Persistence B-spline Grids: A Stable Vector Representation of Persistence Diagrams Based on LSPIA

Anonymous Authors[1]

Abstract

Recent years have witnessed the integration of machine learning and topological data analysis. One of prominent problems in applying persistent homology to machine learning tasks is to find a vector representation of a persistence diagram (PD), a summary diagram to represent topological features. In the perspective of data fitting, we propose a stable vector representation, *persistence B-spline grid* (PB), based on the efficient technique of progressive-iterative approximation for least squares B-spline surface fitting (LSPIA). Meanwhile, we offer the theoretical guarantee that our PB method is stable with respect to the p-Wasserstein and the bottleneck distance, the metrics defined on the PD space. We test our method on the synthetic data set and data of dynamical systems.

1. Introduction

In the advent of big data era, one of the significant tasks is to infer the topological structures of data in the different scales. Recently, *persistent homology* (Edelsbrunner et al., 2002; Zomorodian & Carlsson, 2005) provides us a robust tool to capture topological features of data such as holes based on the homology theory. And these multi-dimensional homological features are represented as a *persistence diagram*(PD) or a *barcode* (Ghrist, 2008). Moreover, persistent homology has been applied successfully to deal with various practical problems in different scientific and engineering fields, such as biochemistry (Xia & Wei, 2015; Xia et al., 2018), information science (De Silva & Ghrist, 2007; Rieck et al., 2018), and so on. Meanwhile, machine learning community has had a growing interest in combining traditional ML methods and the current techniques of computational topology (Chazal et al., 2015; Chen & Quadrianto, 2016; Khrulkov & Oseledets, 2018). In the applications of ML, the distance metrics (the p-*Wasserstein distance* and the *bottleneck distance*) of two PDs proposed by Cohen-Steiner et al. (2007) allow one to compute the distance matrix in the tasks of clustering and classification. However, since some usually-used ML tools require vectors as input such as support vector machines (SVMs), it is necessary to represent PDs by stable vectors with fixed length.

Related Work. Many approaches are proposed to transform PDs into vectors (Pachauri et al., 2011; Carriere et al., 2015; Di Fabio & Ferri, 2015; Adcock et al., 2016). In the perspective of statistics, Mileyko et al. (2011) developed the probability measures on the space of PDs. Bubenik (2015) defined a functional representation, persistence landscapes, in a Banach space based on persistent module (Zomorodian & Carlsson, 2005). Persistence landscape satisfies the proposed ∞-landscape stability and p-landscape stability with the same condition as the stability of the p-Wasserstein distance (Cohen-Steiner et al., 2010). However, it's hard to interpret the results of persistence landscape.

Kernel method is an effective framework for PD representation. In the task of researching the cortical thickness measurements on the human cortex for the study of Alzheimer's disease, Pachauri et al. (2011) rasterized PDs on a regular grid, compute a kernel density estimation to generate the concentration map using Gaussian kernel, and vectorize concentration maps for SVM training. But the relationship between the kernel distance they use and the p-Wasserstein distance or the bottleneck distance is unclear. Enlightened by a heat diffusion problem with a Dirichlet boundary condition, Reininghaus et al. (2015) proposed the framework of multi-scale kernels. Unfortunately, due to the additive property of the kernel on PDs, the kernel is proved to be unstable with respect to p-Wasserstein distance where $1 < p \leq \infty$ (It is the bottleneck distance if $p = \infty$). On the other hand, for the purpose to get an interpretable result, an easy approach was used by Rouse et al. (2015) They constructed a finite grid in the birth-persistence coordinates, quantized the birth-persistence pairs to the grid, counted the number of pairs at each grid, and rearranged the counts into a vector in the order of the grid. Nevertheless, this method is sensitive to the division of the grid region, which might not robust with

[1] Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

respect to noise, and the importance of the points in a PD is not taken into consideration.

The idea of persistence images (Adams et al., 2017) are similar as the kernel framework of Reininghaus et al. (2015). First, A product of a 2-dimensional Gaussian distribution and a weighted function are assigned to each point in the birth-persistence coordinates and added up to form a persistence surface. And then the domain is divided by a regular grid into regular small regions called ”pixels”. The integral value of persistence surface is computed on each pixel to generate an image. Note that if the variance σ approaches to zero, the Gaussian function is seen as the mass-density, a Dirac delta, and the integral value on each pixel tends to be the sum of weighted function values of the points on the pixel, which degenerates into the weighted sum in Rouse et al. (2015). But a disadvantage occurs because their persistence surface has the similar form as the feature map from PD space to L^2 space in Reininghaus et al. (2015), which inherits the instability with respect to p-Wasserstein distance where $1 < p \leq \infty$.

Recently, Kusano et al. (2018) summarized the previous works based on kernel functions and developed a statistical framework of kernel method to vectorize persistence diagrams by employing kernel embedding of measures into reproducing kernel Hilbert spaces. Meanwhile, they offered the stability proofs of the kernel method. And Le & Yamada (2018) employed another definite kernel, persistence Fisher kernel, which has some nice properties such as stability and infinite divisibility. On the other hand, a new vector representation of persistence barcode based on tropical coordinates is proposed (Kališnik, 2018), which shows the potential to attempt new frameworks to represent PDs and persistence barcodes.

Our Contribution. In the perspective of data fitting and approximation, we notice that the kernel framework essentially can be regarded to interpolate with kernel functions the ”importance” value assigned on each PD point given by weighting functions. In order to overcome additive property (Reininghaus et al., 2015) of kernels and separate ”importance” values from the basis, we propose a data fitting framework based on the B-spline surface and redesign bounded ”importance” values named *eminence functions* to guarantee the stability with respect to the p-Wasserstein distance and the bottleneck distance. The information of PDs is encoded into the vectors produced by the control points of the *persistence B-spline surface* (PB) so that we can reconstruct precisely the persistence surface from the vectors. We then employ the technique of *progressive-iterative approximation for least squares B-spline surface fitting* (LSPIA) to compute the vectors efficiently and robustly. Generally, the advantages of PB method include that it is

- stable with respect to the p-Wasserstein distance and the bottleneck distance,
- efficient to compute and understandable,
- flexible to design the ”importance” values for different practical tasks,
- interpretable and visible from vectors to the persistence B-spline surfaces,
- and invertible to recover the surfaces from vectors.

2. Background

The features such as connected components, loops and voids keep invariant under a proper geometric transformation. In algebraic topology, homology is a mathematical formalism to measure these ”holes” of different dimensions. Given a topological space X, n-dimensional homology group $H_n(X)$ is generated the elements representing various type of cycles without boundary. The Betti number is defined to the rank of abelian group $H_n(X)$, denoted by β_n. Intuitively, β_0, β_1 and β_2 suggest respectively the number of connected components, one-dimensional loops and two-dimensional voids that a topological space contains. On the other hand, homology allows a fast matrix reduction algorithm working with module 2 coefficients to compute the reduced Betti numbers. Refer to the book (Edelsbrunner & Harer, 2010).

In the real world, one samples from a continuous object to obtain a discrete point cloud. Meanwhile, there exist measurement errors and inevitable sampling noises. A question arises to infer the topological structures of point clouds. *Persistent homology* provides such a tool that the topological features of different dimensions can be captured with their importances marked by their *persistences*. Of the most importance is that it has stability with respect to small perturbations and noises. It is customary to use the *Vietoris-Rips complex* (V-R complex) and the *Čech complex* in the vast toolkit of topology to set up the structure of *simplicial complexes* (Hatcher, 2002). We take the V-R complex as an example. V-R complex contains the points p_i as vertices and n-simplices generated by $n+1$ points of which any pair of closed balls has non-empty intersection. Given a series of parameters $\varepsilon_0 \leq \varepsilon_1 \leq \cdots \leq \varepsilon_m$, we obtain a nested sequence

$$VR(\varepsilon_0) \subseteq VR(\varepsilon_1) \subseteq \cdots \subseteq VR(\varepsilon_m),$$

which is called a *filtration*. We have an inclusion between $VR(\varepsilon_i)$ and $VR(\varepsilon_{i+1})$. It can induce a homomorphism $H_n(VR(\varepsilon_i)) \to H_n(VR(\varepsilon_{i+1}))$ for each n. We have

$$H_n(VR(\varepsilon_0)) \to H_n(VR(\varepsilon_1)) \to \cdots \to H_n(VR(\varepsilon_m)).$$

As ”time” i goes by, some new homological classes emerge and some disappear because they become the boundary of a higher chain or merge with other classes. Vividly, a homological class is *born* at i if it emerges in $H_n(VR(\varepsilon_i))$, and it *dies* at j if it disappears in $H_n(VR(\varepsilon_j))$. Its lifetime $j-i$ is the (index) *persistence* of the homological class. If a homological class never dies, its persistence is set to infinity. Persistence diagrams (PD) and barcodes (Ghrist, 2008) are two frequently-used ways to represent the persistence of homological features. The pairs (i,j) for n-dimension are marked in the birth-death coordinates on the $\mathbb{R}^2$ plane. Since $i \geq j$ for each pair, it is necessary that all the points on a PD are above or exactly on the diagonal line. In practice, the points close to diagonal line are subjectively regarded as noise.

Finally, we introduce the filtration of functions and stability of PDs. Given a topological space $\mathbb{X}$, $f : \mathbb{X} \to \mathbb{R}$ is a scalar function. We define the *sublevel set* by $f^{-1}(-\infty, a] = \{x \in \mathbb{X} | f(x) \leq a\}$ consisting of all points with its scalar value at most a. It induces a filtration of the topological space $\mathbb{X}$ by the function f, if there is a series of scalar values $a_0 \leq a_1 \leq \cdots \leq a_m$, then

$$f^{-1}(-\infty, a_0] \subseteq f^{-1}(-\infty, a_1] \subseteq \cdots \subseteq f^{-1}(-\infty, a_m]$$

In the sense of the distance function d on the point set P, we can also determine a Čech complex over P. For a function $f : \mathbb{X} \to \mathbb{R}$, if some small perturbations occur, what happens to the PD? The stability of PDs is studied by Cohen-Steiner et al. (2007). With some mild assumptions for the function f and $\widetilde{f}$ with respect to small perturbations, the p-*Wasserstein distance* of two corresponding PDs can be controlled by a certain term of L_∞-norm of $f - \widetilde{f}$. The metrics to measure the similarity of two PDs are the p-Wasserstein distance and the bottleneck distance.

Definition 2.1 *The p-Wasserstein distance between PD_1 and PD_2 is given by*

$$W_p(PD^{(1)}, PD^{(2)}) = \inf_b \Big(\sum_{u \in PD^{(1)}} \|u - b(u)\|_\infty^p \Big)^{1/p}$$

and the bottleneck distance is given by

$$W_\infty(PD^{(1)}, PD^{(2)}) = \inf_b \sup_{u \in PD^{(1)}} \|u - b(u)\|_\infty$$

where u is a point in $PD^{(1)}$, b is a bijection between $PD^{(1)}$ and $PD^{(2)}$, and W_∞ denotes the bottleneck distance.

3. Method and Algorithm

In this section, we introduce our persistence B-spline grid to vectorize PDs. We transform a PD into the birth-persistence coordinates and assign an *eminent value* to each point in a PD to mark its importance. A *persistence B-spline surface* is defined to fit the data points in 3D Euclidean space. The surface reflects local tendencies of the eminences on each topological feature due to the local support property of the B-spline basis, i.e. each basis has a small close interval as its support set. Eventually, we propose the algorithm to efficiently and robustly compute the persistence B-spline surface by the technique of LSPIA (Deng & Lin, 2014).

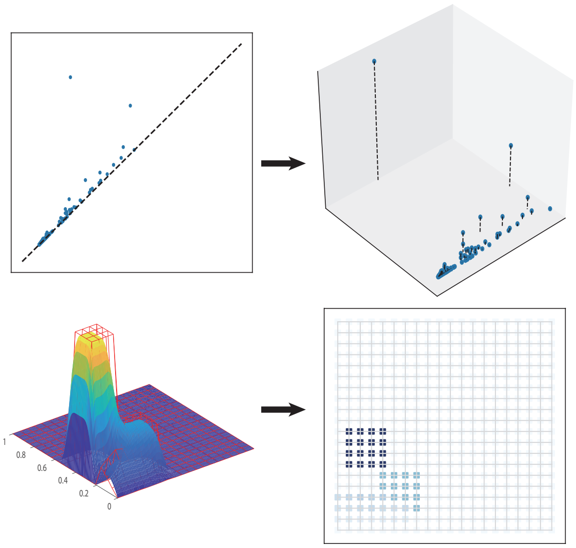

Figure 1. Pipeline: A persistence diagram is transformed in the birth-persistence coordinates and an ”importance” value is assigned on each point to illustrate the importance of the topological feature. Then the 3D data points are fitted by a cubic uniform B-spline surface with the technique of progressive and iterative approximation for least squares B-spline surface (LSPIA). Finally, the control grid (the red grid in bottom left figure) is obtained to generate a vector by concatenating rows of z-coordinates of the control grid. Persistence **B-spline Grid** (bottom right) is defined by the matrix formed by z-coordinates of the control points .

3.1. Construction of Persistence B-spline surfaces

First, we transform a PD into the birth-persistence coordinates. For each (x_l, y_l) in a PD, let $\phi : \mathbb{R}^2 \to \mathbb{R}^2$ be a linear transform satisfying $\phi(x,y) = (x, y-x)$, And we normalize the domain in birth-persistence coordinates from $\mathbb{R}^2$ to $[0,1]^2$ by a normalizing function $c(x,y) = (x/m, y/m)$. Denote

$$(s_l, t_l) = c \circ \phi(x_l, y_l) \tag{1}$$

for each (x_l, y_l) in a PD. Since, in practice, a collection of PDs $\{PD\}_\Lambda$, obtained by V-R complex usually contain finite number of points with finite persistence [1], we make

[1] Although there exists a topological feature with infinite persistence that represents the connectedness of the whole simplicial complex in H_0 PDs, to compute PDs of the corresponding H_0 reduced homology gets rid of this challenge.

m be the maximum of the both x and y coordinate for each point in all PDs, i.e. $m = \max_{\Lambda}\{\max_{(x,y)\in PD}\{x, y\}\}$.

Since the importance of each point in a PD is illustrated variously in different tasks, an eminence value z_l is assigned on each point in a PD by an *eminence function* $\mathscr{E}(s,t)$. We offer a simple model for the eminence functions while taking the persistence and the local density of a point in a PD into consideration. We can take

$$p(s,t) = \begin{cases} t & t \le L \\ L & t > L \end{cases} \tag{2}$$

where L is a positive constant integer.

Define $\mathscr{E} : \mathbb{R}^2 \to \mathbb{R}$ by

$$z_l := \mathscr{E}(s_l, t_l) = \begin{cases} n_\varepsilon p(s_l, t_l) & 1 \le n_\varepsilon \le M \\ M p(s_l, t_l) & n_\varepsilon > M \end{cases} \tag{3}$$

where n_ε is the number of points contained by an ε-open ball of (s_l, t_l) in a PD, and M is a positive constant integer.

To make our PB stable with respect to the distance metrics in PD space, the eminence function $\mathscr{E}$ is required to be bounded. One may choose more complicated eminence function to emphasize local density or the points with different persistences.

Eventually, we use a uniform cubic B-spline surface with a collection of control points $\{(s_i, t_j, p_{ij})\}_{i,j=1}^{h}$ to fit the data points (s_l, t_l, z_l), where each (s_i, t_j) is a knot of an $h \times h$ uniform grid in $[0,1]^2$. Precisely, we give the following definitions.

Definition 3.1 (Persistence B-spline Surface) *Given a PD, for each (x_l, y_l) in the PD, after transformation, we have (s_l, t_l). And z_l (3) is the assigned eminence value on each (s_l, t_l). A uniform cubic B-spline surface with its uniform control grid $\{(s_i, t_j, p_{ij})\}_{i,j=1}^{h}$ satisfying*

$$\min_{\{p_{ij}\}} ||z_l - S_p(s_l, t_l)||_2$$

$$s.t. \quad S_p(s,t) = \sum_{i=0}^{h}\sum_{j=0}^{h} p_{ij} B_i(s) B_j(t) \quad s,t \in [0,1]$$

is a persistence B-spline surface for the corresponding PD, where $B_i(\cdot)$ and $B_j(\cdot)$ are the uniform cubic B-spline basis determined by uniform knot sequences. $\{p_{ij}\}$ are the z-coordinates of the control points.

The control points of PB surfaces provide a flexible method to vecterize the PDs.

Definition 3.2 (Persistence Grid & Persistence Vector) *If $S_p(s,t)$ is the persistence B-spline surface for a PD, then the z-coordinate of the control points, $\{p_{ij}\}_{i,j=1}^{n}$, is called the persistence grid of a PD. A vector generated by arranging it with the lexicographic order,*

$$\widetilde{\boldsymbol{Z}} = (p_{11}, p_{12}, \cdots, p_{1h}, p_{21}, \cdots, p_{hh})^T \in \mathbb{R}^{h^2},$$

is called the persistence vector of a PD.

Especially when we deal with the PDs for H_0, in which the birth coordinate s_l for each point in PDs is equal to zero, PB allows to fit the $\{(t_l, z_l)\}$ data set by the uniform cubic B-spline curves.

3.2. Algorithm and LSPIA

According to definition 3.1, we have the following linear equations

$$\boldsymbol{B}\widetilde{\boldsymbol{Z}} = \boldsymbol{Z} \tag{4}$$

where $\boldsymbol{B}$ is the cubic B-spline basis matrix, and

$$\widetilde{\boldsymbol{Z}} = (p_{11}, p_{12}, \cdots, p_{1h}, p_{21}, \cdots, p_{hh})^T,$$
$$\boldsymbol{Z} = (z_1, z_2, \cdots, z_k)^T.$$

Normally, $\boldsymbol{B}$ is singular. An alternative approach of computing the minimum-norm least squares solution of the linear equations is to figure out the Moore-Penrose (M-P) pseudo-inverse of the matrix $\boldsymbol{B}$. Unfortunately, it possibly suffers from numerical instability. Considering the efficiency and numerical robustness to work out $\{p_{ij}\}$, we employ a flexible and efficient tool, LSPIA, from scientific computation to avoid the expensive time cost and to keep the robustness.

Precisely, given a data set $\{\boldsymbol{Q}_l\}_{l=1}^{k}$ where each $\boldsymbol{Q}_l = (s_l, t_l, q_l)$, assume that each (s_l, t_l) is in the domain $[0,1]^2$. Before the iteration, the control points $\{\boldsymbol{P}_{ij}\}_{i,j=0}^{n}$, where $\boldsymbol{P}_{ij} = (s_i, t_j, p_{ij})$, is intialized. For each $\boldsymbol{P}_{ij}^{(0)} = (s_i, t_j, p_{ij}^{(0)})$, let $\{(s_i, t_j)\}$ be the vertices of a uniform grid on $[0,1]^2$ and $p_{ij}^{(0)} = 0$. The initial surface is constructed by

$$S^{(0)}(s,t) = \sum_{i=0}^{n}\sum_{j=0}^{n} p_{ij}^{(0)} B_i(s) B_j(t) \quad s,t \in [0,1]$$

Let $\delta_l^{(0)} = q_l - S^{(0)}(s_l, t_l)$ be the difference vector between measured data and the corresponding points on the surface $S^{(0)}$. And take the initial adjusting matrix of the control points as

$$\Delta_{ij}^{(0)} = \mu_{ij} \sum_{l=1}^{k} B_i(s_l) B_j(t_l) \delta_l^{(0)}.$$

Then add the adjusting matrix to the matrix of the control points $[p_{ij}^{(0)}]$. We obtain the new control points $\boldsymbol{P}_{ij}^{(1)} = (s_i, t_j, p_{ij}^{(1)})$, where $p_{ij}^{(1)} = p_{ij}^{(0)} + \Delta_{ij}^{(0)}$, which generates a new surface $S^{(1)}$. Hence, in the βth iteration, the βth surface is obtained. Then we have

$$\begin{aligned}\delta_l^{(\beta)} &= q_l - S^{(\beta)}(s_l, t_l)\\ \Delta_{ij}^{(\beta)} &= \mu_{ij} \sum_{l=1}^{k} B_i(s_l) B_j(t_l) \delta_l^{(\beta)} \\ p_{ij}^{(\beta+1)} &= p_{ij}^{(\beta)} + \Delta_{ij}^{(\beta)}\end{aligned} \tag{5}$$

and $(\beta+1)$st surface

$$S^{(\beta+1)}(s,t) = \sum_{i=0}^{n} \sum_{j=0}^{n} p_{ij}^{(\beta)} B_i(s) B_j(t) \quad s,t \in [0,1].$$

The iteration will pause once the number of iteration is satisfied.

In our PB, let $c_{i,j} = \sum_{l=1}^{k} B_i(s_l) B_j(t_l)$ and $C := \max_{i,j}\{c_{i,j}\}$ for $i,j = 1,\cdots,h$ respectively. Note that $C \neq 0$. We take

$$\mu_{ij} = 2/C \tag{6}$$

that is, for different i and j, the weight μ_{ij} are the same (Deng & Lin, 2014).

We give the algorithm to compute the persistence grids.

Algorithm 1 LSPIA for PB

```
Input: A measured data set $\boldsymbol{Z} = \{(s_l, t_l, z_l)\}_{l=1}^{k}$,
a positive integer $h$ representing the grid density,
and a positive number $N$ to represent the pause iterations.
Function $S^{(\beta)}(s,t) = \sum_{i=0}^{h} \sum_{j=0}^{h} \tilde{z}_{ij}^{(\beta)} B_i(s) B_j(t)$
$\boldsymbol{P}^{(0)} := [\tilde{z}_{ij}] = \boldsymbol{O}$;
Initiate $\boldsymbol{\delta} = [\delta_l]$ and $\boldsymbol{\Delta} = [\Delta_{ij}]$
for $\beta = 0$ to $N$ do
    $\delta_l^{(\beta)} = z_l - S^{(\beta)}(s_l, t_l)$;
    $\Delta_{ij}^{(\beta)} = 2/C \sum_{l=1}^{k} B_i(s_l) B_j(t_l) \delta_l^{(\beta)}$;
    $\tilde{z}_{ij}^{(\beta+1)} = \tilde{z}_{ij}^{(\beta)} + \Delta_{ij}^{(\beta)}$;
end for
Output: the persistence vector $\widetilde{\boldsymbol{Z}}$ concatenated from each row of $\boldsymbol{P}$
```

At last, we mention that the format (5) of LSPIA used in our PB is proved to be convergent (Lin et al., 2017).

4. Stability

One of the most important properties of PDs is the stability with respect of tiny noise of input data and error due to measurements. In this section, we show that our vectors (3.2) generated by the control points of persistence B-spline surface are stable with respect to the p-Wasserstein distance and the bottleneck distance between two PDs (definition 2.1).

Assumed that each PD has finite number of points $\{(x_l, y_l)\}_1^k$ with persistences of its topological features being finite, i.e. $0 \le y_l - x_l < \infty$ for each l, and $k \le \infty$. In our PB, the eminence value z_l (3) is assigned to emphaze the importance of the points in a PD, Note that $\mathscr{E}$ (3) is bounded. We get a $PD^{(1)}$ before the perturbation and the other $PD^{(2)}$ after it.

We have the following lemmas.

Lemma 4.1 *For $PD^{(1)}$ and $PD^{(2)}$, b is the matching bijection which achieves the infimum in the p-Wasserstein distance or the bottleneck distance ($p = \infty$). $(x_l^{(2)}, y_l^{(2)}) = b((x_l^{(1)}, y_l^{(1)}))$. Then we have*

$$||\boldsymbol{Z}^{(2)} - \boldsymbol{Z}^{(1)}||_p \le C_{p,M,|p'|} W_p(PD^{(1)}, PD^{(2)}) \quad 1 \le p \le \infty$$

where $\boldsymbol{Z}^{(1)}$ and $\boldsymbol{Z}^{(2)}$ are the column vectors formed by $z_l^{(1)}$ and $z_l^{(2)}$ respectively, $C_{p,M,|p'|}$ is a constant dependent on the function $p(s,t)$ (2), and the norm $||\cdot||_p$.

And the coordinates (s_l, t_l) (1) after the transformation are also stable.

Lemma 4.2 *For $PD^{(1)}$ and $PD^{(2)}$, we have*

$$||\boldsymbol{u} - \boldsymbol{v}||_p \le C_{p,m} W_p(PD^{(1)}, PD^{(2)}) \qquad 1 \le p \le \infty$$

where

$$\begin{aligned}\boldsymbol{u} &= (s_1^{(1)}, \cdots, s_k^{(1)}, t_1^{(1)}, \cdots, t_k^{(1)}),\\ \boldsymbol{v} &= (s_1^{(2)}, \cdots, s_k^{(2)}, t_1^{(2)}, \cdots, t_k^{(2)}).\end{aligned}$$

$C_{p,m}$ is a constant dependent on m in (1) and the p-norm.

The uniform cubic B-spline basis $B_*(\cdot)$ $(* = i,j)$ are defined on uniform knot sequences

$$\xi_* = \{0,0,0,0,\frac{1}{h-1},\cdots,\frac{h-2}{h-1},1,1,1,1\}, \tag{7}$$

for each $i,j = 1,\cdots,h$.

We give the explicit form of B-spline basis matrix $\boldsymbol{B}$ in equation (4). Each row of $\boldsymbol{B}$ can be written as the row vectors $[b_j(s,t)]$:

$$\begin{aligned}b(s,t) = [&B_1(s)B_1(t), B_1(s)B_2(t), \cdots, B_1(s)B_h(t),\\ &B_2(s)B_1(t), B_2(s)B_2(t), \cdots, B_2(s)B_h(t), \cdots,\\ &B_h(s)B_1(t), B_h(s)B_2(t), \cdots, B_h(s)B_h(t)]\\ :=& [B_1(s,t), B_2(s,t), \cdots, B_r(s,t)]\end{aligned}$$

$$\boldsymbol{B} = \begin{bmatrix} B_1(\cdot_1) & B_2(\cdot_1) & B_3(\cdot_1) & \cdots & B_r(\cdot_1) \\ B_1(\cdot_2) & B_2(\cdot_2) & B_3(\cdot_2) & \cdots & B_r(\cdot_2) \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ B_1(\cdot_k) & B_2(\cdot_k) & B_3(\cdot_k) & \cdots & B_r(\cdot_k) \end{bmatrix} \tag{8}$$

where $(\cdot_j)$ denotes (s_j, t_j), $j = 1, 2, \cdots, k$ and $r = h^2$.

We rewrite the iterative format (5) in matrix form,

$$\begin{aligned} \widetilde{\boldsymbol{Z}}^{(\beta+1)} &= \widetilde{\boldsymbol{Z}}^{(\beta)} + \Lambda \boldsymbol{B}^{\boldsymbol{T}}(\boldsymbol{Z} - \boldsymbol{B}\widetilde{\boldsymbol{Z}}^{(\beta)}) \\ \widetilde{\boldsymbol{Z}}^{(0)} &= \boldsymbol{O} \end{aligned} \tag{9}$$

where $\Lambda = diag(2/C, 2/C \cdots, 2/C)$ is a diagonal matrix, and $\boldsymbol{B}$ is the B-spline basis matrix shown above.

At present, we begin to discuss the stability of the persistence vectors computed by LSPIA method. According to the iterative format (9), we have an equivalent form

$$\widetilde{\boldsymbol{Z}}^{(\beta+1)} = (\boldsymbol{E} - \boldsymbol{\Lambda}\boldsymbol{B}^{\boldsymbol{T}}\boldsymbol{B})\widetilde{\boldsymbol{Z}}^{(\beta)} + \Lambda\boldsymbol{B}^{\boldsymbol{T}}\boldsymbol{Z},$$

We run the iteration for a finite number N to compute the persistence grids. Hence, we have

$$\widetilde{\boldsymbol{Z}}^{(N)} = [\sum_{i=0}^{N-1}(\boldsymbol{E} - \boldsymbol{\Lambda}\boldsymbol{B}^{\boldsymbol{T}}\boldsymbol{B})^i]\boldsymbol{\Lambda}\boldsymbol{B}^{\boldsymbol{T}}\boldsymbol{Z},$$

where $\boldsymbol{E}$ is indentity matrix.

Let

$$\bar{\boldsymbol{B}} := [\sum_{i=0}^{N-1}(\boldsymbol{E} - \boldsymbol{\Lambda}\boldsymbol{B}^{\boldsymbol{T}}\boldsymbol{B})^i]\boldsymbol{\Lambda}\boldsymbol{B}^{\boldsymbol{T}}.$$

We have the following lemma.

Lemma 4.3 *For each $(s_1, t_1, \cdots, s_k, t_k) \in [0,1]^{2k}$, every element in the matrix $\bar{\boldsymbol{B}}$ is a continuous and piecewise-C^1 rational function on the close interval $[0,1]^{2k}$.*

Finally, we have the stability of the persistence vectors generated by the control points of persistence B-spline surface.

Theorem 4.1 (Stability) *For two persistence diagrams $PD^{(1)}$ and $PD^{(2)}$ and their p-Wasserstein distance ($1 \leq p \leq \infty$, $p = \infty$ means the bottleneck distance) is given by $W_p(PD^{(1)}, PD^{(2)})$, then we have*

$$||\widetilde{\boldsymbol{Z}}^{(2)} - \widetilde{\boldsymbol{Z}}^{(1)}||_p \leq C_{m,N,\mathscr{E}} W_p(PD^{(1)}, PD^{(2)}) \qquad 1 \leq p \leq \infty$$

where $\widetilde{\boldsymbol{Z}}^{(1)}$ and $\widetilde{\boldsymbol{Z}}^{(2)}$ are the persistence vectors (the ordered z-coordinate sequences of the control points of the persistence B-spline surface) generated by $PD^{(1)}$ and $PD^{(2)}$ respectively. $C_{m,N,\mathscr{E}}$ is a constant dependent on the transformation (1), the iterations N of LSPIA, and the eminence function (3).

The proofs of lemmas and the theorem above are presented in the supplementary manuscript.

5. Experiments

In this section, we discuss the choice of the parameters of PB algorithm by the task of classification on a relatively simple synthetic data set sampled from five models in a normal 2D square. And we apply our PB on discrete dynamical system models and compare the performance of PBs, PIs, PLs and the direct computation of distances on PD space. In all of our experiments, we compute the H_0 and H_1 PDs by constructing the Vietoris-Rips filtration on each point cloud in the sense of Euclidean metric on $\mathbb{R}^2$ or $\mathbb{R}^3$. And the technique of the reduced homology is used to avoid the topological feature with infinite persistence occuring in H_0 PDs. All experiments run on a PC with Inter(R) Core(TM) i7-7700 CPU@3.60GHz×8.

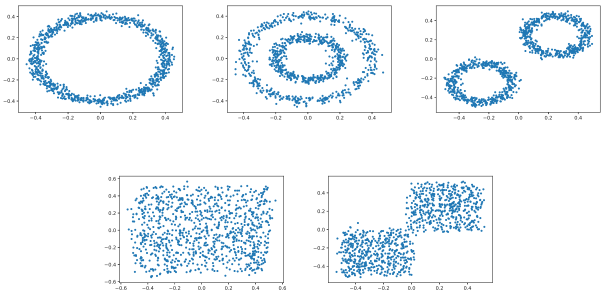

Figure 2. The sampling points from five classes of the toy data. Each sampling is perturbed by a Gaussian noise with $\eta = 0.025$.

5.1. Parameter Choice

We test our algorithm for a task of classification on a simple synthetic toy data set which contains a circle with its radius 0.4, two concentric circles with their radii 0.2 and 0.4 respectively, two disjoint circles with both their radii 0.2, a cluster of points sampled at random in the normal square and two clusters of points sampled at random separatively in two squares with edge length 0.5. All of data is sampled in the range of $[-0.5, 0.5]^2$ on the 2D plane. Each sampling forms an image (Figure 2). We obtain 50 images for each kind and sample 1000 points for each image. In this tests, we perturb the sampled data by adding Gaussian noises ($\eta = 0.025$) onto each image respectively.

In practice, PB relies on the iteration of LSPIA, the density of underlying control grid and the eminence function which indicate the importance of each point in a PD. We increase the iteration from 100 to 1000 with the increment 100 in the uniform control grid of 20 × 20. We compute the vectors in the different density of uniform control grid from the sparse grid (15 × 15) to the dense one (100 × 100) with the increment 5 in every coordinate. In both tests, we iterate the LSPIA for 100 times. The eminence function is considered

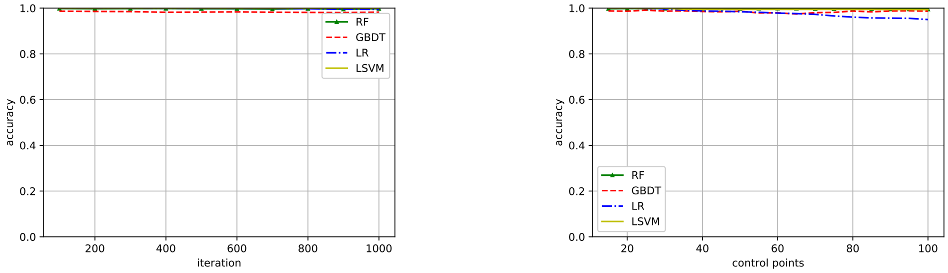

Figure 3. Classification accuracy of 4 methods with the increasing of iteration of LSPIA and density of control grid according to the H_1 features. Left figure: we increase the iteration from 100 to 1000 with the increment 100 in the uniform control grid of 20×20. Right figure: we increase the density of uniform control grid from a sparse grid (15×15) to a dense one (100×100) with the increment 5 in every coordinate.

in different tasks and left to be decided by users. In this task, we take $\varepsilon = 10^{-3}$ for computing the vectors from H_0 PDs, $\varepsilon = 10^{-10}$ for H_1 PDs and the bound constant $M = 10$. [2]

We use the following four classifiers: Random Forest (RF) (Breiman, 2001), Gradient Boosted Decision Trees (GBDT) (Friedman, 2001), Logistic Regression (LR) and Linear Support Vector Machine (LSVM) (Fan et al., 2008). They are powerful algorithms for supervised learning. RF and GBDT are ensemble learning algorithms which train multiple weak learners to obtain better classification performance. RF is a classifier that fits many decision trees on diffierent subsets of data and then uses averaging to overcome over-fitting. GBDT is one of boosting algorithms combing weak learners to a strong learner iteratively. We also apply logistic regression for multiclass classification using the one-vs-rest (OvR) scheme and linear support vector machine using one-vs-rest similarly. For PDs, we used a persistent homology package for Python, Ripser.py(Bauer, 2017). For each classifiers, we split data into 70% training set and 30% test set and perform 100 trials and compute the average classification accuracy.

Classifier	H_0 Average Accuracy	H_0 40×40 Accuracy	H_1 Average Accuracy	H_1 40×40 Accuracy
RF	93.6%	95.2%	99.4%	99.6%
GBDT	94.9%	95.7%	97.5%	96.9%
LR	91.3%	90.1%	98.3%	99.5%
LSVM	91.7%	94.0%	98.5%	99.5%

Table 1. The classification accuracy under 40×40 control grid and the average classification accuracy of different density of uniform control grids ranging from 15 to 100 with the increment 5. We show the classification performances using four classification methods for H_0 and H_1 PDs respectively.

As shown in Figure 3, in this task, PB is relatively insensitive to the iteration of LSPIA and the density of control grid. We observe that the classification accuracies keep stable when the density of gird is greater than 40×40. According to Table 1, we observe that the PB representation enable us to capture the topological feature of the data accurately and all of the classifiers reach high accuracies.

5.2. Classification of Parameters for Various Dynamical Systems

Dynamical system models can simulate some natural phenomena. In order to make the model approximate to the reality, several parameters in a dynamical system have to be determined. PD provides a visual tool for representing the potential topological features in dynamical data. Adams et al. (2017) offer an application of parameter determination of the linked twist map (Hertzsch et al., 2007), a 2D dynamical model. In this experiment, we test PLs, PIs, PBs and the direct computation of PD distances (PDs) in a 3D dynamical system to observe their performances on classifying more complicated data set.

The 3D dynamical system proposed by Lindström (Lindstrom, 2002) describes a discrete food chain model. The model is defined by

$$\begin{aligned} X_{t+1} &= \frac{M_0 X_t \exp(-Y_t)}{1 + X_t \max\{\exp(-Y_t), K(Z_t)K(Y_t)\}} \\ Y_{t+1} &= M_1 X_t Y_t \exp(-Z_t) K(Y_t) \cdot K(M_3 Y_t Z_t) \\ Z_{t+1} &= M_2 Y_t Z_t \end{aligned}$$

where

$$K(x) = \begin{cases} \dfrac{1-\exp(-x)}{x}, & x \neq 0 \\ 1 & x = 0 \end{cases}$$

The variables X, Y and Z are related to the trophic levels

[2] For the eminence function $\mathscr{E}(s,t)$ (3), we take $L = m$ in (2), and M equals to the maxium of the numbers of points of PDs. The parameter ε in (3) is selected in different tasks.

of food chain system. X and Z are proportional to vegetation abundance and carnivore abundance respectively, and Y has nonlinear and compilcated relation with herbivore abundance. Osipenko (2006) studies the attractor of the system with fixed parameters $M_1 = 1.0, M_2 = M_3 = 4.0$ and variable parameter $M_0 \in [3.00, 3.65]$. We refer to chapter 17 of (Osipenko, 2006) and take

$$M_0 = 3.0, 3.3, 3.48, 3.54, 3.57, 3.532, 3.571, 3.3701, 3.4001$$

respectively and obtain 9 classes of results with various parameter M_0. For each class, we generate the initial values (X_0, Y_0, Z_0) randomly in the region $(1, 2) \times (0, 1)^2$ and iterate the system for 2000 times to obatin the points in the 3D Euclidean space. And we repeat 50 times in each combination of parameters and eventually we get 450 results from the system. We do this experiment to compare the classification effects among the methods of PLs, PIs, PBs and PDs.

Time (s)	H_1, L_1	H_1, L_2	H_1, L_∞
PD	667s	1251s	983s
PL	32s	48s	**6s**
PI	**10s**	**10s**	10s
PB	**10s**	**10s**	10s

Table 2. The time cost (seconds) from inputting PDs to generating distance matrices for direct computation of 1-Wasserstein distance, 2-Wasserstein distance and the bottleneck distance (PDs, L_1; PD, L_2; PD, L_∞), PLs, PIs and PBs. We compute the representations of PLs, PIs and PBs and then their distance matrices. PBs are computed in the grid of 20 × 20.

Accuracy (%)	H_1, L_1	H_1, L_2	H_1, L_∞
PD	78.8±0.6%	76.8±0.6%	71.9±0.7%
PL	75.7±0.7%	73.4±0.6%	71.5±0.7%
PI	78.6±0.5%	80.9±0.6%	78.9±0.8%
PB	**82.8±0.4%**	**82.6±0.3%**	**81.0±0.3%**

Table 3. The classfication accuracies for direct computation of 1-Wasserstein distance, 2-Wasserstein distance and the bottleneck distance (PD, L_1; PD, L_2; PD, L_∞), PLs, PIs and PBs. PIs are computed with resolutions 20 × 20 and $\sigma = 10^{-3}$. And we take $\varepsilon = 10^{-1}$ in the eminence function (3) and use a 20 × 20 uniform control point grid in PB.

In the comparison, we use the software (Kerber et al., 2017) to compute the Wasserstein distances and the bottleneck distance and generate the distance matrix. For PLs, we employ the Persistence Landscapes Toolbox (Bubenik, 2015). For PIs, we use the open source MatLab codes (Adams et al., 2017). We compute the persistence images with their resolutions 20×20 and adjust the $\sigma = 10^{-3}$ in PI to guarantee the best result, and we take $\varepsilon = 10^{-1}$ in the eminence function and use a 20 × 20 uniform control point grid in our PB We employ the classifier of k-nearest neighbors (KNN) in this classification task. KNN is a simple classification algorithm that the distance metric on the space can be utilized. We create distance matrices according to L_1, L_2 or L_∞ distances [3]. And then we employ KNN to classify the data by using the distance matrices. Here we set $k = 3$ and split data randomly into 70% training set and 30% test set. We perform the classification for 10 times, in each of which we do 100 trials and compute the average classification accuracy.

H_1 **Accuracy**	**PI**	**PB**
k-nearest neighbors	80.9±0.6%	**82.6±0.3%**
Random Forest	79.0±0.6%	**83.2±0.7%**
Gradient Boosting Decision Tree	78.8±0.4%	**82.8±0.7%**
Logistic Regression	67.8±0.6%	**84.8±0.5%**
Linear Support Vector Machine	73.4±0.5%	**86.8±0.5%**

Table 4. Classification performance for H_1 PDs in 5 classifying approaches. We compute the PIs with their resolutions 20 × 20 and adjust the $\sigma = 10^{-3}$ in PI to guarantee the best result, and we take $\varepsilon = 10^{-1}$ in the eminence function and use a 20 × 20 uniform control point grid in PB.

As Table 2 and Table 3 shown, in the case of low density of the control grid in PB, PBs are almost as efficient as PIs in this task. We observe the classification accuracy with respect to 1, 2 and infinity norms of vector space. Our PB performs nice under the classifier of KNN. On the other hand, we also test our method under various classifers (RF, GB, LR and LSVM) comparing with the state-of-the-art PI. Though the parameters of PI are adjusted to guarantee the best performance, we observe that PB performs better (see Table 4) in this more complicated task of classifying 9 collections of this 3D dynamical system with different parameters. Inevitably, we select the parameters of PB (the eminence function) in the experiment, but the choice of parameters does not cost a lot of time (we just test $\varepsilon = 10^{-1}, 10^{-2}, 10^{-3}$ and pick up the best one).

6. Conclusion

We proposed a stable and efficient persistence B-spline surface with the technique of LSPIA. Our PB combines the classic B-spline basis in geometric design with current tools of persistent homology in topological data analysis, and offers a stable vector representation for numerous machine learning tools. The experiments show that our PB vectors generated by LSPIA are efficient, insensitive to the iteration of LSPIA, and mildly influenced by density of control grid. In the classification tasks, PB performs nice under five classifiers and is computed efficiently. We find that the PB vectors are sparse and point out that PB vectors enable one to reconstruct the persistence surfaces promptly. Related applications of these properties need exploring in future works.

[3] For PDs, they are W_1, W_2 and W_∞ distances. For vector representations, they are $1, 2, \infty$-norms of vector space. —

References

Adams, H., Emerson, T., Kirby, M., Neville, R., Peterson, C., Shipman, P., Chepushtanova, S., Hanson, E., Motta, F., and Ziegelmeier, L. Persistence images: A stable vector representation of persistent homology. *The Journal of Machine Learning Research*, 18(1):218–252, 2017.

Adcock, A., Carlsson, E., and Carlsson, G. The ring of algebraic functions on persistence bar codes. *Homology, Homotopy and Applications*, 18(1):381–402, 2016.

Bauer, U. Ripser: a lean c++ code for the computation of vietoris-rips persistence barcodes. *Software available at https://github. com/Ripser/ripser*, 2017.

Breiman, L. Random forests. *Machine learning*, 45(1): 5–32, 2001.

Bubenik, P. Statistical topological data analysis using persistence landscapes. *The Journal of Machine Learning Research*, 16(1):77–102, 2015.

Carriere, M., Oudot, S. Y., and Ovsjanikov, M. Stable topological signatures for points on 3d shapes. In *Computer Graphics Forum*, volume 34, pp. 1–12. Wiley Online Library, 2015.

Chazal, F., Fasy, B., Lecci, F., Michel, B., Rinaldo, A., and Wasserman, L. Subsampling methods for persistent homology. In *International Conference on Machine Learning*, pp. 2143–2151, 2015.

Chen, C. and Quadrianto, N. Clustering high dimensional categorical data via topographical features. In *International Conference on Machine Learning*, pp. 2732–2740, 2016.

Cohen-Steiner, D., Edelsbrunner, H., and Harer, J. Stability of persistence diagrams. *Discrete & Computational Geometry*, 37(1):103–120, 2007.

Cohen-Steiner, D., Edelsbrunner, H., Harer, J., and Mileyko, Y. Lipschitz functions have l p-stable persistence. *Foundations of computational mathematics*, 10(2):127–139, 2010.

De Silva, V. and Ghrist, R. Coverage in sensor networks via persistent homology. *Algebraic & Geometric Topology*, 7 (1):339–358, 2007.

Deng, C. and Lin, H. Progressive and iterative approximation for least squares b-spline curve and surface fitting. *Computer-Aided Design*, 47:32–44, 2014.

Di Fabio, B. and Ferri, M. Comparing persistence diagrams through complex vectors. In *International Conference on Image Analysis and Processing*, pp. 294–305. Springer, 2015.

Edelsbrunner, H. and Harer, J. *Computational topology: an introduction*. American Mathematical Soc., 2010.

Edelsbrunner, H., Letscher, D., and Zomorodian, A. Topological persistence and simplification. *Discrete Comput Geom*, 28:511–533, 2002.

Fan, R.-E., Chang, K.-W., Hsieh, C.-J., Wang, X.-R., and Lin, C.-J. Liblinear: A library for large linear classification. *Journal of machine learning research*, 9(Aug): 1871–1874, 2008.

Friedman, J. H. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pp. 1189–1232, 2001.

Ghrist, R. Barcodes: the persistent topology of data. *Bulletin of the American Mathematical Society*, 45(1):61–75, 2008.

Hatcher, A. Algebraic topology. 2002.

Hertzsch, J.-M., Sturman, R., and Wiggins, S. Dna microarrays: design principles for maximizing ergodic, chaotic mixing. *Small*, 3(2):202–218, 2007.

Kališnik, S. Tropical coordinates on the space of persistence barcodes. *Foundations of Computational Mathematics*, pp. 1–29, 2018.

Kerber, M., Morozov, D., and Nigmetov, A. Geometry helps to compare persistence diagrams. *Journal of Experimental Algorithmics (JEA)*, 22:1–4, 2017.

Khrulkov, V. and Oseledets, I. V. Geometry score: A method for comparing generative adversarial networks. *international conference on machine learning*, pp. 2621–2629, 2018.

Kusano, G., Fukumizu, K., and Hiraoka, Y. Kernel method for persistence diagrams via kernel embedding and weight factor. *Journal of Machine Learning Research*, 18(189):1–41, 2018.

Le, T. and Yamada, M. Persistence fisher kernel: A riemannian manifold kernel for persistence diagrams. In *Advances in Neural Information Processing Systems*, pp. 10027–10038, 2018.

Lin, H., Cao, Q., and Zhang, X. The convergence of least-squares progressive iterative approximation with singular iterative matrix. *arXiv preprint arXiv:1707.09109*, 2017.

Lindstrom, T. On the dynamics of discrete food-chains: Low- and high-frequency behavior and optimality of chaos. *Journal of Mathematical Biology*, 45(5):396–418, 2002.

Mileyko, Y., Mukherjee, S., and Harer, J. Probability measures on the space of persistence diagrams. *Inverse Problems*, 27(12):124007, 2011.

Osipenko, G. *Dynamical systems, graphs, and algorithms*. Springer, 2006.

Pachauri, D., Hinrichs, C., Chung, M. K., Johnson, S. C., and Singh, V. Topology-based kernels with application to inference problems in alzheimer's disease. *IEEE transactions on medical imaging*, 30(10):1760–1770, 2011.

Reininghaus, J., Huber, S., Bauer, U., and Kwitt, R. A stable multi-scale kernel for topological machine learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4741–4748, 2015.

Rieck, B., Fugacci, U., Lukasczyk, J., and Leitte, H. Clique community persistence: A topological visual analysis approach for complex networks. *IEEE transactions on visualization and computer graphics*, 24(1):822–831, 2018.

Rouse, D., Watkins, A., Porter, D., Harer, J., Bendich, P., Strawn, N., Munch, E., DeSena, J., Clarke, J., Gilbert, J., et al. Feature-aided multiple hypothesis tracking using topological and statistical behavior classifiers. In *Signal processing, sensor/information fusion, and target recognition XXIV*, volume 9474, pp. 94740L. International Society for Optics and Photonics, 2015.

Xia, K. and Wei, G.-W. Multidimensional persistence in biomolecular data. *Journal of computational chemistry*, 36(20):1502–1520, 2015.

Xia, K., Li, Z., and Mu, L. Multiscale persistent functions for biomolecular structure characterization. *Bulletin of mathematical biology*, 80(1):1–31, 2018.

Zomorodian, A. and Carlsson, G. Computing persistent homology. *Discrete & Computational Geometry*, 33(2): 249–274, 2005.