

Data-Driven Modelling & Forecasting of Chaotic Systems

B.L. Nortier

Abstract

Collecting measurements from underlying artificial, natural, and physical systems through observables to realise a dynamical model with greater descriptive power is a well-established practice. Physical systems can often be modelled by instance with differential equations derived from the laws of physics, if we knew the laws. However, we often do not fully understand the many complexities of the more intricate systems we interact with on a daily basis. In this project we start with the hypothesis that an discrete-time dynamical systems, particularly those which are chaotic generates the data. The general learning problem is then formulated and a well-known approach to the problem, alongside its practical limitations are considered. We then introduce the notion of driven dynamical systems and establish the theoretical existence of dynamical systems possessing certain properties that guarantee them to be topologically conjugate ('dynamically equivalent') to the dynamics of the underlying system from which measurements were taken. The methodology is practically implemented by making use of a recurrent neural network and using deep learning methods to train on the recurrent neural network states to show that we may obtain long-term consistent predictions for simple physical systems such as the double pendulum as well as a few other attractors exhibiting chaotic behaviour. The machine learning algorithms for modelling the double pendulum thus far have not been successful when it exhibits chaos.

Contents

Chapter 1

Introduction

Experimenting on biological, physical, and artificial systems in order to generate a more informative dynamical model is a well-established practice in modern science. Traditional methods for modelling physical systems are based on laws of physics that are based either on empirical relationships or on intuition. For systems that evolve with time, physical laws yield mathematical equations that govern how the quantities evolve with time. However our world around us, for the most part, much more complex than that which can be distilled into elegant equations. We do not fully understand many of the more complex systems, nor do they even provide us with a good physical intuition of the underlying principles governing the dynamics. To complicate this - the underlying systems we observe often display sensitive dependence on initial conditions; despite having highly similar initial conditions, differing orbits diverge quickly and to such an extent that it becomes seemingly impossible to retrace their steps back to the original conditions. Even the presence of usually 'negligible' computational noise or measurement error renders long-term point-wise prediction infeasible. There are many difficulties that one encounters while modelling such systems:

- i. One may not have access to the complete states of the systems
- ii. The actual system may be described by functions that behave wildly in the sense their graphs have a wild oscillatory behaviour, i.e., they have a large functional complexity.

Models derived primarily from data can be classified into three categories:

- i. Interpretable models (i.e., they establish relationships between internal physical mechanisms),
- ii. Partially interpretable models capturing some modes of the dynamics,
- iii. Non-Interpretable models, defined as such mainly due to the fact that they are defined on a different phase space that is usually high-dimensional.

Examples in the literature attempting to forecast data from such systems have tried a number of different approaches with differing degrees of success. When we have complete access to states of

the system, an ordinary differential equation can be obtained from data ([?, ?] and [?, ?]) wherein one could approximate the vector field by a library of functions to obtain interpretable models.

Recent partially interpretable models available in the literature have been based on the Koopman operator (e.g.,[?, ?]) to employ observables mapping the data onto a higher-dimensional space. This then makes the dynamics in the higher-dimensional space more amenable for approximation by a linear transformation. Such methods, obviously, do not guarantee an exact reconstructions for nonlinear models, and in practice provide poor long-term consistency for a large class of systems. **(Citation)**

The non-interpretable models include the delay embedding and the machine learning algorithms. One could learn a system conjugate to the underlying system by applying the Takens delay embedding [?] when one has “good” observations from the system; this learnt system could then be used for forecast the observed data. Takens delay embedding theorem [?] and its various generalisations (e.g., [?, ?, ?]) establishes the learnability of a system constructed by concatenating sufficiently large previous time-series observations of a dynamical system into a vector (called delay coordinates). This then establishes the existence of a map on the space of delay coordinates equivalent (or topologically conjugate) to the underlying map from which the observed time-series was first obtained. Although topological conjugacy guarantees an alternate representation of the underlying system, the quality of this representation still depends on numerous parameters, making the comprehension of the dynamics rather unreliable (e.g., [?]). One reason for this fragility is that the embedded attractor in the reconstruction space not always an attractor of the map that is learnt in the reconstruction space, despite unquestionably being an invariant set. When the embedded object is not explicitly known to be an attractor (as explained in Chapter ??) it can cause predictions to fail.

Practically, the application of Takens embedding involves learning a map through some technique and consequently one wishes that these would have low functional complexity[?], i.e., functions with fewer oscillatory graphs. Pure machine learning methodology that processes temporal information (like the echo state networks (Citations to echo state networks)) by mapping data onto a higher dimensional space for further processing. Although they perform well on forecasting some dynamical systems, they fail completely on others (Citations here) as there is often no guarantee that the right function was learnt during training.

This thesis deals with the implementation and analysis of non-interpretable models that can guarantee exact reconstruction. The project work concerns the study and implementation of a method ([?]) that incorporates learning a function by mapping the data on to a higher dimensional space using what is called a driven dynamical system (See Chapter 4). With a clear understanding of how the data is mapped onto the higher dimensional space, the method then permits the learning of a dynamical system topologically conjugate to that of the underlying system. Instead of linear regression as in the training of echo state networks, deep learning methods are employed to learn the correct function. With slight modifications to the implementation in the paper (with Adriaan), we show that one can construct accurate non-interpretable models with the ability to reconstruct

attractors from more hard-to-forecast systems like the double pendulum. (The forecasting of the time-series from a double pendulum has not been reported before.) Moreover, we also demonstrate that long-term statistical consistency is preserved.

By solidifying the mathematical underpinnings of our theory, we hope to guarantee the ability to construct models with predictive power ranging from molecular biology to neuroscience. in the near future. (We can modify this sentence when the report is completed.)

The more intricate details of proofs are referred to where relevant throughout. This report is organised as follows:

In Chapter ??, we recall the definition of a discrete-time dynamical system, how a discrete-system arises from a flow of an ODE and then proceed to define the inverse-limit space and topological conjugacy of autonomous systems.

In Chapter ??, we introduce the problem of forecasting dynamical systems, state the Takens delay embedding theorem and discuss various issues faced while forecasting.

In Chapter ??, a driven dynamical system is defined and discuss the properties of a specific class of driven dynamical systems that we make use of in this thesis/report/paper.

Finally in Chapter ?? we show the implementation of these forecasting methods, and conclusions are provided in Chapter ??.

Chapter 2

Discrete-time Dynamical Systems

In this chapter, we provide a brief description of what a discrete-time dynamical system is and what it means for it to exhibit chaos. We refer to [?, ?] for more details.

At its most elementary level, a dynamical system is just something that evolves deterministically through time. In the context of this thesis, deterministic refers to the fact that a system evolves according to specified rules rather than based on random events. Dynamical systems arise in a variety of situations. A continuous-time dynamical system describes the states for all values of the time. Specifically, if the motion of a pendulum in which the quantities such as the angular position and angular momentum are known at all times, then it is a continuous-time dynamical system. The equations of the dynamical system can take the form of one or more ordinary differential equations that determine the relevant quantities at any future time if we know the initial location and momentum. In ecology, discrete-time dynamical systems are widely used to model population growth. The model in this case is a function calculating the following generation's population given the population of the previous generation. If we know the starting population, we may once again calculate the population at any time in the future.

Formally, a function $T : U \rightarrow U$, where U is some set is a *discrete-time dynamical system* and its iterates $\{u, Tu, T^2u, \dots\}$, where T^n denotes the n -fold composition of T with itself, describe the evolution of an initial condition $u \in U$ (Note that we frequently drop the brackets and denote $T(u)$ by Tu so as to simplify notations).

Continuous-time dynamical systems modelled using ordinary differential equations can give rise to discrete-time dynamical systems. To see this, consider a differential equation $\dot{x} = f(x)$, $f : \mathbb{R}^n \rightarrow \mathbb{R}^n$, $n \in \mathbb{N}$ given to have a unique solution passing through each point $x \in \mathbb{R}^n$, call it $x(t)$ where $x(0) = x_0$.

Definition 1 (Flow of an Equation). The flow of the equation $\dot{x} = f(x)$ is defined to be a mapping $\varphi : \mathbb{R}^n \times \mathbb{R} \rightarrow \mathbb{R}$ where $\varphi(x_0, t) = x(t)$ for the solution $x(t)$ with $x(0) = x_0$

By fixing $t = K \in (0, \infty)$, we can define the *time- K map* as $T(x) := \varphi(x_0, K)$, and it is easily verified that $T \circ T(x_0) = \varphi(x_0, 2K)$. In general the m^{th} iterate of x_0 under T would be the value of the solution of the ODE evaluated at time mK with the initial condition x_0 , i.e. $\varphi(x_0, mK)$. Thus ordinary differential equations give rise to a discrete-time dynamical system by sampling the value of the solution $x(t)$ at time intervals K units apart.

A numerical discretization of a differential equation can also give rise to a discrete-time dynamical system. For instance, Euler's method approximates $\dot{x}(t)$ by $(x(t + h) - x(t))/h$; if h is fixed throughout, the solution of a differential equation $\dot{x} = f(x)$ can be approximated at the time instant $t + (m + 1)h$ by iterating the equation

$$x(t + (m + 1)h) = x(t + mh) + hf(x(t + mh)) \quad (2.1)$$

Adopting more succinct notation by replacing $x(t + mh)$ with u_m , we rewrite the above equation as

$$u_{m+1} = u_m + hf(u_m) \quad (2.2)$$

or in even simpler terms as the discrete-time dynamical system with map $T(u) = u + hf(u)$, where T , u and f are understood to be as above.

Of course, discrete-time dynamical systems need not always arise through a differential equation, and once again we may consider the field of ecology where discrete dynamical systems are often directly derived or assumed. **Citation** There is a school of thought that advocates discrete-time dynamical systems to be more natural for modelling real-world observations than differential equations and we refer the interested reader to [?].

2.1 Invariant Sets

A core concept in the study of dynamical systems is that of invariance.

Definition 2 (Invariant Set). Given a discrete-time dynamical system $T : U \rightarrow U$, a subset $A \subset U$ is said to be an *invariant set* if $T(A) = A$.

We also define the orbit of a function T .

Definition 3 (Orbit of T). The orbit of T is to be defined the sequence $\bar{u} = \{u_n\}_{n \in \mathbb{Z}}$ obeying the update equation, $u_{n+1} = Tu_n$, $n \in \mathbb{Z}$.

Two examples of invariant sets include a fixed point where $Tu = u$ for $u \in U$, and a periodic orbit, i.e., a set of iterates $\{u, Tu, T^2u, \dots, T^pu\}$, where $T^{p+1}u = u$ for some $p \in \mathbb{Z}$. The entire space

U could be also be invariant. Consider for example the space $U = [0, 1]$, where $Tu = 4u(1 - u)$ and then U is invariant (as every $u \in U$ can be written as $u = 4x(1 - x)$ for some $x \in U$).

We may learn a great deal about the iterates of a dynamical system by considering the types of invariant sets of a discrete-time dynamical system. For example, if $U = [0, 1]$ has map $Tu = u/2$, then the only invariant set is $\{0\}$, and all orbits approach this invariant set as time flows in the forward direction. Indeed, if a some non-zero invariant set (call it B) exists, then there is some $r \in (0, 1] \cap B$. But $r \notin T(B)$ since any orbit with initial value r will be a decreasing sequence. Moreover, every orbit of T will be decreasing and therefore approach the value 0 as $n \rightarrow \infty$.

One may ask if every orbit approaches an invariant set? In general the answer is no, since for the dynamical system $T : \mathbb{R} \rightarrow \mathbb{R}$ defined by $Tu = 2u$, any orbit that does not intersect the invariant set $\{0\}$, will not approach any invariant set.

However, when the space U is compact, all orbits approach an invariant set. This is since the set of limit points of the orbit can be shown to be invariant [?]. So, when U is compact, the ω -limit set $\omega(u; T)$ of a point u defined to the collection of limit points of the sequence $\{x, Tu, T^2u, \dots\}$ is nonempty, and $\omega(u; T)$ is invariant.

Example 1. For the map $Tu = u^2$ defined on $[0, 1]$ all orbits lie in the invariant set $\{1\}$ or else would approach the invariant set $\{0\}$.

Invariant sets have various properties. Invariant sets can be attracting or repelling depending on how orbits in its vicinity behave. Recall the examples $Tu = u/2$ and $Tu = 2u$ defined on \mathbb{R} (where $\{0\}$ “attracts” orbits) or the example, $Tu = u^2$ defined on $[0, 1]$ (where $\{1\}$ “repels” orbits). We are interested in attractive invariant sets since they capture the long-term dynamics as time increases. In particular we are interested in those invariant sets named attractors.

Definition 4 (Attractor). Let $T : U \rightarrow U$, where U is a metric space with metric d . A compact subset $A \subset U$ is said to be an attractor if it satisfies the three conditions:

1. A is invariant.
2. A is asymptotically stable, i.e., for every $\epsilon > 0$ and for all u so that $d(u, A) < \epsilon$, we have $d(T^n u, A) \rightarrow 0$ as $n \rightarrow \infty$.
3. A has Lyapunov stability, i.e., for every $\epsilon > 0$ there exists a $\delta(\epsilon) > 0$ so that $d(u, A) < \delta$ implies $d(T^n u, A) < \epsilon$ for all $n \geq 0$.

Indeed in our previous examples, it can be verified that the system $Tu = u/2$ defined on \mathbb{R} and $Tu = u^2$ defined on $[0, 1]$ the singleton set $\{0\}$ is an attractor. For the system, $Tu = 1 - |2u - 1|$ on $[0, 1]$, one may easily verify that the only attractor is the entire space $[0, 1]$. This is since between any two points $u < v$ in $[0, 1]$, and for any a, b so that $u \leq a < b \leq v$, we can find an n so that $T^n(a, b) = [0, 1]$ (as would be explained later in this chapter).

A dynamical system can have several attractors, and may also be contained in another attractor. For the example, $Tu = u^2$ on $[0, 1]$, both $\{0\}$ and $[0, 1]$ are attractors. It is a known result that for dynamical systems on a compact space there always exists an attractor.

The dynamics restricted to an invariant set can be complicated. For instance an invariant set could be just a single point or it could have an infinite set. If the invariant set is infinite then there is a possibility of complicated dynamics, and a particular well-studied phenomenon of such complexity gives rise to what is called chaotic behaviour, and has been studied in detail over the last fifty years.

2.2 Chaos

In the 1960s, a number of mathematicians and mathematically interested scientists independently discovered chaos in the mathematical sense. The meteorologist Edward Lorenz may have been the first to explain this phenomenon in his 1963 paper [?]. The notions of invariance, attractivity and chaos may also be described for continuous systems, and Lorenz's system comprised of a system of differential equations. The narrative of Lorenz's discovery of chaos, as well as the history of other forerunners in this subject, are fascinating. For those who are interested, we highly recommend James Gleick's book *Chaos: The Making of a New Science* [?]. It clearly illustrates these experiences, explains why chaos was such a startling and crucial mathematical and scientific discovery, and describes the underlying mathematical notions for non-specialists.

There are several intricately different definitions of chaos available in the literature (**Cite**), with each of them indicating some aspect of complexity. In practice, it is not possible to verify which definition or notion does the underlying dynamical system satisfies when only data is observed from it. We do, however, specifically recall the definition of chaos in the sense of Devaney [?, ?] so as to understand some nuances behind the complexity. Devaney's definition of chaos has three requirements, with the first being the notion of sensitive dependence on initial conditions.

Definition 5. [Sensitive Dependence on Initial Conditions] A dynamical system $T : U \rightarrow U$ is said to have sensitive dependence on initial conditions (SDIC) if there exists a $\delta > 0$ such that for every $u \in U$ and in every neighborhood of $u \in U$ there exists a $v \in U$ and an integer $N := N(u, v) \in \mathbb{Z}$ such that $d(T^N u, T^N v) > \delta$.

It is important to acquire a sense of this definition as it can easily be misunderstood. It is common misconception to interpret SDIC as two close points (u and v) that eventually become separated by a distance δ under iteration by T . But this is not true. To fully comprehend all the subtleties of the concept, one needs to discuss it more thoroughly by considering each sentence with care and attention, whereafter one may examine how they fit together to convey the concept of sensitive dependence.

To this end, we make 3 remarks:

1. First, the $\delta > 0$ in the definition of SDIC is independent of u .
2. Second, in every neighborhood of u , we may not necessarily find all points v in the neighborhood distinct from u that would separate from the forward iterates of u .
3. Finally, N depends upon u and v chosen, and their iterates may not separate for ever (i.e., for all $n > N$) and we allow their iterates to get arbitrarily close in the future.

To illustrate the concept of SDIC, we present two examples - one from Mathematics and the other from the field of Physics.

Example 2. The logistic map(LM), a recursion relation of the form $x_{n+1} = rx_n(1 - x_n)$ where $x_n \in [0, 1]$ and $r < 4$ is of particular interest to us as it is classically used to illustrate the influence of chaos on a system ([Cite](#)). Below are plotted the first 75 iterates of the LM for $r = 3.7$ and $x_0 = 0.2$ with an adjustment of 0.001 in x_0 and r respectively.

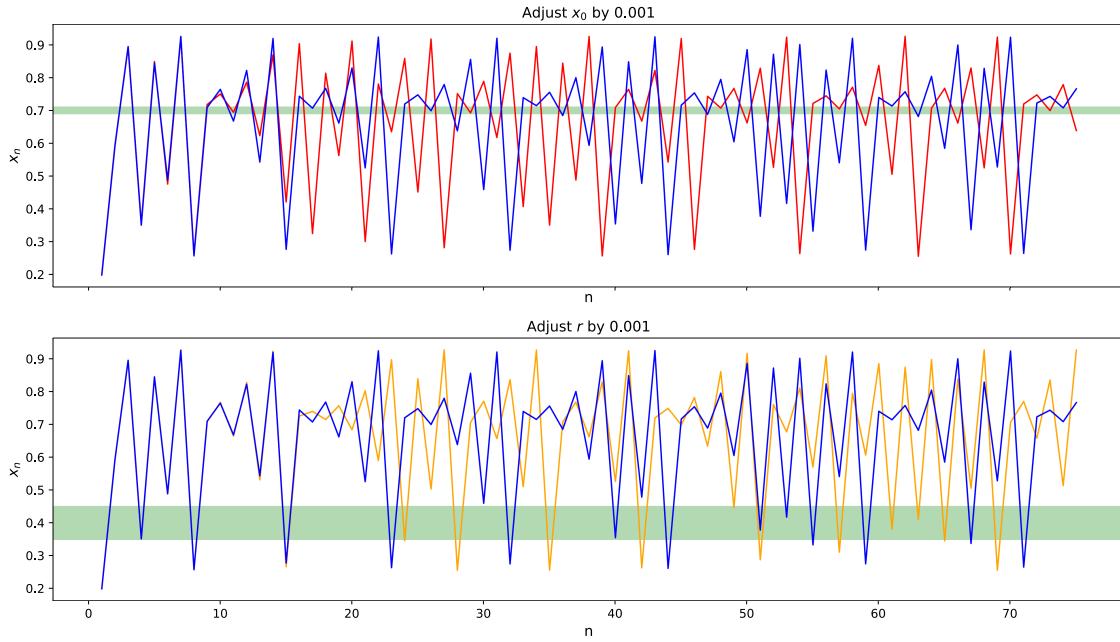


Fig. 1. Figure generated for Logistic Map for $r = 3.7$, $x_0 = 0.2$ where each parameter is varied separately by 0.001.

For an intuitive sense of the fact that the LM exhibits SDIC, we draw two green neighbourhoods of diameters $\delta_1 = 0.1$, $\delta_2 = 0.02$ around the points $u_1 = 0.7$, $u_2 = 0.4$ respectively. It is easy to see that every neighbourhood around these two points will contain some $v_1, v_2 \in [0, 1]$ and natural number $N_1, N_2 \in \mathbb{N}$ respectively s.t. the distance $d(T^N u_1, T^N v) > \delta_1 = 0.02$ and $d(T^N u_2, T^N v) > \delta_2 = 0.1$.

Example 3. A second example pertains to a simple physical system called the double pendulum - a pendulum with another pendulum attached to its head given an initial position and angular velocity.

Below are two figures denoting the trajectories of a double pendulum's second head after some elapsed time. Depicted are three pendulum heads with equal angular velocity, but differing ever so slightly in initial position. As can be seen below, the trajectories diverge very quickly to become completely different. The double pendulum will be discussed in much greater detail in a later chapter (??), but we mention the example here to provide a numerical example of a practical system exhibiting SDIC.

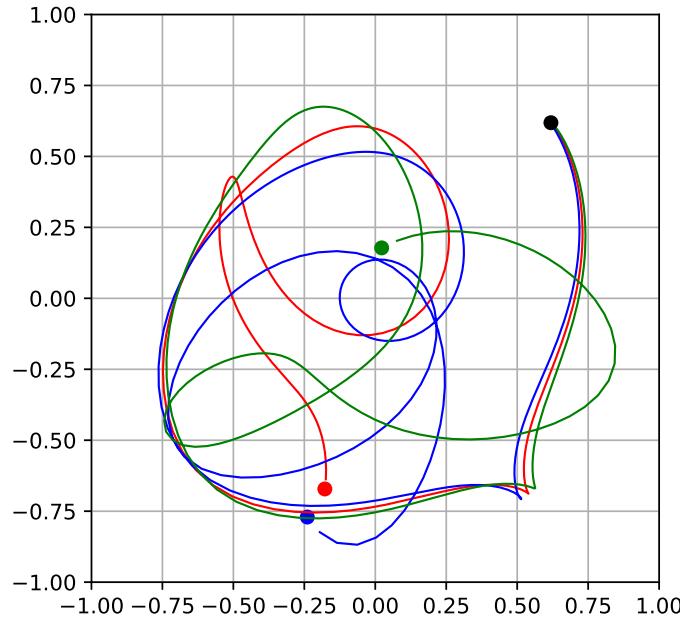


Fig. 2. Three double pendulum heads with equal angular velocity and initial angles differing by 0,025 radians

The second part in Devaney's definition of chaos concerns topological transitivity.

Definition 6 (Topologically Transitive). A dynamical system $F : U \rightarrow U$ is topologically transitive if for any pair of nonempty open sets E_1 and E_2 there exists a $n \in \mathbb{N}$ such that $T^n(E_1) \cap E_2 \neq \emptyset$.

Topologically transitivity implies that the iterates of an open set of initial conditions gets mixed up with other open sets. On a compact metric space, it can be shown that topological transitivity also implies the existence of a point whose forward iterates are dense [?]; or in other words, the

¹EDNOTE: B: Is it desirable to have a more rigorous explanation here?

orbit going through this point will be dense in the compact metric space. In fact, it is topologically more likely that the choice of an arbitrary point will be one whose iterates are almost dense.

Definition 7 (Meager Set). A subset of a topological space U is said to be a meager set if it can be written as a countable union of sets of with empty interior. The complement of the a meager set is set to be a residual set.

To be more precise (see [?]), for a discrete-time dynamical system on a compact space, the set of points with dense iterates are residual, and they are typical or likely to be observed in practice. In this thesis, we assume that when data is observed from a topologically transitive system, we assume that the data arises from a dense orbit.

We may now define the notion of Chaos as formulated by Devaney[?].

Definition 8 (Devaney's Chaos). A dynamical system $T : U \rightarrow U$ is said to exhibit chaos in the sense of Devaney if it satisfies the three properties:

1. T has SDIC.
2. T is topologically transitive.
3. The set of periodic points of T are dense in U .

Example 4. The standard tent map $Tu = 1 - |2u - 1|$ defined on $[0, 1]$ is a well-known example of a dynamical system satisfying these three properties.

We now reason as to why this is true. The graph of the map T is piecewise linear with two straight lines, one connecting the points $(0, 0)$ and $(\frac{1}{2}, 1)$ and the other connecting $(\frac{1}{2}, 1)$ with $(1, 0)$. They form a so-called tent with base centered at $u = 1/2$. The graph of the map T^2 comprises two symmetric tents with their base centered at $1/4$ and $3/4$. See Figure ??

In general, the graph of T^k contains 2^{k-1} tents. Given any point u and an open neighborhood $W \subset [0, 1]$ of u , we can find $k \in \mathbb{Z}$ large enough so that $T^k(W) = [0, 1]$ because we can accommodate a tent whose base is contained in W . This implies the existence of two points w_1 and w_2 in W so that $d(T^k w_1, T^k w_2) = 1$, and by the triangle inequality at least one of the inequalities $d(T^k u, T^k w_1) > \delta$ or $d(T^k u, T^k w_2) > \delta$ hold when $\delta \in (0, \frac{1}{2})$. So T has sensitive dependence on initial conditions.

Next, let W_1 and W_2 be two nonempty open sets. Given any open set W_1 , we can find a $k \in \mathbb{N}$ large enough so that $T^k(W_1) = [0, 1]$ because we can accommodate a tent with base contained in W . So $T^k(W_1) \cap W_2 \neq \emptyset$, and thus T has topological transitivity.

Finally, for every open interval (a, b) , the graph of T^k intersects the graph of the identity map on $[0, 1]$. This follows from the fact established above that there exists a $k \in \mathbb{N}$ such that $T^k(a, b) = [0, 1]$. If the intersection point has coordinates of the form (p, p) , the point p is then a fixed point of

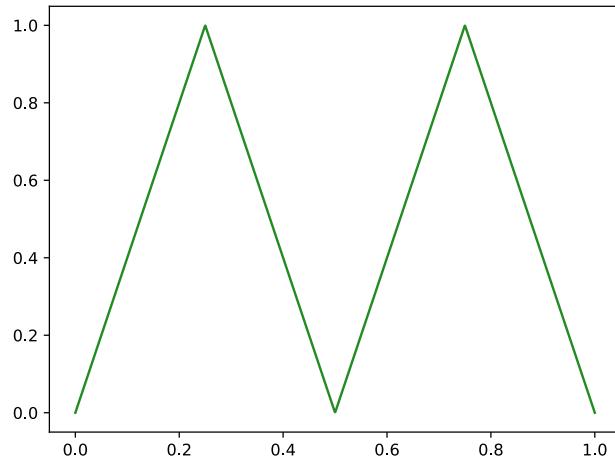


Fig. 3. Graph of the T^2 map with bases at $x = 0$, $x = \frac{1}{2}$ and $x = 1$ respectively.

T^k and therefore also a periodic point of T . Since we have found a periodic point in the interval W , the set of periodic points are dense in T . This follows because any fixed point of T^k is also a periodic point, and we have found this in an arbitrary interval.

2.3 Conjugacy

We now turn our attention to the subject of conjugacy that describes when two dynamical systems are equivalent dynamically.

To show topological similarity or sameness between two metric or topological spaces, one needs to establish a homeomorphism between the two spaces. However, in the study of dynamical systems defined on two spaces, establishing a homeomorphism does not indicate that the systems are dynamically related in any way. For instance, the maps $Tu = u^2$ and $Tu = 1 - |2u - 1|$ defined on $[0, 1]$ have totally different behaviour. To find systems that are dynamically similar, one rather needs to establish a kind of dynamical equivalence which we illustrate in a simple commutativity diagram below.

$$\begin{array}{ccc} U & & U \\ & \downarrow & \\ V & & V. \end{array} \quad (2.3)$$

To understand the diagram we note that if we travel “right and then down,” the diagram instructs

us to use T (top arrow right) first, followed by ϕ . (right arrow downwards). Consequently the pathway amounts to finding $\phi(T(u))$. If we proceed "down and then right", the diagram instructs us to apply ϕ (left arrow downwards) first, and then apply S (bottom arrow right). This then amounts to finding $S(\phi(u))$. When the relationships in this diagram hold, we say $\phi(T(u)) = S(\phi(u))$, and we formally denote it as $\phi \circ T = S \circ \phi$.

Definition 9. [Conjugacy & Semi-Conjugacy] Consider the dynamical systems $T : U \rightarrow U$, $S : V \rightarrow V$ and suppose the relationship $\phi \circ T = S \circ \phi$ holds. If $\phi : U \rightarrow V$ is a homeomorphism, then S is said to be conjugate to T and ϕ is called a conjugacy. If we relax the criterion on ϕ and merely require ϕ to be continuous and surjective where $\phi : U \rightarrow V$, then ϕ is a semi-conjugacy between T and S where T has domain U and S has domain V ; S is said to be semi-conjugate to T .

EdN:2

² When S is conjugate to T , the dynamics of the two systems are in some way "dynamically equivalent". Specifically, they are in one-to-one correspondence with one another. However when S is semi-conjugate to T with $\phi : U \rightarrow V$ a many-to-one mapping, the dynamics on V provide merely a coarse-grained description of the dynamics on U [?]. When S is semi-conjugate to T , it is also common to call S a *factor of T* , or conversely that T is an *extension of S* . In essence, an extension (e.g., [?]) is a larger system capturing all of the important dynamics of its factor.

It is a very hard or nearly an impossible task to establish the existence of a conjugacy or a semi-conjugacy ϕ between two systems [?]. However, one can verify that a function ϕ satisfies the commutativity diagram ??.

Example 5. For example $\phi(x) = \sin\left(\frac{\pi}{2}x\right)^2$ is a conjugacy between two systems $Tu = 1 - |2u - 1|$ and $Sv = 4v(1 - v)$ defined on $U = [0, 1]$.

Indeed, $\phi : [0, 1] \rightarrow [0, 1]$ is a homeomorphism and by employing simple trigonometric identities, it is easily proved that ϕ satisfies ??

$$\begin{aligned} (\phi \circ T)u &= \sin^2\left(\frac{\pi}{2} - \frac{\pi}{2}|2u - 1|\right) = \sin^2(\pi u) = \left(2 \sin\left(\frac{\pi}{2}\right) \cos\left(\frac{\pi}{2}\right)\right)^2 = 4 \sin^2\left(\frac{\pi}{2}u\right) \left(1 - \sin^2\frac{\pi}{2}u\right) \\ &= (S \circ \phi)u. \end{aligned}$$

And it follows that ϕ is a conjugacy between the two systems.

In establishing that two systems are conjugate (semi-conjugate) to one another, we have also shown that one may choose to work with one system as opposed to the second and still be guaranteed to obtain information on the latter. This is especially useful while forecasting the future evolution of dynamical systems.

²EDNOTE: M: Saying ϕ is then a semi-conjugacy between T and S could be ambiguous as it does not specify which system is on the top in the commutativity. So please edit accordingly. B: Does defining explicitly the domain and codomain of ϕ make it clear enough?

Chapter 3

The Learning Problem

In this chapter we acquaint ourselves with the question of forecasting dynamical systems with unknown underlying structures, state and discuss the shortcomings of the Takens delay embedding theorem and discuss various issues faced while forecasting.

Consider a relatively simple learning problem: Given the sequence (u_0, u_1, \dots, u_m) for $m \in \mathbb{Z}$, a finite segment of an orbit of the map T where T is defined by the update equation $u_{n+1} = Tu_n$, forecast the values u_{m+1}, u_{m+2} where the map T is unknown, given that u_m .

A practical example of this would be the subsequent scenario: given the time-sequential coordinates of an object moving in space, predict the future positions of that object. However we are very rarely, if at all, presented with a problem where the entire state information is available to us in the form u_n at some time-step $n \in \mathbb{N}$. Consider then a more involved learning problem:

Suppose we only have the observations $\theta(u_0), \theta(u_1), \dots, \theta(u_m)$ of the true system states u in an unknown dynamical system (U, T) and we wish to predict the values $\theta(u_{m+1}), \theta(u_{m+2})$ and so forth.

First we establish the method of Takens delay embedding. In this method, one considers a discrete-time dynamical system defined on a 'nice' space (a smooth manifold)³ that can be obtained as time- K map of a continuous time-dynamical system, concepts which will be formalised below. We make observations from such a system, i.e., we observe the evolution of $\theta(w_0)$ (where w_0 represents the initial state) i.e., we observe a finite set of values of $u_n := \theta(w_n) = \theta(Tw_{n-1})$. The sequence $\{u_n\}$ represents a scalar time-series and intuitively θ may be thought to represent a probe inserted into a larger system which is itself only measuring/extracting a small part of the greater system state at time $t = n$. Consider for example a thermometer erected to measure the ambient temperature in a local village. This measurement function, the thermometer, is capturing only a single aspect, the

EdN:3

³EDNOTE: Should this manifold be further explained?

temperature, of a much grander dynamical system entailing the present weather of the surrounding area. Even more than that though- it is measuring a minuscule part of the global weather system.

However, the true state u of a system is seldom, if ever, fully known. In almost all cases we can at most insert probes into a system to obtain partial information by means of the measurements taken. Moreover, the process of taking a measurement itself introduces 2 additional aspects which complicate the problem:⁴

- i. A series of measurements over a specific time-interval is inherently a discretisation procedure of the underlying continuous-time dynamical system. (Hence why we restricted our attention to discrete-time systems in the preceding section)
- ii. A probe will never be fully accurate, and so the act of measurement introduces a certain measure of numerical noise/inaccuracy.

From here we construct a multidimensional observable using the method of stacking previous observations, i.e., we create delay-coordinate map defined by $\Phi_{k,\theta}(w) := (\theta(T^{-k}w), \dots, \theta(T^{-1}w), \theta(w))$. The concept of an observable is understood in the sense of an observable as originally introduced by [?, ?] to refer to a probe or measurement function inserted into the system. The essence of Takens' theorem relates to the fact that when k is sufficiently large, we can define a dynamical system on the space \mathbb{R}^{k+1} whose states are $\Phi_{k,\theta}(w)$, $\Phi_{k,\theta}(Tw)$, $\Phi_{k,\theta}(T^2w)$, etc. and this dynamical system is topologically conjugate to the unknown underlying system (W, T) . We recall the Takens delay embedding theorem next.

3.1 Takens Embedding Theorem

We define first the concepts of homeomorphism and embedding:

Definition 10. [Homeomorphism] A homeomorphism is a function $f : Z \rightarrow Y$ between two topological spaces Z and Y that is continuous, bijective and has a continuous inverse.

Definition 11 (Embedding). Consider a homeomorphism $f : Z \rightarrow Y$ for $Y \subset X$. Z is said to be embedded in X by f .

Takens Theorem states a result establishing a relationship between the observed and underlying dynamical systems by showing that the concatenation of a sufficiently large number of previous observations into a vector will, under certain conditions, generate a map between the vectors from the respective systems. We formulate the theorem from [?].

⁴EDNOTE: M: Maybe replace difficulty by saying that we need to consider two aspects. B: Perhaps this edit makes more sense?

Theorem 1 (Takens Embedding Theorem (adopted from [?])). Let W be a compact manifold of dimension m , and $d \geq m$ so that $2d$ is an integer. It is a generic property for the pair (T, θ) , where $T : W \rightarrow W$ is a smooth diffeomorphism, and $\theta : W \rightarrow \mathbb{R}$ a smooth function, the map $\Phi_{2d,\theta} : W \rightarrow \mathbb{R}^{2d+1}$ defined on W by $\Phi_{2d,\theta}(w) := (\theta(T^{-2d}w), \dots, \theta(T^{-1}w), \theta(w))$ is a diffeomorphic embedding; by ‘smooth’ we mean at least C^2 . Consequently, there exists a map $F_\theta : \Phi_{2d,\theta}(W) \rightarrow \Phi_{2d,\theta}(W)$ defined by

$$F_\theta : (\theta(T^{-2d}w), \dots, \theta(T^{-1}w), \theta(w)) \mapsto (\theta(T^{-2d+1}w), \dots, \theta(w), \theta(Tw))$$

so that (W, T) is topologically conjugate to $(\Phi_{2d,\theta}(W), F_\theta)$.

By generic we mean a residual set on a certain topology on appropriate spaces of functions (that we do not describe here) (**Cite**). By C^2 , we make reference to a twice-differentiable function with a continuous second derivative. In our scenario (and this remains important throughout) the input space U is considered the attractor.

Below we reproduce Figure 1 from [?].

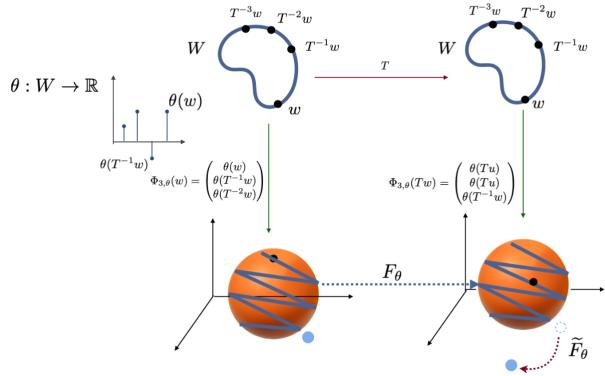


Fig. 1. Schematic of Takens Embedding Theorem’s usage and impact of iterates slipping from the attractor when the approximation \tilde{F}_θ of F_θ is learnt.

If we can learn the map F_θ , from sufficient number of data points of $\{\phi_{2d,\theta}(w_n)\}$, then we also know how to forecast how $\theta(w_n)$.

3.2 Practical Limitations to Takens’ Theorem

It is a fact that Takens’ Embedding Theorem is both a powerful result and that it provides compelling reason to believe that one could conceivably reconstruct accurately a system conjugate to the underlying system. Nonetheless, it does present some serious practical limitations:

1. Even supposing that we could indeed find F_θ , our *approximation* of F_θ is a map from a larger set \mathbb{R}^{2d+1} containing the embedded attractor. There are, however, no theoretical guarantees that F_θ will retain $V = \Phi_{2d,\theta}(W)$ as an attractor although W could itself be an attractor.
2. Takens theorem is stated only for noiseless observations. Due to noise ϵ_n the delay vector $\Phi_{2d,\theta}(w_n) + \epsilon_n$ may lie outside V . Furthermore, due to the chaotic nature of the underlying system (i.e. the fact that it has SIDC), the evolution of $\Phi_{2d,\theta}(w_n) + \epsilon_n$ under the map F_θ could move out of V completely. This problem can be overcome by using a driven dynamical system with some properties, and we discuss this in Chapter??.

EdN:5 The Takens embedding theorem does not guarantee global dissipativity. ⁵

EdN:6 ⁶

Though not a uniquely related to the Takens embedding theorem, we do opt to make mention of this requirement for our work. The methods described here are applicable for data originating from a surjective map. This follows from the fact that when T maps the space U into a proper subset B of U , we can conclude that T is not surjective as there exists some point in $U - A$ which does not have a preimage. An example of this would be the case when a system displays energy-loss. Consider, for instance, consider the oscillations dying down for a damped pendulum. (A more in-depth account will be given in ??)

EdN:7 This requirement is not overly restrictive, as a great number of chaotic systems do have a surjective map. (Cite) ⁷

⁵EDNOTE: B:Links with ?? in discussion of dissipativity of Γ . Sir, I'm not sure how to write this here - I think it a beneficial point to make so as to set the stage for the later discussion in ??, but I am unsure there as well.

⁶EDNOTE: Include note on learning failure because of sensitivity as in graph commented out?

⁷EDNOTE: B: I remember you saying this in one of our sessions, but don't have a source for it.

Chapter 4

Driven Dynamical Systems to Forecast Problems

In this chapter, we discuss results pertaining to the mapping of temporal data obtained from a discrete-dynamical system onto a different space through the notion of a driven dynamical system. We also consider the conditions a driven system should possess to avoid adding distortion to its state-space representation. We then describe the *single-delay dynamics* (SDD) of the system and finally recall conditions on the driven system so that the SDD are conjugate (or at least semi-conjugate) to the underlying system. The SDD can then be used to forecast and reconstruct the underlying system.

4.1 Nonautonomous and Driven Dynamical Systems

Definition 12 (Nonautonomous Dynamical System (adopted from [?])). A nonautonomous dynamical system (NDS) on a space X is simply a dynamical system comprising a family of maps $\{g_n\}_{n \in \mathbb{Z}}$ where $g_n(\cdot) := g(u_n, \cdot) : X \rightarrow X$ is continuous. Each g_n arises as a consequence of an input u_n from the input space U , a topological space.

We immediately recall the figure first defined in (??) and remind ourselves of our ultimate goal to learn a system that is conjugate to an underlying or unknown dynamical system (U, T) by making use of the measurements obtained from the unknown system.

$$\begin{array}{cc} U & U \\ V & V. \end{array}$$

To find a function ϕ , the need now arises to consider a so-called driven dynamical system, one that can generate a nonautonomous dynamical system, where two spaces are considered: input is taken from an input space U and the state of a space X is updated. (Previously, only the state was considered.) We do this to mimic the true conditions. By accounting for an extogenous input sequence in U which can also influence the system-state at a specific time-step, it produces more general models than an autonomous system.

Definition 13 (Driven, Compactly Driven Dynamical System). A driven dynamical system comprises two topological metric spaces U, X and a continuous function $g : U \times X \rightarrow X$ where $g(u_n, x_n) = x_{n+1}$. If the input space U is compact, we refer to the system as compactly driven.

The dynamics on X are generated by the update equation $x_{n+1} = g(u_n, x_n)$ where $n \in \mathbb{Z}$, input u_n from U and state x_n belonging to X , where X is compact. Abbreviated, we shall refer only to the *driven system* g , with all other entities being understood implicitly.

Notably, a nonautonomous dynamical system may be generated from U ; any input \bar{u} , a bi-infinite sequence from U , gives rise to the sequence of self-maps $\{g(u_n, \cdot)\}_{n \in \mathbb{Z}}$ contained in X . Physically, one may think of this bi-infinite sequence as referring to a system that has been running for a long period at the time of the first measurement taken from the system (alternatively, the first time a probe is inserted into the system to take an observation)

Definition 14 (Entire Solution). A sequence $\{x_n\}_{n \in \mathbb{Z}} \subset X$ is called an entire solution (or simply a solution) of the driven system g with input \bar{u} when it satisfies

$$g(u_{n-1}, x_{n-1}) = x_n$$

for all $n \in \mathbb{Z}$

It is important to emphasise that a sequence $\{x_n\}$ satisfying the update equation above can only be a solution if $x_n \in U$ holds for all $n \in \mathbb{Z}$ and just for $n > 0$. Consider the example below.

Example 6. The only solution $\{x_n\}_{n \in \mathbb{Z}}$ to the driven system $g(u, x) = \frac{ux}{2}$, where $X = [0, 1]$, $U = [0, 1]$, is the zero solution $x_n \equiv 0$. To see this, consider any $x_n = a \in [0, 1]$ where $a \neq 0$. Let $\bar{u} \in U$ be a non-zero constant sequence, say $u_n = 0.5$. The driven system may be rewritten as $x_{n-1} = \frac{2x_n}{u_{n-1}} = 4x_n$ and the iterates of x_n in backward time will increase by a factor of 4 at each timestep. Thus for some $m \leq n$, $1 < x_m$ i.e. $x_m \notin X$. So $\{x_n\}_{n \in \mathbb{Z}}$ is not a solution and it follows that the only possible solution is the zero solution.

A system may also have multiple solutions as is evidenced in the example below.

Example 7. Consider the driven system $g(u, x) = x^2$ for $X = [0, 1]$, $U = \mathbb{R}$. The system has an uncountable number of solutions, as there exists a solution for every $x \in X$ which also passes through the point x and $\lim_{n \rightarrow \infty} x_n = 0$, $\lim_{n \rightarrow -\infty} x_n = 1$. The proof is deferred to immediately after the next paragraph (see ??).

As the solutions to a driven system are often considered to be an important entity [?], we next identify a subspace X_U of X that contains all possible solutions. To realize such a subspace of a driven system g , the concept of a reachable set is defined.

Definition 15 (Reachable Set). The reachable set of a driven system g is exactly the union of all the elements of all the (entire) solutions, i.e.,

$$X_U := \left\{ x \in X : x = x_k \text{ where } \{x_n\} \text{ is a solution for some } \bar{u} \right\}.$$

The set of all reachable states at a specific time n for input \bar{u} is denoted by $X_n(\bar{u})$

The reachable set itself can be defined independently from the fact that g does or does not have the USP. We note that $x \in X_n(\bar{u})$ if and only g has a solution $\{x_k\}$ for $x_n = x$ and input \bar{u} . (Cite). The set $X_n(\bar{u})$ is exactly the set of all points x that have some entire solution Ψ passes through it at time n .⁸

EdN:8

We can give an alternate formulation of the set $X_n(\bar{u})$: suppose, for some fixed input \bar{u} , we define $g_i = g(u_i, \cdot)$ for all $i \in \mathbb{Z}$, then the set of states

$$X_{n,i}(\bar{u}) := g_{n-1} \circ \cdots \circ g_{i+1} \circ g_i(X). \quad (4.1)$$

As the image of a finite composition of continuous maps, the set $X_{n,i}(\bar{u})$ is compact whenever X is compact and is also nonempty. Further, $X_{n,i}(\bar{u}) \supset X_{n,i-1}(\bar{u})$. Hence $\bigcap_{i < n} X_{n,i}(\bar{u})$ is a nested intersection of closed nonempty subsets, and whenever X is compact, the intersection is nonempty.⁹

EdN:9

Definition 16 (Topological Contraction). We say g is a topological contraction if $\bigcap_{i < n} X_{n,i}(\bar{u})$ is a singleton subset of X for each \bar{u} .

It turns out that $\bigcap_{i < n} X_{n,i}(\bar{u})$ is identically equal to $X_n(\bar{u})$ that we had defined earlier¹⁰. This will lead to a result (for. e.g., [?]) that g being a topological contraction is equivalent to the existence of exactly one entire solution.

EdN:10

⁸EDNOTE: B: Changed sentence for better readability - still understandable?

⁹EDNOTE: B: Should I cite here?

¹⁰EDNOTE: (Insert citations.)

Remark 1 (Proof of Example ??). For the driven system, $g(u, x) = x^2$ the evolution does not depend on the input at all. Hence $g(u, x)$ can be written as a single map $f(x) = x^2$ which is a homeomorphism on $[0, 1]$. A left-infinite orbit is defined and it converges to 1 while the right-infinite orbit converges to 0. Or in other words the iterates of f^{-1} converges to 1 while the iterates of f converge to 0

Thus far it has been demonstrated that a system may have one or more solutions; one may ask if a driven system always has a solution and, if so, whether it satisfies certain properties such as uniqueness. Should the driven system be compact, existence follows immediately.

Theorem 2. *Let g be a driven system. If X is compact, then for each input \bar{u} , there exists at least one solution to the driven system $g(., x)$*

Proof. May be found in [?, ?, ?]

4.2 Unique Solution Property

Definition 17 (Unique Solution Property). A driven system g is said to have the Unique Solution Property (USP) if for each input \bar{u} there exists exactly one solution. Alternatively we may formulate the USP as follows: g has the Unique Solution Property if there exists a well-defined map $\Psi : U \rightarrow X$, with $\Psi(\bar{u})$ denoting the unique solution.

One of the first result obtained after defining the USP is the fact that every solution will attract different initial conditions towards the component parts of the solution. If g has the USP, then any solution to g is also a uniform attractor in nonautonomous dynamical systems literature [?]. The discussion on nonautonomous attractors is beyond the scope of this project and we refer the reader to [?, ?] for further reading.

EdN:11

¹¹ Reservoir Computing(RC) refers to a machine learning approach in which input is mapped onto a different, but higher dimensional, space through a system called the reservoir after which a readout measuring the reservoir state is trained. In the majority of Reservoir Computing(RC) literature (**Cite**), a notion of forgetting the states of the reservoir asymptotically is often used. Concepts like the echo state property(ESP) and fading memory [?] are two of these. If g possesses the ESP (equivalent in our context to the USP (**Cite**)), we are guaranteed that the whole of an input's left-infinite history will exactly determine the current system state; i.e. there is only ever the possibility of a single reachable state at a given instance of time [?, ?]. Beyond forgetting the past states, the ESP is also often discussed in terms of a stability property that plays a key role in the robustness of recurrent neural networks(RNN). The concept, termed input-related stability [?], refers the phenomenon of nearby inputs creating responses (solutions) close to one another.

¹¹EdNOTE: B: RC introduced a bit abruptly. Adjust.

4.3 Establishing Conjugacies

Having already defined the reachable set X_U as the collection of all elements of all entire solutions, we pause briefly in order to fix additional notation. Defining $\bar{u}^n := (\dots, u_{n-2}, u_{n-1})$ as the left-infinite subsequence of an input up until time n , \overleftarrow{U} is then the notation for all these left-infinite sequences in U . Moreover, $\bar{u}^n v := (\dots, u_{n-2}, u_{n-1}, v)$ will symbolise the input up to time n with $v \in U$ being the specific input value at time n . The introduction of a new input at time n can be described by the mapping $\sigma_v : \bar{u}^n \mapsto \bar{u}^n v$. The right-shift map, $r : \overleftarrow{U} \rightarrow \overleftarrow{U}$, of an input sequence is defined $r : (\dots, u_{-2}, u_{-1}) \mapsto (\dots, u_{-3}, u_{-2})$

The goal now assumes the form of the question on our ability to establish a semi-conjugacy as presented below for the driven system g .

$$\begin{array}{ccc} \overleftarrow{U} & & \overleftarrow{U} \\ X_U & & X_U. \end{array} \quad (4.2)$$

We proceed to consider a specific subclass of conjugacies.

Definition 18 (Universal Semi-Conjugacy). Given a driven system g , we call a continuous and surjective map $h : \overleftarrow{U} \rightarrow X_U$ a universal semi-conjugacy if diagram ?? commutes for all $v \in U$.

If the universal semi-conjugacy h exists (i.e. the diagram in ?? commutes) then the solution $\Psi(\bar{u})$ will intuitively have no more ‘complexity’ than the input \bar{u} .

But does such a function h for the above driven system g above exist? Whenever g has the USP and $\Psi(u) = \{x_n\}_{n \in \mathbb{Z}}$ it follows that h , defined by $h(\bar{u}_n) := g(u_n, x_{n-1}) = x_n$, will satisfy the semi-conjugacy in the graph above ?. Regrettably, such a continuous mapping h is not guaranteed to exist [?] when g does not have the USP [?, Lemma 5]. Note that even when h does exist, we are not guaranteed its injectivity. Considering again example ??, we see that even if h were to exist, it could not be injective as $X_U = \{0\}$.

Re-sketching the commutativity diagram ?? above by replacing X_U by its left-infinite sequence space \overleftarrow{X}_U , we obtain the diagram below. In this case, the function $H : \overleftarrow{U} \rightarrow \overleftarrow{X}_U$, a map that is both continuous and surjective, is called a *causal mapping* that is defined next.

Definition 19 (Causal Mapping). A continuous, surjective map $H : \overleftarrow{U} \rightarrow \overleftarrow{X}_U$ such that

$$H \circ \tilde{g}_v = \sigma \circ H$$

holds for all $v \in U$ where \tilde{g}_v maps (\dots, u_{-2}, u_{-1}) to $(\dots, u_{-2}, u_{-1}, g(v, u_{-1}))$ is called a causal mapping.

$$\begin{array}{cc} \overleftarrow{U} & \overleftarrow{U} \\ \overleftarrow{X}_U & \overleftarrow{X}_U. \end{array} \quad (4.3)$$

Theorem 3. For a compactly driven system, a causal mapping H exists if and only if g has the USP.

Proof. See [?, Th.3].

The driven system g can induce an embedding of \overleftarrow{U} in \overleftarrow{X} as follows: If the causal mapping $H : \overleftarrow{U} \rightarrow \overleftarrow{X}_U$ is injective (in addition to being surjective), it becomes the embedding of \overleftarrow{U} in \overleftarrow{X} induced by the driven system g . The continuity of H^{-1} follows from the fact that H is itself a continuous and surjective mapping of a compact space \overleftarrow{U} in a Hausdorff space. and we refer to the map H as a *causal embedding*.

When g has the USP, the diagram below (adopted from [?]) illustrates the operation of the mappings h and H . The mapping $h : \overleftarrow{U} \rightarrow X_U$ is also considered an observable as mentioned in the introduction of Chapter ??).

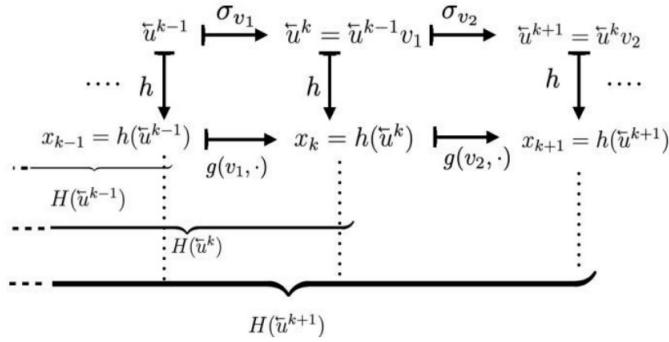


Fig. 1. Ladder-like behaviour of h and H to illustrate the causal mapping. (Figure reproduced from [?])

The function H contains a countably infinite number of coordinate (or factor maps) since its codomain is \overleftarrow{X}_U . As it is not practically feasible to consider a left-infinite sequence in X_U (in other words an element of \overleftarrow{X}_U) for learning, we wish to restrict the inputs we consider to a subspace of \overleftarrow{U} . Particularly, we restrict our attention to the so-called inverse-limit space of a dynamical system and hope to embed it within some finite product of X .

The left-infinite orbit space is significant because a great amount of information about the original system can be gleaned from its topological structure [?, ?]. The formal topological structure for the inverse-limit space of a chaotic dynamical system is complicated, and further discussion is relegated to [?, ?]. Broadly, we may conceive of the inverse-limit system of a dynamical system as

a subspace of an infinite-dimensional space where each point in the inverse-limit space corresponds to a backward orbit of the map T . Following [?, ?] we provide a definition:

Definition 20 (Inverse-Limit Space, Inverse-Limit System). The subset $\widehat{U}_T \subset \overleftarrow{U}$ defined by

$$\widehat{U}_T := \{(\dots, u_{-2}, u_{-1}) : Tu_i = u_{i+1}\}$$

is called the inverse-limit space of (U, T) . The self-map \widehat{T} induced by T on \widehat{U}_T is defined by

$$\widehat{T} : (\dots, u_{-2}, u_{-1}) \mapsto (\dots, u_{-2}, u_{-1}, T(u_{-1}))$$

and the resulting dynamical system $(\widehat{U}_T, \widehat{T})$ is called the inverse-limit system of (U, T) .

The inverse-limit space is well-defined since $T : U \rightarrow U$ is surjective by assumption. We now get to the concept of embedding the inverse limit-space of the dynamical system in $X \times X$.

Definition 21. We say a driven system g *causally embeds* a dynamical system (U, T) if it satisfies the two properties: (i) a universal semi-conjugacy exists and (ii) $H_2(\tilde{u}) := (h(r\tilde{u}), h(\tilde{u}))$ embeds the inverse-limit space \widehat{U}_T in $X \times X$.

It is imperative to take note of a subtlety here: we refer to the *map H* as a (causal) embedding, whereas the driven *system g* (causally) embeds another *system (U, T)*.

4.4 Choosing the driven system g

So far it is not clear if the USP alone ensures the ability to causally embedding a dynamical system and we mention that a driving function g cannot just be chosen in a frivolous manner. This could possibly complicate our work.

In the above example(??), the input's temporal variation could not be related to the reachable set as X_U only consisted of a single element; little, if not no, information is encoded. To obtain a suitably complex function g , it is thus desired that the reachable set of a driven system be large enough to relate to the input. This is true even for embedding the inverse-limit space of a dynamical system.¹²

EdN:12

To this end, we recall the notion of State-Input (SI) Invertibility.

Definition 22 (SI-Invertibility). A driven system g is said to be SI-Invertible if $g(*, x) : U \rightarrow X$ is invertible for all $x \in X$. Alternatively it may be said that if, given x_n and x_{n-1} , u_{n-1} can be uniquely determined from $x_n = g(u_n, x_{n-1})$, then g is said to be SI-invertible.

¹²EDNOTE: B: Expand or refer?

SI-invertibility means that one's choice of g will still ensure 'enough' information is retained without introducing unwanted complexity. 'Enough' here refers to the fact that we may always recover the previous input value u_n (if we know successive states x_{n-1}, x_n).

Subsequently we define the relation Y_T induced by (U, T) on $X_U \times X_U$ for a driven system g possessing SI-invertibility. To describe the SDD formally, we consider a dynamical system $T : U \rightarrow U$ and define a relation on the reachable set X_U , i.e. a subset on $X_U \times X_U$ defined by

$$Y_T := \{(x_{n-1}, x_n) : \{x_k\}_{k \in \mathbb{Z}} \text{ is a solution for some orbit of } T \text{ and } n \in \mathbb{Z}\}.$$

The following theorem establishes the existence of a well-defined map G_T describing the SDD of the system above.

Theorem 4. *For an SI-invertible driven system g and a dynamical system (U, T) , the map $G_T : Y_T \rightarrow Y_T$ defined by the relation $(x_{n-1}, x_n) \mapsto (x_n, x_{n+1})$ is well-defined. (This results holds even in the absence of g possessing over the USP)*

Proof. See [?, Th.3]

At this stage it is worth taking note of a specific driven system in the form of a discrete state-space model which has acquired some adherence in applications [?]- especially those pertaining to Echo State Networks and the ESP. The function

$$g(u, x) = (1 - a)x + a\overline{\tanh}(Au + \alpha Bx) \quad (4.4)$$

is SI-invertible and, if αB has a spectral norm < 1 , also possesses the USP [?, Th.2]. It is easy to show that g is SI-invertible by recovering u_n in

$$u_{n-1} := A^{-1} \left(\overline{\tanh}^{-1} \frac{1}{a} \left(x_{n+1} - (1 - a)x_n \right) \right) - \alpha Bx_n \quad (4.5)$$

when x_{n-1} and x_n are known.

This specific driving function g is used in our implementation and is discussed more completely in chapter ??

Despite the ease that with which one manipulates a left-infinite history in the realm of theory, it is impossible to obtain or use such a sequence in any real-life application. Fortunately, one does not need the entire left-infinite history of an input in practice thanks to the Uniform Attraction Property(UAP). We use an alternate version of the definition of UAP – since the notion of nonautonomous attractors in this thesis would take some time to establish and detracts from the principal thrust of this project, we refer to [?].

Definition 23 (Uniform Attraction Property). A driven system g has the uniform attraction property (UAP) if we initialize the driven system with an arbitrary initial value $y_m \in X$, and the sequence $y_{m+1}, y_{m+2}, y_{m+3}, \dots$ satisfying $y_{k+1} = g(u_k, y_k)$ for $k \geq m$ then approximates an actual solution $\{x_n\}$ uniformly in the sense that given $\epsilon > 0$ (independent of y_m) there is an integer n so that $d(x_{n+i}, y_{n+i}) < \epsilon$ for all $i \geq 0$, where $\{x_m\}$ is a solution.

The UAP guarantees that all trajectories will converge to the same trajectory as time moves forward. An illustration of this is shown in Fig. ?? where the trajectory in red locks on to the supposed actual solution when the system has the USP.

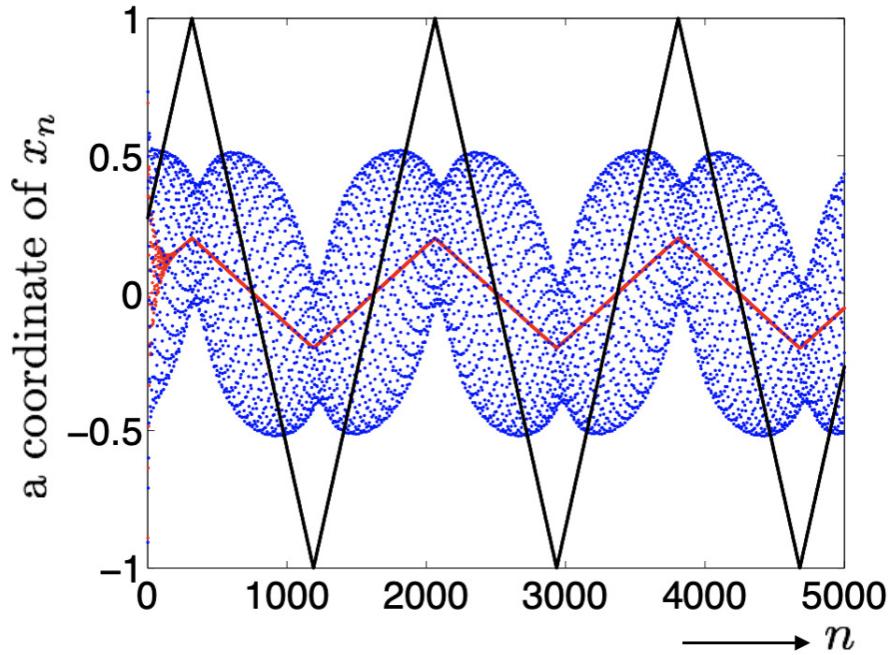


Fig. 2. A coordinate of a simulated solution $(x_0, x_1, \dots, x_{5000})$ of a RNN plotted in red (with parameter $\alpha = 0.99$ (where g has the USP) and blue ($a = 1$ and $\alpha = 1.05$ where g does not have the USP) while the matrices A and B are randomly generated, and B has unit spectral radius. Reproduced from [?].

One may now appreciate an even more astounding result: g having the USP is equivalent to the UAP as is proved in [?].

4.5 The next step in Dynamics

We are interested in determining whether \widehat{T} and G_T are related. For this, we introduce the function $H_2 := (h(r\bar{u}, h(\bar{u})))$ mapping the entirety of a left-infinite sequence to some element in $X \times X$.

¹³EDNOTE: B: How can I explain the black line? I don't think this makes sense to the reader immediately.

Indeed, it may be shown that g having SI-invertibility and the USP immediately guarantees the existence of at least a semi-conjugacy between the systems (Y_T, G_T) and $(\widehat{U}_T, \widehat{T})$. This is formalised in the Causal Embedding Theorem, which we state next:

Theorem 5 (Causal Embedding Theorem (adopted from [?])). *Let g be a driven system with SI-invertibility and the USP. Let h denote the universal semi-conjugacy and $H_2(\overleftarrow{u}) := (h(r\overleftarrow{u}), h(\overleftarrow{u}))$, where r is the right-shift map. Let $(\widehat{U}_T, \widehat{T})$ be the inverse-limit system of a dynamical system (U, T) . Then the restriction of H_2 to \widehat{U}_T is a topological semi-conjugacy between the inverse-limit system $(\widehat{U}_T, \widehat{T})$ and the induced dynamical system (Y_T, G_T) , i.e., the following diagram commutes*

$$\begin{array}{ccc} \widehat{U}_T & & \widehat{U}_T \\ \downarrow & & \downarrow \\ Y_T & & Y_T. \end{array} \quad (4.6)$$

or in other words, (Y_T, G_T) is a factor of $(\widehat{U}_T, \widehat{T})$. Further, if $T : U \rightarrow U$ is a homeomorphism, then H_2 embeds \widehat{U}_T in $X_U \times X_U$, and hence (Y_T, G_T) is conjugate to $(\widehat{U}_T, \widehat{T})$.

Proof. May be found as the proof of [?, Th.4]

Recall that H_2 maps an entire left-infinite solution sequence from Ψ to an element in $X \times X$. We are drawing nearer and nearer to our principal target and our results carry more and more weight: If we are able to learn G_T , we will also have learnt \widehat{T} ; and since \widehat{T} is an extension of T , we will have obtained T as well.

Summarising the discussing thus far:

It is easy to lose the birds-eye view, so we take a moment to review our progress up until this point.

1. We are interested in a some dynamical system (U, T) with unknown dynamics.
2. To determine properties about this system (U, T) and predict its future evolution, we determine the dynamics of the inverse-limit system $(\widehat{U}, \widehat{T})$.
3. If the driven system g is SI-invertible (and $\{u_n\} \in U$ is an orbit of T), the map G_T exists that describes the single delay dynamics.
4. If, furthermore, g has the USP, (Y_T, G_T) is semi-conjugate to $(\widehat{U}, \widehat{T})$.
5. If we can assume that T is a homeomorphism, (Y_T, G_T) is topologically conjugate to $(\widehat{U}, \widehat{T})$, an extension space of (U, T)

One can therefore in practice learn the SDD of the driven system states via G_T with enough data thanks to the USP/UAP. This enables us to do at least two things:

- Forecast x_{n+1}, x_{n+2}, \dots via iterates of G_T (if G_T can be learnt).
- Forecast future values of u_n since x_n and x_{n+1} determine u_n since g is SI-invertible.

Finally, we note that although G_T exists with SI-invertibility, we need the USP as well (see ?? below for a visual illustration). If not, the driven states would have to be running for all time since they would not have forgotten their past states.

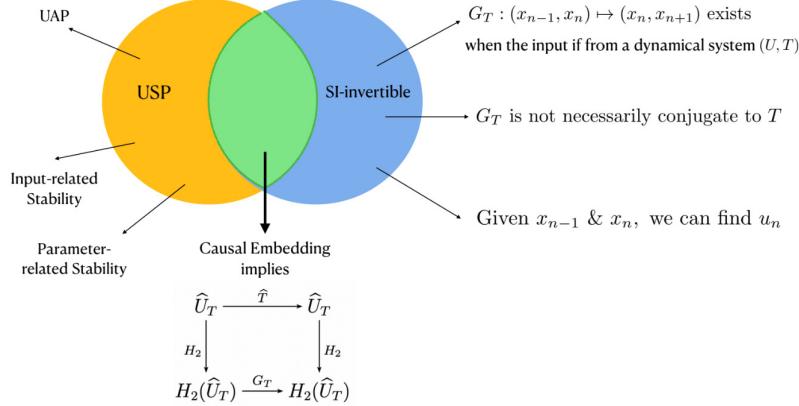


Fig. 3. Pictorial summary of results obtained up to now and how they overlap. Reproduced from (Cite SIG-25May)

4.6 A discussion of G_T

In the above sections, we established the map G_T describing the SDD of a driven system. By establishing the existence of this map, we've essentially embedded the attractor U (recall from ??) into the higher dimensional space $X \times X$. In layman's terms, this ensures that there is more "dimensional room" for the underlying system's underlying dynamics to "move". Since the dynamics aren't as "squashed", we might therefore hope that the dynamics of G_T are in some sense simpler than that of T . (Taken note of the fact that G_T is a homeomorphism even when T is just continuous)

In [?] it is illustrated in an empirical fashion in that the map G_T describes dynamics which are less functionally complex than that of T or of the map $\Phi_{2d,\theta}$. This is done by implementing a Recurrent Neural Network (RNN), which is discussed in ??.¹⁴

EdN:14

We opt to learn G_T in an indirect manner by defining a new map $\Gamma : (x_{n-1}, x_n) \mapsto u_n$. It will follow immediately from G_T 's existence that Γ also exists.

The reasons for taking this roundabout approach remain to be discussed in ??, but a pat answer

¹⁴EDNOTE: B: Should I add in that we'll be using Pearson coefficient?

may immediately be given: When Γ has been learnt, the system can be driven autonomously and then G_T is known anyway. We formalise this with a theorem.

Theorem 6. *When x_n and x_{n-1} are successive points on a solution obtained for input-orbit of $T\{u_n\}$, then the map $\Gamma : X \times X \rightarrow U$ defined by $(x_{n-1}, x_n) \mapsto u_n$ exists whenever G_T exists*

Proof. See [?, Th. 3c].

Projection mappings π_i are defined in the traditional meaning where a k -dimensional vector is projected to its i^{th} component such that

$$\pi_i : (a_1, a_2, \dots, a_{k-1}, a_k) \rightarrow a_i$$

The graph in equation ?? is then extended as below:

$$\begin{array}{ccc}
 \widehat{U}_T & & \widehat{U}_T \\
 & \downarrow & \downarrow \\
 Y_T & & Y_T \\
 & \downarrow & \downarrow \\
 & \color{red} U \times X &
 \end{array} \tag{4.7}$$

The problem finally simplifies to the issue of learning the map Γ and combining this with the projection mapping π_2 and the function g , which will be known. A final set of equations is obtained - equations that have been entirely constructed from data.

$$u_{k+1} = \pi_1 \circ (\Gamma, \pi_2) \circ (\pi_2, g)(u_k, x_k) \tag{4.8}$$

$$x_{k+1} = \pi_2 \circ (\Gamma, \pi_2) \circ (\pi_2, g)(u_k, x_k). \tag{4.9}$$

4.7 Advantages of learning Γ

One may immediately ask why we opt to take such a roundabout route; why not just learn the map G_T from the get-go? On the surface, this seems to be an arbitrary decision path with no real reasoning, so we take a pause again and discuss the motivation for learning Gamma. There are a number of distinct advantages.

In the first place, learning Γ saves computational resources. This is due to the fact that the input u_n lies in a lower-dimensional subspace in comparison to the high-dimensional space $X \times X$.

Secondly, the function Γ is known to be stable. Learning Γ makes use of equations ?? which in turn employs the driven system g possessing the USP, and offers distinct advantages with regards to stability in the presence of perturbation. It is desirable that inputs that differ only slightly would have only ‘slight’ effects on the output of the system. Moreover, one would hope to prevent numerical errors originating from input and measurement noise (see [?, Th. 5]). As the data fed into the system is a function of both the input given and the parameters of the system itself (i.e. the value a , α in \tanh), we exploit both input-related and parameter-related stability. Both these notions are defined by way of the continuity of an encoding map of an input and design parameter respectively in [?].¹⁵

EdN:15

Informally, we may consider input-related stability as concerned with the question of whether or not small variations in input would result in small responses. If two inputs \bar{u} and \bar{u}_{noisy} are close in the product topology, then their tails could differ greatly due to the metric that generates the product topology has insensitivity to the differences in sequence’s tails. However if the system has the USP then the function h is continuous and so $h(\bar{u}), h(\bar{u}_{noisy})$ remain close-by for small levels of noise maintaining a measure of robustness to input and measurement noise[?]. In a similar fashion, parameter-related stability is obtained thanks to a result [?, Lemma 3.2] relating this form of stability with the ESP (which is here equivalent to the USP).

Thirdly, learning a chaotic map on a set(here Y_T) with even a small measure of noise means that the learnt system accepts arguments out of the set (Problems similar to Takens, cf. ??). We thus have a map G_T^+ acting on Y_T^+ and since Y_T is not guaranteed to be an attractor of G_T this could lead to errors. If one learns G_T directly, we are in danger of replicating the problems arising for Takens. To see this, consider Fig. ??.

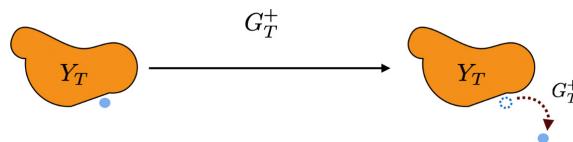


Fig. 4. Diagram of potential errors developing when G_T is learnt.

To circumvent this: if a driven system with a saturation function¹⁶ is used, we can indirectly (via g) ensure the existence of an attractor A of the map which implements G_T , and containing Y_T^+ .¹⁷ By defining the dynamics to A , this prevents large errors from occurring. More details on the global dissipativity of g , as it is termed there, may be found in [?].

EdN:16
EdN:17

In effect, global dissipativity prevents large numerical errors due to the noise in input data from

¹⁵EDNOTE: B: Is this satisfactory?

¹⁶EDNOTE: B: What is a saturation function?

¹⁷EDNOTE: B: I'm not sure this sentence makes complete sense.

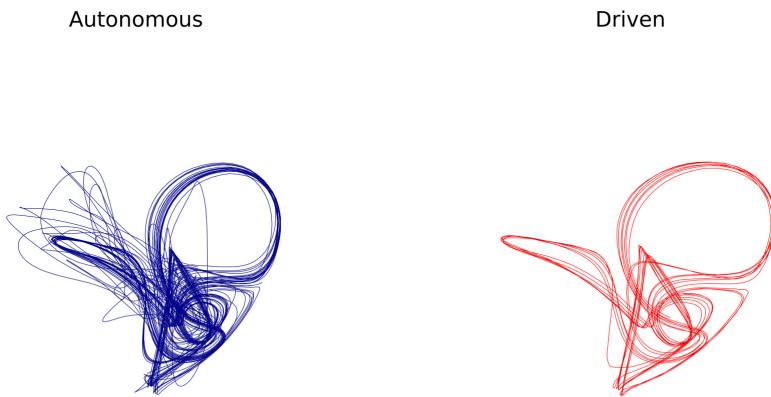


Fig. 5. Illustrating empirically the importance of global dissipativity. Plotted below are the three principal components of randomly initiated points(blue) of a learnt version of G_T versus the trajectory (x_{n-1}, x_n) (red) of driven states obtained from the Fractal Dream Attractor's data ??).

arising (see Fig. ??). If a system is not globally dissipative, negligibly small errors could lead to major inaccuracy which thus induce predictions to utterly fail.

Chapter 5

Implementation

In this section we turn to the implementation of the function G_T by discussing a method to learn the function Γ . We then indicate where this implementation differs from previous work done by ([?]) and empirically show that the SDD of G_T are less functionally complex than $\Phi_{k,\theta}$ or T by considering the Pearson correlation coefficient. Finally, we present and discuss some numerical results for a focus system of this project: the double pendulum. We also consider other chaotic attractors.

5.1 Implementing G_T

The map G_T is implemented by applying the functions Γ , g and the projection function π_2 in the relevant order as described by the equations ???. To reach this point, one must first learn Γ . After first normalising the data to ensure it lies within the interval $|x| < 1$ and has zero mean value, we continue by driving the system with external input through the function g .

From (??) we know that $g(u, x) = (1 - a)x + a \cdot \tanh(Au + \alpha aBx)$. Recall that g is SI-invertible. Adopting the terminology of [?], A is called the “input matrix” and B the “reservoir matrix”.

We make use of a Recurrent Neural Network (RNN) in our implementation. RNN’s are widely used in the literature for discrete-time processing and are considered to be discrete-time version of continuous dynamical systems modelling a network of neurons. (**Citation.**) A complete discussion of RNN’s is outside the scope of this thesis, so we present only the details necessary to the current discussion. For information, (**Cite**, **Cite**, **Cite**) may be consulted. We implement our RNN in the *Python* programming language and making use of the *Scipy* library, alongside the *Keras* package in the *Tensorflow* library.

If the RNN we will be using in the next step has N neurons, then A and B are each $N \times N$ matrices. Inputs u_n of dimension $K < N$ are padded by zeroes to obtain an input of dimension N . As we do not need the entire left-infinite history of input, an arbitrary initialisation of the system with initial values followed by the driving of the system with the function g means that the generated values will approach the actual solution's elements in a uniform manner thanks to the UAP. (**Citation**)

Thirdly we prepare the network. Utilising a network N neurons (the same value N as in the dimensions of the matrices A, B). Training is done using the Adam Optimiser where the MSE loss function is optimised. The system is trained several times(in our case 4) with incrementally smaller learning rates on each run (0.001 and thereafter divided by 10 at each run). Additional parameters such as the number of network-layers, neurons per layer and training-length are presented case-by-case below.

We then reach the prediction step. Making use of g 's SI-invertibility and g as defined in (??), we have the following expression for u_n replicated from (??).

$$u_n := A^{-1} \left(\frac{1}{\tanh^{-1}} \frac{1}{a} (x_{n+1} - (1-a)x_n) \right) - \alpha B x_n$$

The two model equations for learning from data ((??)) allows us to determine what the state x_{n+1} at time $n + 1$ will be. We specifically remind ourselves of

$$x_{k+1} = \pi_2 \circ (\Gamma, \pi_2) \circ (\pi_2, g)(u_k, x_k)$$

5.2 Changes in this project to previous work of [?]

EdN:18 To understand the first and most drastic¹⁸ change, we recall the the conjugacy in ??:

$$\widehat{U}_T \quad \widehat{U}_T$$

$$Y_T \quad Y_T$$

We recall that G_T is topologically conjugate to \widehat{T} , a result which we have already made use of.

The authors of [?] did not, however, consider the relationship between the inverse-limit space \widehat{U}_T

¹⁸EDNOTE: change word

and the input $u_n = \theta(w_n)$ (where θ is our observable and w_n the complete input), a concept we now explore.

Let (V, T) be a dynamical system with dynamics generated by the update equation $v_{n+1} = Tv_n$. Consider the case where the only available information is observations θ from an orbit of the map T where $\theta : V \rightarrow \mathbb{R}$ is an observable. Our task is still to find some system topologically conjugate to (V, T) so that we may use the datapoints $\theta(v_0), \theta(v_1), \dots, \theta(v_m)$ to predict the subsequent observations $\theta(v_{m+1}), \theta(v_{m+2}), \dots$.

Applying the Takens embedding theorem, we may conclude that when an input from the delay-coordinate mapping $\Phi_{\theta,2d}(\theta(v_n)) := (\theta(v_{n-2d}), \dots, \theta(v_{n-1}), \theta(v_n))$ is fed to the driven system g as the input u_n , there exists a homeomorphism $F_\theta : \Phi_{\theta,2d}(\theta(v_n)) \mapsto \Phi_{\theta,2d}(\theta(v_{n+1}))$.

This may be stated alternatively as follows: If the input u_n is of the form $u_n := \Phi_{\theta,2d}(\theta(v_n))$ and is fed to a driven system g possessing both the USP and SI-invertibility, it follows from the Causal Embedding Theorem ?? that the newly generated space (Y_F, G_F) topologically conjugate to the inverse-limit space of $(\Phi_{\theta,2d}(V), F_\theta)$. Moreover, we can forecast the values u_n, u_{n+1}, \dots and $\theta(v_n), \theta(v_{n+1}), \dots$ as well as having the guarantee that this embedding is stable when considered in terms of global dissipativity as discussed before.

$$\begin{array}{ccc} \widehat{\Phi}_{2d,\theta} & & \widehat{\Phi}_{2d,\theta} \\ Y_F & & Y_F \end{array} \quad (5.1)$$

The existence of \widehat{F} is guaranteed, but the map $v_n \mapsto v_{n+1}$ itself is unknown. We may, however, make use of the SI-invertibility of the function $g(u_{n-1}, x_{n-1}) = x_n$ to conclude that (x_{n-1}, x_n) determines u_{n-1} uniquely.

Accordingly (x_{n-2}, x_{n-1}, x_n) determines u_{n-2} and u_{n-1} uniquely and generally $(x_{n-d}, \dots, x_{n-1}, x_n)$ determines u_{n-d}, \dots, u_{n-2} and u_{n-1} uniquely.

We therefore need not know what \widehat{F} is, as we can determine $v_{n+1} = [u_n, u_{n-1}, \dots, u_{n-d}]^T$ indirectly by determining the value of $(x_{n-d}, \dots, x_{n-1}, x_n)$, a much more feasible task.

Accordingly we define the following abbreviations:

- Delay Coordinates (DC): A vector composed by appending a fixed amount of previous values to the final value.
- Single-Delay Dynamics(SDD): Dynamics generated by 2-element delay-coordinate vectors of

the form (x_{n-1}, x_n) where only one previous value is used.

- Multi-Delay Dynamics(MDD): Dynamics generated by arbitrary-length delay-coordinate vectors of the form $(x_{n-d}, \dots, x_{n-1}, x_n)$. (d must be sufficiently large, but is unknown. We vary this parameter in experimentation.)

Within this framework, the PCA-implementation of [?] is then considered to fall underneath the of auspices of what we term *Approximated Multi-Delay Dynamics(AMDD)*.

Naturally, one would immediately suppose that the MDD approach would increased computation time and resource use as the internal matrices and datasets generated during learning would grow with every increase in the delay-parameter d .

In fact, preliminary results indicate the opposite due the observation that the implementation of MDD is inversely proportional to the the dimension of the network (i.e. the number of neurons in the RNN) and possibly also the number of epochs/batch size necessary to produce reliable predictions.

EdN:19 This, in turn, actually speeds up the computation-time and provides results of similar accuracy. ¹⁹

Secondly

Sparse Matrices:

- Well-conditioned: condition number close to 1
- Scaled
- Sparsity/density ratio

Third

Density of matrices A/B(i.e. number of non-zero entries in each matrix) (separate from sparse matrix implementation)

5.3 Numerical simulations

In this section we showcase numerical results pertaining to 3 attractors that were simulated using the methodology described in the previous chapters. We first consider a simple physical system.

¹⁹EDNOTE: I did not have enough time to explore this properly. I can only hazard guesses and promise future research work in this project. How should I best convey this?

5.3.1 Double Pendulum

The Double Pendulum consists of two pendulums fastened to one another such that the system moves as a whole. The first (or higher pendulum) is attached to a fixed point while the second is attached to the endpoint of the first. (See below) This system is a classical example of a system exhibiting chaos. (**Citation.**) Consider again [??](#). The system has four variables: angle and angular velocity of each pendulum rod. An ideal system is assumed where the rods are massless and no friction in the pivots are considered.

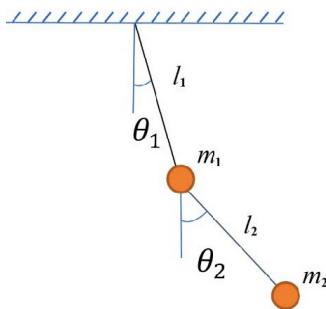


Fig. 1. Pictorial representation of the setup of a DP. Image taken from [\[?\]](#)

The scalar equations of motion are derived using Newtonian physics and can be found in [\[?\]](#). The second-order system is converted to a first-order ODE and solved using the Runge-Kutta numerical method. Noise-free, undamped DP data-sets were generated and utilised in this project due to the requirement that the map T be surjective. We discuss the case of a DP exhibiting energy-loss (i.e. where the map T is not surjective) below.

Data originating from a non-surjective map.

The importance of the requirement that data originates from a surjective map is observed learning failure of a map describing a damped DP. Due to the assumption, the program discounted any possibility of energy-loss and we consider the example below.

Despite feeding data that originated from a DP with dying oscillations into the system, a map was instead learnt which continued to oscillate in a chaotic, but conservative manner.

Having spent ample time to discuss the challenges, we now change tone so as to satisfy the optimist. We present successful run for DP-data to repredict the future movement of the pendulum.

The system was driven for 15500 steps, of which the first 500 were immediately discarded to simulate the network 'forgetting'. The driving function g had input and reservoir matrices were implemented as sparse matrices filled to 20%. Parameter-values of $\alpha = 0.99$ and $a = 0.5$ were used. The remaining 15000 datapoints were used to train a network with 128 neurons. The network was

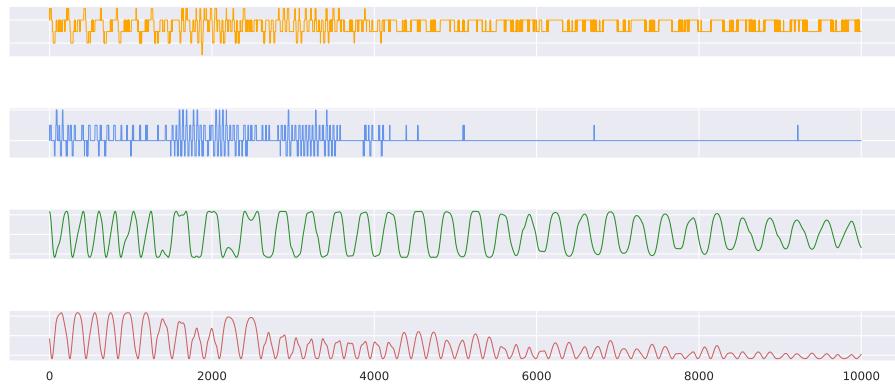


Fig. 2. The map generating data for a damped pendulum is not surjective. Data obtained from chaotic double pendulum dataset generated by [?]

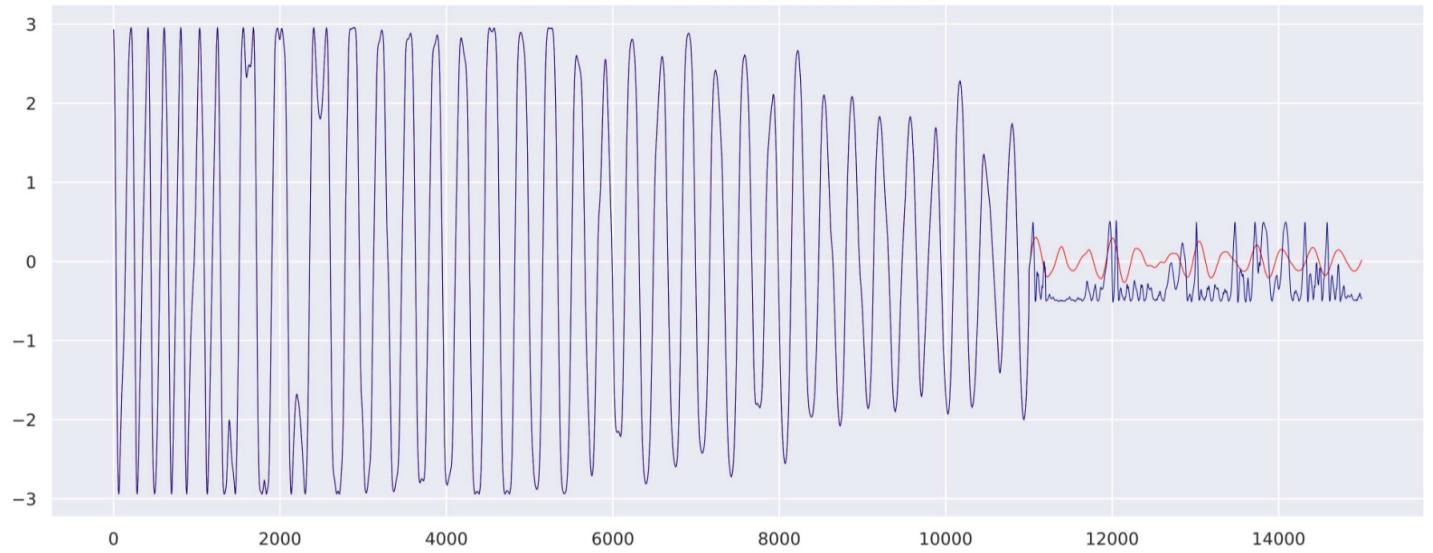


Fig. 3. Failure to learn a map which is not surjective.

initialised to contain 16 hidden layers each with a layer dimension of 64, activated with the tanh activation function. Training was accomplished using 256 epochs with batch sizes 128 and a delay of 1.

5.3.2 Additional Attractors: Clifford & Thomas

We follow the methodology presented in [?] but opt to consider attractors different from the Lorenz system and Hénon map in this project.

Thomas' Cyclically Symmetric Strange Attractor

Thomas' Cyclically Symmetric Strange Attractor, a 3D attractor proposed in 1999 by René Thomas in [?], is described by a set of three equations which is distinctly reminiscent of the Lorenz system, another deterministic three-equation model exhibiting chaotic behaviour. It has a single parameter β and has been shown to transition to chaotic behaviour when $\beta < 0.208186$ [?].

$$x_{n+1} = \sin(y_n) - \beta x_n \quad (5.2)$$

$$y_{n+1} = \sin(z_n) - \beta y_n \quad (5.3)$$

$$z_{n+1} = \sin(x_n) - \beta z_n \quad (5.4)$$

The behaviour of the system with a β -parameter value of 0.1056 was simulated by scaling it to fit inside the interval $[-1, 1]$. Two cases were considered: noise-free and perturbed. Noise was generated from a normal distribution with mean 0 and standard deviation 0.05, which translates approximately to *dB. A sequence of 3000 observations in both the clear and noisy data-sets respectively were fed into the system with 300 entries discarded to simulate the network's loss of memory. The next 9000 steps were predicted and compared with true data.

The RNN was initialised as follows: the dimension of the network is set equal to 500 neurons and 12 hidden layers of dimension 64 each were initialised. Training is realised through 150 epochs of batch sizes of 128 each; the map is learnt via the Adam Optimiser using finer and finer learning rates (0.001 and then divided by 10 at every iteration).

Fractal Dream Attractor

Another 2-equation discrete-time map named the Fractal Dream Attractor, or more commonly known as the Pickover Map (first discovered by Clifford A. Pickover and discussed in his fascinating book "Chaos in Wonderland" [?]), was also considered.

$$x_{n+1} = \sin(ay_n) + c \cos(ax_n) \quad (5.5)$$

$$y_{n+1} = \sin(bx_n) + d \cos(by_n) \quad (5.6)$$

²⁰EDNOTE: B: Expand more.

Input	Dimension of X	u_n vs u_{n+1}	$\begin{bmatrix} x_{n-1} \\ x_n \end{bmatrix}$ vs $\begin{bmatrix} x_n \\ x_{n+1} \end{bmatrix}$
(w_n) comes from the Thomas Attractor (for $\beta = 0.1056$): $w_{n+1} = \begin{bmatrix} \sin(w_{n,y}) - \beta w_{n,x} \\ u_y = \sin(w_{n,z}) - \beta w_{n,y} \\ u_z = \sin(w_{n,x}) - \beta w_{n,z} \end{bmatrix}$ $u_n = w_n - \bar{w}$	10 100 1000	0.99989 0.999913 0.999901	0.999886 0.999913 0.999901
(w_n) comes from Noise-Added Thomas Attractor (for $\beta = 0.1056$): $w_{n+1} = \begin{bmatrix} \sin(w_{n,y}) - \beta w_{n,x} \\ u_y = \sin(w_{n,z}) - \beta w_{n,y} \\ u_z = \sin(w_{n,x}) - \beta w_{n,z} \end{bmatrix}$ $u_n = w_n - \bar{w}$	10 100 1000	0.9994 0.99991 0.999895	0.99985 0.99991 0.999895
(w_n) comes from the Thomas Attractor (for $\beta = 0.208186$): $w_{n+1} = \begin{bmatrix} \sin(w_{n,y}) - \beta w_{n,x} \\ u_y = \sin(w_{n,z}) - \beta w_{n,y} \\ u_z = \sin(w_{n,x}) - \beta w_{n,z} \end{bmatrix}$ $u_n = w_n - \bar{w}$	10 100 1000	0.99982 0.999851 0.99984	0.99981 0.999851 0.99984
(w_n) comes from the Clifford Attractor (for $a = -1.7; b = 1.8; c = -1.9; d = -0.4$): $w_{n+1} = \begin{bmatrix} \sin(aw_{n,y}) + c \cos(aw_{n,x}) \\ \sin(bw_{n,x}) + d \cos(bw_{n,y}) \end{bmatrix}$ $u_n = w_n - \bar{w}$	10 100 1000	-0.21713 0.8334 0.84854	0.78719 0.8334 0.84854
(w_n) comes from the DP (for): $u_n = w_n - \bar{w}$	10 100 1000	0.99141 0.99238 0.99278	0.98916 0.99238 0.99278

Table 5.1. Following [?], we determine the multidimensional correlation coefficients ρ to evidence a general reduction in the functional complexity of the map G_T emerging due to a RNN. Rows correspond to distinct dynamical systems considered in the experiments. The second column corresponds to the amount of artificial neurons (the dimension of X) in the implemented RNN. The last two columns comprise numerical estimates of ρ for the relevant vectors. The linear association between u_n vs $T(u_n)$, as well as the linear relationship $\begin{bmatrix} x_{n-1} \\ x_n \end{bmatrix}$ vs $G_T\left(\begin{bmatrix} x_{n-1} \\ x_n \end{bmatrix}\right)$ may be contrasted by the reader. Finally, the RNN's dimension (from ??)) is provided to exhibiting that an increase in dimension of the driven system will typically result in a map G_T with less functional complexity.

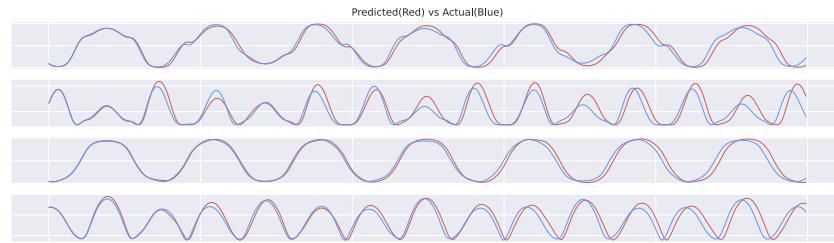
5.4 Generalized Pearson Correlation

EdN:21 ²¹

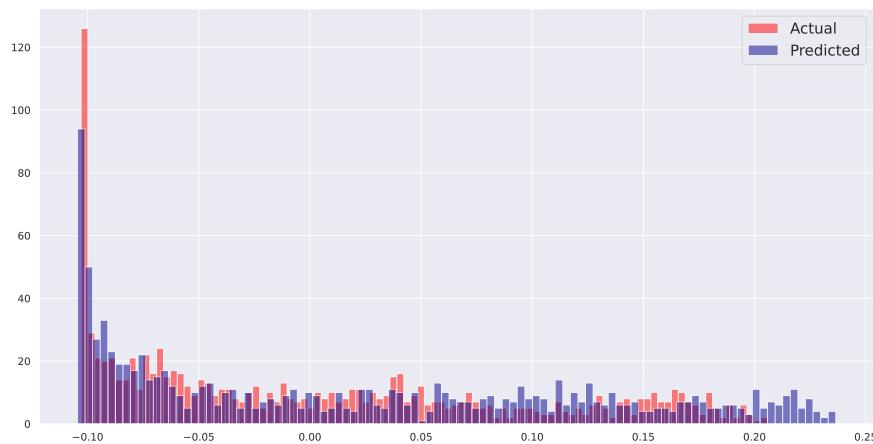
EdN:22 ²²

²¹EDNOTE: B: Alter heading.

²²EDNOTE: B: Write Pearson Introduction. Could you give me some help here, Sir?

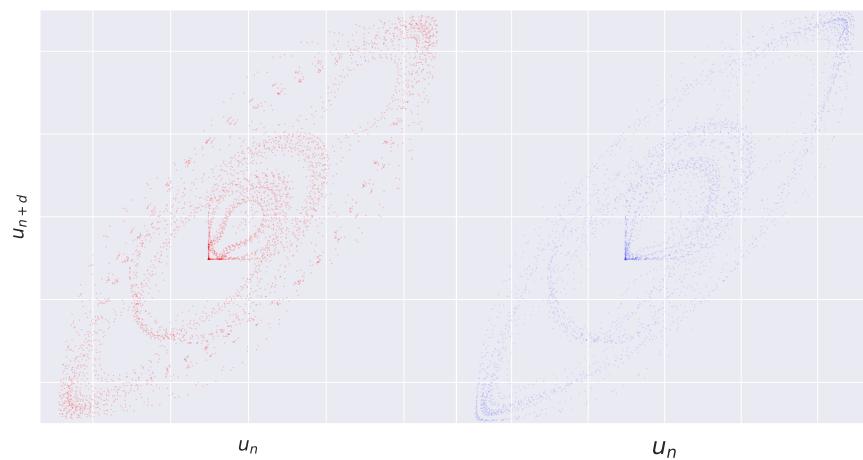


Predicted(red) and actual(blue) trajectories of the x - and y -coordinates of each of the pendulum heads.

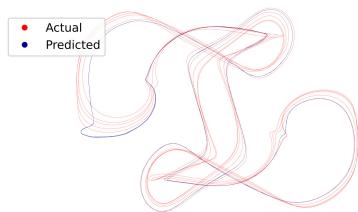
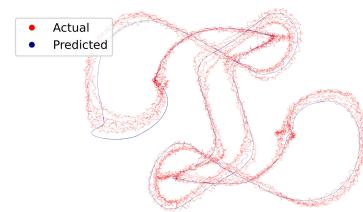


Densities of predicted and actual values of x-coordinate of first pendulum head.

$d=5$

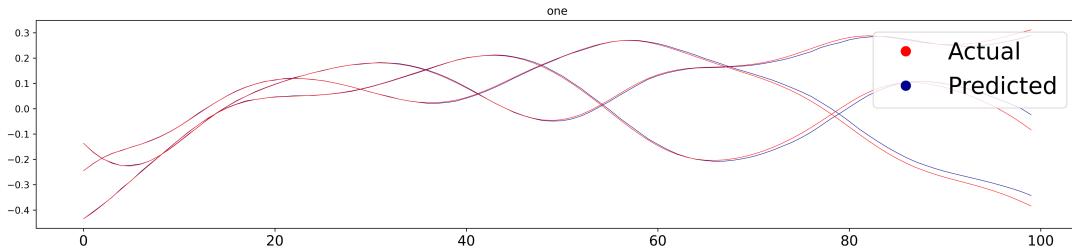


Takens Delay Map with delay=5 to illustrate behaviour of predicted and actual values of double pendulum data.

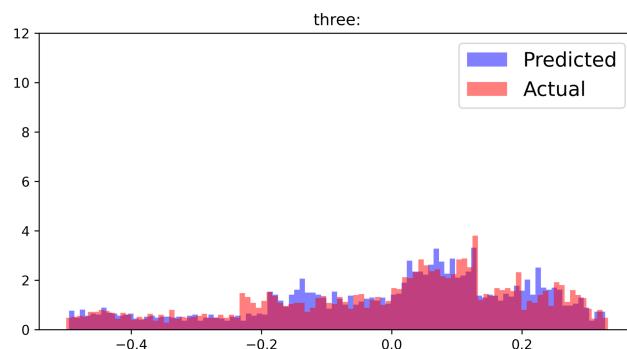
Thomas' attractor for $\beta=0.1056$ Thomas' attractor for $\beta=0.1056$ 

True and predicted trajectories in the absence of noise.

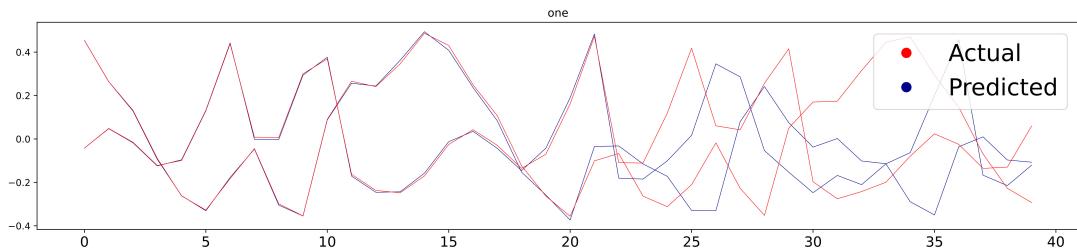
True and predicted trajectories for noise added.



Predicted trajectories for the attractor demonstrate empirically the ability to predict the evolution of the trajectory for the next 100 timesteps



Represented here the learnt (blue) and actual (red) densities of the first coordinate of the dataset.



These graphs were constructed by predicting the Clifford system 15000 steps into the future and in so doing illustrating the long-term consistency and accuracy of the learnt system. As perceived here, we are able to lock on to the trajectory of the Clifford map almost exactly for the first 25 steps.

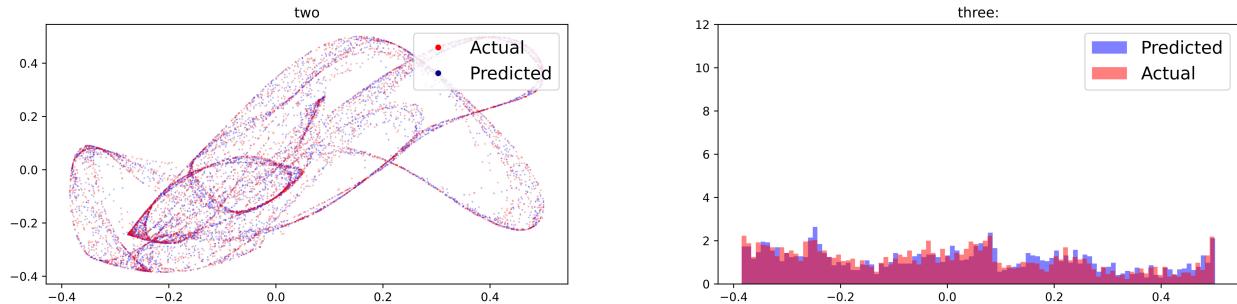


Fig. 4. The bottom row, left, shows a plot of the true attractor(red) and the predicted values(blue). Bottom row, right, shows the distribution of the first coordinate of the learnt system (blue) and the actual Clifford system (red).

Chapter 6

Conclusion

As technology has advanced over the past century, humanity’s ability to record and manipulate rich, complex data in order to produce forecasts has profoundly increased. Collecting measurements from underlying artificial, natural, and physical systems through observables to realise a dynamical model with greater descriptive power is well-established in present-day science.

Core to the success of these techniques is the fact that dynamics discerned from the observed space can often be characterised by a mapping of much lower functional complexity when compared to the map describing the unknown underlying system. Takens theorem and the numerous generalizations thereof is one such method establishing (under some generic conditions) the learnability of a system produced by creating a delay-coordinate vector from a sufficiently large number of previous observations of a dynamical system. This system is topologically conjugate to the system from which the observed time-series was obtained.

A critical concern of this method is the requirement of full knowledge of the variables of the underlying state space that govern the complex dynamical system one is interested in. In real-world conditions this intelligence is seldom available and one must depend heavily on human expertise and insight.

Each discrete-time state space model has at its core a driven dynamical system. In this project, it has been shown that an input originating from a compact space entailing exactly one solution (i.e., if the driven system has the USP) is equivalent to representing information without distortion, a concept which was formalised by establishing a causal mapping’s existence. The established results are general, implying that any driven dynamical system with specific properties can give rise to a causal mapping, and in so doing providing an answer as to the success of arbitrary choices of driven systems in information processing applications in [4, 5].

In particular, we have a topological conjugacy (semi-conjugacy) between the data comprising

the single-delay lag dynamics of a driven system's and the underlying homeomorphic dynamical system. Finite-length input data is sufficient for forecasting as driven system states are influenced only in an almost negligibly small manner by the left-infinite segment of the past of the input beyond some arbitrarily large but finite time (a result which holds due to the universal semi-conjugacy of a driven system's continuity)

Viewed through the lens reservoir computing approaches, we have also demonstrated that when single-delay dynamics are utilised, a learnable map in a reservoir computing network certainly exists. Moreover, such a driven system within a reservoir computing network enjoys certain proven qualities pertaining to the robustness w.r.t input- and parameter-perturbation. Accordingly, we are able to preserve the quality of a representation that would otherwise remain sensitive to various fluctuations, rendering the learning of the system's dynamics wholly unpredictable. In addition, we obtain the robustness of the system against external noise, a phenomenon which is observed for the Fractal Dream Attractor.

Finally, we consider some physical systems to observe the methodology in practice and verify that we can indeed predict the future evolution of chaotic systems such as the double pendulum. Empirical evidence shows that a linearity measure like a Pearson correlation coefficient points at reduced functional complexity while learning the dynamics in the state space of the driven system

Considered with a philosophical-scientific lens, this project has aimed to establish a juncture between rationalism (the theory of nonautonomous dynamical systems) and empiricism (machine learning) to yield results profitable to the discipline of Applied Mathematics. This was done by showing that dynamics in an observed space can be established as topologically conjugate to the underlying system. As a result of the topological conjugacy, deep learning techniques are then subsequently applied to physically such equations from the data. In so doing, we acquire models of high fidelity that provide exceptional forecasting results on chaotic data that exhibit both statistical and topological consistency in the long run.