

# DATA-DRIVEN IDENTIFICATION OF ATTRACTORS USING MACHINE LEARNING

MARCIO GAMEIRO, BRITTANY GELB, WILLIAM D. KALIES, MIROSLAV KRAMÁR,  
KONSTANTIN MISCHAIKOW, AND PAUL TATASCIORE

**ABSTRACT.** In this paper we explore challenges in developing a topological framework in which machine learning can be used to robustly characterize global dynamics. Specifically, we focus on learning a useful discretization of the phase space of a flow on compact, hyper-rectangle in  $\mathbb{R}^n$  from a neural network trained on labeled orbit data. A characterization of the structure of the global dynamics is obtained from approximations of attracting neighborhoods provided by the phase space discretization. The perspective that motivates this work is based on Conley's topological approach to dynamics, which provides a means to evaluate the efficacy and efficiency of our approach.

## 1. INTRODUCTION

The analysis of applied dynamical systems is undergoing a paradigm shift toward data-driven methods. This change has been triggered by an explosion of data available for large, multiscale problems, as well as the development of powerful, computational methods for data-based analysis facilitated by machine learning. Nevertheless, the fundamental questions remain the same: how do we achieve an insightful characterization of the dynamics, and how can we predict the dynamics?

Current methods often fit the data to an explicit model that takes the form of a parameterized differential equation or difference equation [6, 23]. The clear advantage of this approach is that it allows the user to exploit a wide variety of well-developed techniques to analyze the associated dynamics. Alternatively, black-box surrogate models can be learned directly from the data. This approach typically provides for less interpretability of the model, but may lead to efficient predictability of the dynamics [41, 43]. However, bifurcation theory suggests that descriptions of long-term dynamics provided by either of these approaches may not be robust.

The perspective that motivates this work is based on Conley's topological approach to dynamics [9], which provides a framework for deducing global information about dynamics that is robust with respect to perturbations. This framework has been successfully applied to the analysis of time series data [3, 35], machine-learned dynamics [48], and rigorous verification of numerical analysis of differential equations [4, 18, 36] and difference equations [1, 11]. Two steps are common to most of these applications: (**Step 1**) identification of regions in which the dynamics of interest occurs, and (**Step 2**) use of the Conley index, an algebraic topological invariant, to identify the existence and structure of the dynamics in the regions of interest. Theoretical results [17] indicate that with sufficient data and sufficient computational resources, both **Step 1** and **Step 2** can be accessed via machine learning. However, [17] provides no quantification as to how much data or how much computation is required.

In this paper we undertake an initial exploration of the challenges of using this topological framework to machine learn dynamics and focus on **Step 1**, identifying regions of interest. As is discussed in Section 2, all regions of interest can be derived from a lattice of attracting neighborhoods. With this in mind, we work with systems that have multiple global basins of attraction and explore the use of machine learning to identify discretizations of phase space that provide approximations of attracting neighborhoods from which the correct Conley index information can be computed.

Using various examples, with dynamics coming from known ordinary differential equations (ODEs), we show that if test loss is sufficiently low, then the Conley indices of the attracting

neighborhoods match the Conley indices of the known attractors. This is significant because, as is mentioned above, the information about the structure of dynamics is deduced using the Conley index. In this sense we view this work as a proof of concept that machine learning and computational Conley theory can be used together for applications in nonlinear dynamics.<sup>1</sup>

However, there are important caveats to this claim. To compute the Conley index we need to be able to compute homology. Computational homology is a relatively new endeavor [24], and while considerable progress is being made [15, 21], to the best of our knowledge the most manageable package for dealing with complexes of moderate dimension is `pyCHomP` [16]. Since `pyCHomP` is most efficient on cubical complexes, we restrict our study to special neural networks to achieve a decomposition of the space that fits into the data structure of a cubical grid. We use the strategy mentioned in [2] to ensure that the network partitions the space into parallelotopes that can be transformed to cubes by a linear coordinate change. There is a price to be paid for using these networks, namely they cannot efficiently approximate attracting neighborhoods with complicated geometry. However, we demonstrate on several examples that as long as the network is a sufficiently good approximation to the labeling function introduced in Section 3.1, then we obtain the correct Conley indices.

We also present a few examples where the geometry is too complicated to be captured properly by the restricted network. For these examples we present heuristics that detect if the network is versatile enough to capture the geometry of the basins of attraction. Our heuristics apply to unrestrained networks as well, which is critical for further development of the proposed framework. Finally, we show that the number of grid elements (parallelotopes) in the machine learned decomposition is radically smaller than in a regular grid. This observation is crucial because despite the fast improvements in computational homology, a large number of cubes is still a serious bottleneck.

The rest of the paper is organized as follows. In Section 2, we recall the basic concepts of the Conley-Morse theory for ordinary differential equations. Development of our method in Section 3 is divided into two parts. In Section 3.1, we present possible strategies for identifying attractors and creating a sample of a labeling function that describes to which basin of attraction a given point belongs. In Section 3.2, we use a neural network to partition the domain of the ODE into polytopes. We also explain there how to use these polytopes to approximate the attracting neighborhoods of the attractors identified in Section 3.1. These attracting neighborhoods are crucial for computing the Conley indices. The details about training the neural networks and benchmarking our method are presented in Section 3.3. The results are divided into three parts. Section 4.1 familiarizes the reader with the pipeline of our method on simple examples, and Section 4.2 highlights the reduction in the size of the grid produced by our method. Two systems that exceed the versatility of the restrained network are discussed in Section 4.3, where we also provide heuristics on deciding if the network is sufficiently expressive to produce the correct Conley indices. Finally in Section 5, we outline how our framework can be further developed to overcome the limitations of this initial study.

## 2. DYNAMICS AND CONLEY INDEX THEORY

We are interested in characterizing the dynamics of an ODE

$$(1) \quad \frac{dx}{dt} = g(x)$$

on a compact region  $X \subset \mathbb{R}^d$ . For the purposes of this paper we choose  $X$  to be a hyperrectangle of the form  $\prod_{i=1}^d [a_i, b_i]$ . For simplicity of exposition we assume that the solutions to the ODE give rise to a *flow*  $\varphi: \mathbb{R} \times \mathbb{R}^d \rightarrow \mathbb{R}^d$ ; that is, a continuous function satisfying  $\varphi(0, x) = x$  and  $\varphi(t, \varphi(s, x)) = \varphi(t + s, x)$  for all  $t, s \in \mathbb{R}$  and  $x \in \mathbb{R}^d$ . Our description of the dynamics of (1) involves identifying *invariant sets*; that is, sets  $S \subset X$  such that  $\varphi(t, S) = S$  for all  $t \in \mathbb{R}$ .

---

<sup>1</sup>Code is available at: <https://github.com/begelb/MLCD>

As indicated in the introduction, **Step 1** involves the identification of regions of phase space in which dynamics of interest occurs. *Ab initio* these regions are not known and thus must be identified via the dynamics. Perhaps the most obvious region to attempt to identify is an *attracting neighborhood*; that is a compact set  $N \subset X$  with the property that there exists  $\tau > 0$  such that  $\varphi(t, N) \subset \text{int}(N)$  for all  $t \geq \tau$ , where  $\text{int}$  denotes interior. Observe that the condition that  $N$  is compact and  $\varphi(\tau, N) \subset \text{int}(N)$  guarantees that an attracting neighborhood is *robust*; that is, if  $N$  is an attracting neighborhood for  $\frac{dx}{dt} = g(x)$ , then  $N$  remains an attracting neighborhood even if the nonlinearity  $g$  is slightly perturbed.

Attracting neighborhoods give rise to invariant sets. In particular, if  $N$  is an attracting neighborhood, then

$$A = \omega(N) := \bigcap_{t>0} \text{cl}(\varphi([t, \infty), N))$$

is an invariant set. An invariant set  $A$  of this form is called an *attractor*, and  $N$  is referred to as an attracting neighborhood of  $A$ . The *basin of attraction* of an attractor consists of the union of all of its attracting neighborhoods. This last remark emphasizes an important point; an attracting neighborhood  $N$  identifies a unique attractor  $A = \omega(N)$ , but an attractor typically has uncountably many attracting neighborhoods. Thus, at least abstractly, it is easier to characterize dynamics in terms of attractors than in terms of attracting neighborhoods.

Observe that identification of an attracting neighborhood  $N$  provides predictive power; an initial condition in  $N$  asymptotically approaches the attractor  $A = \omega(N)$ . In particular, we leave it to the reader to check that if  $A$  is an attractor and  $x \in X$ , then  $\omega(x) \subset A$  if and only if  $x$  is an element of the basin of attraction of  $A$ . The dynamics on an attractor can be quite complicated, for example chaotic dynamics on strange attractors. More importantly, bifurcation theory tells us that the dynamics can change dramatically even with respect to small perturbations of the system. This implies that direct numerical simulation cannot in general provide a precise characterization of the dynamics on an attractor. However, as indicated above, attracting neighborhoods are robust with respect to perturbations, and thus together with Conley index information can provide coarse but accurate information about dynamics within the attracting neighborhood.

As indicated in the introduction, we focus on multistable systems, and thus we need a language in which to organize the structure of attractors. As is discussed in [10], the collection of attractors of a dynamical system has the structure of a bounded, distributive lattice with partial order given by inclusion. The attractor lattice of a system need not be finite, but for systems in  $\mathbb{R}^d$ , it is countable [42]. Furthermore, finite sublattices of attractors can be used to approximate the dynamics at finer and finer scales [25, 26]. In this setting, a system exhibits *multistability* if there is a finite sublattice  $\mathbf{A}$  of attractors that contains more than one *minimal* attractor, i.e. an attractor with no nonempty predecessor in  $\mathbf{A}$ .

By [10, Proposition 4.3], the collection of attracting neighborhoods forms a bounded distributive lattice with partial order given by inclusion, and  $\omega$  maps lattices of attracting neighborhoods to lattices of attractors homomorphically. From this structure, one can deduce the following result that constructs  $\mathbf{A}$  from multiple, distinct attracting neighborhoods.

**Theorem 2.1.** *Suppose that  $X$  is an attracting neighborhood for (1). Let  $\{N_k \subset X \mid k = 0, \dots, K\}$  be a set of mutually disjoint attracting neighborhoods with  $A_k := \omega(N_k)$ . Then*

$$\mathbf{A} := \omega(X) \cup \left\{ \bigcup_{k \in I} A_k \mid I \subset \{0, \dots, K\} \right\}$$

*is a finite lattice of attractors with minimal attractors  $A_k$ .*

The importance of Theorem 2.1 for this work is that from data we machine learn a collection of disjoint attracting neighborhoods, and therefore identify a lattice of attractors. By definition of the lattice, the number of elements in  $\mathbf{A}$  grows rapidly with the number of disjoint attracting neighborhoods  $K$ . Thus, rather than working with  $\mathbf{A}$  we turn to an equivalent representation that takes the form of a partially ordered set (poset).

**Definition 2.2.** A *Morse representation* of a dynamical system on a compact metric space  $X$  is a finite, partially ordered set  $\{M_i\}_{i=1,\dots,n}$  of nonempty, pairwise-disjoint, compact, isolated invariant sets such that for all  $x \in \omega(X) \setminus \bigcup_{i=1}^n M_i$  there exist  $M_i < M_j$  such that  $\omega(x) \subset M_i$  and  $\alpha(x) \subset M_j$ , where  $\alpha(x) := \bigcap_{t>0} \text{cl}(\varphi(x, (-\infty, -t]))$ . The elements of a Morse representation are called *Morse sets*.

Even a cursory explanation of the general correspondence between lattices of attractors and Morse representations is beyond the scope of this article (the interested reader is referred to [26]). Instead, we consider a specific example that we hope provides sufficient intuition.

*Example 2.3.* Consider a hyperrectangle  $X \subset \mathbb{R}^d$  and assume it is an attracting neighborhood for a flow  $\varphi: \mathbb{R} \times \mathbb{R}^d \rightarrow \mathbb{R}^d$ . Let  $\{N_k \subset X \mid k = 0, 1, 2\}$  be mutually disjoint attracting neighborhoods. As discussed in Section 3, we use machine learning to identify these sets. Figure 1(a) shows the associated lattice of attracting neighborhoods. Let  $A_k := \omega(N_k)$ . A topological argument ( $X$  is connected and hence  $\omega(X)$  is connected) implies that  $\omega(X) \neq A_0 \cup A_1 \cup A_2$ . By Theorem 2.1, we obtain the lattice of attractors  $\mathbf{A}$  shown in Figure 1(b).

The associated Morse representation is shown in Figure 1(c). The minimal Morse sets  $\{M_k \mid k = 0, 1, 2\}$  are the attractors, i.e.,  $M_k = A_k$ . The remaining Morse set is defined as follows

$$M := \{x \in \omega(X) \mid \omega(x) \cap M_k = \emptyset \text{ for all } k = 0, 1, 2\}.$$

A point set topological argument based on the continuity of the flow  $\varphi$  shows that  $M \neq \emptyset$ ,  $M$  is compact, and  $M$  is disjoint from  $M_k$  for  $k = 0, 1, 2$ . To justify the partial order relation, we note that if  $x \in N_k$ , then  $\omega(x) \subset A_k$ . Therefore, if  $x \in \omega(X)$ , then it is impossible for  $\alpha(x) \subset M_k$  and  $\omega(x) \subset M$  for any  $k = 0, 1, 2$ .

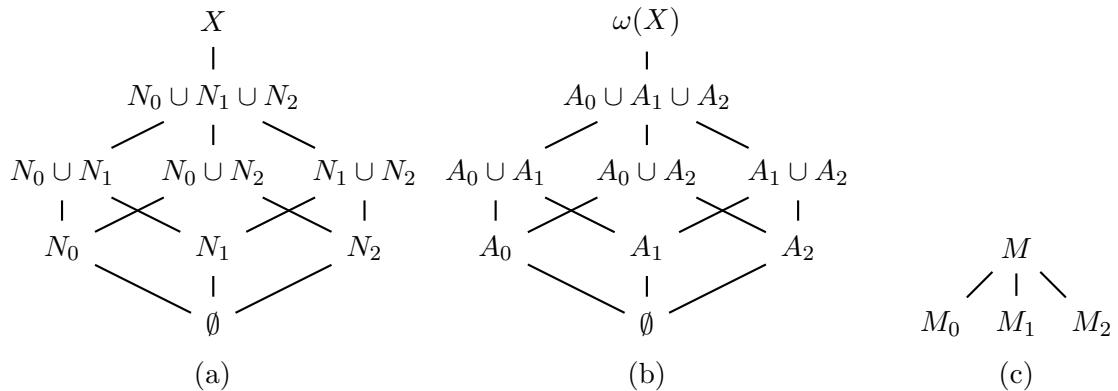


FIGURE 1. (a) Lattice of attracting neighborhoods derived from three mutually disjoint attracting neighborhoods. (b) Associated lattice of attractors. (c) Associated Morse representation with partial order relations  $M_k < M$  and  $M_k$  not related to  $M_i$  for  $i, k \in \{0, 1, 2\}$ .

Observe that in Example 2.3, we explicitly used topology to argue that  $\omega(X) \neq A_0 \cup A_1 \cup A_2$  or equivalently that  $M \neq \emptyset$ . More sophisticated topological arguments allow us to extract additional information concerning the structure of the dynamics. In particular, the *Conley index*, a homological invariant, can be used to prove the existence of invariant sets and provide information about their structure. The most fundamental result is that if a Conley index is nontrivial, then there is an associated nonempty invariant set [9]. However, it can also be used to prove the existence of equilibria [44, 33], periodic orbits [34], heteroclinic orbits [9], and chaotic dynamics [36, 45, 12].

The Conley index is defined to be the relative homology of an index pair [9]. Rather than providing the most general definition, we restrict our attention to a special case that

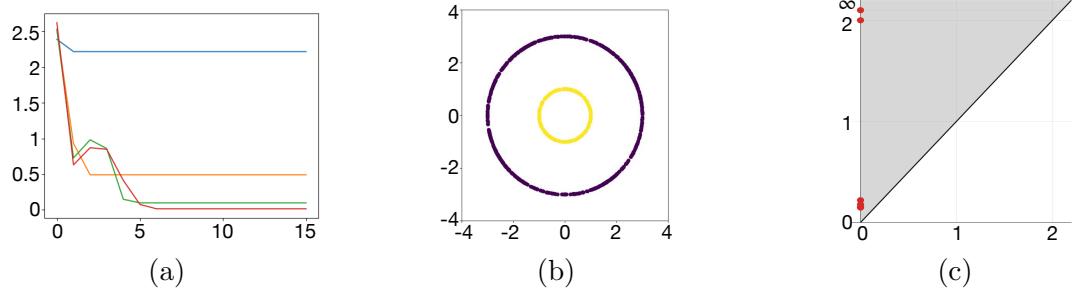


FIGURE 2. (a) Hausdorff distance between iterations for 10 points (blue), 100 points (orange), 1,000 points (green), and 10,000 points (red) for the radially symmetric flow defined by Equation (2). (b) The image of  $f_1^{14}(\mathcal{I})$  for  $\mathcal{I}$  consisting of 1,000 points. (c) The zero-dimensional persistence diagram of the Vietoris-Rips complex with vertices given by  $f_1^{14}(\mathcal{I})$ , shown in part (b). Two clearly separated points which are far from the diagonal indicate the presence of two distinct clusters.

is related to the computational efforts of this paper. An *attracting block* is an attracting neighborhood  $N$  with the property that  $\varphi(t, N) \subset \text{int}(N)$  for all  $t > 0$ . As in the case of attracting neighborhoods, attracting blocks form a bounded distributive lattice with partial order given by inclusion.

*Remark 2.4.* Assume that we are given a finite lattice of attracting blocks  $\mathbb{N}$ . If  $N_0, N_1 \in \mathbb{N}$  and  $N_0 \subset N_1$ , then  $H_*(N_1, N_0)$  is the Conley index for the maximal invariant set in  $N_1 \setminus N_0$ . In the case that  $N_1$  is minimal, then  $N_0 = \emptyset$ , as in Figure 1(a). Thus the Conley index is simply given by  $H_*(N_1)$ .

### 3. DATA-DRIVEN IDENTIFICATION OF ATTRACTING NEIGHBORHOODS

The goal of this section is to describe a data-driven computational scheme for identifying attracting neighborhoods.<sup>2</sup> The input is a collection of orbit segments from a finite set of initial conditions  $\mathcal{I} \subset X$ . In Section 3.1, we use the orbit segments to identify finite approximations of attractors  $A_k$  for  $k \in \{0, \dots, K\}$ . This produces a labeling function  $F: \mathcal{I} \rightarrow \{0, \dots, K\}$  that maps each initial condition to an attractor. In Section 3.2, we discuss the use of a neural network to extend the labeling function  $F$  to all of  $X$  and how the trained network can be used to identify an attracting neighborhood  $N_k$  for each  $A_k$ . In Section 3.3 we explain the neural network training.

**3.1. Attractor identification scheme.** The input for the data-driven identification of the attracting neighborhoods is a collection of orbit segments of a given ordinary differential equation (ODE). In this paper we consider the ODE on a hyperrectangle  $X$ , and the orbit segments start from initial conditions in  $\mathcal{I} \subset X$ . We use numerical simulation to approximate orbit segments of an explicit system, but in principle for physical systems, they could be obtained from experimental data. The set  $\mathcal{I}$  of the initial conditions for the input orbit segments can be a random sample from  $X$ , set of vertices of some grid, or a set constructed by some space-filling design. In this paper, we apply a Latin hypercube space-filling design [31] and perform the sampling by using the method `LatinHypercube` implemented in the `scipy.stats.qmc` module.

Different choices of  $\mathcal{I}$  might lead to identification of different sets  $A_k$  and  $N_k$ , but Theorem 2.1 guarantees that the constructed lattice of attractors is correct. However, if the sample  $\mathcal{I}$  is too small then the attractors might not be correctly identified. Similarly, learning of the labeling function  $F$  can be negatively affected by a small sample size. In this section we provide some heuristics to assess if the sample size is sufficient, but we do not conduct a

<sup>2</sup>Data used for the paper is available at: <https://github.com/begelb/MLCD-data>

systematic investigation of the sample size necessary to properly train the network. To avoid this issue, we use a reasonably large sample as explained in Section 6.1.

To generate samples of the orbit segments starting from initial conditions in the given set  $\mathcal{I}$ , we use a standard numerical ODE solver `odeint` implemented in the `scipy.integrate` library. For each initial condition  $x_0 \in \mathcal{I}$  we construct a sample  $\{x(i)\}_{i=0}^I$ , of the orbit  $x$  satisfying the ODE and initial value condition  $x(0) = x_0$ . The number of time steps  $I$  is the same for all the orbits. The sampled orbits provide an approximation of the time-1 map  $f_1$  and its iterates  $f_1^i$  for  $x \in \mathcal{I}$  and  $0 \leq i \leq I$ . To identify the asymptotic behavior from the sampled orbits we suppose that at time  $I$  the orbits are close to their  $\omega$ -limit sets. To check if this assumption is reasonable, for our choice of the time horizon  $I$ , we consider the Hausdorff distance  $d_H(f_1^i(\mathcal{I}), f_1^{i+1}(\mathcal{I}))$  between the images of the set  $\mathcal{I}$  under consecutive iterates of the time-1 map. In particular if the set  $\mathcal{I}$  is sufficiently dense and the time  $I$  is long enough to ensure that the orbits are already close to their attractors, then  $d_H(f_1^i(\mathcal{I}), f_1^{i+1}(\mathcal{I}))$  should be small. The appropriate size of  $\mathcal{I}$  and the value  $I$  are system dependent. However, since the goal is to identify attracting neighborhoods rather than precise dynamics on the attractors, there is considerable flexibility in the simulation time as well as the size and distribution of the initial points to obtain an accurate but potentially coarse result.

To demonstrate the behavior of the distances  $d_H(f_1^i(\mathcal{I}), f_1^{i+1}(\mathcal{I}))$ , we consider a radially symmetric ODE with an attractor consisting of two periodic orbits. The domain is chosen to be the square  $X = [-4, 4]^2$  and equations, in the polar coordinates, are given by

$$(2) \quad \begin{cases} \dot{\theta} = 1 \\ \dot{r} = -r(r-1)(r-2)(r-3) \end{cases}$$

Figure 2(a) shows that for large enough sample  $\mathcal{I}$  the distances  $d_H(f_1^i(\mathcal{I}), f_1^{i+1}(\mathcal{I}))$  become small around  $i = 6$  and then they are almost constant. This suggests that the points  $f_1^I(\mathcal{I})$  for  $I > 6$  are close to the respective attractors. Figure 2(b) depicts the set  $f_1^{14}$  for the longest integration time of our simulation. As expected this set is a good approximation of the two periodic orbits that form the distinct attractors of the system.

As demonstrated above, for sufficiently large  $I$  and reasonably dense  $\mathcal{I}$ , the set  $f_1^I(\mathcal{I})$  is a good approximation of the attractors. To identify the approximations of the attractors  $A_k$  we search for clusters in  $f_1^I(\mathcal{I})$ . There are many different clustering methods that can achieve this goal [47, 37]. We use a hierarchical clustering algorithm based on persistence homology [29, 8] of a Vietoris-Rips complex [7] defined on the points in  $f_1^I(\mathcal{I})$ . This method is similar to clustering with dendograms [38]. Points in the 0-dimensional persistence diagram [14] indicate scales at which distinct clusters merge together. The points in the persistence diagram with larger death coordinates correspond to better separated clusters [29]. The persistence diagram shown in Figure 2(c) contains two points whose death coordinates are considerably larger than death coordinates of the other points. This indicates that there are two distinct and well separated clusters corresponding to two circles visible in Figure 2(b). We identify the distinct clusters with the sets  $A_k$ , i.e. the set  $A_k$  consists of the points in  $f_1^I(\mathcal{I})$  that belong to the  $k$ -th cluster. Finally we define the labeling function  $F: \mathcal{I} \rightarrow \{0, \dots, K\}$  by  $F(x) := k$  if  $f_1^I(x) \in A_k$ .

**3.2. Attracting neighborhood construction.** Potential attracting neighborhoods  $N_k$  for the sets  $A_k$  are constructed by using a partition of the domain  $X$  into polytopes. This partition is induced by a neural network that extends the labeling function  $F: \mathcal{I} \rightarrow \{0, \dots, K\}$  to  $X$  [2]. There are many different machine learning methods for constructing this extension [28]. We consider a neural network model based on the MultiLayer Perceptron (MLP) [39, 13], so that the resulting neural network model is a piecewise-linear map  $F_\theta: \mathbb{R}^d \rightarrow \mathbb{R}$ . Polytopes on which this map is linear depend on the network parameters (weights)  $\theta$  [40], and there are algorithms to compute these polytopes from  $\theta$ , e.g. [32].

To keep the computations simple, we use a regression perceptron network with a single hidden layer to extend the function  $F$ . It is well known [22, 30] that for every continuous

function  $F: X \rightarrow \mathbb{R}$  defined on a compact set  $X$ , and  $\varepsilon > 0$  there exists a perceptron network  $F_\theta$  with a single hidden layer and weights  $\theta$  such that  $\sup_{x \in X} |F(x) - F_\theta(x)| < \varepsilon$ . We note that our function  $F$  is discontinuous at the boundaries of the basins of attraction so this result does not apply. However, due to finite size of our sample  $\mathcal{I}$ , there is a continuous function  $\bar{F}$  that agrees with  $F$  on  $\mathcal{I}$  and satisfies  $\bar{F}(x) = k$  for every  $x$  in some closed neighborhood of  $A_k$  for every  $k$ .

In practice, it is more efficient to use multiclass classification to construct the approximation of  $\bar{F}$ . This approach also yields a vector function  $p: X \rightarrow \mathbb{R}^k$  whose  $k$ -th coordinate  $p_k(x)$  expresses the probability that  $\bar{F}(x) = k$ . Using this vector we could define  $N_k := \{x \in X : p_k(x) \approx 1\}$ . If the accuracy of the classification is sufficiently high, then this set is an attracting neighborhood of  $A_k$ . There are many heuristic results about the accuracy of such classification methods, but only recently have the first theoretical results started to appear [46, 5]. Moreover, this method does not in general provide a decomposition of the sets  $N_k$  into polytopes.

Hence, as already mentioned, we consider a regression network with a single hidden layer of width  $p$ . This network is a map  $F_\theta: \mathbb{R}^d \rightarrow \mathbb{R}$  that maps the input value  $x \in \mathbb{R}^d$  to an output prediction that depends on the weights  $\theta$ . In particular,  $\theta$  is a triplet  $(A, b, w)$ , where  $A$  is a real matrix in  $M_{p \times d}(\mathbb{R})$  and  $b, w$  are  $p$ -dimensional real vectors. For a given  $\theta$  the map is defined by a scalar product

$$(3) \quad F_\theta(x) = h^{0,K-1}(\langle w, h^{0,1}(Ax + b) \rangle).$$

The nonlinear activation function  $h^{a,b}: \mathbb{R}^p \rightarrow \mathbb{R}^p$  is defined coordinate-wise as HardTanh

$$h_i^{a,b}(x_i) = \begin{cases} a & \text{if } x_i < a, \\ x_i & \text{if } a \leq x_i \leq b, \\ b & \text{if } x_i > b. \end{cases}$$

The function  $F_\theta$  is piecewise-linear, and we denote the supporting hyperplanes of the polytopes on which this function is linear by  $H_i^j$  with  $1 \leq i \leq p$  and  $j = 0, 1$ . The hyperplane  $H_i^j$  is defined by

$$(4) \quad \langle A[i], x \rangle + b_i = j$$

where  $A[i]$  is the  $i$ -th row of the matrix  $A$  and  $b_i$  is the  $i$ -th element of the vector  $b$ .

To take advantage of the existing code for computing homology in higher dimensions [16], we restrict the weights of the matrix  $A$  in such a way that the relevant polytopes are parallelotopes. For simplicity, we set the width of the network  $p = qd$  for some  $q \geq 1$  and divide the rows of the matrix  $A$  into  $d$  groups each containing  $q$  rows. Within each group we require that the rows are identical.

Each group of the identical rows of the matrix  $A$  induces  $2q$  parallel hyperplanes defined by Equation (4). Moreover, if the rows of the matrix  $A$  span  $\mathbb{R}^d$ , then two hyperplanes corresponding to rows in distinct groups are not parallel. In this case, the collection of all the hyperplanes  $H_i^j$  divides  $\mathbb{R}^d$  into a collection  $\{S_i\}_{i=1}^M$  of  $d$ -dimensional subsets that are either parallelotopes or infinitely long strips as in Figure 3. To ensure that the sets  $S_i$  intersecting  $X$  are bounded, we sometimes need to add two hyperplanes to each group of parallel hyperplanes. In the rest of the paper, the resulting collection of parallelotopes that intersect  $X$  is denoted  $\{C_i\}_{i=1}^N$ . Due to its cubical data structure, the decomposition of  $X$  given by this method is called a Machine-Learned Cubical Decomposition (MLCD).

We utilize the learned approximation  $F_\theta$  of the labeling function  $F$  to define the potential attracting neighborhoods as follows. Given  $\epsilon > 0$ , define

$$(5) \quad N_k := \bigcup_{C_i \in \mathcal{N}_k} C_i \quad \text{where} \quad \mathcal{N}_k = \{C_i : |F_\theta(x) - k| \leq \varepsilon \text{ for all } x \in C_i\}.$$

By Bauer's minimum principle,  $C_i$  is contained in  $\mathcal{N}_k$  if and only if  $|F_\theta(x) - k| \leq \varepsilon$  for all the vertices  $x$  of  $C_i$ . The union of all parallelotopes  $C_i$  that are not contained in any  $\mathcal{N}_k$  form the *uncertain region* denoted by  $U$ . We note that larger values of  $\varepsilon$  should be used if the quality

of the approximation  $F_\theta$  is low. In this paper, we do not investigate systematic methods for choosing  $\epsilon$ . Instead, we consider a range of values and indicate how the results depend on  $\epsilon$  for particular problems.

We close this section by pointing out that the chosen nonlinearity  $h^{a,b}$  allows us to approximate a broader class of functions than the standard nonlinearity ReLU. However, already for two-dimensional domains, this network cannot achieve an arbitrarily good approximation mentioned at the beginning of this section. To see this consider an arbitrary function  $F_\theta$ , and let  $u_1$  and  $u_2$  be two linearly independent rows of the matrix  $A$ . Note that in the coordinates given by this basis,  $F_\theta(x, y) = f_1(x) + f_2(y)$  for some functions  $f_1, f_2: \mathbb{R} \rightarrow \mathbb{R}$ . Clearly this class of functions is too restrictive to represent an arbitrary step function. Thus to study general dynamical systems it is necessary to use an unconstrained network architecture and deal with the general polytopes. In Section 4 we provide heuristics for detecting if the network is sufficiently versatile to adequately approximate the labeling function. The same heuristics can be applied to unconstrained networks to decide if their width and depth is sufficient.

**3.3. Training and benchmarking.** In this section we describe the initialization and training procedure for the neural network  $F_\theta$ , which is trained to approximate the labeling function  $F$ . The values of the training hyperparameters are reported in Section 6.1. We typically use a testing dataset with size equal to the size of the training dataset. For systems where producing balanced data is more expensive, we use a testing to training data ratio of 1 : 4.

In order to ensure that data sparsity does not affect the results, we create conservatively large training datasets  $\mathcal{I}$ . In cases where the sampling procedure produces unbalanced data, oversampling of the least witnessed class is used as the primary technique to produce balanced training data. This technique is insufficient for the ellipsoidal bistable system in four and five dimensions described in Section 4.2 due to the decay of the volume of the ellipsoidal attracting neighborhood compared to the volume of the considered domain  $X$  as dimension increases. To avoid the lengthy simulations of the orbits we exploit the knowledge of the attracting neighborhoods. We non-uniformly sample the domain  $X$  and determine to which attracting neighborhood points belong without integrating. The ratio of the points outside and inside the ellipsoid (70 : 30 in four dimensions and 86 : 14 in five dimensions) is determined from preliminary experiments.

To study the performance of our method, we create 100 different realizations of  $F_\theta$  for each system. The initial weights of each realization depend on the size of the domain  $X = \prod_{i=1}^d [a_i, b_i]$  and the number of nodes  $p = qd$ . We recall that we split the nodes into  $d$  groups, and the  $q$  nodes in each group correspond to  $2q$  parallel hyperplanes in  $\mathbb{R}^d$ . We always choose the initial weights in such a way that the hyperplanes in the  $i$ -th group are perpendicular to the  $i$ -th standard basis vector  $e_i$ . To be more precise the rows of the matrix  $A$  corresponding to the  $i$ -th group of nodes are set equal to the transpose of  $e_i$ . Only the weights corresponding to the hyperplane offsets are chosen randomly. We want these hyperplanes to be initially spread along the domain  $X$ , so for the  $i$ -th group we sample the offsets randomly according to the uniform distribution on  $(a_i, b_i)$ .

To train the network, we use the Adam optimizer [27] with learning rate 0.01. We compute a test loss on a testing dataset at every epoch of training. We use a validation-based early stopping criterion defined by a patience parameter  $\rho$  so that the training stops if the average of the test loss over the last  $\rho$  epochs stops decreasing. Otherwise we train for a problem-dependent number of epochs specified in Section 6.1.

There are two cases in which the realization is not included in the set of 100 realizations: 1) if the normal vectors of the hyperplanes do not span  $\mathbb{R}^d$ , and 2) the training process did not converge. To assess if the training process did not converge, we use a heuristic based on reduction of the loss between the first and last epoch. If the fraction of the training losses in the first and last epoch is less than a given parameter  $\beta$ , then the training did not converge, and we do not use this realization. We do not investigate how to systematically choose  $\beta$  and  $\rho$ , but report the values used for each example in Section 6.1.

We establish a benchmark to compare the number of polytopes from the Machine-Learned Cubical Decomposition (MLCD) to the number of grid elements needed to construct the sets  $N_k$  and  $U$  from a regular cubical decomposition, by which we mean a partition of  $X$  by uniform subdivision in each coordinate direction, as shown in Figure 7(c). First, we train an (unconstrained) multilayer perceptron network using `sklearn.MLPClassifier` to label points according to their basin of attraction. The network is trained until the test accuracy exceeds 95% on a test set consisting of 10% of the original data. The vertices of a regular cubical grid on  $X$  are labeled by the classifier. If all vertices of a cube have the same label  $k$ , then the cube is included in  $N_k$ ; otherwise, it is included in  $U$ . In Section 4, we consider the minimum grid size required to recover correct Morse representations and Conley indices of the corresponding Morse sets, as described by the homology of the sets  $N_k$  and  $U$  listed in Section 6.2.

#### 4. ATTRACTOR IDENTIFICATION RESULTS

In this section we demonstrate that the Machine-Learned Cubical Decomposition (MLCD) can provide correct Morse representations of a dynamical system together with Conley indices of its Morse sets. We also show that the number of polytopes in the MLCD is significantly smaller than the number of cubes required by a regular cubical decomposition. In Section 4.1 we start with simple dynamical systems whose attractors consist of fixed points and familiarize the reader with the workflow of MLCD. Then in Section 4.2 we move to systems containing periodic orbits and highlight the small number of polytopes in MLCD. Finally, in Section 4.3 we consider more complex systems.

All the statistics presented are based on 100 independent realizations of the network  $F_\theta$ , described in Section 3.3. While we cannot achieve the rigor of the universal approximation theorem [22], we argue that the following remark, corroborated by all the examples considered in this paper, provides a useful guideline for practical applications.

*Remark 4.1.* If the architecture of the network is sufficiently expressive and a sufficiently small value of the loss function is achieved during the training, then the approximation  $F_\theta$  of the labeling function  $F$  should lead to the correct Conley indices. Moreover, if the computed Conley indices are the same for all the realizations with the final testing loss below some value, then this value is likely to be sufficiently small. However, if the indices keep changing for all observed loss values, then the network is not sufficiently expressive.

The rational of this remark relies on the fact that if the testing loss is small, then  $F_\theta$  closely approximates the piecewise constant labeling function  $F$ . Hence it is likely that for small values of  $\varepsilon$ , each set  $N_k$ , defined by (5), is a subset of  $F^{-1}(k)$  and has the same Conley index as the attractor  $A_k$ . In this paper, we consider  $\varepsilon \in \{0.1, 0.2, 0.3, 0.4, 0.49\}$  and often report the results for the  $\varepsilon$  that yields the highest success rate. The detailed study of the dependence on  $\varepsilon$  is beyond the scope of this paper, and we only discuss it briefly in Section 5.

**4.1. Fixed points.** The first system that we consider is an ODE on the rectangle  $X = [-2, 2] \times [-3.5, 3.5]$  given by:

$$(6) \quad \dot{x} = u(1 - u^2) \quad \text{and} \quad \dot{y} = u^2(3 - 2u^2) - v$$

where

$$u = \frac{x}{2} + \frac{\sqrt{3}}{2}y \quad \text{and} \quad v = \frac{y}{2} - \frac{\sqrt{3}}{2}x.$$

This system has two stable fixed points at approximately  $(-0.366, 1.366)$  and  $(-1.366, -0.366)$ . Their basins of attraction are separated by the straight line  $y = -\sqrt{3}x$ , which is the stable manifold of the saddle point at the origin.

To produce the MLCD we first sample the system and construct the labeling function  $F: \mathcal{I} \rightarrow \{0, 1\}$  as described in Section 3.1. The shape of the function  $F$ , shown in Figure 3(a), clearly indicates two distinct basins of attractions separated by a straight line. Due to this simple geometry a very small network provides a good approximation of  $F$ . Figure 3(b)

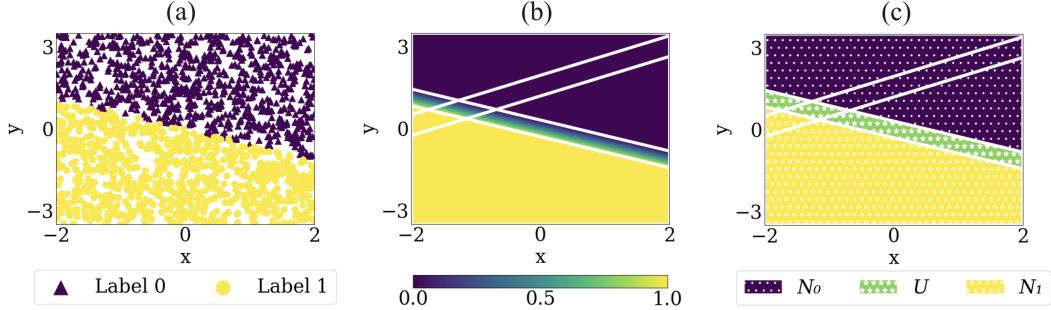


FIGURE 3. (a) A sample of the labeling function  $F$  for the bistable, two-dimensional system given by (6). (b)  $F_\theta$  (single realization) approximates  $F$  and is linear on the parallelotopes bounded by the hyperplanes displayed in white. These hyperplanes partition the set  $X$  into one parallelotope and eight infinitely long strips. By adding two hyperplanes to each set of the parallel hyperplanes, shown in white, the set  $X$  is partitioned into nine finite parallelotopes. (c) Parallelotopes in the sets  $N_0$ ,  $N_1$  and  $U$  for the given  $F_\theta$ . In this case the same parallelotopes are obtained for all considered values of  $\varepsilon$ .

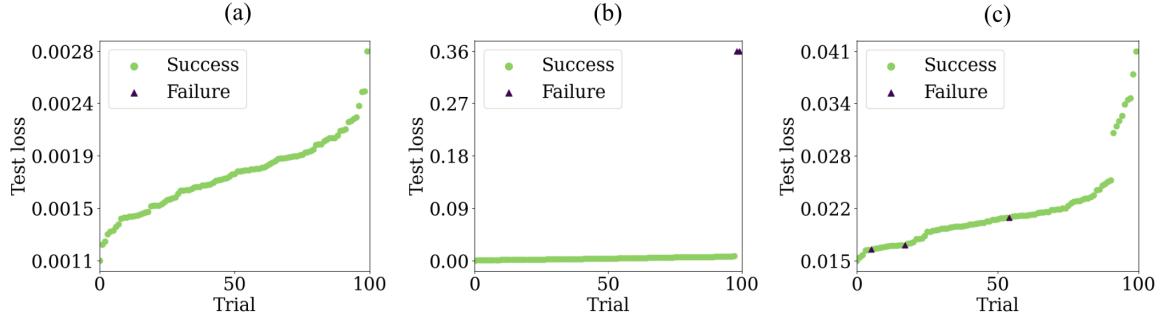


FIGURE 4. The final values of the test loss for successes (green points) and failures (violet triangles) obtained by networks with one node per dimension for the (a) Two-dimensional system with the linear separatrix given by (6), (b) Four-dimensional system with the nonlinear separatrix given by (7), and (c) Six-dimensional EMT Hill system given by (15).

documents that one hidden neuron (two hyperplanes) per dimension is sufficient to separate the basins of attraction from the separatrix. Hence, we use a network with width  $p = 2$ .

For two-dimensional input, the constrained network  $F_\theta$  with two hidden neurons typically divides  $\mathbb{R}^2$  into nine subsets out of which eight are unbounded. For the  $F_\theta$  depicted in Figure 3(b), we need to add two hyperplanes in each direction to ensure that the domain  $X$  is covered by finite parallelotopes. After adding the hyperplanes, the set  $X$  is covered by nine parallelotopes and the sets  $N_0$ ,  $N_1$  and  $U$ , shown in Figure 3(c), are identical for all considered values of  $\varepsilon$ . Moreover, their Conley indices are correct; see Section 6.2 for the correct Conley indices of all the systems considered in this paper.

Now we study the performance on 100 independent realizations of  $F_\theta$ . For each realization we record the final test loss and the number of polytopes in the MLCD (by definition, all polytopes intersect  $X$ ). We also check if the Conley indices computed using the MLCD are correct. If yes, we label the realization as a success. Otherwise we label it as a failure. Figure 4(a) shows the final test losses for the individual realizations. All these values are relatively small and the sets  $N_0$ ,  $N_1$  and  $U$  constructed from grids generated by each realization  $F_\theta$  always yield the correct Conley indices. The average number of polytopes (over 100 trials) is six while the standard deviation is one. Hence, even in this simple case MLCDs slightly

outperform the regular cubical decomposition, which requires nine uniform cubes to obtain the correct Conley index. Moreover, the number of polytopes is insensitive to the orientation of the separatrix while the number of cubes increases significantly if the separatrix is closely aligned with the diagonal of the domain.

Next we consider a higher-dimensional system with two stable fixed points in the domain  $X = [-2, 2] \times [-3.5, 3.5] \times [-2, 2] \times [-2, 2]$ . The system is given by the following ODE:

$$(7) \quad \begin{cases} \dot{x}_1 = -x_1 \\ \dot{x}_2 = (x_2 - x_1^2)(9 - x_2^2) \\ \dot{x}_3 = -x_3 \\ \dot{x}_4 = -x_4 \end{cases}$$

The basins of attraction of the two stable fixed points are now separated by a three-dimensional stable manifold of a saddle equilibrium. Since neither the curvature of the manifold nor the size of the set  $X$  is too large, it is again possible to delineate the separatrix from the basins of attraction by using a neural network with one node per dimension.

In this case, 2% of the trained networks are failures while the other 98% are successes for all values considered of  $\varepsilon$ . Figure 4(b) shows the final test loss for all realizations from both successes and failures. All the trials with small final loss lead to the same Conley indices. As predicted by Remark 4.1, all these trials are successes. The improvement over the regular cubical decomposition, which requires 81 cubes, is more remarkable because on average MLCD only produces 33 parallelotopes (the standard deviation is 10).

We also test the very small network architecture ( $q = 1$  and  $p = 6$ ) on a six-dimensional system defined by (15). This system contains two fixed points, but the geometry of the separatrix is more complicated. We still obtain a 97% success rate. The small size of the network results in a mean number of 331 parallelotopes in the MLCD (with a standard deviation of 90), whereas the regular cubical decomposition produces 46,656 cubes. However, the final test losses, depicted in Figure 4(c), do not indicate stabilization of the Conley indices for small values. As predicted by Remark 4.1, the network is too small to sufficiently approximate the labeling function. This is also corroborated by the relatively high value of the lowest loss achieved over 100 realizations. We note that in this case, the results are very sensitive to the particular choice of  $\varepsilon$ . When the size of the network is increased to  $q = 2$  nodes per dimension and width  $p = 12$ , the separation between the successes and failures is more pronounced and the results are less sensitive to the value of  $\varepsilon$ . Even for this larger network, the average number of parallelotopes is 2,631 which is still a significant improvement. However, we do not achieve a convincingly low test loss. This is likely a consequence of the restriction imposed on the matrix  $A$  in Section 3.2. We discuss this problem in more detail in Section 4.3.

**4.2. Fixed points and periodic orbits.** In this section we consider systems with basins of attraction of more complicated structure. The first system is an ODE on the set  $X = [-4, 4] \times [-4, 4]$ . The equations are given in polar coordinates by:

$$(8) \quad \begin{cases} \dot{\theta} = 1 \\ \dot{r} = -r(r - 1)(r - 2)(r - 3) \end{cases}$$

The asymptotic dynamics of the system includes two attracting periodic orbits, a repelling periodic orbit as a separatrix, and the origin as a repelling fixed point. Figure 5(a) shows the finest Morse representation. In this representation, the repelling fixed point at the origin is a distinct Morse set. However, due to the finite nature of the data, this unstable structure cannot be fully resolved. The constructed labeling function depicted in Figure 6(a) misses the fact that the origin is a fixed point. Hence, the neural network  $F_\theta$  presented in Figure 6(b) combines the origin, the attracting limit cycle of radius one, and the connecting orbits from the origin to the periodic orbit into a single attractor. Figure 5(b) shows a coarser Morse representation obtained for the sets  $N_0, N_1$ , and  $U$  depicted in Figure 6(b). Although we

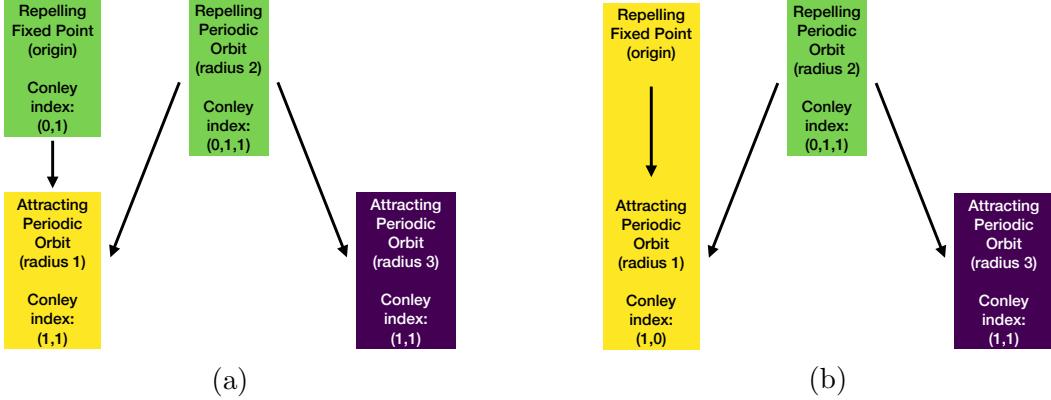


FIGURE 5. (a) Finest Morse representation for the radially symmetric system defined by Equation (8). (b) A coarsened Morse representation that still exhibits the bistability of the system.

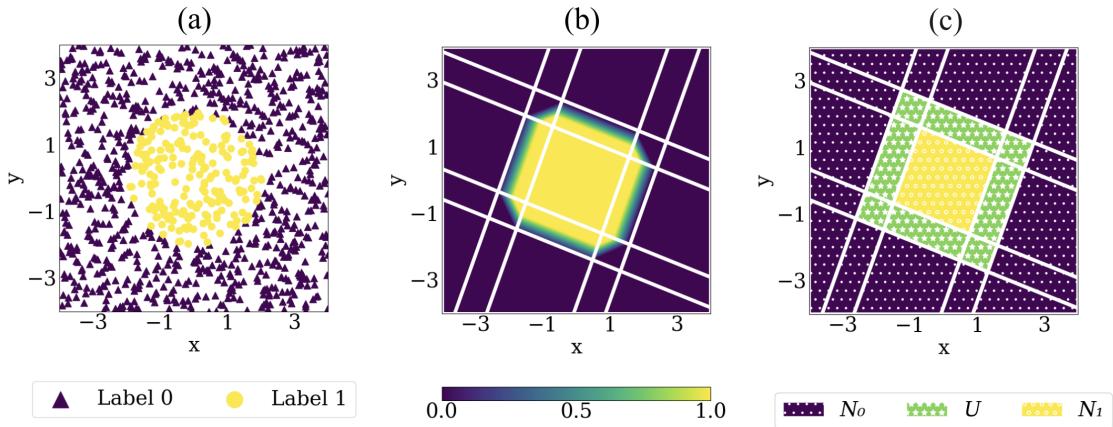


FIGURE 6. (a) A sample of the labeling function  $F$  for the radially symmetric system defined by Equation (8). (b)  $F_\theta$  (single realization) approximates  $F$  and is linear on the parallelotopes bounded by the hyperplanes displayed in white. (c) Parallelotopes in the sets  $N_0$ ,  $N_1$  and  $U$  for the given  $F_\theta$ .

cannot find the finest Morse representation, we still properly capture the bistability in the system.

Figure 6 demonstrates that two hyperplanes per dimension are not sufficient to separate the basins of attraction and the separatrix. We use the next smallest network and set the number of hidden nodes per dimension equal to  $q = 2$ , which results in a network of width  $p = 4$ . The same figure shows that this small network can provide a sufficient approximation  $F_\theta$  of the labeling function. In particular, the sets  $N_1$ ,  $N_2$ , and  $U$  yield the correct Conley indices of the Morse sets in the coarser representation.

More complicated geometry of the attractor and the small size of the network result in a lower success rate, which is 31%. All the realizations with sufficiently small final test loss yield the same indices and as predicted by Remark 4.1, they are correct. However, for this system there is no improvement over the uniform grid. Figures 6(b) and (c) show a typical grid obtained by MLCD and the structure of the sets  $N_0$ ,  $N_1$  and  $U$ . There are always 25 polytopes which is exactly the necessary number of cubes in the regular grid. Figure 7 shows that if the system is transformed so that the periodic orbit is an ellipse with diagonal-aligned axes, then the improvement is significant.

More precisely, to study the dependence of the number of grid elements on the dimension we modify the system defined by Equation (8) by a coordinate change. This coordinate change

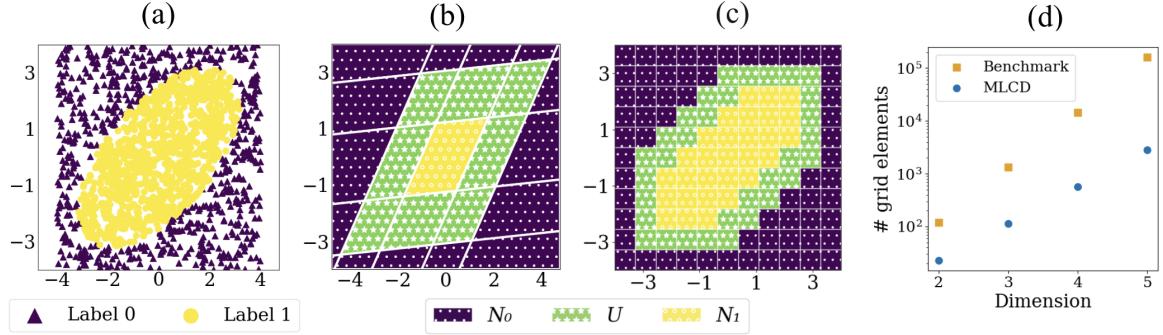


FIGURE 7. (a) A sample of the labeling function  $F$  for the radially symmetric system defined by Equation (8) after the coordinate change. (b) Parallelotopes produced by the MLCD that form the sets  $N_0$ ,  $N_1$  and  $U$ . (c) Cubes in the regular cubical decomposition that form the sets  $N_0$ ,  $N_1$  and  $U$ . d) Number of parallelotopes and cubes necessary to capture the sets  $N_0$ ,  $N_1$  and  $U$  versus the dimension of the system.

deforms the circular periodic orbit to an ellipsoidal periodic orbit. The labeling function (see Figure 7(a)) shows the geometry of the basins of attraction for the modified system. We extend this system to 3, 4 and 5 dimensions so that the separatrix given by the unstable periodic orbit changes to an ellipsoid that separates the space into two basins of attraction.

To approximate the labeling function for the higher-dimensional systems we again use the network architecture with two hidden nodes per dimension. Figure 8 indicates that as predicted by Remark 4.1, the computed Conley indices are correct for a sufficiently small test loss. We note that realizations that achieve a reasonably small test loss become progressively more rare as the dimension of the systems increases. This leads to a progressively smaller percentage of successes. In particular, the success rates are 40%, 18%, 6% and 4% as the dimension increases. However, training the network is inexpensive, and a realization with a small loss can be found very quickly. In what follows we demonstrate that the amount of time required to find a low loss approximation is negligible compared to the savings in computing the Conley indices (homology).

Figure 7(b) shows a typical grid produced by MLCD. In each direction, the set  $X$  intersects with 4 hyperplanes. Hence, the MLCD of  $X$  contains  $5^2$  parallelotopes. Figure 7(c) depicts the regular grid, formed by subdividing the set  $X$  with 10 hyperplanes in each direction, resulting in  $11^2$  cubes. As we increase the dimension, the number of required hyperplanes per dimension is roughly preserved, i.e. 4 for MLCD and 10 for the uniform grid. Therefore, the number of grid elements grows exponentially for both grids, but the base of the exponential function is reduced by more than half by using MLCD. Figure 7(d) indicates that as the dimension  $n$  of the system increases, the grid sizes indeed grow as  $5^n$  and  $11^n$  respectively.

**4.3. Systems with more complex attractors.** In Section 3.2, we describe a restriction of the weight matrix  $A$  of the neural network (3) so that the polytopes in the grid are parallelotopes. This restriction allows us to easily identify the grid elements and also to use efficient (cube-based) code for computing homology. In this section, we present two simple examples to show that this approach is too restrictive. The first system is given by an ODE

$$(9) \quad \begin{cases} \dot{\theta} = 1 \\ \dot{r} = -r(r-1)(r-2)(r-3)(r-4) \end{cases}$$

on the domain  $X = [-5, 5] \times [-5, 5]$ . This system has two attracting periodic orbits, an attracting fixed point at the origin, and two repelling periodic orbits acting as separatrices. Figure 9(a) depicts the sample of the labeling function for this system.

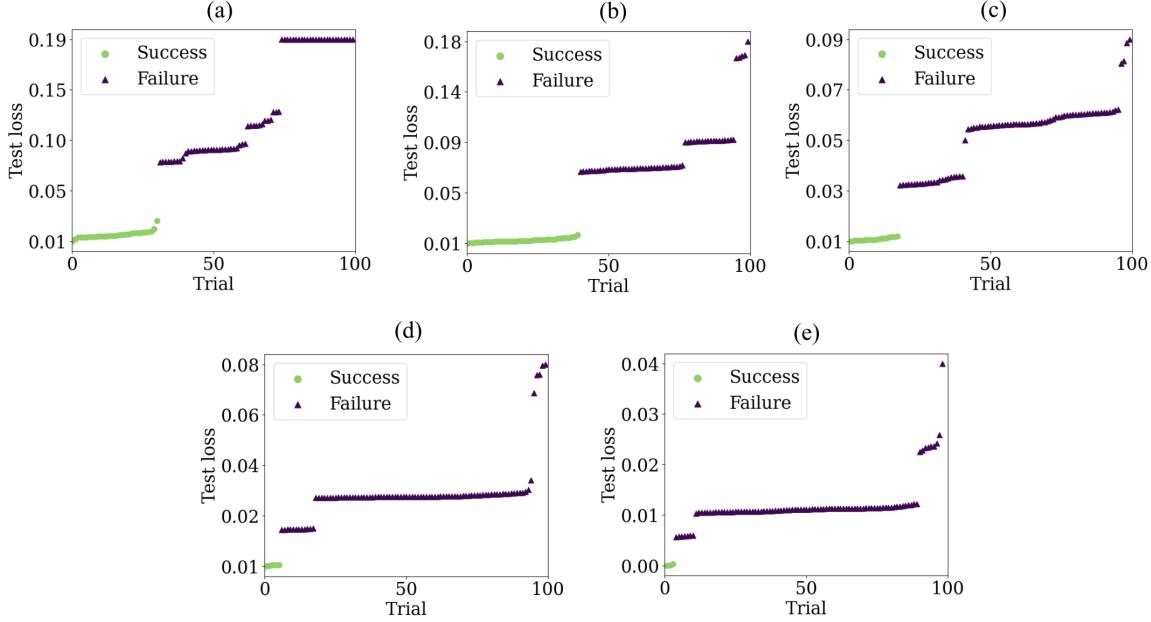


FIGURE 8. The final values of the test loss for successes (green points) and failures (violet triangles) obtained by networks with two nodes per dimension. (a) Two-dimensional radial bistable system given by (8). The ellipsoidal bistable system in dimension (b) 2, (c) 3, (d) 4 and (e) 5.

We recall that the restricted network can be expressed as  $F_\theta(x, y) = f_1(x) + f_2(y)$  by using a coordinate system given by the basis of the independent rows of the matrix  $A$ , which is aligned with the hyperplanes (white lines) shown in Figure 9(b). Regardless of the size of the network, it is impossible to find a function  $F_\theta$  that properly approximates the radially symmetric labeling function with three distinct integer values. Typically, the region where the labeling function is equal to one is poorly resolved, and subsequently the set  $N_1$  often does not have the right topology (homology). Figure 10(a) shows that the correct homology is sometimes recovered. However, the large value of the lowest test loss and the fact that there is no separation between the successes and failures suggest that the architecture of the network is not sufficiently expressive and the results should not be trusted.

For the three-dimensional Hill system defined by Equation (13), the labeling function  $F$  and its approximation cannot be easily visualized. However, we know that this system has four minimal attractors consisting of a periodic orbit and three fixed points. Even without plotting the basins of attraction, we can detect that our method is not successful. This again follows from Remark 4.1. Inspection of the final loss values, plotted in Figure 10(b), reveals that they are all relatively large and the Conley indices do not stabilize as we approach the minimal loss achieved over the 100 runs.

This example highlights one more issue that has to be carefully considered. In particular, our simulation suggests that one fixed point has a comparatively small basin of attraction; points limiting to this fixed point make up less than 4% of the initial dataset  $\mathcal{I}$ . If the data is not properly balanced, then training the network can result in small loss even if one of the minimal attractors is missed. The last two examples clearly demonstrate that further research and algorithm development is necessary to realize the full potential of the proposed method.

## 5. CONCLUSION

In this paper we presented a data-driven method for quantifying the dynamics of complex systems. The method is based on Conley's topological approach that provides a framework for quantifying the global dynamics. The main advantage of this framework over a wide range

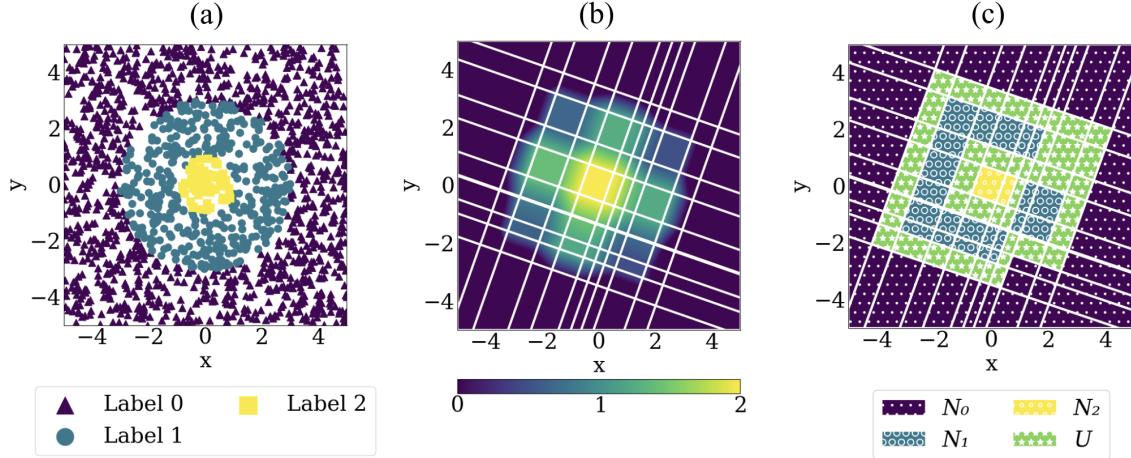


FIGURE 9. (a) A sample of the labeling function  $F$  for the system defined by Equation (9). (b)  $F_\theta$  (single realization) demonstrates that the restricted network architecture cannot properly approximate the labeling function. Thus (c) the sets  $N_0$ ,  $N_1$ ,  $N_2$ , and  $U$  do not capture the radially symmetric nature of the basins of attraction and separatrixes.

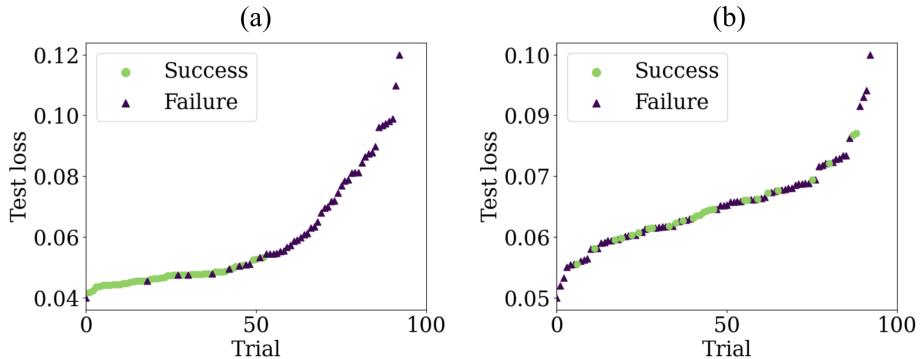


FIGURE 10. The final values of the test loss for successes (green points) and failures (violet triangles) for (a) two-dimensional system given by (9) and (b) three-dimensional Hill system defined by (13).

of other methods is its robustness with respect to perturbations and hence to noise. It has been successfully applied to a variety of systems. However, its application to high-dimensional systems is still challenging. The major computational bottleneck is the discretization of the phase space, which is typically done using a regular cubical decomposition. The number of grid elements in this grid grows exponentially with the dimension. While the exponential growth of the grid elements cannot be avoided, we demonstrated that it is possible to dramatically reduce their number by using machine-learned cubical decompositions. Table 1 highlights the reduction in the number of the grid elements necessary to properly resolve the dynamics for the systems considered in Section 4.

Most data-driven methods do not provide any theoretical guarantees that the reconstructed dynamics is correct. For our method, the main limitation for providing rigorous theoretical guarantees stems from a lack of control over the quality of the approximation  $F_\theta$  of the labeling function  $F$ . Assessing the quality of the approximation  $F_\theta$  produced by a neural network is a deep open problem in machine learning. In most applications, quality of the approximation is estimated by holdout methods in which the loss function is evaluated on a part of the dataset that is not used for training. We used this final testing loss in Remark 4.1 to formulate

Comparison of Cube Counts to Benchmark					
Example	Eqn. #	$\dim(X)$	Regular grid	MLCD	
			Min. # Cubes	Mean # cubes	SDEV
Linear separatrix	(6)	2	9	6	1
Radial bistable	(8)	2	25	25	0
Radial tristable	(9)	2	169	120	3
Nonlinear separatrix	(7)	4	81	33	10
Hill system with PO	(13)	3	1,728	705	177
EMT Hill system	(15)	6	46,656	331	90
Ellipsoidal bistable	Sec. 4.2	2	121	23	0
Ellipsoidal bistable	Sec. 4.2	3	1,331	115	2
Ellipsoidal bistable	Sec. 4.2	4	14,641	571	1
Ellipsoidal bistable	Sec. 4.2	5	161,051	2,852	45

TABLE 1. Comparison of the minimum number of cubes needed to obtain the correct Conley indices using a regular cubical decomposition to the mean and standard deviation (SDEV) of the number of cubes from successful trials of MLCD.

the following heuristic. We claim that if the final testing loss is sufficiently small then the computed Conley indices are correct. This is corroborated by all the examples considered in Section 4. To decide if the final loss is sufficiently small we compute several realizations of the function  $F_\theta$ . If the computed Conley indices are the same for all the realizations with the final testing loss below some value, then this value is likely to be sufficiently small.

To carry out the above mentioned computations, we have restricted the weights of the neural network (3). As discussed in Section 3.2, this restriction ensures that a grid decomposition consists of parallelotopes, and we can use efficient algorithms based on a cubical complex to compute homology in higher dimensions [16]. However, as discussed in Section 4.3, this approach is too restrictive, and techniques based on more general polygonal decompositions are necessary to efficiently handle more complex dynamics. In [32], methods to extract the polygonal decomposition generated by a ReLU neural network are presented for dimensions 2 and 3, but the efficacy and efficiency of these methods in higher dimensions require further investigation.

Algorithms for computing the homology of a general polygonal complex are presented in [21] and implemented in [16]. Nevertheless, the implementation in the context of cubical complex benefits from the rigid cell structure of cubes [20] that allows for efficient computations even in moderately high dimensions, e.g., ten dimensions [19]. To successfully deal with higher dimensional problems requires new ideas. A potentially promising approach (see [48]) is to use autoencoding neural networks to encode the dynamics in a lower-dimensional latent space on which the techniques of this paper can be applied.

Understanding global dynamics requires identifying unstable structures as well as attractors. However, for all examples in Section 4, we do not attempt to resolve the structure of the global dynamics beyond the minimal attractors, i.e. we do not resolve the unstable Morse set  $M$  shown in Figure 1(c). Indeed, we group all cells of the MLCD that do not belong to an approximate attracting neighborhood  $N_k$  into a single uncertain set  $U$ .

In Section 4.3, the dynamics of the system given by (9) has three attractors separated by two unstable periodic orbits. As shown in Figure 10, the MLCD often does recover the correct homology, but there is no separation between successes and failures. However, in the successful cases, the uncertain set  $U$  has two connected components (and has the homology of two disjoint circles).

Figure 11 shows the final values of the test loss for a three-dimensional system with three attracting fixed points and two unstable fixed points. In this case, success is characterized by whether  $U$  has two connected components, and the existence of three attracting neighborhoods

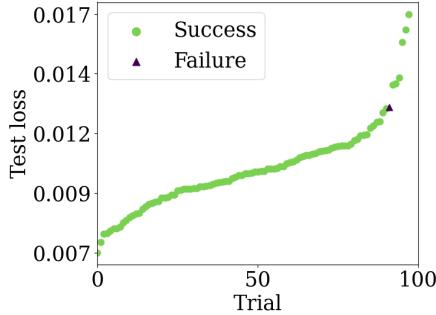


FIGURE 11. The final values of the test loss for successes (green points) and failures (violet triangle) for a three-dimensional system with two repelling fixed points and three attracting fixed points.

each with Conley index  $(1, 0, 0)$ . There is success for all trials with sufficiently low loss, and there is separation between the successes and failures, since the geometry of the sets is more simple. This suggests that these methods can be used to characterize further unstable structures, and we leave this to future research.

#### ACKNOWLEDGEMENTS

B.G. was supported by the National Science Foundation under Grant DGE-1842213. M.G., W.K., and K.M. were partially supported by the Air Force Office of Scientific Research under awards numbered FA9550-23-1-0011 and FA9550-23-1-0400. The authors acknowledge the Office of Advanced Research Computing (OARC) at Rutgers, The State University of New Jersey for providing access to the Amarel cluster and associated research computing resources that have contributed to the results reported here.

#### REFERENCES

- [1] Zin Arai et al. “A Database Schema for the Analysis of Global Dynamics of Multiparameter Systems”. In: *SIAM Journal on Applied Dynamical Systems* 8.3 (2009), pp. 757–789. DOI: [10.1137/080734935](https://doi.org/10.1137/080734935).
- [2] Randall Balestroiero et al. “The Geometry of Deep Networks: Power Diagram Subdivision”. In: *Advances in Neural Information Processing Systems*. Ed. by H. Wallach et al. Vol. 32. Curran Associates, Inc., 2019. URL: [https://proceedings.neurips.cc/paper\\_files/paper/2019/file/0801b20e08c3242125d512808cd74302-Paper.pdf](https://proceedings.neurips.cc/paper_files/paper/2019/file/0801b20e08c3242125d512808cd74302-Paper.pdf).
- [3] Bogdan Batko et al. “Conley Index Approach to Sampled Dynamics”. In: *SIAM Journal on Applied Dynamical Systems* 19.1 (2020), pp. 665–704.
- [4] Jan Bouwe van den Berg and Jean-Philippe Lessard. “Rigorous numerics in dynamics”. In: *Notices Amer. Math. Soc.* 62.9 (2015), pp. 1057–1061. ISSN: 0002-9920,1088-9477. DOI: [10.1090/noti1276](https://doi.org/10.1090/noti1276). URL: <https://doi.org/10.1090/noti1276>.
- [5] Monica Bianchini and Franco Scarselli. “On the complexity of neural network classifiers: A comparison between shallow and deep architectures”. In: *IEEE transactions on neural networks and learning systems* 25.8 (2014), pp. 1553–1565.
- [6] Steven L. Brunton, Joshua L. Proctor, and J. Nathan Kutz. “Discovering governing equations from data by sparse identification of nonlinear dynamical systems”. In: *Proc. Natl. Acad. Sci. USA* 113.15 (2016), pp. 3932–3937. ISSN: 0027-8424,1091-6490. DOI: [10.1073/pnas.1517384113](https://doi.org/10.1073/pnas.1517384113). URL: <https://doi.org/10.1073/pnas.1517384113>.
- [7] Gunnar Carlsson. “Topological pattern recognition for point cloud data”. In: *Acta Numerica* 23 (2014), pp. 289–368. DOI: [10.1017/S0962492914000051](https://doi.org/10.1017/S0962492914000051).

- [8] Frédéric Chazal et al. “Persistence-Based Clustering in Riemannian Manifolds”. In: *J. ACM* 60.6 (Nov. 2013). ISSN: 0004-5411. DOI: 10.1145/2535927. URL: <https://doi.org/10.1145/2535927>.
- [9] Charles Conley. *Isolated Invariant Sets and the Morse Index*. Vol. 38. CBMS Regional Conference Series in Mathematics. American Mathematical Society, Providence, R.I., 1978, pp. iii+89. ISBN: 0-8218-1688-8.
- [10] William D. Kalies, Konstantin Mischaikow, and Robert C.A.M. Vandervorst. “Lattice structures for attractors I”. In: *Journal of Computational Dynamics* 1.2 (2014), pp. 307–338. DOI: 10.3934/jcd.2014.1.307.
- [11] S. Day, O. Junge, and K. Mischaikow. “A rigorous numerical method for the global analysis of infinite-dimensional discrete dynamical systems”. In: *SIAM J. Appl. Dyn. Syst.* 3.2 (2004), 117–160 (electronic). ISSN: 1536-0040.
- [12] Sarah Day and Rafael Frongillo. “Sofic shifts via Conley index theory: computing lower bounds on recurrent dynamics for maps”. In: *SIAM J. Appl. Dyn. Syst.* 18.3 (2019), pp. 1610–1642. DOI: 10.1137/18M1192007. URL: <https://doi.org/10.1137/18M1192007>.
- [13] Walter H Delashmit, Michael T Manry, et al. “Recent developments in multilayer perceptron neural networks”. In: *Proceedings of the seventh annual memphis area engineering and science conference, MAESC*. Vol. 7. 2005, p. 33.
- [14] Herbert Edelsbrunner and John Harer. “Persistent homology—a survey”. In: *Surveys on discrete and computational geometry*. Vol. 453. Contemp. Math. Amer. Math. Soc., Providence, RI, 2008, pp. 257–282. ISBN: 978-0-8218-4239-3. DOI: 10.1090/conm/453/08802. URL: <https://doi.org/10.1090/conm/453/08802>.
- [15] Ulderico Fugacci, Federico Iuricich, and Leila De Floriani. “Efficient computation of simplicial homology through acyclic matching”. In: *2014 16th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing*. IEEE. 2014, pp. 587–593.
- [16] Marcio Gameiro. *pyCHomP: Computational Homology Project (with Python bindings)*. <https://github.com/marciogameiro/pyCHomP2>. 2024.
- [17] Marcio Gameiro, Brittany Gelb, and Konstantin Mischaikow. *Rigorously Characterizing Dynamics with Machine Learning*. 2025. arXiv: 2505.17302 [math.DS]. URL: <https://arxiv.org/abs/2505.17302>.
- [18] Marcio Gameiro et al. *Global Dynamics of Ordinary Differential Equations: Wall Labelings, Conley Complexes, and Ramp Systems*. 2024. arXiv: 2412.11078 [math.DS]. URL: <https://arxiv.org/abs/2412.11078>.
- [19] Shaun Harker, Konstantin Mischaikow, and Kelly Spendlove. “A computational framework for connection matrix theory”. In: *J. Appl. and Comput. Topology* 5.3 (2021), pp. 459–529. DOI: 10.1007/s41468-021-00073-3. URL: <https://doi.org/10.1007/s41468-021-00073-3>.
- [20] Shaun Harker, Konstantin Mischaikow, and Kelly Spendlove. *Morse Theoretic Templates for High Dimensional Homology Computation*. 2021. arXiv: 2105.09870 [math.AT]. URL: <https://arxiv.org/abs/2105.09870>.
- [21] Shaun Harker et al. “Discrete Morse theoretic algorithms for computing homology of complexes and maps”. In: *Found. Comput. Math.* 14.1 (2014), pp. 151–184. ISSN: 1615-3375,1615-3383. DOI: 10.1007/s10208-013-9145-0. URL: <https://doi.org/10.1007/s10208-013-9145-0>.
- [22] K. Hornik. “Approximation capabilities of multilayer feedforward networks”. In: *Neural Networks* 4 (2 1991), pp. 251–257. DOI: 10.1016/0893-6080(91)90009-T.
- [23] A. Ives et al. “High-amplitude fluctuations and alternative dynamical states of midges in Lake Myvatn”. In: *Nature* 452 (2008), pp. 84–87.
- [24] Tomasz Kaczynski, Konstantin Mischaikow, and Marian Mrozek. *Computational homology*. Vol. 157. Applied Mathematical Sciences. Springer-Verlag, New York, 2004,

- pp. xviii+480. ISBN: 0-387-40853-3. DOI: 10.1007/b97315. URL: <https://doi.org/10.1007/b97315>.
- [25] William D. Kalies, Konstantin Mischaikow, and Robert C. Vandervorst. “Lattice structures for Attractors II”. In: *Foundations of Computational Mathematics* 16.5 (Aug. 2015), pp. 1151–1191. DOI: 10.1007/s10208-015-9272-x.
- [26] William D. Kalies, Konstantin Mischaikow, and Robert C. A. M. Vandervorst. “Lattice structures for attractors III”. In: *Journal of Dynamics and Differential Equations* 34 (2021), pp. 1729–1768. DOI: 10.1007/s10884-021-10056-8.
- [27] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2017. arXiv: 1412.6980 [cs.LG]. URL: <https://arxiv.org/abs/1412.6980>.
- [28] Sotiris B Kotsiantis, Ioannis D Zaharakis, and Panayiotis E Pintelas. “Machine learning: a review of classification and combining techniques”. In: *Artificial Intelligence Review* 26 (2006), pp. 159–190.
- [29] Miroslav Kramár et al. “Analysis of Kolmogorov flow and Rayleigh–Bénard convection using persistent homology”. In: *Physica D: Nonlinear Phenomena* 334 (2016). Topology in Dynamics, Differential Equations, and Data, pp. 82–98. ISSN: 0167-2789. DOI: <http://dx.doi.org/10.1016/j.physd.2016.02.003>. URL: <http://www.sciencedirect.com/science/article/pii/S0167278916000270>.
- [30] Moshe Leshno et al. “Multilayer feedforward networks with a nonpolynomial activation function can approximate any function”. In: *Neural Networks* 6.6 (1993), pp. 861–867. DOI: 10.1016/S0893-6080(05)80131-5. URL: <https://www.sciencedirect.com/science/article/pii/S0893608005801315>.
- [31] D. Lin and B. Tang. “Latin hypercubes and space filling designs”. In: *Handbook of Design and Analysis of Experiments (1st ed.)* Chapman and Hall/CRC, 2015. Chap. 17.
- [32] Marissa Masden. *Algorithmic Determination of the Combinatorial Structure of the Linear Regions of ReLU Neural Networks*. 2022. arXiv: 2207.07696 [cs.LG]. URL: <https://arxiv.org/abs/2207.07696>.
- [33] Christopher McCord. “Mappings and homological properties in the Conley index theory”. In: *Ergodic Theory Dynam. Systems* 8\*. Charles Conley Memorial Issue (1988), pp. 175–198. ISSN: 0143-3857. DOI: 10.1017/S014338570000941X. URL: <https://doi.org/10.1017/S014338570000941X>.
- [34] Christopher McCord, Konstantin Mischaikow, and Marian Mrozek. “Zeta functions, periodic trajectories, and the Conley index”. In: *J. Differential Equations* 121.2 (1995), pp. 258–292. ISSN: 0022-0396. DOI: 10.1006/jdeq.1995.1129. URL: <https://doi.org/10.1006/jdeq.1995.1129>.
- [35] K. Mischaikow et al. “Construction of Symbolic Dynamics from Experimental Time Series”. In: *Phys. Rev. Lett.* 82 (6 Feb. 1999), pp. 1144–1147.
- [36] Konstantin Mischaikow and Marian Mrozek. “Chaos in the Lorenz equations: a computer-assisted proof”. In: *Bull. Amer. Math. Soc. (N.S.)* 32.1 (1995), pp. 66–72. ISSN: 0273-0979.
- [37] Fionn Murtagh and Pedro Contreras. “Algorithms for hierarchical clustering: an overview”. In: *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 2.1 (2012), pp. 86–97.
- [38] Frank Nielsen and Frank Nielsen. “Hierarchical clustering”. In: *Introduction to HPC with MPI for Data Science* (2016), pp. 195–211.
- [39] Leonardo Noriega. “Multilayer perceptron tutorial”. In: *School of Computing. Staffordshire University* 4.5 (2005), p. 444.
- [40] Maithra Raghu et al. *On the Expressive Power of Deep Neural Networks*. 2017. arXiv: 1606.05336 [stat.ML].
- [41] Xiaoli Ren et al. “Deep learning-based weather prediction: a survey”. In: *Big Data Research* 23 (2021), p. 100178.
- [42] Clark Robinson. *Dynamical Systems: Stability, Symbolic Dynamics, and Chaos*. CRC Press, 1998. ISBN: 9780849384950.

- [43] Martin G Schultz et al. “Can deep learning beat numerical weather prediction?” In: *Philosophical Transactions of the Royal Society A* 379.2194 (2021), p. 20200097.
- [44] Roman Srzednicki. “On rest points of dynamical systems”. In: *Fund. Math.* 126.1 (1985), pp. 69–81. ISSN: 0016-2736. DOI: 10.4064/fm-126-1-69-81. URL: <https://doi.org/10.4064/fm-126-1-69-81>.
- [45] Andrzej Szymczak. “The Conley index and symbolic dynamics”. In: *Topology* 35.2 (1996), pp. 287–299. ISSN: 0040-9383. DOI: 10.1016/0040-9383(95)00029-1. URL: [https://doi.org/10.1016/0040-9383\(95\)00029-1](https://doi.org/10.1016/0040-9383(95)00029-1).
- [46] Christos Thrampoulidis, Samet Oymak, and Mahdi Soltanolkotabi. “Theoretical insights into multiclass classification: A high-dimensional asymptotic view”. In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 8907–8920.
- [47] Allyn Treshansky and Robert M McGraw. “Overview of clustering algorithms”. In: *Enabling Technology for Simulation Science V* 4367 (2001), pp. 41–51.
- [48] Ewerton R. Vieira et al. “MORALS: Analysis of High-Dimensional Robot Controllers via Topological Tools in a Latent Space”. In: *2024 IEEE International Conference on Robotics and Automation (ICRA)*. 2024, pp. 27–33. DOI: 10.1109/ICRA57147.2024.10610383.

## 6. APPENDIX

**6.1. Parameters and Hyperparameters for neural networks.** In Section 3.3, we describe how the networks  $F_\theta$  are trained. For each considered example, Table 2 lists the network width ( $p$ ), the number of training points ( $\#\mathcal{I}$ ), the batchsize, number of epochs, patience parameter ( $\rho$ ) for early stopping, and loss reduction threshold ( $\beta$ ) for determining convergence.

Table of Training Parameters and Hyperparameters							
Example	dim( $X$ )	$p$	$\#\mathcal{I}$	batchsize	# epochs	$\rho$	$\beta$
Linear separatrix	2	2	$10^4$	$10^3$	$10^2$	10	$10^{-1}$
Radial bistable	2	4	$10^3$	$10^2$	$10^3$	$10^2$	$10^{-1}$
Radial tristable	2	10	$10^3$	$10^2$	$10^3$	$10^2$	$5 \cdot 10^{-1}$
Nonlinear separatrix	4	4	$10^4$	$10^3$	$10^2$	20	$10^{-1}$
Hill system with PO	3	15	$1.1 \cdot 10^4$	$10^3$	$10^3$	$10^2$	$5 \cdot 10^{-1}$
EMT Hill system	6	12	$10^5$	$10^4$	$2 \cdot 10^3$	$10^2$	$10^{-1}$
Ellipsoidal bistable	2	4	$10^4$	$10^3$	$10^3$	$10^2$	$10^{-1}$
Ellipsoidal bistable	3	6	$2 \cdot 10^4$	$2 \cdot 10^3$	$10^3$	$10^2$	$10^{-1}$
Ellipsoidal bistable	4	8	$8 \cdot 10^5$	$8 \cdot 10^4$	$2 \cdot 10^3$	$10^2$	$10^{-1}$
Ellipsoidal bistable	5	10	$8 \cdot 10^5$	$8 \cdot 10^4$	$2 \cdot 10^3$	$10^2$	$10^{-1}$

TABLE 2. Parameters and hyperparameters used in training realizations of  $F_\theta$  as described in Section 3.3.

**6.2. Conley indices of the attractors for the considered examples.** For the examples in this paper we know a priori the Conley indices of the attractors  $A_k$ . These are presented in Table 3. We compute the homology of each  $N_k$  (see (5)) to determine if we have identified the correct Conley indices. We similarly check for the correctness of the homology for the uncertain region  $U$ .

Table of Correct Betti Numbers ( $\beta_0, \beta_1, \dots$ )						
Example	dim( $X$ )	$U$	$N_0$	$N_1$	$N_2$	$N_3$
Linear separatrix	2	(1, 0)	(1, 0)	(1, 0)		
Radial bistable	2	(1, 1)	(1, 1)	(1, 0)		
Radial tristable	2	(2, 2)	(1, 1)	(1, 1)	(1, 0)	
Nonlinear separatrix	4	(1, 0, 0, 0)	(1, 0, 0, 0)	(1, 0, 0, 0)		
Hill system with PO	3	(1, 0, 0)	(1, 0, 0)	(1, 0, 0)	(1, 0, 0)	(1, 0, 0)
EMT Hill system	6	(1, 0, 0, 0, 0, 0)	(1, 0, 0, 0, 0, 0)	(1, 0, 0, 0, 0, 0)		
Ellipsoidal bistable	2	(1, 1)	(1, 1)	(1, 0)		
Ellipsoidal bistable	3	(1, 0, 1)	(1, 0, 1)	(1, 0, 0)		
Ellipsoidal bistable	4	(1, 0, 0, 1)	(1, 0, 0, 1)	(1, 0, 0, 0)		
Ellipsoidal bistable	5	(1, 0, 0, 0, 1)	(1, 0, 0, 0, 1)	(1, 0, 0, 0, 0)		

TABLE 3. The correct Betti numbers for the sets  $U$  and  $N_k$  for each example.

**6.3. Hill system with a periodic orbit.** We define a dynamical system on  $X = [0, 3.062] \times [0, 4.072] \times [0, 11.26362]$ . Let  $n = 10$ ,  $\gamma = [1, 1, 1]^T$ ,

$$L = \begin{bmatrix} 0.133 & 1.743 & 0 \\ 0.599 & 0.124 & 0.245 \\ 0.175 & 0 & 0.458 \end{bmatrix}, U = \begin{bmatrix} 0.578 & 3.299 & 0 \\ 1.994 & 0.725 & 5.205 \\ 1.737 & 0 & 2.164 \end{bmatrix}, \text{ and}$$

$$\Theta = \begin{bmatrix} 0.363 & 1.531 & 0 \\ 2.036 & 0.862 & 0.233 \\ 0.818 & 0 & 0.168 \end{bmatrix}.$$

Define

$$(10) \quad \begin{cases} h_1(x) = H^+(x_1, L_{11}, U_{11}, \Theta_{11}, n)H^-(x_2, L_{21}, U_{21}, \Theta_{21}, n)H^-(x_3, L_{31}, U_{31}, \Theta_{31}, n) \\ h_2(x) = H^+(x_2, L_{22}, U_{22}, \Theta_{22}, n)H^-(x_1, L_{12}, U_{12}, \Theta_{12}, n) \\ h_3(x) = H^+(x_3, L_{33}, U_{33}, \Theta_{33}, n)H^-(x_2, L_{23}, U_{23}, \Theta_{23}, n) \end{cases}$$

where

$$(11) \quad H^-(y, \ell, u, \theta) = \ell + (u - \ell) \frac{\theta^n}{y^n + \theta^n},$$

and

$$(12) \quad H^+(y, \ell, u, \theta) = \ell + (u - \ell) \frac{y^n}{y^n + \theta^n}.$$

Define the system  $\dot{x} = f(x)$  componentwise by

$$(13) \quad \dot{x}_i = -\gamma_i x_i + h_i(x), \quad i = 1, 2, 3.$$

**6.4. Six-dimensional EMT Hill system.** We now define a dynamical system on  $X = [0, 0.5] \times [0, 3.558] \times [0, 1.57345566] \times [0, 5.762] \times [0, 15.169196784] \times [0, 5.056847]$ . Let  $n = 10$ ,  $\gamma = [1, 1, 1, 1, 1, 1]^T$ ,

$$L = \begin{bmatrix} 0 & 0 & 0.318 & 0 & 0 & 0 \\ 0.148 & 0 & 0 & 0 & 0.610 & 0 \\ 0 & 1.210 & 0.510 & 0 & 1.054 & 0.993 \\ 0.150 & 0 & 0 & 0 & 0.6454 & 0 \\ 0 & 1.024 & 0 & 0.707 & 0 & 0.684 \\ 0 & 0 & 0.533 & 0 & 0 & 0 \end{bmatrix},$$

$$U = \begin{bmatrix} 0 & 0 & 1.316 & 0 & 0 & 0 \\ 0.520 & 0 & 0 & 0 & 2.037 & 0 \\ 0 & 1.442 & 0.965 & 0 & 2.056 & 3.917 \\ 0.670 & 0 & 0 & 0 & 3.622 & 0 \\ 0 & 2.113 & 0 & 3.720 & 0 & 1.291 \\ 0 & 0 & 1.239 & 0 & 0 & 0 \end{bmatrix},$$

and

$$\Theta = \begin{bmatrix} 0 & 0 & 0.250 & 0 & 0 & 0 \\ 1.312 & 0 & 0 & 0 & 1.779 & 0 \\ 0 & 0.668 & 0.206 & 0 & 0.478 & 0.166 \\ 1.333 & 0 & 0 & 0 & 2.881 & 0 \\ 0 & 5.698 & 0 & 1.676 & 0 & 0.950 \\ 0 & 0 & 0.705 & 0 & 0 & 0 \end{bmatrix}.$$

$$(14) \quad \begin{cases} h_1(x) = H^-(x_2, L_{21}, U_{21}, \Theta_{21}, n)H^-(x_4, L_{41}, U_{41}, \Theta_{41}, n) \\ h_2(x) = H^-(x_3, L_{32}, U_{32}, \Theta_{32}, n)H^-(x_5, L_{52}, U_{52}, \Theta_{52}, n) \\ h_3(x) = H^+(x_1, L_{13}, U_{13}, \Theta_{13}, n)H^-(x_3, L_{33}, U_{33}, \Theta_{33}, n)H^-(x_6, L_{63}, U_{63}, \Theta_{63}, n) \\ h_4(x) = H^-(x_5, L_{54}, U_{54}, \Theta_{54}, n) \\ h_5(x) = H^+(x_3, L_{35}, U_{35}, \Theta_{35}, n)H^-(x_2, L_{25}, U_{25}, \Theta_{25}, n)H^-(x_4, L_{45}, U_{45}, \Theta_{45}, n) \\ h_6(x) = H^-(x_3, L_{36}, U_{36}, \Theta_{36}, n)H^-(x_5, L_{56}, U_{56}, \Theta_{56}, n), \end{cases}$$

where  $H^\pm$  are defined by (11) and (12).

Define the system  $\dot{x} = f(x)$  componentwise by

$$(15) \quad \dot{x}_i = -\gamma_i x_i + h_i(x), \quad i = 1, 2, \dots, 6.$$

DEPARTMENT OF MATHEMATICS, RUTGERS UNIVERSITY  
*Email address:* gameiro@math.rutgers.edu

DEPARTMENT OF MATHEMATICS, RUTGERS UNIVERSITY  
*Email address:* brittany.gelb@rutgers.edu

DEPARTMENT OF MATHEMATICS AND STATISTICS, UNIVERSITY OF TOLEDO  
*Email address:* william.kalies@utoledo.edu

DEPARTMENT OF MATHEMATICS, UNIVERSITY OF OKLAHOMA  
*Email address:* miro@ou.edu

DEPARTMENT OF MATHEMATICS, RUTGERS UNIVERSITY  
*Email address:* mischaik@math.rutgers.edu

DEPARTMENT OF PHYSICS, FLORIDA ATLANTIC UNIVERSITY  
*Email address:* ptataciore2016@fau.edu