Journal of Machine Learning Research 23 (2022) 1-35 Submitted 12/20; Revised 7/22; Published 10/22

Linearization and Identification of Multiple-Attractor Dynamical Systems through Laplacian Eigenmaps

Bernardo Fichera BERNARDO.FICHERA@EPFL.CH
Learning Algorithms and Systems Laboratory (LASA)
École polytechnique fédérale de Lausanne
Lausanne, Switzerland

Aude Billard AUDE.BILLARD@EPFL.CH
Learning Algorithms and Systems Laboratory (LASA)
École polytechnique fédérale de Lausanne
Lausanne, Switzerland

Editor: Corinna Cortes

Abstract

Dynamical Systems (DS) are fundamental to the modeling and understanding time evolving phenomena, and have application in physics, biology and control. As determining an analytical description of the dynamics is often difficult, data-driven approaches are preferred for identifying and controlling nonlinear DS with multiple equilibrium points. Identification of such DS has been treated largely as a supervised learning problem. Instead, we focus on an unsupervised learning scenario where we know neither the number nor the type of dynamics. We propose a Graph-based spectral clustering method that takes advantage of a velocity-augmented kernel to connect data points belonging to the same dynamics, while preserving the natural temporal evolution. We study the eigenvectors and eigenvalues of the Graph Laplacian and show that they form a set of orthogonal embedding spaces, one for each sub-dynamics. We prove that there always exist a set of 2-dimensional embedding spaces in which the sub-dynamics are linear and n-dimensional embedding spaces where they are quasi-linear. We compare the clustering performance of our algorithm to Kernel K-Means, Spectral Clustering and Gaussian Mixtures and show that, even when these algorithms are provided with the correct number of sub-dynamics, they fail to cluster them correctly. We learn a diffeomorphism from the Laplacian embedding space to the original space and show that the Laplacian embedding leads to good reconstruction accuracy and a faster training time through an exponential decaying loss compared to the state-of-the-art diffeomorphism-based approaches.

Keywords: Dynamical System, Graph, Laplacian, Clustering, Learning

1. Introduction

Relying on the mathematical framework of differential equations, *Dynamical Systems* (DS) describe how a system evolves temporally and spatially. Their application span various fields, such as physics, biology, engineering, and economics. In recent years, DS have been successfully applied to model and control robots (e.g. Heinzmann and Zelinsky (2003); Corteville et al. (2007)).

Finding an analytical description of the dynamics is, however, often difficult. This led researchers to turn to data-driven identification of the dynamics using machine learning algorithms. Data is usually provided by an expert, within the generic framework of learning from demonstration (Billard et al. (2016)). Data can also be gathered from trial and error in a reinforcement learning framework (Wabersich and Zeilinger (2018)).

From a machine learning perspective, approximating a DS can be framed as a regression problem. One estimates a non-linear function $\mathbf{f} : \mathbb{R}^d \to \mathbb{R}^d$, mapping the d-dimensional input state $\mathbf{x}(t) \in \mathbb{R}^d$ to its time-derivative $\dot{\mathbf{x}}(t) \in \mathbb{R}^d$, such that:

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t)). \tag{1}$$

Training data consists of a set of trajectories, as examples of path integrals of the DS. These trajectories cover a limited portion of the state space. To ensure that the learned DS stably generalizes over regions not covered by the data represents one of the most challenging and active research area. One option to tackle this problem is to embed constraints explicitly in the algorithm so that the learned dynamics offers similar guarantees as those generated from control theory. One desirable property is stability at an equilibrium point, or *attractor*. If $\mathbf{x}^*$ is the attractor, we wish to guarantee that

$$\lim_{t\to\infty} \mathbf{x}(t) = \mathbf{x}^*, \quad \mathbf{f}(\mathbf{x}^*) = 0. \tag{2}$$

1.1 Learning Stable Dynamical Systems

In the Stable Estimator of Dynamical Systems (SEDS) proposed by Mohammad Khansari-Zadeh and Billard (2014), the density estimation with Gaussian Mixture Models is reformulated to enforce constraints on the parameters of the Gaussian Mixtures, by imposing conditions derived via Lyapunov's second method for stability. The DS is then estimated through Gaussian Mixture Regression (GMR). This approach was, however, limiting. Constraints from a quadratic Lyapunov function were conservative and led to a poor approximation of the flow when the dynamics was highly nonlinear.

Accuracy and stability turned out to be conflicting objectives whilst constraints were derived from a quadratic Lyapunov function, and this prevented modeling dynamics that are non-monotonic, namely temporarily moving away from the attractor. To address this issue, other works used more complex expression for the Lyapunov function (e.g. Mirrazavi Salehian (2018); Figueroa and Billard (2018)). An alternative to using Lyapunov stability is Contraction Theory (CT) introduced by Lohmiller and Slotine (1998). Stability under CT is less restrictive, as it follows a differential perspective and enforces solely that the flow contracts locally. Further, it does not require prior knowledge of the location of the attractor. Ravichandar et al. (2017) used CT to reformulate the SEDS constraints. Similarly Sindhwani et al. (2018) constraints a Support Vector regression problem with CT constraints. Although CT represents a promising approach, current works adopting it are limited to local stability guarantees making the approximated vector field unsuitable for regions with sparse or no data.

A third approach to tackle the problem of increasing accuracy while preserving stability is based on the idea of using latent representation to ease the stabilization of the DS. That is achieved via diffeomorphic mapping. One of the first attempts was offered in Neumann and

Steil (2015), which uses a diffeomorphic map to send a complex Lyapunov function, learned from the sampled trajectories, to a space where it appears quadratic. In this space, SEDS is applied, and the original vector field is recovered via the inverse diffeomorphic map. The two-step learning approach requires fitting both a Lyapunov function (or a diffeomorphism) and a DS from training data, increasing the number of tunable parameters and the overall learning time for non-convex optimization problems. Perrin and Schlehuber-Caissier (2016) follow a similar approach but apply the diffeomorphism directly to the DS using one single sampled trajectory. The diffeomorphism learning process is purely geometrical, namely only original and target points' position are considered. Reconstruction of the proper velocity profile is achieved via re-scaling the learned DS. The recent work proposed by Rana et al. (2020) follows a similar approach but introduces a formulation of the optimization framework that includes dynamics information (e.g., velocity) within the process, providing for a one-step learning algorithm.

All the methods reviewed above focus on single-attractor DS, and, except for works using stability constraints based on CT-derived conditions, they require prior knowledge of the location of the attractor. Assuming single dynamics and single attractor DS considerably constrains the applicability of these methods to learn uni-modal dynamics. Embedding multiple dynamics in a single control law based on DS increases the complexity of the dynamics that we can model. Next, we review recent efforts that have been dedicated to learning DS with multiple attractors.

1.2 Learning Dynamical Systems with Multiple Attractors

Learning DS with multiple-attractor can be achieved by explicitly partitioning the space to separate each dynamics and their respective attractors, as shown by Shukla and Billard (2012). This work requires knowing how many sub-dynamics exist in the system and the attractors' locations. Such knowledge may not always be easy to obtain.

Hence, if, for learning single-attractor DS, the main requirement is to know the location of the attractor, when learning multiple-attractor DS, correct labelling and classification of data points to distinguish the attractors and their associated dynamics is necessary. In realistic scenarios, proper segmentation and labeling might not be available either because data are generated by naive users or because the dataset is constructed by sampling demonstrations of complicated physical tasks (e.g., cleaning up a messy office). One cannot expect lay users to properly identify the number of sub-dynamics with the relative attractors location. Asking users to divide the task into sub-segment will be very cumbersome and prevent proper skill display and transfer; finally, even for expert users, it might often be difficult to classify the dynamics, especially when these require data that are not easy to interpret, such as force and haptic information.

In order to overcome this limitation, one option is to offer automatic segmentation and identification of the dynamics. Such an approach was followed by Medina and Billard (2017), using Hidden Markov Models to extract automatically the attractors' positions of multiple-attractor DS. The algorithm assumes that such multiple-attractor DS can be thought of as a long sequence of multiple sub-dynamics to be performed one after the other. Such approach implicitly assumes time labeling of the points and can handle only multiple-attractor DS considered as a long sequence of dynamics. Manschitz et al. (2018) presented another interesting application of multiple-attractor DS. The DS-based control law is modeled as a weighted combination of linear systems, each of which is stable with respect to an attractor. The location of the attractors along with the parameters of each linear sub-dynamics is automatically derived via optimization problem. However the minimum number of attractors necessary to solve a specific task is a user-defined hyper-parameter that has to be known a priori.

To avoid providing prior information on both the number of dynamics and attractors, we see a need for a fully *unsupervised learning* approach to the identification of multiple-attractor DS. In this light, we offer an algorithm whose goal is to: (a) cluster the sub-dynamics within a certain data set, (b) find underlying interesting structure of the data that would allow to locate the attractors and ease the task of learning stable vector field. To do so, we seek to exploit structure discovery techniques to identify the number of sub-dynamics automatically.

1.3 Manifold Learning for Latent Embedding Spaces of Dynamical Systems

Manifold learning techniques, such as *Laplacian Eigenmaps* (Belkin and Niyogi (2003)) and *Isometric Mapping* (Tenenbaum et al. (2000)), are particularly relevant to our problem. These techniques can generate Euclidean spaces recovering the intrinsic geometry of a certain manifold (e.g., Tenenbaum et al. (2000)). As in Kernel PCA (Principal Component Analysis) (Schölkopf and Smola (2002)), these methods are based on eigenvalue decomposition of a matrix. The matrix embeds information about the feature of the data, as encapsulated by the kernel. Depending on the kernel used, different features can be extracted by the manifold.

We find applications of manifold learning techniques for DS in biology for analyzing emergent dynamics behaviors in data measured from bioelectric signals (Erem et al. (2016)), or for person identification from ECG (Sulam et al. (2017)); in control theory for data-driven time series analysis (Shnitzer et al. (2017)); in finance for describing the characteristics of financial system (Huang et al. (2018)); in chemistry for stochastic model of cellular chemotaxis (Dsilva et al. (2018)).

In the works reviewed above, the primary goal is to use manifold learning to reduce the dimensionality of the data and simplify the estimation of the DS. The scope of our work is different. We do not aim at reducing the dimensionality but rather at finding an embedding that can simplify the control of DS by linearizing a non-linear dynamics that would otherwise be difficult to stabilize.

The success of manifold learning depends heavily on the choice of kernel. We define a velocity-augmented kernel to extract the temporal evolution of data points. We generate the desired graph structure with this kernel and compute the associated Laplacian. We study the embedding spaces generated by the eigendecomposition of such a graph-based Laplacian matrix. We show that an analysis of the eigenvalues enables us to determine the number of underlying dynamics and to identify a set of eigenvectors that forms an embedding space in which the DS is linearized.

We validate our technique for sub-dynamics clustering and stable equilibria localization of multiple-attractor DS using theoretical and noisy instances of DS. We compare our algorithm to Kernel K-Means, Spectral Clustering, and Gaussian Mixtures. We showcase that even when these algorithms are provided with the correct number of sub-dynamics, they fail to cluster them correctly.

2. Problem Formulation

Let $\mathbf{x} \in \mathbb{R}^d$ be the state of a DS. Its temporal evolution is governed by the non-linear function $\mathbf{f}(\mathbf{x}(t))$:

$$\mathbf{f} : \mathbb{R}^d \to \mathbb{R}^d, \quad \dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t)). \tag{3}$$

Assume that $\mathbf{f}$ is composed of Q sub-dynamics, each asymptotically stable at one equilibrium point, the attractor.

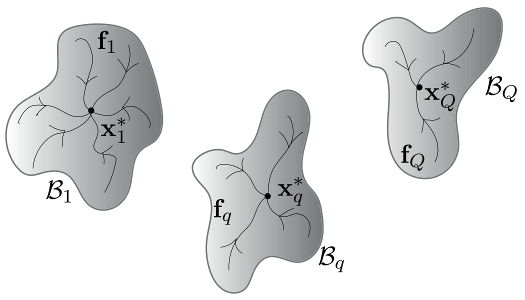


Figure 1: Schematic representation of a multiple-attractor DS. Each sub-dynamics $\mathbf{f}_q$ takes within a subspace $\mathcal{B}_q$ and converges towards the attractor $\mathbf{x}_q^*$.

Let the q-th sub-dynamics be

$$\dot{\mathbf{x}}(t) = \mathbf{f}_q(\mathbf{x}(t)) \quad \forall q \in [1, Q] \subset \mathbb{N}. \tag{4}$$

Each *sub-dynamics*, $\mathbf{f}_q(\mathbf{x})$, is Lyapunov stable and there exists $\delta > 0$ such that if $\left\|\mathbf{x}(0) - \mathbf{x}_q^*\right\| < \delta$, then

$$\lim_{t\to\infty} \left\|\mathbf{x}(t) - \mathbf{x}_q^*\right\| = 0, \tag{5}$$

where $\mathbf{x}_q^*$ is the attractor of the sub-dynamics $\mathbf{f}_q(\mathbf{x})$, Fig. 1.

We know neither the number of sub-dynamics Q, the shape of the dynamics $\mathbf{f}_q$, nor the number and location of the attractors. All we have at our disposal is a finite set of observations, samples of trajectories from the DS. These are constituted by *unlabeled* position-velocity pairs $\mathcal{V} = \{\nu_i = (\mathbf{x}_i, \dot{\mathbf{x}}_i), i = 1, \dots, M\}$ sampled from $\mathbf{f}(\mathbf{x})$ at a constant frequency, $f_{sampling}$. Among these trajectories, a subset belongs to some of the sub-dynamics, but we know neither to which dynamics they belong nor how many trajectories belong to the same dynamics. All sampled trajectories end at the attractor of the associated sub-dynamics.

This paper presents a method by which we can a) determine the number Q of sub-dynamics present in the dataset, b) identify to which sub-dynamics each data point belongs and the attractor of the corresponding sub-dynamics and c) project each dynamics into a separate subspace in which the dynamics is linear. The method is based on a representation of the data through a graph and exploits properties from the eigendecomposition of the graph-based Laplacian matrix, as we present next.

3. Graph Embedding and Linearization of Dynamical Systems

Consider a symmetric weighted graph $G(\mathcal{V}, \mathcal{E})$. The nodes $\nu_i \in \mathcal{V}$ represent the data points $\nu_i = \{\mathbf{x}_i, \dot{\mathbf{x}}_i\}$, $i = \{1, \ldots, M\}$ sampled from our DS, where $M = \sum_{q=1}^{Q} p_q$ with p_q being the number of data points sampled from each of the sub-dynamics and Q being the number of sub-dynamics. $\mathcal{E}$ is the set of edges $e_{ij} \in \mathcal{E}$ connecting nodes ν_i and ν_j. For a generic DS containing Q sub-dynamics, the graph G is the result of the union of Q sub-graphs $\mathcal{F}$ such that $G = \mathcal{F}_1 \cup \mathcal{F}_2 \cup \cdots \cup \mathcal{F}_Q$ and $\mathcal{F}_1 \cap \mathcal{F}_2 \cap \cdots \cap \mathcal{F}_Q = \emptyset$. Each sub-graph $\mathcal{F}$ is given by the composition of K *path graphs*, equal to the number of sampled trajectories. We order the nodes (vertices) of $\mathcal{F}$ monotonically as $\{\nu_1^k, \nu_2^k, \ldots, \nu_{p_k}^k\}$, $k = 1, \ldots, K$, such that each $\nu_{p_k}^k$ is the last node of the k-th path graph and $p_k > 1$. The edges are $e_{ij}^k = \{\nu_i^k, \nu_{i+1}^k\}$. The K nodes $\{\nu_{p_1}^1, \ldots, \nu_{p_K}^K\}$ form a *cyclic graph* (or simple circuit). Each node belonging to the cyclic path has degree 3. All other nodes along each path graph have degree 1 or 2. The nodes are numbered as

$$\{\nu_1^1, \ldots \nu_{p_1}^1, \nu_1^2, \ldots, \nu_{p_2}^2, \ldots, \nu_1^K, \ldots, \nu_{p_K}^K\} \sim \{1, 2, \ldots, M\}$$

With reference to Fig. 2, $\{\nu_1^k, \ldots, \nu_{p_k}^k\}$ represents the nodes of the k-th path graph where p_k is the number of samples within the k-th path graph.

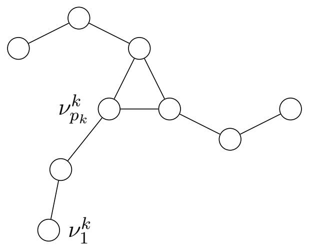


Figure 2: Each sub-dynamics is embedded in a graph where each trajectory forms a path connected by a set of termination nodes that form a cyclic path around the attractor. ν_1^k $\nu_{p_k}^k$ are the first and last of the k-th path graphs, respectively. p_k is the number of nodes with the k-th path graph.

The Graph Laplacian matrix, $L(G) = D(G) - A(G)$, where $A(G)$ is the adjacency matrix

$$A_{ij} = \begin{cases} 1, & \text{if } e_{ij} \in \mathcal{E} \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

and $D(G)$ is a diagonal matrix composed of the sum of the rows of $A(G)$

$$D_{ii} = \sum_j A_{ij}. \tag{7}$$

$L(G)$ is positive semi-definite and, hence, admits an eigendecomposition. The eigenvalues of the $L(G)$ are non-negative and we have at least one eigenvalue equal to zero. The multiplicity of the eigenvalue zero allows to determine the number of sub-dynamics embedded in the graph.

Proposition 1 *The eigendecomposition of the Laplacian generates a set of M positive eigenvalues $\Lambda = \{0 = \lambda_0^1 = \lambda_0^2 = \cdots = \lambda_0^Q < \lambda_1 \leq \cdots \leq \lambda_{M-Q}\}$. The multiplicity of the eigenvalue 0 is equal to the number of sub-dynamics Q.*

Proof The multiplicity of the eigenvalue zero of the Laplacian matrix is equal to the number of connected components, hence, by construction of G is equal to the number of sub-dynamics Q. ■

As the graph G is composed of a set of disconnected components, the eigendecomposition of $L(G)$ can be performed by blocks and we can associate a subset of eigenvectors to each of the Q blocks. We show in the next subsection that each set of eigenvectors of $L(G)$ generates Q separate sub-spaces in which each sub-dynamics is linearized.

Observe first that, in each of the graph connected component, the nodes are connected in such a way that each trajectory forms a path in the graph. With K trajectories for each sub-dynamics, we have K paths. Each trajectory is connected to another trajectory through the last node of each path graph.

3.1 Determining the Eigenvectors that Generate a Linear Embedding

We are now ready to present the main results of this paper, namely that a well-chosen set of eigenvectors of the graph Laplacian generates a linear embedding of the sub-dynamics.

We start by showing that, for each path in the graph, the corresponding entries in the eigenvectors follow a recursive law.

Lemma 2 *Consider K different path graphs that form a graph G with one connected component. Let $\mathbf{u}$ be an eigenvector of the Graph Laplacian $L(G)$. The elements of $\mathbf{u}$ entail K sets of scalars $\{u_1^k, \ldots, u_{p_k}^k\}$, corresponding to the nodes within the k-th path graph. Each set follows the recursive relation:*

$$\begin{aligned} &u_1^k \in \mathbb{R}, \\ &u_2^k = (1-\lambda)u_1^k, \\ &u_n^k = (2-\lambda)u_{n-1}^k - u_{n-2}^k, \quad \text{for}\, n = 3, \ldots, p_k. \end{aligned} \tag{8}$$

Proof See Appendix A. ■

Lemma 3 *In each of the K sets of elements in $\mathbf{u}$ denoted as $\{u_1^k, \ldots, u_{p_k}^k\}$, the recursive relation in Eq. 8 corresponds to a combination of Chebyshev polynomials T and V, of first and second kind, given by:*

$$u_n^k = u_1^k \left[T_n(\lambda) - \frac{\lambda}{2} V_{n-1}(\lambda) \right] \quad \textit{for } n \geq 1. \tag{9}$$

Let $\theta := \cos^{-1}(1 - \frac{\lambda}{2})$. Eq. 9 can be expressed as the following combination of trigonometric functions:

$$u_n^k(\lambda, \theta(\lambda)) = u_1^k \left[\cos((n-1)\theta) - \frac{\lambda}{2} \frac{\sin((n-1)\theta)}{\sin(\theta)} \right]. \tag{10}$$

Proof See Appendix B. ■

Next, we show that, when we select a specific set of eigenvectors, the coordinates of the path graphs in these eigenvectors either grow or decrease monotonically. This a key property to prove the linearity of the embedding.

Proposition 4 *If each of the path graphs have at least 3 nodes, $p_k \geq 3\ \forall k$, the coordinates of a path graph $\{u_1^k, \dots, u_{p_k}^k\}$ in the eigenvectors $\mathbf{u}$ with corresponding eigenvalues $0 < \lambda \leq 2\left[1 - cos\left(\frac{\pi}{p_k - \frac{1}{2}}\right)\right]$, either increase or decrease monotonically within each path graph, s.t.:*

$$u_n^k \geq (\leq) u_{n+1}^k \quad \textit{for } n = 1, \dots, p_k, \quad \textit{and } k \in \{1, \dots, K\} \tag{11}$$

Proof See Appendix C. ■

Next, we prove that, when all the path graphs have the same length, i.e $p_k = N$, $\forall k$, there always exist, at least, one pair of eigenvectors with properties of Prop. 4, by showing that there exists an eigenvalue $\lambda \leq 2\left[1 - \cos\left(\frac{\pi}{p_k - \frac{1}{2}}\right)\right]$ with algebraic multiplicity 2.

The proof is done in two steps: first, we show that the spectrum of $L(G)$ presents repeated eigenvalues; second, we show that there is at least one repeated eigenvalue upper bounded by a value inferior to the monotonicity constraint introduced in Prop. 4.

Following Gupta et al. (2022), observe that the graph Laplacian, analyzed in this work, has the following structure:

$$L(G) = 2I - J(G), \tag{12}$$

where J has a block circulant matrix structure of the following type

$$J(G) = \operatorname{circ}(B_0, B_1, \underbrace{\mathbf{0}^{(N\times N)}}_{K-3 \text{ times}}, B_1), \tag{13}$$

with K the number of path graphs and N the number of nodes in each path graph. The matrices $B_0 \in \mathbb{R}^{N\times N}$ is a tri-diagonal matrix of the form

$$B_0 = \begin{bmatrix} 1 & 1 & & & \\ 1 & 0 & \ddots & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & 0 & 1 \\ & & & 1 & -1 \end{bmatrix} \tag{14}$$

and $B_1 \in \mathbb{R}^{N\times N}$ is defined as

$$B_1 = \begin{bmatrix} \mathbf{0}^{(N-1\times N-1)} & 0 \\ 0 & 1 \end{bmatrix}. \tag{15}$$

We repeat results from Gupta et al. (2022) in Proposition 5 and 6:

Proposition 5 *If the number of paths K is even, $\frac{K}{2} - 1$ eigenvalues of $L(G)$ have multiplicity 2 and for K odd, $\frac{K-1}{2}$ eigenvalues of $L(G)$ have multiplicity 2.*

Proof See Appendix D. ■

Proposition 6 *The second smallest eigenvalue of the Graph Laplacian $L(G)$ with algebraic multiplicity 2 is denoted by $\lambda_{min}(L)$ and is bounded above and below as follows:*

$$1 - 2\cos\left(\frac{\pi}{N}\right) < \lambda_{min}(L) \leq 2\left(1 - \cos\left(\frac{\pi}{N - \frac{1}{2}}\right)\right) \tag{16}$$

where N is the number of nodes in each of the K paths.

Proof See Appendix E. ■

We have shown that there exist at least a pair of eigenvectors with same eigenvalues in which the corresponding entries of our path graph grow or decrease monotonically.

Next, we show that the DS is linear in the embedding formed by these two eigenvectors.

Corollary 7 *For each eigenvector u with associated eigenvalue $0 < \lambda < 1$, the rate of change $r(u_n) = u_n - u_{n-1}$ along the entries $\{u_1, \ldots, u_{p_k}\}$, for each path $k = 1, \ldots, K$ decreases monotonically according to:*

$$\dot{r}(u) = -\lambda u. \tag{17}$$

Proof $r(u_{n+1}) = u_n - u_{n-1} - \lambda u_n = r(u_n) - \lambda u_n$. ■

Theorem 8 *If the graph Laplacian $L(G)$ admits a set of eigenvectors $\mathbf{u}$, with same eigenvalue, the K paths of the graph expressed in the domain spanned by $\mathbf{u}$ form K lines.*

Proof Consider a group of three consecutive nodes $\{\nu_{n-1}, \nu_n, \nu_{n+1}\}$ of one of the paths of the graph. The coordinate of these points when projected in the space spanned by two eigenvectors u^z and u^k are $\{u^{z,k}_{n-1}, u^{z,k}_n, u^{z,k}_{n+1}\}$. We show that the three points are on the same line, as illustrated in Fig.3. We prove

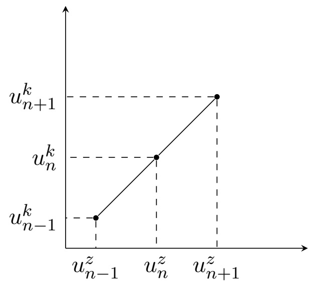


Figure 3: Coordinates of three points on two eigenvectors with constant rate of change, resulting in a linear path.

that these coordinates grow at the same rate for all the eigenvectors that have equal eigenvalues $\lambda < 1$. Hence the spacing across each group of coordinates on each axis is the same, as illustrated in Fig.3.

In the following, we drop the upper index referring to the specific eigenvector. For any eigenvector with eigenvalue $0 < \lambda < 1$, using Eq. 33, we have

$$u_{n+1} = \gamma u_n - u_{n-1}, \tag{18}$$

with $\gamma = 2-\lambda, \gamma > 1$. Furthermore, for the first two points on the path graph, we have, from Eq. 32: $u_2 = \delta u_1$, with $\delta = 1 - \lambda, \delta > 0$. Combining these two equations, the third point can be expressed as a function of the first coordinate u_1 only: $u_3 = \gamma(\delta u_1) - u_1 = (\gamma\delta - 1)u_1$. Similarly the fourth point can be expressed solely as a function of the first coordinate u_1. We have: $u_4 = \gamma u_3 - u_2 = \gamma(\gamma\delta - 1)u_1 - \delta u_1) = (\gamma(\gamma\delta - 1) - \delta)u_1$. The same reasoning holds for all points and hence by induction, we have:

$$u_n = P_n(\lambda)u_1. \tag{19}$$

$P_n(\lambda)$ is a polynomial of order $n-2$ that depends only on the eigenvalue λ. If the eigenvectors considered have same eigenvalue λ, the coordinates along each axis grow with the same series of polynomial $P_n(\lambda)$. Hence, all points are located on a line. For a pair of eigenvectors z, k, the line has slope $\frac{u_1^k}{u_1^z}$. ■

Note that, while Thm. 8 holds for all the eigenvectors in the spectrum of $L(G)$, the original ordering of the nodes along each path graph is preserved only when $\lambda < 2\left[1 - \cos\left(\frac{\pi}{p_k - \frac{1}{2}}\right)\right]$. For the eigenvectors with larger λ, the coordinates of the nodes change sign within the path graph. Hence, while the path may still appear linear in the embedding, the order of the nodes will not be preserved anymore.

3.2 Creating a Linear Embedding

We summarize how the above theoretical results allow us to find embedding spaces so that we can separate each of the Q sub-dynamics and that they are linear in each embedding.

First, let us recall that, since the matrix $L(G)$ can be decomposed by block with each block representing data points that belong to each sub-dynamics $\mathbf{f}_q$, we can generate a set of Q separate embedding spaces by using only the subset of eigenvectors associated to each sub-block. By orthogonality of the eigenvectors, all other sub-dynamics $\mathbf{f}_k$, $k \neq q$ will project to the origin in the embedding space of the q-th dynamics. We now need to determine the existence of such embedding spaces.

Given a K-paths graph, from Prop. 5, for $N \geq 3$ and $K \geq 3$, there is at least one pair of eigenvector with eigenvalue $0 < \lambda \leq 2\left[1 - \cos\left(\frac{\pi}{N - \frac{1}{2}}\right)\right]$. It follows that if the Q dynamics of the graph have at least $N \geq 3$ nodes and $K \geq 3$, we have $2Q$ eigenvectors with properties of Prop. 4 and hence Q embedding spaces.

The properties stated in the previous propositions are illustrated in Fig. 4a. We see a three-path graph connected by a cycle path across nodes $\{5, 10, 15\}$.The entries of the second and third eigenvectors (we exclude the first eigenvector, as it corresponds to the eigenvalue zero) are displayed in Fig. 4c. To ease the interpretation, we enumerate the data points in increasing order as we move from the start to the end of each path. We plot the entries of the eigenvectors against the point label. Observe that, within each path graph, the corresponding entries on the eigenvectors are either strictly increasing or decreasing. Further, observe that the spectrum of the Laplacian, see Fig. 4b, entails a pair of eigenvalues with algebraic multiplicity equal to 2. The embedding space reconstructed using the eigenvectors

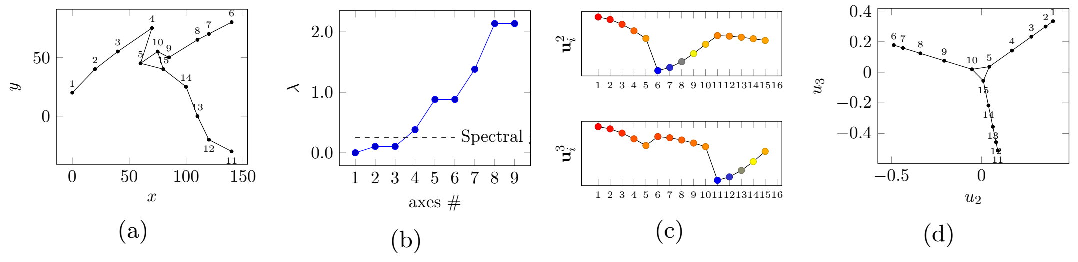


Figure 4: (a) Toy graph composed by 3 path graphs connected each other with a cycle graph passing through nodes $\{5, 10, 15\}$; (b) Spectrum of the first 9 eigenvalues of the Laplacian applied to graph in Fig. 4a; (c) Entries of the first 2 components $\{\mathbf{u}^2, \mathbf{u}^3\}$; (d) 2D embedding space reconstructed using components $\{\mathbf{u}^2, \mathbf{u}^3\}$.

$\{\mathbf{u}^2, \mathbf{u}^3\}$ corresponding to these two eigenvalues is shown in Fig. 4d. Observe that the structure now entails 3 paths all of which are linear. The original non-linear structure is hence linear in the embedding space.

In practice, we observe that several pairs of eigenvectors provide these embedding spaces. We also observe that the third and above eigenvalues are numerically close to the pair of smallest eigenvalues, and hence allow quasi-linear embedding spaces larger than 2 dimensions and up to $K-1$. This follows by observing that the polynomial $P_n(\lambda)$ in Eq. 19 is continuous and continuously differentiable up to $N-1$.

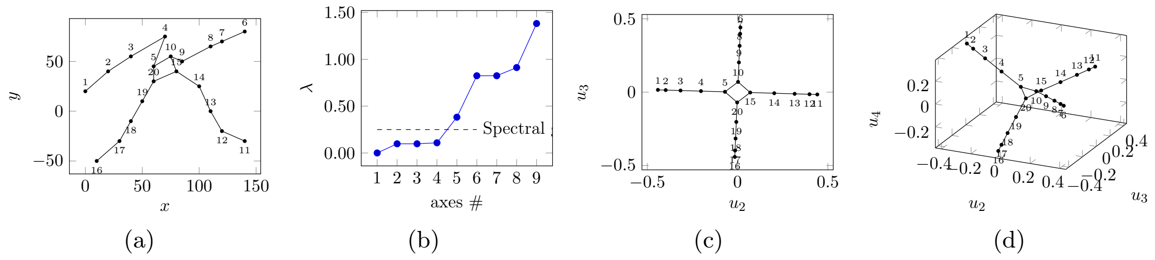
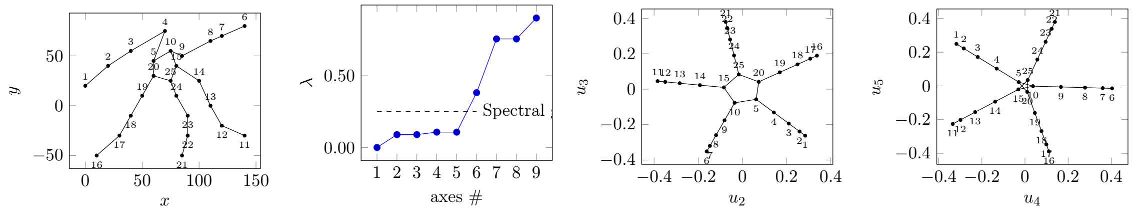


Figure 5: (a) Toy graph composed by 4 path graphs connected each other with a cycle graph passing through nodes $\{5, 10, 15, 20\}$; (b) Spectrum of the first 9 eigenvalues of the Laplacian applied to graph in Fig. 5a; (c) 2D embedding space reconstructed using components $\{\mathbf{u}^1, \mathbf{u}^2\}$; (d) 3D embedding space reconstructed using components $\{\mathbf{u}^1, \mathbf{u}^2, \mathbf{u}^3\}$.

We illustrate this property in two examples using 4-path and 5-path graphs, respectively, see Fig. 5 and 6. As in our previous example with a 3-path graph, the path graph are connected through one cycle graph. For the 4-path graph, we see that the spectrum of the Laplacian entails two eigenvalues of equal magnitude. The third eigenvalue has magnitude comparable but larger than the previous two, Fig. 5b. The graph can be linearized using the two eigenvectors with equal eigenvalues, see Fig. 5c. Quasi-linearization in the $3D$ embedding space can be achieved by using as coordinates the first three components, see Fig. 5d.

Using the 5-path graphs, we obtain two pairs of eigenvalues with equal magnitude, visible in the spectrum, Fig. 6b. We can hence reconstruct two distinct $2D$ embedding spaces where the DS is perfectly linear. While the embedding space constructed using eigenvectors



(a) (b) (c) (d)

Figure 6: (a) Toy graph composed by 5 path graphs connected each other with a cyclic graph passing through nodes $\{5, 10, 15, 20, 25\}$; (b) Spectrum of the first 9 eigenvalues of the Laplacian applied to graph in Fig. 6a; (c) 2D embedding space reconstructed using components $\{\mathbf{u}^1, \mathbf{u}^2\}$; (d) 2D embedding space reconstructed using components $\{\mathbf{u}^3, \mathbf{u}^4\}$.

$\{\mathbf{u}^2, \mathbf{u}^3\}$ preserves the radial ordering of the path graphs, Fig. 6c, the embedding constructed with components $\{\mathbf{u}^4, \mathbf{u}^5\}$ does not, as shown Fig. 6d.

4. Generating the Graph with Real Data

To exploit the results stated in the previous section, we must first embed the data in a graph. To achieve this, we propose a kernel that takes advantage of positions and velocities.

4.1 Velocity-Augmented Kernel

The kernel provides a distance measure across nodes ν_i, ν_j:

$$k(\nu_i, \nu_j) = \exp\left(-\tfrac{1}{2\sigma^2}\left(\underbrace{\|\mathbf{x}_i - \mathbf{x}_j\|^2}_{\text{locality}} + \overbrace{\gamma\left(|g(\mathbf{x}_i, \mathbf{x}_j, \dot{\mathbf{x}}_i)|\right)^2 + \gamma\left(|g(\mathbf{x}_j, \mathbf{x}_i, \dot{\mathbf{x}}_j)|\right)^2}^{\text{directionality}}\right)\right), \tag{20}$$

where $\dot{\mathbf{x}}_i = \mathbf{f}(\mathbf{x}_i)$ is the DS map that relates positions to velocities.

We introduce g a function to measure the angular distance across the velocity vectors for a pair of data points $\mathbf{x}_i$ and $\mathbf{x}_j$. We relate the distance vector $\mathbf{x}_j - \mathbf{x}_i$ and the velocity vector $\dot{\mathbf{x}}_i$ through a cosine kernel

$$g(\mathbf{x}_i, \mathbf{x}_j, \dot{\mathbf{x}}_i) = \underbrace{\delta_{\sigma_f}\left(\|\dot{\mathbf{x}}_i\|\|\dot{\mathbf{x}}_j\|\right)}_{\text{Filter}} + \underbrace{\left\langle \frac{\mathbf{x}_j - \mathbf{x}_i}{\|\mathbf{x}_j - \mathbf{x}_i\|}, \frac{\dot{\mathbf{x}}_i}{\|\dot{\mathbf{x}}_i\|} \right\rangle}_{\text{Cosine similarity kernel}}, \tag{21}$$

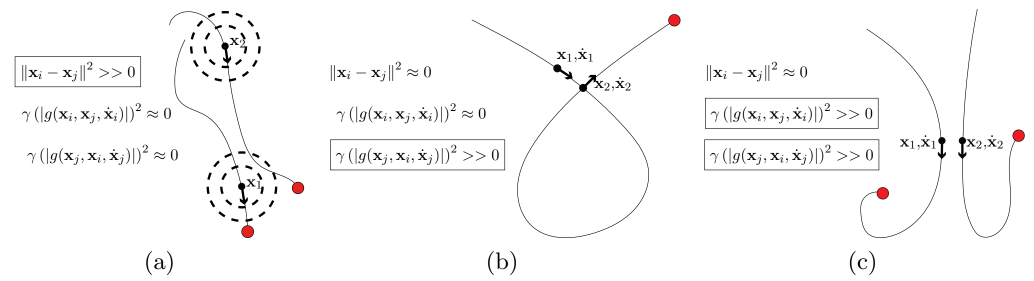
where δ_{σ_f} is a normal Gaussian distribution $\delta_{\sigma_f}(x) = \exp\left(-\frac{x^2}{2\sigma_f^2}\right)$ centered in $\|\dot{\mathbf{x}}\| = 0$, with σ_f set approximately to 0. The presence of the filtering part in Eq. 21 is necessary to withhold the "directionality" penalization among zero velocity points potentially close to an attractor. The filter δ_{σ_f} takes care of this situation outputting 1 whenever the velocity is zero. The parameter σ_f can be regulated to take into account known noise, e.g., when one knows that the velocity at an attractor point is not zero for numerical reasons.

The linear function $\gamma : \mathbb{R} \to \mathbb{R}$ in Eq. 20 is defined as

$$\gamma(g) = \frac{3}{2}\theta_r\,(1 - g)\,\sigma. \tag{22}$$

The co-domain the function g, in Eq. 21, is in between -1 and 1. The function γ, in Eq. 22, maps the part of the co-domain of g in between $cos(\theta_r)$ and 1 to the range between 3σ and 0. Consequently, the range of values of g in between -1 and $cos(\theta_r)$ will be mapped to values greater than 3σ. When the angle between two velocity vectors is θ_r the linear map yields a value equal to 3σ causing a penalization of the weight about 99%. When the output of the cosine is proximal to 1, due to the almost complete co-linearity, the linear function will yield 0 causing no penalization over the weight produced by the standard RBF (Radial Basis Functions) kernel. The reference angle, θ_r, is a parameter that can be set depending on the sampling frequency. High frequencies allow for a more strict ($\theta_r \approx 0$) selection of such angle and vice-versa.

In order to understand the kernel better, we illustrate the effect of each term in three scenarios depicted in Fig 7. The directed kernel encompasses two components: "locality" and "directionality". The locality term gives a measure of the spatial distance between pairs



$\boxed{\|\mathbf{x}_i - \mathbf{x}_j\|^2 >> 0}$
$\gamma\left(|g(\mathbf{x}_i, \mathbf{x}_j, \dot{\mathbf{x}}_i)|\right)^2 \approx 0$
$\gamma\left(|g(\mathbf{x}_j, \mathbf{x}_i, \dot{\mathbf{x}}_j)|\right)^2 \approx 0$
$\mathbf{x}_2$
$\mathbf{x}_1$
(a)
$\|\mathbf{x}_i - \mathbf{x}_j\|^2 \approx 0$
$\gamma\left(|g(\mathbf{x}_i, \mathbf{x}_j, \dot{\mathbf{x}}_i)|\right)^2 \approx 0$
$\boxed{\gamma\left(|g(\mathbf{x}_j, \mathbf{x}_i, \dot{\mathbf{x}}_j)|\right)^2 >> 0}$
$\mathbf{x}_1,\dot{\mathbf{x}}_1$
$\mathbf{x}_2,\dot{\mathbf{x}}_2$
(b)
$\|\mathbf{x}_i - \mathbf{x}_j\|^2 \approx 0$
$\boxed{\gamma\left(|g(\mathbf{x}_i, \mathbf{x}_j, \dot{\mathbf{x}}_i)|\right)^2 >> 0}$
$\boxed{\gamma\left(|g(\mathbf{x}_j, \mathbf{x}_i, \dot{\mathbf{x}}_j)|\right)^2 >> 0}$
$\mathbf{x}_1,\dot{\mathbf{x}}_1$
$\mathbf{x}_2,\dot{\mathbf{x}}_2$
(c)

Figure 7: Illustration of the effect of the spatial distribution of points and velocity vectors on the kernel Eq. 20: (a) the first term in the exponential generates low values for points that are far apart; (b) the third term in the exponential generates low values when consecutive velocity vectors are not aligned; (c) the second and third term in the exponential generate low values whenever the distance vector connecting two points is not aligned with the velocity vector of one of them.

of points based on the Euclidean distance, similar to the standard RBF kernel. The two directionality terms measure the co-linearity of the velocity vectors. Two points far apart with co-linear velocity vectors or two points close with distinct velocity vector will have a small value on the kernel and hence a loose connectivity in the similarity matrix used to generate the graph. Sole points that are both close and with co-directed velocity will reach the maximum kernel value, 1.

Points $\mathbf{x}_1$ and $\mathbf{x}_2$ in Fig. 7a will have negligible connection weights due to the large distance between them even though the "directionality" terms would yield values closed to 1 (about 0 after mapping γ) since $\mathbf{x}_2 - \mathbf{x}_1 \parallel \dot{\mathbf{x}}_1$ and $\mathbf{x}_1 - \mathbf{x}_2 \parallel \dot{\mathbf{x}}_2$. The second example in Fig. 7b considers the possibility of having intersecting trajectories perhaps due to second order dynamics or imprecise user demonstration. Since the figure might be misleading we clarify that point $\mathbf{x}_1$ belongs to the beginning of the trajectory while point $\mathbf{x}_2$ lies at the end, close to the attractor. In this situation we would like to "separate" the two points since they are not close from a dynamical perspective. Due to the proximity of the points, the "Locality" part contributes to generate a strong connection between the two points. The second component of the "directionality" part will take care of drastically decreasing the weight connection. Indeed, if the first term yields a high value, $\mathbf{x}_2 - \mathbf{x}_1 \parallel \dot{\mathbf{x}}_1$, the second term will be decisive in cutting of the connection between the points since $\mathbf{x}_1 - \mathbf{x}_2 \approx \dot{\mathbf{x}}_1 \perp \dot{\mathbf{x}}_2$. As last scenario in Fig. 7c, we consider the case of two proximal trajectories belonging to different sub-dynamics. As in the previous example the "Locality" cannot account for this situation; the "directionality" part will take care of decreasing the weight connection given that $\mathbf{x}_2 - \mathbf{x}_1 \perp \dot{\mathbf{x}}_1$ and $\mathbf{x}_1 - \mathbf{x}_2 \perp \dot{\mathbf{x}}_2$.

Fig. 8a shows streamlines of an expanding vector field $\dot{\mathbf{x}} = \mathbf{x}$ and a reference point with its velocity vector. In Fig. 8b it is possible to see how the proposed kernel generates, with

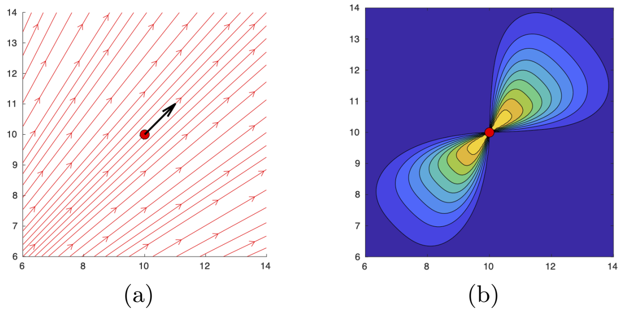


Figure 8: (a) Background vector field and reference evaluation point/velocity; (b) Contour of the velocity-augmented kernel.

respect to the considered vector field, high value regions symmetrically placed with respect to the plane orthogonal to reference velocity in the proximity of the evaluation point.

4.2 Selecting the Hyperparameters

The choice of the hyperparameter σ is important as it modulates the granularity of the distance measure in Cartesian space. To inform the choice of σ, we can use the sampling frequency $f_{sampling}$ of the acquisition system, when recording trajectories. The higher the frequency, the closer the two consecutive data points. The maximum distance between two consecutive point is $d_{max} = \frac{\text{argmax}(\mathcal{D})}{f_{sampling}}$ with $\mathcal{D} := \{\dot{\mathbf{x}}_i, i = 1, \ldots, m\}$. If sampling is perfect (no frame loss), we can set $\sigma = d_{max}$. Otherwise, a margin of error may be warranted.

The adjacency matrix is computed as follows: $A_{ij} = 1$ if $k(\nu_i, \nu_j) \geq \epsilon$ otherwise $A_{ij} = 0$. The tolerance ϵ must be chosen in relation to σ. For instance, if $\sigma = d_{max}$, the kernel will be equal to at maximum 0.6 for the two most distant pair of points.

5. Clustering Dynamics & Attractor Search

We consider as toy example a 2-attractor DS where each sub-dynamics is constituted by 3 sampled trajectories. Fig. 9 shows the connection strength generated by the velocity-augmented kernel in Eq. 20. Connection between nodes belonging to different trajectories will have very low weight. Only points at the end of each trajectory, close to an attractor, will have high weight even if they belong to different trajectories.

The labeling of points, for clustering dynamics, is done by taking advantage of the first right eigenvectors, $\{\mathbf{u}^1, \ldots, \mathbf{u}^Q\}$, connected to the eigenvalues equal to zero. Assuming n data points for each of the Q demonstrated dynamics the k-th eigenvectors extracted

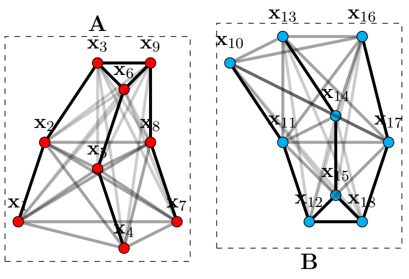

A
B
$\mathbf{x}_1$ $\mathbf{x}_2$ $\mathbf{x}_3$ $\mathbf{x}_4$ $\mathbf{x}_5$ $\mathbf{x}_6$ $\mathbf{x}_7$ $\mathbf{x}_8$ $\mathbf{x}_9$
$\mathbf{x}_{10}$ $\mathbf{x}_{11}$ $\mathbf{x}_{12}$ $\mathbf{x}_{13}$ $\mathbf{x}_{14}$ $\mathbf{x}_{15}$ $\mathbf{x}_{16}$ $\mathbf{x}_{17}$ $\mathbf{x}_{18}$

Figure 9: Toy data set of 2-attractor samples DS.

will have non-zero entries between $\mathbf{u}_{n(k-1)+1}$ and $\mathbf{u}_{nk}$, see Fig. 10. For each eigenvector the coefficients corresponding to the points belonging to a particular DS (sub-graph) are equal to a constant value c while all the others are 0. In summary, the number of 0 eigenvalues is used

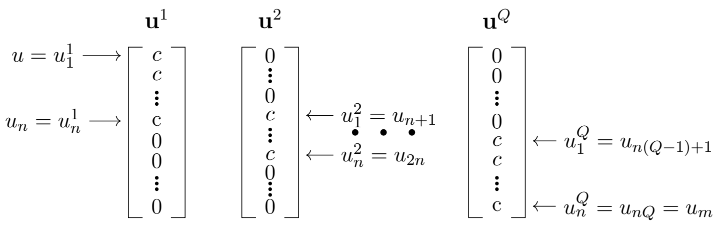

$\mathbf{u}^1$
$\mathbf{u}^2$
$\mathbf{u}^Q$
$u = u_1^1 \longrightarrow$
$u_n = u_n^1 \longrightarrow$
$\longleftarrow u_1^2 = u_{n+1}$
$\longleftarrow u_n^2 = u_{2n}$
$\longleftarrow u_1^Q = u_{n(Q-1)+1}$
$\longleftarrow u_n^Q = u_{nQ} = u_m$

Figure 10: Structure of the constant right eigenvectors associated to eigenvalues equal to 0.

to determine the number of attractors present in the DS. The corresponding eigenvectors are then used to cluster the data points, belonging to each sub-dynamics. Plotting the first four eigenvectors, Fig. 11, it is possible to observe that the spectral decomposition of the Laplacian produces a set of eigenvectors that are "expanding" one dynamics while "compressing", in zero, the other ones for each projected space. In this example $\mathbf{u}^3$ and

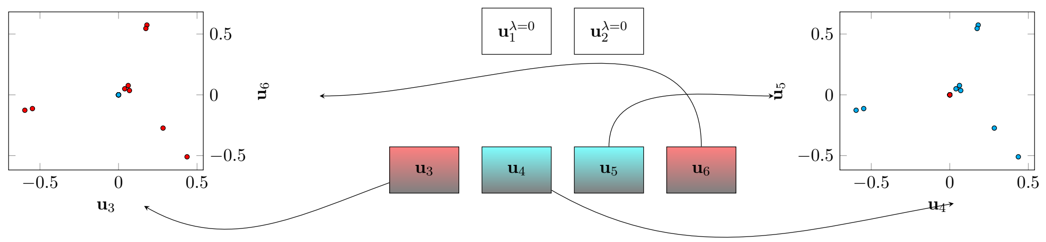


Figure 11: (left) $\{\mathbf{u}_3, \mathbf{u}_6\}$ embedding space; (center) Scatter matrix of the embedding space for the DS in 9; (right) $\{\mathbf{u}_4, \mathbf{u}_5\}$ embedding space.

$\mathbf{u}^6$ focused on sub-dynamics **A** while $\mathbf{u}^4$ and $\mathbf{u}^5$ on sub-dynamics **B**. The selection of the "interesting" components can be easily achieved by checking the change of the slope in the spectrum of the Laplacian matrix. Fig. 12 gives the spectrum analysis for varying the number of attractors, keeping the number of demonstrated trajectories to three. The number of relevant components grows proportionally to the number of sub-dynamics present in the dataset, $n_{DS} = \sum_{q=1}^{K} K_q - 1$ with Q being the total number of sub-dynamics and

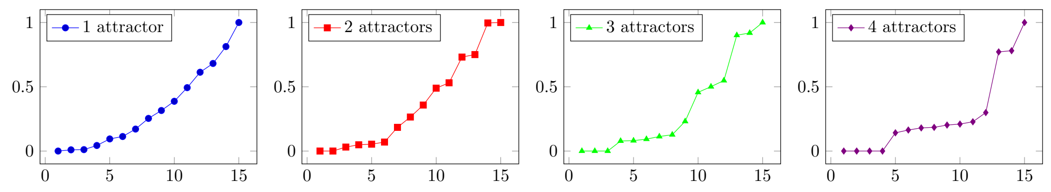


Figure 12: Spectrum analysis of the multiple-attractor DS for three demonstrated trajectories and increasing number of attractors.

K_q the number of sampled trajectories for the q-th sub-dynamics. As shown in Alg. 1, taking advantage of the stationary right eigenvectors used to label points, each eigenvector is assigned to either one dynamics or the other.

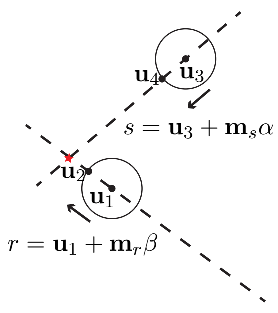
```
Algorithm 1: Clustering Dynamics & Embedding Reconstruction.
Input: $\Lambda = \{\lambda_1, \dots, \lambda_m\}$, $U = \{\mathbf{u}_1, \dots, \mathbf{u}_m\}$
Output: $\mathbf{x}_q^*$, $Q$
$n_{\text{sub-DS}} = (\lambda_k == 0)$            // Compute the number of attractors
$U_{clusters} = \{\mathbf{u}^1_{\lambda=0}, \dots, \mathbf{u}^{n_{\text{sub-DS}}}_{\lambda=0}\}$   // Right "stationary" eigenvectors
while $stop = false$ do
    stop = false
    /* From the first non-stationary eigenvector till the last
       one                                                          */
    for $i = n_{attractors} + 1$ to $m$ do
        /* Iterate over the number of attractors                    */
        for $k = 1$ to $n_{attractors}$ do
            /* Assign eigenvector to sub-dynamics/attractor $k$     */
            if $\mathbf{1}^T(u^k_{\lambda=0} \circ |u_i|) \neq 0$ then
                $U^k \leftarrow u_i$
        /* Stop loop at the spectral gap                            */
        if $|\lambda_{i+1} - \lambda_i| > tol$ then
            stop = true
```

For the toy case in exam we have two eigenvectors associated to zero eigenvalues. The eigenvectors associated the least two eigenvalues, with algebraic multiplicity equal to two, are used for the reconstruction of the embedding spaces as shown in Fig. 11.

The search of the attractor positions is carried out by exploiting the particular structure of the embedding spaces. In each embedding space the related dynamics is "linearized",

```
Algorithm 2: Attractors Search
Input: $U^k = \{\mathbf{u}_1, \dots, \mathbf{u}_d\}$, $n_{\text{attractors}}$
Output: $\mathbf{x}_l^*$
for $k = 1$ to $n_{attractors}$ do
    $i \sim p_{uniform}(u)$
    $u_1 = U^k(i)$
    $u_2 = U^k(\arg\min_{\ell \neq i}(\|u_1 - U^k(\ell)\|))$
    while $stop = false$ do
        $i \sim p_{uniform}(u)$
        $u_3 = U^k(i)$
        $u_4 = U^k(\arg\min_{\ell \neq i}(\|u_3 - U^k(\ell)\|))$
        if $\frac{<u_2 - u_1, u_4 - u_3>}{\|u_2 - u_1\|\|u_4 - u_3\|} < 0.9$ then
            stop = true
        end
    end
end
```

(a)


$\mathbf{u}_4$
$\mathbf{u}_3$
$s = \mathbf{u}_3 + \mathbf{m}_s\alpha$
$\mathbf{u}_2$
$\mathbf{u}_1$
$r = \mathbf{u}_1 + \mathbf{m}_r\beta$

(b)

Figure 13: Finding the attractor in the embedding space.

while the other ones, as said before, are "compressed" in zero. In this space the position of the attractor, $\mathbf{u}^*$, is easily found at the intersection of the lines whose slopes can be calculated using diffusion distances. With reference to Fig. 13b, the algorithm randomly selects a point in the embedding space, $\mathbf{u}_1$ representing the vector containing the embedding coordinates of the chosen point. For a clear visualization of the problem, the process is illustrated in 2-dimensional space but, as shown before, the embedding space can have a higher dimension depending on the number of demonstrated trajectories or the input space dimension. $\mathbf{u}_2$ is selected searching among the nearest neighbors of $\mathbf{u}_1$. The line direction $\boldsymbol{m}_r = \mathbf{u}_2 - \mathbf{u}_1$ is stored. The process is repeated until all the line directions, $\boldsymbol{m}_s = \mathbf{u}_4 - \mathbf{u}_3$, have been found. Pairwise intersections of the of the found lines can be simply calculated by solving the overdetermined linear system $A\boldsymbol{x} = \boldsymbol{b}$, where $A = [\boldsymbol{m}_s, \boldsymbol{m}_r]$, $\boldsymbol{x} = [\alpha, \beta]^T$ and $\boldsymbol{b} = \mathbf{u}_3 - \mathbf{u}_1$ for the two parametric lines, $r = \mathbf{u}_1 + \boldsymbol{m}_r\alpha$ and $s = \mathbf{u}_3 + \boldsymbol{m}_s\beta$. The mean of the intersection points calculated is used to established the position of the attractor, $\mathbf{u}^*$, in the embedding space.

6. Results

We validate our approach first at correctly identifying the underlying dynamics of well-known DS. We choose three nonlinear DS known to embed two separate sub-dynamics which are asymptotically stable at two separate attractors. Second, to test the sensitivity of the approach to real and noisy data, we validate the approach to decompose DS generated from handwritten data. For each of the systems, we generate a set of three example trajectories so as to obtain an embedding of the same dimension as the original system, namely $D = 2$.

To quantify the clustering results, we compare our approach, to three clustering techniques: Kernel K-means, Spectral Clustering and Gaussian Mixture Model (GMM). GMM adopts full covariance matrix. Kernel K-means and GMM require the number of clusters. For those, we provide them with the correct number of dynamics. For Kernel K-means we adopt, as similarity metric, a Radial Basis Functions (RBF) kernel with the hyper-parameter σ set as previously explained. Spectral Clustering adopts *k-nearest* neighborhoods technique for building the adjacency matrix and the identification of the sub-dynamics clusters is achieved through spectral decomposition of the symmetric normalized Laplacian. In order to provide the same information to all the algorithms, the feature state for the compared approaches has been augmented with the velocities.

As none of the clustering techniques we compare can extract the attractors, we compute the reconstruction error for the attractors only for our approach., the *Orthogonal Expansion*. The error is calculated as the squared error between the actual and the extracted location, normalized by the standard deviation of the position data set

$$e(\mathbf{x}^*) = \left((\mathbf{x}^*_{real} - \mathbf{x}^*_{estimated})^T \Sigma^{-1} (\mathbf{x}^*_{real} - \mathbf{x}^*_{estimated})\right)^{\frac{1}{2}}, \tag{23}$$

where Σ is the diagonal matrix containing the standard deviation of the dataset.

Note that K-means and GMM are iterative approaches sensitive to initialization. For each of them we provide average performance over 10 trials as the number of iterations increases.

Example 1 *As a first example we consider the following 2-dimensional DS*

$$\begin{aligned} \dot{x}_1 &= 2x_1 - x_1 x_2 \\ \dot{x}_2 &= 2x_1^2 - x_2, \end{aligned} \tag{24}$$

which present two stable equilibrium points in $(-1, 2)$ and $(1, 2)$. The vector field generated from Eq. 24 and the sampled trajectories are shown in Fig. 14, case (a) and (b) respectively. Such multiple-attractor DS represents a challenging case for our algorithm due to the proximity of the trajectories belonging to different sub-dynamics, symmetrically displaced with respect to the vertical axis passing through zero, Fig. 14b. In order to have a successful reconstruction of the graph we sampled from the DS for $10s$ at a frequency of $100Hz$ reaching a total of 6006 points sampled. The correct reconstruction of the graph can be assessed by looking at Fig. 14c and 14d, where the two sub-dynamics are correctly linearized.

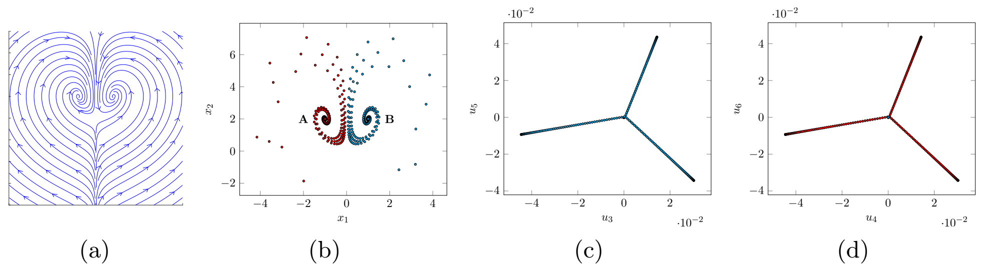
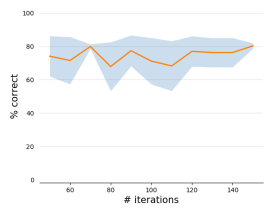


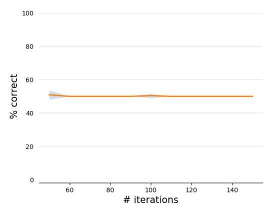
Figure 14: (a) Vector field generated by Eq. 24. (b) Sampled trajectories from the DS in Fig.14a. (c) Embedding space of the sub-dynamics with local attractor in $(1, -2)$. (d) Embedding space of the sub-dynamics with local attractor in $(1, 2)$.

# CLUSTER	**A**	**B**	$e(\mathbf{x}^*)$
Our Approach			
Cluster 1	100%	0%	0.061
Cluster 2	0%	100%	0.061
Kernel K-Means			
Cluster 1	0%	100%	-
Cluster 2	58.42%	41.58%	-
Spectral Clustering			
Cluster 1	0%	100%	-
Cluster 2	79.04%	20.96%	-
Gaussian Mixture Models			
Cluster 1	12.71%	87.29%	-
Cluster 2	12.71%	87.29%	-

(a)



(b)



(c)

Table 1: For sampled points in Fig. 14b: (a) Clustering labeling and attractor location error results, (b) Kernel K-Means clustering error over iteration, (c) Gaussian Mixture Model clustering error over iteration.

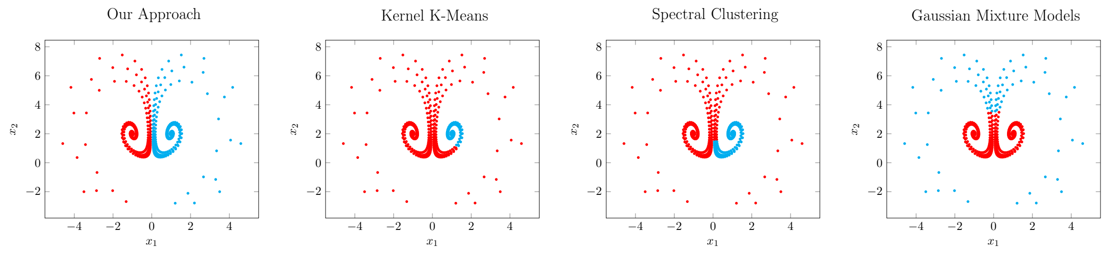
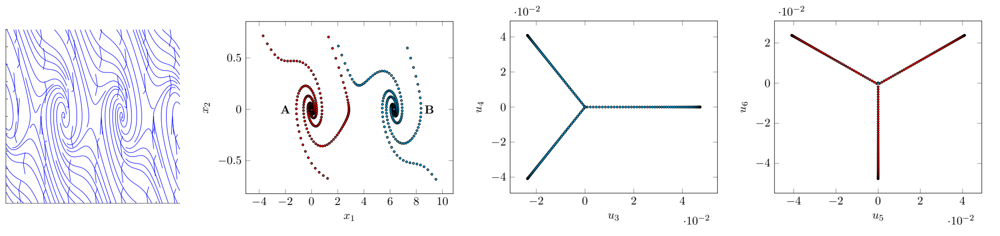


Figure 15: Clustering results of Tab. 1a.

In particular using components $\mathbf{u}_3$ *and* $\mathbf{u}_5$ *the embedding space linearizes sub-dynamics* **B** *while sub-dynamics* **A** *is compressed in zero; vice-versa for the embedding space constructed adopting* $\mathbf{u}_4$ *and* $\mathbf{u}_6$ *components. Results of clustering are reported in Tab. 1a and Fig.15. Kernel K-means does not yield good performance and, as shown in Fig. 1b, it remains fairly sensitive to centroids initialization. Spectral Clustering shows good performance being able to cluster correctly the two high-density region areas. Although it fails in clustering correctly regions with sparse data. GMM present stable solution with respect to parameters initialization, see Fig. 1c, but poor performance failing in placing the mixtures in a consistent way with respect to the two sub-dynamics. In particular the two high-density regions around the attractors are described by a single component yielding incorrect clustering.*

Example 2 *We consider in this example the pendulum equation with friction. The second*



(a) (b) (c) (d)

Figure 16: (a) Vector field generated by Eq. 25. (b) Sampled trajectories from the DS in Fig.16a. (c) Embedding space of the sub-dynamics with local attractor in $(0, 0)$. (d) Embedding space of the sub-dynamics with local attractor in $(0, 2\pi)$.

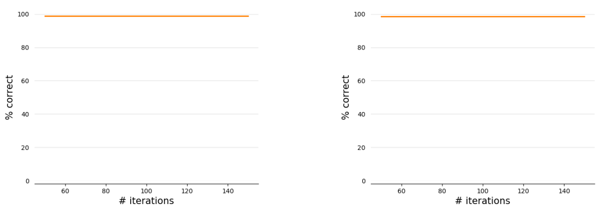
order differential equation describing the dynamics of the pendulum is $\ddot{\theta} = -\frac{l}{g} sin(\theta) - kl\dot{\theta}$, *where* l *is the length of the pendulum,* g *the gravity constant and* k *the friction coefficient. The phase space of such system yields a multiple-attractor vector fields with periodic stable equilibria at* $k2\pi$, *with* $k \in \mathbb{Z}$. *Let* $x_1 = \theta$ *and* $x_2 = \dot{\theta}$. *The pendulum dynamics can be rewritten as a system of first order differential equations*

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= -\frac{l}{g} sin(x_1) - klx_2 \end{aligned} \tag{25}$$

We consider a domain $x_1 \in [0, 2\pi]$ *where such DS presents 2 attractors at the boundary*

# CLUSTER	**A**	**B**	$e(\mathbf{x}^*)$
Our Approach			
Cluster 1	100%	0%	1.34%
Cluster 2	0%	100%	1.34%
Kernel K-Means			
Cluster 1	0%	100%	-
Cluster 2	97.67%	2.33%	-
Spectral Clustering			
Cluster 1	3.63%	96.37%	-
Cluster 2	100%	0%	-
Gaussian Mixture Models			
Cluster 1	0%	100%	-
Cluster 2	97.00%	3.00%	-

(a)



(b) (c)

Table 2: For sampled points in Fig. 16b: (a) Clustering labeling and attractor location error results, (b) Kernel K-Means clustering error over iteration, (c) Gaussian Mixture Model clustering error over iteration.

of the domain and one unstable point at π. *In order to asses the ability of our kernel* *to generate the desire graph structure to build the Laplacian matrix we appositely sampled proximal trajectories pointing to two different attractors (central region in Fig. 16b). Each trajectories has been sampled for* 60s *at a frequency of* 10*Hz for a total* 3592 *points. As it is possible to see from the embedding spaces extracted, Fig. 16c and 16d, the kernel is able to approximate the theoretical graph proposed leading to linearization of the sub-dynamics. In Tab. 2a and Fig. 17 the results of clustering are shown. All the algorithms yields good*

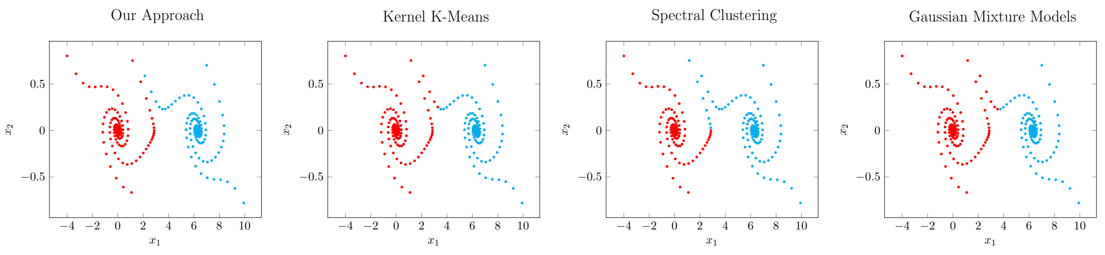
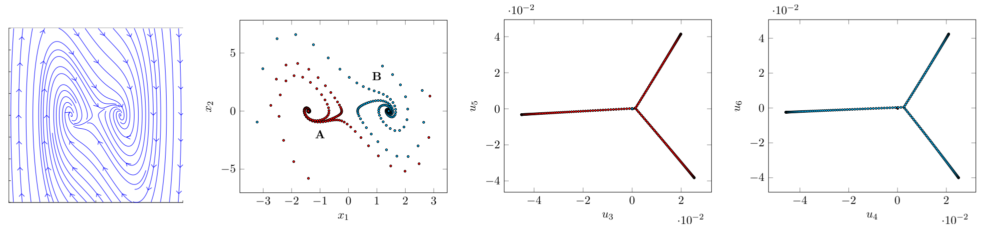


Figure 17: Clustering results of Tab. 2a.

performance in this case correctly assigning points belonging to high-density regions around attractors. Nevertheless, with exception of our approach, they all fail to achieve perfect clustering due to mis-clustering in regions with sparse data.

Example 3 *As a third example we consider the Duffing equation (or oscillator), a non-*



(a) (b) (c) (d)

Figure 18: (a) Vector field generated by Eq. 27. (b) Sampled trajectories from the DS in Fig.18a. (c) Embedding space of the sub-dynamics with local attractor in $(0, \sqrt{-\alpha/\beta})$. (d) Embedding space of the sub-dynamics with local attractor in $(0, -\sqrt{-\alpha/\beta})$.

linear second-order differential equation used for modeling damped oscillator

$$\ddot{x} + \delta\dot{x} + \alpha x + \beta x^3 = 0. \tag{26}$$

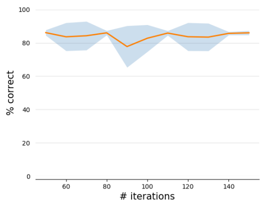
Differently from a simple harmonic oscillator such equation models more complex potential by including a non-linear spring with restoring force $\alpha x + \beta x^3$*. Such DS exhibits chaotic behavior. We consider positive damping case,* $\delta = 0.3$*, with* $\alpha = -1.2 < 0$ *and* $\beta = 0.3 > 0$ *which present two stable at* $+\sqrt{-\alpha/\beta}$ *and* $-\sqrt{-\alpha/\beta}$*. The phase space of the duffing equation can be achieved by the following two dimension DS*

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= 0.3(4x_1 - x_1^3 - x_2). \end{aligned} \tag{27}$$

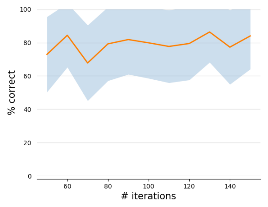
As for example 1, due the presence of highly compact and non-linear regions, high frequency sampling is required. Each trajectory is sample at $200Hz$ *for* $5s$ *yielding a dataset of* 6006 *samples. For this case eigenvectors* $\mathbf{u}_3$ *and* $\mathbf{u}_5$ *yield an embedding space where sub-dynamics*

# CLUSTER	**A**	**B**	$e(\mathbf{x}^*)$
Our Approach			
Cluster 1	100%	0%	1.34%
Cluster 2	0%	100%	1.34%
Kernel K-Means			
Cluster 1	100%	0%	-
Cluster 2	27.81%	72.19%	-
Spectral Clustering			
Cluster 1	94.61%	5.39%	-
Cluster 2	5.65%	94.35%	-
Gaussian Mixture Models			
Cluster 1	4.12%	95.88%	-
Cluster 2	94.88%	5.12%	-

(a)



(b)



(c)

Table 3: For sampled points in Fig. 18b: (a) Clustering labeling and attractor location error results, (b) Kernel K-Means clustering error over iteration, (c) Gaussian Mixture Model clustering error over iteration.

A *is linearized while the space generate by* $\mathbf{u}_4$ *and* $\mathbf{u}_6$ *achieve linearization of sub-dynamics* **B**. *The results of clustering in Tab. 3a and Fig. 19 show a similar behavior for Spectral*

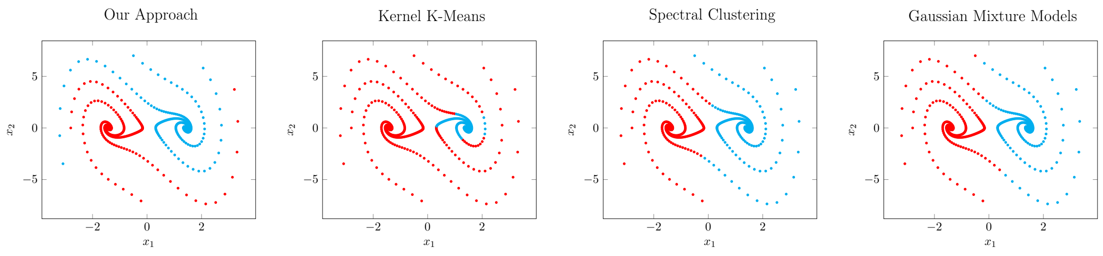


Figure 19: Clustering results of Tab. 3a.

Clustering and GMM. Both algorithms yields quantitative good results. Although, due to the chaotic behavior of the DS they fail in clustering correctly trajectories close to the attractor belonging to another sub-dynamics. Kernel K-means is able to cluster correctly points lying close to the attractors but it is incapable of clustering the majority of the trajectories.

Example 4 *As last examples we show applicability to hand-drawn multiple-attractor DS*

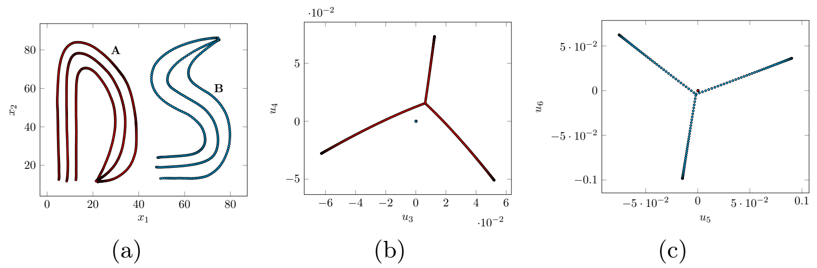


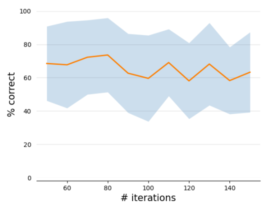
Figure 20: (a) Demonstrated trajectories. (b) Embedding space of the red sub-dynamics. (d) Embedding space of the cyan sub-dynamics.

where our algorithm can be used to cluster the sub-dynamics and locate the various attractors for then providing such information to stable learning algorithms present in the literature.

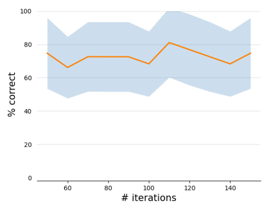
Fig. 20a shows the demonstrated trajectories for a 2-attractor DS in 2D. For drawing such trajectories we took advantage of Wacom tablet and the software provided by ML_toolbox[1]*. Spectral Clustering is able to achieve perfect clustering while GMM misplaces the Gaussian*

# CLUSTER	**A**	**B**	$e(\mathbf{x}^*)$
Our Approach			
Cluster 1	100%	0%	1.56%
Cluster 2	0%	100%	1.56%
Kernel K-Means			
Cluster 1	48.85%	51.15%	-
Cluster 2	48.67%	51.33%	-
Spectral Clustering			
Cluster 1	100%	0%	-
Cluster 2	0%	100%	-
Gaussian Mixture Models			
Cluster 1	55.23%	44.77%	-
Cluster 2	47.71%	52.29%	-

(a)



(b)



(c)

Table 4: For sampled points in Fig. 20a: (a) Clustering labeling and attractor location error results, (b) Kernel K-Means clustering error over iteration, (c) Gaussian Mixture Model clustering error over iteration.

components yielding poor results. Kernel K-means yields particular poor results in this case

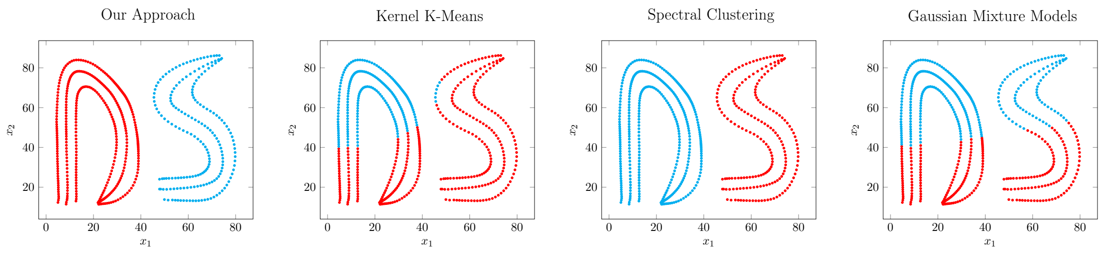
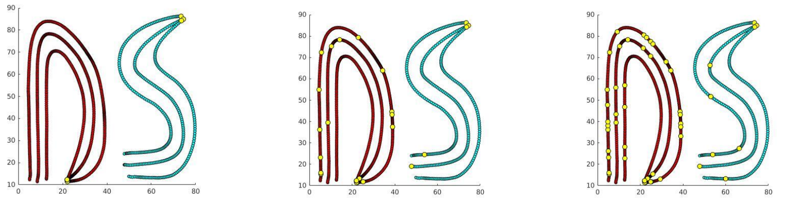


Figure 21: Clustering results of Tab. 4a.

when a small kernel width such the one adopted for our algorithm is used. In order to increase the performance of Kernel K-means we set the kernel width equal to the standard deviation of the dataset. Nevertheless Kernel K-means is not able to consistently cluster the



(a) (b) (c)

Figure 22: Quasi-zero velocity heuristic for attractor location. Threshold at (a) 10%, (b) 5% and (c) 1% of the average velocity.

two sub-dynamics as shown in Fig. 4b. For such case the real attractor location is assumed to be the mean average of the position of the last point of the demonstrated trajectories. For this specific case we show results for the location of the attractor based on quasi-zero velocity

1. https://github.com/epfl-lasa/ML_toolbox

heuristic. Fig.22 shows the identification of potential locations for the attractors based on different velocity thresholds. Notice that zero-velocity crossing ends up with identifying multiple and incorrect attractor locations. This happens often at the beginning of a trajectory or in region of high curvature where the velocity can be proximal to zero.

7. Dynamical System Learning via NVP Transformations

From a differential geometry perspective Manifold Learning has a simple and straightforward interpretation. Consider the *differentiable* manifold $\mathcal{M}$ in Fig. 23. The maps $\mathbf{x}$ and $\mathbf{u}$, generally called *chart maps*, represent two different (local) representation of the manifold in the Euclidean spaces $\mathbb{R}^d$ and $\mathbb{R}^s$.

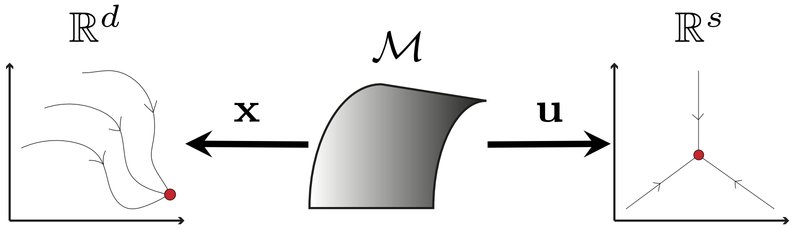


Figure 23: (left) Original Euclidean Space where the dataset is sampled from; (center) manifold where the DS is taking place; (right) extracted Euclidean embedding space.

Going back to the problem of learning stable DS, we view trajectories as motions over a differentiable manifold $\mathcal{M}$. Starting from an Euclidean representation $\mathbb{R}^d$ of such motions (the original dataset), our approach is capable of reconstructing an alternative Euclidean representation $\mathbb{R}^s$ of the manifold in which the DS looks linear. If the number of selected eigenvectors to reconstruct the embedding space is equal to the dimension of the original space, namely $s = d$, such ensemble represents the discrete counterpart of the continuous diffeomorphic map that links the two different representation of the manifold $\mathbf{u} \circ \mathbf{x}^{-1} : \mathbb{R}^d \to \mathbb{R}^s$. A differentiable manifold ensures that any two charts (representations of the same manifold) are differentiable-compatible, namely $\mathbf{u} \circ \mathbf{x}^{-1}$ and its inverse $\mathbf{x} \circ \mathbf{u}^{-1}$ are continuous and differentiable. This properties allows to relate velocity components in the two different Euclidean spaces by the means of the so called Jacobian. For easier notation let the diffeomorphism be $\psi = \mathbf{u} \circ \mathbf{x}^{-1}$, and the Jacobian $\mathbf{J}_\psi = \partial_\mathbf{x} \psi$ then

$$\dot{\mathbf{u}} = \mathbf{J}_\psi \dot{\mathbf{x}}. \tag{28}$$

It is easy to show that if Lyapunov stability is guaranteed in one Euclidean representation of $\mathcal{M}$ the diffeomorphism *induces* the same property in the other one. Consider the existence of a Lyapunov function $V : \mathbb{R}^d \to \mathbb{R}$ radially unbounded and positive in all the space with $\dot{V}(\mathbf{x}) < 0$ except in $\mathbf{x}^*$, equilibrium point, where $V(\mathbf{x}) = \dot{V}(\mathbf{x}) = 0$. The diffeomorphism ψ generates a Lyapunov function $\tilde{V} : \mathbb{R}^s \to \mathbb{R}$ and a related attractor $\mathbf{u}^* = \psi(\mathbf{x}^*)$

$$\dot{\tilde{V}}(\mathbf{u}) = \frac{\partial V}{\partial \mathbf{x}} \frac{\partial \psi^{-1}}{\partial \mathbf{u}} \frac{\partial \mathbf{u}}{\partial t} = \frac{\partial V}{\partial \mathbf{x}} \mathbf{J}_\psi^{-1} \mathbf{J}_\psi \dot{\mathbf{x}} = \frac{\partial V}{\partial \mathbf{x}} \dot{\mathbf{x}} = \dot{V}(\mathbf{x}) < 0. \tag{29}$$

Given the bijectivity of the diffeomorphism if $\tilde{V}$ is a Lyapunov function defining a stable equilibrium point at $\mathbf{u}^*$, $\mathbf{x}^*$ is a globally asymptotically stable equilibrium point. We apply this principle to reconstruct the dynamics in the original space.

We start by constructing a linear DS in embedding space that follows the linearized trajectories of our DS: $\dot{\mathbf{u}} = \mathbf{u}^* - \mathbf{u}$. This system is globally asymptotically stable at $\mathbf{u}^*$. Stability can be proved easily by considering the quadratic Lyapunov function $\tilde{V}(\mathbf{u}) = \frac{1}{2}(\mathbf{u}^* - \mathbf{u})^T(\mathbf{u}^* - \mathbf{u})$. Observe that $\tilde{V}$ is also the potential of the vector field (namely $\dot{\mathbf{u}} = \nabla\tilde{V}$). Following from Eq. 28, the original dynamics can be recovered through

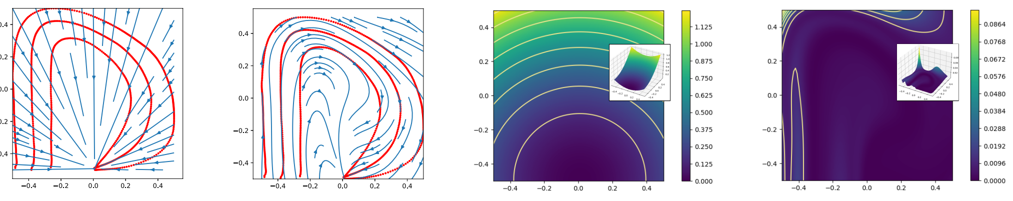
$$\dot{\mathbf{x}} = \mathbf{J}_{\psi}^{-1}(\mathbf{u}^* - \mathbf{u}) = \mathbf{J}_{\psi}^{-1}(\psi(\mathbf{x}^*) - \psi(\mathbf{x})). \tag{30}$$

The corresponding (deformed) Lyapunov function in original space can be recovered as $V = \tilde{V} \circ \psi = \frac{1}{2}(\psi(\mathbf{x}^*) - \psi(\mathbf{x}))^T(\psi(\mathbf{x}^*) - \psi(\mathbf{x}))$.

In order to reconstruct the diffeomorphic map using the embedding space as a ground truth, we adopt Non-Volume Preserving (NVP) transformations introduced by Dinh et al. (2016). The diffeomorphism is obtained through a sequence of k *coupling layers* each of which is given by:

$$\begin{cases} \mathbf{u}_{1:n} &= \mathbf{x}_{1:n} \\ \mathbf{u}_{n+1:d} &= \mathbf{x}_{n+1:d} \odot \exp(s(\mathbf{x}_{1:n})) + t(\mathbf{x}_{1:n}) \end{cases} \tag{31}$$

with $n < d$, d the dimension of the original space. $s(\cdot)$ and $t(\cdot)$ are scaling and translating functions, respectively. Each of these functions is approximated through Random Fourier feature approximation (Rahimi and Recht (2007)) of a vector-valued isotropic Radial Basis Functions kernel as shown by Rana et al. (2020). As loss function, we use the standard Mean



(a) (b) (c) (d)

Figure 24: (a) Linear DS generated when using identity diffeomorphism; (b) reconstructed dynamics through learned diffeomorphism; (c) initial quadratic potential function generating a linear DS in Fig. 24a; (d) deformed potential under the action of the learned diffeomorphism generating the nonlinear DS in Fig. 24b.

Squared Error, MSE $= \sum_{i=1}^{N_D} \|\mathbf{u}_i - \psi(\mathbf{x}_i)\|^2$, adopting the embedding space coordinates as ground truth.

We applied the diffeomorphism to learn a mapping from our original space to the embedding spaces for the hand-drawn multiple-attractor unknown DS, presented in Fig. 20a. Fig. 24 focuses on the red sub-dynamics. Fig. 24a shows the original linear dynamics produced by Eq. 30 when the diffeomorphism ψ is the identity. After having learned the diffeomorphism, the deformed nonlinear DS is shown in Fig. 24b. Fig. 24c and 24d show the potential function $V = \tilde{V} \circ \psi$ with identity and learned diffeomorphism, respectively. Results for the second extracted sub-dynamics are shown in Fig. 25.

Our approach to learning DS through diffeomorphic mapping is highly inspired by the works of Perrin and Schlehuber-Caissier (2016) and Rana et al. (2020). Different from the approach of Perrin and Schlehuber-Caissier (2016), our learned diffeomorphism generates a map from the original space to the embedding space, rather than the opposite. This leads

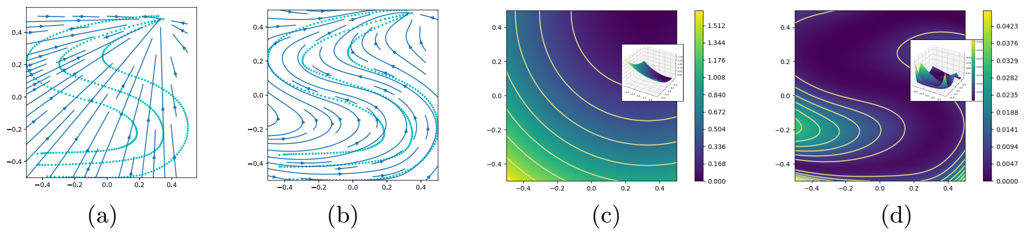


Figure 25: (a) Linear DS generated via identity diffeomorphism; (b) deformed DS under the action of the learned diffeomorphism; (c) initial quadratic potential function generating the linear DS in Fig. 25a; (d) deformed potential under the action of the learned diffeomorphism generating the nonlinear DS in Fig. 25b.

to a different formulation of the deformed non-linear DS. The advantage of our approach is that it does not require to construct explicitly the inverse diffeomorphism. This comes at the cost of having to invert the Jacobian when reconstructing the DS. Compared to the approach proposed in Rana et al. (2020), we rely solely on the geometrical deformation of the space, induced by the embedding space information, without taking into account dynamics information. As consequence our loss function considers only position information (original

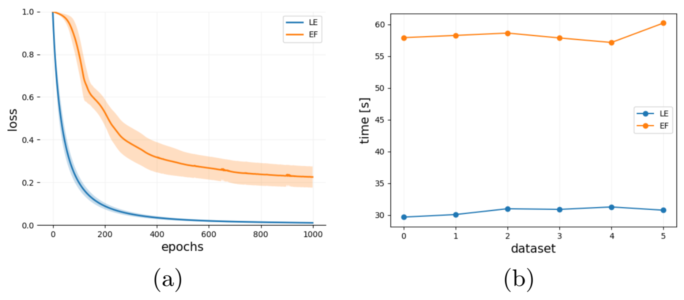


Figure 26: For Laplacian Embedding (LE) and Euclideanizing Flows (EF) approaches: (a) loss decay over 1000 epochs; (b) training time for 1000 epochs.

vs. embedding space) discarding the velocities. This allows to achieve faster convergence, due to the simpler learning problem, and sensible lower learning time due to the advantage of not having to calculate and invert the Jacobian in the process. Fig.26 shows a comparison between our approach, termed as Laplacian Embedding (LE), and Euclideanizing Flows (EF) presented in Rana et al. (2020). All the tests have been performed on a machine endowed with an Intel i9-10900K CPU and a NVIDIA GeForce RTX 2080Ti GPU[2]. On the left it is possible to see how the easier formulation proposed in LE leads to a fast, exponential decaying of the loss while EF generally requires more epochs to properly converge. On the

2. Repository available at: https://github.com/nash169/learn-diffeomorphism

right the training time over 1000 epochs is shown. As LE does not require to invert the Jacobian, it requires approximately half training time compared to EF.

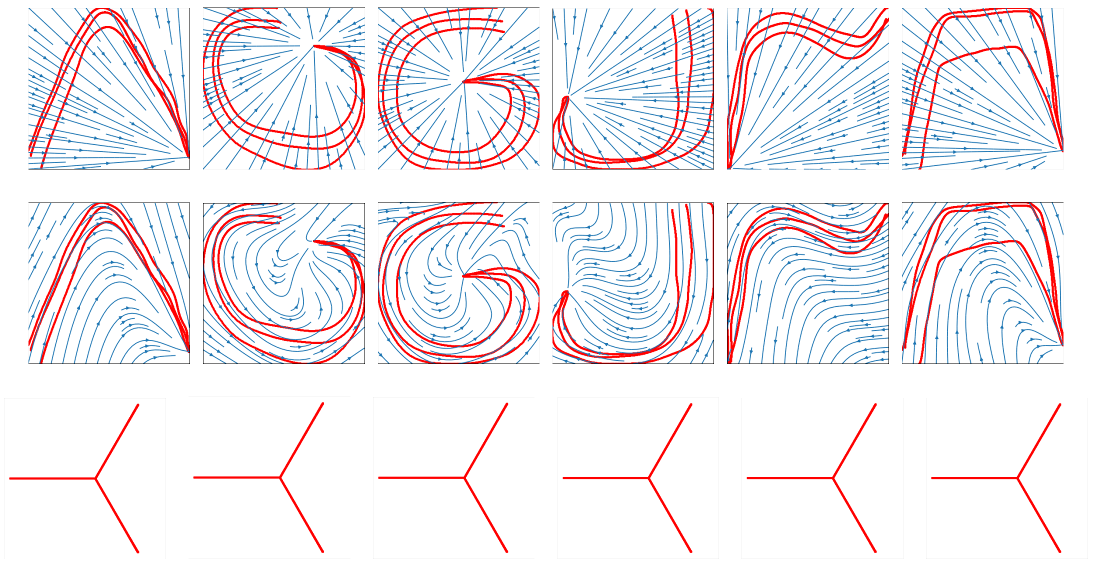

(a) Angle (b) CShape (c) GShape (d) JShape (e) RShape (f) Trapezoid

Figure 27: For each demonstrated DS (column wise), the first row shows the DS generated by the identity diffeomorphism (no train), the second row shows the DS generated after having learned the diffeomorphism between the original space and the embedding space, the third row shows the embedding space reconstructed using the eigenvectors extracted from the eigen-decomposition of the Laplacian matrix.

We evaluate the ability of the proposed non-volume preserving transformation to reconstruct the diffeomorphism between the original and the embedding space on the LASA dataset[3], a dataset composed of hand-drawn letters, that has been often used to compare performance of learned non-linear DS. Each demonstration is composed by 7 trajectories (1000 samples of position-velocity pairs). Since, in two dimensions, the reconstruction of the embedding does not require more than three trajectories, for each demonstration, we discard 4 trajectories that we use at testing time to evaluate performance.

For the reconstruction of the embedding to be successful, it is necessary that the demonstrated trajectories are instances of a first order DS. While noise is tolerated, the trajectories should not intersect, as this would violate a fundamental property of first order DS and core assumption of the DS graph representation. Therefore, whenever it was possible, from each demonstration, we selected trajectories in line with this requirement. In those cases where it was not possible, in order to retain a good statistical analysis on the diffeomorphism learning part, we reconstructed the graph structure manually.

Fig. 28 shows the deformed quadratic potential function under the action of the learned diffeomorphism for a subset of demonstrations, while Fig. 28 shows the relative learned DS. In order to evaluate the performance of the algorithm we employ three metrics: (1) prediction cosine similarity error $\dot{e} = \frac{1}{N_D} \sum_{i=1}^{N_D} \left| 1 - \frac{f(\mathbf{x}_i^{ref})^T \dot{\mathbf{x}}_i^{ref}}{\left\| f(\mathbf{x}_i^{ref}) \right\| \left\| \dot{\mathbf{x}}_i^{ref} \right\|} \right|$, (2) *average* Dynamic Time Warping

3. https://bitbucket.org/khansari/lasahandwritingdataset

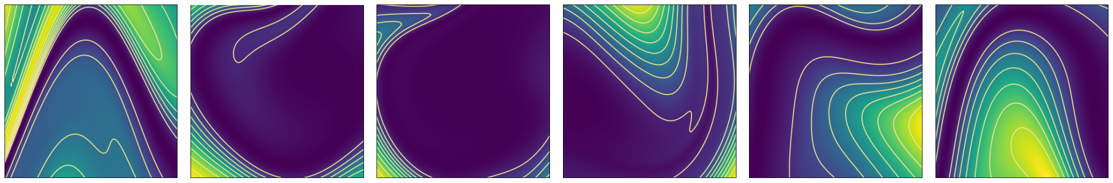

(a) Angle (b) CShape (c) GShape (d) JShape (e) RShape (f) Trapezoid

Figure 28: Deformed potential under the action of the learned diffeomorphism generating the nonlinear DS for the LASA dataset.

Distance (DTWD) Salvador and Chan (2007) and (3) prediction Root Mean Square Error RMSE $= \frac{1}{N_D}\sum_{i=1}^{N_D}\left\|\dot{\mathbf{x}}_i^{ref} - f(\mathbf{x}_i^{ref})\right\|$. Table 5 shows the performance at reconstructing each of the demonstrations for each of the 12 dynamics shown in Figure 27.

	Angle	CShape	GShape	JShape	RShape	Trapezoid
RSME	23.92	16.55	18.50	16.11	14.36	15.13
DTWD	0.015	0.066	0.098	0.060	0.046	0.073
CS	0.679	0.992	0.979	0.861	0.504	0.749

Table 5: Performance evaluation at reconstructing the demonstrations for each of the 12 handdrawn examples of Figure 27. Performance is measured according to three metrics: root mean square error (RSMR), dynamic time warping distance (DTWD) and cosine similarity error (CS).

The proposed implementation of the diffeomorphism appears more capable at shaping the streamlines of the vector field according to the demonstrated trajectories. It also has good accuracy, with values of prediction cosine similarity error and average DTWD about 0.5 and 0.2, respectively, all of which are comparable to the current state-of-the-art (e.g. Figueroa and Billard (2018); Rana et al. (2020)) as shown in Fig.29. This comes at the cost of

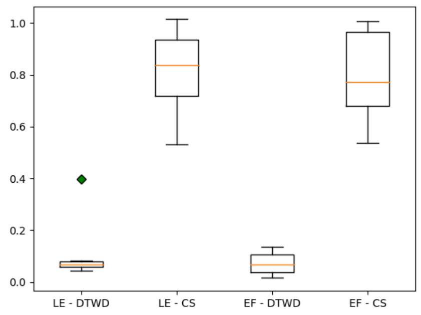


Figure 29: For Laplacian Embedding (LE) and Euclideanizing Flows (EF) approaches box plots comparison of DTWD and CS metrics.

a poor RMSE, over the velocities, given that the velocity information has been discarded in the learning process. Reconstruction of the desired velocity profile can be achieved through proper re-scaling techniques of the reconstructed DS as shown, for instance, by Perrin and Schlehuber-Caissier (2016).

8. Discussion & Conclusion

This paper showed that it is possible to automatically decompose a set of unlabelled data, stemming from a multiple-attractor DS, and to identify the number of underlying dynamics and their associated attractors. We further provided theoretical guarantees for DS linearization based on Manifold Learning reconstruction of multiple embedding spaces. We proved that, for a graph structure of the shape described in Sec. 3, the eigenvectors of the Laplacian matrix generate an embedding space where the sampled trajectories from a given DS are linearized. We proposed a novel velocity-augmented kernel to achieve the desired graph structure from real data.

Relying on these theoretical results, we proposed an algorithm to cluster the sub-dynamics and identify the equilibria locations of multiple-attractor DS through eigendecompostion of a graph-based Laplacian matrix. In particular, we utilized the spectral properties of a Laplacian matrix applied to the particular graph proposed to reconstruct multiple embedding spaces, where each sub-dynamics is linearized while the others are compressed into a single point at zero. We showcased that, in such space, it is possible to identify the position of the attractor while, at the same time, clustering the sub-dynamics.

In comparison to alternative clustering techniques, our approach clusters correctly all examples we tested, even the complicated, chaotic DS generated by the Duffing equation. Our approach also allowed to identify correctly in each cases the attractors' location within an error inferior to 5%. Nevertheless, our algorithm relies heavily on the reconstruction of the correct graph. Whenever the velocity-augmented kernel is not able to reconstruct the correct graph, the clustering performance degrades drastically and it is not possible to locate the attractors anymore. This generally happens in extreme cases of considerable high curvature of the sampled trajectories. In addition we highlight that Spectral Clustering taking advantage of the kernel proposed in this work is capable of matching the clustering performance of our algorithm. However Spectral Clustering does not exploit the particular structure of the DS in the embedding space, and therefore, it is incapable of assessing the location of the attractors.

A usual approach to determining the attractors is to search for zero-velocity crossing. Velocity-crossing usually performs poorly around sharp curvatures since, at the junction, the velocity may decrease importantly. By using the velocity-augmented kernel, our approach is robust to sharp curvatures and can embed highly curvy dynamics, while determining the true attractors.

In all examples used in this paper, the DS were 2-dimensional and represented single motion patterns, However, our algorithm is not limited theoretically to 2D and scales well to higher dimensions since both the proposed kernel and the Laplacian matrix are dimension-independent. The dimension of the embedding space is upper bounded by the number of trajectories sampled and it does not depend on the dimension of the original space.

Combined with state-of-the-art techniques for learning diffeomorphic maps, our method provides an algorithm for stable learning of multiple-attractor DS in a complete unsupervised learning scenario.

Acknowledgments

We would like to acknowledge Aradhana Nayak for her contribution to Prop. 5 and 6.

Appendix A. Proof of Lem. 2 from Sec 3

For each m-th row L_m of $L(G)$, we have $D+1$ non-zero entries, where D is the degree of the m-th node with $L_{m,n} = D$ for $m = n$ and $L_{m,n} = -1$ for $m \neq n$. In the following, for the sake of clarity, we will drop the upper index k for the eigenvector's entries.

The first node, $n = 1$ has degree $D = 1$, as it is connected only to its direct neighbor. Using the monotonic ordered labelling of the nodes, $L_1\mathbf{u} = \lambda u_1$ yields $u_1 - u_2 = \lambda u_1$, that simplifies into:

$$u_2 = (1-\lambda)u_1. \tag{32}$$

The second node of the path graph is connected to the previous and next node. It has hence degree $D = 2$. $L_2\mathbf{u} = \lambda u_2$ yields $2u_2 - u_1 - u_3 = \lambda u_2$. The same holds for all the other nodes in the path. Hence, we have $2u_n - u_{n-1} - u_{n+1} = \lambda u_n$. This recurrence can be rewritten into the recursion:

$$u_{n+1} = (2-\lambda)u_n - u_{n-1} \quad \text{for } n = 2, \dots, p_k - 1, \tag{33}$$

or, equivalently,

$$u_n = (2-\lambda)u_{n-1} - u_{n-2} \quad \text{for } n = 3, \dots, p_k. \tag{34}$$

Appendix B. Proof of Lem. 3 from Sec 3

Consider the following Chebyshev polynomial of first kind T:

$$\begin{aligned} &T_1(\lambda) = 1 \\ &T_2(\lambda) = 1 - \frac{\lambda}{2} \\ &T_n(\lambda) = 2\left(1 - \frac{\lambda}{2}\right) T_{n-1}(\lambda) - T_{n-2}(\lambda) \quad \text{for } n \geq 3, \end{aligned} \tag{35}$$

equivalent to the following trigonometric expression:

$$T_n(\lambda) = \cos\left((n-1)\cos^{-1}\left(1 - \frac{\lambda}{2}\right)\right) \quad \text{for } n \geq 1. \tag{36}$$

Consider the following Chebyshev polynomial of the second kind V:

$$\begin{aligned} &V_1(\lambda) = 1 \\ &V_2(\lambda) = 2\left(1 - \frac{\lambda}{2}\right) \\ &V_n(\lambda) = 2\left(1 - \frac{\lambda}{2}\right) V_{n-1}(\lambda) - V_{n-2}(\lambda) \quad \text{for } n \geq 3, \end{aligned} \tag{37}$$

equivalent to the following trigonometric expression:

$$V_n(\lambda) = \frac{\sin\left(n\cos^{-1}\left(1 - \frac{\lambda}{2}\right)\right)}{\sin\left(\cos^{-1}\left(1 - \frac{\lambda}{2}\right)\right)} \quad \text{for } n \geq 1. \tag{38}$$

The combination of the Chebyshev polynomials in Eq. 36 and 38, as indicated in Eq. 9, yields the trigonometric expression in Eq. 10.

Appendix C. Proof of Prop. 4 from Sec 3

For simplicity, we remove the superscript k in u_n^k and denote it as u_n. To preserve monotonicity, we must determine the conditions for which the entries do not change sign along one path. From Lem. 3, we know that u_n follows periodic functions that are expressed as a combination of trigonometric functions given by Eq. 10 for $n \geq 1$. We study the stationary points of Eq. 10 with respect to the index n:

$$\begin{aligned}\frac{\partial}{\partial n} u_n &= u_1 \left[-\theta \sin((n-1)\theta) - \theta \frac{\lambda}{2} \frac{\cos((n-1)\theta)}{\sin(\theta)} \right] \\ &= \sin((n-1)\theta) + \frac{\lambda}{2} \frac{\cos((n-1)\theta)}{\sin(\theta)} = 0. \end{aligned} \tag{39}$$

Let the eigenvalue λ be

$$\lambda = 2\left(1 - \cos(\theta - 2\pi j)\right), \quad j \in \mathbb{N}. \tag{40}$$

This expression of λ is sufficient to represent all eigenvalues in $[0, 4]$. Replacing Eq. 40 in Eq. 39, we obtain

$$\begin{aligned}\sin((n-1)\theta) + \frac{2(1-\cos(\theta - 2\pi j))}{2} \frac{\cos((n-1)\theta)}{\sin(\theta - 2\pi j)} &= 0 \\ \sin((n-1)\theta) + \tan(\frac{\theta}{2} - \pi j) \cos((n-1)\theta) &= 0 \\ \tan((n-1)\theta) + \tan(\frac{\theta}{2} - \pi j) &= 0. \end{aligned} \tag{41}$$

The stationary points correspond to all n, such that:

$$n = \frac{\pi j}{\theta} + \frac{1}{2}. \tag{42}$$

Expressing the stationary points in term of λ, we have:

$$n = \frac{\pi j}{\cos^{-1}(1 - \frac{\lambda}{2})} + \frac{1}{2}. \tag{43}$$

The monotonicity of the entries u_n is hence preserved until one reaches a stationary point.

Observe, from Eq. 43, that the smaller λ, the larger the number of nodes in the path with monotonicity preserved. If p_k is the number of nodes within each path graph k, we can determine an upper bound on λ for which all u_n within one path would evolve monotonically :

$$\lambda \leq 2 \left[1 - \cos\left(\frac{\pi}{p_k - \frac{1}{2}} \right) \right]. \tag{44}$$

From (44), it is clear that for $p_k \geq 3$ one has additionally $\lambda < 1$. From Prop. 2, we have $u_2 = (1 - \lambda) u_2$, hence $u_2 < u_1$ if u_1 positive and $u_2 > u_1$, otherwise.

Appendix D. Proof of Prop. 5 from Sec 3

In the following we will drop the reference to the underlying graph structure G. Given the circulant block structure of the matrix J in Eq. 13, as shown by Tee (2007), the eigenvalues and corresponding eigenvectors of such matrix are determined by the following K equations, each giving us N eigenvalues and eigenvectors

$$H_j v = \lambda_j v \qquad j \in \{0, \dots, K-1\}, \tag{45}$$

with the matrix H_j defined as

$$H_j \coloneqq B_0 + B_1 \left(\rho_j + \rho_j^{K-1} \right), \tag{46}$$

where $\rho_j = \exp\left(i\frac{2\pi j}{K}\right)$. Observe that $\rho_j^{K-1} = \rho_j^{-1}$ for all j. Further,

$$\rho_j + \rho_j^{-1} = 2\cos\left(\frac{2\pi j}{K}\right) = 2\cos\left(\frac{2\pi(K-j)}{K}\right) = \rho_{K-j} + \rho_{K-j}^{-1}. \tag{47}$$

Therefore the matrices H_j and H_{K-j}, for $j \geq 1$, yield the same eigenvalues. This shows how in the matrix J has at least $\frac{(K-1)}{2}$, if K is odd, or $\frac{K}{2} - 1$, if K is even, eigenvalues with algebraic multiplicity equal to 2.

Appendix E. Proof of Prop 6 from Sec 3

In the following we will drop the reference to the underlying graph structure G. The eigenvalues of the Laplacian matrix L are given by

$$\Lambda_L = 2I - \Lambda_J, \tag{48}$$

where Λ_L and Λ_J are the diagonal matrices containing the eigenvalues of L and J, respectively. Therefore the smallest non-zero eigenvalue of L corresponds to the largest non-zero eigenvalue of J. The matrices H_j in Eq. 46 can be expressed as

$$H_j = I + M_j, \tag{49}$$

where M_j is the matrix

$$M_j = \begin{bmatrix} 0 & 1 & & & \\ 1 & -1 & \ddots & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & -1 & 1 \\ & & & 1 & \alpha_j - 2 \end{bmatrix} = M_{1,j} + M_2, \tag{50}$$

wherein

$$M_{1,j} = \begin{bmatrix} M_0 & v \\ v^T & \alpha_j - 2 \end{bmatrix}, \quad M_2 = \begin{bmatrix} 0 & & & & \\ & -1 & & & \\ & & \ddots & & \\ & & & -1 & \\ & & & & 0 \end{bmatrix} \tag{51}$$

with $\alpha_j = 2\cos\left(\frac{2\pi j}{K}\right)$ and

$$M_0 = \begin{bmatrix} 0 & 1 & & \\ 1 & \ddots & \ddots & \\ & \ddots & \ddots & 1 \\ & & 1 & 0 \end{bmatrix}, \quad v = [0, \ldots, 0, 1]^T . \tag{52}$$

M_0 is the adjacency matrix of a single path graph with $N-1$ vertices. From Thm. 3.7 in Bapat (2010) the eigenvalues of M_0 are given by

$$\lambda_n(P) = 2\cos\left(\frac{\pi n}{N}\right), \quad n = 1, \ldots, N-1. \tag{53}$$

Let λ_{max} and λ_{min} the largest and the smallest eigenvalues in the spectrum, respectively. From Thm 4.3.17 in Horn and Johnson (2012), the largest eigenvalue of each $M_1(i)$ matrices is equal or larger than the largest eigenvalue of M_0. The largest eigenvalue of M_0 is found for $n = 1$. Hence, we have

$$2\cos\left(\frac{\pi}{N}\right) \leq \lambda_n(M_{1,j}), \quad \forall j. \tag{54}$$

From Cor. 4.3.15 in Horn and Johnson (2012) the largest eigenvalue among all M_j matrices is bounded above and below by

$$2\cos\left(\frac{\pi}{N}\right) - 1 = \lambda_{max}(M_{1,j}) + \lambda_{min}(M_2) < \lambda_{max}(M_j) < 2\cos\left(\frac{\pi}{N}\right) \quad \forall j. \tag{55}$$

The inequality is strict as $M_{1,j}$ and M_2 do not have a common eigenvector. Recalling that $\lambda(L) = 1 - \lambda(M_j)$, this is equivalent to

$$1 - 2\cos\left(\frac{\pi}{N}\right) < \lambda_{min}(L) < 2\left(1 - \cos\left(\frac{\pi}{N}\right)\right), \tag{56}$$

being $\lambda_{min}(L)$ the smallest non-zero eigenvalue of the Laplacian matrix. Since $\cos\left(\frac{\pi}{N}\right) > \cos\left(\frac{\pi}{N-\frac{1}{2}}\right)$ for $N \geq 2$, we obtain the inequality in Eq. 16.

References

Ravindra B Bapat. *Graphs and matrices*, volume 27. Springer, 2010.

Mikhail Belkin and Partha Niyogi. Laplacian Eigenmaps for Dimensionality Reduction and Data Representation. *Neural Computation*, 15(6):1373–1396, June 2003. ISSN 0899-7667. doi: 10.1162/089976603321780317.

Aude G Billard, Sylvain Calinon, and Rüdiger Dillmann. Learning from humans. In *Springer handbook of robotics*, pages 1995–2014. Springer, 2016.

Brecht Corteville, Erwin Aertbeliën, Herman Bruyninckx, Joris De Schutter, and Hendrik Van Brussel. Human-inspired robot assistant for fast point-to-point movements. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pages 3639–3644. IEEE, 2007.

Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*, 2016.

Carmeline J Dsilva, Ronen Talmon, Ronald R Coifman, and Ioannis G Kevrekidis. Parsimonious representation of nonlinear dynamical systems through manifold learning: A chemotaxis case study. *Applied and Computational Harmonic Analysis*, 44(3):759–773, 2018.

Burak Erem, Ramon Martinez Orellana, Damon E Hyde, Jurriaan M Peters, Frank H Duffy, Petr Stovicek, Simon K Warfield, Rob S MacLeod, Gilead Tadmor, and Dana H Brooks. Extensions to a manifold learning framework for time-series analysis on dynamic manifolds in bioelectric signals. *Physical Review E*, 93(4):042218, 2016.

Nadia Figueroa and Aude Billard. A Physically-Consistent Bayesian Non-Parametric Mixture Model for Dynamical System Learning. In *Conference on Robot Learning (CoRL)*, pages 927–946, October 2018.

Sthithpragya Gupta, Aradhana Nayak, and Aude Billard. Learning high dimensional demonstrations using laplacian eigenmaps. *arXiv preprint arXiv:2207.08714*, 2022.

Jochen Heinzmann and Alexander Zelinsky. Quantitative safety guarantees for physical human-robot interaction. *The International Journal of Robotics Research*, 22(7-8):479–504, 2003.

Roger A Horn and Charles R Johnson. *Matrix analysis*. Cambridge university press, 2012.

Jessie Huang, Fa Wu, Doina Precup, and Yang Cai. Learning Safe Policies with Expert Guidance. *arXiv:1805.08313 [cs, stat]*, May 2018.

Winfried Lohmiller and Jean-Jacques E. Slotine. On Contraction Analysis for Non-linear Systems. *Automatica*, 34(6):683–696, June 1998. ISSN 0005-1098. doi: 10.1016/S0005-1098(98)00019-3.

S. Manschitz, M. Gienger, J. Kober, and J. Peters. Mixture of Attractors: A Novel Movement Primitive Representation for Learning Motor Skills From Demonstrations. *IEEE Robotics and Automation Letters*, 3(2):926–933, April 2018. ISSN 2377-3766. doi: 10.1109/LRA.2018.2792531.

Jose R. Medina and Aude Billard. Learning Stable Task Sequences from Demonstration with Linear Parameter Varying Systems and Hidden Markov Models. In *Conference on Robot Learning*, pages 175–184, October 2017.

Seyed Sina Mirrazavi Salehian. Compliant control of uni/multi-robotic arms with dynamical systems. Technical report, EPFL, 2018.

S. Mohammad Khansari-Zadeh and Aude Billard. Learning control Lyapunov function to ensure stability of dynamical system-based robot reaching motions. *Robotics and Autonomous Systems*, 62(6):752–765, June 2014. ISSN 0921-8890. doi: 10.1016/j.robot.2014.03.001.

Klaus Neumann and Jochen J. Steil. Learning robot motions with stable dynamical systems under diffeomorphic transformations. *Robotics and Autonomous Systems*, 70:1–15, August 2015. ISSN 0921-8890. doi: 10.1016/j.robot.2015.04.006.

Nicolas Perrin and Philipp Schlehuber-Caissier. Fast diffeomorphic matching to learn globally asymptotically stable nonlinear dynamical systems. *Systems & Control Letters*, 96: 51–59, October 2016. ISSN 01676911. doi: 10.1016/j.sysconle.2016.06.018.

Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. *Advances in neural information processing systems*, 20:1177–1184, 2007.

Muhammad Asif Rana, Anqi Li, Dieter Fox, Byron Boots, Fabio Ramos, and Nathan Ratliff. Euclideanizing Flows: Diffeomorphic Reduction for Learning Stable Dynamical Systems. *arXiv:2005.13143 [cs, eess]*, May 2020.

Harish Ravichandar, Iman Salehi, and Ashwin Dani. Learning Partially Contracting Dynamical Systems from Demonstrations. In *Conference on Robot Learning*, pages 369–378, October 2017.

Stan Salvador and Philip Chan. Toward accurate dynamic time warping in linear time and space. *Intelligent Data Analysis*, 11(5):561–580, 2007.

Bernhard Schölkopf and Alexander J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. Adaptive Computation and Machine Learning. MIT Press, Cambridge, Mass, 2002. ISBN 978-0-262-19475-4.

Tal Shnitzer, Ronen Talmon, and Jean-Jacques Slotine. Diffusion Maps Kalman Filter for a Class of Systems with Gradient Flows. *arXiv:1711.09598 [cs, eess]*, November 2017.

Ashwini Shukla and Aude Billard. Augmented-SVM: Automatic space partitioning for combining multiple non-linear dynamics. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1016–1024. Curran Associates, Inc., 2012.

Vikas Sindhwani, Stephen Tu, and Mohi Khansari. Learning Contracting Vector Fields For Stable Imitation Learning. *arXiv:1804.04878 [cs, stat]*, April 2018.

Jeremias Sulam, Yaniv Romano, and Ronen Talmon. Dynamical system classification with diffusion embedding for ecg-based person identification. *Signal Processing*, 130:403–411, 2017.

Garry J Tee. Eigenvectors of block circulant and alternating circulant matrices. *New Zealand Journal of Mathematics*, 36(8):195–211, 2007.

Joshua B. Tenenbaum, Vin de Silva, and John C. Langford. A Global Geometric Framework for Nonlinear Dimensionality Reduction. *Science*, 290(5500):2319–2323, December 2000. ISSN 0036-8075, 1095-9203. doi: 10.1126/science.290.5500.2319.

Kim P Wabersich and Melanie N Zeilinger. Safe exploration of nonlinear dynamical systems: A predictive safety filter for reinforcement learning. *arXiv preprint arXiv:1812.05506*, 2018.