Sliced Wasserstein Kernel for Persistence Diagrams
Mathieu Carriere, Marco Cuturi, Steve Y. Oudot

To cite this version:

HAL Id: hal-01633105
https://hal.archives-ouvertes.fr/hal-01633105
Submitted on 11 Nov 2017

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Abstract
Persistence diagrams (PDs) play a key role in topological data analysis (TDA), in which they are routinely used to describe topological properties of complicated shapes. PDs enjoy strong stability properties and have proven their utility in various learning contexts. They do not, however, live in a space naturally endowed with a Hilbert structure and are usually compared with non-Hilbertian distances, such as the bottleneck distance. To incorporate PDs in a convex learning pipeline, several kernels have been proposed with a strong emphasis on the stability of the resulting RKHS distance w.r.t. perturbations of the PDs. In this article, we use the Sliced Wasserstein approximation of the Wasserstein distance to define a new kernel for PDs, which is not only provably stable but also discriminative (with a bound depending on the number of points in the PDs) w.r.t. the first diagram distance between PDs. We also demonstrate its practicality, by developing an approximation technique to reduce kernel computation time, and show that our proposal compares favorably to existing kernels for PDs on several benchmarks.

1. Introduction
Topological Data Analysis (TDA) is an emerging trend in data science, grounded on topological methods to design descriptors for complex data—see e.g. (Carlsson, 2009) for an introduction to the subject. The descriptors of TDA can be used in various contexts, in particular statistical learning and geometric inference, where they provide useful insight into the structure of data. Applications of TDA can be found in a number of scientific areas, including computer vision (Li et al., 2014), materials science (Hiraoka et al., 2016), and brain science (Singh et al., 2008), to name a few. The tools developed in TDA are built upon persistent homology theory (Edelsbrunner & Harer, 2010; Oudot, 2015), and their main output is a descriptor called persistence diagram (PD), which encodes the topology of a space at all scales in the form of a point cloud with multiplicities in the plane $\mathbb{R}^2$—see Section 2.1 for more details.

PDs as features. The main strength of PDs is their stability with respect to perturbations of the data (Chazal et al., 2009b; 2013). On the downside, their use in learning tasks is not straightforward. Indeed, a large class of learning methods, such as SVM or PCA, requires a Hilbert structure on the descriptors space, which is not the case for the space of PDs. Actually, many simple operators of $\mathbb{R}^n$, such as addition, average or scalar product, have no analogues in that space. Mapping PDs to vectors in $\mathbb{R}^n$ or in some infinite-dimensional Hilbert space is one possible approach to facilitate their use in discriminative settings.

Related work. A series of recent contributions have proposed kernels for PDs, falling into two classes. The first class of methods builds explicit feature maps: One can, for instance, compute and sample functions extracted from PDs (Bubenik, 2015; Adams et al., 2017; Robins & Turner, 2016); sort the entries of the distance matrices of the PDs (Carrière et al., 2015); treat the PD points as roots of a complex polynomial, whose coefficients are concatenated (Fabio & Ferri, 2015). The second class of methods, which is more relevant to our work, defines implicitly feature maps by focusing instead on building kernels for PDs. For instance, Reininghaus et al. (2015) use solutions of the heat differential equation in the plane and compare them with the usual $L^2(\mathbb{R}^2)$ dot product. Kusano et al. (2016) handle a PD as a discrete measure on the plane, and follow by using kernel mean embeddings with Gaussian kernels—see Section 4 for precise definitions. Both kernels are provably stable, in the sense that the metric they induce in their respective reproducing kernel Hilbert space (RKHS) is bounded above by the distance between PDs. Although these kernels are injective, there is no evidence that their induced RKHS distances are discriminative and therefore follow the geometry of the bottleneck distances, which are more widely accepted distances to compare PDs.

Contributions. In this article, we use the sliced Wasserstein (SW) distance (Rabin et al., 2011) to define a new ker-
nel for PDs, which we prove to be both stable and discriminative. Specifically, we provide distortion bounds on the SW distance that quantify its ability to mimic bottleneck distances between PDs. This is in contrast to other kernels for PDs, which only focus on stability. We also propose a simple approximation algorithm to speed up the computation of that kernel, confirm experimentally its discriminative power and show that it outperforms experimentally both proposals of (Kusano et al., 2016) and (Reininghaus et al., 2015) in several supervised classification problems.

2. Background on TDA and Kernels

We briefly review in this section relevant material on TDA, notably persistence diagrams, and technical properties of positive and negative definite kernel functions.

2.1. Persistent Homology

Persistent homology (Zomorodian & Carlsson, 2005; Edelsbrunner & Harer, 2008; Oudot, 2015) is a technique inherited from algebraic topology for computing stable signatures on real-valued functions. Given \( f : X \to \mathbb{R} \) as input, persistent homology outputs a planar point set with multiplicities, called the persistence diagram of \( f \) and denoted by \( Dg f \). See Figure 1 for an example. To understand the meaning of each point in this diagram, it suffices to know that, to compute \( Dg f \), persistent homology considers the family of sublevel sets of \( f \), i.e., the sets of the form \( f^{-1}((−\infty, t]) \) for \( t \in \mathbb{R} \), and it records the topological events (e.g., creation or merge of a connected component, creation or filling of a loop, void, etc.) that occur in \( f^{-1}((−\infty, t]) \) as \( t \) ranges from \( −\infty \) to \( +\infty \). Then, each point \( p \in Dg f \) represents the lifespan of a particular topological feature (connected component, loop, void, etc.), with its creation and destruction times as coordinates. See again Figure 1 for an illustration.

For the interested reader, we point out that the mathematical tool used by persistent homology to track the topological events in the family of sublevel sets is homological algebra, which turns the parametrized family of sublevel sets into a parametrized family of vector spaces and linear maps. Computing persistent homology then boils down to computing a family of bases for the vector spaces, which are compatible with the linear maps.

**Distance between PDs.** We now define the \( p \)th diagram distance between PDs. Let \( p \in \mathbb{N} \) and \( Dg_1, Dg_2 \) be two PDs. Let \( \Gamma : Dg_1 \supseteq A \to B \subseteq Dg_2 \) be a partial bijection between \( Dg_1 \) and \( Dg_2 \). Then, for any point \( x \in A \), the cost of \( x \) is defined as \( c(x) := \|x - \Gamma(x)\|_{\pi_c} \), and for any point \( y \in (Dg_1 \cup Dg_2) \setminus (A \cup B) \), the cost of \( y \) is defined as \( c'(y) := \|y - \pi_\Delta(y)\|_{\pi_c} \), where \( \pi_c \) is the projection onto the diagonal \( \Delta = \{(x, x) \mid x \in \mathbb{R} \} \). The cost \( c(\Gamma) \) is defined as: \( c(\Gamma) := (\sum_x c(x) + \sum_y c'(y))^{1/p} \). We then define the \( p \)th diagram distance \( d_p \) as the cost of the best partial bijection between the PDs:

\[
d_p(Dg_1, Dg_2) = \inf_{\Gamma} c(\Gamma).
\]

In the particular case \( p = +\infty \), the cost of \( \Gamma \) is defined as \( c(\Gamma) := \max\{\max_x \delta(x) + \max_y \delta'(y)\} \). The corresponding distance \( d_\infty \) is often called the bottleneck distance. One can show that \( d_p \to d_\infty \) when \( p \to +\infty \). A fundamental property of PDs is their stability with respect to (small) perturbations of their originating functions. Indeed, the stability theorem (Bauer & Lesnick, 2015; Chazal et al., 2009a; 2016; Cohen-Steiner et al., 2007) asserts that for any \( f, g : X \to \mathbb{R} \), we have

\[
d_\infty(Dg f, Dg g) \leq \|f - g\|_{\infty},
\]

See again Figure 1 for an illustration.

In practice, PDs can be used as descriptors for data via the choice of appropriate filtering functions \( f \), e.g., distance to the data in the ambient space, eccentricity, curvature, etc. The main strengths of the obtained descriptors are: (a) to be provably stable as mentioned previously; (b) to be invariant under reparametrization of the data; and (c) to encode information about the topology of the data, which is complementary and of an essentially different nature compared to geometric or statistical quantities. These properties have made persistence diagrams useful in a variety of contexts, including the ones mentioned in the introduction of the paper. For further details on persistent homology and on applications of PDs, the interested reader can refer e.g. to (Oudot, 2015) and the references therein.

2.2. Kernel Methods

**Positive Definite Kernels.** Given a set \( X \), a function \( k : X \times X \to \mathbb{R} \) is called a positive definite kernel if for all integers \( n \), for all families \( x_1, ..., x_n \) of points in \( X \), the matrix \( [k(x_i, x_j)]_{i,j} \) is itself positive semi-definite. For brevity we will refer to positive definite kernels as kernels in the rest of the paper. It is known that kernels generalize scalar products, in the sense that, given a kernel \( k \), there exists a Reproducing Kernel Hilbert Space (RKHS) \( \mathcal{H}_k \) and a feature map \( \phi : X \to \mathcal{H}_k \) such that \( k(x_1, x_2) = \langle \phi(x_1), \phi(x_2) \rangle_{\mathcal{H}_k} \). A kernel \( k \) also induces a distance \( d_k \) on \( X \) that can be computed as the Hilbert norm of the difference between two embeddings:

\[
d_k^2(x_1, x_2) \triangleq k(x_1, x_1) + k(x_2, x_2) - 2k(x_1, x_2).
\]

We will be particularly interested in this distance, since one of the goals we will aim for will be that of designing a kernel \( k \) for persistence diagrams such that \( d_k \) has low distortion with respect to \( d_1 \).
Sliced Wasserstein Kernel for Persistence Diagrams

Figure 1. Sketch of persistent homology: (a) the horizontal lines are the boundaries of sublevel sets \( f((\infty, t]) \), which are colored in decreasing shades of grey. The vertical dotted lines are the boundaries of their different connected components. For instance, a new connected component is created in the sublevel set \( f^{-1}(\infty, t]) \) when \( t = f(p) \), and it is merged (destroyed) when \( t = f(s) \); its lifespan is represented by a copy of the point with coordinates \((f(p), f(s))\) in the persistence diagram of \( f \) (Figure (c)); (b) a piecewise-linear approximation \( g \) (blue) of the function \( f \) (red) from sampled values; (c) superposition of \( Dg_f \) (red) and \( Dg_y \) (blue), showing the partial matching of minimum cost (magenta) between the two persistence diagrams.

Negative Definite and RBF Kernels. A standard way to construct a kernel is to exponentiate the negative of a Euclidean distance. Indeed, the Gaussian kernel for vectors with parameter \( \sigma > 0 \) does follow that template approach: \( k_\sigma(x, y) = \exp \left( -\frac{\|x - y\|^2}{2\sigma^2} \right) \). An important theorem of Berg et al. (1984) (Theorem 3.2.2, p.74) states that such an approach to build kernels, namely setting \( k_\sigma(x, y) \) for an arbitrary function \( f \) can only yield a valid positive definite kernel for all \( \sigma > 0 \) if and only if \( f \) is a negative semi-definite function, namely that, for all integers \( n \), \( \forall x_1, \ldots, x_n \in X, \forall \alpha_1, \ldots, \alpha_n \in \mathbb{R}^n \) such that \( \sum_i \alpha_i = 0 \), \( \sum_{i,j} \alpha_i \alpha_j f(x_i, x_j) \leq 0 \).

Unfortunately, as observed in Appendix A of Reininghaus et al. (2014), \( d_1 \) is not negative semi-definite (it only suffices to sample a family of point clouds to observe experimentally that more often than not the inequality above will be violated for a particular weight vector \( a \)). In this article, we use an approximation of \( d_1 \) with the Sliced Wasserstein distance, which is provably negative semi-definite, and we use it to define a RBF kernel that can be easily tuned thanks to its bandwidth parameter \( \sigma \).

2.3. Wasserstein distance for unnormalized measures on \( \mathbb{R} \)

The Wasserstein distance (Villani, 2009, §6) is a distance between probability measures. For reasons that will become clear in the next section, we will focus on a variant of that distance: the \( l_1 \)-Wasserstein distance for nonnegative, not necessarily normalized, measures on the real line (Santambrogio, 2015, §2). Let \( \mu \) and \( \nu \) be two nonnegative measures on the real line such that \( |\mu| = \mu(\mathbb{R}) \) and \( |\nu| = \nu(\mathbb{R}) \) are equal to the same number \( r \). We define the three following objects:

\[
\mathcal{W}(\mu, \nu) = \inf_{P \in \Pi(\mu, \nu)} \int_{\mathbb{R} \times \mathbb{R}} |x - y| P(dx, dy) \quad (2)
\]

\[
\mathcal{Q}_r(\mu, \nu) = r \int |M^{-1}(x) - N^{-1}(x)| dx \quad (3)
\]

\[
\mathcal{L}(\mu, \nu) = \inf_{f \in \text{Lipschitz}} \int_{\mathbb{R}} f(x)[|\mu(dx)| - |\nu(dx)|] \quad (4)
\]

where \( \Pi(\mu, \nu) \) is the set of measures on \( \mathbb{R}^2 \) with marginals \( \mu \) and \( \nu \), and \( M^{-1} \) and \( N^{-1} \) the generalized quantile functions of the probability measures \( \mu/\nu \) and \( \nu/\nu \) respectively.

Proposition 2.1. We have \( \mathcal{W} = \mathcal{Q}_r = \mathcal{L} \). Additionally (i) \( \mathcal{Q}_r \) is negative definite on the space of measures of mass \( r \); (ii) for any three positive measures \( \mu, \nu, \gamma \) such that \( |\mu| = |\nu| \), we have \( \mathcal{L}(\mu + \gamma, \nu + \gamma) = \mathcal{L}(\mu, \nu) \).

Equation (2) is the generic Kantorovich formulation of optimal transport, which is easily generalized to other cost functions and spaces, the variant being that we consider an unnormalized mass by reflecting it directly in the set \( \Pi \). The equality between (2) and (3) is only valid for probability measures on the real line. Because the cost function \( |\cdot| \) is homogeneous, we see that the scaling factor \( r \) can be removed when considering the quantile function and multiplied back. The equality between (2) and (4) is due to the well known Kantorovich duality for a distance cost (Villani, 2009, Particular case 5.4) which can also be trivially generalized to unnormalized measures, proving therefore the main statement of the proposition. The definition of \( Q_r \) shows that the Wasserstein distance is the \( l_1 \) norm of
be the orthogonal projection onto $L$ and $d = \sum \delta_{y_i}$ be two PDs, and let $\Pi$ be the orthogonal projection onto the diagonal. Then, the Sliced Wasserstein kernel $SW$ is defined as:

\[ SW(D_{g_1}, D_{g_2}) \equiv \frac{1}{2\pi} \int_{\Theta} W(\mu_1^\theta + \mu_2^\theta, \mu_2^\theta + \mu_1^\theta) d\theta. \]

Note that, by symmetry, one can restrict on the half-circle $[-\pi, \pi]$ and normalize by $\pi$ instead of $2\pi$. Since $Q$ is negative semi-definite, we can deduce that SW itself is negative semi-definite:

**Lemma 3.2.** Let $X$ be the set of bounded and finite PDs. Then, SW is negative semi-definite on $X$.

**Proof.** Let $n \in \mathbb{N}^*$, $a_1, \ldots, a_n \in \mathbb{R}$ such that $\sum_{i} a_i = 0$ and $D_{g_1}, \ldots, D_{g_n} \in X$. Given $1 \leq i \leq n$, we let $\mu_i^\theta = \mu_1^\theta + \sum_{q \in D_{g_i}, k \neq i} \delta_{\sigma_i \sigma(q)}$, $\tilde{\mu}_i^\theta = \sum_{q \in D_{g_i}, k \neq i} \delta_{\sigma_i \sigma(q)}$, and $d = \sum_{i} |D_{g_i}|$. Then:

\[
\begin{align*}
&\sum_{i,j} a_i a_j W(\mu_i^\theta + \mu_j^\theta, \mu_j^\theta + \mu_i^\theta) \\
&= \sum_{i,j} a_i a_j \mathcal{L}(\mu_i^\theta + \mu_j^\theta, \mu_j^\theta + \mu_i^\theta) \\
&= \sum_{i,j} a_i a_j \mathcal{L}(\tilde{\mu}_i^\theta, \tilde{\mu}_j^\theta) \\
&= \sum_{i,j} a_i a_j Q_d(\tilde{\mu}_i^\theta, \tilde{\mu}_j^\theta) \leq 0
\end{align*}
\]

The result follows by linearity of integration. 

Hence, the theorem of Berg et al. (1984) allows us to define a valid kernel with:

\[ k_{SW}(D_{g_1}, D_{g_2}) = \exp \left( -\frac{SW(D_{g_1}, D_{g_2})}{2\sigma^2} \right). \]

**Metric equivalence.** We now give the main theoretical result of this article, which states that SW is equivalent to $d_1$. This has to be compared with (Reininghaus et al., 2015) and (Kusano et al., 2016), which only prove stability and injectivity. Our equivalence result states that the $k_{SW}$, in addition to be stable and injective, also preserves the metric between PDs, which should intuitively lead to an improvement of the classification power. This intuition is illustrated in Section 4 and Figure 4, where we show an improvement of classification accuracies on several benchmark applications.

**Theorem 3.3.** Let $X$ be the set of bounded PDs with cardinalities bounded by $N \in \mathbb{N}^*$. Let $D_{g_1}, D_{g_2} \in X$. Then, one has:

\[ d_1(D_{g_1}, D_{g_2}) \leq SW(D_{g_1}, D_{g_2}) \leq 2\sqrt{2}d_1(D_{g_1}, D_{g_2}), \]

where $M = 1 + 2N(N - 1)$.

**Proof.** Let $s^\theta : D_{g_1} \cup \pi_\Delta(D_{g_2}) \to D_{g_2} \cup \pi_\Delta(D_{g_1})$ be the one-to-one bijection between $D_{g_1} \cup \pi_\Delta(D_{g_2})$ and $D_{g_2} \cup \pi_\Delta(D_{g_1})$ induced by $\mathcal{L}(\mu_1^\theta + \mu_2^\theta, \mu_2^\theta + \mu_1^\theta)$, and let $s$ be the one-to-one bijection between $D_{g_1} \cup \pi_\Delta(D_{g_2})$ and $D_{g_2} \cup \pi_\Delta(D_{g_1})$ induced by the partial bijection achieving $d_1(D_{g_1}, D_{g_2})$.

**Upper bound.** Recall that $\|\theta\|_2 = 1$. We have:

\[ W(\mu_1^\theta + \mu_2^\theta, \mu_2^\theta + \mu_1^\theta) = \sum |(p - s^\theta(p), \theta)| \leq \sum |(p - s(p), \theta)| \leq \sqrt{2} \sum |p - s(p)||_\infty \leq 2\sqrt{2}d_1(D_{g_1}, D_{g_2}), \]

where the sum is taken over all $p \in D_{g_1} \cup \pi_\Delta(D_{g_2})$. The upper bound follows by linearity.

**Lower bound.** The idea is to use the fact that $s^\theta$ is a piecewise-constant function of $\theta$, and that it has at most $2 + 2N(N - 1)$ critical values $\Theta_0, \ldots, \Theta_M$ in $[-\frac{\pi}{N}, \frac{\pi}{N}]$. Indeed, it suffices to look at all $\theta$ such that $\{p_1 - p_2, \theta\} = 0$ for some $p_1, p_2$ in $D_{g_1} \cup \pi_\Delta(D_{g_2})$ or $D_{g_2} \cup \pi_\Delta(D_{g_1})$. Then:

\[
\int_{\Theta_i} |(p - s^\theta(p), \theta)| d\theta \\
= \sum |p - s^\theta(p)||_2 \int_{\Theta_i} |\cos(\angle(p - s^\theta(p), \theta))| d\theta \\
\geq \sum |p - s^\theta(p)||_2 |\Theta_{i+1} - \Theta_i)|^2 / 2\pi \\
\geq (\Theta_{i+1} - \Theta_i)^2 d_1(D_{g_1}, D_{g_2}) / 2\pi,
\]

where $\Theta_i = \{p_1 - p_2, \theta\}$ is the upper bound for the $\tilde{\mu}_i^\theta$.
where the sum is again taken over all $p \in Dg_1 \cup \pi_\Delta(Dg_2)$, and where the inequality used to lower bound the integral of the cosine is obtained by concavity. The lower bound follows then from the Cauchy-Schwarz inequality. □

Note that the lower bound depends on the cardinalities of the PDs, and it becomes close to 0 if the PDs have a large number of points. On the other hand, the upper bound is oblivious to the cardinality. A corollary of Theorem 3.3 is that $d_{kSW}$, the distance induced by $k_{SW}$ in its RKHS, is also equivalent to $d_1$ in a broader sense: there exist continuous, positive and monotone functions $g,h$ such that $g(0) = h(0) = 0$ and $g \circ d_1 \leq d_{kSW} \leq h \circ d_1$.

When the condition on the cardinalities of PDs is relaxed, e.g. when we only assume the PDs to be finite and bounded, with no uniform bound, the feature map $\phi_{kSW}$ associated to $k_{SW}$ remains continuous and injective w.r.t. $d_1$. This means that $k_{SW}$ can be turned into a universal kernel by considering $\exp(k_{SW})$ (cf Theorem 1 in (Kwitt et al., 2015)). This can be useful in a variety of tasks, including tests on distributions of PDs.

**Computation.** In practice, we propose to approximate $k_{SW}$ in $O(N \log(N))$ time using Algorithm 1. This algorithm first samples $M$ directions in the half-circle $\mathbb{S}^1_+$; it then computes, for each sample $\theta_i$ and for each PD $Dg$, the scalar products between the points of $Dg$ and $\theta_i$, to sort them next in a vector $V_{\theta_i}(Dg)$. Finally, the $l_1$-norm between the vectors is averaged over the sampled directions: $SW_M(Dg_1, Dg_2) = \frac{1}{M} \sum_{i=1}^M \| V_{\theta_i}(Dg_1) - V_{\theta_i}(Dg_2) \|_1$. Note that one can easily adapt the proof of Lemma 3.2 to show that $SW_M$ is negative semi-definite by using the linearity of the sum. Hence, this approximation remains a kernel. If the two PDs have cardinalities bounded by $N$, then the running time of this procedure is $O(M N \log(N))$. This approximation of $k_{SW}$ is useful since, as shown in Section 4, we have observed empirically that just a few directions are sufficient to get good classification accuracies. Note that the exact computation of $k_{SW}$ is also possible in $O(N^2 \log(N))$ time using the algorithm described in (Carrière et al., 2017).

4. Experiments

In this section, we compare $k_{SW}$ to $k_{PSS}$ and $k_{PWG}$ on several benchmark applications for which PDs have been proven useful. We compare these kernels in terms of classification accuracies and computational cost. We review first our experimental setting, and then all our tasks.

**Experimental setting** All kernels are handled with the LIBSVM (Chang & Lin, 2011) implementation of C-SVM, and results are averaged over 10 runs on a 2.4GHz Intel Xeon E5530 Quad Core. The cost factor $C$ is cross-validated in the following grid: \{0.001, 0.01, 0.1, 1, 10, 100, 1000\}. Table 1 summarizes the number of labels, and the number of training and test instances for each task. Figure 2 illustrate how we use PDs to represent complex data. We first describe the two baselines we considered, along with their parameterization, followed by our proposal.

**PSS.** The Persistence Scale Space kernel $k_{PSS}$ (Reininghaus et al., 2015) is defined as the scalar product of the two solutions of the heat diffusion equation with initial Dirac sources located at the PD points. It has the following closed form expression: $k_{PSS}(Dg_1, Dg_2) = \frac{1}{\sqrt{\pi t}} \sum_{p \in Dg_1} \sum_{q \in Dg_2} \exp \left( -\frac{||p-q||^2}{2 \pi t} \right) - \exp \left( -\frac{||p-q||^2}{2 \pi t} \right)$, where $q = (y,x)$ is the symmetric of $q$ (y, x) along the diagonal. Since there is no clear heuristic on how to tune $t$, this parameter is chosen in the applications by ten-fold cross-validation with random 50%-50% training-test splits and with the following set of $N_{\text{PSS}} = 13$ values: 0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1, 5, 10, 50, 100, 500 and 1000.

**PWG.** Let $K, p, \rho$ be positive parameters. Let $k_\rho$ be the Gaussian kernel with parameter $\rho$ and associated RKHS $\mathcal{H}_\rho$. Let $Dg_1, Dg_2$ be two PDs, and let $\mu_1 := \sum_{x \in Dg_1} \arctan(K \rho \Delta(x, \Delta)^p) k_\rho(\cdot,x) \in \mathcal{H}_\rho$ be the kernel mean embedding of $Dg_1$ weighted by the diagonal distances. Let $\mu_2$ be defined similarly.
Let $\tau > 0$. The Persistence Weighted Gaussian kernel $k_{\text{PWG}}$ (Kusano et al., 2016; 2017) is defined as $k_{\text{PWG}}(D_{g_1}, D_{g_2}) = \exp \left( -\frac{\|\mu_1 - \mu_2\|^2}{2\tau^2} \right)$, i.e. the Gaussian kernel with parameter $\tau$ on $\mathcal{H}_p$. The authors in (Kusano et al., 2016) provide heuristics to compute $K$, $\rho$ and $\tau$ and give a rule of thumb to tune $p$. Hence, in the applications we select $p$ according to the rule of thumb, and we use ten-fold cross-validation with random 50%-50% training-test splits to chose $K$, $\rho$ and $\tau$. The ranges of possible values is obtained by multiplying the values computed with the heuristics with the following range of 5 factors: 0.01, 0.1, 1, 10 and 100, leading to $N_{\text{PWG}} = 5 \times 5 \times 5 = 125$ different sets of parameters.

**Parameters for $k_{\text{SW}}$.** The kernel we propose has only one parameter, the bandwidth $\sigma$ in Eq. (5), which we choose using ten-fold cross-validation with random 50%-50% training-test splits. The range of possible values is obtained by computing the squareroot of the median, the first and the last deciles of all $SW(D_{g_i}, D_{g_j})$ in the training set, then by multiplying these values by the following range of 5 factors: 0.01, 0.1, 1, 10 and 100, leading to $N_{\text{SW}} = 5 \times 3 = 15$ possible values.

**Parameter Tuning.** The bandwidth of $k_{\text{SW}}$ is, in practice, easier to tune than the parameters of its two competitors when using grid search. Indeed, as is the case for all infinitely divisible kernels, the Gram matrix does not need to be recomputed for each choice of $\sigma$, since it only suffices to compute all the Sliced Wasserstein distances between PDs in the training set once. On the contrary, neither $k_{\text{PSS}}$ nor $k_{\text{PWG}}$ share this property, and require recomputations for each hyperparameter choice. Note however that this improvement may no longer hold if one uses other methods to tune parameters. For instance, using $k_{\text{PWG}}$ without cross-validation is possible with the heuristics given by the authors in (Kusano et al., 2016), and leads to smaller training times, but also to worse accuracies.

### 4.1. 3D shape segmentation

Our first task, whose goal is to produce point classifiers for 3D shapes, follows that presented in (Carrière et al., 2015).

**Data.** We use some categories of the mesh segmentation benchmark of Chen et al. (Chen et al., 2009), which contains 3D shapes classified in several categories (“airplane”, “human”, “ant”...). For each category, our goal is to design a classifier that can assign, to each point in the shape, a

<table>
<thead>
<tr>
<th>TASK</th>
<th>$k_{\text{PSS}} (10^{-3})$</th>
<th>$k_{\text{PWG}} (10^3)$</th>
<th>$k_{\text{SW}} (6)$</th>
<th>$k_{\text{PSS}} (10^{-3})$</th>
<th>$k_{\text{PWG}} (10^3)$</th>
<th>$k_{\text{SW}} (6) - NC^c$</th>
</tr>
</thead>
<tbody>
<tr>
<td>ORBIT</td>
<td>65.6 ± 1.2</td>
<td>77.7 ± 1.2</td>
<td><strong>83.7 ± 0.5</strong></td>
<td>N(121 ± 8.4)</td>
<td>N(144 ± 14)</td>
<td>415 ± 7.9</td>
</tr>
<tr>
<td>TEXTURE</td>
<td><strong>98.8 ± 0.0</strong></td>
<td>95.8 ± 0.0</td>
<td>96.1 ± 0.4</td>
<td>N(165 ± 27)</td>
<td>N(101 ± 9.6)</td>
<td>482 ± 68</td>
</tr>
</tbody>
</table>

Table 2: Classification accuracies (%) and Gram matrices computation time (s) for the benchmark applications. As explained in the text, $N$ represents the size of the set of possible parameters, and we have $N = 13$ for $k_{\text{PSS}}$, $N = 5 \times 5 \times 5 = 125$ for $k_{\text{PWG}}$ and $N = 3 \times 3 \times 3 = 27$ for $k_{\text{SW}}$. $C$ is a constant that depends only on the training size. In all our applications, it is less than 0.1s.

**Figure 2.** Examples of PDs computed on orbits, texture images and 3D shapes.
Sliced Wasserstein Kernel for Persistence Diagrams

Figure 3. The first row corresponds to the orbit recognition and the texture classification while the second row corresponds to 3D shape segmentation. On each row, the left plot shows the dependence of the accuracy on the number of directions, the middle plot shows the dependence of a single Gram matrix computation time, and the right plot shows the dependence of the ratio of the approximation of SW and the exact SW. Since the box plot of the ratio for orbit recognition is very similar to that of 3D shape segmentation, we only give the box plot of texture classification in the first row.

Results. Classification accuracies are given in Table 2. For most categories, $k_{SW}$ outperforms competing kernels by a significant margin. The variance of the results over the run is also less than that of its competitors. However, training times are not better in general. Hence, we also provide the results for an approximation of $k_{SW}$ with 10 directions. As one can see from Table 2 and from Figure 3, this approximation leaves the accuracies almost unchanged, while the training times become comparable with the ones of the other competitors. Moreover, according to Figure 3, using even less directions would slightly decrease the accuracies, but still outperform the competitors performances, while decreasing even more the training times.

4.2. Orbit recognition

In our second experiment, we use synthetized data. The goal is to retrieve parameters of dynamical system orbits, following an experiment proposed in (Adams et al., 2017).

Data. We study the linked twist map, a discrete dynamical system modeling fluid flow. It was used in (Hertzsch et al., 2007) to model flows in DNA microarrays. Its orbits can be computed given a parameter $r > 0$ and initial positions $(x_0, y_0) \in [0,1] \times [0,1]$ as follows:

\[
\begin{align*}
    x_{n+1} &= x_n + ry_n(1 - y_n) \mod 1 \\
    y_{n+1} &= y_n + rx_{n+1}(1 - x_{n+1}) \mod 1
\end{align*}
\]

Depending on the values of $r$, the orbits may exhibit very different behaviors. For instance, as one can see in Figure 2, when $r$ is 2, there seems to be no interesting topological features in the orbit, while voids form when $r$ is 1. Following (Adams et al., 2017), we use 5 different parameters $r = 2.5, 3.5, 4.1, 4.3$, that act as labels. For each parameter, we generate 100 orbits with 1000 points and random initial positions. We then compute the PDs of the distance functions to the point clouds with the GUDHI
library (The GUDHI Project, 2015) and we use them (in all homological dimensions) to produce an orbit classifier that predicts the parameter values, by training over a 70%-30% training-test split of the data. Since data points are in $\mathbb{R}^2$, we set the $p$ parameter of $k_{PWG}$ to 4.

**Results.** Since the PDs contain thousands of points, we use kernel approximations to speed up the computation of the Gram matrices. In order for the approximation error to be bounded by $10^{-3}$, we use an approximation of $k_{SW}$ with 6 directions (as one can see from Figure 3, this has a small impact on the accuracy), we approximate $k_{PWG}$ with 1000 random Fourier features (Rahimi & Recht, 2008), and we approximate $k_{PSS}$ using Fast Gauss Transform (Morariu et al., 2009) with a normalized error of $10^{-10}$. One can see from Table 2 that the accuracy is increased a lot with $k_{SW}$. Concerning training times, there is also a large improvement since we tune the parameters with grid search. Indeed, each Gram matrix needs not be recomputed for each parameter when using $k_{SW}$.

4.3. Texture classification

Our last experiment is inspired from (Reininghaus et al., 2015) and (Li et al., 2014). We use the OUTEX00000 data base (Ojala et al., 2002) for texture classification.

**Data.** PDs are obtained for each texture image by computing first the sign component of CLBP descriptors (Guo et al., 2010) with radius $R = 1$ and $P = 8$ neighbors for each image, and then compute the persistent homology of this descriptor using the GUDHI library (The GUDHI Project, 2015). See Figure 2. Note that, contrary to the experiment of (Reininghaus et al., 2015), we do not downsample the images to $32 \times 32$ images, but keep the original $128 \times 128$ images. Following (Reininghaus et al., 2015), we restrict the focus to 0-dimensional persistent homology. We also use the first 50%-50% training-test split given in the database to produce classifiers. Since data points are in $\mathbb{R}^2$, we set the $p$ parameter of $k_{PWG}$ to 4.

**Results** We use the same approximation procedure as in Section 4.2. According to Figure 3, even though the approximation of SW is rough, this has again a small impact on the accuracy, while reducing the training time by a significant margin. As one can see from Table 2, using $k_{PSS}$ leads to almost state-of-the-art results (Ojala et al., 2002; Guo et al., 2010), closely followed by the accuracies of $k_{SW}$ and $k_{PWG}$. The best timing is given by $k_{SW}$, again because we use grid search. Hence, $k_{SW}$ almost achieves the best result, and its training time is better than the ones of its competitors, due to the grid search parameter tuning.

**Metric Distortion.** To illustrate the equivalence theorem, we also show in Figure 4 a scatter plot where each point represents the comparison of two PDs taken from the Airplane segmentation data set. Similar plots can be obtained with the other datasets considered here. For all points, the x-axis quantifies the first diagram distance $d_1$ for that pair, while the y-axis is the logarithm of the distance between PDs in the RKHS induced by either $k_{PSS}$ (blue points), $k_{PWG}$ (green points), $k_{SW}$ (red points) and a Gaussian kernel on $d_1$ (black points).

5. Conclusion

In this article, we introduce the *Sliced Wasserstein kernel*, a new kernel for PDs that is provably equivalent to the first diagram distance between PDs. We provide fast algorithms to approximate it, and show on several datasets substantial improvements in accuracy and training times (when tuning parameters is done with grid search) over competing kernels. A particularly appealing property of that kernel is that it is infinitely divisible, substantially facilitating the tuning of parameters through cross validation.

**Acknowledgements.** We thank the anonymous referees for their insightful comments. SO was supported by ERC grant Gudhi and by ANR project TopData. MC was supported by a chaire de l’IDEX Paris Saclay.
Sliced Wasserstein Kernel for Persistence Diagrams

References


