J Appl. and Comput. Topology (2018) 1:421–449
https://doi.org/10.1007/s41468-018-0013-5

Persistence diagrams with linear machine learning models

Ippei Obayashi[1] · **Yasuaki Hiraoka**[2,3,4] · **Masao Kimura**[5,6]

Received: 6 July 2017 / Accepted: 16 April 2018 / Published online: 5 May 2018

Abstract Persistence diagrams have been widely recognized as a compact descriptor for characterizing multiscale topological features in data. When many datasets are available, statistical features embedded in those persistence diagrams can be extracted by applying machine learnings. In particular, the ability for explicitly analyzing the inverse in the original data space from those statistical features of persistence diagrams is significantly important for practical applications. In this paper,

This work is partially supported by JSPS KAKENHI Grant Number JP 16K17638, JST CREST Mathematics15656429, JST "Materials research by Information Integration" Initiative (MI^2I) project of the Support Program for Starting Up Innovation Hub, Structural Materials for Innovation Strategic Innovation Promotion Program D72 and D66, and New Energy and Industrial Technology Development Organization (NEDO).

✉ Ippei Obayashi
ippei.obayashi.d8@tohoku.ac.jp

Yasuaki Hiraoka
hiraoka.yasuaki.6z@kyoto-u.ac.jp

Masao Kimura
masao.kimura@kek.jp

[1] Advanced Institute for Materials Research (WPI-AIMR), Tohoku University, 2-1-1 Katahira, Aoba-ku, Sendai 980-8577, Japan

[2] Kyoto University Institute for Advanced Study, Kyoto University, Yoshida Ushinomiya-cho, Sakyo-ku, Kyoto 606-8501, Japan

[3] Center for Advanced Intelligence Project, RIKEN, Wako, Japan

[4] Center for Materials research by Information Integration (CMI2), National Institute for Materials Science (NIMS), Tsukuba, Japan

[5] Photon Factory, Institute of Materials Structure Science, High Energy Accelerator Research Organization, Tsukuba, Japan

[6] Department of Materials Structure Science, School of High Energy Accelerator Science, SOKENDAI (The Graduate University for Advanced Studies), Tsukuba, Japan

we propose a unified method for the inverse analysis by combining linear machine learning models with persistence images. The method is applied to point clouds and cubical sets, showing the ability of the statistical inverse analysis and its advantages.

Keywords Topological data analysis · Persistent homology · Machine learning · Linear models · Persistence image

Mathematics Subject Classification 55-04 · 55U99 · 62P35 · 62J07

1 Introduction

Given a dataset, its statistical features can be extracted by applying machine learning methods (Bishop 2007). Needless to say, machine learning is now one of the central scientific and engineering subjects, and is rapidly enlarging its theoretical foundations and ranges of practical applications. For example, in materials science, the amount of available data has recently been increasing due to improvement of experimental methods and computational resources. These datasets are expected to be used for further developments of high performance materials based on machine learnings, leading to a new concept called "materials informatics" (Rajan 2005, 2012; Buchet et al. 2018).

As another branch of data science, topological data analysis (TDA) (Carlsson 2009; Edelsbrunner and Harer 2010) has also been rapidly developed from theoretical aspects to applications in the last decade. In TDA, persistent homology and its persistence diagram (Edelsbrunner et al. 2002; Zomorodian and Carlsson 2005) are widely used for capturing multiscale topological features in data. Recent improvements of efficient computations of persistence diagrams (Bauer et al. 2014, 2017) enable us to apply them into practical problems such as materials science (Hiraoka et al. 2016; Saadatfar et al. 2017; Ichinomiya et al. 2017; Kimura et al. 2017), sensor networks (de Silva and Ghrist 2007), evolutions of virus (Chen et al. 2013) etc. As a descriptor of data, persistence diagrams have the following significant properties: translation and rotation invariance, and robustness for noise. Persistence diagrams are also multi-scalable, that is, persistence diagrams can capture the geometric structures in multiple length scales. Together with developments of statistical foundations (Bubenik 2015; Chazal et al. 2015; Fasy et al. 2014; Kusano et al. 2016, 2017; Reininghaus et al. 2015; Turner et al. 2014; Robins and Turner 2016), persistence diagrams nowadays have been recognized as a compact descriptor for complicated data.

In a series of works on materials TDA (Hiraoka et al. 2016; Saadatfar et al. 2017; Ichinomiya et al. 2017; Kimura et al. 2017), analyzing the inverse in the original data space (atomic configurations or digital images) from persistence diagrams is significantly important to explicitly study the materials structures and properties. Therefore, toward further progress that materials TDA incorporates with materials informatics, we need to develop a framework of machine learnings on persistence diagrams which allows the inverse analysis.

In this paper, we propose a unified method for studying the shape of data by using persistence diagrams with machine learnings in both direct and inverse problems. The essence of our method is to combine persistence images (Adams et al. 2017) and linear machine learning models.

For standard machine learning methods, the input data is supposed to be given by vectors, and therefore we need to transform persistence diagrams into vectors. Some vectorization methods of persistence diagrams have been proposed in the literatures (Adams et al. 2017; Bubenik 2015; Kusano et al. 2016, 2017; Reininghaus et al. 2015), and we here use persistence images. This is because it allows us to reconstruct persistence diagrams from vectors obtained by machine learning results, providing a key step in the inverse route of our analysis.

Taking this advantage, we apply linear models of machine learnings to persistence images. Since the learned result of linear machine learning models is given by a (dual) vector with the same dimension as input vectors, we can reconstruct the persistence diagram from the learned result by simply reversing the construction process of persistence images. Namely, the persistence diagram itself is obtained as learning. Furthermore, by studying inverse problems from the reconstructed (dual) persistence diagram to the original data space, we can explicitly characterize statistically significant topological features embedded in data. In this paper, we deal with an inverse problem studying the locations of birth-death pairs of persistence diagrams in the original data space. As another advantage using linear machine learning models, we also propose an important concept called sparse persistence diagram. This new concept allows us to discard irrelevant generators and to focus on most significant ones in the reconstructed persistence diagram for learning tasks.

It should be remarked that, for only direct problems such as predictions from data, nonlinear methods such as kernel methods and neural networks are possibly appropriate, because such nonlinear transformations often make the prediction performance better than linear models. However, if our interest is to understand mechanisms of data structures, the inverse route going back to the original data from the learned results is inevitable.

As summary, the contribution of this paper is to propose a unified method in topological data analysis with the ability to study inverse problems by combining the following methods:

1. Persistence images.
2. Linear machine learning models.
3. Inverse analysis of persistence diagrams.

In Sect. 2, after brief introduction of our input data formats and persistent homology, we recall persistence images and linear models of machine learnings used in this paper. In Sect. 3, our method is demonstrated to some problems on point clouds and cubical sets and its performance is compared to other methods. Here, in addition to the synthetic data, we also apply the method to a practical problem in materials science; geometric characterization of heterogeneous chemical reactions in the iron ore sinter. Some future problems and related topics are summarized in Sect. 4.

2 Methods

We first explain some preliminaries about geometric models and persistent homology. Although the theory of persistent homology has been rapidly extended in various general settings, we here introduce the minimum necessary for later discussions. Readers who want to understand the theory in higher generality are encouraged to study the latest literatures.

2.1 Geometric models

In this paper, we mainly consider two types of input data. The first type is given by a finite points $P = \{x_i \in \mathbb{R}^N : i = 1, \dots, m\}$ in a Euclidean space $\mathbb{R}^N$, which is also called a *point cloud* in TDA. For example, this data type is frequently used for expressing atomic configurations in materials science.

Our interest is to characterize multiscale topological properties in P, and to this aim, we consider the r-ball model

$$P_r = \bigcup_{i=1}^{m} B_r(x_i),$$

where $B_r(x_i) = \{y \in \mathbb{R}^N : ||y - x_i|| \leq r\}$ is the ball with radius r centered at x_i. By construction, when the radius r is very small (resp. large), P_r has the same topology as m disconnected points (resp. one point). Between these two extremal cases, P_r may exhibit appearance and disappearance of holes by changing the radius r. Note that we have a natural inclusion $P_r \subset P_s$ for $r \leq s$, meaning that the radius parameter r can be regarded as a resolution of the point cloud P.

For practical data analysis, the r-ball model P_r is not convenient to handle in computers, and hence we usually build simplicial complex models from P_r. For instance, the *Čech complex* Čech(P, r) and the *Rips complex* (or Vietoris-Rips complex) Rips(P, r) are simplicial complexes with the vertex set P whose k-simplex is assigned by the following rule, respectively,

$$\{x_{i_0}, \dots, x_{i_k}\} \in \check{\text{C}}\text{ech}(P, r) \Leftrightarrow \bigcap_{s=0}^{k} B_r(x_{i_s}) \neq \emptyset,$$

$$\{x_{i_0}, \dots, x_{i_k}\} \in \text{Rips}(P, r) \Leftrightarrow B_r(x_{i_s}) \cap B_r(x_{i_t}) \neq \emptyset,\ 0 \leq \forall s < \forall t \leq k.$$

Note that, by construction, both simplicial complex models naturally define a (right continuous) *filtration*. Namely, for $X_r = \check{\text{C}}\text{ech(P,r)}$ or $X_r = \text{Rips}(P, r)$, it satisfies $X_r \subset X_s$ for $r \leq s$ and $X_s = \bigcap_{s<t} X_t$. In this section, we denote the filtration by $\mathbb{X} = \{X_r : r \in \mathbb{R}\}$.

Our next data type is given by a cubical set, which is a standard mathematical expression for digital images. Following the notation used in the reference (Kaczynski et al. 2004), let $I \subset \mathbb{R}$ be an elementary interval, i.e.,

$$I = [\ell, \ell + 1] \quad \text{or} \quad I = [\ell, \ell]$$

for some $\ell \in \mathbb{Z}$. An elementary cube $Q = I_1 \times \cdots \times I_N \subset \mathbb{R}^N$ is defined by a product of elementary intervals I_i. Then, a subset $X \subset \mathbb{R}^N$ is said to be cubical if X can be expressed as a union of elementary cubes in $\mathbb{R}^N$.

Let us denote by $\mathcal{K}_W^N$ the set of all elementary cubes in the window $\Lambda_W = [-W, W]^N \subset \mathbb{R}^N$. Given a function $f : \mathcal{K}_W^N \to \mathbb{R}$, we can build a cubical set in Λ_W as a sublevel set

$$X_t = \bigcup \{Q \in \mathcal{K}_W^N : f(Q) \leq t\} \tag{1}$$

for each parameter t. In practical applications such as digital image analysis, this function is often given by the Manhattan distance (e.g., see Fig. 2) or a gray-scale function. It is easy to see that the cubical sets X_t also lead to a filtration $\mathbb{X} = \{X_t : t \in \mathbb{R}\}$.

These are the two standard types of our input data. We note that those filtrations satisfy the properties that $X_t = \emptyset$ for sufficiently small t and X_t is acyclic[1] for sufficiently large t, respectively.

2.2 Persistent homology

Let $\Bbbk$ be a field. In this paper, the qth homology $H_q(X)$ of a topological space X is defined over the field $\Bbbk$, and hence $H_q(X)$ is given as a $\Bbbk$-vector space. Intuitively, the dimension of $H_q(X)$ as a $\Bbbk$-vector space counts the number of q-dimensional holes in X, and each basis vector expresses the corresponding q-dimensional hole in X, where, for example, $q = 0, 1, 2$ express connected components, rings, and cavities, respectively. Then, given a pair of topological spaces $X \hookrightarrow Y$, we can define the induced linear map $\varphi : H_q(X) \to H_q(Y)$, which characterizes whether a hole in X persists in Y or not.

The input to the persistent homology is given by a filtration $\mathbb{X} = \{X_t : t \in \mathbb{R}\}$ of topological spaces. In this paper, X_t is given by a simplicial complex or a cubical set. For simplicity, we also assume the properties for filtrations remarked in the final paragraph in Sect. 2.1, although we do not really need it by modifying the argument here. Then, the qth *persistent homology* $H_q(\mathbb{X}) = (H_q(X_t), \varphi_s^t)$ of the filtration $\mathbb{X}$ is defined by the family of homologies $\{H_q(X_t) : t \in \mathbb{R}\}$ and the induced linear maps $\varphi_s^t : H_q(X_s) \to H_q(X_t)$ for all $s \leq t$.

Under the assumption of our filtrations, the persistent homology $H_q(\mathbb{X})$ can be uniquely decomposed by using the so-called interval representations:

[1] A topological space X with $\tilde{H}_q(X) = 0$ for any q is called acyclic, where $\tilde{H}_q(X)$ is the reduced homology of X.

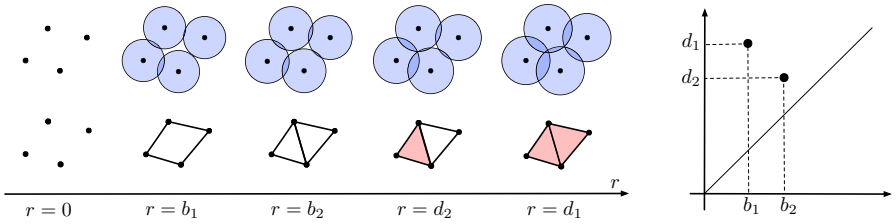

Fig. 1 Left top: Filtration of the r-ball models. Left bottom: Filtration of the corresponding Čech complexes. Right: The 1st persistence diagram. (1) A ring is bone at $r = b_1$. (2) Another ring is bone at $r = b_2$. (3) The second ring dies at $r = d_2$. (4) The first ring dies at $r = d_1$

$$H_q(\mathbb{X}) \simeq \bigoplus_{i=1}^{p} I(b_i, d_i), \tag{2}$$

where $b_i, d_i \in \mathbb{R}$ with $b_i < d_i$. Here, $I(b_i, d_i) = (U_t, f_s^t)$ consists of a family of vector spaces

$$U_t = \begin{cases} \Bbbk, & b_i \le t < d_i, \\ 0, & \text{otherwise}, \end{cases}$$

and the identity map $f_s^t = \mathrm{id}_{\Bbbk}$ for $b_i \le s \le t < d_i$. Note that the 0th persistent homology in (2) is understood as the reduced sense, meaning that one connected component which persists for any large $t \in \mathbb{R}$ is removed. Each interval representation $I(b_i, d_i)$ is also called a generator of $H_q(\mathbb{X})$.

Each generator $I(b_i, d_i)$ expresses that a q-dimensional hole appears in $\mathbb{X}$ at the parameter $t = b_i$, persists up to $t < d_i$, and then disappears at $t = d_i$. We call $b_i, d_i, d_i - b_i$ the birth time, death time, and lifetime of $I(b_i, d_i)$, respectively.

Under the unique decomposition (2), the qth persistence diagram $D_q(\mathbb{X})$ of $\mathbb{X}$ is defined by a multiset[2]

$$D_q(\mathbb{X}) = \{(b_i, d_i) \in \Delta : i = 1, \dots, p\},$$

where $\Delta = \{(b, d) \in \mathbb{R}^2 : b < d\}$. It is known that the birth-death pair $(b_i, d_i) \in D_q(\mathbb{X})$ with large lifetime can be regarded as reliable topological structure in $\mathbb{X}$, while that with small lifetime is likely to be a noisy structure. This statement is justified by the stability theorem of persistent homology (Cohen-Steiner et al. 2007).

For a review about computational aspect of persistent homology, we refer the readers to the paper (Otter et al. 2017).

[2] A multiset is a set with multiplicity of each point.

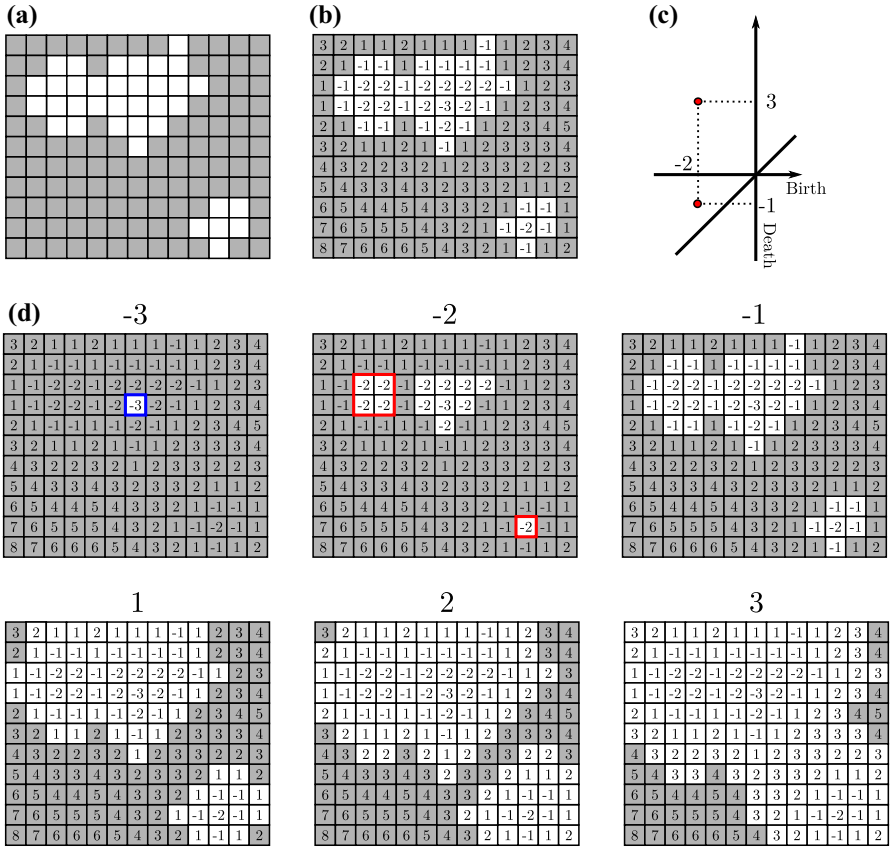

Fig. 2 **a** Input binary image. **b** Manhattan distance. **c** 0th reduced persistence diagram. **d** Filtration of binary images with respect to the Manhattan distance. The colored squares in **d**:-3 and **d**:-2 indicate the initial locations of three connected components. The blue square in **d**:-3 indicates the connected component removed in the reduced persistent homology (color figure online)

2.3 Examples

Here, we show several examples to make clear the concepts explained so far. To this aim, the examples are chosen to be simple enough for demonstration.

We first consider an example of a point cloud given by four points on the plane shown in the left ($r = 0$) of Fig. 1. As we explained, each point is replaced by a ball and we study topological changes during the fattening process of the balls by increasing the radii. This fattening process is drawn on the left top of Fig. 1, while the sequence below expresses its Čech complex filtration.

At the radius $r = b_1$, the first ring is born, and we record its birth parameter as b_1. Similarly, the second ring appears at the birth parameter $r = b_2$. On the other hand, at radius $r = d_1, d_2$, those rings disappear and we record them as their death

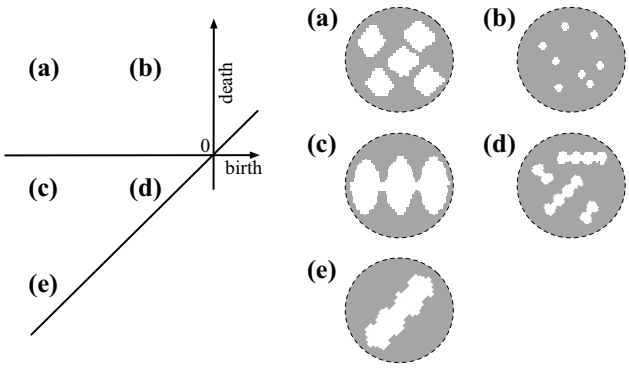

Fig. 3 Cubical sets drawn in the dashed circles (**a**–**e**) express typical geometric structures in the 0th persistence diagram. **a** Large islands. **b** Small islands. **c** Large islands with narrow bridges. **d** Narrow bands. **e** Broad bands. For **a** and **b**, the births correspond to the radii of the islands. For **d** and **e**, the births and deaths correspond to the half widths of the bands

parameters. Hence, the 1st persistence diagram of the Čech complex filtration is given by $\{(b_1, d_1), (b_2, d_2)\}$, which is shown on the right of Fig. 1.

Next, we consider an example of a cubical set. The input data is given by a binary image (a) in Fig. 2, and we consider a function f assigning an integer on each pixel shown in (b). Here, positive (resp. negative) numbers are assigned to the gray (resp. white) pixels using the Manhattan distance. The function f is called the signed distance function or signed distance transform with Manhattan distance. Then, following the construction (1) of sublevel sets, we obtain a filtration of cubical sets of white pixels shown in (d). The signed distance transform is already used by various digital image analysis, including TDA (Delgado-Friedrichs et al. 2014, 2015; Robins et al. 2016). These researches used the signed distance transform with Euclidean metric, but we use Manhattan metric as an approximation of Euclidean metric for easy handling of the singed distance transform on a computer.

In this example, three connected components appear in the filtration and those birth events are colored in blue and red. The death of those generators corresponds to a merging to another connected component. Then, the 0th reduced persistence diagram is given by $\{(-2, -1), (-2, 3)\}$, shown in (c). Note that the first connected component born at -3 is removed in the reduced persistence diagram. We also note that, from the assignment f using Manhattan distance, all birth parameters take negative values. Figure 3 summarizes typical geometric structures captured by the 0th persistence diagram based on the Manhattan distance.

For deep analysis using persistence diagrams, we often want to know the origin of each birth-death pair. One easy and useful way is to utilize a death simplex (resp. death cube) for a point cloud (resp. cubical set). In the Čech filtration model of Fig. 1, two generators (i.e., rings) die when each red simplex fills the corresponding ring, and those simplices show the locations of the generators. We call these locations *death positions* of the generators. Even for generators with higher dimensions and also for the setting of cubical sets, this idea works in a similar way.

On the other hand, for generators with dimension zero, birth events may possess the information of locations. In Fig. 2d, there are two red squares and they express the central locations of each connected component. We call these locations *birth positions* of the corresponding birth-death pairs.

We note that the birth/death positions are easily obtained in standard algorithms of computing persistence diagrams, and hence no additional computations are required. These techniques, which will be demonstrated in the later section, are exploited for some practical analysis in materials science (Kimura et al. 2017; Robins et al. 2016).[3] It should be remarked that, if one wants to obtain further information about the inverse of birth-death pairs, the technique of optimal cycles (Dey et al. 2011; Escolar and Hiraoka 2016) can be another choice, although it requires a much computer resource.

2.4 Persistence images

Recall that a persistence diagram is a multi-set on $\mathbb{R}^2$. Hence, we need to vectorize persistence diagrams to apply machine learning models. In this paper, we use the *persistence image* (Adams et al. 2017) for vectorization.

Given a qth persistence diagram $D_q = \{(b_k, d_k) \in \Delta : k = 1, \dots, \ell\}$, the persistence image ρ is defined by a function on $\mathbb{R}^2$ as

$$\begin{aligned} \rho(x, y) &= \sum_{k=1}^{\ell} w(b_k, d_k) \exp\left(-\frac{(b_k - x)^2 + (d_k - y)^2}{2\sigma^2} \right), \\ w(b, d) &= \arctan(C(d - b)^p). \end{aligned} \tag{3}$$

Here, $C > 0$, $p > 0$, $\sigma > 0$ are parameters, $w(b, d)$ is a weight function, and we regard the function ρ as a vector in a function space $L^2(\mathbb{R}^2)$. We remark that the weight function is chosen so that we can respect the significance of generators according to its lifetimes in the statistical analysis. As we see in Sect. 3, the parameters are usually determined to be appropriate values using cross validations.

For computations, we discretize the persistence image ρ and construct a histogram on the plane with an appropriate finite mesh. Obviously, since all birth-death pairs are located in $\{(b, d) \in [b_-, b_+] \times [d_-, d_+] : b < d\}$ with some constants b_-, b_+, d_-, d_+, the histogram is constructed on this area. Then, we obtain a vector from the discretization of ρ by ordering the elements on the grids in a prefixed order. Note that the dimension of the vector is equal to the number of grids used for the histogram. In the following, we also call the discretization of ρ the persistence image. See Algorithm 1 for the explicit algorithm of this construction.

[3] In Robins et al. (2016), the birth/death positions are called critical points.

Algorithm 1 Computation of a discretized persistence image

input: a persistence diagram D_q, parameters $C, p, \sigma, b_-, b_+, d_-, d_+, N_b, N_d$

1. Compute a histogram of D_q on $[b_-, b_+] \times [d_-, d_+]$ with $N_b \times N_d$ grids.
2. For each grid at (b, d), multiply the weight value $w(b, d)$ to the value in the grid
3. Apply Gaussian blur with the standard deviation σ (σ should be adjusted by the grid size)
4. Order the histogram values of the elements on the grids in a prefixed order and regard the ordered values as a vector

We note that there are several methods for vectorizations of persistence diagrams. One important advantage using persistence images is that we can easily reconstruct a histogram from a vector, and hence can obtain a corresponding persistence diagram. However, it is not straightforward in general to reconstruct persistence diagrams from vectors in nonlinear vectorizations. This advantage is effectively used in our method.

We also remark that, precisely speaking, the weight function (3) is not used in the original paper (Adams et al. 2017) but first studied in the paper (Kusano et al. 2017), in which performance comparisons with different weights for persistence images and also with other vectorizations are thoroughly discussed. For details, we refer the readers to the paper (Kusano et al. 2017).

2.5 Linear machine learning models

In this section, we briefly recall the logistic regression and the linear regression as standard supervised machine learning methods (Bishop 2007).

In the linear regression model, we consider a pair of an input vector $x \in \mathbb{R}^n$ (called *explanatory variable*) and its output value $y \in \mathbb{R}$ (called *response variable*), and study the relation between them in the linear form

$$y = w \cdot x + b + (\text{noise}),$$

where $w \in \mathbb{R}^n$ and $b \in \mathbb{R}$ are unknown parameters and the noise is randomly determined from a normal distribution with mean 0. From a set of known input–output pairs $\{(x_i, y_i)\}_{i=1}^M$, called a *training set*, we find an optimal w and b for the model. Such optimal parameters are derived by minimizing the following *mean squared loss error function* with respect to w and b:

$$E(w, b) = \frac{1}{2M} \sum_{i=1}^{M} (w \cdot x_i + b - y_i)^2. \tag{4}$$

In the logistic regression model for a binary classification task, we consider a pair of an input vector $x \in \mathbb{R}^n$ and its output value $y \in \{0, 1\}$, and study the relation of classification 0/1 based on the following form

$$\begin{aligned} P(y=1 \mid w,b) &= g(w \cdot x + b), \\ P(y=0 \mid w,b) &= 1 - P(y=1 \mid w,b) = g(-w \cdot x - b), \\ g(z) &= 1/(e^{-z}+1), \end{aligned} \tag{5}$$

where $w \in \mathbb{R}^n$ and $b \in \mathbb{R}$ are unknown parameters. From training data $\{(x_i, y_i)\}_{i=1}^M$, we find an optimal w and b in a similar way to the linear regression. Here, optimal parameters are given by minimizing the following *cross entropy error function*:

$$\begin{aligned} L(w,b) &= -\frac{1}{M}\sum_{i=1}^M \left\{ y_i \log \hat{y}_i + (1-y_i)\log(1-\hat{y}_i) \right\}, \\ \hat{y}_i &= g(w \cdot x_i + b). \end{aligned} \tag{6}$$

We note that, for both the linear regression and logistic regression, these optimization problems are equivalent to the maximization of the log likelihood.

In our method, the input vector $x \in \mathbb{R}^n$ is given by vectorized persistence diagrams using persistence images. Then, the learned vector w becomes a dual vector to the persistence images, and especially, its dimension is the same as x. Hence, w can be expressed as a (dual) persistence diagram by the reverse process of the vectorization using persistence images. In this way, our method outputs persistence diagrams as learning results.

For practical applications, we often encounter the problem of over-fittings, if the dimension n of input vectors is relatively large compared to the sample size (the number of datasets) M. Under this condition, the result of the optimization problem excessively fits the training set and does not give appropriate performance for untrained data. In our setting, since the dimension of vectors obtained from persistence images is very large, we usually face the over-fitting problem. The vectors given by persistence images also have another statistical problem called multicollinearity (Bingham 2010). Adjacent grids elements of a vector by persistence image are strongly correlated because of Gaussian diffusion, and such a strong correlation causes the difficulty of determining coefficients and the numerical instability.

One effective way for avoiding the over-fitting and multicollinearity is to add a regularization (penalty) term $R(w)$ into the error function. Namely, we minimize the following modified error functions for w and b

$$\begin{aligned} &E(w,b) + \lambda R(w) \ \text{(for a linear regression)}, \\ &L(w,b) + \lambda R(w) \ \text{(for a logistic regression)}, \end{aligned}$$

where $\lambda > 0$ is a weight parameter controlling the regularization effect. Typical regularization terms are given as

$$R(w) = \frac{1}{2}\|w\|_2^2, \quad R(w) = \|w\|_1.$$

The former is called an ℓ^2-regularization and the latter is called an ℓ^1-regularization. A linear regression with the ℓ^2-regularization is called ridge, while a linear regression with the ℓ^1-regularization is called lasso (Robert 1996).

The advantage of the ℓ^2-regularization is its good mathematical property. For example, the ℓ^2-regularization term is differentiable but the ℓ^1-regularization term is not. The ridge optimization problem has the closed form solution. However, the lasso does not have such forms.

On the other hand, the ℓ^1-regularization has a significant property of the *sparsity*. A vector w is called sparse if its elements are all zero except for only a few elements. It is well-known that the learned vector w under ℓ^1-regularization becomes a sparse vector, and hence we obtain a *sparse persistence diagram* as a result of learning. As we will see later, the sparseness of the learned persistence diagram is often very useful, when we interpret the learned results.

The parameter λ of the regularization term controls the complexity of the learned result (Bishop 2007). When the weight λ becomes larger, the regularization term $R(w)$ becomes smaller. This means that w becomes more sparse in the ℓ^1-regularization. Such a reduction of the complexity is useful for finding the most essential elements for regressions. However, when λ is too large, the learned results may drop important information. Therefore, we need to determine a suitable λ in practice. A validation set or cross validation method are often applied to choose such a parameter (Bishop 2007). The effect of changing λ in our method is discussed in Sect. 3.

2.6 Summary of our method

1. Prepare an input data $\{(g_i, y_i)\}_{i=1}^M$. Here, each g_i is a point cloud or a digital image, and y_i is a real value for the linear regression or 0/1 value for the logistic regression.
2. Compute the persistence diagram $D^{(i)}$ from g_i.
3. Compute the vectorization $x_i \in \mathbb{R}^n$ of $D^{(i)}$ using the persistence image.
4. Apply the linear regression or the logistic regression with a regularization term to the data $\{(x_i, y_i)\}_{i=1}^M$ and find $w \in \mathbb{R}^n$ and $b \in \mathbb{R}$. Choose the ℓ^2- or ℓ^1-regularization, depending on the purpose.
5. The learned result w is visualized by the reconstruction of the persistence diagram from w. From the reconstructed dual persistence diagram, one may extract important areas on the diagram.
6. For explicitly identifying the geometric structure of those important areas on the diagram, one can study the birth/death positions.

3 Results and discussions

In this section, we demonstrate the performance of our methods for logistic regressions and linear regressions with binary images and point clouds. Here, we use filtrations of cubical sets using Manhattan distance (black: positive, white: negative) for binary images, while alpha complex filtrations,[4] which are homotopic to Čech

[4] CGAL: https://www.cgal.org/ (Da. et al. 2017).

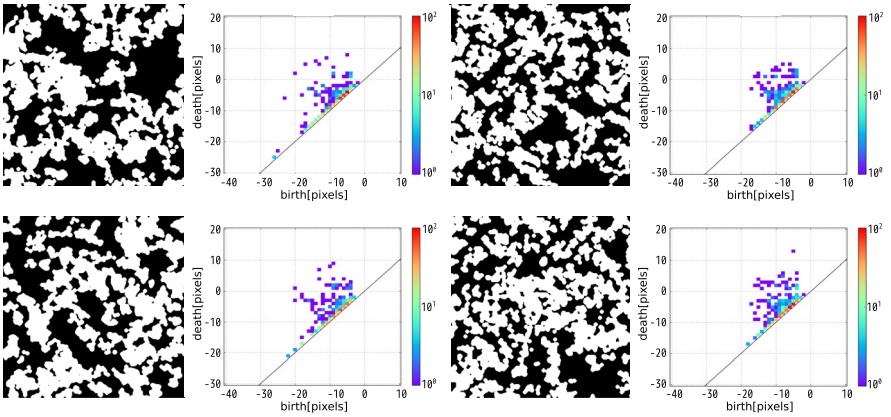

Fig. 4 Input binary images and their 0th persistence diagrams. The left and right two images are sampled from the parameter pairs (**A**) and (**B**), respectively

complex filtrations, are applied for point clouds. All examples are experimented using scikit-learn[5] and HomCloud.[6] Here, let us summarize the details of these softwares used in our analysis in the following.

In scikit-learn, LogisticRegression and LogisticRegressionCV classes in sklearn.linear_model module are used for the logistic regression, while Lasso, LassoCV, Ridge and RidgeCV classes in sklearn.linear_model module are used for the linear regression. Here, we remark that these classes automatically determine the weight of the regularization term by using cross validations, and we follow the default cross validation strategies of scikit-learn (stratified threefolds for logistic regressions, leave-one-out for ridge, and threefolds for lasso).

In HomCloud, persistence diagrams are computed using DIPHA.[7] Manhattan distance for cubical filtrations are computed using the distance transform function provided by scipy's ndimage module[8] To identify birth/death positions, we assign the indices to all pixels sorted by the Manhattan distance in increasing order. Then, the birth-death pairs computed in DIPHA are expressed by the indices, and we can easily identify the corresponding pixel of each birth/death position from its index.

3.1 Logistic regression on binary images—an easy example

First, we examine the logistic regression on persistence diagrams of binary images. Here, the binary image data is randomly generated by Algorithm 2 in Appendix 1, where a pair of parameters (N, S) is used to generate two types of images. One pair (A) is set to be $N = 100, S = 30$ and the other (B) is $N = 250, S = 10$. Figure 4

[5] Scikit-learn: http://scikit-learn.org/ (Pedregosa et al. 2011).

[6] http://www.wpi-aimr.tohoku.ac.jp/hiraoka_labo/homcloud/index.en.html.

[7] DIPHA: A Distributed Persistent Homology Algorithm (Bauer et al. 2014).

[8] SciPy: Open Source Scientific Tools for Python, 2001-, http://www.scipy.org/ (Jones et al. 2011–).

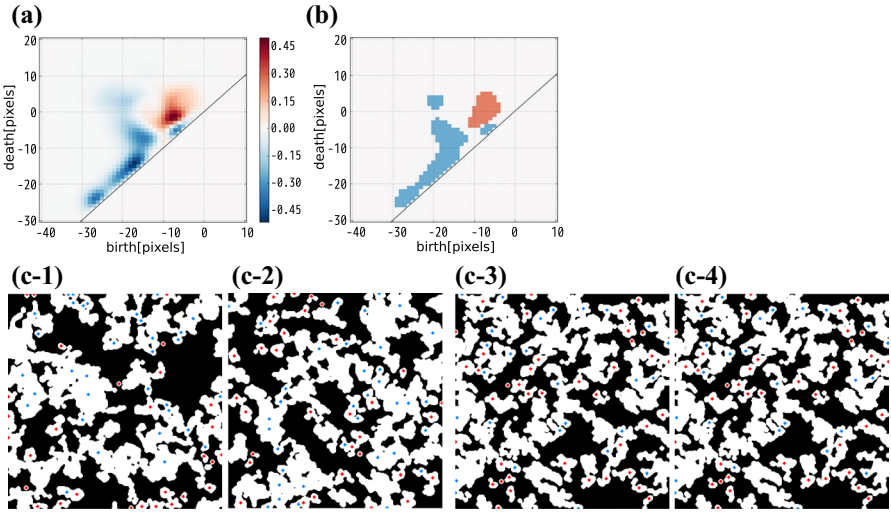

Fig. 5 **a** The reconstructed persistence diagram from the learned vector w. The blue (resp. red) area contributes to the class 0 (resp. 1). **b** A thresholding of (**a**). **c** 1–4 The birth positions of the generators in blue and red areas in (**b**) are plotted with the same color (color figure online)

shows the samples of both data (left: (A), right: (B)). We may intuitively observe that the images from (B) have somewhat finer structures than the images from (A). Our task is the classification of the parameters (A) and (B) from images, where we assign 0 and 1 for (A) and (B), respectively.

For each parameter pair, 300 images are generated (total 2×300) and 200 of these images are sampled as a training set (total 2×200). Then, 2×100 remaining images are used as a test set to evaluate the learned result. Here, 0th persistence diagrams are applied for the task. The parameters of the persistence images are set to be $\sigma = 2.0, C = 0.5, p = 1.0$ and the mesh for the discretized persistence images is obtained by dividing the rectangle $[-40.5, 10.5] \times [-30.5, 20.5]$ into 51×51 grids. The ℓ^2-regularization is used and the weight parameter λ of the regularization term is determined by the cross validation.

In this example, the score, evaluated as the mean accuracy, of the learned result is 1.0, that is, we can perfectly identify the parameter pairs behind the images. In fact, we could also distinguish these two parameter pairs by simply counting the number of connected components, if we had this prior knowledge. In Sect. 3.2, we examine a more sophisticated classification problem. For a while, let us use this example in order to explain some properties of our method.

Figure 5a shows the reconstructed persistence diagram from the learned vector w, and (b) shows the area at which the magnitude is above a certain threshold. Recalling the classification rule (5), nonzero elements in w (and hence nonzero generators in its reconstructed persistence diagram) work for making classification decisions. Namely, from the 0/1 assignment rule, the reconstructed persistence diagram concludes that generators in the blue (resp. red) area statistically contribute to the classification (A) (resp. (B)).

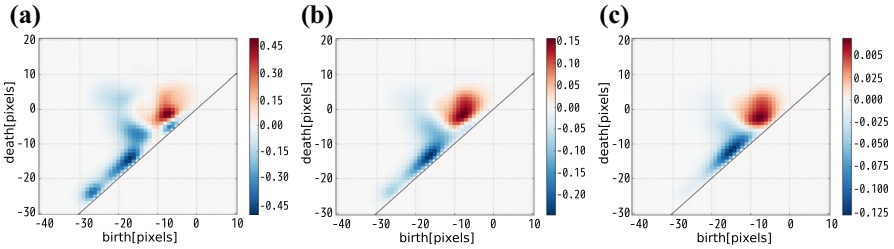

Fig. 6 The reconstructed persistence diagrams with several weight parameters λ. **a** $\lambda = 0.35938$ (determined by the cross validation), **b** $\lambda = 10$, and **c** $\lambda = 100$

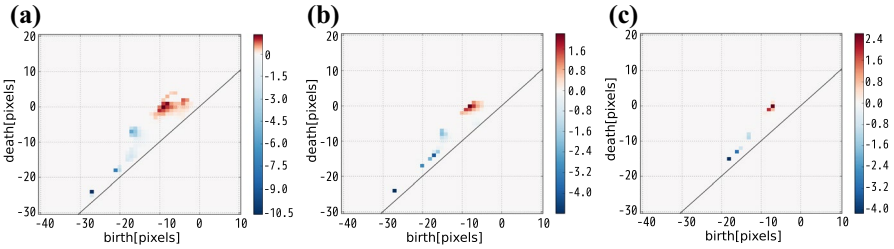

Fig. 7 The reconstructed persistence diagrams using the ℓ^1-regularization with several weight parameters λ. **a** $\lambda = 0.01$, **b** $\lambda = 0.1$, **c** $\lambda = 1$

Furthermore, by plotting the birth positions of these generators, we can explicitly identify the geometric structures which characterize the classification task. Figure 5c1–4 show those birth positions, where the blue (resp. red) points correspond to the blue (resp. red) area in (b). Recalling the interpretation in Fig. 3, we find that the characteristic geometric structures of (B) are explained by small islands and narrow bands whose inner radii are $4 \sim 10$ pixels; this is consistent to our intuition that (B) contains finer structures.

Using this example, let us study the effect of the weight parameter λ for the regularization. Figure 6 shows the reconstructed persistence diagrams from the learned vectors for several weight parameters λ. When λ becomes larger, in addition to the fact that the magnitude of the persistence diagram becomes smaller, its distribution becomes simpler. This is because the weigh parameter λ of the regularization controls the complexity of the learned result, which is expressed in the distribution of the reconstructed persistence diagram.

We also compare with the ℓ^1-regularization in this example. Figure 7 shows the reconstructed persistence diagrams using the ℓ^1-regularization with several parameters λ. As mentioned in Sect. 2.5, an important property of the ℓ^1-regularization is the sparseness of the learned result w. In our method, this property is reflected as sparse persistence diagram. Hence, again recalling the classification rule (5), the selected few grids in the sparse persistence diagram are supposed to work most effectively for the classification task. In other words, the birth-death pairs around

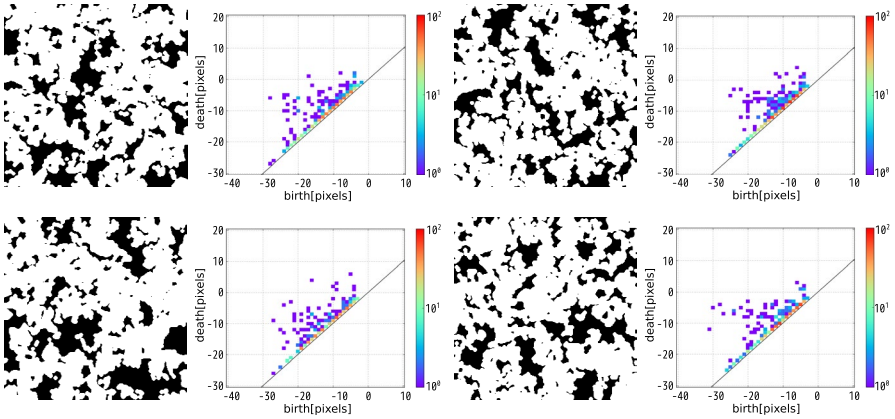

Fig. 8 Input binary images and their 0th persistence diagrams. The left and right images are sampled from the parameter pairs (**C**) and (**D**), respectively

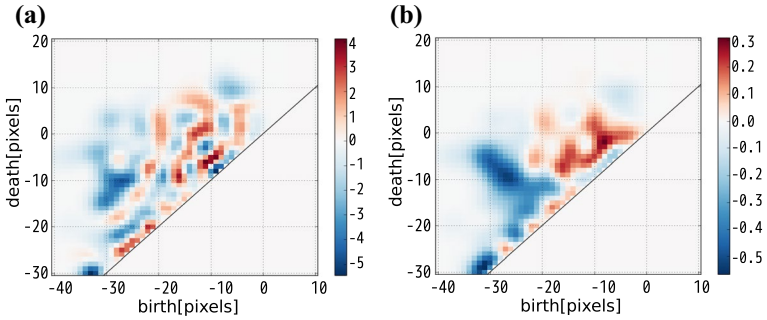

Fig. 9 The reconstructed persistence diagrams using the ℓ^2-regularization for the parameter pairs (**C**) and (**D**). **a** $\lambda = 0.0059948$ (chosen by cross validation), **b** $\lambda = 1$

the grids are especially important for the classification. Furthermore, the number of selected grids decreases for large λ as before, providing us with more compressed result and easier understandings of the learning.

3.2 Logistic regression on binary images—a hard example

Next, let us set the parameter pairs for generating random binary images so that the classification task becomes more difficult. Here, one parameter pair (C) is set to be $N = 160, S = 34$ and the other pair (D) is $N = 270, S = 18$. Figure 8 shows the sample input data [left: (C), right: (D)].

In this example, it seems difficult to distinguish two parameter pairs based on our intuition. In fact, simple descriptors such as the average numbers of connected components and white pixels do not work at all in this case.

Table 1 Performance comparison (PI: persistence image, SVM: support vector machine).

Method	Mean accuracy
PI, logistic regression, ℓ^2-penalty	0.92
Bag of keypoints using sift with grid sampling, SVM classifier with χ^2 kernel	0.85
# of connected components of black pixels	0.73
# of connected components of white pixels	0.50
# of white pixels	0.50

The setting for the classification is the same as before, i.e., 2×200 for training and 2×100 for the test, and we assign 0 and 1 to (C) and (D), respectively. In this case, the score on the test set is 0.92 (baseline: 0.5). Figure 9a shows the reconstructed persistence diagram as the learned result using the ℓ^2-regularization with the weight $\lambda = 0.0059948$ determined by the cross validation.

In this learning, the distribution of the reconstructed persistence diagram looks complicated to observe clear features. Hence, let us increase the weight parameter λ for simplifying the distribution. Figure 9b shows the result with $\lambda = 1$, where its score of the learning is 0.91. It should be noted that, although the score becomes only a little worse, the distribution turns out to be simple enough to conclude that the red area is dominant in the region with the birth scale > -20. From this simplification, we can explicitly obtain geometric reasonings for this classification in a similar way to Sect. 3.1.

Now we compare our method to other standard methods for image classifications. The list of methods and those scores are summarized in Table 1. These demonstrations show that persistence images with the logistic regression have better accuracy than the others. In particular, we note that the performance of our method is better than the bag of keypoints approach with sift feature, which is one of the standard techniques for image classifications (Lowe 1999; Sivic and Zisserman 2003; Csurka et al. 2004; Nowak et al. 2006)

This is because such standard image classification techniques are developed mainly for clearly distinguishable and well-structured objects such as photos of faces, artificial objects, or landscapes, and not for images like this example. This suggests that our approach using persistence diagrams has an advantage to disordered images, which are frequently observed in materials science data (Kimura et al. 2017).

We remark that OpenCV's python interface[9] is used for the computation of sift feature (cv2.xfeatures2d module) and bag of keypoints (cv2.BOWImgDescriptorExtractor). For the classification task, we use SVC class in sklearn.svm module for support vector classifier with χ^2 kernel from sklearn.metrics.pairwise module).

3.3 Logistic regression on point clouds

In this example, the input point clouds are prepared from two different random point processes; one is Poisson point process (PPP) and the other is Ginibre point process

[9] https://opencv.org/

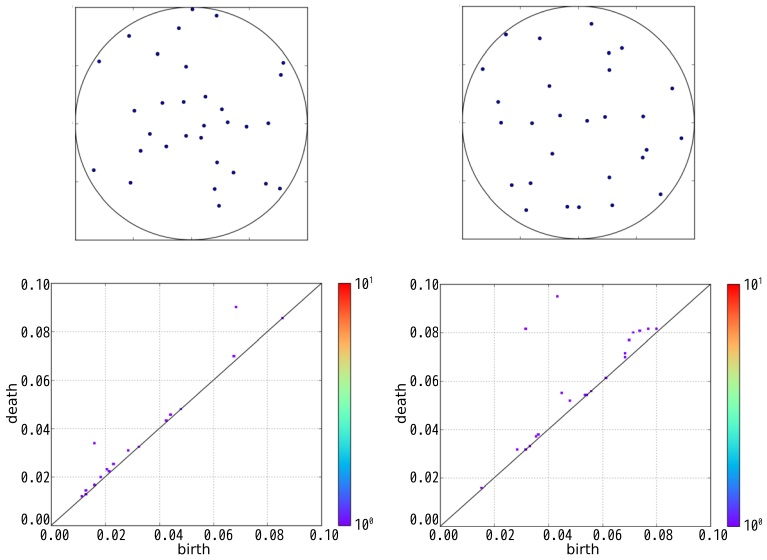

Fig. 10 Random point clouds (Left: PPP, Right: GPP) and their 1st persistence diagrams

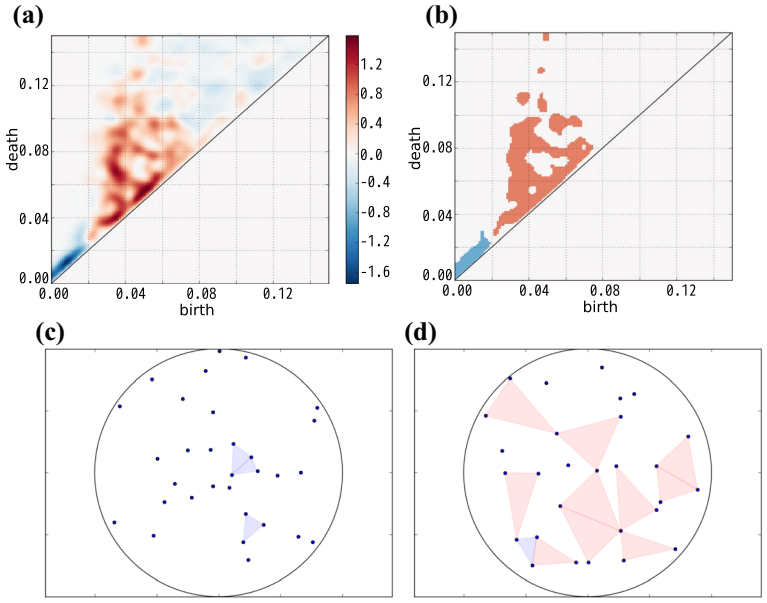

Fig. 11 **a** The reconstructed persistence diagram. The blue (resp. red) area contributes to the class 0 (resp. 1). **b** A thresholding of (**a**). **c** The death positions (triangles) in PPP. **d** The death positions (triangles) in GPP (color figure online)

(GPP) on a unit disk. It is known that PPP has no interaction between points, while GPP has a repulsive interaction. The parameters for these two point processes are adjusted so that the mean number of points on the disk is 30. The task in this

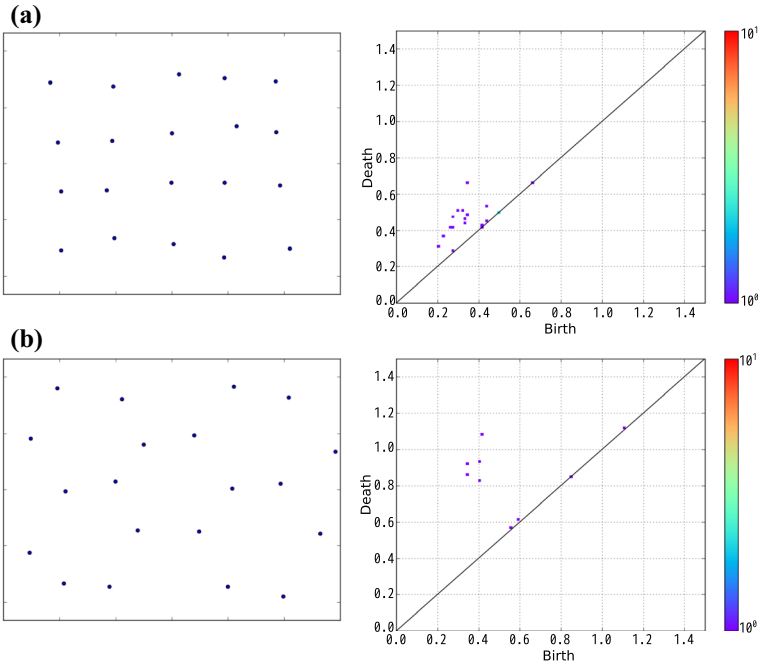

Fig. 12 Input point clouds and their 1st persistence diagrams. **a** A square lattice with noise. **b** A regular honeycomb pattern with noise

example is to identify PPP or GPP for test point clouds. To this task, we apply our method to the 1st persistence diagrams with the ℓ^2-logistic regression.

Figure 10 shows point clouds generated by PPP and GPP. The parameters of the persistence images are set to be $\sigma = 0.003, C = 80, p = 1.0$ and the mesh for the discretized persistence diagrams is obtained by dividing the square $[0, 0.15]^2$ into 150×150 grids. For each point process, 300 point clouds are generated (total 2×300) and 200 of these point clouds are sampled as a training set (total 2×200), where we assign 0 and 1 for PPP and GPP, respectively. The remaining 2×100 point clouds are used as a test set for evaluation. Here, the weight parameter λ of the ℓ^2-regularization is determined by the cross validation. The score of the learned result is 0.94.

Figure 11a shows the reconstructed persistence diagram from the learned vector w and (b) shows the positive and negative areas of (a) with a certain threshold. Recall that, from the 0/1 assignment, the generators in the blue (resp. red) area contributes to classifying into PPP (resp. GPP). From the learned persistence diagram, we observe that the red area is located on the region with large birth values. This is consistent to the fact that GPP has a repulsive interaction, and hence it prevents the point cloud from constructing rings with small birth values. Figure 11c, d show the death positions of the generators in the blue and red areas of (b) with the same colors, where (c) (resp. (d)) corresponds to PPP (resp. GPP). Similarly to the

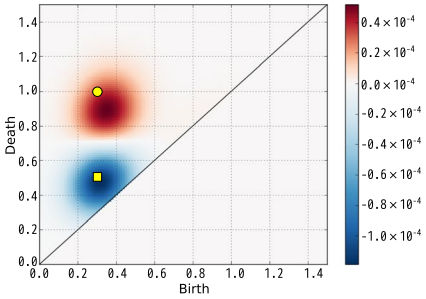

Fig. 13 The reconstructed persistence diagram. The yellow circle (resp. rectangle) shows the birth–death pair corresponding to the regular hexagon (resp. square) (color figure online)

discussion in Fig. 5, these death positions express characteristic geometric features used for learnings more explicitly.

We remark that PPP and GPP can also be distinguished by using other descriptors such as average nearest neighbor distances. An advantage of our method is that we do not need any prior knowledge, providing us with more universal method compared to problem-specific descriptors. In fact, the analysis using average nearest neighbor distance can be realized by the 0th persistence diagram.

Now let us test another example for point clouds. The task is classifying two types of point clouds; one is a square lattice with Gaussian noise, and the other is a regular honeycomb point pattern with Gaussian noise. Figure 12 shows the input point clouds and their 1st persistence diagrams. In this example, the average distance between two nearest neighbors is one for both cases, and hence it is difficult to distinguish these two types of point clouds using average nearest neighbor distances. In Pearson et al. (2015), persistent homology is used to quantitatively measure the regularity of noisy triangular lattice patterns generated by the numerical simulation of ion bombardment. The example will be helpful for that problem.

We set the number of points to be 20 and the standard deviation of the noise to be $\sigma = 0.1$. Figure 13 shows the reconstruct persistence diagram from the learned vector w. The yellow circle (resp. rectangle) in the diagram shows the birth–death pair of the regular hexagon (resp. square). One interesting feature in this result is that the positive peak position in the reconstructed diagram is shifted to the diagonal from yellow circle. Probably this is because such a regular shape is optimal in order to leave from the diagonal, and many birth–death pairs in noisy honeycomb patterns tend to move toward the diagonal.

3.4 Linear regression on binary images

In this example, we examine the linear regression on binary images. The input binary images are generated by Algorithm 2 with $N = 150$ and S is randomly chosen from $\{20, 21, \cdots, 29\}$ uniformly. The task is to determine the random parameter S from images. Figure 14 shows sample images with $S = 21$ and $S = 28$ and those persistence diagrams.

From the construction of Algorithm 2, we know that S controls the area of white pixels. Hence, to our task, we study the following descriptors:

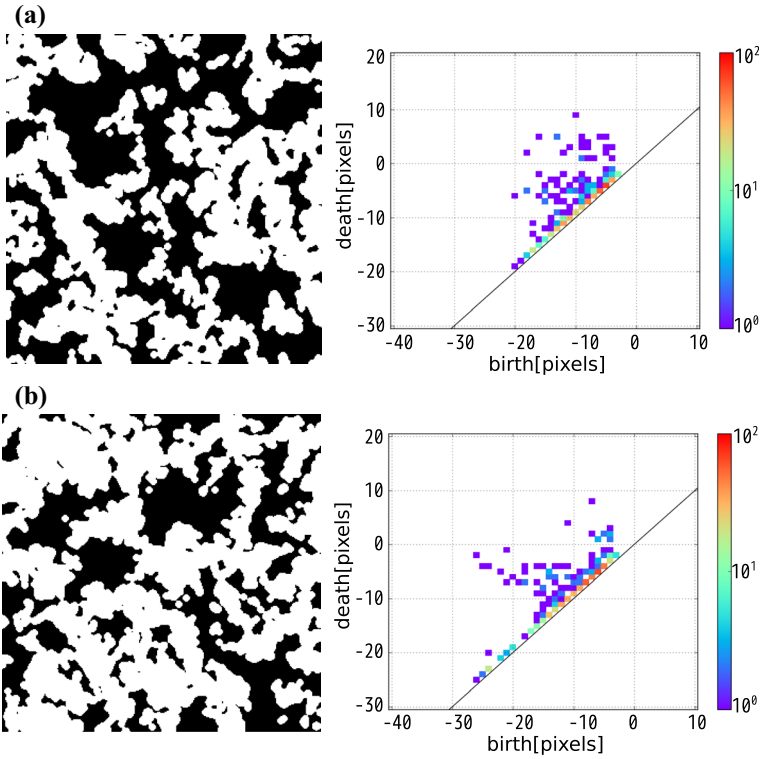

Fig. 14 Sample images for the linear regression and their 0th persistence diagrams. **a** $S = 21$, **b** $S = 28$

Table 2 R^2 coefficients on the test set of the linear regression problem. These values become larger when the learned model gives better predictions

	Method	R^2 coefficient
(i)	PI with ridge (ℓ^2)	0.86
	PI with lasso (ℓ^1)	0.86
(ii)	# of white pixels	0.88
(iii)	Both with ridge	0.93
	Both with lasso	0.94

(i) persistence image.
(i) the number of white pixels.
(iii) the combination of (i) and (ii).

as explanatory variables and compare these performances. Here, the third descriptor means that the response variable S is explained by the following model

$$S = v \cdot (\#\,\text{of white pixels}) + w \cdot (\text{PI}) + b + (\text{noise}), \tag{7}$$

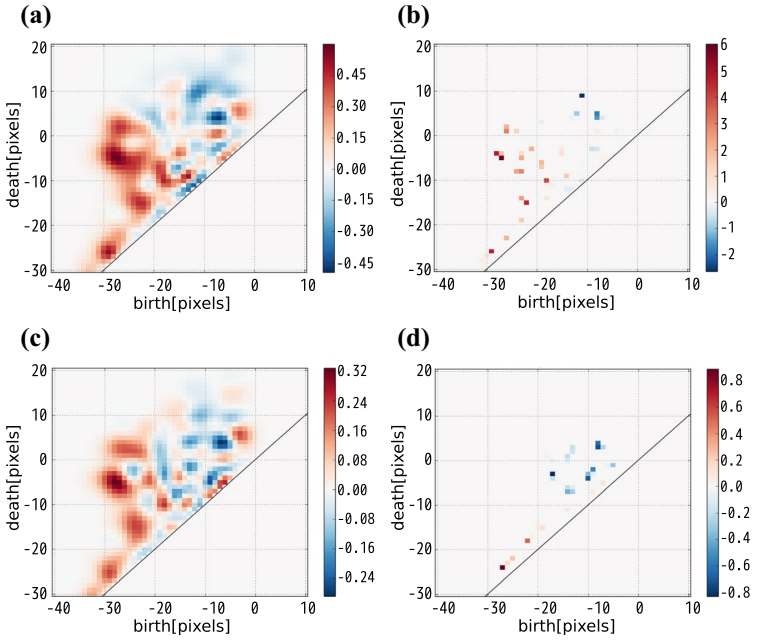

Fig. 15 The reconstructed persistence diagrams. **a** PI with ridge. **b** PI with lasso. **c** Both with ridge. **d** Both with lasso

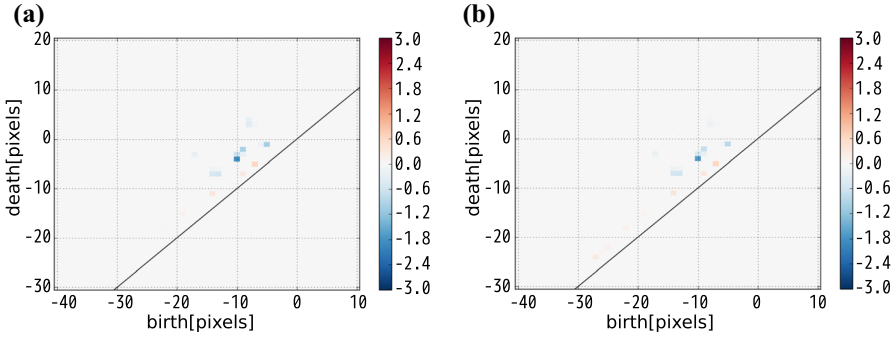

Fig. 16 The weighted persistence diagrams for Fig. 14a, b

where $v, b \in \mathbb{R}$ and $w \in \mathbb{R}^n$ are unknown parameters and determined from a training set. For (i) and (iii), we apply both ℓ^2- and ℓ^1-regularizations. The weight parameter λ of the regularization is determined by the cross validation.

The training set and test set consist of randomly generated 500 images and 100 images, respectively. The learned results are assessed using the R^2 coefficients of determination (Bingham 2010) on the test set, which are shown in Table 2. As we observe, our methods (i) using ℓ^1- and ℓ^2-regularizations attain almost the same performance as (ii), while the combination (iii) improves the performance better.

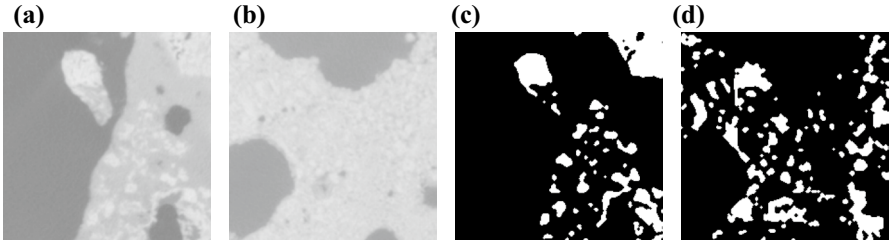

Fig. 17 X-ray CT images of iron ore sinters in the early (**a**) and intermediate stage (**b**). The sinters are composed from iron oxide (white regions), calcium ferrites (gray), and pores (black). The iron oxide regions in the early (**c**) and intermediate stage (**d**) are deconvoluted by the image analysis algorithms from (**a**) and (**b**), respectively

Figure 15 shows the reconstructed persistence diagrams obtained from (i) and (iii). By construction of our regression model, the areas with positive (resp. negative) values on the diagrams positively (resp. negatively) contribute to the response variable S. Even in the linear regression model, we can observe the sparseness property for the ℓ^1-regularization, which is useful for extracting the most essential features for the response variable S from sample data.

From the mixed model (7), we can estimate the contributions of (i) and (ii) in (iii) for predictions. For example, the following prediction results applied to Figure 14 (a) and (b) with the ℓ^1-regularization imply that the prediction mainly consists of the term $v \cdot (\#\text{ of white pixels})$ and is modified negatively by the term $w \cdot (\text{PI})$.

S	$\approx$ (prediction of S)	$= v \cdot (\#\text{ of white pixels})$	$+ w \cdot (\text{PI})$	$+ b$
21	≈ 20.628	$= 30.272$	$+ (-5.917)$	$+ (-3.728)$
28	≈ 27.959	$= 35.718$	$+ (-4.031)$	$+ (-3.728)$

Furthermore, by showing the weighted persistence diagram $(w_i x_i)_{i=1}^n$ for the test persistence diagram x, we can explicitly clarify the important generators for modifications. Figures 16 shows the weighted persistence diagrams of Fig. 14a, b, and in this case, we find that generators around $(-10, -4)$ effectively work for predictions of S.

For applications in materials science, S can be regarded as a certain physical quantity such as conductivity of battery materials. Then, by this approach, we can identify geometric structures in the images which most effectively affect that physical quantity.

3.5 Application on heterogeneous chemical reactions

We apply our method to X-ray CT images studied in the practical research problem of iron ore sinters. The analysis here is originally shown in the supplementary notes in the paper (Kimura et al. 2017). In this section, we show the comparison of performance in more detail.

Table 3 Mean accuracies for the classification task

Method	Mean accuracy
Logistic regression on PI with ℓ^1-regularization	0.83
SVM classifier with χ^2-kernel by using the bag of keypoints with sift feature	0.74
Logistic regression with ℓ^1-regularization by using the bag of keypoints with sift feature	0.71
# of connected components	0.81
# of white pixels	0.56

An iron ore sinter is an initial material for iron making process. It is produced by liquid-sintering iron oxide grains with calcium ferrites (CFs) at high temperature, and then is reduced in blast furnaces to produce pig iron. Figure 17 shows sliced images extracted from the three-dimensional X-CT dataset of iron ore sinters that experienced different degrees of reduction. Here, Fig. 17a, b exemplify images at the early and intermediate stage of reduction.[10] The mechanical property of iron ore sinters, degraded by localized stress or micro cracks, is important for efficient iron-making, and to control their mechanical property, we need to characterize chemical reactions progressing heterogeneously during reduction. For more details of the background and the experiment, we refer the readers to the original paper (Kimura et al. 2017).

The task of this example is to identify the characteristic change of heterogeneous distribution in chemical states of iron oxide during the reduction process of iron ore sinters. To this aim, we prepare 60 images for each of early and intermediate stage of reduction, extracted from the three-dimensional X-CT dataset. From these images, the regions of iron oxide are deconvoluted by standard image processing techniques such as denoising and thresholding. Figure 17c, d show the region of iron oxides, obtained by this process from (a) and (b), respectively. For characterizing the heterogeneous distributions in the early and intermediate stages, we use the logistic regression on persistence diagrams with ℓ^2-regularization term. For the comparison, we also apply the following image analyses to the data:

- SVM classifier with χ^2-kernel by using the bag of keypoints with sift feature.
- logistic regression with ℓ^1-regularization term by using the bag of keypoints with sift feature.
- the number of connected components as a descriptor.
- the number of white pixels as a descriptor.

Since the data size is small, we use the following approach to compute the mean accuracy:

[10] In the paper (Kimura et al. 2017), images in the final stage are also used. In this paper, we only use early and intermediate stage images to focus on the initial changes in the reaction.

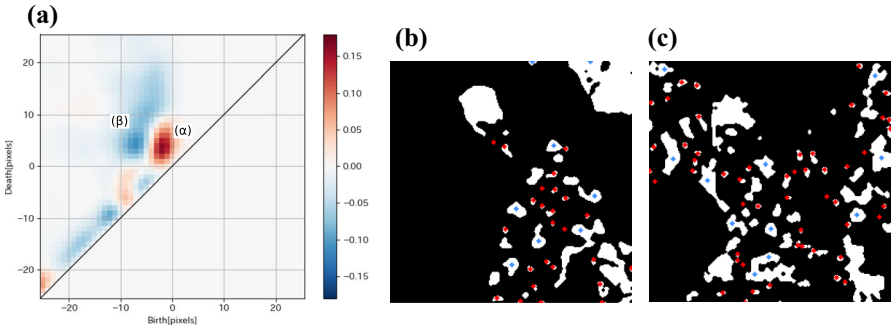

Fig. 18 **a** Reconstructed persistence diagram. Red and blue marks show typical geometrical features found in the early and intermediate state of reduction. High values of the learned vectors, marked as (α) and (β), were converted into real space and are shown in (**b**) and (**c**) for the early and intermediate state of reduction, respectively (color figure online)

1. Randomly pick up 5×2 images from the early and intermediate stages.
2. Learn from the remaining 55×2 images.
3. Compute the mean accuracy on the selected 5×2 images.
4. Repeat the above experiments 100 times and compute the average of the accuracies.

Table 3 shows the mean accuracies of the example.

The table shows that the logistic regression on the persistence images gives the highest score, and our method is the best descriptor in the list to describe geometric features of practical materials data: the heterogeneous distribution of iron oxide region. However, we note that this is not the point we want to emphasize here in this analysis. As we explained at the beginning of this section, our goal is to explicitly identify the change of the heterogeneous distribution during the sintering process, and hence, the classification performance itself is the minimum request to be guaranteed for analysis.

As we already explained in the previous sections using the synthetic data, the inverse analysis of the reconstructed persistence diagram provides us with an appropriate method for finding geometric features corresponding to the most important correlations. Figure 18a shows the reconstructed persistence diagram from the learned vector, where the classification label 0 and 1 are assigned for the early and intermediate stage, respectively. Hence, the negative (resp. positive) birth–death regions in the reconstructed persistence diagram contribute to classify into the early (resp. intermediate) stage. High values of the learned vectors, marked as (α) and (β), were converted into real space and are shown in Fig. 18b, c. Namely, these marked red and blue points are the geometric features statistically characterizing the key geometric features in heterogeneous distributions of iron oxides at the early and intermediate state of reduction, respectively.

This result was verified by another type analysis using the number of connected components (Kimura et al. 2017). The classification task using the number of connected components achieves a sufficiently good score, and it means that the number

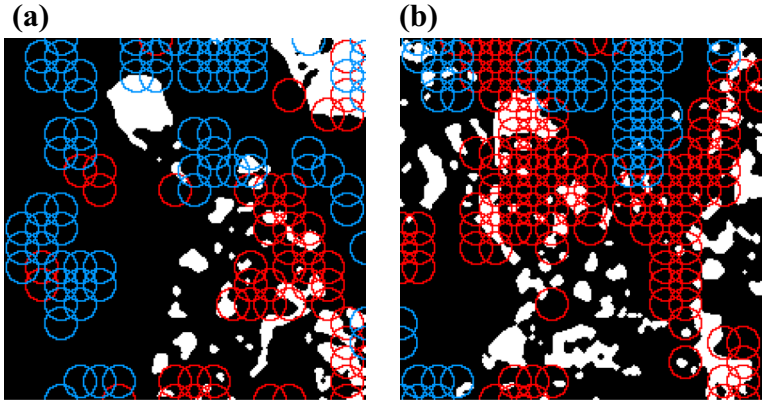

Fig. 19 Characteristic regions identified by the combination of sift feature, bag of keypoints, logistic regression with ℓ^1-regularization term. **a** Early stage, **b** intermediate stage (color figure online)

of connected components becomes larger from the early to intermediate stage. If our interest is simply the classification, the number of connected components can be a good candidate for the descriptor. However, since our goal is to explicitly extract the geometric features behind this classification, this simple descriptor is not sufficient for our purpose. In fact, the number of connected components cannot clarify which connected components are important to distinguish data at the early and intermediate stage. In contrast, our method using the reconstructed persistence diagram successfully revealed that the small connected components whose radii are less than 6 pixels are important to distinguish them.

It is often the case that prior knowledge on the obtained data is too limited to analyze in practical problems. In such a case, a wrong selection from simple descriptors may cause the serious difference in the machine learning tasks such as selecting the number of connected components or that of white pixels in this analysis (see Table 3). However, our method can find the characteristic geometric features in a more systematic and straightforward way without any prior knowledge.

We also applied an inverse analysis to the logistic regression with ℓ^1- regularization with the bag of keypoints approach. Since each keypoint characterizes a local region in the original image data, we can identify the corresponding region from the learned result by using the feature selection technique. Figure 19 shows the result of the inverse analysis. The shapes around blue (resp. red) circles statistically contribute to the classification into the early stage (resp. intermediate stage). Since the ratio of the numbers of red and blue circles becomes larger from the early to intermediate stages, these circles are expected to capture some differences between the early and intermediate stages. But, as we observe, it is very difficult to understand the typical geometric features to distinguish the two stages. This example concludes that the sift feature is more obscure as a descriptor than the persistence diagrams.

In the original paper (Kimura et al. 2017), we discussed more comprehensive analysis by using even final stage data and the principal components analysis on persistence diagrams for identifying specific geometric features (or trigger sites)

determining macroscopic mechanical properties, through the initiation of micro cracks, in heterogeneous chemical reactions.

4 Conclusion

In this paper, we have proposed a unified method by combining persistence images and linear machine learning models with the ability to study the inverse problem in the original data space. One of the important properties of our method is that a persistence diagram is obtained as a learned result. From such a reconstructed dual persistence diagram and the inverse analysis using birth/death positions, we can explicitly characterize significant geometric features embedded in dataset. We have also presented sparse persistence diagrams as an important concept of machine learnings in topological data analysis.

Although we applied our method to linear regressions and logistic regressions, it is obviously not limited to them, and many other linear machine learning models such as support vector machine with a linear kernel and elastic nets are also applicable. Moreover, we can similarly apply our method to point clouds and cubical sets in higher dimensions.

The proposed method is recently applied to several practical problems. For example, in the paper (Kimura et al. 2017), the authors develop a method for predicting locations of micro cracks generated by reduction reaction process of iron ore sinters. In Sect. 3.5, we have analyzed several related topics in this problem. In that application, they apply the persistence images with the ℓ^1-linear regression to a huge amount of X-CT images, and select the crack areas as a response variable. Then, it follows that the reconstructed persistence diagram from the learned vector identifies generators which have significant effects on crack formations, and hence, by studying their birth/death positions, we can explicitly detect the location of micro cracks. We believe that the same analysis is also useful to other problems dealing with large amount of images such as pathology.

Compliance with ethical standards

Conflict of interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

A Algorithm for generating random images

The algorithm for generating random binary images is given by Algorithm 2. It consists of six parameters, $W, N, S \in \mathbb{N}, \sigma_1 > 0, \sigma_2 > 0$, and $t > 0$. The area of white pixels in the generated image is given by the orbits of the Brownian motion of N particles on a flat torus with the size $W \times W$. The parameters S and σ_1 determine the length of each orbit and σ_2 and t determine the radii of particles. In this paper we fix $W = 300$, $\sigma_1 = 4$, $\sigma_2 = 2$, $t = 0.01$, and only N and S are changed. When N and S become larger, the generated image tend to have more white pixels.

These kinds of random images are frequently obtained by experimental measurements in materials science such as X-CT and TEM (Kimura et al. 2017). These seemingly disordered images are supposed to be utilized for materials informatics, and one of the motivations of this paper is to develop a universal framework for this purpose.

Algorithm 2 Generate a random binary image

procedure GEN-IMAGE($W, N, S, \sigma_1, \sigma_2, t$)
  Let T be $[0, W] \times [0, W]$
  for $n = 1, \ldots, N$ **do**
    Take $x_{n,1}$ uniformly randomly on T
    for $s = 1, \ldots, S$ **do**
      Take d_1 and d_2 randomly from $\mathcal{N}(0, \sigma_1)$
      $x_{n,s+1} \leftarrow x_{n,s} + (d_1, d_2) \mod W \times W$
  $H \leftarrow$ The Histogram of $\{x_{n,s}\}$ with $W \times W$ mesh on T
  Apply Gaussian filter to H with the standard deviation σ_2 and set the result to $\tilde{H}$
  return The Binary image from $\tilde{H}$ by thresholding with t

References

Adams, H., Chepushtanova, S., Emerson, T., Hanson, E., Kirby, M., Motta, F., Neville, R., Peterson, C., Shipman, P., Ziegelmeier, L.: Persistence images: a stable vector representation of persistent homology. J. Mach. Learn. Res. **18**(8), 1–35 (2017)

Bauer, U., Kerber, M., Reininghaus, J.: Distributed computation of persistent homology. Proceedings of the Sixteenth Workshop on Algorithm Engineering and Experiments (ALENEX) (2014)

Bauer, U., Kerber, M., Reininghaus, J., Wagner, H.: Phat—persistent homology algorithms toolbox. J. Symb. Comput. **78**, 76–90 (2017)

Bingham, N.H., Fry, J.M.: Regression—Linear Models in Statistics. Springer, Berlin (2010)

Bishop, C.M.: Pattern Recognition and Machine Learning (Information Science and Statistics). Springer, Berlin (2007)

Bubenik, P.: Statistical topological data analysis using persistence landscapes. J. Mach. Learn. Res. **16**(1), 77–102 (2015)

Buchet, M., Hiraoka, Y., Obayashi, I.: Persistent homology and materials informatics. In: Tanaka, I. (ed.) Nanoinformatics, pp. 75–95. Springer, Berlin (2018)

Carlsson, G.: Topology and data. Bull. Am. Math. Soc. **46**, 255–308 (2009)

Chazal, F., Glisse, M., Labruére, C., Michel, B.: Convergence rates for persistence diagram estimation in topological data analysis. J. Mach. Learn. Res. **16**, 3603–3635 (2015)

Chan, J.M., Carlsson, G., Rabadan, R.: Topology of viral evolution. PNAS **110**(46), 18566–18571 (2013)

Cohen-Steiner, D., Edelsbrunner, H., Harer, J.: Stability of persistence diagrams. Discret. Comput. Geom. **37**(1), 103–120 (2007)

Csurka, G., Bray, C., Dance, C. Fan, L.: Visual categorization with bags of keypoints. In: Proceeding of ECCV Workshop on Statistical Learning in Computer Vision, pp. 59–74 (2004)

Da, T.K.F., Loriot, S., Yvinec, M.: 3D Alpha Shapes. CGAL User and Reference Manual 4.11, CGAL Editorial Board (2017)

Delgado-Friedrichs, O., Robins, V., Sheppard, A.: Morse theory and persistent homology for topological analysis of 3D images of complex materials. In: 2014 IEEE International Conference on Image Processing (ICIP), pp. 4872–4876 (2014)

Delgado-Friedrichs, O., Robins, V., Sheppard, A.: Skeletonization and partitioning of digital images using discrete Morse theory. IEEE Trans. Pattern Anal. Mach. Intell. **37**(3), 654–666 (2015)

de Silva, V., Ghrist, R.: Coverage in sensor networks via persistent homology. Algebraic Geom. Topol. **7**, 339–358 (2007)

Dey, T.K., Hirani, A.N., Krishnamoorthy, B.: Optimal homologous cycles, total unimodularity and linear programming. SIAM J. Comput. **40**(4), 1026–1044 (2011)
Edelsbrunner, H., Letscher, D., Zomorodian, A.: Topological persistence and simplification. Discret. Comput. Geom. **28**(4), 511–533 (2002)
Edelsbrunner, H., Harer, J.: Computational Topology: An Introduction. AMS, Providence (2010)
Escolar, E.G., Hiraoka, Y.: Optimal cycles for persistent homology via linear programming. Optimization in the Real World Toward Solving Real-World Optimization Problems, pp. 79–96. Springer Japan, Osaka (2016)
Fasy, B.T., Lecci, F., Rinaldo, A., Wasserman, L., Balakrishnan, S., Singh, A.: Confidence sets for persistence diagrams. Ann. Stat. **42**(6), 2301–2339 (2014)
Hiraoka, Y., Nakamura, T., Hirata, A., Escolar, E.G., Matsue, K., Nishiura, Y.: Hierarchical structures of amorphous solids characterized by persistent homology. Proc. Nat. Acad. Sci. USA **113**, 7035–7040 (2016)
Ichinomiya, T., Obayashi, I., Hiraoka, Y.: Persistent homology analysis of craze formation. Phys. Rev. E **95**(1), 012504 (2017)
Jones, E., Oliphant, T., Peterson, .P, et al.: SciPy: Open source scientific tools for Python. http://www.scipy.org/ (2001–) [Online; accessed 2018-01-20]
Kaczynski, T., Mischaikow, K., Mrozek, M.: Computational Homology. Springer, Berlin (2004)
Kimura, M., Obayashi, I., Takeuchi, Y., Hiraoka, Y.: Finding trigger sites in heterogeneous reactions using persistent-homology without preliminary material scientific information. Sci. Rep. **8**, 3553 (2018)
Kusano, G., Fukumizu, K., Hiraoka, Y.: Persistence weighted Gaussian kernel for topological data analysis. Proceedings of the 33rd International Conference on Machine Learning, JMLR: W&CP 48. 2004-2013 (2016)
Kusano, G., Fukumizu, K., Hiraoka, Y.: Kernel method for persistence diagrams via kernel embedding and weight factor. Accepted in Journal of Machine Learning Research
Lowe, D.G.: Object recognition from local scale invariant features. In: Proc. of IEEE International Conference on Computer Vision, pp. 1150–1157 (1999)
Nowak, E., Jurie, F., Triggs, B.: Sampling Strategies for Bag-of-Features Image Classification. In: Computer Vision – ECCV 2006: 9th European Conference on Computer Vision, Graz, Austria, May 7-13, 2006, Proceedings, Part IV, pp. 490–503 (2006)
Otter, N., Porter, M.A., Tillmann, U., Grindrod, P., Harrington, H.A.: A roadmap for the computation of persistent homology. arXiv:1506.08903
Pearson, D.A., Bradley, R.M., Motta, F.C., Shipman, P.D.: Producing nanodot arrays with improved hexagonal order by patterning surfaces before ion sputtering. Phys. Rev. E **92**(6), 062401 (2015)
Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Erplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. J. Mach. Learn. Res. **12**, 2825–2830 (2011)
Rajan, K.: Materials informatics. Mater. Today **8**(10), 38–45 (2005)
Rajan, K.: Materials informatics. Mater. Today **15**(11), 470 (2012)
Reininghaus, J., Huber, S., Bauer, U., Kwitt, R.: A Stable Multi-Scale Kernel for Topological Machine Learning. 2015 IEEE Conference on Computer Vision and Pattern Recognition, 4741–4748 (2015)
Robert, T.: Regression shrinkage and selection via the lasso. J. R. Stat. Soc. Ser. B (Methodol.) **58**(1), 267–288 (1996)
Robins, V., Turner, K.: Principal component analysis of persistent homology rank functions with case studies of spatial point patterns, sphere packing and colloids. Phys. D **334**, 99–117 (2016)
Robins, V., Saadatfar, M., Delgado-Friedrichs, O., Sheppard, A.P.: Percolating length scales from topological persistence analysis of micro-CT images of porous materials. Water Resour. Res. **52**(1), 315–329 (2016)
Saadatfar, M., Takeuchi, H., Francois, N., Robins, V., Hiraoka, Y.: Pore configuration landscape of granular crystallisation. Nat. Commun. **8**, 15082 (2017). https://doi.org/10.1038/ncomms15082
Sivic, J. and Zisserman, A.: Video Google: A Text Retrieval Approach to Object Matching in Videos. In: Proc. of IEEE International Conference on Computer Vision, pp.1470–1477 (2003)
Turner, K., Mileyko, Y., Mukherjee, S., Harer, J.: Fréchet means for distributions of persistence diagrams. Discret. Comput. Geom. **52**(1), 44–70 (2014)
Zomorodian, A., Carlsson, G.: Computing persistent homology. Discret. Comput. Geom. **33**(2), 249–274 (2005)