\ \text{mRNA}_A - 1.5\ A \\
\frac{d\,\text{mRNA}_B}{dt} &= 2.2\ \text{DNA}_B - 2.0\ \text{mRNA}_B - 2.0\ \text{mRNA}_B C \\
\frac{dB}{dt} &= 2.0\ \text{mRNA}_B - 2.5\ B \\
\frac{d\,\text{mRNA}_C}{dt} &= 3.2\ \text{DNA}_C - 2.3\ \text{mRNA}_C - 1.0\ \text{mRNA}_C A \\
\frac{dC}{dt} &= 3.0\ \text{mRNA}_C - 2.5\ C \\
\frac{d\,\text{DNA}_A}{dt} &= \frac{d\,\text{DNA}_B}{dt} = \frac{d\,\text{DNA}_C}{dt} = 0.
\end{aligned} \tag{3.22}$$

We use various initial conditions which we design by

$$X_0 = [\underset{\text{DNA}_A}{1}, \underset{\text{DNA}_B}{1}, \underset{\text{DNA}_C}{1}, \underset{\text{mRNA}_A}{\epsilon_4}, \underset{\text{mRNA}_B}{0}, \underset{\text{mRNA}_C}{0}, \underset{A}{0}, \underset{B}{\epsilon_8}, \underset{C}{0}]^T \tag{3.23}$$

where $\epsilon_4 \sim \mathcal{N}(2, 0.09), \epsilon_8 \sim \mathcal{N}(3, 0.09)$. With this, we start with an initial condition and perturb entries corresponding to mRNA and protein concentration by a normally distributed number. The variance of 0.09 of the relative perturbation comes from a standard deviation of 0.3 that was used. For four different initial conditions, the ensuing trajectories are depicted in Figure 3.6.

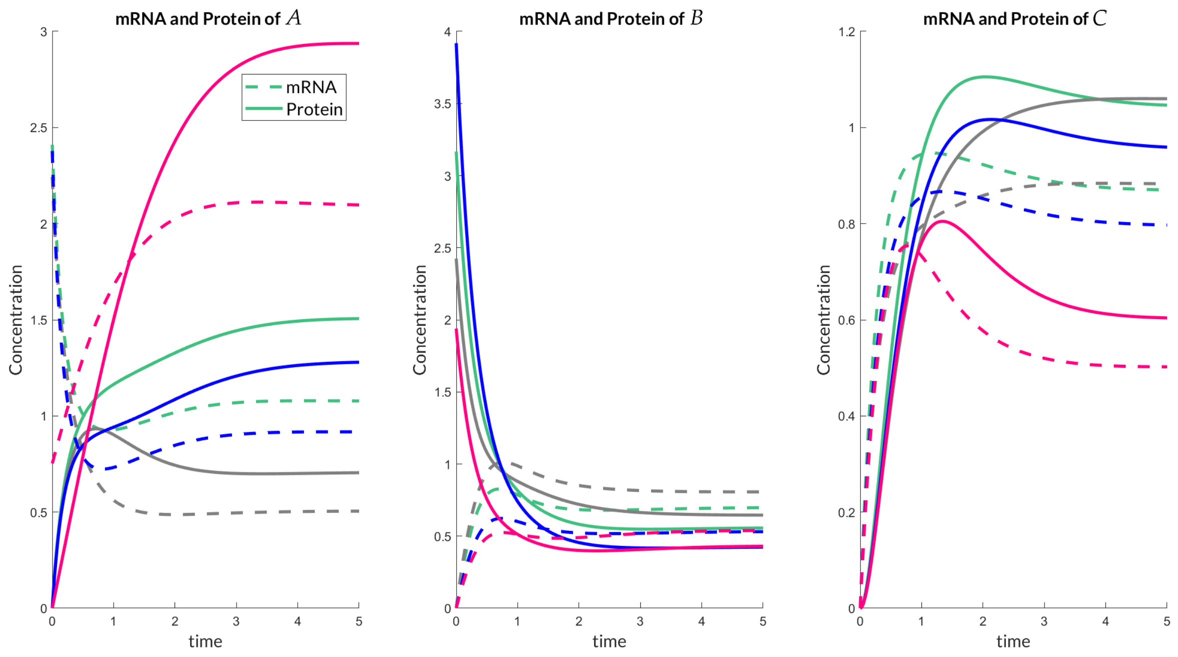


Figure 3.6: Evolution of concentrations of mRNA and protein concentrations for all three proteins for four different initial conditions. Matching colors denote the same initial conditions with straight lines denoting the protein and dashed lines denoting the mRNA concentration.

We then investigate the accuracy of the methods used in the previous example with varying number of trajectories used for parameter estimation while we start with one trajectory and subsequently add further ones. The data of the trajectories comes with a step size of 0.01 and each trajectory contains 501 time steps.

We use 20 trajectories for testing for which we make forecasts with each model starting at different points of the testing trajectories. We then evaluate the average Euclidean k-step forecast error for one time step and 200 time steps.

As in the example on the Hénon system, we use for Kernel regression a Gaussian Kernel and a bandwidth $h = 0.01$. For SINDy, we use as basis function ψ_{SINDy} all monomials of degree 1 and 2 and the constant 1 function. For SINAR the basis functions are given by $\psi_{\text{SINAR}}(\hat{x}_{t-1}) = [\psi_{\text{SINDy}}(x_{t-1})^T, \dots, \psi_{\text{SINDy}}(x_{t-p})^T]^T$. Again we omit the sparsity constraints in SINDy and SINAR by choosing $c = 0$. While previously the Koopman-based methods were introduced as estimating a model based on one trajectory., they can straightforwardly be modified for multiple trajectories $\mathbf{X}_1, \dots, \mathbf{X}_r$ by defining $\mathbf{X} = [\mathbf{X}_1, \dots, \mathbf{X}_r]$, $\mathbf{X}' = [\mathbf{X}'_1, \dots, \mathbf{X}'_r]$ and $\mathbf{H} = [\mathbf{H}_1, \dots, \mathbf{H}_r]$ with corresponding Hankel matrices $\mathbf{H}_1, \dots, \mathbf{H}_r$.

We consider two different observables: (1) DNA, mRNA and protein concentration of A and (2) protein concentrations of A, B and C.

Results

Since from neither observable we possess the full state information, it should not be surprising that in both settings memory is decisive in improving the prediction accuracy.

AR and SINAR We can see that AR and SINAR predictions – both with a memory of up to 4 time steps – give similarly good performances. With sufficient data, SINAR is slightly better, especially in the 1-step forecasts due to the nonlinearity of the system induced by the mRNA regulation. The system is, however, almost linear as indicated by the good accuracy AR yields for the 200-step forecast error. In case (1) with DNA, mRNA and protein A, SINAR overfits and identifies nonlinear terms which do not contribute decisively to the dynamics. Therefore, it often diverges in the long-term so that the 200-step forecast error in the first setting cannot be shown (Figure 3.7, **II**). For the second setting SINAR requires at least 10 training trajectories to find a non-diverging model (**IV**).

DMD and SINDy DMD and SINDy perform similarly well in both settings but their performances do not match the ones of their memory-using counterparts AR and SINAR. SINDy does not diverge, in contrast to SINAR with memory.

Kernel regression Kernel regression is not competitive with the Koopman-based methods. The reason for that, one could suspect, is that contrary to the Hénon system, the trajectories are not on an attractor of the system. All trajectories seem to converge to different fixed points (see Figure 3.6) so that the set of fixed points depending on the initial conditions might in fact assemble an attractor. However, at the start of the trajectories, this attractor is not reached, meaning that all trajectories come from the so-called *transient* phase of the system. Kernel regression requires that the domain of interest is densely filled with points so that a prediction from a point x is made on the basis of nearby points to x. In this example, the trajectories seem not to be sufficiently close to each other. Therefore, since Kernel regression does not learn governing equations as the Koopman-based methods do but works with weighted averages of the training data, it is ill-suited for this example.

Neural networks RNN and LSTM do not perform well in this example. Merely the RNN gives accuracy that is comparable to DMD and SINDy for the 1-step predictions. The accuracy of LSTM is worse than that of all methods for most amounts of training data. However, please note that this thesis does not strongly focus on neural networks and spending much more effort in determining the optimal network structure, one could be able to generate better accuracy. The point here is to show that neural networks do not immediately out-perform Kernel regression and

Koopman-based methods and also suffer from lack of information if certain variables are not observed.

Influence of perturbations of the data

Given the similar performances of AR and SINAR, let us investigate what happens if the data are perturbed by noise. To that end, we multiply each value in the training data by a factor given by $\epsilon_{ti} \sim \mathcal{N}(1, \sigma^2)$ with different values for σ. We use 20 training trajectories for training and compare the forecasting accuracy for AR(4), SINAR with memory 4 without sparsity constraint and Kernel regression with memory of 4. For comparison, we generate predictions of 50 time steps starting at 4 different starting points in each of the 20 testing trajectories. We can see (Figure 3.8) that the performances of all methods decrease with increasing noise. The AR and Kernel regression predictions, however, do not diverge as we can see in the non-diverging error graphs. For SINAR the predictions diverge regularly. Further, the higher the noise level, the earlier the divergence occurs. We therefore see that since the underlying system is almost linear, AR models can give comparably good performance as SINAR but are less prone to suffer from noise. This is in line with the message deduced from Theorem 2.4: we argued that the more different the chosen basis functions are in their magnitude, the less robust is the parameter estimation under perturbations. Note that the theorem is not directly applicable here since in SINAR we used the constant-1-function for which assumption (A1) from the theorem only holds for $\psi_{min} = 0$ and $\psi_{max} = \infty$, making the ensuing inequality ill-defined. Still, the general message is validated here since in SINAR we used linear and quadratic monomials of memory terms and in AR we only used linear memory terms. The Lipschitz constant ψ_{max} needed to bound the action of the quadratic monomials from above is higher than for linear terms only, making parameter estimation in SINAR less robust.

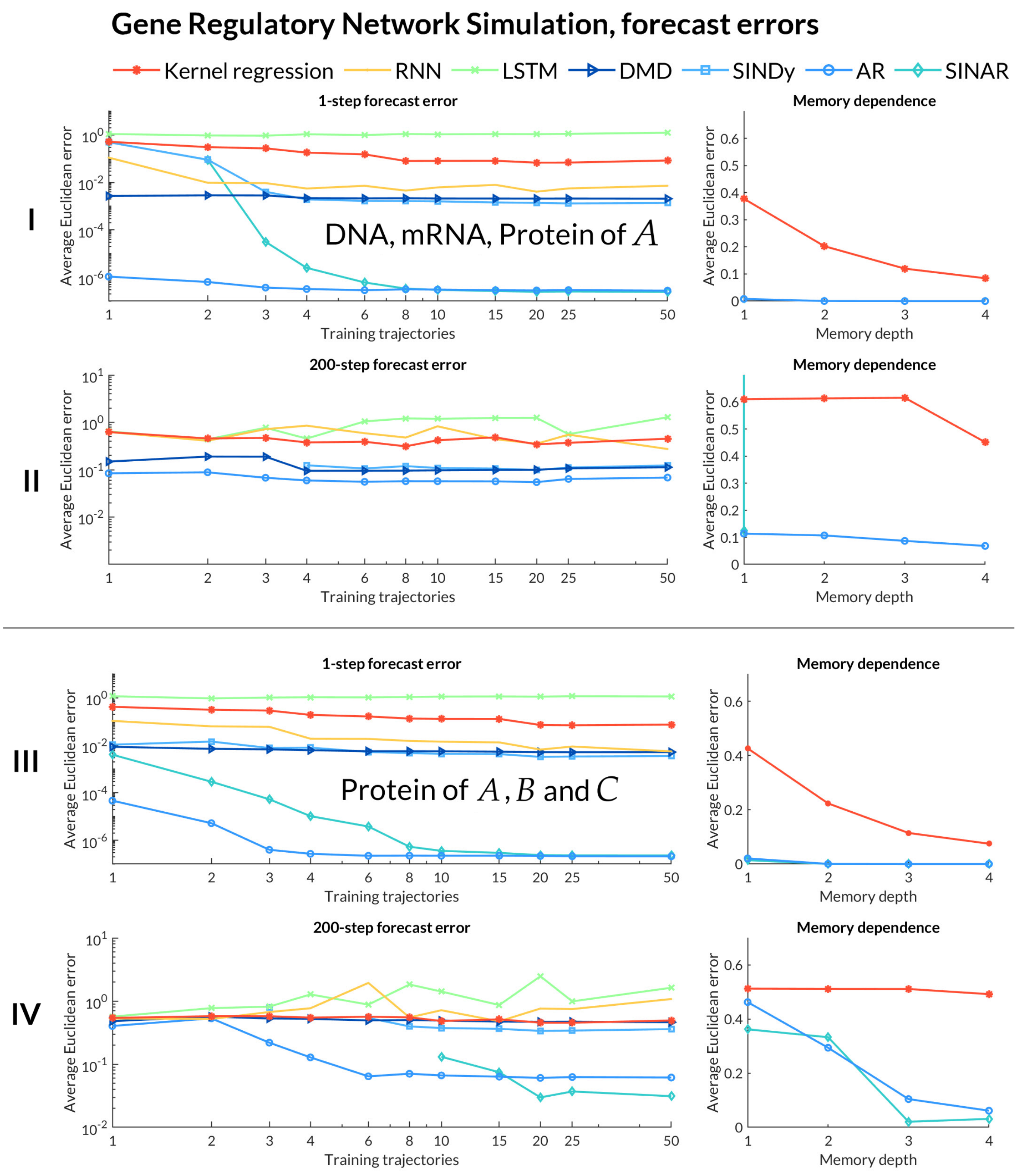


Figure 3.7: Forecasting accuracy of different methods for the GRN. 1- and 200-step (with step size 0.01) forecasting error over number of different trajectories in the training. Kernel regression, AR and SINAR use memory $p = 4$. Right: error over memory depth for 50 training trajectories. **I** and **II**: observed and predicted are the concentration of DNA (constantly 1), mRNA and Protein of A. **III** and **III**: observed and predicted are protein concentrations of A, B and C.

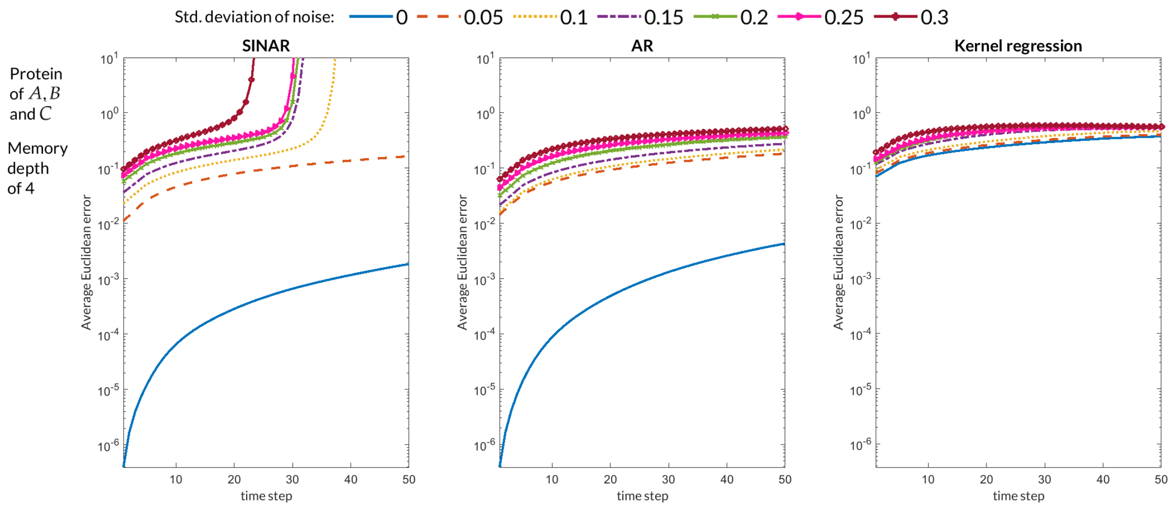


Figure 3.8: Forecasting error with SINAR, AR and Kernel regression with protein concentration of A, B and C observed and predicted. Training data (20 training trajectories) was contaminated with normally distributed relative noise with standard deviation between 0 and 0.3. We choose for the memory depth $p = 4$.

3.3.3 Summary of the Experimental Results

In this experimental section, we discussed two different scenarios in which we compared different dynamical modelling methods: (1) a chaotic attractor for which we considered the short-term prediction precision and the ability to reconstruct the shape of the attractor of the methods and (2) a non-chaotic almost linear system with many short trajectories from different initial conditions. We further used different subsets of the system variables as our observables.

It should have become clear that the various methods have varying applicability depending on the setting: memory generally needs to be incorporated if the full state information is not available. For nonlinear systems, it generally does not suffice to use linear models. However, one has to be cautious when using nonlinear methods such as SINDy and SINAR due to the danger of overfitting and unstable models. The neural network approach with an RNN and LSTM gave mediocre short-term predictions, possibly because they often require tremendous amounts of training data, but in the case of the extended Hénon system, gave good long-term predictions. Kernel regression was well-suited for the extended Hénon system since the training trajectory densely filled the attractor but ill-suited when training trajectories in the GRN example came from the transient phase of the system and were apart from each other.

We have therefore seen structurally different methods, which in part can be directly deduced from different modelling assumptions (Kernel regression from Takens, AR and SINAR from MZ), in action on two examples which came in a classical fashion of low-dimensional, deterministic equations. In the next chapter, we will consider the significantly different scenario of an *agent-based model*, which constitutes a high-dimensional, stochastic description of opinion changes in a population of human individuals. This will help to connect the research field of these agent-based models to the field of modelling memory-exhibiting dynamics.

Ensuing Research Questions

For further research it would be desirable if the equivalence between the Koopman-based methods was more strongly exploited so that results on one method could ideally be translated to other methods, letting several communities of research benefit. For example, the implications of the observation of the long-term behaviour of linear DMD models for the long-term behaviour of linear AR models were already mentioned. Additionally, Theorem 2.4, the perturbation theorem for SINDy, could be applied to the other methods to obtain bounds for the perturbation of the estimated coefficients under perturbation of the data.

Moreover, systematic comparisons between numerical methods could shed more light on the question for which type of problem which method is the most suited.

Uncovering Agent-based Dynamics with Memory

In the previous chapter, we have been introduced to multiple conceptually different methods for the modelling of dynamical systems and tested them on two low-dimensional examples. In this chapter, we will bring them into the context of a thematically very different research field: agent-based models (ABMs).

4.1 Introduction to Agent-based Models

In the field of modelling complex systems, agent-based models have seen a tremendous rise in prominence over the last several decades. This is especially true for systems which human behaviour is involved in, e.g., from the social sciences. While in problems from the natural sciences one can often formulate a set of equations based on a physical intuition about the problem, such an intuition often does not exist for systems outside of them. In agent-based modelling, instead of modelling relevant quantities directly, one rather, in absence of a physical intuition about which family of equations could capture the dynamics of those quantities well, models all elements which play a role individually. These elements are then called *agents* and are equipped with mathematically formulated *rules* which govern their behaviour. One then tries to observe patterns that emerge, i.e., statistics of the full population of agents. Remembering the Gene Regulatory Network example in Chapter 3, we directly modelled the concentrations of substances by imposing implicit modelling assumptions that lead to the formulation of an ODE. In contrast, an ABM would model the involved particles directly and simulate their individual interactions.

ABMs are often used to model real-world processes in which human behaviour is involved, such as in economics (see [Han17] for an overview), traffic modelling [HRR15] including evacuation [TCP06] and archaeology [CHZ+18]. The goal typically is to make predictions about how the behaviour of a society changes with respect to certain, naturally occurring or imposed, events. The interest usually does not lie in investigating the resulting behavioural changes of individual persons, the so-called *microscale*, but rather of the society overall, the so-called *macroscale*. The goal of many ABMs is hence to model the behaviour of synthetic populations in order to draw conclusions of the evolution of relevant statistics in the real world.

However, since ABMs demand the simulation of actions of all agents, they can yield high computational cost. It can therefore be worthwhile to use realisations of the ABM simply to find an equation-based model for the direct modelling of the evolution of the statistics over time and deploy this on the real-world problem, e.g., as similarly done in the context of pandemic modelling in [WCDC+21]. For an equation-based model of the macrostates, it has to be taken into account that since the macrostates emerge from statistics of the microscale, they can also be seen as *observables* in the sense of the previous three chapters. Therefore, it becomes immediately advisable that in order to model their evolution one should use memory terms. The main point of this chapter is to show that in order to model observations of ABMs and complex phenomena in human populations, one generally requires memory for the same reasons as introduced in Chapter 1.

4.1.1 Opinion Dynamics

A specific application area of ABMs is *social dynamics*. The research on social dynamics is the attempt to characterize patterns of social interaction and their implications. Of particular interest in the literature on this field has been *opinion dynamics*. It covers the investigation of how opinions spread across a population or how significant sudden events on the distribution of opinions in a population are. Over the last 15 years, there has been progress in this field on a mathematical basis, as can be found in, e.g., [CS73, Ban16, KLT07, CFL09, SLST17].

Many of these models serve to simulate events on the micro- but not the macroscale. When considering the macroscale, of particular interest in the literature are the opinion concentrations inside the population, as in opinion polls. While there exists literature on the modelling of opinions with memory on the microscale, e.g., [JSW18, CDFB18, BCK20], literature that indeed discusses modelling ideas for these opinion concentrations, such as in [NWWS21, WWS18, RHLD19, DBB+19], ignores the use of memory terms, with the exception of [Ban14]. However, the arguments in [Ban14] are of combinatorial nature and although a strong contribution to the topic, they give no practical way to model the evolution of the opinion concentrations. Hence, in this chapter we hope to fill this gap with some of the tools which have been introduced in this thesis so far on two new ABMs for opinion spreading. Some of the work on the first of the two ABMs in this chapter was done by the author of this thesis in [WKS21] together with co-authors. The rest, including the entire work on the second ABM, is previously unpublished.

4.2 Opinion Dynamics ABM I: Discrete Opinions

In the first ABM, we simulate N agents out of which each has exactly one out of m opinions at the time which it can change over discrete time steps. Agents are connected, we say they are *neighbours*, governed by a symmetric adjacency matrix $A \in \{0,1\}^{N\times N}$ which stays constant over time. An agent is always neighbouring to itself. The opinion changes happen upon a stochastic rule but will be influenced by the opinions of the neighbours of each agent. The higher the number of neighbours of an agent with a certain opinion, the higher the probability that the agent adapts this opinion.

4.2.1 The Microdynamics of ABM I

Formally, let X_t denote the vector of opinions of each agent at the tth time step. Then $X_t \in \mathbb{X} = \{1, \dots, m\}^N$. Agents i and j are neighbours of each other if and only if $A_{ij} = A_{ji} = 1$. Denote by $N_i = \#(j : A_{ij} = 1)$ the number of neighbours of agent i. Further let us introduce opinion change coefficients $\alpha_{m'm''} \in [0, 1]$ for all pairs of $m', m'' \in \{1, \dots, m\}$. They regulate the likelihood for an agent to switch from one specific opinion to another. This ABM is inspired from the time-continuous ABM in [Mis12].

Now, in each time step, the probability of an agent i with opinion $X_i = m'$ to change its opinion to m'' is given by

$$\mathbb{P}[(X_{t+1})_i = m'' | (X_t)_i = m'] = \alpha_{m'm''} \frac{\#(j : A_{ij} = 1 \text{ and } (X_t)_j = m'')}{N_i} \text{ for } m' \neq m''. \tag{4.1}$$

We denote this term by $P_i^t(m', m'')$. The probability for an agent to maintain its opinion then is

$$P_i^t(m', m') = 1 - \sum_{m'' \neq m'} P_i^t(m', m''). \tag{4.2}$$

The ABM can then be written in the following algorithmic form:

Algorithm 1: Agent-based opinion change model 1

```
1 Choose end time $T$, number of agents $N$, network adjacency matrix $A$,
    opinion change coefficients $\alpha_{m'm''}$, initial opinions $X_0$
2 for $t = 0, \dots, T$ do
3     for $i = 1, \dots, N$ do
4         Draw $j$ from $\{j : A_{ij} = 1\}$ uniformly at random (Choose neighbour)
5         Draw $u_i \sim \mathcal{U}[0, 1]$
6         If $u_i < \alpha_{(X_t)_i (X_t)_j}$: $(X_{t+1})_i = (X_t)_j$ (Adapt neighbour's opinion)
7     end
8 end
```

In order to align the definitions and observations in Eqs. (4.1)–(4.2) with the notation of the previous two chapters, we define

$$X_{t+1} = F(X_t, \omega_t). \tag{4.3}$$

We define the random influence ω_t as follows:

$$\omega_t = [j_1, \dots, j_N, u_1, \dots, u_N], \; j_i \sim \mathcal{U}\{j : A_{ij} = 1\}, \; u_i \sim \mathcal{U}[0, 1]. \tag{4.4}$$

We then define F by

$$(X_{t+1})_i = F(X_t, \omega_t)_i = \begin{cases} (X_t)_{j_i} & \text{if } u_i < \alpha_{(X_t)_i,(X_t)_{j_i}} \\ (X_t)_i & \text{otherwise.} \end{cases} \tag{4.5}$$

It might seem unnecessarily complex to define the random influences in this way compared to simply letting them denote the new opinion of each agent. However, in this case the distribution that the ω_t are drawn from would change over time since the distribution of opinions of an agent's neighbours is subject to change in each time step. This would contrast the way in which we introduced both Takens' Theorem and the Mori–Zwanzig formalism in Sections 1.1.1 and 1.2.2 where the distribution of the ω_t was fixed over time. With the way we defined both ω_t and F here, we make the mathematical definition of the ABM consistent with the previously derived theory.

4.2.2 The Macrodynamics of ABM I

Social dynamics and opinion dynamics in particular are typically concerned with how the concentrations of different opinions evolve over time in contrast to who exactly has which opinion. We therefore define as the *opinion concentrations* the function

$$\phi(X) = \frac{1}{N} \begin{bmatrix} \#X_i = 1 \\ \vdots \\ \#X_i = m \end{bmatrix} \tag{4.6}$$

whose evolution we will attempt to model. For a complete network, i.e., $A_{ij} = 1$ $\forall i, j$, the ensuing expected macrodynamics can be derived analytically and without the need for memory terms in the limit of the number of agents. They read

$$\mathbb{E}[(x_{t+1})_{m'} \mid x_t] = (x_t)_{m'} + \sum_{m'' \neq m'} (\alpha_{m''m'} - \alpha_{m'm''})(x_t)_{m''}(x_t)_{m'} \text{ for } m' = 1, \ldots, m. \tag{4.7}$$

The derivation of this equation uses arguments similar to that used to construct predator-prey or SIR infection models: if all agents are neighbouring to each other, the opinion concentrations of all agents' neighbours are identical so that $P_i^t(m', m'') \equiv P^t(m', m'') = \alpha_{m'm''}(x_t)_{m''}$ is independent of i. In any given time step, the number of agents with opinion equal to m' is equal to $N \cdot (x_t)_{m'}$. Hence the expected absolute number of agents that change their opinion from m' to m'' is given by

$$\begin{aligned}
&\mathbb{E}[\#\text{Agents changing opinion from } m' \text{ to } m''] \\
&= \sum_{i:(X_t)_i = m'} P^t(m', m'') \\
&= N \cdot (x_t)_{m'} \cdot P^t(m', m'') \\
&= N \cdot (x_t)_{m'} \cdot \alpha_{m'm''} \cdot (x_t)_{m''}.
\end{aligned} \tag{4.8}$$

Since this is the absolute number of agents expected to change their opinion from m' to m'', dividing the last term by N gives the change in x_t which is $\alpha_{m'm''}(x_t)_{m''}(x_t)_{m'}$. We have to subtract the analogous term which covers the amount of agents changing their opinion from m'' to m'. This results in the factor $(\alpha_{m''m'} - \alpha_{m'm''})$ in Eq. (4.7). We see that for a complete network where all agents are neighbours of each other, the expected evolution admits a memoryless analytical representation. The main argument is that the concentration of opinions among an agent's neighbours is identical for all agents, since all agents are neighbours of each other. For other networks than a complete one, this does not have to be and, as we will see, generally is not the case.

4.2.3 Modelling the Macrodynamics for Different Networks

We now strive to model macrodynamics of this ABM under different networks. To this end, we create r realisations of the form $[X_0, \dots, X_T]$ and deduce $[x_0, \dots, x_T]$ from each. We denote the trajectories of the macrodynamics by $[\mathbf{X}_1, \dots, \mathbf{X}_r]$, divided into training data $[\mathbf{X}_1, \dots, \mathbf{X}_{\text{train}}]$ and testing data $[\mathbf{X}_{\text{train}+1}, \dots, \mathbf{X}_r]$. We choose the number of opinions to be $m = 3$.

Since the entries of each x_t sum up to 1, we model only the dynamics of the first 2 entries, i.e., the concentrations of the first two opinions, as the concentrations of the third opinion can always be deduced from those of the first two ones. In order to model these macrodynamics, we use only the SINAR method with basis functions given by

$$\begin{aligned}
\psi(\hat{x}_{t-1}) = [&\underbrace{(x_{t-1})_1, (x_{t-1})_2, (x_{t-1})_1^2, (x_{t-1})_2^2, (x_{t-1})_1 (x_{t-1})_2,}_{\text{Markovian terms}} \\
&\underbrace{(x_{t-2})_1, (x_{t-2})_2, (x_{t-2})_1^2, (x_{t-2})_2^2, (x_{t-2})_1 (x_{t-2})_2, \dots]}_{\text{Memory terms}}.
\end{aligned} \tag{4.9}$$

These are inspired from the macrodynamics for the complete network in Eq. (4.7). Surely one could use other methods of the ones introduced in Chapters 2 and 3 but the point of this chapter is to demonstrate that memory generally improves the modelling of the evolution of macrostates of an ABM. Since SINAR produced sound results in Chapter 2, we use it in this chapter. We deploy SINAR both without the sparsity constraint ($c = 0$) and with $c = 0.05$.

For each memory depth $p = 1, \dots, 10$, we estimate a SINAR model with the basis function given above. We then evaluate its accuracy in the following way: we divide each trajectory $\mathbf{X}_i$ of testing data into blocks of length $l \geq p$, with the jth block given by $\mathbf{x}_i^{(j)} = [x_{jl}, \dots, x_{(j+1)l-1}]$. We then use the last p states as starting values for a reconstruction with SINAR, denoted by $\tilde{\mathbf{x}}_i^{(j)} = [\tilde{x}_{jl}, \dots, \tilde{x}_{(j+1)l-1}]$. The relative Euclidean error between data and this reconstruction is calculated for each block by

$$\mathcal{E}(\hat{\mathbf{x}}_i^{(j)}) = \frac{\|\mathbf{x}_i^{(j)} - \tilde{\mathbf{x}}_i^{(j)}\|_F}{\|\mathbf{x}_i^{(j)}\|_F}. \tag{4.10}$$

Finally, we compute the mean over all $\mathcal{E}(\hat{\mathbf{x}}_i^{(j)})$.

Please note that with this choice of basis functions there is no guarantee that the predicted opinion concentrations remain non-negative and always sum up to 1. However, we will always make short-term predictions so that violation of this property should typically be at most negligible.

As opinion change coefficients, we specify

$$\begin{bmatrix} \alpha_{11} & \alpha_{12} & \alpha_{13} \\ \alpha_{21} & \alpha_{22} & \alpha_{23} \\ \alpha_{31} & \alpha_{32} & \alpha_{33} \end{bmatrix} = \begin{bmatrix} 0 & 0.165 & 0.03 \\ 0.03 & 0 & 0.165 \\ 0.165 & 0.03 & 0 \end{bmatrix}. \tag{4.11}$$

In each experiment, we create $r = 20$ realisations out of which 12 are used for training and 8 for testing and use as block length in the testing process $l = 20$.For this thesis, new realisations of the ABM have been performed compared to the ones used in [WKS21]. They admit interpretations which are consistent with the ones derived in [WKS21].

A complete network

To begin, we use a complete network, i.e., $A_{ij} = 1$ for all i, j. We choose $N = 5000$ and $T = 300$. The initial opinions are chosen so that $\phi(X_0) = [0.45, 0.1, 0.45]^T$ in each realisation of the ABM.

The results are depicted in Figure 4.1. The number of agents seems to be high enough so that macrostates of the realisations are close to the expected macrodynamics: we can see oscillating behaviour of the opinion concentrations since agents with opinion 1 are likely to move to opinion 2 (see α in Eq. (4.11)), from 2 to 3 and from 3 to 1. Unsurprisingly, the inclusion of memory terms does not yield an improvement in accuracy, since for a complete network, the expected dynamics can be formulated without memory. We can further observe that enforcing the sparsity constraint while using memory even gives an improvement, possibly because it prevents overfitting. For the one-step prediction error we can see only a negligible improvement with memory. It is unclear as to why there is an improvement at all.

Comparing the estimated model with the analytically derived expected macro-dynamics using Eq. (4.7) (see Appendix C.1 in [WKS21]), we can see that the amount of data and number of agents used is enough to allow a very precise reconstruction of the model coefficients: the analytic model reads

$$\begin{aligned}\mathbb{E}[(x_{t+1})_1 \mid x_t] &= 1.135(x_t)_1 - 0.135(x_t)_1^2 - 0.27(x_t)_1(x_t)_2,\\ \mathbb{E}[(x_{t+1})_2 \mid x_t] &= 0.865(x_t)_2 + 0.135(x_t)_2^2 + 0.27(x_t)_1(x_t)_2.\end{aligned} \tag{4.12}$$

while the estimated one (with $p = 1$) reads

$$\begin{aligned}(x_{t+1})_1 &= 1.1353(x_t)_1 - 0.1351(x_t)_1^2 - 0.2709(x_t)_1(x_t)_2\,,\\ (x_{t+1})_2 &= 0.8655(x_t)_2 + 0.1344(x_t)_2^2 + 0.2699(x_t)_1(x_t)_2\,,\end{aligned} \tag{4.13}$$

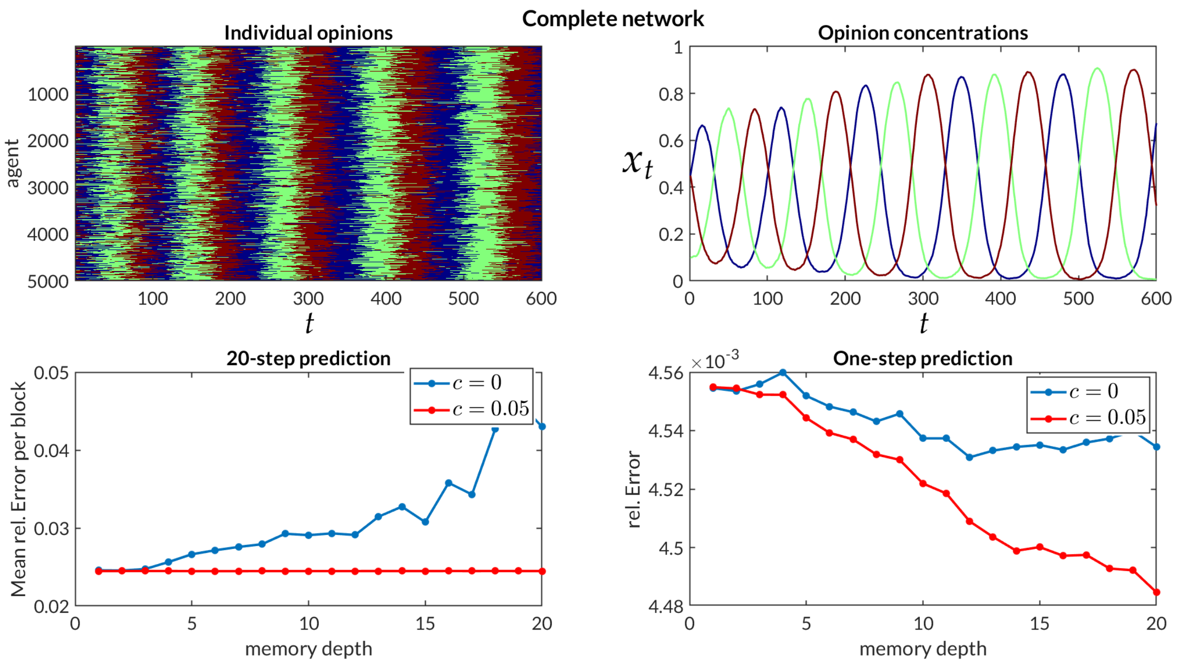


Figure 4.1: Results for the complete network. Top left: microstates of one realisation of ABM I over time. Each column denotes the individual opinions of all agents at a given time step. Top right: macrodynamics corresponding to this realisation. Bottom left: 20-step relative Euclidean prediction errors for SINAR with memory depths $p = 1, \ldots, 10$ with and without sparsity constraint. Bottom right: one-step relative prediction error.

A two-cluster network

In this experiment, we maintain the number of agents at $N = 5000$ but divide the network into two clusters. In each cluster, all agents are neighbours of each other while there are few such links between the clusters. Links between clusters are assigned randomly with a probability of 10^{-4} for each pair of agents. With clusters of size 2500 each, there are $2500^2 = 6.25 \cdot 10^6$ possible links between agents from different clusters so that we expect 625 links to exist between both clusters. We set $T = 600$. For the initial macrostate, opinions in the first cluster are distributed by $[0.8, 0.1, 0.1]$ and in the second cluster by $[0.1, 0.1, 0.8]$. In order to demonstrate the difference in the macrodynamics compared to the full network, it is important to start the microdynamics in each cluster with different opinion concentrations. Otherwise the macrostates in each cluster would develop almost identically, yielding the macrodynamics across all agents to be almost identical to the one in the previous example.

We can see in Figure 4.2 that in this case, memory does improve the short-term predictions. The sparse model is slightly less accurate but more interpretable since only few coefficients are non-zero:

$$\begin{aligned} &c = 0 : A_{\text{SINAR}} = \\ &\begin{bmatrix} 2.04 & 0.03 & -0.07 & -0.08 & 0.02 & -1.05 & -0.02 & 0.07 & 0.07 & -0.02 \\ -0.05 & 1.88 & 0.00 & 0.11 & 0.06 & 0.06 & -0.89 & -0.01 & -0.12 & -0.05 \end{bmatrix} \\ &c = 0.05 : A_{\text{SINAR}} = \begin{bmatrix} 1.9691 & 0 & 0 & 0 & 0 & -0.9700 & 0 & 0 & 0 & 0 \\ 0 & 1.9662 & 0 & 0 & 0 & 0 & -0.9671 & 0 & 0 & 0 \end{bmatrix} \end{aligned} \tag{4.14}$$

yielding a linear model of the form (with $c = 0.05$)

$$\begin{aligned} (x_{t+1})_1 &= 1.9691(x_t)_1 - 0.9700(x_{t-1})_1 \\ (x_{t+1})_2 &= 1.9662(x_t)_2 - 0.9671(x_{t-1})_2 \end{aligned} \tag{4.15}$$

For higher memory depths however, the models become nonlinear.

There is an intuitive reason for the importance of memory here and it is conceptually identical to the one given in the simple example at the beginning of Chapter 1:

Assume for now that the clusters are fully distinct, i.e., no links between them exist. In this setting, the macrodynamics for the opinion concentrations in each cluster can be formulated without memory as explained previously. Across the full network, the opinion concentrations are then given by the averages of the cluster-wise concentrations, denoted by $x_t^{(i)}$, so that $x_t = \frac{1}{2}(x_t^{(1)} + x_t^{(2)})$. Obviously there exists a range of different pairs of $x_t^{(1)}, x_t^{(2)}$ which fulfil this. Now, additionally given x_{t-1}, we could now find $x_{t-1}^{(1)}$ and $x_{t-1}^{(2)}$ so that these equations would yield those values for $x_t^{(1)}$ and $x_t^{(2)}$ whose average is x_t. There are much fewer pairs of $x_t^{(1)}$ and $x_t^{(2)}$ compared to all pairs whose average is x_t. From these $x_t^{(i)}$ subsequent values of $x_{t+1}^{(i)}$ could be computed. We have therefore derived a more precise estimate of $x_t^{(1)}$ and $x_t^{(2)}$ and therefore of x_{t+1}. The evolution of x_t is stochastic so that one would not search for those $x_{t-1}^{(i)}$ which exactly yield x_t but rather get different probabilities for the $x_t^{(i)}$ depending on what x_{t-1} is.

The numerical example on the two-cluster network thus demonstrates both the

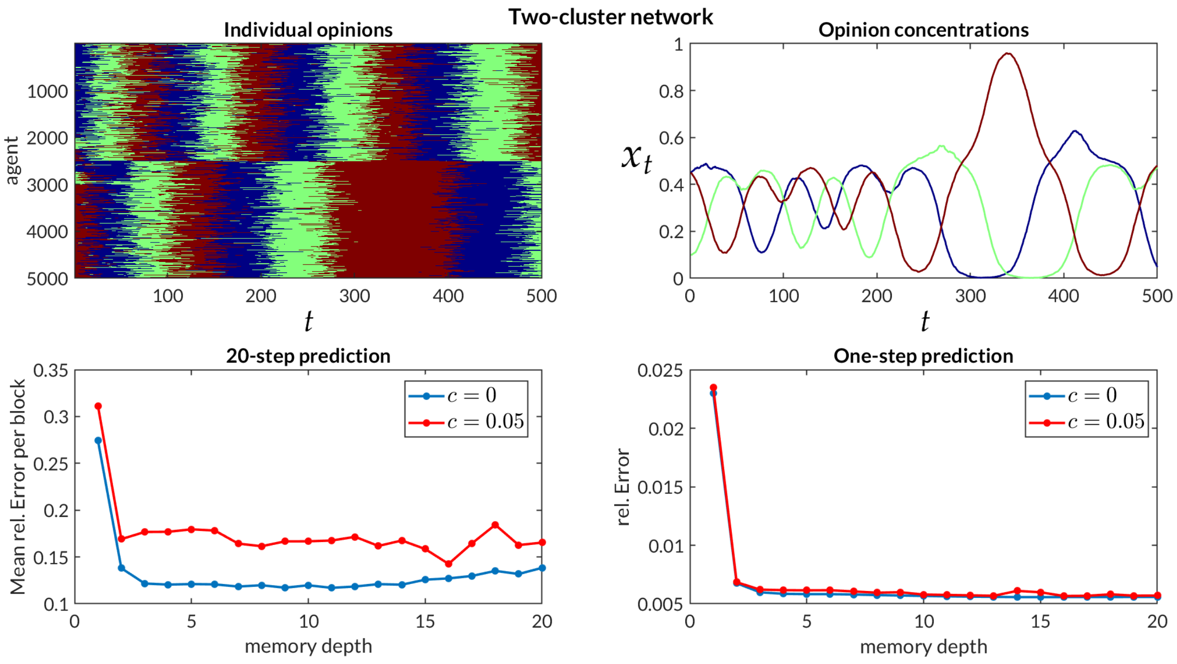


Figure 4.2: Results for the two-cluster network. Analogously to Figure 4.1, the figure in the top left shows one realisation of the microdynamics. We can see the division into two almost distinct evolutions between the two clusters. Top right: the corresponding macrostates evolve much more irregularly than in the full-network setting. Bottom: 20- and one-step prediction errors show that memory generally improves the prediction accuracy. For the 20-step predictions, the non-sparse model is slightly better.

intuitive and the practical indication that memory improves the estimation of the macrodynamics. We have formulated the ABM and the macrostate such that the setting is consistent with the Mori–Zwanzig formalism from which a nonlinear autoregressive model emerges which we estimate with SINAR. In future work on the modelling of statistics of ABMs and real-world social phenomena, this combination of theory and numerical examples should emphasise that a modelling method which uses memory is advisable to deploy.

A five-cluster network

We further demonstrate this theme on a network with five clusters. Again $N = 5000$, while the initial concentrations per cluster are given by $[0.8, 0.1, 0.1]$, $[0.1, 0.1, 0.8]$, $[0.1, 0.8, 0.1]$, $[0.3, 0.4, 0.3]$ and $[0.5, 0.3, 0.2]$. We set $T = 800$. All other simulation parameters are identical to the ones used in the previous two examples. We can see even more clearly than in the two-cluster network setting that memory improves the predictions. For $p = 3$, the non-sparse SINAR model reads,

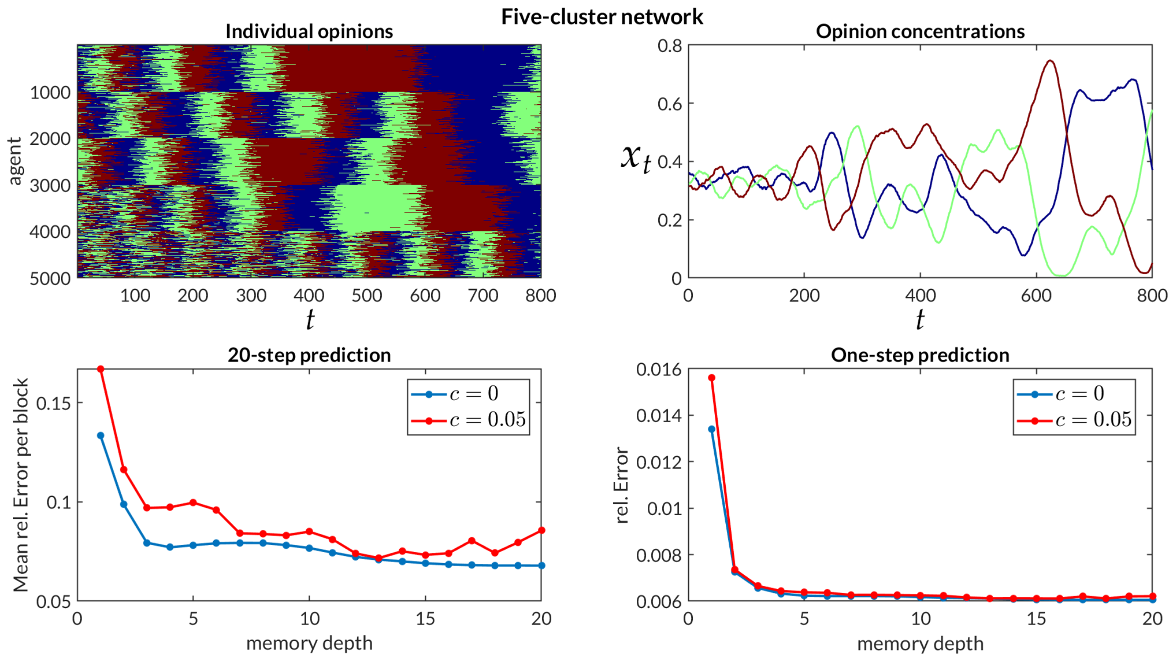


Figure 4.3: Results for the five-cluster network. Top left: one realisation of the microdynamics. Top right: corresponding macrostates over time. We can see more complex behaviour than in the first two examples. Bottom: 20-step and one-step relative Euclidean prediction errors of SINAR for different memory depths p with $c = 0$ and $c = 0.05$.

$$\begin{aligned}(x_{t+1})_1 = &1.4662(x_t)_1 - 0.1188(x_t)_2 + 0.0552(x_t)_1^2 + 0.1318(x_t)_1(x_t)_2 + 0.2309(x_{t-1})_2 \\ &- 0.1899(x_{t-1})_1(x_{t-1})_2 - 0.2021(x_{t-1})_2^2 - 0.4658(x_{t-2})_1 - 0.1060(x_{t-2})_2 \\ &+ 0.1206(x_{t-2})_1^2 + 0.0644(x_{t-2})_2^2\end{aligned} \tag{4.16}$$

which clearly is a highly nonlinear model. Figure 4.3 shows that again memory drastically improves the prediction accuracy.

Figure 4.4 is an exemplary illustration of predictions using different memory depths. While the memoryless models are imprecise approximations – they actually seem linear – with a higher memory depth the predictions become more and more accurate.

4.2.4 Dependence of Performance on Amount of Training Data

Let us now investigate how the prediction accuracy behaves under varying amounts of training data.

For the five-cluster network, we subsequently increase the number of time steps in the 12 realisations used for training. We then calculate the prediction errors with the ensuing SINAR models analogously to previously. Figure 4.5 shows the prediction errors over the amount of training data used for $p = 1, 2, 10$.

We can see that the one-step prediction error decreases logarithmically with data

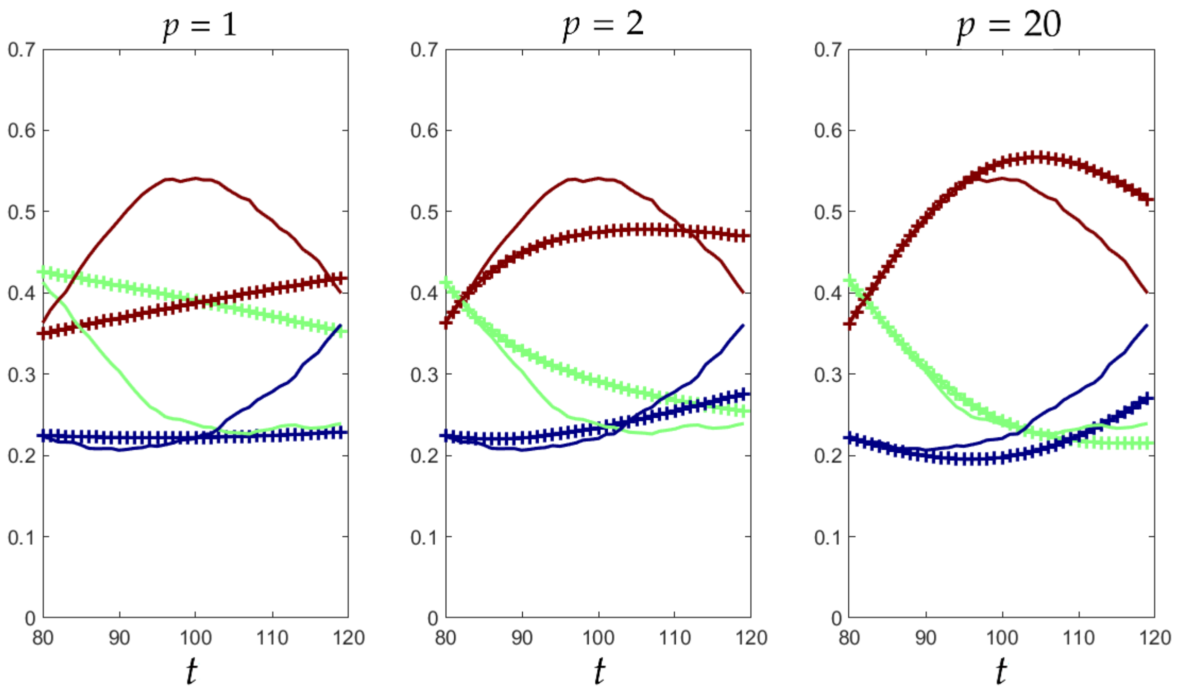


Figure 4.4: Predicted (dashed lines) and observed (solid lines) opinion concentrations over 40 time steps using $p = 1, 2$ and 20 and $c = 0$ in SINAR. Observed concentrations are shown in thin lines and predicted concentrations with lines with crosses.

size. For low amount of training data, the models with $p = 10$ are much less precise than the others, potentially since they are more complex and require more data. For sufficient data, they are better than the ones with little or no memory. For the 20-step prediction error, errors which are higher than 0.7 are left out in the figure for the sake of visualisation. The models determined with less than 125 time steps per realisation diverge for all three depicted memory depths. For $p = 10$, even with 250 time steps the model is inaccurate and gives a high error. These results indicate that the more complex high-memory models also require more data for the parameter estimation. As we have previously seen in SINAR, if the data quality is not sufficient, then ensuing models are prone to induce diverging dynamics.

4.2.5 Analytical and Numerical Search for the Expected Two-cluster Macrodynamics

After tackling the macrodynamics from a numerical point of view using simulated realisations of ABM I, let us now try to determine an analytical solution for the expected macrodynamics for non-complete networks. This could shed light on the numerical results and bring further research question to the surface.

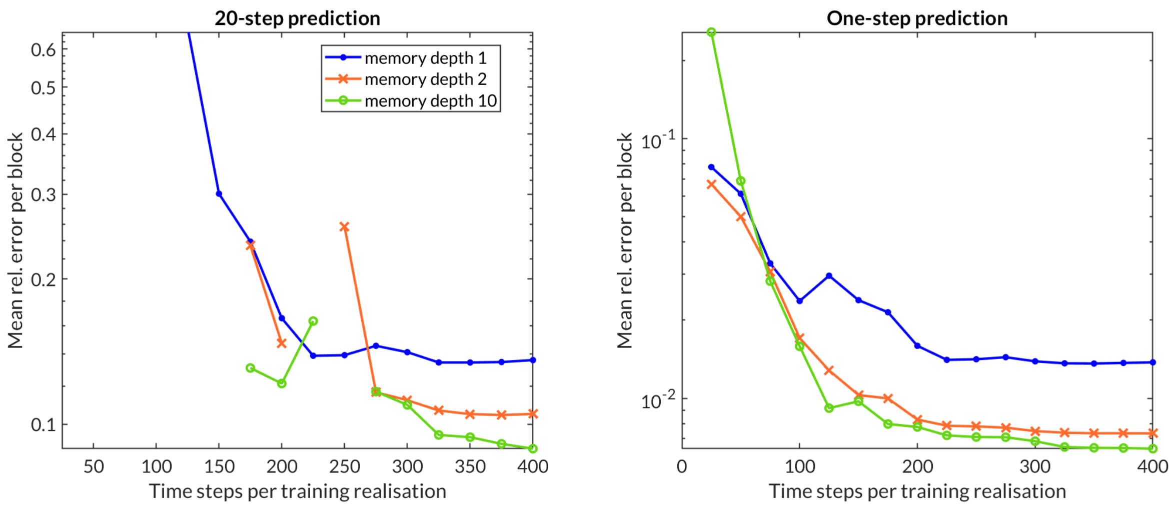


Figure 4.5: Prediction error of SINAR without sparsity constraint for macrodynamics from ABM I over size of training data for the five-cluster network. Left: 20-step prediction error. Right: one-step prediction error.

Intra-cluster macrodynamics for two clusters

Let us first derive a closed expression for the evolutions of opinion concentrations in each of the clusters of a two-cluster network, i.e., the intra-cluster macrodynamics for $x_t^{(1)}$ and $x_t^{(2)}$. It turns out that as in the case of the complete network, letting the number of agents converge towards infinity, we can state memoryless expected intra-cluster macrodynamics. Let the probability for agents between different clusters to be neighbours be given by p_{link} while all agents in the same cluster are connected. For $N \to \infty$, the distribution among clusters of an agent's neighbours is then given by

$$\rho := \frac{p_{\text{link}}}{1+p_{\text{link}}} \text{ in the other cluster and } 1-\rho = \frac{1}{1+p_{\text{link}}} \tag{4.17}$$

in the same cluster. The probability for an agent with opinion m' in cluster 1 to adapt opinion m'' is thus given by

$$P_1^t(m', m'') = \alpha_{m'm''}((1-\rho)(x_t)_{m''}^{(1)} + \rho(x_t)_{m''}^{(2)}). \tag{4.18}$$

Let us denote the indices corresponding to the agents in cluster 1 by $\mathcal{C}_1$. We can then observe for the expected number of agents changing their opinion from m' to m'' in cluster 1, without loss of generality,

$$\begin{aligned}
&\mathbb{E}[\#\text{Agents changing opinion from } m' \text{ to } m'' \text{ in cluster } 1] \\
&= \sum_{i\in\mathcal{C}_1:(X_t)_i=m'} P_1^t(m',m'') \\
&= \frac{N}{2}\cdot (x_t)_{m'}^{(1)} \cdot \alpha_{m'm''}((1-\rho)(x_t)_{m''}^{(1)} + \rho(x_t)_{m''}^{(2)}).
\end{aligned} \tag{4.19}$$

For the intra-cluster macrodynamics, this means, by dividing by $\frac{N}{2}$ and collating terms from Eq. (4.19),

$$\begin{aligned}&\mathbb{E}[(x_{t+1})^{(1)}_{m'}] = \\ &(x_t)^{(1)}_{m'} + \sum_{m''\neq m'} \underbrace{(x_t)^{(1)}_{m''}(x_t)^{(1)}_{m'}(1-\rho)(\alpha_{m''m'} - \alpha_{m'm''})}_{\text{(I)}} \\ &+ \underbrace{\rho((x_t)^{(1)}_{m''}(x_t)^{(2)}_{m'}\alpha_{m''m'} - (x_t)^{(1)}_{m'}(x_t)^{(2)}_{m''}\alpha_{m'm''})}_{\text{(II)}}.\end{aligned}$$

(I): win minus loss of agents which adapted opinion of neighbour from same cluster

(II): win minus loss of agents which adapted opinion of neighbour from other cluster

(4.20)

and analogously for the second cluster. The derivation is analogous to the one for the macrodynamics for the full network in Eq. (4.8).

We can see in Figure 4.6 that with a value of $p_{\text{link}} = 0.05$, the intra-cluster macrostates quickly converge towards each other component-wise. In the numerical examples above we chose $p_{\text{link}} = 10^{-4}$ so that this did not happen in the depicted time frame and instead the dynamics seemed more irregular due to the more stochastic influences from the low number of links.

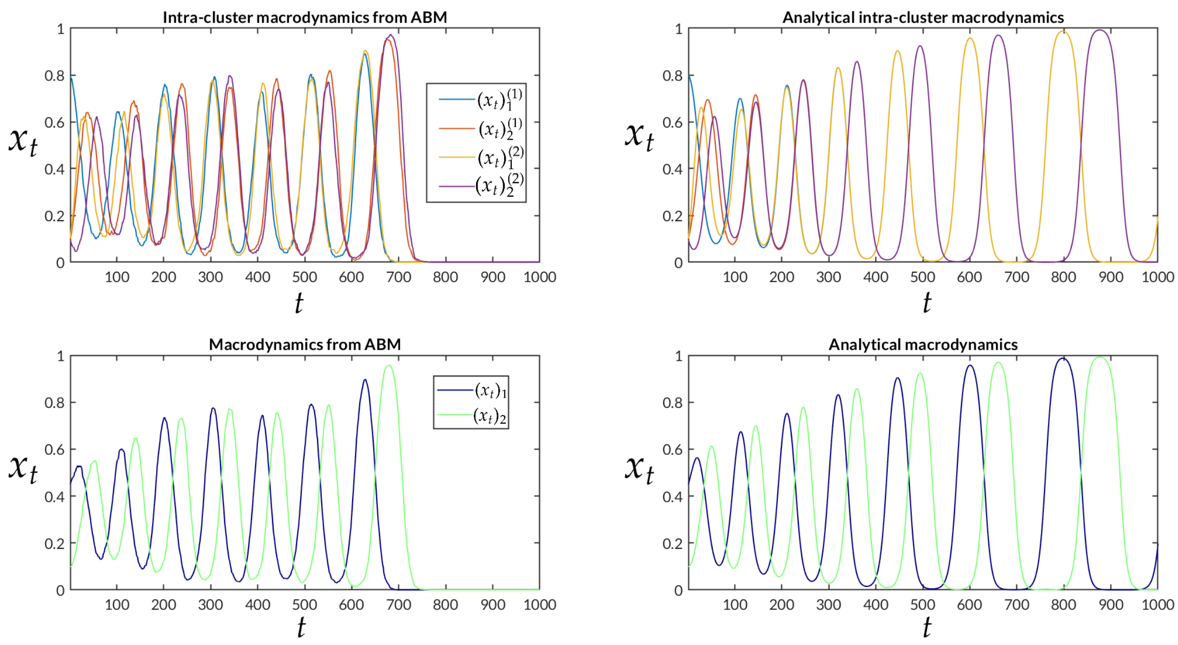


Figure 4.6: Evolution of intra-cluster macrostates (top) and full macrostates (bottom) from an ABM realisation with 5000 agents (left) and analytical solution (right) with the same parameters as in the numerical example on the two-cluster example but with $p_{\text{link}} = 0.05$ and initial values $x_0^{(1)} = [0.8, 0.1, 0.1], x_0^{(2)} = [0.1, 0.8, 0.1]$. After approximately 200 time steps the intra-cluster macrostates are almost congruent in both clusters. This happened regardless of the initial values. In this ABM realisation, the opinion concentrations converge unlike the expected dynamics.

For completion, we can also formulate the intra-cluster macrodynamics for a general number of clusters and different numbers of links between the clusters. Let us denote the probability of a link between agents between clusters i and j by $(p_{\text{link}})_{ij}$. Then for cluster i, the expected ratio of neighbours in other clusters is

$$\rho_{ij} := \frac{(p_{\text{link}})_{ij}}{1 + \sum_{k \neq i} (p_{\text{link}})_{ik}} \text{ for } i \neq j \text{ and } \rho_i := 1 - \sum_{j \neq i} \rho_{ij} \tag{4.21}$$

in the same cluster where $(p_{\text{link}})_{ii} = 1$. Then it follows,

$$\begin{aligned} \mathbb{E}(x_{t+1})^{(i)}_{m'}] = \\ & (x_t)^{(i)}_{m'} + \sum_{m'' \neq m'} \underbrace{\rho_i (x_t)^{(i)}_{m''} (x_t)^{(i)}_{m'} (\alpha_{m''m'} - \alpha_{m'm''})}_{\text{(I)}} \\ & \underbrace{+ \sum_{j \neq i} \rho_{ij} \big((x_t)^{(i)}_{m''} (x_t)^{(j)}_{m'} \alpha_{m''m'} - (x_t)^{(i)}_{m'} (x_t)^{(j)}_{m''} \alpha_{m'm''} \big) .}_{\text{(II)}} \end{aligned} \tag{4.22}$$

(I): win minus loss of agents which adapted opinion of neighbour from same cluster

(II): win minus loss of agents which adapted opinion of neighbour from other clusters

In order to formulate the evolution for the case of directed networks or varying number of agents inside a cluster, one can simply modify the terms for the ρ_{ij} accordingly.

Further numerical examples showed that in the long term, the intra-cluster macrostates converge towards each other. This happened the earlier the larger the values for the $(p_{\text{link}})_{ij}$ were. These observations do not answer the question how to formulate the exact expected macrodynamics for two clusters, but they should serve as a starting point for the influence of the number of links on the expected macrodynamics. Findings could be generalized to observed dynamics for clustered networks in other contexts, e.g., gene regulatory networks [HDPCBS18].

Analytical macrodynamics for two clusters

To approach the analysis of ABM I from a different persepctive, let us now attempt to derive closed memory-exhibiting equations for the full macrodynamics, i.e., the evolution of the opinion concentration across the full network of agents as done in [WKS21]. For simplification, let us assume separated clusters, i.e., $p_{\text{link}} = 0$. Then the intra-cluster dynamics are given as in Eq. (4.7) by

$$\begin{aligned}(x_{t+1}^{(i)})_1 &= (1+\alpha_{31}-\alpha_{13})(x_t^{(i)})_1 + (\alpha_{13}-\alpha_{31})(x_t^{(i)})_1^2 + (\alpha_{21}-\alpha_{12}-\alpha_{31}+\alpha_{13})(x_t^{(i)})_1(x_t^{(i)})_2\\ (x_{t+1}^{(i)})_2 &= (1+\alpha_{32}-\alpha_{23})(x_t^{(i)})_2 + (\alpha_{23}-\alpha_{32})(x_t^{(i)})_2^2 + (\alpha_{12}-\alpha_{21}-\alpha_{32}+\alpha_{23})(x_t^{(i)})_1(x_t^{(i)})_2.\end{aligned} \tag{4.23}$$

With $x_t = \frac{1}{2}(x_t^{(1)} + x_t^{(2)})$ and, for simplification, denoting $a = \alpha_{31} - \alpha_{13}, b = \alpha_{21} - \alpha_{12} - \alpha_{31} + \alpha_{13}, c = \alpha_{32} - \alpha_{23}, d = \alpha_{12} - \alpha_{21} - \alpha_{32} + \alpha_{23}$, we can reformulate this to

$$\begin{aligned}(x_{t+1})_1 &= (1+a)\underbrace{\frac{1}{2}((x_t^{(1)})_1 + (x_t^{(2)})_1)}_{(x_t)_1} - \frac{a}{2}((x_t^{(1)})_1^2 + (x_t^{(2)})_1^2)\\ &\quad + \frac{b}{2}((x_t^{(1)})_1(x_t^{(1)})_2 + (x_t^{(2)})_1(x_t^{(2)})_2)\\ (x_{t+1})_2 &= (1+c)\underbrace{\frac{1}{2}((x_t^{(1)})_2 + (x_t^{(2)})_2)}_{(x_t)_2} - \frac{c}{2}((x_t^{(1)})_2^2 + (x_t^{(2)})_2^2)\\ &\quad + \frac{d}{2}((x_t^{(1)})_1(x_t^{(1)})_2 + (x_t^{(2)})_1(x_t^{(2)})_2).\end{aligned} \tag{4.24}$$

For coefficients chosen in the numerical examples, it holds $a = -c$ and $b = -d = -2a$. This gives

$$\begin{aligned}(x_{t+1})_1 &= (1+a)\underbrace{\frac{1}{2}((x_t^{(1)})_1 + (x_t^{(2)})_1)}_{(x_t)_1} - \frac{a}{2}((x_t^{(1)})_1^2 + (x_t^{(2)})_1^2)\\ &\quad - a((x_t^{(1)})_1(x_t^{(1)})_2 + (x_t^{(2)})_1(x_t^{(2)})_2)\\ (x_{t+1})_2 &= (1-a)\underbrace{\frac{1}{2}((x_t^{(1)})_2 + (x_t^{(2)})_2)}_{(x_t)_2} + \frac{a}{2}((x_t^{(1)})_2^2 + (x_t^{(2)})_2^2)\\ &\quad + a((x_t^{(1)})_1(x_t^{(1)})_2 + (x_t^{(2)})_1(x_t^{(2)})_2).\end{aligned} \tag{4.25}$$

It proved infeasible to further derive a closed expression for the macrodynamics from here onwards so that this would require further research. However, the following numerical results give intuition on the structure of the macrodynamics. For certain initial conditions, already a memory depth of $p = 2$ is enough to almost exactly model the macrodynamics.

Symmetric initial conditions Let us assume so-called *symmetric* initial conditions for which it holds

$$(x_0^{(1)})_1 = (x_0^{(2)})_2 = 1 - 2(x_0^{(1)})_2, \quad (x_0^{(2)})_1 = (x_0^{(1)})_2 = 1 - 2(x_0^{(2)})_2. \tag{4.26}$$

so that they are completely determined by the value $(x_0^{(1)})_1$. As in the examples, we choose $a = 0.135$.

We generate trajectories of length $T = 1000$ for $(x_0^{(1)})_1 = 0.01, 0.02, \dots, 0.99$, and for each value train an NAR model of memory depth $p = 2$ with SINAR without sparsity constraint and the same basis functions as used before, using the first 500 time steps. The second 500 time steps are used to assess the quality of the models by computing one-step and long-term predictions on trajectories from each initial condition, giving us 99 different testing trajectories. We then consider the average relative one-step and 500-step prediction errors as defined in Eq. (4.10) on all testing trajectories. The trajectory for $(x_0^{(1)})_1 = 0.25$ is shown in Figure 4.7. We can see that the trajectory spirals outwards onto a triangular attractor. For higher values of $(x_0^{(1)})_1$, the trajectory starts closer to the attractor and reaches it earlier.

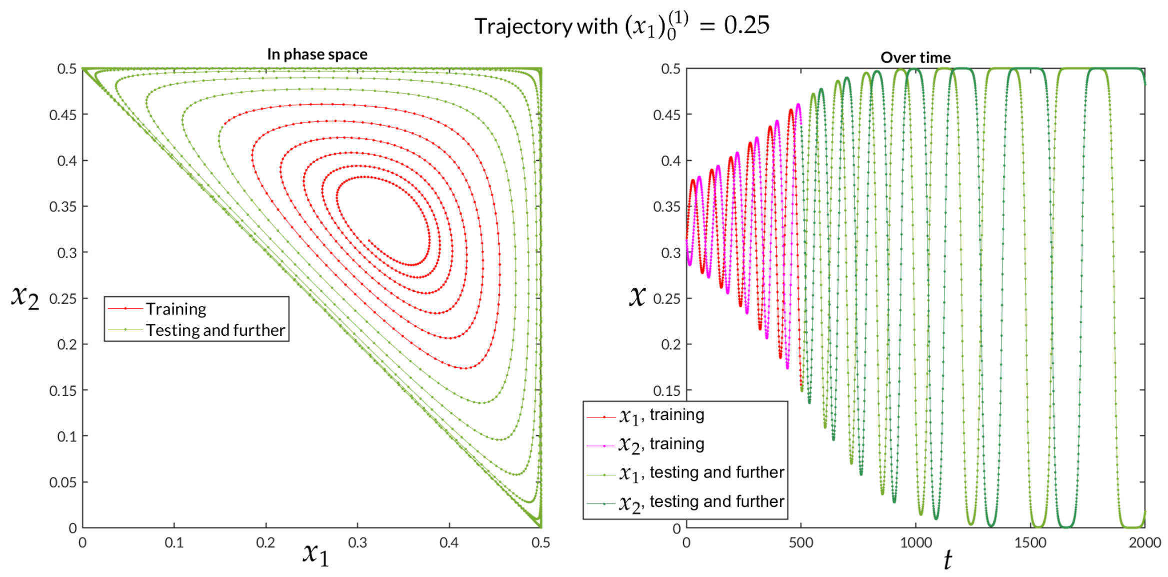


Figure 4.7: Trajectory of the two-cluster macrodynamics with $(x_0^{(1)})_1 = 0.25$, divided into training data (time steps 1–500), testing data (501–1000) and further time steps not used in the results but depicted for illustration. The starting value of the macrodynamics is given by $(0.3125, 0.3125)$ (see Eq. (4.26)).

We obtain various different models which all give a one-step error close to the machine precision of 10^{-16} (Figure 4.8). The long-term prediction error is higher but usually smaller than 10^{-5}. For $p = 2$, some predicted trajectories diverge but this seems due to numerical errors. For $p = 10$, the errors are comparable in the short-term and slightly smaller in the long-term and no predictions diverge. Considering the detected models for $p = 2$ obtained using trajectories from different starting

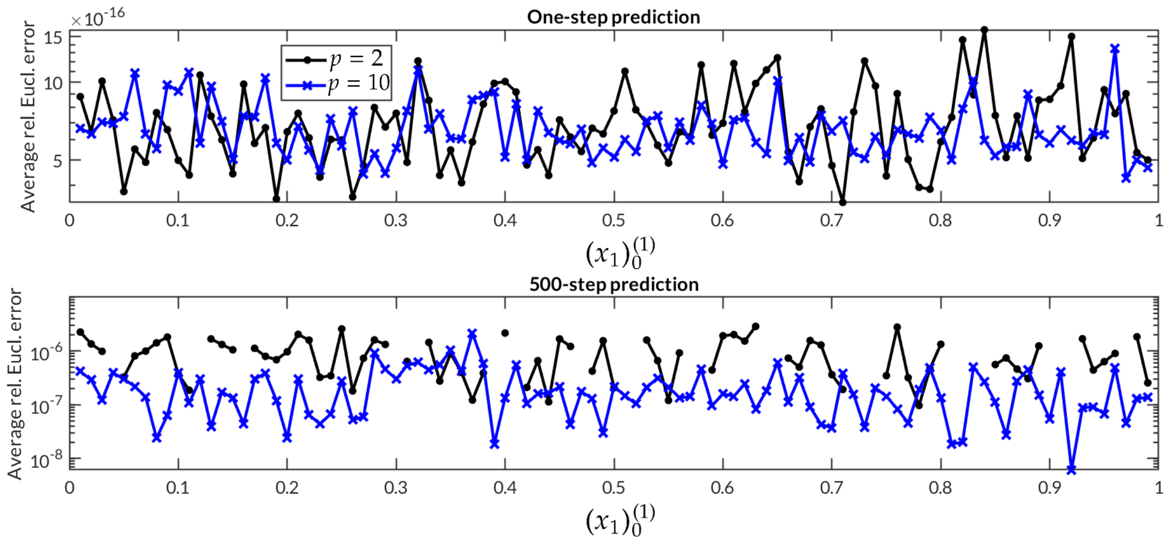


Figure 4.8: One-step and 500-step prediction error of models produced with trajectories from different symmetric initial conditions and memory depths $p = 2, 10$ in SINAR. For $p = 1$, the one-step prediction error is approximately 10^{-2}.

values, for example, for $(x_0^{(1)})_1 = 0.8$ and 0.25, we find for the coefficient matrices,

$$
\begin{aligned}
(x_0^{(1)})_1 = 0.8 : &\begin{bmatrix} 0.91 & -0.96 & 0.27 & 0.54 & 0 & 0 & 1.05 & -0.08 & -0.54 & -0.19 \\ -0.04 & 2.09 & 0 & -0.54 & -0.27 & 0 & -1.05 & 0.08 & 0.54 & 0.19 \end{bmatrix} \\
(x_0^{(1)})_1 = 0.25 : &\begin{bmatrix} 1.83 & -0.03 & 0.27 & 0.54 & 0 & -0.80 & 0 & -0.33 & -0.54 & 0.0642 \\ -0.97 & 1.17 & 0 & -0.54 & -0.27 & 0.80 & 0 & 0.33 & 0.54 & -0.06 \end{bmatrix}.
\end{aligned}
\tag{4.27}
$$

Both models occur for many other initial conditions although it was not possible to detect a structure as to which initial condition produced which model form. Although the models do not admit an immediate or obvious interpretation, the coefficients 0.27 and −0.27 for the basis functions $(x_t)_1^2$, $(x_t)_2^2$ and 0.54, −0.54 for the basis functions $(x_t)_1(x_t)_2$ and $(x_{t-1})_1(x_{t-1})_2$ occur for all values of $(x_0^{(1)})_1$. The others are generally different but the latter 5 coefficients (the ones corresponding to the memory terms) are usually equal for both coordinates but with a different sign. Only the coefficients for $(x_t)_1$ and $(x_t)_2$ show no apparent structure. Using basis functions of degree 3 by additionally including all products of the form $(x_t)_i(x_t)_j(x_t)_k$, their coefficients are estimated as 0. Apparently, for symmetric initial conditions, the two-cluster dynamics allow for an exact description with the degree 1 and 2 basis functions as in Eq. (4.9) and a memory depth of $p = 2$.

Non-symmetric initial conditions For non-symmetric initial concentrations, NAR models with the chosen basis functions for $p = 2$ are generally much less accurate. We construct initial conditions by starting with symmetric ones and adding a value

δ to $(x_0^{(1)})_2$ so that

$$(x_0^{(1)})_1 = (x_0^{(2)})_2 = 1 - 2((x_0^{(1)})_2 - \delta), \quad (x_0^{(2)})_1 = (x_0^{(1)})_2 - \delta = 1 - 2(x_0^{(2)})_2. \tag{4.28}$$

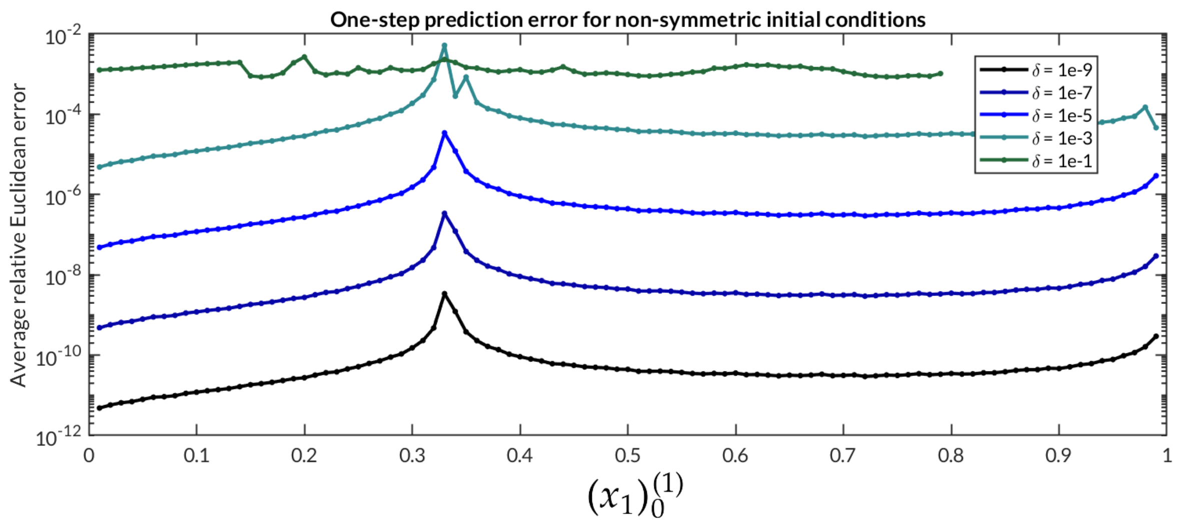
We then conduct the same analysis as above and consider the one-step and 500-step prediction error for $p = 2$ with different values for δ (Figure 4.9a). The testing trajectories are again the ones from symmetric initial conditions. The one-step prediction error increases by a factor of 10 for an increase of δ by a factor of 10. Long-term predictions diverge regularly within 500 steps for $\delta \geq 1e-7$ so that the long-term prediction error is not shown. We can see that the highest error comes from the model generated with $(x_0^{(1)})_1 = 0.33$. The reason is that for $(x_0^{(1)})_1 = 1/3$ all initial concentrations are equal and the macrostate stays constant so that SINAR cannot learn the model properly. For initial values close to this fixed point of the macrodynamics, the dynamics apparently take long to reach behaviour from which a correct model can be inferred.

These observations indicate that for non-symmetric initial conditions a memory depth of $p = 2$ is not enough to capture all relevant features of the dynamics.

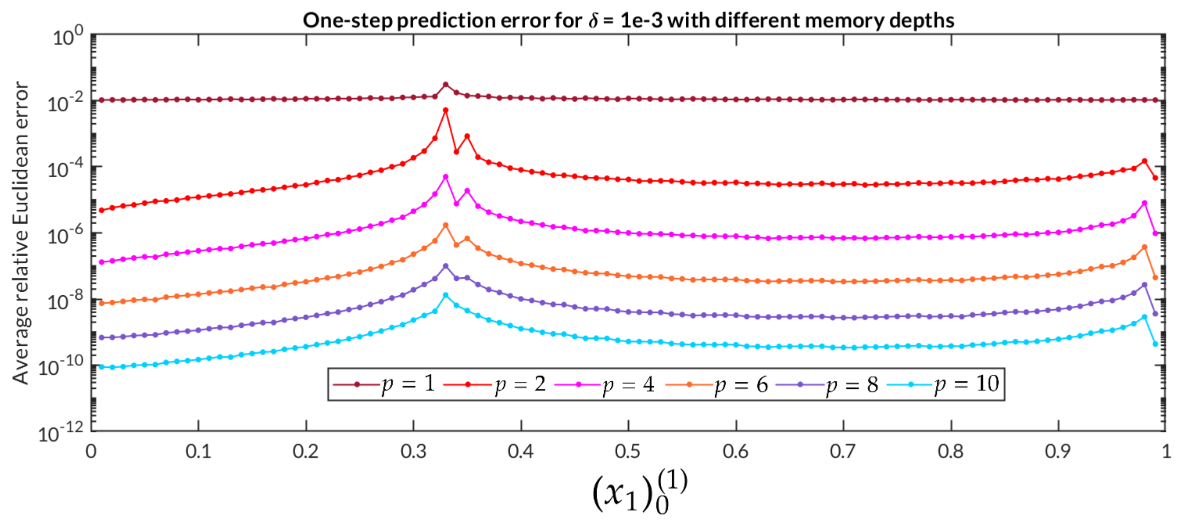
For higher memory depths, we can observe in Figure 4.9b an improvement although even for the one-step error, no model generates accuracy which is better than 10^{-11}. Although this may seem precise, the obtained models regularly diverge within 100 steps. The model coefficients also differ from the ones obtained with symmetric initial conditions. The training trajectories, however, do not seem very different from the ones starting at symmetric trajectories (see Figure 4.10). Apparently the dynamics themselves are not chaotic but either the ensuing parameter estimation problem in SINAR is badly conditioned or the set of model coefficients for which the resulting long-term predictions do not diverge is very small.

It is, however, unclear at this point whether the models are error-prone or the testing trajectories unsuited. Therefore, we compare model performances from different initial conditions and on different testing trajectories. Figure 4.9c shows a comparison of the one-step prediction errors for models obtained with symmetric and non-symmetric ($\delta = 1e-3$) initial conditions on testing trajectories starting at symmetric, non-symmetric ($\delta = 1e-3$) and randomly chosen (uniformly) initial conditions for $p = 2$. For the testing data from symmetric initial conditions, the model from symmetric initial conditions fairs better and vice versa. For random initial conditions, the models for non-symmetric starting values are slightly better while still giving a large error of approximately 10^{-2}.

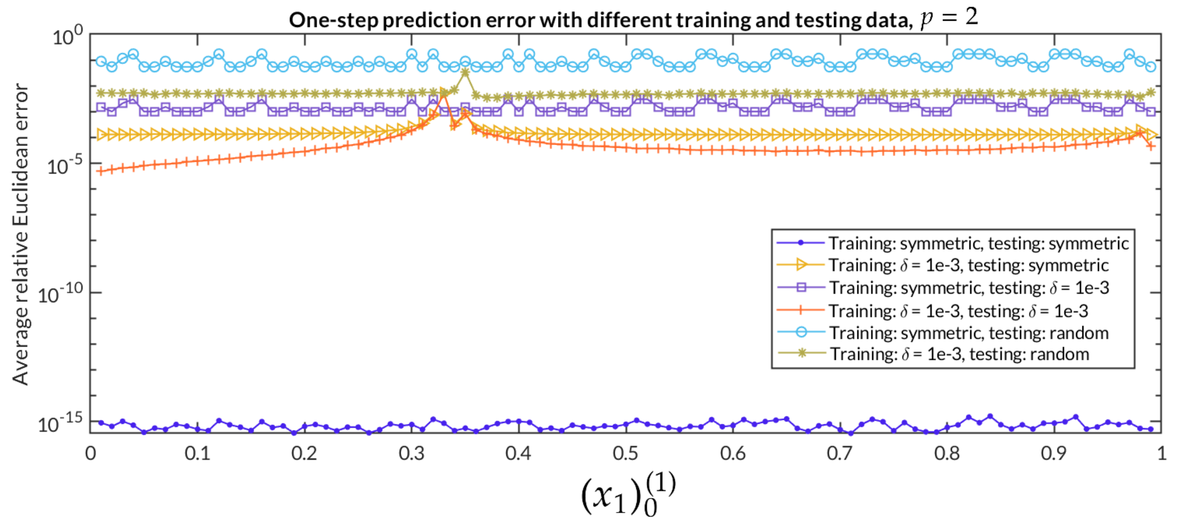
In summary, finding an analytical expression for the macrodynamics for non-complete networks requires further research. It seems that the initial hope that the macrodynamics can be described with an NAR model consisting of the basis functions used here and a memory depth of 2 was deceptive. Apparently, only for a small domain of initial conditions, yielding a small domain of trajectories, a simple



(a) From different non-symmetric initial conditions with different values for δ and $p = 2$.



(b) From different non-symmetric initial conditions with $\delta = 1e - 3$ and for different memory depths.



(c) From symmetric and non-symmetric ($\delta = 1e - 3$) initial conditions and memory depth $p = 2$ in SINAR. As testing trajectories, ones from symmetric, non-symmetric and randomly chosen initial conditions were used.

Figure 4.9: One-step prediction error of models produced with trajectories from varying initial conditions, different memory depths in SINAR and different testing trajectories.

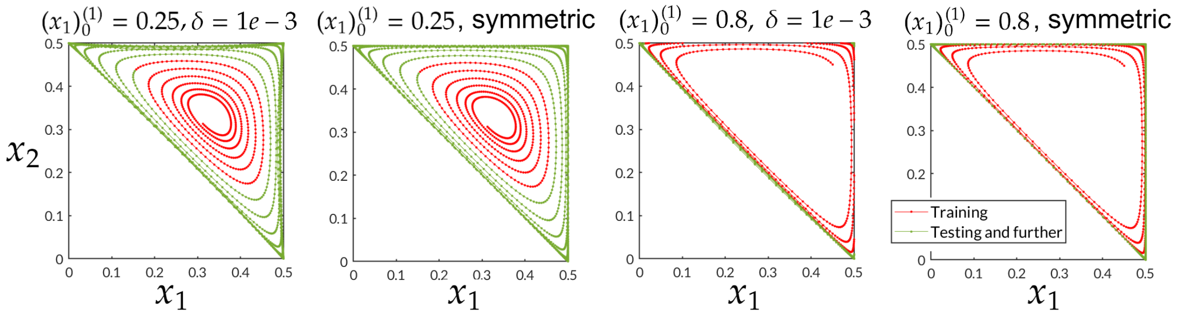


Figure 4.10: Trajectories with $(x_0^{(1)})_1 = 0.25$ and 0.8 for symmetric initial conditions and $\delta = 1e-3$. Only small differences depending on the initial conditions are apparent. For example, for $(x_0^{(1)})_1 = 0.25$ and $\delta = 1e-3$, the trajectory only touches the boundary of the depicted domain on the corners.

model is sufficient while generally a high memory depth is required to find even an approximation of the macrodynamics. Remembering that the dynamics investigated here are the expected macrodynamics of the agent-based model, this analysis should emphasise that for ABMs, parameter estimation can strongly depend on the initial conditions of the realisations.

4.3 Opinion Dynamics ABM II: Continuous Opinions

We will now introduce a second ABM which is constructed under modifications of ABM I. It assumes that opinions are continuous instead of discrete. The set of possible opinions for an agent is not longer given by $\{1, \dots, m\}$ but by the convex hull of a d-dimensional polytope with equidistant vertices $v_1, \dots, v_m$, i.e., by the area inside these vertices. The vertices denote the extremes in the opinion space. One could view them as the discrete opinions from ABM I while now space in between is introduced.

As in ABM I, at each time step every agent i is influenced by a neighbour with a given probability and changes its opinion into the direction of the opinion of the neighbour j. Both of these agents are assigned closeness measures to each vertex, given by a function $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^+$. The closeness to each vertex is computed by $K((X_t)_i, v_l)$ for $l = 1, \dots, m$ and the values are normalized by dividing by $\sum_{l=1}^{m} K((X_t)_i, v_l)$, yielding values $k_t(i) = [k_t(i,1), \dots, k_t(i,m)]$. One then assigns each of the agents to a vertex randomly drawn from the distributions $k_t(i)$ resp. $k_t(j)$, denoted by l_i and l_j. Then, as in ABM I, with a probability of α_{l_i,l_j}, agent i changes its opinion to

$$(X_{t+1})_i = (1 - \Delta t)(X_t)_i + \Delta t (X_t)_j \tag{4.29}$$

for a $\Delta t \in [0, 1]$.

The intuition behind this is that people can have opinions on multiple related topics and while they might not change their opinion from one extreme to the other, they might adjust their opinion towards the one of the person they are interacting with. Not all opinions are compatible with each other so that the likelihoods for adjusting one's opinion after interaction with another person depend on the opinion state of both persons. The reason this second ABM is introduced here is to demonstrate the need for memory terms on an example which comes from continuous space, potentially making it more realistic than ABM I. Moreover, in the next chapter we will introduce a novel method which models high-dimensional dynamics in a reduced way so that this ABM might be a good example for its applicability. Plus, as every field of research in dynamical systems modelling has its standard examples, such as the harmonic oscillator or the Lorenz-63 system, in the ABM community this ABM might be of interest to test and validate novel methods on.

In algorithmic form, ABM II reads

Algorithm 2: Agent-based opinion change model 2

```
1  Choose end time $T$, number of agents $N$, network adjacency matrix $A$,
     opinion change coefficients $\alpha_{m'm''}$, step size $\Delta t \in [0,1]$, initial opinions $X_0$,
     vertices $v_1, \dots, v_l$
2  for $t = 0, \dots, T$ do
3      for $i = 1, \dots, N$ do
4          for $l = 1, \dots, m$ do
5              Compute $k_t(i,l) = K((X_t)_i, v_l) / \sum_{k=1}^{m} K((X_t)_i, v_k)$
6          end
7      end
8      for $i = 1, \dots, N$ do
9          Draw $j$ from $\{j : A_{ij} = 1\}$ uniformly at random (Choose neighbour)
10         Draw $u_i, w_i^{(1)}, w_i^{(2)} \sim \mathcal{U}[0,1]$
11         Define $l_i$ so that $k_t(i,l_1) < w_i^{(1)} \leq k_t(i,l_1+1)$ (Draw from $k_t(i)$)
12         Define $l_j$ so that $k_t(j,l_1) < w_i^{(2)} \leq k_t(j,l_1+1)$ (Draw from $k_t(j)$)
13         If $u_i < \alpha_{l_i,l_j}$: $(X_{t+1})_i = (1-\Delta t)(X_t)_i + \Delta t (X_t)_j$ (Change opinion)
14     end
15 end
```

For ω_t we can make a straightforward definition as in Eq. (4.4) for ABM I, letting

$$\omega_t = [j_1, \dots, j_N, u_1, \dots, u_N, w_1^{(1)}, w_1^{(2)}, \dots, w_N^{(1)}, w_N^{(2)}] \tag{4.30}$$

with

$$(X_{t+1})_i = F(X_t, \omega_t)_i = \begin{cases} (1-\Delta t)(X_t)_i + \Delta t (X_t)_j & \text{if } u_i < \alpha_{l_i,l_j} \\ (X_t)_i & \text{otherwise.} \end{cases} \tag{4.31}$$

We choose as closeness measure $K((X_t)_i, v_l) = \exp(-\frac{1}{2}\frac{\|(X_t)_i - v_j\|_2^2}{0.05})$. We let $m = 3$ and define the vertices by $v_1 = [0,0]^T, v_2 = [1,0]^T$ and $v_3 = [0.5, \sqrt{0.75}]^T$, making the opinions two-dimensional and apart from each other by a distance of 1. The opinion change coefficients α are the same as the ones we used in ABM I. This defines a dynamical system in $\mathbb{R}^{2N}$.

As the observable we choose the average opinion across all agents, i.e.,

$$\phi(X_t) = \frac{1}{N}\sum_{i=1}^{N} X_t \in \mathbb{R}^2. \tag{4.32}$$

A complete network

At first, we again use a complete network with $N = 3000$ agents out of which two thirds have opinion equal to $v_1 = [0,0]^T$ and one sixth each has opinion v_2 and v_3. For the opinion change time step, we choose $\Delta t = 0.5$, so that if an agent is successfully influenced by its neighbour, its new opinion will be in the middle between their opinions.

As for ABM I, we attempt to model the ensuing macrodynamics with SINAR, using the same basis functions as before. In contrast to ABM I, we have not derived a closed form for the expected macrodynamics, leaving us with merely an educated guess for suitable basis functions. Since both ABMs are conceptually similar, the hope is that the same basis functions are still well-suited.

The results are shown in Figure 4.11. The agents behave quite similarly to each other (**I**,**II** left), yielding the macrostate to oscillate (**I**,**II** right).

The discovered Markovian models are unable to generate good prediction accuracy and a memory depth of 2 immediately yields a significant improvement (**III**). If the 60-step prediction error of a model was higher than 1.5, it was not included into the figure. We can see that most of the non-sparse models generated high errors, regardless of the memory depth. The sparse models, with $c = 0.2$, have significantly fewer non-zero coefficients, but yield sound accuracy which converges with memory.

Apparently, unlike in ABM I, the here defined macrodynamics do not admit a simple Markovian formulation, at least not with the given basis functions. Further tests using additional basis functions such as trigonometric ones and higher order monomials have not revealed any different outcome.

Two- and five-cluster networks

We now divide the $N = 3000$ agents into two respectively five equally-sized clusters. Again, between each pair of agents a link is installed with probability 10^{-4}.

For the two-cluster network, the initial opinions are such that in the first cluster two thirds start at v_1 (with the rest equally distributed to v_2 and v_3) and in the second

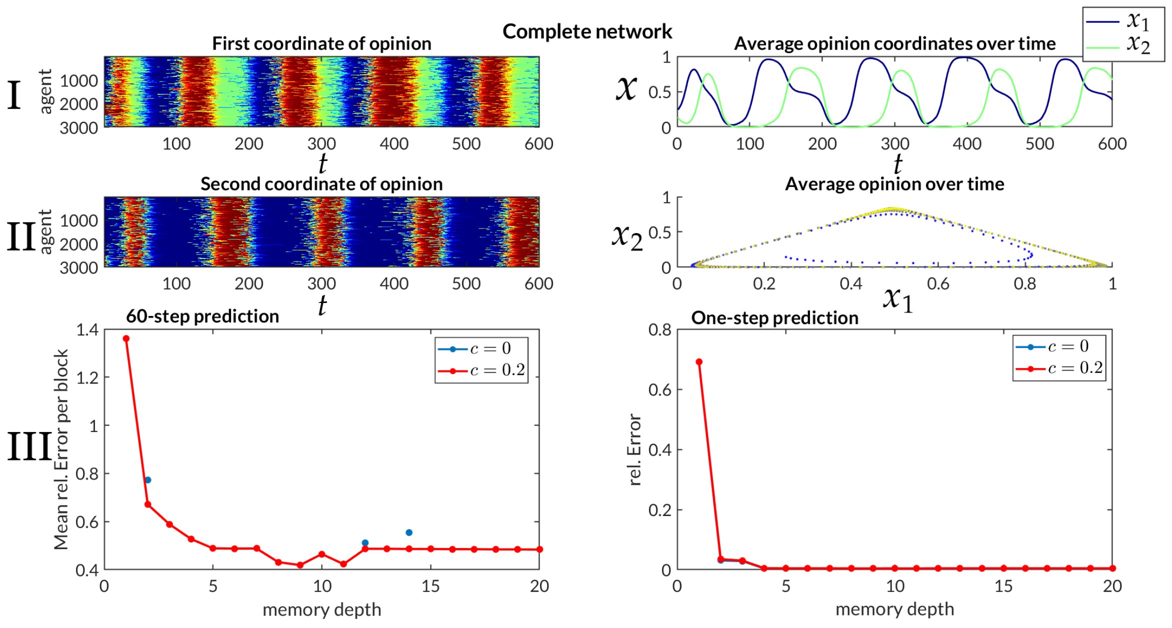


Figure 4.11: Results for the full network in ABM II: **I** and **II** show the evolution of the microdynamics (left) and macrodynamics (right). The two-dimensional macrostates are coloured according to time, developing from blue to yellow to illustrate the path of the macrostates in one realisation. **III** shows the 60- and one-step prediction error of the macrostates over memory depth using SINAR.

cluster, two-thirds start at v_2. In Figure 4.12 we can see that similarly to the two-cluster scenario in ABM I, the microdynamics run in parallel to each other for some time and then one cluster influences the other so that the micro- and macrodynamics become irregular (**I**,**II**).

Again, the accuracy of the models improves drastically with memory. For one-step predictions, already a memory depth of 2 gives a vast improvement over the Markovian models. In contrast to the full-network case, the non-sparse models produce good accuracy. This could indicate that in the full-network case, they overfit the training data, rendering them useless for the testing data. The sparse models are slightly worse this time.

For the five-cluster network, the initial opinions are such that agents are distributed onto the vertices of the triangle with different distributions per cluster, given by $[2/3, 1/6, 1/6]$, $[1/6, 2/3, 1/6]$, $[1/6, 1/6, 2/3]]$, $[0.45, 0.45, 0.1]$, $[0.45, 0.1, 0.45]$. The corresponding micro- and macrodynamics seem even more irregular. The findings regarding the model accuracies are analogous to the previous networks (Figure 4.13). Memory vastly improves the prediction quality. In this case, a memory depth of more than 2 does not give a large improvement for the 60-step error.

Considering the two-dimensional evolution of the average opinions, we can see some intricacies: for the complete network, the macrodynamics follow a path approximately on a triangle (the polytope spanned by the vertices). For two and five clusters, the macrostates from evolution follow different paths, covering different geometrical shapes (Figure 4.14). We can see that for the two clusters, generally in the long term the boundary of the polytope is covered with a smaller triangle in between, both connected with short transient phases. Tests have shown that this generally is the outcome of long-term trajectories. For the five-cluster network, more regions inside the triangle are covered and the trajectory is much more complex. The figures above showed that for the 60-step predictions, the error on the complete network was higher for the other two networks. This seems surprising since Figure 4.14 reveals the more complex and seemingly more nonlinear behaviour in the multi-cluster networks. Moreover, the fact that higher memory depth does not yield a large improvement compared to small memory depths indicates that one might need different nonlinear basis functions to uncover to macrodynamics. An investigation in this direction would require further research.

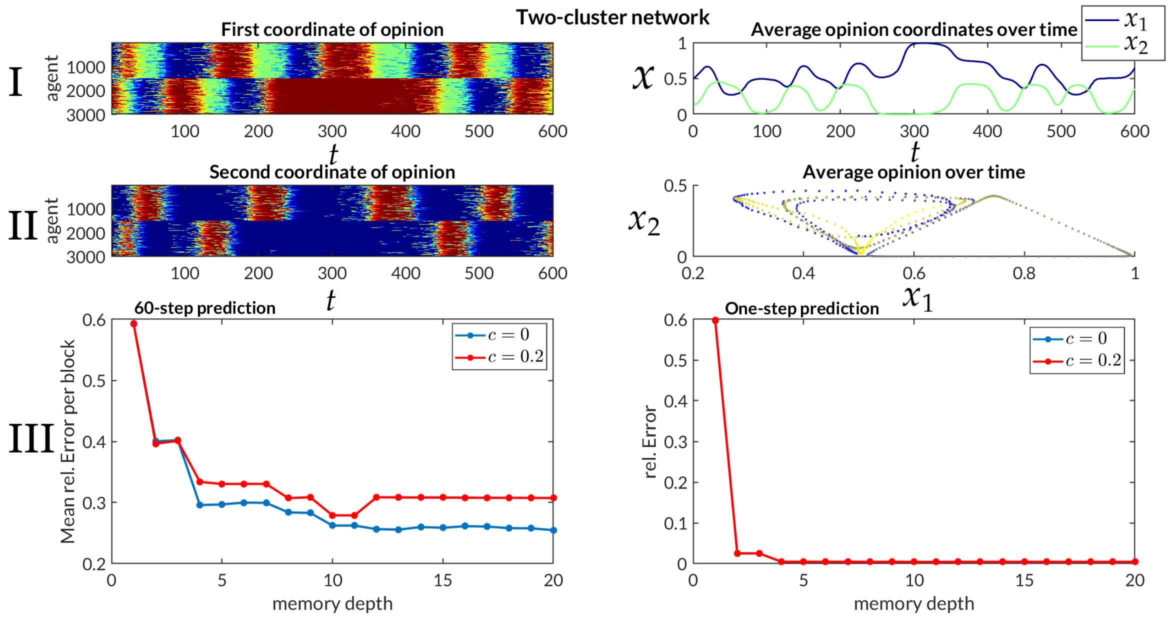


Figure 4.12: Results for the two-cluster network in ABM II: **I** and **II** show the evolution of one realisation of the microdynamics (left) and macrodynamics (right). **III** shows the 60- and one-step prediction error of the macrostates over memory depth using SINAR.

This analysis of the two ABMs signals the following: (1) in order to model macrodynamics of ABMs, one should generally incorporate memory terms into the model identification. This should generalize to complex real-world systems where many individual particles are involved and where the observable does not already capture all relevant information for its own evolution. (2) Sparse models can circumvent potential overfitting of data from stochastic ABM realisations. (3) The model identification can be strongly dependent on the training data and their initial conditions.

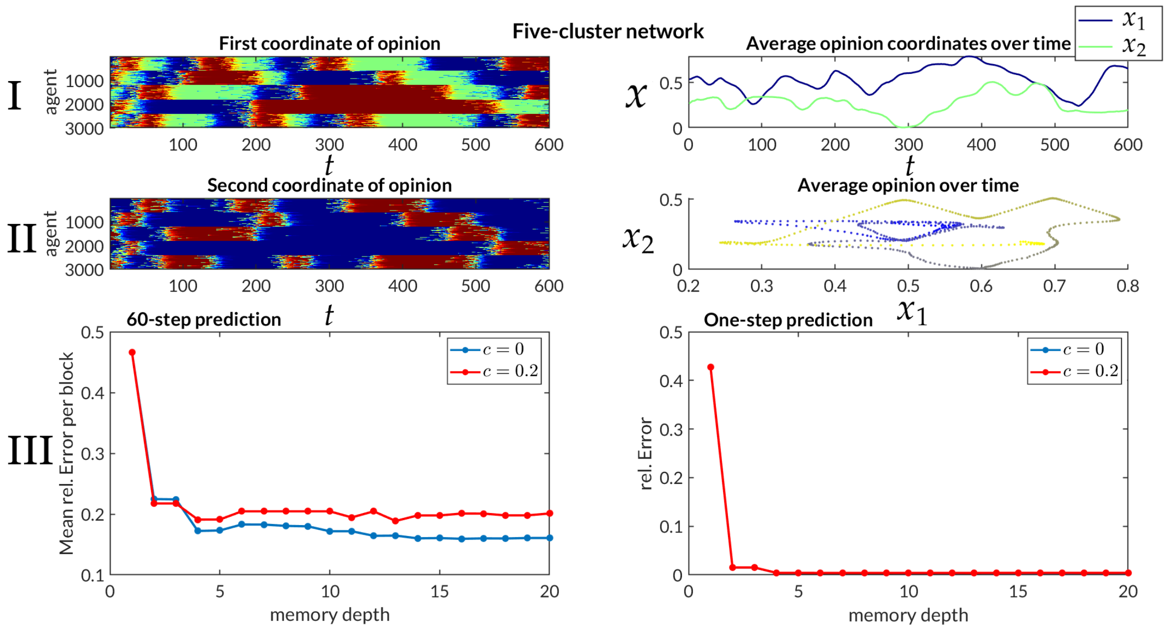


Figure 4.13: Results for the five-cluster network in ABM II: **I** and **II** show the evolution of one realisation of the microdynamics (left) and macrodynamics (right). **III** shows the 60- and one-step prediction error of the macrostates over memory depth using SINAR.

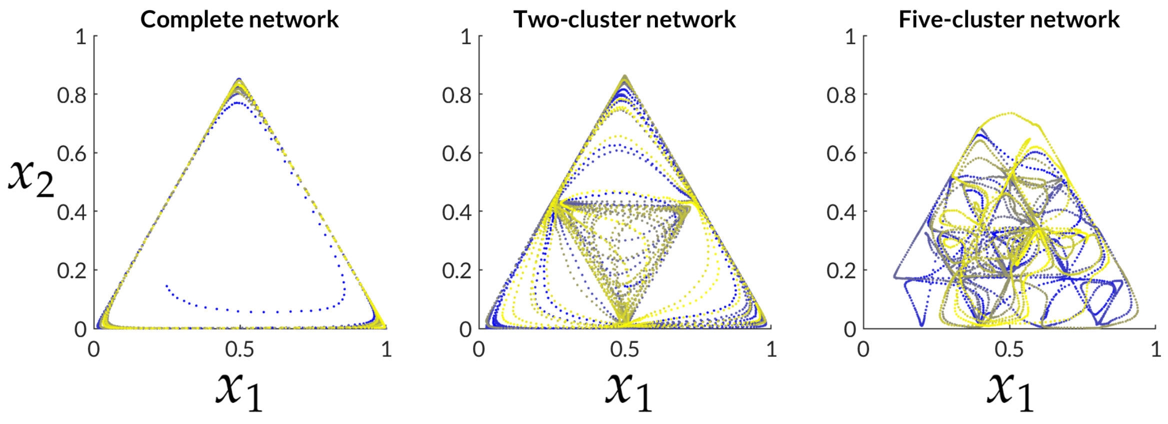


Figure 4.14: Macrodynamics from long-term realisation of ABM II for the complete, the two-cluster and the five-cluster network. The colors denote time evolution, running from blue to yellow.

Ensuing Research Questions

In the future, it should be interesting to see how strong the inclusion of memory is in practice to model observed social dynamics in the real world and which philosophical results can be derived from this. Furthermore, analyses of the macrodynamics dependent on the network structure underlying the microdynamics as in Section 4.2.5 could improve the understanding of how to model real-world macrodynamics if the structure of the network is at least approximately known. From a more theoretical point of view, it would be of great help if a precise mathematical generalization of ABMs could be developed and theoretically connected with MZ in a general form. This could in turn contribute to develop an intuition about which model class is suited to model the macrodynamics from complex social systems in the real world.

Memory SPA – a Novel Method for Dynamical Systems Modelling

In the previous chapters we have approached the problem of propagating an observable of a dynamical system over time from different view points: a theoretical one using Takens' Theorem and the Mori–Zwanzig formalism, a numerical one using different classes of methods to infer a model and in application to an agent-based model. In this chapter a novel method for dynamical systems modelling will be presented that is based on the same perspective: now, to fully accessible dynamics, we voluntarily deploy a specific low-dimensional observable, use memory to propagate it over time and relate predicted states of the observable back to predicted values of the actual state of the system.

The concept of projecting a dynamical system onto lower-dimensional coordinates, the *dimension reduction*, is not new. There exist various methods such as the linear Singular Value Decomposition as used in [BBP$^+$16], neural network autoencoders as in [CLKB19] or the detection of a low-dimensional manifold with nonlinear reaction coordinates as in [BKK$^+$17]. All these methods, however, require the dynamics to admit a bijective transformation to a low-dimensional subspace or manifold.

For the method that is introduced, points are projected onto a low-dimensional *polytope*. Then inspired from Takens' Theorem, it uses memory to counteract the potential loss of information on system states that the projection results in. On the polytope the projected dynamics are estimated and its states are transformed back to the original state space.

For this we use the recently introduced method Scalable Probabilistic Approximation [GPNH20] (SPA). SPA is a versatile method, suited for (1) optimal discretization of the state space using probabilistic representations, including low-dimensional ones, and (2) constructing a probabilistic linear model between two variables X and Y. For the latter aspect, SPA is especially suited in comparison to other methods when X and Y are high-dimensional, stochastic and little intuition about their relation exists. Dynamics constitute a special case, mapping X_t to X_{t+1}. SPA has achieved strong performance in both representing high-dimensional data using low-dimensional coordinates and predicting states of variables. This could be achieved in both purely mathematical and real-world contexts of varying dimension and with only a low amount of data available. Moreover, its numerical complexity is comparable to the simple clustering method K-means [Mac67]. For dynamics, SPA is can be suitable but generally not for long-term forecasts of nonlinear dynamics, as will be shown shortly. For this reason, in this chapter a new method called memory SPA (mSPA) is introduced which uses the SPA representations of points and is suited for nonlinear dynamics.

5.1 Scalable Probabilistic Approximation (SPA)

The main aim of SPA is to perform the transformation of points into a new coordinate system. It then constructs a mapping between these new coordinates for two generally different variables. To this end, it tackles two distinct optimization problems, one for the representation of points and one for the mapping, which can also be simultaneously solved in one problem. We refer to them as SPA I and SPA II.

5.1.1 SPA I: Transformation of Coordinates

Given data $\mathbf{X} = [X_0, \dots, X_T] \in \mathbb{R}^{d \times T+1}$ with points $X_t \in \mathcal{M} \subset \mathbb{R}^d$, choose $K \in \mathbb{N}$ and find matrices $\Sigma \in \mathbb{R}^{d \times K}$ and $\Gamma \in \mathbb{R}^{K \times T+1}$ which fulfil

$$[\Sigma, \Gamma] = \underset{\Sigma^*, \Gamma^*}{\arg\min} \|\mathbf{X} - \Sigma^* \Gamma^*\|_F,$$

subject to,

$$\Sigma := [\sigma_1 | \cdots | \sigma_K] \in \mathbb{R}^{d \times K}, \quad \Gamma := [\gamma_1 | \cdots | \gamma_T] \in \mathbb{R}^{K \times T+1},$$

with

$$\sum_{k=1}^{K} (\gamma_t)_k = 1 \text{ and } (\gamma_t)_k \geq 0 \text{ for all } t \in \{0, \dots, T\} \text{ and } k \in \{1, \dots, K\}. \tag{SPA I}$$

With this, SPA I seeks a matrix Σ so that the data can be well approximated by convex combinations of its columns. This is strongly related to Principal Component Analysis [JC16] where one seeks a similar decomposition but without constraining it to convex combinations. Note that the set of all convex combinations of the columns of Σ, the *convex hull*, denoted by

$$\mathcal{M}_\Sigma := \operatorname{conv}(\Sigma) = \{X \in \mathbb{R}^d : \exists \gamma \text{ with } \sum_{k=1}^{K} \gamma_k = 1, \gamma_k \geq 0 \text{ s.t. } X = \Sigma\gamma\} \tag{5.1}$$

is a convex polytope with vertices given by $\sigma_1, \dots, \sigma_K$. A vector γ corresponding to a state X is then called the *barycentric coordinate* of X. The dimension of a polytope with K vertices is at most $K-1$ so that the dimension of the data determines whether SPA I can be solved without loss, i.e., so that $X_t \in \mathcal{M}_\Sigma \ \forall t$, yielding that each point is exactly represented by barycentric coordinates, or not. The following lemma details this relation.

Lemma 5.1. For $\mathbf{X} \in \mathbb{R}^{d \times T+1}$, there exist $\Sigma \in \mathbb{R}^{d \times K}, \Gamma \in \mathbb{R}^{K \times T+1}$ so that (SPA I) can be solved exactly if and only if $K > \operatorname{rank}(\mathbf{X})$.

Proof. If $\operatorname{rank}(\mathbf{X}) = r$ then the data lie in a bounded subset of an affine-linear r-dimensional subspace of $\mathbb{R}^d$. If $K > r$ the polytope can naturally be chosen to be at least r-dimensional and big enough to contain this subset in its interior.

If $K \leq r$, then the polytope can only cover a subset of an $(r-1)$-dimensional subspace so that there exists a point X_t that lies outside of the polytope, yielding a positive projection loss between X_t and $\Sigma\gamma_t$. □

Note that depending on the relation between K and d, the interpretations of SPA I which present itself strongly differ. Assuming that $K < d$, one achieves a dimension reduction. SPA I then seeks a polytope to which states are projected orthogonally with minimal total error:

Lemma 5.2. Let $\mathcal{M}_\Sigma$ be a polytope with vertices given by the columns of Σ, let $X \notin \mathcal{M}_\Sigma$ and let γ be the minimizer of $\|X - \Sigma\gamma\|_2$. Then $\Sigma\gamma$ is the orthogonal projection of X onto $\mathcal{M}_\Sigma$.

Proof. By construction, $\Sigma\gamma$ minimizes the Euclidean distance of points in $\mathcal{M}_\Sigma$ to X and hence must be the unique orthogonal projection of X onto $\mathcal{M}_\Sigma$. □

Such a projection is always unique but it is not injective. Multiple points can be collapsed onto the same point on the polytope, yielding a potential loss of information of states.

If $K > d$, the barycentric coordinates can be interpreted as closeness or affiliation measures of a state to each vertex. One does not reduce the dimension of states but represents them in dependence of landmark points. Therefore, SPA I can be seen as generating a *fuzzy discretization* of the state space similarly as in PCCA+ [DW05]. With a suitable choice of Σ, (SPA I) can be solved without loss. However, a point $\mathbf{X}$ might admit non-unique representations as barycentric coordinates (see Example 5.1).

Example 5.1. Let $X = [0,0]$. Let $\Sigma = \begin{bmatrix} -2 & 2 & 1 & -1 \\ 1 & 1 & -1 & -1 \end{bmatrix}$. Let $\gamma = (3/8, 1/8, 1/2, 0)^T$ and $\tilde{\gamma} = (1/8, 3/8, 0, 1/2)^T$. Then $X = \Sigma\gamma = \Sigma\tilde{\gamma}$ (see Figure 5.1).

5.1.2 SPA II: Dynamics Reconstruction

SPA contains a second step which estimates a model between two potentially different variables. To this end, SPA I is solved for two variables, X and Y, giving barycentric coordinates of dimensions K_X and K_Y with respect to vertices Σ_X and Σ_Y. Then an optimal column-stochastic matrix is determined which maps barycentric coordinates of one variable to those of the other.

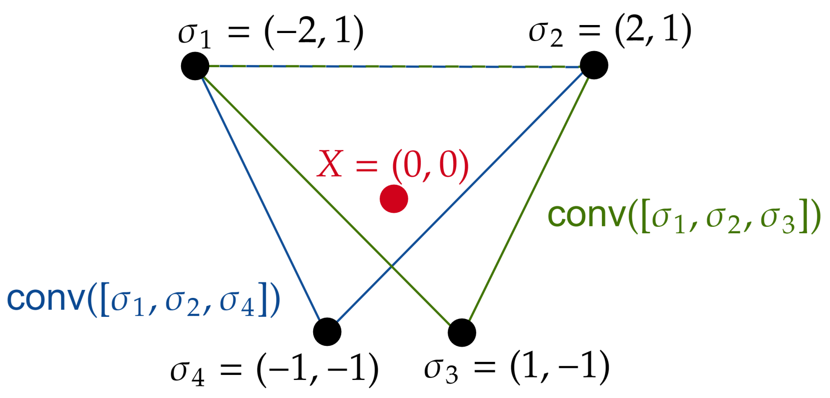


Figure 5.1: Non-uniqueness of SPA I if $K > \text{rank}(\mathbf{X}) + 1$. X lies inside the convex hull of two sets of vertices and can hence be represented exactly with two different barycentric coordinates.

We consider the special case of dynamics, i.e., $X = X_t$ and $Y = X_{t+1}$, having barycentric coordinates with respect to the same vertices Σ. In order to estimate the dynamics in the barycentric coordinates one solves SPA II, which reads

$$\begin{aligned} \Lambda = \underset{\Lambda^* \in \mathbb{R}^{K \times K}}{\arg\min} \, \| [\gamma_1 | \cdots | \gamma_T] - \Lambda^* [\gamma_0 | \cdots | \gamma_{T-1}] \|_F, \\ \text{subject to} \qquad\qquad\qquad\qquad\qquad\qquad \\ \Lambda \geq 0 \text{ and } \sum_{k=1}^{K} \Lambda_{k,\bullet} = 1. \end{aligned} \tag{SPA II}$$

Λ is column-stochastic, meaning that it fulfils the conditions imposed on it in (SPA II). Therefore, $\Lambda\gamma$ is always a barycentric coordinate, too, i.e., $\Sigma\Lambda\gamma \in \mathcal{M}_\Sigma$ as the following lemma asserts.

Lemma 5.3. Let $\gamma \in \mathbb{R}^K$ be a stochastic vector, i.e., with $\|\gamma\|_1 = 1, \gamma_k \geq 0 \; \forall k$. Let $\Lambda \in \mathbb{R}^{K \times K}$ be a column-stochastic matrix. Then $\Lambda\gamma$ is a column-stochastic vector.

Proof.

$$\sum_{j=1}^{K} (\Lambda\gamma)_j = \sum_{i=1}^{K} \sum_{j=1}^{K} \Lambda_{ij}\gamma_j = \sum_{j=1}^{K} \gamma_j \sum_{i=1}^{K} \Lambda_{ij} = \sum_{j=1}^{K} \gamma_j = 1. \tag{5.2}$$

□

Therefore, Λ is a linear propagator of the dynamics on the polytope.

This small observation on the product of a column-stochastic matrix and a stochastic vector is fundamental in the theory of Markov Chains. In essence, SPA II constructs a Markov State Model (see, e.g., [Sar11, HP18]) along the vertices where a state can be interpreted as the strength of affiliation of a point to each vertex. In [GPNH20] this probabilistic interpretation is emphasised and illustrated in detail.

For $K \leq d$ this model should generally be imprecise for two reasons: it works only on the reduced coordinates which might not admit identification with the states in $\mathbb{R}^d$ and it is linear, making it ill-suited if the dynamics are not approximately linear, too. Note that Markov State Models typically work with a fine enough discretization of the state space to generate accuracy.

For $K > d$, however, Λ can be a good approximation of the dynamics. Firstly, there is generally no projection loss and, secondly, SPA has been shown in [GPNH20] to give strong accuracy for nonlinear dynamics if K is large enough. For this it requires the dynamics F to act similarly on a vertex as on points close to it. Let $\Lambda_{|i}$ denote the ith column of Λ. Then SPA yields the forward model

$$\gamma_{t+1} = \Lambda_{|1}(\gamma_t)_1 + \cdots + \Lambda_{|K}(\gamma_t)_K. \tag{5.3}$$

Thus, the dynamics at one point are approximated by a mixture – more precise, a convex combination – of the approximated dynamics at the nearby vertices.

Remark. The notation of SPA I and SPA II was not taken from [GPNH20], where SPA rather denotes only SPA I but with an elegant way of augmenting it so that both problems that are here denoted by SPA I and SPA II can be solved simultaneously (see Theorem 2 in [GPNH20]). This comes at the cost of adding a constraint to the search for Σ. Still, both approaches produce linear models.

5.1.3 SPA in the Context of Dynamical Systems

Let us formalize the intuition of SPA in the case of dynamics. As explained above, for $K \leq d$ we can generally derive a unique representation of points in barycentric coordinates but with a loss of information about the states. For $K > d$, there is no loss of information but generally no unique representation. We therefore make the following definition.

Definition 5.4 (Polytope projection function)**.** Let $\mathcal{M} \subset \mathbb{R}^d$ be a compact manifold, $X \in \mathcal{M}$ and let γ be a K-dimensional barycentric coordinate. The mapping of X into a polytope with vertices given by the columns of $\Sigma \in \mathbb{R}^{d \times K}$ is defined as follows:

Case 1: $K \leq d$:

$$\rho_\Sigma(X) := \underset{\gamma^* \in \mathbb{R}^K}{\arg\min} \|X - \Sigma\gamma^*\|_2, \qquad \text{s.t. } \gamma^*_\bullet \geq 0, \quad \|\gamma^*\|_1 = 1. \tag{5.4}$$

Case 2: $K > D$:

$$\begin{aligned} \rho_\Sigma(X,\gamma) &:= \underset{\gamma^* \in \mathbb{R}^K}{\arg\min} \|\gamma - \gamma^*\|_2 \\ \text{s.t. } \gamma^* &= \underset{\gamma' \in \mathbb{R}^K}{\arg\min} \|X - \Sigma\gamma'\|_2 \text{ with } \gamma'_\bullet, \gamma^*_\bullet \geq 0 \text{ and } \|\gamma'\|_1, \|\gamma^*\|_1 = 1. \end{aligned} \tag{5.5}$$

For $K \leq d$, $\rho_\Sigma(X)$ is given by the barycentric coordinate of the orthogonal projection of X onto $\mathcal{M}_\Sigma$. For $K > d$, we choose $\rho_\Sigma(X)$ as the barycentric coordinate which defines X that is closest to a *reference coordinate* γ. With this, ρ_Σ is well-defined if there is a unique closest suitable barycentric coordinate to the reference coordinate.

Now, for a dynamical system given by the function $F : \mathcal{M} \to \mathcal{M}$ and for $K > d$, let us use as reference coordinate the respective previous barycentric coordinate, i.e., define $\gamma_t = \rho_\Sigma(X_t, \gamma_{t-1})$. Then subsequent steps of the projected dynamics will always make the smallest step that is necessary with the aim of generating smooth dynamics. This also enables us to derive,

$$\gamma_t = \rho_\Sigma(X_t, \gamma_{t-1}) = \rho_\Sigma(F(X_{t-1}), \gamma_{t-1}) = \rho_\Sigma(F(\Sigma\gamma_{t-1}), \gamma_{t-1}) =: \mathbf{v}(\gamma_{t-1}) \qquad (5.6)$$

by substituting X_t by $F(X_{t-1})$ and using the representation of X_{t-1} dependent on γ_{t-1}. We obtain a closed dynamical system on the barycentric coordinates.

This shows that we can construct well-defined smooth dynamics on the polytope. Furthermore, the SPA II propagator Λ is a linear approximation of $\mathbf{v}$, leaving us with the options for the long-term behaviour explained in Chapter 2. Since $\Lambda\gamma$ is a barycentric coordinate, too, the dynamics are kept inside the polytope. Note that the substitution of X_{t-1} by $\Sigma\gamma_{t-1}$ is not possible if ρ is not bijective so that for $K \leq d$ we generally cannot formulate closed dynamics in γ. For high-dimensional systems, choosing $K > d$ would not generate an economic representation of points and thus potentially entail enormous difficulty in estimating the dynamics. Therefore, in the following we will use memory to derive a reformulation of the dynamics on the polytope and present a method which estimates these dynamics.

Again, note that for short-term predictions and mappings from one variable to another, the observations regarding long-time dynamical behaviour are unimportant and the strengths of SPA hold up.

5.2 mSPA: Extending SPA to Model Nonlinear Dynamics

In this section, a novel method is presented which estimates the dynamics on the lower-dimensional barycentric coordinates. It has recently been submitted by the author of this thesis together with co-authors [WKSS21]. Most of the theoretical results explained here can also be found in the article.

In the previous section, we have seen that if $K \leq d$ there is a loss of information when projecting points to the polytope $\mathcal{M}_\Sigma$. This prohibited us from defining a closed dynamical system on the polytope. Viewing the projection to the polytope, ρ_Σ, simply as an observable of the dynamical system F, it should now be natural to deploy Takens' Theorem to derive a one-to-one mapping from states of the dynamics to sequences of barycentric coordinates. We therefore define the *product polytope*

$$\mathcal{M}_\Sigma^p := \underbrace{\mathcal{M}_\Sigma \times \ldots \mathcal{M}_\Sigma}_{p \text{ times}} \tag{5.7}$$

and the delay-coordinate map

$$\Phi_{\rho_\Sigma,F,p} : \mathcal{M} \to \mathcal{M}_\Sigma^p, \quad \Phi_{\rho_\Sigma,F,p}(X_{t-1}) := (\gamma_{t-1}^T, \ldots, \gamma_{t-p}^T)^T. \tag{5.8}$$

We will later on use $\Phi_{\rho_\Sigma,F,p}$ to modify Eq. (5.6) suitably to derive memory-exhibiting dynamics $\mathbf{v}$ on $\mathcal{M}_\Sigma$. For now, a method is presented which uses Takens' Theorem to model nonlinear dynamics in $\mathcal{M}_\Sigma$. Since the theory is meant to justify the method, it will be easier to follow the former if the latter has already been introduced.

5.2.1 The mSPA Method

We define a specific function which transforms sequences of barycentric coordinates into a high-dimensional polytope from where we then find a mapping back into the polytope $\mathcal{M}_\Sigma$. With this we construct a dynamical system along the reduced, barycentric, coordinates. We call this function the *path affiliation function* Ψ. It is defined as follows:

Definition 5.5 (Path affiliation function)**.** Let ρ_Σ be the orthogonal projection of points $X \in \mathcal{M} \subset \mathbb{R}^d$ to a polytope $\mathcal{M}_\Sigma$ with vertices given by the columns of $\Sigma \in \mathbb{R}^{d\times K}$. For a dynamical system $X_{t+1} = F(X_t)$ denote $\gamma_t := \rho_\Sigma(X_t)$. Then the function Ψ, defined as

$$\begin{gathered} \Psi^p(\gamma_{t-1}, \ldots, \gamma_{t-p})_i := (\gamma_{t-1})_{i_1} \cdot (\gamma_{t-2})_{i_2} \cdots (\gamma_{t-p})_{i_p} \\ J(i) = [i_1, \ldots, i_p], \end{gathered} \tag{5.9}$$

is the path affiliation function of barycentric coordinates in the polytope $\mathcal{M}_\Sigma$ for the dynamics F.

The entries of $\Psi^p(\gamma_{t-1}, \ldots, \gamma_{t-p})$ are given by all products of combinations of entries from γ_{t-1} to γ_{t-p}. The function J is an ordering of the set $\{1, \ldots, K\}^p$ (it does not induce a loss of generality). The path affiliation function maps into $\mathbb{R}^{K^p}$.

For example, with $K = 2$ and $p = 2$,

$$\Psi^2(\gamma_t, \gamma_{t-1}) = \begin{pmatrix} (\gamma_t)_1 \cdot (\gamma_{t-1})_1 \\ (\gamma_t)_1 \cdot (\gamma_{t-1})_2 \\ (\gamma_t)_2 \cdot (\gamma_{t-1})_1 \\ (\gamma_t)_2 \cdot (\gamma_{t-1})_2 \end{pmatrix}. \tag{5.10}$$

The path affiliation function can also be written in the form of a sequence of outer products,

$$\begin{aligned}\Psi^p(\gamma_{t-1},\cdots,\gamma_{t-p}) &:= \gamma_{t-1}\otimes\cdots\otimes\gamma_{t-p},\\ u\otimes v &:= \mathrm{vec}(uv^T).\end{aligned} \tag{5.11}$$

The intuition behind the path affiliations is the following: they measure the affiliation of the path of projected points to each possible path of length p along the vertices. This measuring is given by the product of the affiliations to each vertex of each point in the path. Interpreting the barycentric coordinates as an affiliation of a point to the vertices, the affiliation of (X_{t-2}, X_{t-1}) with the *path* (σ_i, σ_j) is quantified as $(\gamma_{t-2})_i(\gamma_{t-1})_j$.

As with the delay-coordinate map in Chapter 1, we write Ψ if the superscript p is of no importance.

The image of Ψ on the product polytope is a unit simplex with K^p vertices, meaning that it consists of stochastic vectors again, as the following two propositions assert:

Proposition 5.6. For $K, p < \infty$, let $\gamma_1,\ldots,\gamma_p$ be a sequence of barycentric coordinates. Then for $\psi := \Psi^p(\gamma_1,\ldots,\gamma_p)$ it holds,

$$\sum_{i=1}^{K^p}\psi_i = 1, \quad \psi_i \geq 0 \text{ for all } i = 1,\ldots,K^p. \tag{5.12}$$

Proposition 5.7. Let $K, p < \infty$. Let $y \in \mathbb{R}^{K^p}$ with $\sum_{i=1}^{K^p} y_i = 1$ and $y_i \geq 0$ for all $i = 1,\ldots,K^p$. Then there exists a sequence of barycentric coordinates $\gamma_1,\ldots,\gamma_p$ so that $y = \Psi^p(\gamma_1,\ldots,\gamma_p)$.

From Proposition 5.6 it follows that no points outside of the unit simplex can be constructed with Ψ^p restricted to the product polytope. Proposition 5.7 implies that Ψ^p is surjective onto this unit simplex. We denote the unit simplex with K^p vertices as the *path affiliation polytope*. For convenience of the reader, the proofs for the results in this section will be given at its end.

Furthermore, Ψ^p is injective on $\mathcal{M}_\Sigma^p$:

Proposition 5.8. For $K, p < \infty$, Ψ^p as defined in Eq. (5.9) is injective on the product polytope $\mathcal{M}_\Sigma$.

With the path affiliation function, we can map paths of barycentric coordinates into a K^p-dimensional space in a nonlinear way. Similarly to SPA II, we want to map points from the image of the path affiliation function into the polytope $\mathcal{M}_\Sigma$ to construct subsequent barycentric coordinates of the dynamics. Specifically, we want $\Psi^p(\gamma_{t-1},\ldots,\gamma_{t-p})$ to map to γ_t. In order to guarantee that Ψ maps to $\mathcal{M}_\Sigma$, we determine a column-stochastic matrix $\hat{\Lambda}^M \in \mathbb{R}^{K \times K^p}$ so that

$$\gamma_t = \hat{\Lambda}^p \, \Psi^p(\gamma_{t-1}, \dots, \gamma_{t-p}). \tag{5.13}$$

The hope is that the path affiliation function provides sufficient nonlinearity to accurately predict nonlinear dynamics using Eq. (5.13).

Analogously to SPA II, we therefore formulate the following optimization problem to obtain a suitable matrix $\hat{\Lambda}^p$. Let $[\Sigma, \Gamma]$ be the solution to (SPA I) for data points $X_0, \dots, X_T \in \mathbb{R}^d$. We further denote from now on

$$\psi^p_{t-1} := \Psi^p(\gamma_{t-1}, \dots, \gamma_{t-p}). \tag{5.14}$$

Then we define as the mSPA problem,

$$\begin{aligned} \hat{\Lambda}^p := \underset{\hat{\Lambda}^* \in \mathbb{R}^{K \times K^p}}{\arg\min} & \left\| [\gamma_p | \cdots | \gamma_T] - \hat{\Lambda}^* [\psi^p_{p-1} | \cdots | \psi^p_{T-1}] \right\|_F, \\ \text{subject to} & \\ \hat{\Lambda}^p \geq 0 \text{ and } & \sum_{k=1}^{K} \hat{\Lambda}^p_{k,\bullet} = 1. \end{aligned} \tag{mSPA}$$

From here onwards, we call $\hat{\Lambda}^p$ the *mSPA propagator*. Note that for $p = 1$, $\Psi^1(\gamma) = \gamma$ and (mSPA) is equivalent to (SPA II). Furthermore, for $p > 1$ we can always define an mSPA model which is equivalent to SPA II:

Proposition 5.9. For every column-stochastic matrix $\Lambda \in \mathbb{R}^{K \times K}$ and every $p \geq 1$, there exists a column-stochastic matrix $\hat{\Lambda}^p$ so that $\Lambda \gamma_{t-1} = \hat{\Lambda}^p \Psi^p(\gamma_{t-1}, \dots, \gamma_{t-p})$ for all $\gamma_{t-1}, \dots, \gamma_{t-p}$.

We can straightforwardly close the dynamical system in Eq. (5.13) by augmenting the mSPA propagator, denoting $\hat{\gamma}_{t-1} := [\gamma^T_{t-1}, \dots, \gamma^T_{t-p}]^T$ and defining

$$\hat{\gamma}_t = \Theta^p \Psi^p(\hat{\gamma}_{t-1}) = \Theta^p \psi^p_{t-1}, \tag{5.15}$$

where $\Theta^p = \begin{bmatrix} \hat{\Lambda}^p \\ E \end{bmatrix} \in \mathbb{R}^{Kp \times K^p}$. $E \in \{0,1\}^{Kp \times K^p}$ is a matrix that copies $\gamma_{t-1}, \dots, \gamma_{t-p+1}$ from ψ^p_{t-1} into $\hat{\gamma}_t$ (see [WKSS21] for details).

Since Ψ^p is bijective between the product polytope and the unit simplex in K^p, we can define an equivalent dynamical system on the path affiliations by

$$\psi^p_t = \Psi^p(\Theta^p \psi^p_{t-1}). \tag{5.16}$$

In summary, we use the path affiliation function to measure closeness of a sequence of points inside the polytope to each sequence of vertices. We then construct the mSPA propagator to derive a stable dynamical system.

Example 5.2. Lorenz-96 model Let us consider the Lorenz-96 model, a time-continuous dynamical system with states $X = (X_1, \dots, X_d)^T \in \mathbb{R}^d$, given by the equations

$$\frac{\mathrm{d}X_i}{\mathrm{d}t} = (X_{i+1} - X_{i-2})X_{i-1} - X_i + G \quad \text{for } i = 1, \dots 5, \tag{5.17}$$

with initial state $X_0 = (3.8, 3.8, 3.8, 3.8, 3.8001)$ and parameter $G = 3.8$. We generate a trajectory of length 2500 time steps with step size 0.1. We discard the first 500 time steps to give the trajectory time to converge towards its attractor which resembles a figure-eight. In order to investigate mSPA on low-dimensional data, we consider only the X_1 and X_4 coordinates, giving data $x_1, \dots, x_T \in \mathbb{R}^2$ (see Figure 5.2). We then solve SPA I with $K = 3$, meaning that it can be solved without loss to derive a trajectory of barycentric coordinates. On these we train mSPA models with varying memory depth with the first 1000 time steps of the data and consider their resulting forward simulations.

We consider the forecasting error for the short term and compare the shape of the long-term predictions with the original attractor. For the k-step forecasting error, we define a forecasting length of $L = 30$, select N points from the testing data $(x_{t_1}, \dots, x_{t_N}) = (x_{1001}, x_{1031}, \dots, x_{1991})$ and from these create forward simulations $\tilde{x}_{t_i+1}, \dots, \tilde{x}_{t_i+L}$, starting with the previous p points. The k-step forecasting error is then defined as

$$\mathcal{E}_k = \frac{1}{N} \sum_{t=1}^{N} \|\tilde{x}_{t_i+k} - x_{t_i+k}\|_2. \tag{5.18}$$

We can see in Figure 5.3 that the k-step forecasting error drastically decreases starting at a memory depth of 5 and becoming even better with $p = 6$. For the long-term predictions, $p = 1$ generates a fixed point, $p = 5$ produces a slight resemblance of the attractor shape but no figure-eight as desired, $p = 6$ gives an approximate reproduction of the shape and with $p = 7$ mSPA is able to precisely recreate the attractor. Note that the Lorenz-96 studied here is 5-dimensional, so that according to Takens' Theorem, we should need at most a memory depth of 5 to create topologically equivalent dynamics since the barycentric coordinates with $K = 3$ contain two independent coordinates, giving $2 \cdot 5 + 1 = 11$. The improvement with higher memory depths seems to stem from the richer set of basis functions given by the path affiliation function.

Remark. In all numerical examples for mSPA, the data were normalized into the unit cube $[-1, 1]^d$ by a linear transformation to give each dimension equal weight.

5.2.2 mSPA in the Context of Dynamical Systems

Now that the mSPA method is introduced, let us investigate its theoretical foundations in light of Takens' Theorem. It can be shown that the dynamical system Eq. (5.16) on the path affiliations is in fact generically topologically equivalent to the

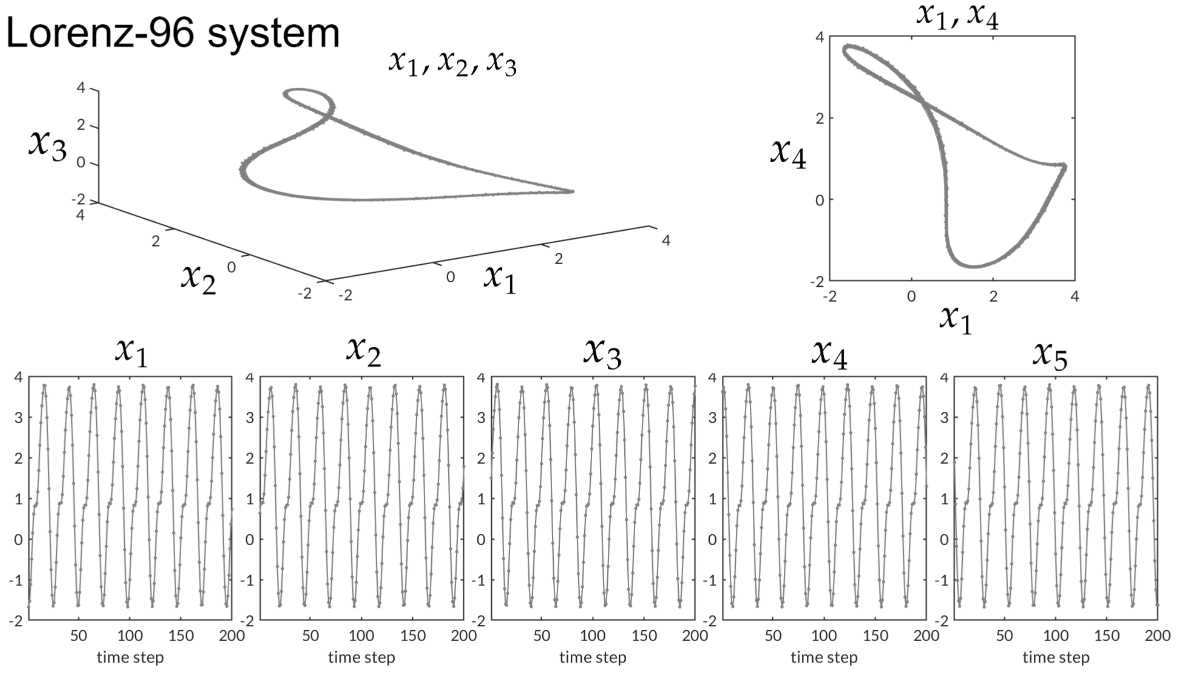


Figure 5.2: Visualisation of the attractor of the Lorenz-96 system with 5 dimensions and $G = 3.8$. We can see that the system is periodic. Projecting onto the x_1 and x_4 coordinate, however, results in an overlap of the trajectory.

original dynamical system F. As explained at the end of Section 5.1.3, in order to define an equivalent dynamical system on the barycentric coordinates in the case $K > d$, we could use an injection from the original states to the barycentric coordinates. For $K \leq d$ such an injection does generally not exist but the corresponding delay-coordinate map could be one. In order to show when this is the case, let us introduce the following variant of Takens' Theorem from [Rob05].

Theorem 5.10 (Delay embedding Theorem for Lipschitz maps [Rob05])**.** Let A be a compact subset of a Hilbert space H with upper box-counting dimension d_{box}, which has thickness exponent σ, and is an invariant set for a Lipschitz map $F : H \to H$. Choose an integer $p > 2(d_{box} + \sigma)$, and suppose further that the set A_p of p-periodic points of F satisfies $d_{box}(A_p) < p/(2 + \sigma)$. Then a prevalent set of Lipschitz maps $\phi : H \to \mathbb{R}$ make the delay-coordinate map $\Phi_{\phi,F,p} : H \to \mathbb{R}^p$ injective on A.

The contribution of Theorem 5.10 lies in extending the class of observables for which the delay-coordinate is injective from differentiable to Lipschitz continuous functions. In [DHvMZ15] it is observed that the Theorem can be generalized to multivariate observables with a modified value for the memory depth, given by the following: if the observable is m-dimensional, consisting of m independent scalar-valued observables then the memory depth should be $p > \frac{2(d_{box}+\sigma)}{m}$. In our case, the barycentric coordinates are K-dimensional but since their sum always has to be 1, they denote $K - 1$ independent observables so that a memory depth of $p > \frac{2(d_{box}+\sigma)}{K-1}$ should be enough.

We can observe that the projection ρ_Σ is suited for the application of Theorem 5.10:

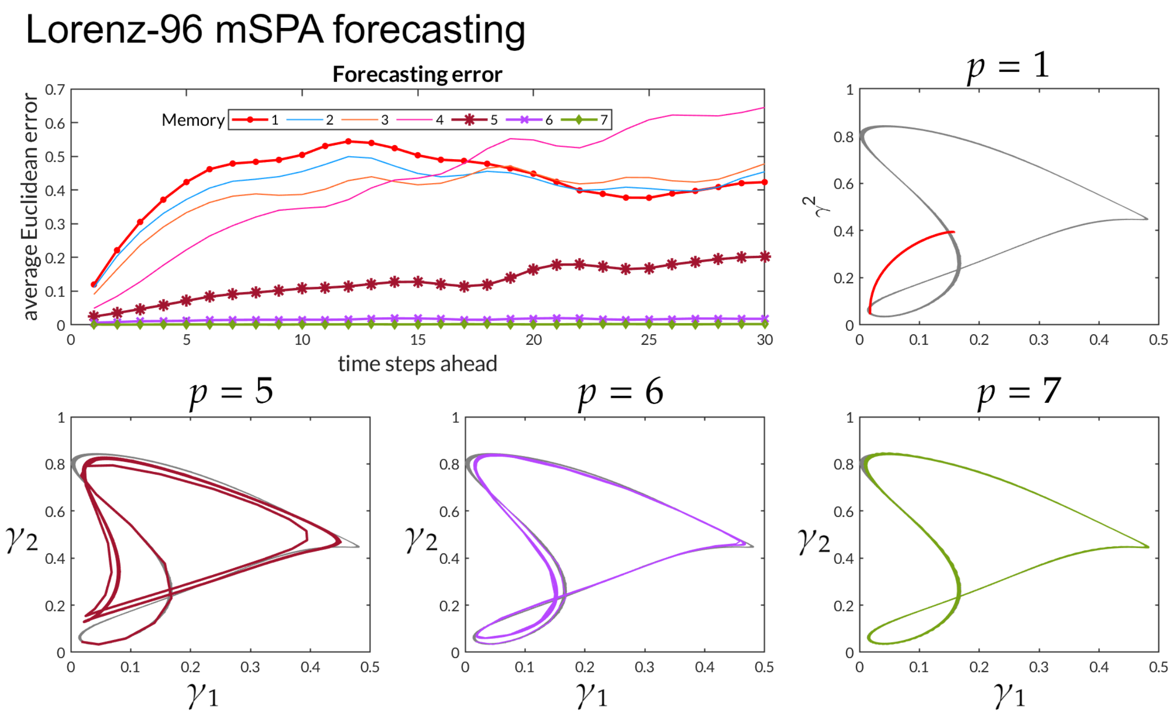


Figure 5.3: Forecasting results for the Lorenz-96 system using mSPA with memory $p = 1, \dots, 7$. We can see that for $p \leq 4$ the forecasting error is comparably high for 15 time steps ahead. It is significantly better with $p \geq 5$. For the long-term predictions, with $p = 6$ the figure-eight shape of the attractor can be reconstructed, although more precisely with $p = 7$.

Lemma 5.11. The function ρ_Σ as defined in Eq. (5.4) is Lipschitz continuous.

This enables us to show that the delay-coordinate map of ρ_Σ defined in Eq. (5.8) is typically injective:

Theorem 5.12. Let A and F be given as in Theorem 5.10. Let the projection ρ_Σ from Eq. (5.4) be inside the prevalent set of Lipschitz maps from Theorem 5.10. Then for a sufficient memory depth p, the delay-coordinate map $\Phi_{F,\rho_\Sigma,p}$ is an injection from A to $\mathcal{M}_\Sigma^p$.

This result implies that the sequences of barycentric coordinates should be equivalent representations of states of the original system. This result can be extended to the path affiliations. To this end, we can first state the following corollary.

Corollary 5.13. The function $\Psi^p \circ \Phi_{F,\rho_\Sigma,p}$ is an injection from A into the path affiliation polytope.

We are now in position to state a theorem which yields the equivalence between the dynamics on the path affiliation polytope and the original dynamics.

Theorem 5.14. Let A be given as in Theorem 5.10 and let $F : A \to A$ generate a discrete-time dynamical system on A, denoted by $X_{t+1} = F(X_t)$. Let the projection ρ_Σ onto barycentric coordinates of the SPA I polytope be inside the prevalent set of maps from Theorem 5.10. Then there exists an operator $\mathbf{v}$ so that for a path affiliation vector $\psi_{t-1}^p = \Psi^p(\gamma_{t-1}, \dots, \gamma_{t-p})$, it holds $\mathbf{v}(\psi_{t-1}^p) = \rho_\Sigma(F(X_{t-1})) = \gamma_t$.

Since this theorem is the main result of the Section 5.2.2, its proof is given here directly.

Proof. We can rewrite

$$\gamma_t = \rho_\Sigma(X_t) = \rho_\Sigma(F(X_{t-1})). \tag{5.19}$$

Since Φ is injective, thus bijective on its image, this is equal to

$$\rho_\Sigma(F(X_{t-1})) = [\rho_\Sigma \circ F \circ \Phi^{-1}](\gamma_{t-1}, \dots, \gamma_{t-p}). \tag{5.20}$$

The injectivity of Ψ^p and the resulting existence of its inverse on the path affiliation polytope gives

$$\underbrace{[\rho_\Sigma \circ F \circ \Phi^{-1} \circ (\Psi^p)^{-1}]}_{=:\mathbf{v}}(\psi_{t-1}^p). \tag{5.21}$$

□

In mSPA, we therefore approximate the operator $\mathbf{v}$ by the matrix $\hat{\Lambda}^p$.

Remark. An alternative way to approximate $\mathbf{v}$ is sketched in the Appendix of [WKSS21]. There instead of a column-stochastic matrix a neural network which outputs stochastic vectors is used.

Moreover, note that the dynamics on the path affiliations are topologically equivalent to the original dynamics F,

$$\psi_t^p = [\Psi^p \circ \Phi_{F,\rho_\Sigma,p} \circ F \circ \Phi_{F,\rho_\Sigma,p}^{-1} \circ (\Psi^p)^{-1}](\psi_{t-1}^p). \tag{5.22}$$

5.2.3 Proofs from Section 5.2

As promised, now are delivered the proofs of the previous results.

Proof of Proposition 5.6 (path affiliations are stochastic vectors)

Let $\gamma_t, \dots, \gamma_{t-p+1}$ be stochastic vectors, i.e., all entries are at least 0 and sum up to 1. Then by induction on p it can be shown that that the path affiliation vectors are stochastic vectors, too.

Let u, v be arbitrary stochastic vectors. Then

$$\sum_i \Psi^2(u,v) = \sum_{i_1,i_2} u_{i_1} v_{i_2} = \sum_{i_1} u_{i_1} \sum_{i_2} v_{i_2} = \sum_{i_1} u_{i_1} = 1, \tag{5.23}$$

so that

$$\sum_{J(i)=[i_1,i_2]} \Psi^2(\gamma_{t-1},\gamma_{t-2})_i = \sum_{i_1,i_2} (\gamma_{t-1})_{i_1} (\gamma_{t-2})_{i_2} = \sum_{i_1} (\gamma_{t-1})_{i_1} \sum_{i_2} (\gamma_{t-2})_{i_2} = \sum_{i_1} (\gamma_{t-1})_{i_1} = 1. \tag{5.24}$$

For every $p > 1$, we find

$$\Psi^{p+1}(\gamma_{t-1},\dots,\gamma_{t-p},\gamma_{t-p-1}) = \Psi^2(\Psi^p(\gamma_{t-1},\dots,\gamma_{t-p}),\gamma_{t-p-1}) \tag{5.25}$$

(at least up to permutation of the entries).

By the induction hypothesis that $\Psi^p(\gamma_{t-1},\dots,\gamma_{t-p})$ is a stochastic vector, the induction start in Eq. (5.23) yields that $\Psi^{p+1}(\Psi^p(\gamma_{t-1},\dots,\gamma_{t-p}),\gamma_{t-p-1})$ is a stochastic vector, too.

Proof of Proposition 5.7 (the image of Ψ^p is a polytope)

Each point $\Psi^p(\gamma_1,\dots,\gamma_p)$ is a K^p-dimensional stochastic vector, so that it is contained in a unit simplex with K^p vertices. These vertices are the K^p-dimensional unit vectors. It is left to show that each point in this polytope can be generated by application of Ψ^p to a point $[\gamma_1,\dots,\gamma_p] \in \mathcal{M}^p_\Sigma$.

Let $[u_1,\dots,u_{K^p}] \in \mathcal{M}^p_\Sigma$. We then need to prove the existence of $\gamma_1,\dots,\gamma_p$ so that

$$\begin{bmatrix} u_1 \\ \vdots \\ u_{K^p} \end{bmatrix} = \begin{bmatrix} (\gamma_1)_1 \cdots (\gamma_p)_1 \\ \vdots \\ (\gamma_1)_K \cdots (\gamma_p)_K \end{bmatrix}. \tag{5.26}$$

For this, several relations between the entries of $\gamma_1,\dots,\gamma_p$ need to be satisfied: the first entry of the above equation yields for the value for $(\gamma_1)_1$ that

$$\begin{aligned} (\gamma_2)_1 \cdots (\gamma_p)_1 = \frac{u_1}{(\gamma_1)_1}, \quad (\gamma_2)_1 \cdots (\gamma_p)_2 = \frac{u_2}{(\gamma_1)_1}, \quad \dots \quad (\gamma_2)_K \cdots (\gamma_p)_K = \frac{u_{K^{p-1}}}{(\gamma_1)_1} \\ \Rightarrow (\gamma_1)_1 \underbrace{\left((\gamma_2)_1 \cdots (\gamma_p)_1 + \cdots + (\gamma_2)_K \cdots (\gamma_p)_K\right)}_{=\sum_j \Psi^{p-1}(\gamma_2,\dots,\gamma_p)_j = 1} = u_1 + \cdots + u_{K^{p-1}}. \end{aligned} \tag{5.27}$$

This can be done equivalently for all entries of $\gamma_1,\dots,\gamma_p$. For all entries of γ_1, the sets of terms required to construct it are disjoint (for $(\gamma_1)_1$, $u_1,\dots,u_{K^{p-1}}$ are required, for $(\gamma_1)_2$ one needs $u_{K^{p-1}+1},\dots,u_{2K^{p-1}}$,...). Moreover, the entries of u sum to one by definition so that this holds for γ_1 as desired. This can analogously be observed for

$\gamma_2, \ldots, \gamma_p$, where again the entries from u needed are disjoint between the entries of a γ_i.

With this, we can see that for each vector u in the path affiliation polytope, we can find, even construct, points $\gamma_1, \ldots, \gamma_p \in \mathcal{M}_\Sigma$ so that the path affiliation function applied to them gives u. Therefore, the path affiliation function Ψ^p is surjective onto the unit simplex with K^p vertices. The entries of Ψ^p always sum up to 1 (see Proposition 5.6) so that no other points lie in its image.

Proof of Proposition 5.8 (Ψ^p is injective)

Let Ψ^p be defined as in Eq. (5.10) on all stochastic vectors $\gamma \in \mathbb{R}^K$. Since the concatenation of injective functions is injective again it is sufficient to show that one outer product as defined in Eq. (5.11) is injective on the set of the barycentric coordinates:

Let $u, \tilde{u} \in \mathbb{R}^n$ and $v, \tilde{v} \in \mathbb{R}^m$ be given. If $u \otimes v = \tilde{u} \otimes \tilde{v}$ it holds that $\tilde{u}_1 = (u_1 v_1)/\tilde{v}_1$. This yields that the condition $u_1 v_i = \tilde{u}_1 \tilde{v}_i$ yields for all $i = 2, \ldots, K$

$$u_1 v_i = \tilde{u}_1 \tilde{v}_i = (u_1 v_1)\tilde{v}_i/\tilde{v}_1 \text{ so that } \tilde{v}_i = v_i(\tilde{v}_1/v_1). \tag{5.28}$$

As a consequence, $\tilde{v}$ must be a scaled version of v with an arbitrary scaling constant given by $\tilde{v}_1/v_1$. Therefore $\tilde{u}$ can only be a scaled version of u. Since the entries of all barycentric coordinates must sum to 1 this means that $u = \tilde{u}$ and $v = \tilde{v}$.

Thus, Ψ^p is injective. As consequence, the inverse of Ψ^p exists on its image.

Proof of Proposition 5.9 (relation between mSPA and SPA II)

Let Λ be a minimizer mSPA and let $p \geq 1$. Then for all paths $i_1, \ldots, i_p$, define $\hat{\Lambda}^p \in \mathbb{R}^{K \times K^p}$ so that $\hat{\Lambda}^p_{ji} = \Lambda_{ji_1}$ where i is the index in the vector of path affiliations ψ_t corresponding to the path $i_1, \ldots, i_p$. This means that for each sequence of p indices, $\hat{\Lambda}^p_{ij}$ is equal to the value in the jth column of Λ that corresponds to the first entry of this sequence. Then from

$$\begin{aligned}&\sum_{i_2,\ldots,i_p} (\gamma_t)_{i_1} (\gamma_{t-1})_{i_2} \cdots (\gamma_{t-p+1})_{i_p} \\ =&(\gamma_t)_{i_1} \sum_{i_2,\ldots,i_p} (\gamma_{t-1})_{i_2} \cdots (\gamma_{t-p+1})_{i_2} = (\gamma_t)_{i_1},\end{aligned} \tag{5.29}$$

it follows that

$$\begin{aligned}(\hat{\Lambda}^p \psi_t^p)_j &= \sum_{J(i)=[i_1,\dots,i_p]} \hat{\Lambda}_{ji}^p (\psi_t^p)_i \\ &= \sum_{J(i)=[i_1,\dots,i_p]} \hat{\Lambda}_{ji}^p (\gamma_t)_{i_1} (\gamma_{t-1})_{i_2} \cdots (\gamma_{t-p+1})_{i_p} \\ &= \sum_{i_1} \Lambda_{ji_1} (\gamma_t)_{i_1} \sum_{i_2,\dots,i_p} (\gamma_t)_{i_1} (\gamma_{t-1})_{i_2} \cdots (\gamma_{t-p+1})_{i_p} \\ &= \sum_{i_1} \Lambda_{ji_1} (\gamma_t)_{i_1} = (\Lambda \gamma_t)_j.\end{aligned} \tag{5.30}$$

Therefore, a $\hat{\Lambda}^p$ defined as above is equivalent to applying a solution of the SPA II problem. As a direct consequence, the training error of mSPA is always bounded from above by the training error of the solution of the standard SPA II problem.

Proof of Lemma 5.11 (ρ_Σ is Lipschitz continuous)

By construction, the polytope $\mathcal{M}_\Sigma$ is closed and convex. The orthogonal projection onto such a set is always Lipschitz continuous with Lipschitz constant 1.

Proof of Theorem 5.12 ($\Phi_{F,\rho_\Sigma,p}$ is injective)

By Lemma 5.11, ρ_Σ is Lipschitz continuous, mapping from A to $\mathcal{M}_\Sigma$, while Φ maps from A to $\mathcal{M}_\Sigma^p$. Then the claim follows directly from Theorem 5.10 and the assumption that ρ_Σ is inside the prevalent set of Lipschitz maps.

Proof of Corollary 5.13 ($\Psi^p \circ \Phi_{F,\rho_\Sigma,p}$ is injective)

We map a point $X \in A$ to $\Phi_{F,\rho_\Sigma,p}$ and apply Ψ^p. The result lies inside the path affiliation polytope by definition of Ψ^p. Φ^p is injective by Theorem 5.12. Ψ^p is injective by Proposition 5.8. The concatenation of injective functions is injective again so that $\Psi^p \circ \Phi_{F,\rho_\Sigma,p}$ is injective on A.

5.2.4 Using mSPA to Model Systems in the Full Space

We have seen how mSPA can model the evolution of the barycentric coordinates from SPA I. The question naturally arises: how to transform the barycentric coordinates back into the original state space? For $K > d$, the answer to this question is simply by multiplication with Σ. For $K \leq d$ however, ρ_Σ is not injective and hence admits no simple back-transformation. For this reason we utilize the same ideas as in mSPA to compute an additional mapping from the path affiliation polytope to the original space. Again, from Takens' Theorem, or rather Theorem 5.14, directly results an injective mapping

$$X_t = \underbrace{[\Phi^{-1}_{F,\rho_\Sigma,p} \circ (\Psi^p)^{-1}]}_{=:\mathbf{w}}(\psi^p_t). \tag{5.31}$$

The task is now to approximate this function.

This is done by defining an additional polytope, called the *lifting polytope*, arising from SPA I with K' vertices Σ' and barycentric coordinates γ'. K' should be chosen so that the projection error is low or, if $K' > d$, equal to 0. We then solve

$$\begin{aligned} \hat{\Lambda}'^p &= \arg\min_{\Lambda^*} \|[\gamma'_p|\cdots|\gamma'_T] - \Lambda^*[\psi_p|\cdots|\psi_T]\|_F, \\ &\text{subject to} \\ \hat{\Lambda}'^p &\geq 0 \text{ and } \sum_{k=1}^{K} \hat{\Lambda}'^p_{k,\bullet} = 1. \end{aligned} \tag{SPA Lifting}$$

The low-dimensional polytope defined previously is in this context called the *learning polytope* since the dynamics are learned in it. Their states are lifted to the original state space.

Clearly, there exist various ways to approximate $\mathbf{w}$. The reason why this particular one is proposed is the following: if the data in the full state space admit a low-dimensional representation, e.g., approximately lying inside a lower-dimensional subspace, the task of finding a direct linear mapping from ψ_t to X_t, e.g., by (SPA Lifting) without the constraints, would be badly conditioned, yielding error-prone or unstable models. This also became apparent when examining this way in practice. The intermediate step of the lifting polytope therefore serves as a preconditioning step.

The complete mathematical forward model of the dynamics we derive thus is given in the following way:

$$\begin{aligned} \psi^p_t &= \Psi^p(\gamma_t, \ldots, \gamma_{t-p+1}) \\ X_t &= \Sigma'\hat{\Lambda}'^p\psi^p_t \\ \gamma_{t+1} &= \hat{\Lambda}^p\psi^p_t. \end{aligned} \tag{5.32}$$

A conceptualization of the *learning and lifting* approach is shown in Figure 5.4. Moreover, an algorithmic form of the sequence of modelling steps to approximate dynamics is given in Algorithm 3. The colors in the algorithm and figure classify steps into the groups projection with SPA I transformation (green), learning and prediction of the dynamics (orange) and lifting (purple).

Remark. When discussing optimal parameters for the delay-coordinate map in Chapter 1, we found that the time lag, respectively sampling rate, between points can severely affect the geometry of the image of the delay-coordinate map. For small time lags, subsequent states can be strongly correlated so that the image of the delay-

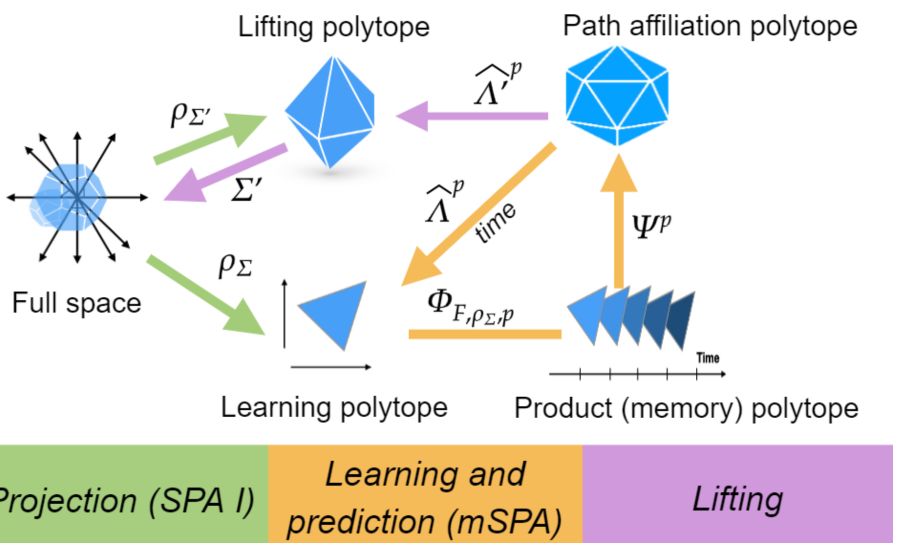


Figure 5.4: After performing dimension reduction with SPA I, we use mSPA to approximate the dynamics on the learning polytope. From the path affiliation polytope, points are lifted to the full state space using the lifting polytope.

coordinate map is close to the low-dimensional subspace given by

$$\{[\phi(X), \phi(F^{-1}(X)), \dots, \phi(F^{-p+1}(X))] \text{ s.t. } \phi(X) = \cdots = \phi(F^{-p+1}(X))\}. \tag{5.33}$$

In order to unfold the geometry of the true system well, it is therefore not always advisable to use a small time lag. However, modelling complex dynamics with a large time steps is often a too challenging problem. We therefore distinguish between the *forward time step* τ_{forward} – typically small – and a *memory time step* τ_{memory}. We then construct $\Phi_{F,\phi,p,\tau_{\text{memory}}}$ and map $\Phi_{F,\phi,p\tau_{\text{memory}}}(X_t)$ to $X_{t+\tau_{\text{forward}}}$. For mSPA, this yields that we compute path affiliations $\psi_t = \Psi^p(\gamma_t, \dots, \gamma_{t-(p-1)\tau_{\text{memory}}})$ and solve the optimization problem

$$\begin{aligned} &\hat{\Lambda}^p = \arg\min_{\Lambda^*} \|[\gamma_{p+\tau_{\text{forward}}}|\cdots|\gamma'_T] - \Lambda^*[\psi_p|\cdots|\psi_{T-\tau_{\text{forward}}}]\|_F, \\ &\text{subject to} \\ &\qquad \hat{\Lambda}^p \geq 0 \text{ and } \sum_{k=1}^{K} \hat{\Lambda}^p_{k,\bullet} = 1. \end{aligned} \tag{5.34}$$

This is illustrated in Figure 5.5. Fortunately, this procedure can be justified with

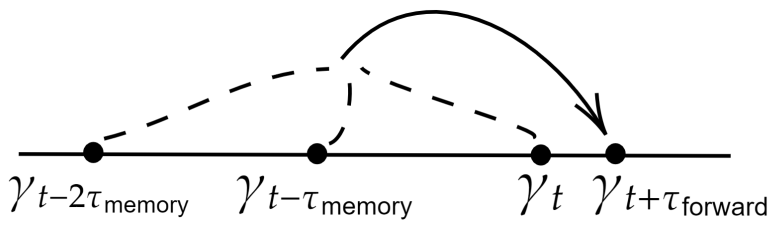

$\gamma_{t-2\tau_{\text{memory}}}$
$\gamma_{t-\tau_{\text{memory}}}$
γ_t
$\gamma_{t+\tau_{\text{forward}}}$

Figure 5.5: Illustration of the concept of a forward and a memory time step.

Algorithm 3: Numerical scheme

```
Input: data points $X_1, \dots, X_T \in \mathbb{R}^d$, $K, K', p \in \mathbb{N}$
Learning of operators:
1  Solve (SPA I) for $K$ to obtain $[\Sigma, \Gamma]$   (learning polytope)
2  Solve (SPA I) for $K'$ to obtain $[\Sigma', \Gamma']$   (lifting polytope)
3  Solve (mSPA) for $\Gamma$ to obtain $\hat{\Lambda}^p$   (learn propagator on learning polytope)
4  Solve (SPA Lifting) to obtain $\hat{\Lambda}'^p$ (learn map from learning to lifting polytope)
Prediction :
5  For starting values $\gamma_1, \dots, \gamma_p$ on learning polytope compute $\psi_p^p$ as in
     Eq. (5.10)
6  for $t = p + 1 : T_{end}$ do
7      $\gamma_t \leftarrow \hat{\Lambda}^p \psi_{t-1}^p$   (Propagation on learning polytope as in Eq. (5.13))
8      $\psi_t^p \leftarrow \Psi^p(\gamma_t, \dots, \gamma_{t-p+1})$   (Computation of path affiliation)
9      $\gamma_t' \leftarrow \hat{\Lambda}'^p \psi_t^p$   (Mapping to lifting polytope)
10     $X_t \leftarrow \Sigma' \gamma_t'$   (Lifting to original space)
```

Theorem 5.14 analogously as the definition of the dynamics on the path affiliations simply by considering Eq. (5.22), replacing F with $F^{\tau_{\text{forward}}}$ and $\Phi_{F,\rho_\Sigma,p,\tau_{\text{memory}}}$. The crucial point to observe is that the delay-coordinate map is an injection between *spaces* and not times, almost regardless of the time lag. We can therefore compose it together with a suitable injective function, say F, which happens to also represent a dynamical system, and preserve the injectivity, thereby defining a dynamical system.

In [WKSS21] are shown several numerical examples of the learning and lifting approach with mSPA, including a comparison to AR and SINDy models in both the short and the long term. We do not provide a detailed comparison here but rather show results of the mSPA results from these experiments to illustrate the capacity of this method. Additional details on some of the examples can be found in [WKSS21]. Afterwards, we will apply mSPA to ABM II from the previous chapter.

Example 5.3. Kuramoto-Sivashinsky equation The Kuramoto-Sivashinsky equation (KS) denotes a one-dimensional fourth-order partial differential equation defined in the 1970s which often serves as an example for complex dynamical systems (see [LL21]). We consider the following variant,

$$u_t + 4u_{xxxx} + 16u_{xx} + 8(u^2)_x = 0, \quad t \geq 0 \tag{5.35}$$

for $u \in [0, 2\pi]$ with periodic boundary conditions. The domain is now discretized into 100 equidistant grid points and the PDE is numerically integrated using a fourth-order exponential time differencing Runge-Kutta method with a time step of 0.001.

As initial values we choose $u(x) = 0.0001\cos(x)(1+\sin(x))$. We integrate until a time of $T = 8$ and discard the data until $T = 4$ because of an apparent strong dependence on initial values in the beginning. This gives the trajectory of a 100-dimensional dynamical system with 4000 time steps. We use the first 3000 for training of mSPA.

We can see that the KS PDE with the chosen parameters describes a travelling wave (see Figure 5.6, top left). For SPA I, we choose $K = 3$ to project the data onto a triangle and $K' = 8$ since this gives a low relative projection error of approximately 0.04. We then apply mSPA with $p = 3,\dots,6$ and find that we can well recreate the dynamics in the long term starting with $p = 4$. Over time, the prediction error in the barycentric coordinates and the full state space stays in the same order of magnitude for long-term predictions on the testing data (Figure 5.6, right). The travelling wave shape of the trajectory is also recreated with mSPA (Figure 5.6, bottom left). The frequency of the travelling was not precisely met, however. This could not be remedied by training on longer time series data, either.

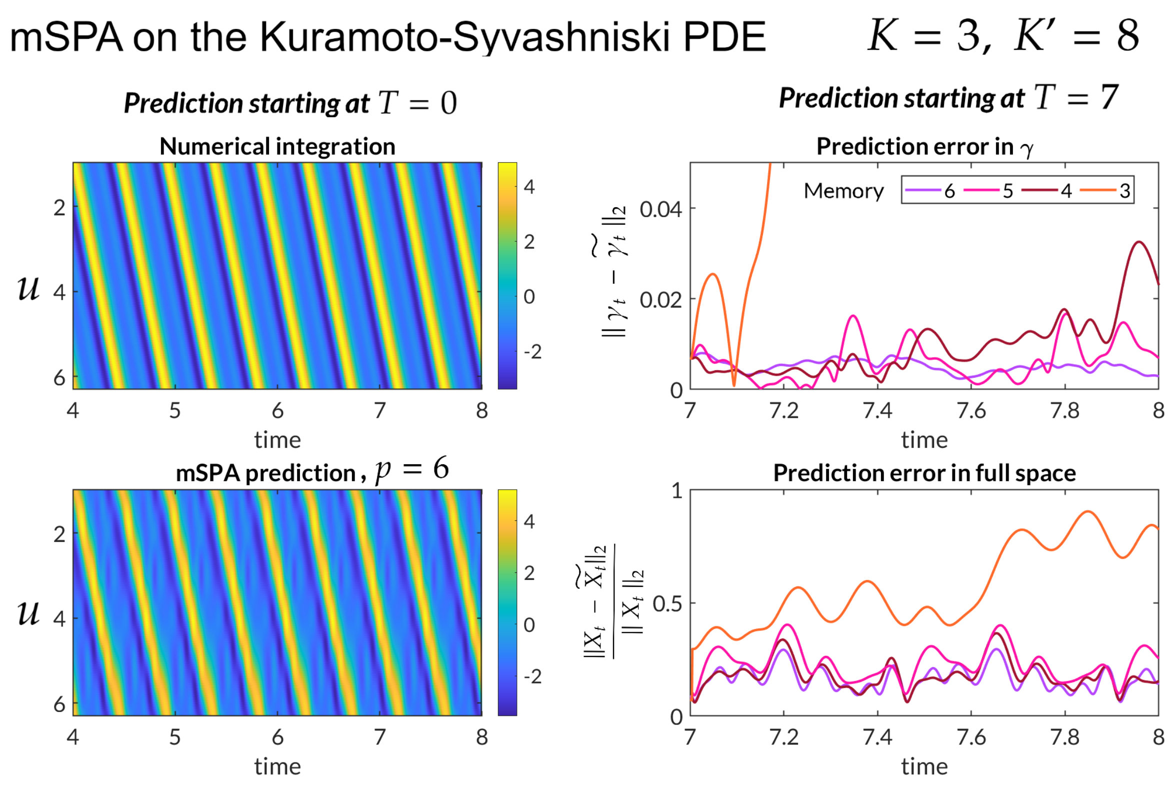


Figure 5.6: Top left: numerical solution of the KS PDE (5.35). Bottom left: mSPA prediction with $K = 3, K' = 8, p = 6$. Right: forecasting error over time in barycentric coordinates (top; absolute Euclidean error) and full space (bottom; relative Euclidean error).

Example 5.4. Chua circuit As a second example we consider the Chua circuit. Originally published in the 1980s in [CKM86,Wu87], Chua circuits describe models of an electric circuit with chaotic behaviour. We consider the following three-dimensional variant:

$$\begin{aligned}\dot{X}_1 &= \alpha X_2 - \mu_0 X_1 - \frac{\mu_1}{3} X_1^3\\ \dot{X}_2 &= X_1 - X_2 + X_3\\ \dot{X}_3 &= -\beta X_2\\ \alpha = 18, \quad \beta = 33, &\quad \mu_0 = -0.2, \quad \mu_1 = 0.01\end{aligned} \tag{5.36}$$

Integrating with a Runge-Kutta-4 scheme with a time step of 0.001 until $T = 20$, we obtain a trajectory of 20000 time steps. Figure 5.7 (top left) shows the lobe-switching behaviour of the system.

We use $K = 3$ and $K' = 4$ in SPA I, yielding no loss in the lifting polytope since $K' > 3$. Since the system evolves more slowly than the KS PDE, we choose a forward time step of $\tau_{\text{forward}} = 0.001$ and a memory time step 30-fold this size, i.e., $\tau_{\text{memory}} = 0.03$ for a better unfolding of the Chua attractor by the delay-coordinate map (see Figure 5.7, bottom left). Since the Chua circuit is a chaotic system, making precise short-term predictions is especially hard and typically not the goal for numerical methods. Instead, we compare the shape of the attractor with the one from the mSPA prediction to assess whether mSPA can recreate the long-term behaviour. Chua is especially interesting in this regard since its attractor consists of two connected components between which the dynamics switch. Figure 5.7, right, shows that especially for the barycentric coordinates this is successful with $p = 7$. In the full space, the difference in geometry between original and reconstructed attractors is stronger, but the lobe-switching behaviour is reconstructed, too.

Example 5.5. Lorenz-96, chaotic regime In order to demonstrate that mSPA can recreate statistical properties of chaotic dynamical systems in the short term, we consider the Lorenz-96 system introduced in Example 5.2 but with different parameters. We choose $d = 10$ and set the parameter $G = 5$ to obtain a chaotic behaviour of the dynamics. This can be checked by noting that the maximal Lyapunov exponent [CCV09] of the dynamics is positive.

We use a time step of length 0.05 and create one trajectory of length 100 (2000 time steps) for training in mSPA and 50 trajectories of length 7.5 (150 time steps) each, starting at randomly chosen points close to the Lorenz-96 attractor for testing. We find learning and lifting polytopes using SPA I with $K = 3$ and $K' = 8$. The lifting polytope induces a relative projection error of 0.23.

After fitting the mSPA model and making predictions starting at the starting values of the testing trajectories, we compute the autocorrelations of the predictions with those from the testing trajectories. For a set of N_{test} trajectories of length T time steps each in the learning polytope, the autocorrelation of the ith coordinate for a

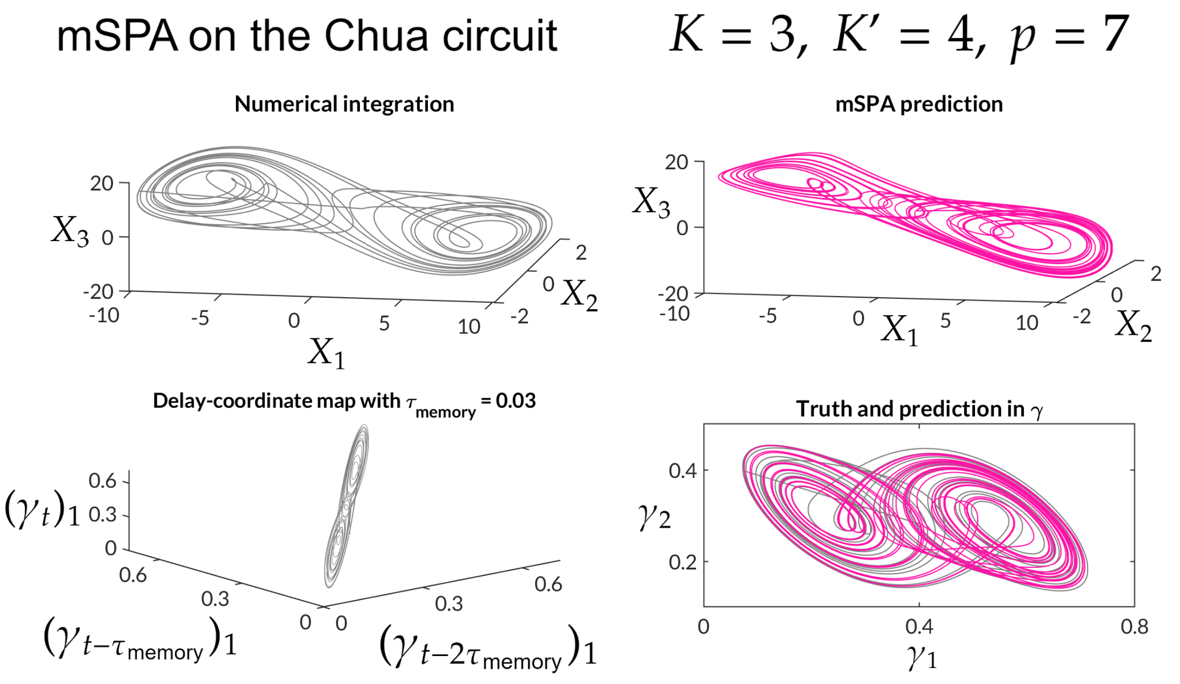


Figure 5.7: Top left: numerical realisation of the Chua circuit (5.36). Bottom left: delay-coordinate map of the first entry of the barycentric coordinates gives a similar geometric shape as the true attractor. Left: mSPA predictions with $p = 7$ in full space (top) and baycentric coordinates (bottom, together with true trajectory).

time lag of l time steps is defined as

$$a_i^l := \frac{1}{N_{\text{test}}(T-l)} \sum_{r=1}^{N_{\text{test}}} \sum_{t=1}^{T-l} ((\gamma_t^r)_i - \bar{\gamma}_i)((\gamma_{t-l}^r)_i - \bar{\gamma}_i). \tag{5.37}$$

$\bar{\gamma}$ is the mean of γ among all testing trajectories and γ_t^r is the tth point in the rth testing trajectory. The definition is analogous for γ' (lifting polytope) and x (full space).

The results are shown in Figure 5.8. mSPA gives accurate predictions in the learning polytope (Figure 5.8, **I.**). In the lifting polytope and the full space, the predictions are not as precise through the additional approximation error from the non-trivial lifting step. The autocorrelations are well recreated (Figure 5.8, **II.–IV.**) in both learning and lifting polytopes and the full space. Only for short time lags for lifting polytope and full space there is a larger difference. As demonstrated in the Chua example, mSPA precisely predicting the evolution of chaotic dynamics is generally too challenging for mSPA, but, as this example shows, it is able to reproduce quantities related to the statistical behaviour.

Example 5.6. Opinion dynamics ABM II As a last example, let us apply the learning and lifting approach with mSPA to ABM II from Chapter 4. ABM II described the evolution of opinions among agents in a bounded two-dimensional space, resulting in a high-dimensional stochastic dynamical system. This example is meant

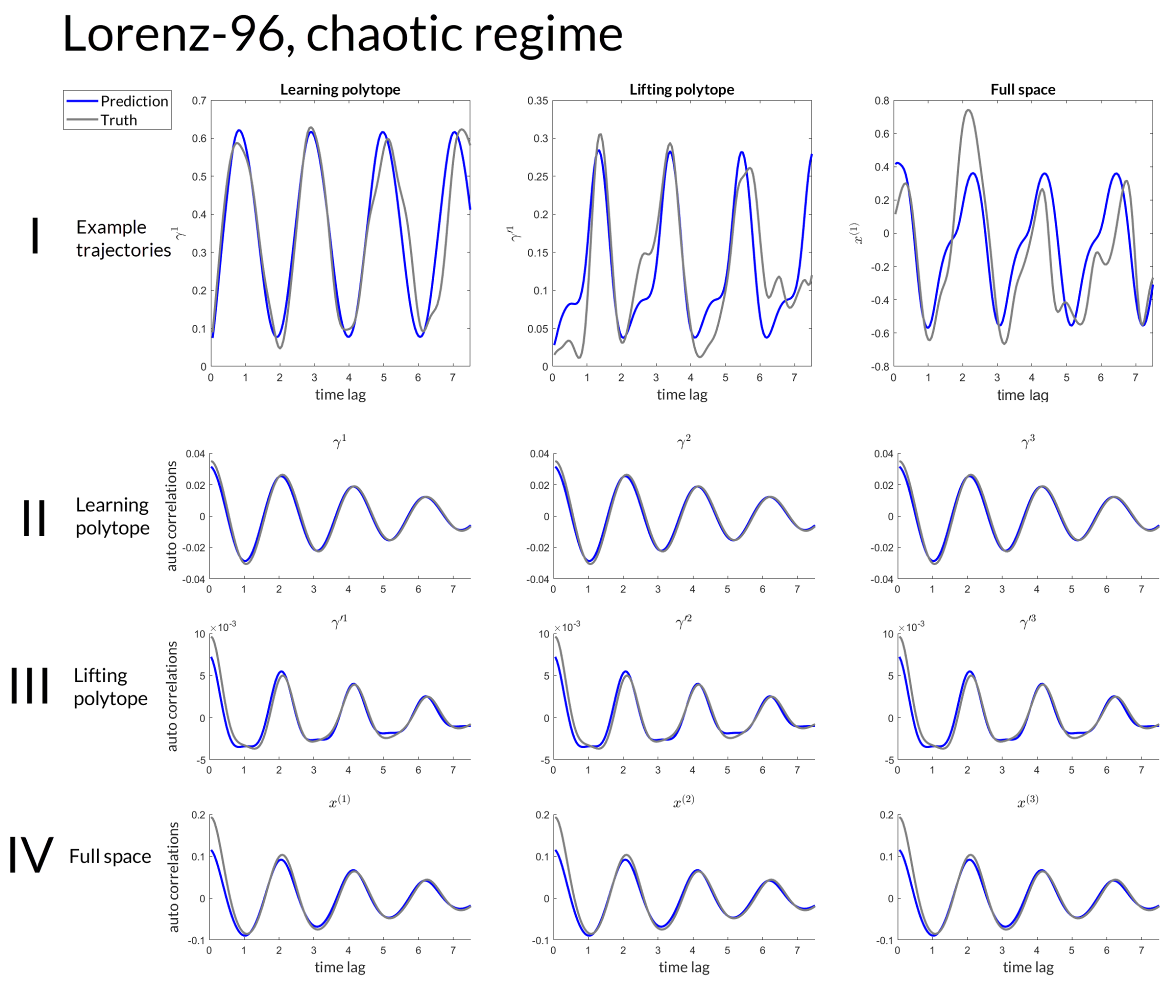


Figure 5.8: Results for mSPA on the chaotic Lorenz-96 system. **I.** Examples of forecasts. Prediction in blue (with $p = 7$), true data in grey. **II.–IV.** Autocorrelations in learning and lifting polytopes and full space. Only the first three coordinates of each are shown for the sake of visualization.

to bridge the two distinct chapters, illustrating that mSPA is directly suitable for certain problems in agent-based modelling.

To this end, in contrast to Chapter 4 we assume to be able to assess the microstate, i.e., the individual opinions of each agent, use SPA I to project this high-dimensional state onto a low-dimensional polytope and deploy mSPA to predict the evolution of the microstate.

Let us consider a complete network first with $N = 3000$ agents. The agents are distributed onto the three vertices with the ratio $[2/3, 1/6, 1/6]$. We create one realisation of length $T = 1000$ and use $K = 3$ for the learning polytope, $K' = 8$ for the lifting polytope. For computational memory reasons, we include only every tenth agent into the data, yielding a 600-dimensional system.

We find (Figure 5.9) that the microdynamics show periodic behaviour. This periodicity is also translated to the barycentric coordinates on the learning and lifting polytopes (Figure 5.9, bottom right). mSPA reconstructs the oscillating behaviour although the frequency is not precisely met (bottom left and right). Lifting back to the full state space creates qualitatively similar dynamics as the true microdynamics in the long term (bottom left).

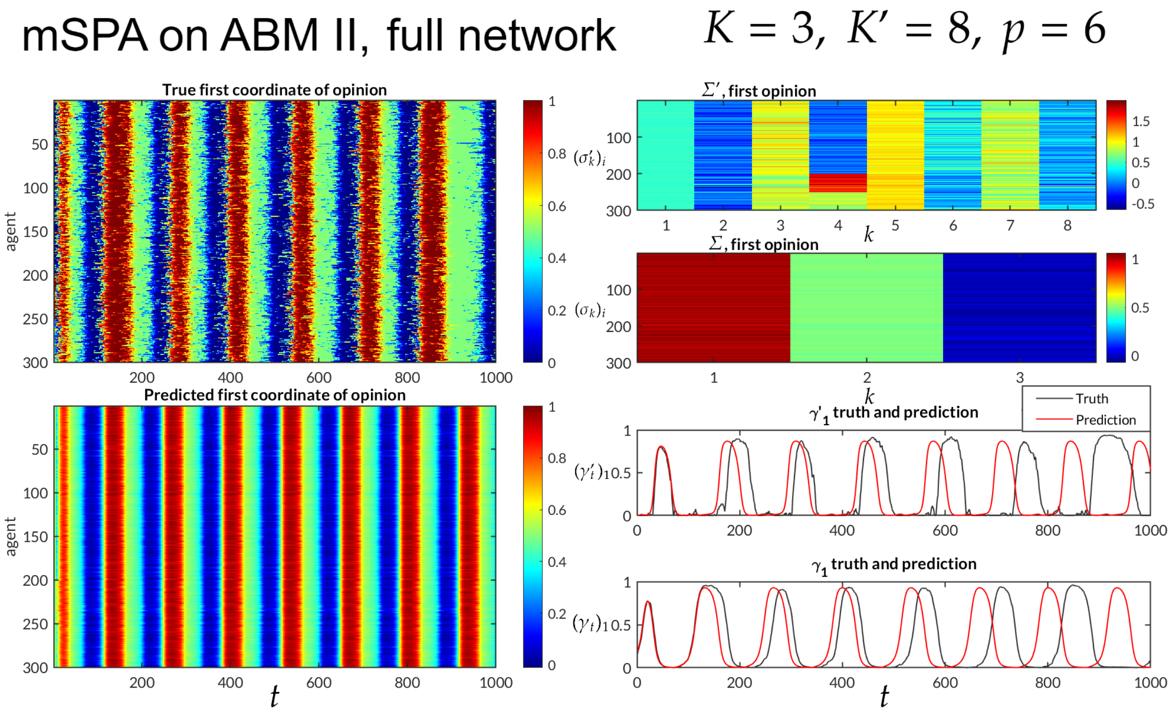


Figure 5.9: Learning and lifting with mSPA on ABM II (full network). For simplicity, only the first coordinate of opinions is shown. Left: true and predicted microstates. Top right: vertices of learning and lifting polytope. Bottom right: first entry of true and predicted barycentric coordinates in learning and lifting polytopes.

For two clusters, we again create one realisation but with different distributions between the two clusters (the ones from Chapter 4). The realisation of the microdynamics is not as regular as for one cluster, with the periodicity inside the clusters occasionally interrupted (Figure 5.10, top left). Still, mSPA manages to reconstruct the periodicity, although again with the wrong frequency compared to the true microdynamics (bottom right). Clearly, mSPA does not recreate the used realisation precisely but the creation of it underlay stochastic influences. Rather, it captures the overall behaviour well, describing time-lagged oscillations of the opinions inside each cluster. Furthermore, while the amplitudes of the true barycentric coordinates in the learning polytope approximately are matched by the prediction, the lifting does not produce a good approximation of the amplitude in the lifting polytope and, as a consequence, in the full state space. Note that the solutions of SPA I for both the learning and the lifting polytope detected the differences in clusters since the vertices represent a separation of microstates between the clusters (top right).

In summary, SPA I manages to find typical patterns in the microstates. mSPA manages to detect the periodicity in the dynamics of the ensuing barycentric coordinates for both the one- and more complicated two-cluster case. The lifting step produces a reasonable, if not perfect, approximation of the microstates in the full

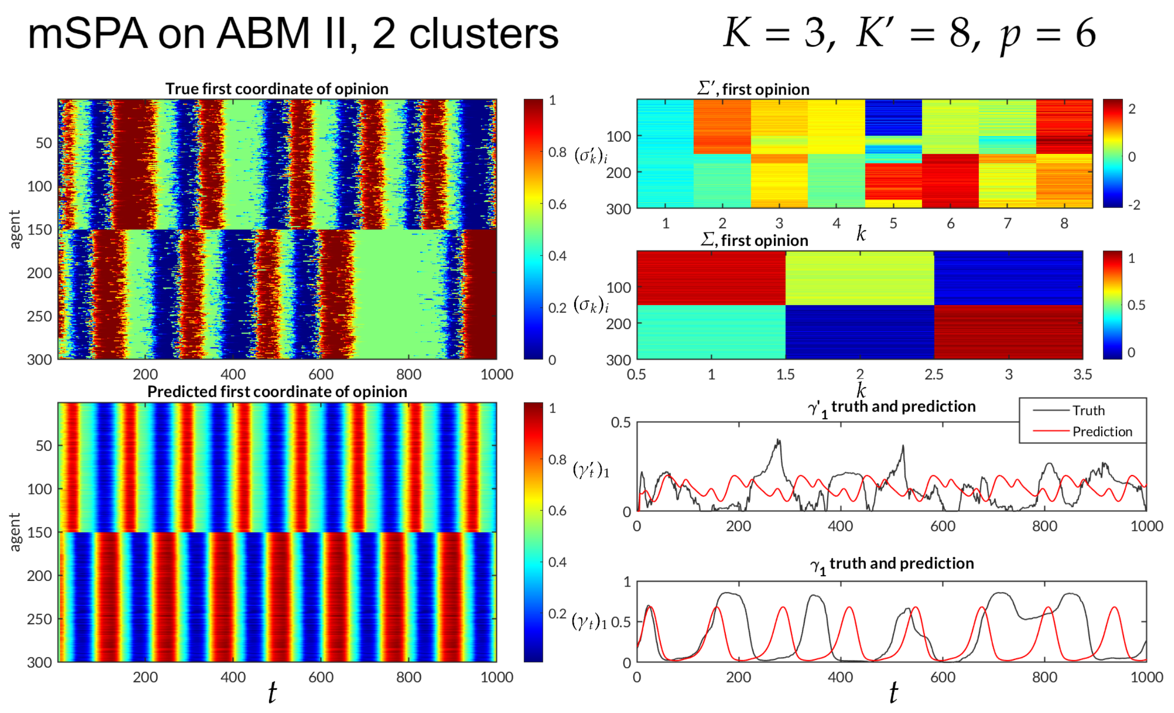


Figure 5.10: Learning and lifting with mSPA on ABM II (two clusters). For simplicity, only the first coordinate of opinions is shown. Left: true and predicted microstates. Top right: vertices of learning and lifting polytope. Bottom right: first entry of true and predicted barycentric coordinates in learning and lifting polytopes.

state space.

Overall, the examples show that mSPA with the learning and lifting approach can model systems of very different complexity and dimension on the basis of the SPA representations of points. After proving that under general conditions on a dynamical system, there exists a topologically equivalent dynamical system on the path affiliations, we estimated these dynamics solving the (mSPA) problem. We then evaluated mSPAs ability to model dynamical systems of varying dimensions that are periodic, chaotic and stochastic and saw that while it did not always generate exact recreations of the true dynamics, it could be utilized to recreate dominant long-term behaviour or short-term statistics.

Ensuing Research Questions

mSPAs shortcomings could be tackled using, e.g., more potent functions for the approximation of the operator $\mathbf{v}$, e.g., a neural network or other nonlinear functions that map into a unit simplex. Moreover, it has to be investigated why certain downfalls exists, e.g., the imperfect recreation of the frequency for periodic systems.

Some first answers on how to choose the polytope are given in [WKSS21]. They include that the polytope should be chosen so that its boundary is tight around the points that are projected onto it. Moreover, an orientation for the choice of the memory time step would be desirable. As explained in [WKSS21], it should be chosen such that the geometry of the trajectory in the product polytope is similar to the one of the true attractor. A more precise answer does not yet exist.

Especially in regard to high-dimensional systems such as ABMs, mSPA could be a valuable tool for the modelling of complex dynamics.

5.3 SPA and mSPA as Connecting Links Between Takens and Mori–Zwanzig

The mSPA method has now been introduced and its capacities and shortcomings demonstrated on various examples. The last example was meant to illustrate its connection to other aspects of this thesis: it can be used to model high-dimensional dynamical systems by projecting to lower-dimensional coordinates, modelling their evolution over time and lifting back to the full space. As motivated in the beginning of this chapter, this procedure is in line with the core theme of this thesis: in mSPA we voluntarily project to a lower-dimensional polytope. The then inaccessible information of the full state is substituted by memory terms of the accessible variables, the barycentric coordinates with respect to the polytope.

As a final step of this thesis, let us discuss where mSPA can be placed in regard to the previously introduced numerical methods and the theoretical perspectives.

The connection between mSPA and the Koopman-based methods can be formulated as follows. In the Koopman context, the barycentric coordinates denote an observable and the path affiliation function denotes a high-dimensional basis. One then solves a least squares problem to find a matrix which connects this basis function, evaluated at one time step of the dynamics, to a future state of the observable. In mSPA, a constraint is imposed on this matrix that it be column-stochastic so that subsequent steps of the dynamics are guaranteed to lie in the domain of the path affiliation function. The ensuing dynamics are of the form

$$\gamma_t = f(\gamma_{t-1}, \dots, \gamma_{t-p}) \text{ where } f = \hat{\Lambda}^p \Psi^p \tag{5.38}$$

and are therefore a nonlinear autoregressive model in γ. Here arises a direct similarity to SINAR which also estimates such models and, as we saw, emerges from the Mori–Zwanzig formalism. To combine mSPA with SINAR, it could be worthwhile to impose a sparsity constraint on the determination of $\hat{\Lambda}^p$ and possibly find more interpretable models.

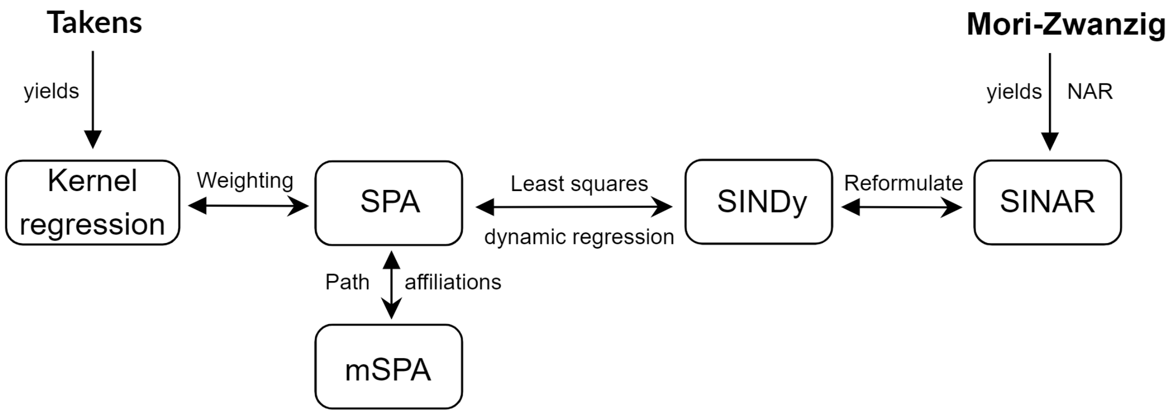
Analogously, it can be observed that the standard SPA II problem can be seen as a DMD problem with a constraint to column-stochastic matrices. Since the projection to barycentric coordinates ρ_Σ is an observable of underlying dynamics, one could equivalently view this as EDMD, too, remembering that EDMD is simply DMD with observations instead of full states.

Moreover, SPA itself can be seen in connection to Kernel regression or nearest neighbours regression, which itself were directly inspired from Takens' Theorem. In Kernel regression, subsequent states were constructed by a weighted average of data points. The weights were chosen according to the closeness of the current dynamical state to other points. In SPA II, we use as weights the closeness to vertices of a polytope and compute a weighted average of the columns of the matrix $\Sigma\Lambda$. Eq. (5.39) recaps the nearest neighbour regression and the SPA II model.

$$\text{Nearest neighbour regression: } x_{t+1} = \sum_{i=1}^{L} x_{t_i+1} w_i, \quad \text{SPA II: } x_{t+1} = \sum_{i=1}^{K} (\Sigma\Lambda_{|i})\gamma_i \tag{5.39}$$

where $\Lambda_{|i}$ denotes the ith column of Λ. In mSPA, the weights are the entries of the path affiliation function. Like Kernel and nearest neighbours regression, SPA and mSPA therefore rely on the smoothness of the projected dynamics to construct a reasonable approximation of it. In contrast to them, however, SPA II and mSPA do not directly use data for predictions but construct a parametric model, although, in mSPA, with many parameters.

Given that establishing connections between topics is an integral part of this thesis, this small section is meant to illustrate how the different theoretical approaches and numerical methods are connected. Takens' Theorem directly inspires Kernel and nearest neighbours regression. Mori–Zwanzig justifies NAR models. SPA and mSPA function as conceptual hybrids of both approaches. See the diagram below for illustration.



A rigorous theory on the connections between Takens and Mori–Zwanzig does, as of now, not exist. The perspective presented here should at least inspire further research in this direction and, furthermore, illustrate that in the field of dynamical systems modelling, there exist innumerable approaches to the same problem that, often times, are more strongly linked than one might suspect at first glance.

Summary

Detecting the governing mathematical rules of a dynamical system from data persists to be a challenge. It becomes particularly difficult when the variables of the system can only be partially observed in the form of a so-called observable function. In this case information about variables that can be vital for the prediction of future states is missing. In order to still formulate the dynamics of the observable, it can be shown that by exploiting its memory terms one can make up for the lost information. This can be placed on a mathematical ground by the delay embedding theorem of Takens and the Mori–Zwanzig formalism (MZ).

In this thesis, novel numerical methods for the modelling of the observed dynamics were developed by using Takens and MZ to extend known methods used for memoryless systems. Firstly, the method Sparse Identification of Nonlinear Dynamics (SINDy) was combined with the family of autoregressive (AR) models to define Sparse Identification of Autoregressive Models (SINAR) which seeks a sparse representation of memory-exhibiting dynamics. It was compared to various others theoretically and on examples coming from different fields.

Another new numerical method was introduced in which a high-dimensional dynamical system is projected onto a low-dimensional convex polytope using the Scalable Probabilistic Approximation (SPA) algorithm. The projection to the polytope was interpreted as an observable and memory was used to estimate the projected dynamics in a newly introduced method called memory SPA (mSPA). It was shown that mSPA can generate strong prediction accuracy for various dynamical systems while guaranteeing stability by keeping the dynamics inside the polytope.

As another contribution of this thesis, the identification of memory-exhibiting dynamics was connected with the field of agent-based modelling. To this end, two new ABMs were defined, the high-dimensional representation of the states of all individual agents was interpreted as the full system state and a low-dimensional statistic as the observable. Then the ABM was translated into the setting introduced before. It was shown in a detailed numerical analysis using SINAR that including memory generally improves the accuracy of the model identification, moreover, that adding a sparsity constraint can improve the model and that the model fitting can strongly depend on the data.

The thesis was concluded with a small note on how along the different methods discussed, the seemingly unrelated theoretical perspectives of Takens and Mori–Zwanzig could be connected.

Zusammenfassung

Die dominanten mathematischen Regeln eines dynamischen Systems aus Daten zu ermitteln ist weiterhin eine Herausforderung, welche besonders schwierig wird, wenn die Variablen eines Systems nur in Form einer so genannten Observablenfunktion partiell beobachtet werden können. Um trotzdem die Dynamik der Observablen formulieren zu können, ist es möglich, die Gedächtnisterme der Observablen zu benutzen, um das Fehlen der Information über die nicht-beobachtbaren Variablen auszugleichen. Dies kann auf ein mathematisches Fundament gestellt werden durch das (eng.) Delay-Embedding-Theorem von Takens und den Mori–Zwanzig Formalismus (MZ).

In dieser Arbeit wurden neue numerische Methoden für die Modellierung der beobachteten Dynamik entwickelt, indem mithilfe von Takens und MZ bereits bekannte Methoden für gedächtnislose Dynamiken erweitert wurden. Zunächst wurde die Methode Sparse Identification of Nonlinear Dynamics (SINDy) mit autoregressiven (AR) Modellen kombiniert, um Sparse Identification of Autoregressive Models (SINAR) zu definieren – eine Methode, die eine (eng.) sparse Darstellung einer gedächtniszeigenden Dynamik zu finden versucht. Diese Methoden wurden untereinander und mit anderen auf theoretischer Basis und anhand verschiedener Beispiele verglichen.

Eine weitere numerische Methode wurde eingeführt, mit der durch den Scalable Probabilistic Approximation (SPA) Algorithmus eine hochdimensionale Dynamik auf ein niedrigdimensionales Polytop projiziert wird. Die Projektion auf das Polytop wurde als Observable interpretiert und Gedächtnis benutzt, um die projizierte Dynamik mit der neuen Methode memory SPA (mSPA) zu schätzen. Es wurde gezeigt, dass mSPA gute Genauigkeit für verschiedene dynamische Systeme erreichen kann und gleichzeitig Stabilität garantiert, indem die Dynamik innerhalb des Polytops bleibt.

Als ein weiterer Beitrag dieser Arbeit wurde die Identifikation von gedächtniszeigender Dynamik mit dem Feld von agentenbasierter Modellierung (ABM) verbunden, wofür zwei neue ABMs definiert wurden. Die hochdimensionale Darstellung der Zustände ihrer einzelnen Agenten wurden als den vollen Systemzustand und eine niedrigdimensionale Statistik als die Observable interpretiert und die ABMs wurden in das zuvor eingeführte Konzept übersetzt. Es wurde mit SINAR u.a. detailliert gezeigt, dass das Hinzunehmen von Gedächtnis im Allgemeinen die Genauigkeit der Modellidentifikation verbessert.

Diese Arbeit wurde mit einer Beobachtung darüber abgeschlossen, wie durch die verschiedenen untersuchten Methoden die scheinbar nicht verwandten theoretischen Perspektiven von Takens und Mori–Zwanzig verbunden werden könnten.

Appendix

It was originally not planned to relegate any results or proofs to an Appendix. However, the length and technical nature of the proof of Theorem 2.4 demands it.

Proof of Theorem 2.4

Proof. For the proof we will use several results from classic linear algebra. For details, please refer to, e.g., [GVL96].

The minimizer y of a term of the form $\|y\tilde{D} - \tilde{b}\|_2$ is equal to the solution of the *normal equation* $yD = b$ with $D = \tilde{D}\tilde{D}^T, b = \tilde{b}\tilde{D}^T$. Let us further assume the perturbed system $(y + \Delta y)(D + \Delta D) = b + \Delta b$. One can show the relations

$$\|(Id - B)^{-1}\|_F \leq \frac{1}{1 - \|B\|_F} \text{ if } \|B\|_F < 1, \quad D + \Delta D = D(Id + D^{-1}\Delta D),$$

and if $\|\Delta D\|_F < \frac{1}{\|D^{-1}\|_F}$, the above lead to $\|(D + \Delta D)^{-1}\|_F \leq \frac{\|D^{-1}\|_F}{1 - \|D^{-1}\|_F\|\Delta D\|_F}$.

From this, we can derive from the perturbed system

$$\begin{aligned}
&(y + \Delta y)(D + \Delta D) = b + \Delta b \Rightarrow \Delta y(D + \Delta D) = (\Delta b - y\Delta D) \\
\Rightarrow &\Delta y = (\Delta b - y\Delta D)(D + \Delta D)^{-1} = (\Delta b - y\Delta D)(Id + D^{-1}\Delta D)^{-1}D^{-1} \\
\Rightarrow &\|\Delta y\|_F \leq \|(Id + D^{-1}\Delta D)^{-1}\|_F\|D^{-1}\|_F(\|\Delta b\|_F + \|\Delta D y\|_F) \\
\Rightarrow &\frac{\|\Delta y\|_F}{\|y\|_F} \leq \frac{\|D^{-1}\|\|D\|_F}{1 - \|D^{-1}\|\|\Delta D\|_F}(\frac{\|\Delta b\|_F}{\|b\|_F} + \frac{\|\Delta D\|_F}{\|D\|_F}).
\end{aligned}$$

For the last step it was used that $yD = b$ yields $\|y\|_F\|D\|_F \geq \|b\|_F$. Further, for the function $\|\cdot\|_{2\Sigma}$, it holds

$$\|B\|_{2\Sigma}^2 = \sum_{t=1}^{T} \|b_t\|_F^2 = \sum_{t=1}^{T}\sum_{i=1}^{m}(b_t)_i^2 = \|B\|_F^2. \quad \forall B \in \mathbb{R}^{m\times T}.$$

where b_t is the tth column of B.

Translating to the setting of the theorem, we want to minimize $\|\mathbf{X}' - A\psi(\mathbf{X})\|_F$, or equivalently solve $A\psi(\mathbf{X})\psi(\mathbf{X})^T = \mathbf{X}'\psi(\mathbf{X})^T$, assuming that $\psi(\mathbf{X})$ has full row-rank (assumption (A2)). We therefore identify D with $\psi(\mathbf{X})\psi(\mathbf{X})^T$, b with $\mathbf{X}'\psi(\mathbf{X})^T$ and y with A.

Let us define

$$\begin{aligned}
\Delta(\psi(\mathbf{X})\psi(\mathbf{X})^T) &:= \psi(\mathbf{X} + \Delta\mathbf{X})\psi(\mathbf{X} + \Delta\mathbf{X})^T - \psi(\mathbf{X})\psi(\mathbf{X})^T \\
\Delta(\mathbf{X}'\psi(\mathbf{X})^T) &:= (\mathbf{X}' + \Delta\mathbf{X}')\psi(\mathbf{X} + \Delta\mathbf{X})^T - \mathbf{X}'\psi(\mathbf{X})^T.
\end{aligned}$$

Then we obtain from the considerations above

$$\frac{\|\Delta A\|_F}{\|A\|_F} \leq \frac{\|(\psi(\mathbf{X})\psi(\mathbf{X})^T)^{-1}\|_F\|\psi(\mathbf{X})\psi(\mathbf{X})^T\|_F}{1-\|(\psi(\mathbf{X})\psi(\mathbf{X})^T)^{-1}\|_F\|\Delta(\psi(\mathbf{X})\psi(\mathbf{X})^T)\|_F}\left(\frac{\|\Delta(\mathbf{X}'\psi(\mathbf{X})^T)\|_F}{\|\mathbf{X}'\psi(\mathbf{X})^T\|_F}+\frac{\|\Delta(\psi(\mathbf{X})\psi(\mathbf{X})^T)\|_F}{\|\psi(\mathbf{X})\psi(\mathbf{X})^T\|_F}\right). \tag{A.1}$$

We can further prove the following helpful relations:

$$\begin{aligned} \|\psi(\mathbf{X})\|_F^2 &= \sum_{i=1}^{N}\sum_{t=0}^{T-1}\psi(x_t)_i^2 \leq \sum_{i=1}^{N}\sum_{t=0}^{T-1}\psi_{max}\|x_t\|_F^2 \leq \psi_{max}N\|\mathbf{X}\|_F^2 \\ \Rightarrow \|\psi(\mathbf{X})\|_F &= \psi_{max}\sqrt{N}\|\mathbf{X}\|_F. \end{aligned} \tag{A.2}$$

$$\begin{aligned} \|\psi(\mathbf{X}+\Delta\mathbf{X})^T-\psi(\mathbf{X})^T\|_F^2 &= \sum_{i=1}^{N}\sum_{t=0}^{T-1}(\psi(x_t+\Delta x_t)-\psi_i(x_t))^2 \\ &\overset{(A1)}{\leq} \sum_{i=1}^{N}\sum_{t=0}^{T-1}\|\Delta x_t\|_F^2\psi_{max}^2 = N\psi_{max}^2\|\Delta\mathbf{X}\|_F^2 \\ &\Rightarrow \|\psi(\mathbf{X}+\Delta\mathbf{X})^T-\psi(\mathbf{X})^T\|_F \leq \sqrt{N}\psi_{max}\|\Delta\mathbf{X}\|_F. \end{aligned} \tag{A.3}$$

and thus

$$\begin{aligned} \|\psi(\mathbf{X}+\Delta\mathbf{X})\|_F &= \|\psi(\mathbf{X})+(\psi(\mathbf{X}+\Delta\mathbf{X})-\psi(\mathbf{X}))\|_F \\ &\leq \|\psi(\mathbf{X})\|_F+\|\psi(\mathbf{X}+\Delta\mathbf{X})-\psi(\mathbf{X})\|_F \leq \|\psi(\mathbf{X})\|_F+\sqrt{N}\psi_{max}\|\Delta\mathbf{X}\|_F. \end{aligned} \tag{A.4}$$

We can now derive the following upper and lower bounds for the numerators and denominators of Eq. (A.1):

1.

$$\begin{aligned} \|\mathbf{X}'\psi(\mathbf{X})\|_F^2 &= \sum_{i=1}^{N}\sum_{j=1}^{m}\sum_{t=0}^{T-1}(x_{t+1})_j^2\psi_i(x_t)^2 \\ &\geq \sum_{i=1}^{N}\sum_{j=1}^{m}\sum_{t=0}^{T-1}(x_{t+1})_j^2\|x_t\|_F^2\psi_{min}^2 \\ &= \sum_{i=1}^{N}\sum_{t=0}^{T-1}\psi_{min}^2\|x_t\|_F^2\|x_{t+1}\|_F^2 \\ &= N\psi_{min}^2\|\mathbf{X},\mathbf{X}'\|_F^2 \\ \Rightarrow \|\mathbf{X}'\psi(\mathbf{X})\|_F &\geq \sqrt{N}\psi_{min}\|\mathbf{X},\mathbf{X}'\|_F. \end{aligned}$$

2.

$$\begin{aligned}
\|\Delta(\mathbf{X}'\psi(\mathbf{X})^T)\|_F &= \|(\mathbf{X}' + \Delta\mathbf{X}')\psi(\mathbf{X} + \Delta\mathbf{X})^T - \mathbf{X}'\psi(\mathbf{X})^T\|_F \\
&= \|\mathbf{X}'(\psi(\mathbf{X} + \Delta\mathbf{X})^T - \psi(\mathbf{X})^T) + \Delta\mathbf{X}'\psi(\mathbf{X} + \Delta\mathbf{X})\| \\
&\leq \|\mathbf{X}'\|\|\psi(\mathbf{X} + \Delta\mathbf{X})^T - \psi(\mathbf{X})^T\|_F + \|\Delta\mathbf{X}'\|_F\|\psi(\mathbf{X} + \Delta\mathbf{X})\|_F \\
&\overset{(A.3),(A.4)}{\leq} \|\mathbf{X}'\|_F\|\Delta\mathbf{X}\|_F\psi_{max}\sqrt{N} + \|\Delta\mathbf{X}'\|_F(\|\psi(\mathbf{X})\|_F + \|\Delta\mathbf{X}\|_F\psi_{max}\sqrt{N}) \\
&\overset{(A.2)}{\leq} \|\mathbf{X}'\|_F\|\Delta\mathbf{X}\|_F\psi_{max}\sqrt{N} + \|\Delta\mathbf{X}'\|_F(\sqrt{N}\psi_{max}\|\mathbf{X}\|_F + \|\Delta\mathbf{X}\|_F\psi_{max}\sqrt{N}) \\
&\leq \psi_{max}\sqrt{N}(\|\mathbf{X}'\|_F\|\Delta\mathbf{X}\|_F + \|\Delta\mathbf{X}'\|_F\|\mathbf{X}\|_F + \|\Delta\mathbf{X}'\|_F\|\Delta\mathbf{X}\|_F)
\end{aligned}$$

3.

$$\begin{aligned}
\|\psi(\mathbf{X})\psi(\mathbf{X})^T\|_F^2 = \sum_{i,j=1}^{N}\sum_{t=1}^{T}\psi_i(x_t)^2\psi_j(x_t)^2 &\geq \sum_{i,j=1}^{N}\sum_{t=1}^{T}\psi_{min}^4\|x_t\|_F^4 \\
\Rightarrow \|\psi(\mathbf{X})\psi(\mathbf{X})^T\|_F &\geq N\psi_{min}^2\|\mathbf{X}\|_{4\Sigma}
\end{aligned}$$

$\|\psi(\mathbf{X})\psi(\mathbf{X})^T\|_F \leq N\psi_{max}^2\|\mathbf{X}\|_{4\Sigma}$ analogously.

4.

$$\begin{aligned}
\|\Delta(\psi(\mathbf{X})\psi(\mathbf{X})^T)\|_F^2 &= \|\psi(\mathbf{X} + \Delta\mathbf{X})\psi(\mathbf{X} + \Delta\mathbf{X})^T - \psi(\mathbf{X})\psi(\mathbf{X})^T\|_F^2 \\
&= \sum_{i,j=1}^{N}\sum_{t=1}^{T}(\psi_i(x_t + \Delta x_t)\psi_j(x_t + \Delta x_t) - \psi_i(x_t)\psi_j(x_t))^2 \\
&\leq \sum_{i,j=1}^{N}\sum_{t=1}^{T}\psi_{max}^4\|\Delta x_t\|_F^4 = \psi_{max}^4 N^2\|\Delta\mathbf{X}\|_{4\Sigma}^2 \\
&\Rightarrow \|\Delta(\psi(\mathbf{X})\psi(\mathbf{X}))^T\|_F \leq \psi_{max}^2 N\|\Delta\mathbf{X}\|_{4\Sigma}.
\end{aligned}$$

Substituting into Eq. (A.1), we can derive

$$\begin{aligned}
\frac{\|\Delta A\|_F}{\|A\|_F} \leq &\frac{\|(\psi(\mathbf{X})\psi(\mathbf{X})^T)^{-1}\|_F\psi_{max}^2 N\|\mathbf{X}\|_{4\Sigma}}{1 - \|(\psi(\mathbf{X})\psi(\mathbf{X})^T)^{-1}\|_F\psi_{max}^2 N\|\Delta\mathbf{X}\|_{4\Sigma}} \\
&(\frac{\psi_{max}(\|\mathbf{X}'\|_F\|\Delta\mathbf{X}\|_F + \|\Delta\mathbf{X}'\|_F\|\mathbf{X}\|_F + \|\Delta\mathbf{X}'\|_F\|\Delta\mathbf{X}\|_F)}{\psi_{min}\|\mathbf{X},\mathbf{X}'\|_{2\Sigma}} + \frac{\psi_{max}^2\|\Delta\mathbf{X}\|_{4\Sigma}}{\psi_{min}^2\|\mathbf{X}\|_{4\Sigma}}).
\end{aligned}$$

Assumption (A2) was needed to guarantee the existence of $(\psi(\mathbf{X})\psi(\mathbf{X})^T)^{-1}$ and assumption (A3) was needed to guarantee that the denominator in the first term is positive. □

Bibliography

[Aey81] D. Aeyels. Generic observability of differentiable systems. *SIAM Journal for Control and Optimization*, 19:595, 09 1981.

[Aka74] H. Akaike. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19(6):716–723, 1974.

[AM17] H. Arbabi and I. Mezic. Ergodic Theory, Dynamic Mode Decomposition and Computation of Spectral Properties of the Koopman Operator. *SIAM J. Appl. Dyn. Syst.,(4)*, 16:2096–2126, 2017.

[AMRT01] H. D. Abarbanel, N. Masuda, M. Rabinovich, and E. Tumer. Distribution of mutual information. *Physics Letters A*, 281(5):368 – 373, 2001.

[ANT19] N. Ali, D. Neagu, and P. Trundle. Evaluation of k-nearest neighbour classifier performance for heterogeneous data sets. *SN Applied Sciences*, 1, 11 2019.

[AQC13] F. Al-Qahtani and S. Crone. Multivariate k-nearest neighbour regression for time series data — a novel algorithm for forecasting UK electricity demand. pages 1–8, 08 2013.

[ASM03] R. Alonso-Sanz and M. Martín. Elementary cellular automata with memory. volume 14, pages 99–126, 01 2003.

[Ban14] S. Banisch. From microscopic heterogeneity to macroscopic complexity in the contrarian voter model. *Advances in Complex Systems*, online ready:1450025, 12 2014.

[Ban16] S. Banisch. *Markov Chain Aggregation for Agent-Based Models*. Springer, 01 2016.

[BBP$^+$16] S. Brunton, B. Brunton, J. Proctor, E. Kaiser, and J. Kutz. Chaos as an intermittently forced linear system. *Nature Communications*, 8, 2016.

[BCK20] G. Boschi, C. Cammarota, and R. Kühn. Opinion dynamics with emergent collective memory: A society shaped by its own past. *Physica A: Statistical Mechanics and its Applications*, 558:124909, 2020.

[BD91] P. J. Brockwell and R. A. Davis. *Time Series: Theory and Methods*, volume 2. Springer, 1991.

[BH05] H. Barreto and F. Howland. *The Gauss-–Markov Theorem*, page 335–377. Cambridge University Press, 2005.

[Bil13] S. Billings. *Nonlinear System Identification: NARMAX Methods in the Time, Frequency, and Spatio-Temporal Domains*, volume 1. Wiley, 2013.

[BK19] S. Brunton and J. Kutz. *Neural Networks and Deep Learning*, pages 195–226. 02 2019.

[BKK$^+$17] A. Bittracher, P. Koltai, S. Klus, R. Banisch, M. Dellnitz, and C. Schütte. Transition manifolds of complex metastable systems: Theory and data-driven computation of effective dynamics. *Journal of Nonlinear Science*, 28, 04 2017.

[BLK16] S. L. Brunton, J. P. L., and J. N. Kutz. Sparse Identification of Nonlinear Dynamics with Control (SINDYc). *IFAC-PapersOnLine Issue 18*, 49:710–715, 2016.

[BNK20] S. L. Brunton, B. R. Noack, and P. Koumoutsakos. Machine learning for fluid mechanics. *Annual Review of Fluid Mechanics*, 52(1):477–508, 2020.

[BNNB20] G. Benrhmach, K. Namir, A. Namir, and J. Bouyaghroumni. Nonlinear autoregressive neural network and extended Kalman filters for prediction of financial time series. *Journal of Applied Mathematics*, (5057801), 2020.

[Bol86] T. Bollerslev. Generalized autoregressive conditional heteroskedasticity. *Journal of Econometrics*, 31(3):307–327, 1986.

[BPK16] S. L. Brunton, J. P. L. Proctor, and J. N. Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, 113(15):3932–3937, 2016.

[BZF$^+$18] Z. Benko, A. Zlatniczki, D. Fabó, A. Sólyom, L. Oss, A. Telcs, and Z. Somogyvári. Exact inference of causal relations in dynamical systems. *arXiv:1808.10806*, pages 1–43, 08 2018.

[CCV09] M. Cencini, F. Cecconi, and A. Vulpiani. *Chaos: From Simple Models to Complex Systems*. World Scientific, 2009.

[CDFB18] G. Chen, X. Duan, N. Friedkin, and F. Bullo. Social power dynamics over switching and stochastic influence networks. *IEEE Transactions on Automatic Control*, PP:1–1, 04 2018.

[CFL09] C. Castellano, S. Fortunato, and V. Loreto. Statistical physics of social dynamics. *Rev. Mod. Phys.*, 81:591–646, May 2009.

[CFS82] I. P. Cornfeld, S. V. Fomin, and Y. G. Sinai. *Ergodic Theory*. Springer, 1982.

[Chi17] C. Chicone. Chapter 2 - differential equations. In C. Chicone, editor, *An Invitation to Applied Mathematics*, pages 11–30. Academic Press, 2017.

[CHK00] A. J. Chorin, O. H. Hald, and R. Kupferman. Optimal prediction and the Mori–Zwanzig representation of irreversible processes. *Proceedings of the National Academy of Sciences*, 97(7):2968–2973, 2000.

[CHK02] A. J. Chorin, O. H. Hald, and R. Kupferman. Optimal prediction with memory. *Physica D*, 166:239–257, 2002.

[CHZ$^+$18] N. Conrad, L. Helfmann, J. Zonker, S. Winkelmann, and C. Schütte. Human mobility and innovation spreading in ancient times: a stochastic agent-based simulation approach. *EPJ Data Science*, 7, 12 2018.

[CKM86] L. Chua, M. Komuro, and T. Matsumoto. The double scroll family. *IEEE Transactions on Circuits and Systems*, 33(11):1072–1118, 1986.

[CL15] A. Chorin and F. Lu. Discrete approach to stochastic parametrization and dimension reduction in nonlinear dynamics. *Proceedings of the National Academy of Sciences of the United States of America*, 112, 07 2015.

[CLKB19] K. Champion, B. Lusch, J. N. Kutz, and S. L. Brunton. Data-driven discovery of coordinates and governing equations. *Proceedings of the National Academy of Sciences*, 116(45):22445–22451, 2019.

[CLL$^+$05] R. Coifman, S. Lafon, A. Lee, M. Maggioni, B. Nadler, F. Warner, and S. Zucker. Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion maps. *Proceedings of the National Academy of Sciences of the United States of America*, 102:7426–31, 06 2005.

[CMCW$^+$20] S. Cao, A. Montoya-Castillo, W. Wang, T. E. Markland, and X. Huang. On the advantages of exploiting memory in Markov state models for biomolecular dynamics. *The Journal of Chemical Physics*, 153(1):014105, 2020.

[CS73] P. Clifford and A. Sudbury. A model for spatial conflict. *Biometrika*, 60(3):581–588, 1973.

[CS18] G. H. Chen and D. Shah. Explaining the success of nearest neighbor methods in prediction. *Foundations and Trends® in Machine Learning*, 10(5-6):337–588, 2018.

[CVM04] H. Chen, B. Vidakovic, and D. Mavris. Multiscale forecasting method using ARMAX models. *Current Development in Theory and Applications of Wavelets*, 4, 01 2004.

[Cyb89] G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems (MCSS)*, 2(4):303–314, December 1989.

[DBB$^+$19] A. De, S. Bhattacharya, P. Bhattacharya, N. Ganguly, and S. Chakrabarti. Learning linear influence models in social networks from transient opinion dynamics. *ACM Transactions on the Web*, 13:1–33, 11 2019.

[DBGM20] V. Deshmukh, E. Bradley, J. Garland, and J. Meiss. Using curvature to select the time lag for delay reconstruction. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 30:063143, 06 2020.

[DHH03] M. Dehnert, W. Helm, and M.-T. Hütt. A discrete autoregressive process as a model for short-range correlations in DNA sequences. *Physica A: Statistical Mechanics and its Applications*, 327:535–553, 09 2003.

[DHvMZ15] M. Dellnitz, M. Hessel-von Molo, and A. Ziessler. On the computation of attractors for delay differential equations. *Journal of Computational Dynamics*, 3, 08 2015.

[DR20] A. S. Dogra and W. Redman. Optimizing neural networks via Koopman operator theory. *arXiv abs/2006.02361, 34th Conference on Neural Information Processing Systems (NeurIPS 2020),*, 10 2020.

[DS11] E. Deyle and G. Sugihara. Generalized theorems for nonlinear state space reconstruction. *PloS one*, 6:e18295, 03 2011.

[DW05] P. Deuflhard and M. Weber. Robust Perron cluster analysis in conformation dynamics. *Linear Algebra and its Applications*, 398:161–184, 2005. Special Issue on Matrices and Mathematical Biology.

[EYWR18] A. Eftekhari, H. L. Yap, M. B. Wakin, and C. J. Rozell. Stabilizing embedology: Geometry-preserving delay-coordinate maps. *Phys. Rev. E*, 97:022222, Feb 2018.

[FGL$^+$21] G. Froyland, D. Giannakis, B. R. Lintner, M. Pike, and J. Slawinska. Spectral analysis of climate dynamics with operator-theoretic approaches. 12(6570), 2021.

[FMZF21] K. Fukami, T. Murata, K. Zhang, and K. Fukagata. Sparse identification of nonlinear dynamics with low-dimensionalized flow representations. *Journal of Fluid Mechanics*, 926:A10, 2021.

[Fra85] J. Franke. A Levinson-Durbin recursion for autoregressive-moving average processes. *Biometrika*, 72(3):573–581, 1985.

[FS86] A. M. Fraser and H. L. Swinney. Independent coordinates for strange attractors from mutual information. *Phys. Rev. A*, 33:1134–1140, Feb 1986.

[Fuk80] K. Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36:193–202, 1980.

[GGH21] F. Gilani, D. Giannakis, and J. Harlim. Kernel-based prediction of non-Markovian time series. *Physica D: Nonlinear Phenomena*, 418:132829, 01 2021.

[Gho18] S. Ghosh. *Kernel Smoothing: Principles, Methods and Applications*. 01 2018.

[GHS87] G. H. Golub, A. Hoffman, and G. Stewart. A generalization of the Eckart-Young-Mirsky matrix approximation theorem. *Linear Algebra and its Applications*, 88/89:317–327, 1987.

[GHS08] S. Ganguli, D. Huh, and H. Sompolinsky. Memory traces in dynamical systems. *Proceedings of the National Academy of Sciences*, 105(48):18970–18975, 2008.

[Gia21] D. Giannakis. Delay-coordinate maps, coherence, and approximate spectra of evolution operators. *Research in the Mathematical Sciences*, 8, 03 2021.

[GKES19] P. Gelß, S. Klus, J. Eisert, and C. Schütte. Multidimensional approximation of nonlinear dynamical systems. *Journal of Computational and Nonlinear Dynamics*, 14, 03 2019.

[GPNH20] S. Gerber, L. Pospisil, M. Navandar, and I. Horenko. Low-cost scalable discretization, prediction, and feature selection for complex systems. *Science Advances*, 6(5), 2020.

[Gut16] Y. Gutman. *Takens' embedding theorem with a continuous observable*, pages 134–141. De Gruyter, 06 2016.

[GVL96] G. H. Golub and C. F. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, third edition, 1996.

[GWSP82] H. Greenside, A. Wolf, J. Swift, and T. Pignataro. Impracticality of a box-counting algorithm for calculating the dimensionality of strange attractors. *Phys. Rev. A*, 25, 06 1982.

[Hé76] M. Hénon. A two-dimensional mapping with a strange attractor. *Comm. Math. Phys.*, 50(1):69–77, 1976.

[Han17] H. Hanappi. Agent-based modelling. history, essence, future. *PSL Quarterly Review*, 70, 05 2017.

[HDPCBS18] E. Herrera-Delgado, R. Perez-Carrasco, J. Briscoe, and P. Sollich. Memory functions reveal structural properties of gene regulatory networks. *PLOS Computational Biology*, 14:e1006003, 02 2018.

[HEVEDB10] C. Hijón, P. Español, E. Vanden-Eijnden, and R. Delgado-Buscalioni. Mori–Zwanzig formalism as a practical computational tool. *Faraday discussions*, 144:301–22; discussion 323, 01 2010.

[HFN19] M. Hoffmann, C. Fröhner, and F. Noé. Reactive sindy: Discovering governing reactions from concentration data. *The Journal of Chemical Physics*, 150:025101, 01 2019.

[HHSN07] I. Horenko, C. Hartmann, C. Schütte, and F. Noé. Data-based parameter estimation of generalized multidimensional langevin processes. *Physical review. E, Statistical, nonlinear, and soft matter physics*, 76:016706, 08 2007.

[Hop82] J. J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, 79(8):2554–2558, 1982.

[Hor10] I. Horenko. On the identification of nonstationary factor models and their application to atmospheric data analysis. *Journal of the Atmospheric Sciences*, 67:1559–1574, 05 2010.

[HP18] B. Husic and V. Pande. Markov State Models: From an art to a science. *Journal of the American Chemical Society*, 140 (7):2386—2396, 01 2018.

[HRR15] K. Hager, J. Rauh, and W. Rid. Agent-based modeling of traffic behavior in growing metropolitan areas. *Transportation Research Procedia*, 10:306–315, 2015. 18th Euro Working Group on Transportation, EWGT 2015, 14-16 July 2015, Delft, The Netherlands.

[HS97] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9:1735–80, 1997.

[HSN20] J. Hermann, Z. Schätzle, and F. Noé. Deep-neural-network solution of the electronic Schrödinger equation. *Nature Chemistry*, 12:1–7, 10 2020.

[HTF09] T. Hastie, R. Tibshirani, and J. Friedman. *Kernel Smoothing Methods*, pages 191–218. Springer New York, New York, NY, 2009.

[Huk93] J. Huke. Embedding nonlinear dynamical systems, a guide to Takens theorem. *Internal Report, DRA Malvern*, 1993.

[Hyn18] R. J. Hyndman. *Forecasting: principles and practice*, pages 72–80. OTexts; 2nd Edition, 2018.

[JC16] I. T. Jolliffe and J. Cadima. Principal component analysis: a review and recent developments. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 374(2065):20150202, 2016.

[JM98] K. Judd and A. Mees. Embedding as a modeling problem. *Physica D: Nonlinear Phenomena*, 120(3):273 – 286, 1998.

[Jor97] M. I. Jordan. Chapter 25 - serial order: A parallel distributed processing approach. In J. W. Donahoe and V. Packard Dorsel, editors, *Neural-Network Models of Cognition*, volume 121 of *Advances in Psychology*, pages 471–495. North-Holland, 1997.

[JSW98] N. Jennings, K. Sycara, and M. Wooldridge. A roadmap of agent research and development. *Autonomous Agents and Multi-Agent Systems*, 1:7–38, 03 1998.

[JSW18] A. Jedrzejewski and K. Sznajd-Weron. Impact of memory on opinion dynamics. *Physica A: Statistical Mechanics and its Applications*, 505, 03 2018.

[KBA92] M. B. Kennel, R. Brown, and H. D. I. Abarbanel. Determining embedding dimension for phase-space reconstruction using a geometrical construction. *Phys. Rev. A*, 45:3403–3411, Mar 1992.

[KDB$^+$18] J. Kappler, J. Daldrop, F. Brünig, M. Boehle, and R. Netz. Memory-induced acceleration and slowdown of barrier crossing. *The Journal of Chemical Physics*, 148:014903, 01 2018.

[KDG15] D. Kondrashov, M. D.Chekroun, and M. Ghil. Data-driven non-Markovian closure models. *Physica D: Nonlinear Phenomena*, 297:33–55, 2015.

[KGPS16] S. Klus, P. Gelß, S. Peitz, and C. Schütte. Tensor-based dynamic mode decomposition. *Nonlinearity*, 31, 06 2016.

[KHN19] J. Kappler, V. B. Hinrichsen, and R. R. Netz. Non-Markovian barrier crossing with two-time-scale memory is dominated by the faster memory component. *The European Physical Journal E*, 42(119):1–16, 2019.

[KKB18] E. Kaiser, J. N. Kutz, and S. L. Brunton. Sparse identification of nonlinear dynamics for model predictive control in the low-data limit. *Proc. R. Soc. A 474: 20180335*, 474, 2018.

[KKS16] S. Klus, P. Koltai, and C. Schütte. On the numerical approximation of the Perron-Frobenius and Koopman operator. *Journal of Computational Dynamics*, 3:51 – 79, 09 2016.

[KLL$^+$21] N. Kovachki, Z. Li, B. Liu, K. Azizzadenesheli, K. Bhattacharya, A. Stuart, and A. Anandkumar. Neural operator: Learning maps between function spaces, 08 2021.

[KLT07] P. Klimek, R. Lambiotte, and S. Thurner. Opinion formation in laggard societies. *EPL (Europhysics Letters)*, 82, 2007.

[KNK$^+$18] S. Klus, F. Nüske, P. Koltai, H. Wu, I. Kevrekidis, C. Schütte, and F. Noé. Data-driven model reduction and transfer operator approximation. *Journal of Nonlinear Science*, 28:985–1010, 06 2018.

[Koo31] B. O. Koopman. Hamiltonian Systems and Transformation in Hilbert Space. *Proceedings of the National Academy of Sciences*, 17(5):315–318, 1931.

[KR99] S. Krantz and C. Robinson. *Dynamical Systems: Stability, Symbolic Dynamics, and Chaos*. Studies in Advanced Mathematics. CRC-Press, 1999.

[KSH12] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[LBBH98] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, volume 86, pages 2278–2324, 1998.

[LBL16] H. Lei, N. A. Baker, and X. Li. Data-driven parameterization of the generalized Langevin equation. *Proceedings of the National Academy of Sciences*, 113(50):14183–14188, 2016.

[LC17] X. Li and W. Chu. The Mori-Zwanzig formalism for the derivation of a fluctuating heat conduction model from molecular dynamics. *Communications in mathematical sciences*, 17:539–563, 2017.

[Lie03] E. Liebscher. Strong convergence of estimators in nonlinear autoregressive models. *Journal of Multivariate Analysis*, 84:247–261, 02 2003.

[LJMR12] R. Laubenbacher, A. S. Jarrah, H. S. Mortveit, and S. Ravi. *Agent Based Modeling, Mathematical Formalism for*, pages 88–104. Springer New York, New York, NY, 2012.

[LKB18] B. Lusch, J. N. Kutz, and S. L. Brunton. Deep learning for universal linear embeddings of nonlinear dynamics. *Nature Communications*, 9(1):4950, 2018.

[LKN20] L. Lavacchi, J. Kappler, and R. R. Netz. Barrier crossing in the presence of multi-exponential memory functions with unequal friction amplitudes and memory times. *EPL (Europhysics Letters)*, 131(4):40004, 09 2020.

[LL21] K. K. Lin and F. Lu. Data-driven model reduction, wiener projections, and the Koopman-Mori-Zwanzig formalism. *Journal of Computational Physics*, 424:109864, 2021.

[LLC15] F. Lu, K. Lin, and A. Chorin. Data-based stochastic model reduction for the Kuramoto–Sivashinsky equation. *Physica D: Nonlinear Phenomena*, 340, 09 2015.

[LTLA21] Y. T. Lin, Y. Tian, D. Livescu, and M. Anghel. Data-driven learning for the Mori–Zwanzig formalism: a generalization of the Koopman learning framework. 07 2021.

[LTR17] H. Lin, M. Tegmark, and D. Rolnick. Why does deep and cheap learning work so well? *Journal of Statistical Physics*, 168, 09 2017.

[Mac67] J. Macqueen. Some methods for classification and analysis of multivariate observations. In *In 5-th Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, 1967.

[MFM$^+$20] I. Manojlović, M. Fonoberova, R. Mohr, A. Andrejčuk, Z. Drmač, Y. Kevrekidis, and I. Mezic. Applications of koopman mode analysis to neural networks. *ArXiv*, abs/2006.11765, 06 2020.

[MGO10] O. Mesa, V. Gupta, and P. O'Connell. Dynamical system exploration of long-term memory in the climate system (invited). *AGU Fall Meeting Abstracts*, pages 209–229, 12 2010.

[Mis12] A. K. Misra. A simple mathematical model for the spread of two political parties. *Nonlinear Analysis: Modelling and Control, 2012, No. 3*, 17:343–354, 2012.

[MLC17] H.-F. Ma, S. Leng, and L. Chen. Data-based prediction and causality inference of nonlinear dynamics. *Science China Mathematics*, 61:403–420, 10 2017.

[MOW15] I. G. K. Matthew O. Williams, Clarence W. Rowley. A kernel-based method for data-driven koopman spectral analysis. *Journal of Computational Dynamics*, 2(2):247–265, 2015.

[MP43] W. Mcculloch and W. Pitts. A logical calculus of ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 5:127–147, 1943.

[MPWN18] A. Mardt, L. Pasquali, H. Wu, and F. Noé. VAMPnets: Deep learning of molecular kinetics. *Nature Communications*, 9, 01 2018.

[MWE18] C. Ma, J. Wang, and W. E. Model reduction with memory and the machine learning of dynamical systems. *Communications in Computational Physics*, 25(4):947–962, 2018.

[Neu16] K. Neusser. *Time Series Econometrics*. Number 978-3-319-32862-1 in Springer Texts in Business and Economics. Springer, May 2016.

[NWWS21] J.-H. Niemann, S. Winkelmann, S. Wolf, and C. Schütte. Agent-based modeling: Population limits and large timescales. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 31(3):033140, 2021.

[OAH20] S. Okuno, K. Aihara, and Y. Hirata. Forecasting high-dimensional dynamics exploiting suboptimal embeddings. *Scientific Reports*, 10:664, 01 2020.

[PCFS80] N. H. Packard, J. P. Crutchfield, J. D. Farmer, and R. S. Shaw. Geometry from a time series. *Phys. Rev. Lett.*, 45:712–716, Sep 1980.

[PD18] S. Pan and K. Duraisamy. Long-time predictive modeling of nonlinear dynamical systems using neural networks. *Complexity*, 2018:1–26, 2018.

[PD20] S. Pan and K. Duraisamy. On the structure of time-delay embedding in linear models of non-linear dynamical systems. *Chaos*, 30 7:073135, 2020.

[Pen55] R. Penrose. A generalized inverse for matrices. *Mathematical Proceedings of the Cambridge Philosophical Society*, 51(3):406–413, 1955.

[PGSH18] L. Pospíšil, P. Gagliardini, W. Sawyer, and I. Horenko. On a scalable nonparametric denoising of time series signals. *Communications in Applied Mathematics and Computational Science*, 13:107–138, 02 2018.

[PKJC18] M. Paluš, A. Krakovská, J. Jakubík, and M. Chvosteková. Causality, dynamical systems and the arrow of time. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 28(7):075307, 2018.

[RHLD19] C. Ravazzi, S. Hojjatinia, C. Lagoa, and F. Dabbene. Randomized opinion dynamics over networks: Influence estimation from partial observations. Proceedings of the IEEE Conference on Decision and Control, pages 2452–2457. Institute of Electrical and Electronics Engineers Inc., January 2019.

[RHW86] D. Rumelhart, G. Hinton, and R. Williams. Leaning internal representations by back-propagating errors. *Nature*, 323, 01 1986.

[RKEV03] K. Roth, I. Kauppinen, P. A. Esquef, and V. Välimäki. Frequency warped Burg's method for AR-modeling. volume 2003, pages 5 – 8, 11 2003.

[Rob05] J. Robinson. A topological delay embedding theorem for infinite-dimensional dynamical systems. *Nonlinearity*, 18:2135, 07 2005.

[Ros58] F. Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65 6:386–408, 1958.

[RRW20] R. Rabben, S. Ray, and M. Weber. Isokann: Invariant subspaces of koopman operators learned by a neural network. *The Journal of chemical physics*, 153:114109, 09 2020.

[RT17] D. Rolnick and M. Tegmark. The power of deeper networks for expressing natural functions, 2017.

[Rud16] S. Ruder. An overview of gradient descent optimization algorithms. arXiv:1609.04747, 2016.

[Sar11] M. Sarich. *Projected Transfer Operators*. PhD thesis, 2011.

[SBDH03] J. Stark, D. Broomhead, M. Davies, and J. Huke. Delay embeddings for forced systems. II. stochastic forcing. *Journal of Nonlinear Science*, 13:519–577, 12 2003.

[SBK20] G. Stepaniants, B. W. Brunton, and J. N. Kutz. Inferring causal networks of dynamical systems through transient dynamics and perturbation. *Phys. Rev. E*, 102:042309, Oct 2020.

[SBSR19] S. D. Salas, A. L. T. Brandão, J. B. P. Soares, and J. A. Romagnoli. Data-driven estimation of significant kinetic parameters applied to the synthesis of polyolefins. *Processes*, 7(5), 2019.

[Sch10] P. J. Schmid. Dynamic mode decomposition of numerical and experimental data. *Journal of Fluid Mechanics*, 656:5–28, 2010.

[SEJ$^+$20] A. Senior, R. Evans, J. Jumper, J. Kirkpatrick, L. Sifre, T. Green, C. Qin, A. Žídek, A. Nelson, A. Bridgland, H. Penedones, S. Petersen, S. Crossan, P. Kohli, D. Jones, D. Silver, K. Kavukcuoglu, and D. Hassabis. Improved protein structure prediction using potentials from deep learning. *Nature*, 577:1–5, 01 2020.

[SLST17] A. Sîrbu, V. Loreto, V. Servedio, and F. Tria. *Opinion Dynamics: Models, Extensions and External Effects*, pages 363–401. 05 2017.

[SM90] G. Sugihara and R. May. Nonlinear forecasting as a way of distinguishing chaos from measurement error m time series. *Nature*, 344:734, 05 1990.

[SMY$^+$12] G. Sugihara, R. May, H. Ye, C.-h. Hsieh, E. Deyle, M. Fogarty, and S. Munch. Detecting causality in complex ecosystems. *Science (New York, N.Y.)*, 338:496–500, 09 2012.

[SS11] R. Shumway and D. Stoffer. *Time Series Analysis and Its Applications With R Examples*, volume 9. 01 2011.

[SSAB20] C. Scherer, R. Scheid, D. Andrienko, and T. Bereau. Kernel-based machine learning for efficient simulations of molecular liquids. *Journal of Chemical Theory and Computation*, 16:3194–3204, 04 2020.

[SSMZK11] A. Schindler, S. Sperlich, I. Martínez-Zarzoso, and T. Keib. *Bandwidth Selection in Nonparametric Kernel Estimation*. Niedersächsische Staats- und Universitätsbibliothek Göttingen, 2011.

[Sta99] J. Stark. Delay embeddings for forced systems. i. deterministic forcing. *Journal of Nonlinear Science*, 9:255–332, 1999.

[Sta20] F. Stahlberg. Neural machine translation: A review. *Journal of Artificial Intelligence Research*, 69(8):343–418, 2020.

[SYC91] T. Sauer, J. Yorke, and M. Casdagli. Embedology. *Journal of Statistical Physics*, 65:579–616, 11 1991.

[Tak06] F. Takens. *Detecting Strange Attractors in Turbulence. Lecture Notes in Mathematics*, volume 898, pages 366–381. 11 2006.

[TCP06] A. Treuille, S. Cooper, and Z. Popovic. Continuum crowds. *ACM Trans. Graph.*, 25:1160–1168, 07 2006.

[Tib96] R. Tibshirani. Regression shrinkage and selection via the Lasso. *Journal of the Royal Statistical Society. Series B (Methodological), no. 1*, 58:267–288, 1996.

[TRL$^+$14] J. H. Tu, C. W. Rowley, D. M. Luchtenburg, S. L. Brunton, and J. N. Kutz. On dynamic mode decomposition: Theory and applications. *Journal of Computational Dynamics*, 1(2):391–421, 2014.

[TZ19] Q. Teng and L. Zhang. Data driven nonlinear dynamical systems identification using multi-step CLDNN. *AIP Advances*, 9:085311, 08 2019.

[VHMN20] G. Van Houdt, C. Mosquera, and G. Nápoles. A review on the long short-term memory model. *Artificial Intelligence Review*, 53, 12 2020.

[VSP$^+$17] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. *31st Conference on Neural Information Processing Systems*, 06 2017.

[WCDC$^+$21] H. Wulkow, T. O. F. Conrad, N. Djurdjevac Conrad, S. A. Müller, K. Nagel, and C. Schütte. Prediction of Covid-19 spreading and optimal coordination of counter-measures: From microscopic to macroscopic models to pareto fronts. *PLOS ONE*, 16(4):1–29, 04 2021.

[WDKW21] N. Wunderling, J. F. Donges, J. Kurths, and R. Winkelmann. Interacting tipping elements increase risk of climate domino effects under global warming. *Earth System Dynamics*, 12(2):601–619, 2021.

[Whi36] H. Whitney. Differentiable manifolds. *Annals of Mathematics*, 37, 6 1936.

[WHR12] E. Wit, E. v. d. Heuvel, and J.-W. Romeijn. 'all models are wrong...': an introduction to model uncertainty. *Statistica Neerlandica*, 66(3):217–236, 2012.

[WKR14] M. Williams, I. Kevrekidis, and C. Rowley. A data-driven approximation of the Koopman operator: Extending Dynamic Mode Decomposition. *Journal of Nonlinear Science*, 25, 2014.

[WKS21] N. Wulkow, P. Koltai, and C. Schütte. Memory-based reduced modelling and data-based estimation of opinion spreading. *Journal of Nonlinear Science*, 31, 02 2021.

[WKSS21] N. Wulkow, P. Koltai, V. Sunkara, and C. Schütte. Data-driven modelling of nonlinear dynamics by barycentric coordinates and memory. *arXiv:2112.06742*, 12 2021.

[WTH$^+$] N. Wulkow, R. Telgmann, K.-D. Hungenberg, C. Schütte, and M. Wulkow. Deterministic and stochastic parameter estimation for polymer reaction kinetics I: Theory and simple examples. *Macromolecular Theory and Simulations*, 2100017.

[Wu87] S. Wu. Chua's circuit family. *Proceedings of the IEEE*, 75(8):1022–1032, 1987.

[WWS18] X. Wu, H.-T. Wai, and A. Scaglione. Estimating social opinion dynamics models from voting records. *IEEE Transactions on Signal Processing*, PP:1–1, 04 2018.

[Yao00] J.-F. Yao. On least squares estimation for stable nonlinear ar processes. *Annals of the Institute of Statistical Mathematics*, 52:316–331, 2000.

[YDGS15] H. Ye, E. Deyle, L. Gilarranz, and G. Sugihara. Distinguishing time-delayed causal interactions using convergent cross mapping. *Scientific Reports*, 5:14750, 10 2015.

[ZBI$^+$19] A. Zeyer, P. Bahar, K. Irie, R. Schluter, and H. Ney. A comparison of transformer and LSTM encoder decoder models for ASR. pages 8–15, 12 2019.

[ZLY$^+$13] L. Zhang, Q. Liu, W. Yang, W. Nai, and D. Dong. An improved k-nearest neighbor model for short-term traffic flow prediction. *Procedia - Social and Behavioral Sciences*, 96:653–662, 11 2013.

[Zwa01] R. Zwanzig. *Nonequilibrium Statistical Mechanics*. Oxford University Press, 2001.