Eurographics Symposium on Geometry Processing 2018
T. Ju and A. Vaxman
(Guest Editors)

Volume 37 (2018), Number 5

Topological Function Optimization for Continuous Shape Matching

Adrien Poulenard[1], Primoz Skraba[2], Maks Ovsjanikov[1]

[1]LIX, Ecole Polytechnique, CNRS, [2]Jozef Stefan Institute

Abstract

We present a novel approach for optimizing real-valued functions based on a wide range of topological criteria. In particular, we show how to modify a given function in order to remove topological noise and to exhibit prescribed topological features. Our method is based on using the previously-proposed persistence diagrams *associated with real-valued functions, and on the analysis of the derivatives of these diagrams with respect to changes in the function values. This analysis allows us to use continuous optimization techniques to modify a given function, while optimizing an energy based purely on the values in the persistence diagrams. We also present a procedure for aligning persistence diagrams of functions on different domains, without requiring a mapping between them. Finally, we demonstrate the utility of these constructions in the context of the functional map framework, by first giving a characterization of functional maps that are associated with* continuous *point-to-point correspondences, directly in the functional domain, and then by presenting an optimization scheme that helps to promote the continuity of functional maps, when expressed in the reduced basis, without imposing any restrictions on metric distortion. We demonstrate that our approach is efficient and can lead to improvement in the accuracy of maps computed in practice.*

1. Introduction

A core problem in geometry processing consists in quantifying similarity between shapes and their parts, as well as detecting detailed region or point-based correspondences [VKZHCO11, TCL*13, BCBB16]. A common approach for both shape comparison and correspondence consists in computing real-valued (for example, descriptor) functions defined on the shapes and comparing the shapes and their parts by comparing the values of such functions. This includes both computing correspondences by matching in descriptor space, and also, more recently, by computing linear transformations between spaces of real-valued functions using the so-called functional map framework [OBCS*12, OCB*17].

Many existing techniques for comparison of functions on the shapes directly rely on comparing function values, without analyzing the global structure of the functions involved. For example, a descriptor function computed on one shape can have several prominent maxima, whereas on another shape, it can be uniform or with low variance. Intuitively, pairs of functions with dissimilar structural properties can lead to large errors in the correspondence computation. This problem is especially prominent in the context of *functional maps* which are linear transformations between spaces of real-valued functions defined on different shapes. In this case, one is often interested in formulating an objective which would promote mapping indicator functions of connected regions to other such indicator functions without knowing in advance which regions should match. At a high level, such an objective should promote the preservation of *the topological structure* of the functions before and after the mapping.

In this paper, we show how such problems can be solved by efficiently optimizing the topological structure of real-valued functions defined on the shapes, either independently (to promote certain structural properties), or jointly (to enforce similarity between such properties), without resorting to combinatorial search or point-to-point maps. The key to our approach is the manipulation of *persistence diagrams* [CZCG05, Car09]. These diagrams have been shown to summarize the properties of very general classes of topological spaces, including, most relevant to us, real-valued functions defined on the surfaces, and also enjoy several key properties such as being stable under a broad range of perturbations [CSEH07, CCSG*09]. Existing methods, however, concentrate on either *efficiently constructing* persistence diagrams from a given signal [CZCG05, MMS11, CK13] or using them as a tool for, e.g. shape or image comparison [CZCG05, LOC14] or shape segmentation [SOCG10] among many others.

Our main insight is that it is possible to formulate optimization objectives on the persistence diagrams of real-valued functions, regardless of their underlying spatial domain, and to optimize a given function to improve such objectives, via continuous non-linear optimization. For this, we first show, how the derivative of a persistence diagram of a function can be computed with respect to the change in the function values, and then how this computation can be used to efficiently optimize various energies defined on persistence diagrams. Crucially, these computations can be performed in any functional basis, which can significantly improve stability and computation speed. Moreover, our approach allows us to jointly optimize the persistence diagrams of multiple functions, without assuming that they are defined over the same domain.

We apply these insights to first provide a characterization of functional maps associated with continuous point-to-point maps, based on the preservation of persistence diagrams. Unlike previous results, this characterization does not assume that the map is isometric or area-preserving and holds for any continuous point-to-point map. We then propose an optimization scheme that helps to promote continuity of functional maps, even when they are expressed in a reduced basis.

2. Related Work

Persistent (Co)homology Topology is the study of connectivity and continuity, and persistent (co)homology is a natural language for describing it in an applied setting. Persistence has been widely studied [EH10] and has become a central tool for the rapidly developing area of topological data analysis. We provide an overview of persistence in Section 4. Most relevant to our work is the numerous applications it has found in geometry processing e.g. [CZCG05, DLL*10, SOCG10].

In this paper, we are interested in optimizing functions to achieve certain prescribed topological criteria. Using persistence for modifying functions has been studied as topological simplification, which also served as one of the motivations of the original work on persistence [ELZ00]. The simplification problem has been primarily stated as a denoising problem [AGH*09, BLW12], changing the function so that "persistent features" are preserved. Unlike these works, we approach this problem via continuous optimization, which allows us to explicitly modify the function, expressed in an arbitrary basis to optimize topological criteria. Our approach is inspired by Morse theory, which is deeply tied to persistence [EHZ01, CCL03].

Most related to our work is [GHO16]. The main application of their work was the continuation of point clouds for dynamical systems and so the authors concentrate on a different class of complexes. They derive a similar chain rule result to ours, although both their application and perspective are very different. The key tool in their analysis is the study of the inverse map from the persistence diagram back to the underlying topological space. These maps have been studied in other contexts in applied topology, including persistence vineyards [CSEM06] as well as studying generalized minimum spanning trees on random complexes [STY17]. Finally, persistence diagrams have recently also been included in deep network architectures. In [LJY16], the authors use persistence landscapes as a layer in their network architecture with a similar optimization step, but limited to one dimensional signals. In other work [WZWW17, CW17], persistence diagrams are used as features with learned weights, whereas we optimize the underlying filtration, i.e. the diagrams themselves change during optimization.

Continuity for Functional Maps Our main application lies in using persistence diagrams to provide a characterization of functional maps associated with continuous point-to-point correspondences and then proposing a practical optimization scheme that helps to improve the accuracy of functional map computations. The functional map representation and the associated correspodence computation pipeline was first introduced in [OBCS*12] and has since then been extended significantly in, e.g., [KBBV15, HO17, RCB*17, EBC17] among many others (see [OCB*17] for an overview). The key practical advantage of this representation is that it allows to encode correspondences between shapes as small-sized matrices that represent linear transformations between function spaces in some reduced basis. Moreover, computing functional maps can be done efficiently by leveraging tools from numerical linear algebra and manifold optimization, as shown in, e.g., [KBBV15, LRBB17, LRB*16]. One challenge with this approach, however, is that the space of linear functional transformations is much larger than that of point-to-point correspondences, which means that in many cases regularization is necessary to compute accurate maps. This has prompted work on characterizing how various properties of pointwise maps are manifested in the functional domain. For example, the original work showed that area-preserving correspondences lead to orthonormal functional maps [OBCS*12] (Theorem 5.1). Other characterizations have been shown for conformal maps [ROA*13, HO17], isometries [OBCS*12, KBBV15] and for partial correspondences [RCB*17, LRBB17]. More recently, some works have exploited the relation of point-to-point maps to functional maps that preserve pointwise products of functions [NO17, NMR*18]. One large missing piece, however, is to characterize *continuous* point-to-point maps purely in the functional domain, and without making assumptions on the metric preservation. In this work, we fill this gap by precisely characterizing functional maps that arise from continuous point-to-point maps and propose an optimization scheme that allows to promote this continuity.

We also note that another commonly used relaxation for matching problems is based on the formalism of optimal transport, which has recently been used for finding bijective and continous correspondences [SPKS16, MCSK*17, VLB*17]. These techniques have benefited from the computational advances in solving large-scale transport problems, especially using the Sinkhorn method under entropic regularization [Cut13, SDGP*15]. For example, several recent methods in this category [MCSK*17, VLR*17, VLB*17] have been proposed to efficiently find bijective maps, while promoting continuity by iteratively solving optimal transport problems. While related to our work, in these techniques, continuity is measured using point-to-point correspondence or via metric distortion, most commonly by minimizing variance with respect to a previous solution in an iterative scheme. On the other hand, we show that continuity can be expressed in the functional domain directly. Ultimately, we believe that the combination of these techniques, in the spatial and functional domains, can be especially beneficial in tackling difficult problems with strong outliers and discontinuities.

3. Overview

The rest of the paper is organized as follows: in Section 4 we give a brief overview of persistence diagrams and their computation, while concentrating on the specific case of real-valued functions defined on surfaces. In Section 5 we describe a simple algorithm

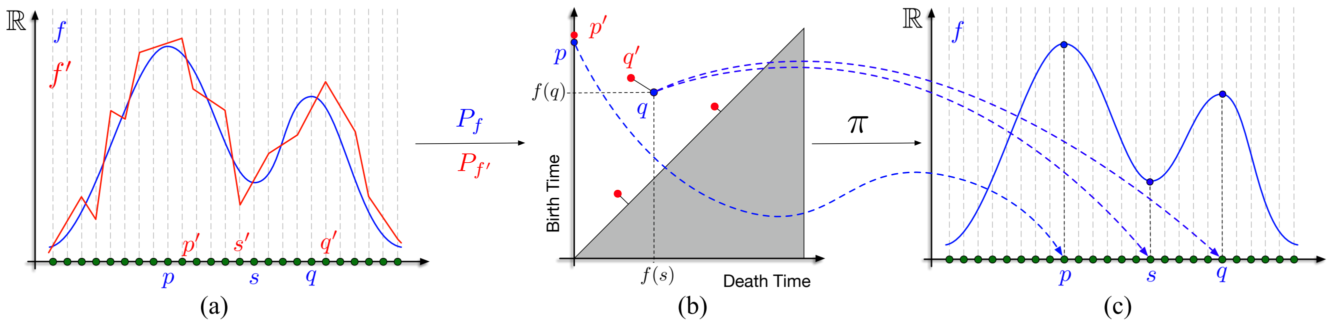


Figure 1: *(a) Two functions (f in blue and f' in red) defined on a 1-d interval. (b) Persistence diagrams of f, f'. (c) The map from the persistence diagram back to the underlying space for f.*

for computing the derivatives of persistence diagrams with respect to function values and outline how functions can be optimized for using various energies based on persistence diagrams. In Section 6 we characterize functional maps associated with continuous point-to-point maps and describe an approach to promote continuity directly in the functional domain. Finally, Section 7 is dedicated to experimental results and practical validation of our methods, while Section 8 concludes the paper. with a summary, a description of limitations and future work.

4. Persistence Diagrams of Real-Valued Functions

We begin with a brief overview of persistence diagrams and their computation [CZCG05, Car09]. We omit a formal introduction as much as possible, referring the reader to [EH10] for a more complete discussion. In a nutshell, persistent homology takes as input a sequence of topological spaces and tracks topological features, specifically homology groups, as they appear and disappear in the sequence. Homology groups, and likewise persistent homology groups are defined for different dimensions, up to the dimension of the underlying space, representing topological features of each dimension.

We concentrate on the 0-dimensional homology associated with real-valued functions defined on surfaces, as this is the setting that is most directly related to our practical scenarios. Namely, throughout our discussion we assume that the topological space is a surface $\mathcal{M}$, represented as an embedded discrete triangle mesh, consisting of N vertices. We also assume that we are given a function $f : \mathcal{M} \to \mathbb{R}$, which is represented as just a N-dimensional vector of real values. To each such function, it is possible to associate a *persistence diagram* of the super-level (resp. sub-level) sets of f, which intuitively captures the number and the relative prominence of all the local maxima (resp. minima) of f. In practice, we define the function on the vertices and linearly interpolate it to the rest of the mesh. The resulting *super-level* set filtration is equivalent (up to homotopy) to the *upper-star filtration* (see [EH10, Chapter VI.3]). Throughout the rest of the paper, for simplicity we only consider filtrations of this type, although our constructions are not restricted to this choice.

More formally, we consider the connected components of $f^{-1}[\alpha,\infty)$ at various values of parameter α. To construct the persistence diagram of a function f, we consider the evolution of the connected components of $f^{-1}[\alpha,\infty)$ as α ranges from ∞ to $-\infty$. The number of points in the persistence diagram equals the number of local maxima of f, i.e., vertices x, such that $f(x) \geq f(y)$ for all y adjacent (in the triangle mesh) to x. For each such vertex x, we will construct a point p in the persistence diagram having two coordinates: its birth and its death. The *birth* of a local maximum x equals simply to $f(x)$. To *death* of x equals the smallest real value β such that $f(x) \geq f(y)$ for all y in the same connected component as x in $f^{-1}[\beta,\infty)$. Note that unless $f(x)$ is the global maximum of f there will always exist some value β such that $f^{-1}[\beta,\infty)$ contains a vertex in the same connected component as x such that $f(y) > f(x)$. Thus, the largest value for which this occurs is called the death of x. By convention, we also declare the death of the global maximum of f on a compact surface, to be the global minimum of f.

The persistence diagram is simply the collection of birth and death times. Each pair defines a point p_i with b_i and d_i coordinates, representing the birth and death time respectively. The *persistence* of the point p in the diagram is half of the difference between its birth and death values which is also the L_∞ distance to the diagonal. In this paper, it will be convenient to view the persistence diagram as a map which takes a topological space and a function to a multi-set of points in $\mathbb{R}^2$.

$$P_f : (\mathcal{M}, f) \to \{(b_i, d_i)\}_{i\in\mathcal{I}} \tag{1}$$

We use $\mathcal{I}$ to denote the index on the points in persistence diagram. In our case, this can simply be an index up to the number of maxima. We can assume that this number is finite as we deal with “nice" functions on, especially discrete, surfaces.

Figure 1 illustrates the persistence diagrams of two functions defined on a 1-dimensional interval. A key result in the theory of persistence is the stability of the diagrams under small perturbations of the function values [CSEH07, CCSG*09], which holds in great generality. Intuitively, given two functions f and g, the distance between their associated persistence diagrams $d(P_f, P_g)$ is bounded by the difference between the functions themselves. Numerous distances between persistence diagrams have been proposed. One of the most commonly used distances is the p-Wasserstein distance:

$$d_{W^p}(P_f, P_g) = \left(\inf_{\substack{\mu: P_f \to P_g \\ \mu \in \text{bijections}}} \sum_{x \in P_f} \|x - \mu(x)\|_p^p \right)^{1/p} \tag{2}$$

This distance is based on the optimal transport between the two dia-

```
Input: Triangle mesh $M$, real-valued function $f$.
Output: Persistence diagram $P_f$ of $f$.
1  Initialization: sort $f$ in descending order.;
2  for each vertex $x$, in descending order of $f(x)$ do
3      if $x$ is local maximum of $f$ in $M$ then
4          add new point to $P_f$ with birth value $f(x)$;
5          parent[$x$] = $x$.
6      else
7          let $y$ be s.t. $f(y)$ is maximal among all parent[$w$], $w$
             adjacent to $x$ and $f(w) > f(x)$;
8          update parent[$w$] = $y, \forall\, w$ in the same connected
             component as $x$ of $f^{-1}[f(x);\infty)$;
9          if parent[w] changed in the previous step then
10             set death of $w$ in $P_f$ to $f(x)$.
11 set death of global max of $f$ in $P_f$ to global min of $f$.
```

Algorithm 1: Computing the persistence diagram of a function f on a triangle mesh.

grams, specifically between the points in the two diagrams. Taking the limit, $p \to \infty$, we recover the *bottleneck distance*:

$$d_B(P_f, P_g) = \inf_{\substack{\mu: P_f \to P_g \\ \mu \in \text{bijections}}} \max_{x \in P_f} \|x - \mu(x)\|_\infty \qquad (3)$$

This is the smallest cost (length of the longest edge) of the perfect matching between the points in P_f and P_g, where each point is also allowed to match with the diagonal. The classical stability theorem [CSEH07] states that $d_B(P_f, P_g) < \|f - g\|_{L^\infty} = \max_x |f(x) - g(x)|$. For the diagrams in Fig. 1, the smallest cost perfect matching would associate p with p', and q with q' while the remaining two red points would be matched with the nearest points on the diagonal. The cost (length of the longest edge) of this matching would be the (relatively small) distance between q and q', capturing the proximity of these two functions.

Both the bottleneck and Wasserstein distances are realized by the matching μ between diagrams. This matching can be computed using the Hungarian algorithm, or any algorithm for computing the minimum weight matching of a bipartite graph.

4.1. Computation of Diagrams and their Distances

Efficiently computing persistence diagrams has been extensively studied, in e.g. [BDM15] (see also [OPT*17] for a recent review of the state-of-the-art). The scalability and practicality of persistent homology computation has vastly increased over the last few years. Computing the 0-dimensional persistence diagram corresponding to the *upper-star filtration* of a piecewise-linear, real-valued function on a triangle mesh is particularly straightforward [ELZ00] and we include it here for completeness. The main steps of this computation are summarized in Algorithm 1. Given a function f, the persistence diagram is computed by first sorting the values of f and then processing each vertex x of the mesh in descending value of f. A new point in the persistence diagram is created whenever a new connected component of $f^{-1}[\alpha, \infty)$ appears, which occurs precisely when $f(x)$ is a local maximum. Otherwise, when two components are merged, the one associated with the smaller value of f dies, and is absorbed into the one associated with the larger value of f. This association between connected components and values of f is maintained in the data structure "parent", which points, for each vertex of the mesh, to the local maximum of f connected to this vertex and having the highest value. Note that line 8. of Algorithm 1 can be implemented efficiently by using a Union-Find data structure, which only requires considering the immediate neighbors w of x. As a result the complexity of entire algorithm is $O(N \log N + N\alpha(N))$, with the former part arising from the sorting of f, while the latter corresponds to processing the vertices and maintaining the association between them and the local maxima, and $\alpha(N)$ is the inverse Ackermann function.

A key aspect of this computation is that for a vertex x that is a local maximum of f, its birth value equals $f(x)$, whereas its death value equals to the value $f(w)$ of some "paired" vertex w which, when processed, merges the connected component of x with that of some other vertex y, where $f(y) > f(x)$. Therefore, when all values of f are distinct, if f is perturbed infinitesimally, the value of the point p in the persistence diagram will change by the amount related to the change at the local maximum and its paired vertex. In the following section, we describe this intuition in detail.

Let us also note that the persistence diagrams of two functions can be compared even if the functions are defined on different domains. In other words, when going from the function f to its diagram P_f all information about the underlying domain on which f is defined is removed. This can be useful in our applications, where we will use persistence diagrams as a way to compare and align functions defined on different domains. Finally, we note that persistence diagrams are *provably stable* under perturbations of the function values [CSEH07], which has motivated their use in many settings, including shape analysis, e.g., [SOCG10, LOC14].

5. Derivatives and Optimization of Persistence Diagrams

We now derive how the persistence diagram changes as we change the underlying functions. Our approach is based on the observation that persistence diagrams can be thought of, in the simplest case, as extensions of the max operator. We follow the general common approach used to compute derivatives of such operators, for example in the case of max pooling in convolutional neural networks [GBCB16]. These derivatives are most often discontinuous, but we show that generically, they are locally well defined, and, just like the max operator, can be successfully optimized for within more complex energies in practice. Finally, we note that all the results in this section can be applied to persistence diagrams of any dimension (as well as to any functionals of persistence diagrams).

Our goal is to understand how the diagram changes as we change the function. Let the filtration function f, depend on a set of parameters which we denote by α_j. To optimize these parameters, we must derive formulas for

$$\frac{\partial b_i}{\partial \alpha_j} \quad \text{and} \quad \frac{\partial d_i}{\partial \alpha_j},$$

for all points in the diagram (b_i, d_i). Our key tool is the existence of a map π from the points in the diagram to pairs of vertices in the space, i.e.

$$\pi : (b_i, d_i) \mapsto (v_b, v_d)$$

We first define a possibly non-unique map from points in the persistence diagram to pairs of simplices

$$\zeta : (b_i, d_i) \mapsto (\sigma, \tau)$$

A finite simplicial filtration can always be extended to a *total order*. For example, one way is to consider lexographical ordering of the vertices to order simplices which share the same function value. In the total ordering, each birth and death time are distinct, hence there is precisely one simplex which corresponds to any birth time (and precisely one simplex which corresponds to a death time respectively). Note that these times refer to the index in the total order. This correspondence defines ζ, which is simply the pairing returned by the standard persistence algorithm [CZCG05]. In our case, each birth time corresponds to the value of a specific vertex, whereas the death time is associated with the edge of the mesh, which merges two connected components when it is added.

Since we are considering a super-level set (upper-star) filtration, edges are added implicitly in the filtration, once both vertices are included. This allows us to define a map from each simplex to one of its vertices. In the case of vertices, this is simply the vertex itself, whereas in the case of edges, it is given by the vertex with the smaller function value:

$$\eta : \sigma \mapsto v_\sigma, \text{ where } \eta(\sigma) = \arg\min_{v \in \sigma} f(v)$$

In general this is not unique, but we can always choose an arbitrary fixed vertex. Finally, we define

$$\pi_b = \eta \circ \zeta(b_i), \qquad \pi_d = \eta \circ \zeta(d_i)$$
$$\pi = (\pi_b, \pi_d)$$

Algorithm 1 implicitly computes this map (see Section 5.2 for details). This map depends on numerous choices, however as we saw in the previous section, we can assume that the function values are unique at each vertex. This can be achieved through an infinitesimal perturbation of the function, either implicitly or explicitly. Finally, while π is defined for each point in the diagram, by applying it to *every point*, we can induce a map from a diagram to a multi-set of vertices. We then obtain the following two results.

Lemma 1 If the vertex function values are distinct, then π is unique.

Proof See Appendix. □

Note that this does not imply that there is a one-to-one correspondence between points in the diagram and pairs of vertices, as multiple points can map to the same vertex. However, we do have the following result.

Lemma 2 If the vertex function values are distinct, there exists a neighborhood where π is constant.

Proof See Appendix. □

To define the neighborhood recall that we can consider the persistence diagram as a map from a pair (X, f) to a multi-set of points (Eq. 1). The neighborhood is defined as all the diagrams with the domain (X, g) such that $||f - g||_\infty$ is sufficiently small. We note that stability [CSEH07] implies that the corresponding multi-sets of points are close with respect to the bottleneck distance. The existence of this neighborhood allows us to derive analytic expressions the derivatives. First, we observe that

$$f(\pi_b(b_i)) = b_i \qquad f(\pi_d(d_i)) = d_i \qquad \forall i \tag{4}$$

Since π is constant, it follows that

$$\frac{\partial b_i}{\partial \alpha_j} = \frac{\partial f(\pi_b(b_i))}{\partial \alpha_j} = \frac{\partial f(v_i)}{\partial \alpha_j} = \frac{\partial f}{\partial \alpha_j}(v_i) \tag{5}$$

In other words, ***the derivative is equivalent to the derivative of the function evaluated at the image of the map*** π_b. The derivation for d_i is analogous.

5.1. Application to optimization

In our application we would like to minimize some functional $\mathcal{F}$ of an input diagram. Using the the chain rule, we obtain

$$\begin{aligned}\frac{\partial \mathcal{F}}{\partial \alpha_j} &= \sum_{i \in \mathcal{I}} \frac{\partial \mathcal{F}}{\partial b_i} \cdot \frac{\partial b_i}{\partial \alpha_j} + \frac{\partial \mathcal{F}}{\partial d_i} \cdot \frac{\partial d_i}{\partial \alpha_j} \\ &= \sum_{i \in \mathcal{I}} \frac{\partial \mathcal{F}}{\partial b_i} \cdot \frac{\partial f}{\partial \alpha_j}(\pi_b(b_i)) + \frac{\partial \mathcal{F}}{\partial d_i} \cdot \frac{\partial f}{\partial \alpha_j}(\pi_d(d_i))\end{aligned}$$

$\frac{\partial \mathcal{F}}{\partial x_i}$ and $\frac{\partial \mathcal{F}}{\partial y_i}$ must be derived for each functional separately. We derive the complete formula for two specific cases. We first consider the bottleneck distance.

Lemma 3 For almost all persistence diagrams P_f and P_g, the point $x \in P_f$ whose matching achieves the bottleneck distance is unique and constant in some ε-neighborhood of P_f (in the bottleneck metric).

Proof See Appendix. □

The neighborhood here again refers to the space of persistence diagrams (here we do not need to restrict the space of diagrams as in Lemma 2). If P_f and P_g are generic, the matching is unique in a sufficiently small neighborhood of nearby diagrams. Since nearby diagrams may have different numbers of points, one needs to be careful to ensure the lemma holds (see Appendix).

We consider the functional of the bottleneck distance to a fixed diagram, denoted by $\mathcal{F}_B$. Let $p_{max} = (b_{max}, d_{max})$ be the point in the diagram whose matching realizes the maximum in the bottleneck matching μ. If $p_i = p_{max}$ and $|b_{max} - \mu(b_{max})| > |d_{max} - \mu(d_{max})|$

$$\frac{\partial \mathcal{F}_B}{\partial b_i} = \begin{cases} -1 & b_i > \mu(b_i) \\ 1 & b_i < \mu(b_i) \end{cases}$$

and 0 otherwise. A similar expression holds for $\frac{\partial \mathcal{F}_B}{\partial d_i}$ if $|b_{max} - \mu(b_{max})| < |d_{max} - \mu(d_{max})|$. Note that if both are equal then the distance is 0, where the derivative can be defined as 0 (since the distance cannot be negative). Therefore, the derivative can be written as two cases: if $|b_{max} - \mu(b_{max})| > |d_{max} - \mu(d_{max})|$, then

$$\frac{\partial \mathcal{F}_B}{\partial \alpha_j} = \operatorname{sgn}\left(\mu(b_{max}) - b_{max}\right) \frac{\partial f}{\partial \alpha_j}(\pi_b(b_{max})) \tag{6}$$

and if $|b_{max} - \mu(b_{max})| < |d_{max} - \mu(d_{max})|$,

$$\frac{\partial \mathcal{F}_B}{\partial \alpha_j} = \operatorname{sgn}\left(\mu(d_{max}) - d_{max}\right) \frac{\partial f}{\partial \alpha_j}(\pi_d(d_{max})). \tag{7}$$

We also derive an equivalent result for the squared 2-Wasserstein distance. Again, we assume by genericity that the solution to the matching is unique and constant in a neighborhood. The squared 2-Wasserstein distance is given by

$$
\begin{aligned}
d^2_{W^2}(P_f, P_g) &= \inf_{\substack{\mu: P_f \to P_g \\ \mu \in \text{bijections}}} \sum_{p \in P_f} \|p - \mu(p)\|_2^2 \\
&= \inf_{\substack{\mu: P_f \to P_g \\ \mu \in \text{bijections}}} \sum_{p \in P_f} (p_b - \mu(p_b))^2 + (p_d - \mu(p_d))^2
\end{aligned}
$$

Minimizing this distance gives the functional, $\mathcal{F}_{W^2}$, whose derivative is

$$
\begin{aligned}
\frac{\partial \mathcal{F}_{W^2}}{\partial b_i} &= \frac{\partial}{\partial b_i} \left(\sum_{i \in \mathcal{I}} (b_i - \mu(b_i))^2 + (d_i - \mu(d_i))^2 \right) \\
&= 2(b_i - \mu(b_i))
\end{aligned}
$$

Putting this together,

$$
\frac{\partial \mathcal{F}_{W^2}}{\partial \alpha_j} = 2 \sum_{i \in \mathcal{I}} (b_i - \mu(b_i)) \frac{\partial f(\pi_b(b_i))}{\partial \alpha_j} + (d_i - \mu(d_i)) \frac{\partial f(\pi_d(d_i))}{\partial \alpha_j} \tag{8}
$$

where b_i and d_i are functions of α_j (see Eq. 4). We conclude this section by noting that most functionals for persistence diagrams map points in $\mathbb{R}^2$ to real valued functions. For most cases, it should be possible to compute a closed form for the derivative, for example in the case of persistence landscapes or specific choices of kernels on the space of persistence diagrams.

5.2. Computing Derivatives

The derivatives derived above can be efficiently computed by appropriately modifying Algorithm 1. We first need to compute π, the map from the persistence diagram to the vertices. Let p be the point added to P_f in line 4. When the point is created, we also add the pair (p,x) to π. Likewise in line 10, we again add the pair (p,x) to π. Note that these are two different vertices, as the x in Line 4 is a local maxima, while the x in line 10. connects two previously disjoint components (i.e. in our case a saddle point).

To compute the derivative, we must be able to evaluate the derivative of the function at a given vertex in the underlying space, i.e. $\mathcal{M}$. The derivative can be obtained by evaluating the derivative of the function at the image of π for that point. Therefore, by using the modified Algorithm 1 and computing the minimum weight matching μ with respect to the chosen distance, we can evaluate π and μ for every point in the persistence diagram P. For the bottleneck distance, we can then directly compute the derivative by evaluating Equations 6 or 7. The derivative for the 2-Wasserstein distance can likewise be computed by evaluating Equation 8.

In our derivation, we assumed a genericity condition by requiring the vertex function values be unique. While this is true for any one function (by infinitesemal perturbation), as we optimize the function, this assumption might be violated. However, in the case of non-uniqueness, we can make a choice randomly. In our implementation, the continuous optimization scheme such as L-BFGS explicitly checks that the energy decreases at every iteration, ensuring that the algorithm converges. Furthermore, in our applications we optimize over a reduced functional basis consisting of smooth functions, which significantly improves the robustness with respect to pathological behavior. In practice, the optimization schemes we use converge to local minima remarkably consistently. Nevertheless, we leave the theoretical study of guarantees of convergence as interesting future work.

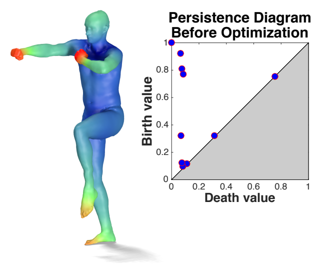
6. Applications: Continuity in Functional Maps

We propose to apply the ideas presented above for improving functional maps. For this, we first provide a novel characterization of functional maps arising from *continuous* point-to-point maps, purely in the functional domain, i.e., without making reference to point-to-point correspondences. We also prove that our characterization is complete: i.e., all functional maps (linear transformations across function spaces) that satisfy our condition exactly must arise from continuous point-to-point maps. We then present an energy, which can be optimized using standard non-linear continuous optimization techniques, such as L-BFGS, and which promotes continuity directly in the functional domain, even when maps are expressed in a reduced basis. In this section, we describe these constructions in detail, and present the main experimental results in Section 7.

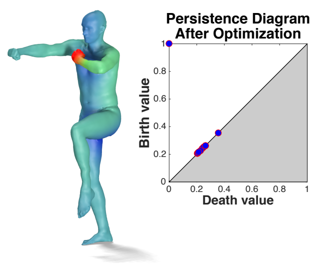
Our main observation is that the information encoded in persistence diagrams can be used to improve the computation of correspondences between shapes using the so-called functional map framework (see [OCB*17] for an recent overview). Works in this domain are based on the idea that when looking for a point-to-point map $T : \mathcal{M} \to \mathcal{N}$ between two shapes $\mathcal{M}$ and $\mathcal{N}$, it is often easier to consider the associated pull-back of real-valued functions $T_F : \mathcal{F}(\mathcal{N}) \to \mathcal{F}(\mathcal{M})$, where $\mathcal{F}(\mathcal{N})$ is the space of real-valued functions on shape $\mathcal{N}$. For a given map T, T_F is defined simply via composition $T_F(f) = f \circ T$, where f is any real valued function $f : \mathcal{N} \to \mathbb{R}$. Since T_F is a linear operator between function spaces, whenever both the domain and range can be endowed with a functional basis, this operator can be written as a change of basis matrix $\mathbf{C}$, also called a functional map matrix, whose size depends on the size of the chosen basis.

The basic pipeline for computing functional maps introduced in [OBCS*12], and extended in follow-up works, aims at recovering the functional map matrix $\mathbf{C}$ directly and then using it to estimate the underlying map T. The general approach follows the following steps (see Section 2.4.4 in [OCB*17]):

1. Construct a set of basis functions, consisting e.g. of the eigenfunctions corresponding to the $k_{\mathcal{M}}$ and $k_{\mathcal{N}}$ smallest eigenvalues of the Laplace-Beltrami (LB) operators on the source and target shapes $\mathcal{M}$ and $\mathcal{N}$, stored in matrices $\Phi^{\mathcal{M}}$ and Φ^{N} respectively.
2. Compute some k_d descriptor functions on the source and target shapes, express them in the corresponding bases and store the coefficients as columns of matrices $\mathbf{A}$ and $\mathbf{B}$, of size $k_{\mathcal{M}} \times k_d$ and $k_{\mathcal{N}} \times k_d$ respectively.
3. Compute the optimal *functional map* matrix $\mathbf{C}$ of size $k_{\mathcal{N}} \times k_{\mathcal{M}}$, that aligns the descriptor functions and commutes with the Laplace-Beltrami operators on the two shapes. That is: $\mathbf{C} = \arg\min_{\mathbf{X}} \|\mathbf{XA} - \mathbf{B}\| + \varepsilon \|\Lambda^{\mathcal{N}}\mathbf{X} - \mathbf{X}\Lambda^{\mathcal{M}}\|$. Alternatively, one can use several extensions, such as manifold-constrained optimization [KBBV15], robust regularization [LRB*16] or requiring



(a) *Function before optimization*



(b) *Function after optimization*

Figure 2: *Topological optimization for removing local maxima from a function: (a) a real-valued function on a surface and its persistence diagram, (b) function and its diagram after topological optimization, where we penalize the persistence of all, except the first most prominent maximum.*

the map to commute with multiplicative operators with respect to descriptor functions [NO17] among others.

4. Convert the functional map **C** to a point-to-point map.

A key advantage of the functional map representation is that the optimization step 3. can be solved efficiently using robust numerical linear algebra tools, and in the most basic case reduces to solving a simple linear system of equations. Unfortunately, without additional regularization, the computed functional map might not correspond to any "natural" point-to-point map. Therefore, several approaches have been proposed to enforce desired properties on functional maps. For example, it was shown in the original article [OBCS*12] that functional maps arising from area-preserving point-to-point maps must be orthonormal.

More recently several works have used the fact that functional maps that correspond to pointwise maps must also preserve pointwise products between functions [NO17]. In practice, however, we are often interested in *continuous* pointwise correspondences and translating the continuity of the point-to-point map into a constraint on the functional map has not been straightforward. To this end, we first establish and then exploit the following theorem.

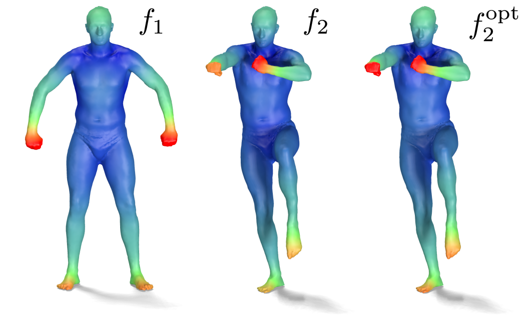
Theorem 1 An invertible linear functional map T_F corresponds to a *continuous* bijective point-to-point map *if and only if* both T_F and its inverse preserve pointwise products of pairs of functions, and moreover both T_F and its inverse preserve the persistence diagrams of all real-valued functions. In other words:

$$d_B(P_f, P_{T_F(f)}) = 0,\ \forall\, f. \tag{9}$$

Note that preservation of products ensures that T_F corresponds to a point-to-point map, whereas preservation of persistence diagrams guarantees that the underlying map is continuous.

Proof See Appendix. □

The key observation in the proof of Theorem 1 is that a functional map associated with a point-to-point bijection will correspond to a pull-back by a *continuous* point-to-point map if it maps indicators of functions of connected regions to indicators functions of connected regions. Moreover, persistence diagrams provide a very convenient way to test whether an indicator function corresponds to some connected region, by simply checking whether it



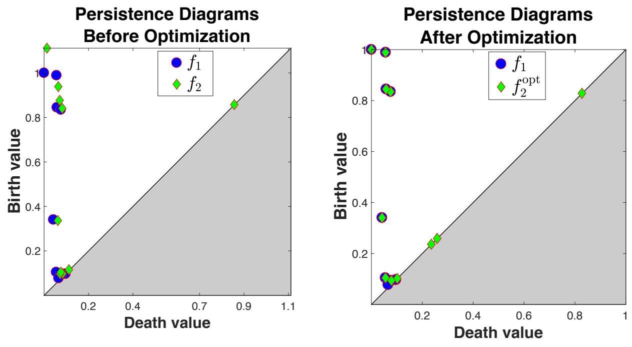


Figure 3: *Topological function alignment. Left and middle: initial functions f_1, f_2 on two shapes (top), and their persistence diagrams (bottom). Right: function f_2^{opt} on the second shape after aligning its persistence diagram with that of f_1, without any cross-shape correspondence (top) and the resulting aligned diagrams (bottom). Note the change in values on the hands.*

has more than one prominent local maximum. The persistence of local maxima gives a stable way to test for this criterion for an *arbitrary* (not necessarily binary, or even positive) function, since an indicator function of a connected region, perturbed by noise, will still have a single prominent local maximum, with other points close to the diagonal on the persistence diagram. This stability directly follows from the stability guarantees of persistence diagrams, which have been established under very broad conditions [CSEH07, CCSG*09].

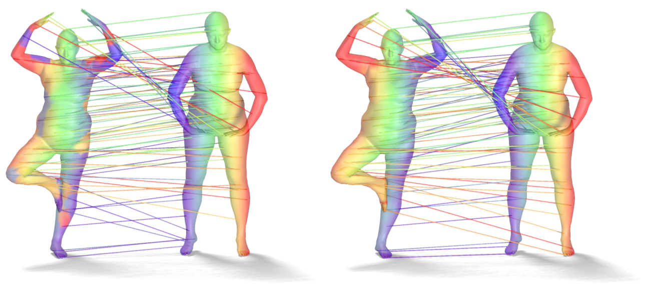
Therefore, we propose to exploit this observation by defining the following (non-linear) energy on functional maps, expressed in an arbitrary basis:

$$E_{\text{cont}}(\mathbf{C}) = \sum_r d_B(P_{\Omega_r}, P_{\mathbf{C}(\Omega_r)}), \tag{10}$$

where r is an index over some set of connected regions defined on the source shape $\mathcal{M}$, Ω_r is the characteristic (indicator) function of region r, and $\mathbf{C}(\Omega_r)$ is the image of indicator function onto the target shape $\mathcal{N}$ via the functional map C. For simplicity we write $\mathbf{C}(\Omega_r)$ instead of the expression in the reduced basis, which should read $\Phi^{\mathcal{N}}\mathbf{C}((\Phi^{\mathcal{M}})^{+}\Omega_r)$, where $+$ is the pseudo-inverse.

Of course, a symmetric energy can be optimized for using a map between the target and source shape.

With this energy at hand, Theorem 1 can be restated simply to say that a bijective point-to-point map T is continuous if and only if both the pullback T_F by T and its inverse satisfy $E_{\text{cont}}(T_F) = 0$, assuming the sum in Eq. 10 is over *all* connected regions r.



(a) *Initial correspondence* **(b)** *Correspondence after optimization*

Figure 4: *(a) Noisy input functional map converted to a pointwise correspondence (b) Same map after topological optimization. Note the drastic reduction in discontinuities. Colors encode the x coordinate function of the target shape and its pull-back on the source.*

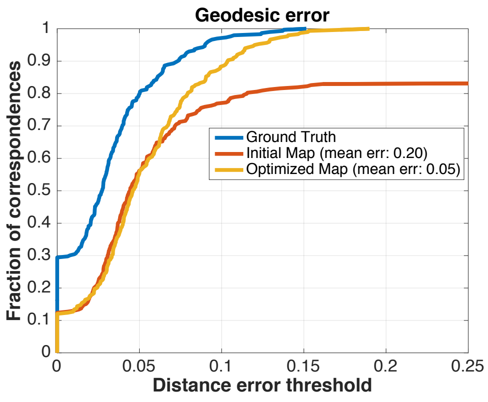


Figure 5: *Quantitative evaluation of the functional maps before and after topological optimization, shown in Fig. 4.*

Intuitively, E_{cont} measures the uncertainty in converting a functional map to a point-to-point map. For example, the image of an indicator function of a connected region or even a delta function via a computed functional map can have multiple prominent local maxima, which means that during the point-to-point conversion, the correspondence can oscillate between multiple possible solutions creating a highly discontinuous pointwise map. This can especially occur when a functional map corresponds to a blending of multiple pointwise maps, which can arise e.g. due to the symmetry present in the shapes. On the other hand, when optimizing for functional map using the energy in Eq. 10 we expect that the image of an indicator function of a region corresponds to a function with exactly one prominent maximum, which eventually should lead to a more continuous point-to-point map.

In practice, we observe that it can be more efficient to replace E_{cont} with a simplified energy:

$$E_{\text{persist}}(\mathbf{C}) = \sum_r \text{Pers}(P_{\mathbf{C}(\Omega_r)}), \tag{11}$$

where the sum is again over connected regions r on the source shape and $\text{Pers}(P)$ is defined as the sum of the squares of persistence values (i.e., the distances to the diagonal) of all points in the diagram P, except for the one with the highest persistence. Intuitively, $\text{Pers}(P_f)$ penalizes all but the prominent local maxima of the function f, with the strength equal to the square of the distance to the diagonal.

Our main objective therefore is to use the energy $E_{\text{persist}}(\mathbf{C})$ to optimize a functional map $\mathbf{C}$ and especially to use it to infer functional maps that arise from *continuous* point-to-point maps. For this, we observe that the ability to differentiate persistence diagrams allows us to compute the gradient of $E_{\text{persist}}(\mathbf{C})$ with respect to the entries of the matrix $\mathbf{C}$. This, in turn, allows us to use quasi-Newton methods such as BFGS, which, as we show below, provide an efficient way to improve functional maps. Specifically, we solve the following continuous optimization problem:

$$\min_{\mathbf{C}} E_{\text{persist}(\mathbf{C})}, \tag{12}$$

where the functional map is represented in the reduced LB basis. In general, this is highly non-convex problem, and we use the recomputed functional map as an initialization in an iterative quasi-Newton scheme. For robustness, we also add a constraint that the functional map $\mathbf{C}$ should map the indicator (constant = 1) function of the source shape to the indicator on the target, and use a projected L-BFGS solver to optimize Eq. (12).

The derivative in this case is a special case of the 2-Wasserstein case in Equation 8. If we match to a diagram with only one point corresponding to a global maximum, all the points map except the global maximum map to the diagonal so

$$\mu(b_i) = \mu(d_i) = \frac{b_i + d_i}{2}$$

Substituting into Equation 8, the derivative can be written as

$$\frac{\partial E_{\text{persist}}}{\partial \alpha_j} = \sum_{i \in \mathcal{J}} (d_i - b_i) \left(\frac{\partial f}{\partial \alpha_j}(\pi_d(d_i)) - \frac{\partial f}{\partial \alpha_j}(\pi_b(b_i)) \right)$$

where $\mathcal{J}$ represents all the points in the diagram except the most persistent one.

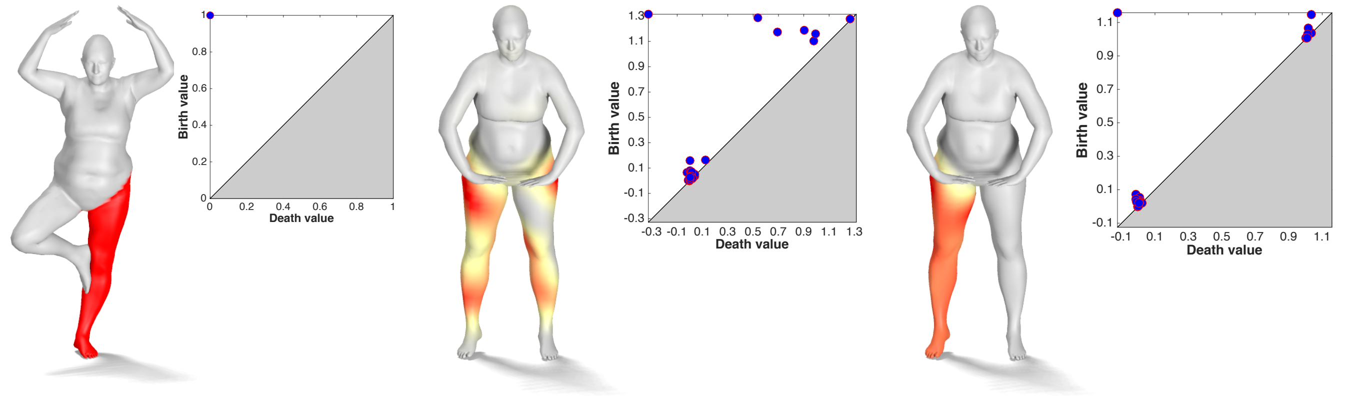
7. Experimental Results

In this section, we describe some practical aspects of the implementation of our topological optimization approach and show results in several applications.

Throughout all of our experiments we assume that shapes are represented as triangle meshes, and functions are defined on their vertices. We also use the standard discretization of the Laplace-Beltrami operator $L = A^{-1}W$, where A is the lumped area matrix and W is the matrix of cotangent weights [PP93, MDSB03]. We use this operator to define a basis for real-valued functions by computing the eigenfunctions corresponding to the k smallest eigenvalues, $L\varphi_i = \lambda_i \varphi_i$. Then we express any real-valued function as a linear combination $f = \sum_{i=1}^{k} a_i \varphi_i$, where a_i are scalar weights. Thus, when optimizing for a function f, we consider the scalar weights a_i as the unknowns.

7.1. Topological Function Simplification

To illustrate the effect of our topological optimization, we first use it to "simplify" a real-valued function on a surface, by removing



(a) *Indicator function of a region* **(b)** *Image before optimization* **(c)** *Image after optimization*

Figure 6: *Topological optimization for improving functional maps: (a) indicator function of a region on a source shape and its persistence diagram, (b) the image of this function via the initial functional map onto the target shape and its persistence. Notice multiple prominent local maxima. (c) Image of the same function after optimizing the functional map.*

all but one most prominent local maxima. Namely, we first construct a real-valued function on one of the shapes from the SCAPE dataset [ASK*05] by computing the Heat Kernel Signature (HKS) [SOG09] for a $t = 0.01$. This function and its associated persistence diagram are shown in Figure 2a. We then simplify this function by minimizing an energy, which penalizes the square of the persistence of all but first most prominent local maxima (which is equivalent to $\mathrm{Pers}(P_f)$ introduced in Eq. 11 above), using $k = 40$ eigenfunctions of the Laplace-Beltrami operator as the basis for the function. The result of the optimization is shown in Figure 2b. Note that the resulting function has only one prominent local maximum. Remark also that our optimization does not impose the *location* of the final local maximum, but simply promotes its uniqueness. Finally, we note that our topological optimization does not assume the manifold structure of triangle meshes, and, once the functional basis is computed, can be applied to arbitrary graphs. The optimization converges after 81 iterations of L-BFGS in 2.26 seconds on a machine with 2.6 GHz Intel Core i7 CPU using a MATLAB implementation of L-BFGS with the analytic gradient of persistence diagrams described in Section 5.

7.2. Topological Function Alignment

Next, we illustrate how our topological function optimization can be used to align the values of two functions defined on different domains without the knowledge of any (functional or point-to-point) correspondences. For this, we first compute two functions, f_1, f_2 corresponding to the HKS for the same value $t = 0.01$ on two different shapes, shown in Figure 3 (top, left and middle). Note that since the shapes is not fully intrinsically symmetric, the function f_2 has two prominent local maxima on the hands that have different values, while the two most prominent maxima of f_1 are closer together, since since the undeformed shape is closer to being symmetric. We then modify f_2 so that the bottleneck distance $d_B(P_{f_1}, P_{f_2})$ is minimized, while keeping f_1 fixed. Let us stress that this energy does not require a map between the two domains, and we can optimize f_2 so that its persistence diagram aligns with that of f_1 by only considering the diagrams themselves. The result of this optimization, f_2^{opt} is shown in Figure 3 (right), where both the function becomes symmetric and the persistence diagrams align nearly perfectly. In this case, the optimization is done again, using $k = 40$ eigenfunctions of the Laplace-Beltrami operator, converges after 29 iterations of L-BFGS and takes 0.84 seconds.

7.3. Functional Map Improvement

In our next experiment, we show how topological function optimization can be used to improve functional maps and to promote the continuity of the recovered point-to-point maps directly in the functional space, without enforcing conditions on area preservation or conformality. For this, we first consider a noisy point-to-point map, which is obtained by a strong perturbation of the symmetric correspondence (i.e., mapping left to right) between a pair of shapes from the FAUST dataset [BRLB14]. The initial correspondence is shown in Figure 4a. We then convert this map to a functional map representation using $k = 80$ eigenfunctions of the Laplace-Beltrami operator, resulting in a matrix $\mathbf{C}$ of size 80×80. We then optimize this functional map using L-BFGS on the energy E_{persist} described in Eq. 11. To construct the connected regions, required in the sum in Eq. 11, we randomly sample n points on the source shape and construct their intrinsic Voronoi diagram. We then use an iterative scheme, where we optimize a functional map with an increasing number of regions $n = 5, 15, \ldots, 45$. Between iterations we project a functional map to a point-to-point one to remain close to the desired solution space. This procedure is reminiscent to the ICP refinement approach proposed in the original functional maps work [OBCS*12], but instead of promoting area preserving maps, it aims to promote continuous maps, without enforcing any constraints on the metric distortion.

The final functional map, converted to a point-to-point map, is shown in Figure 4b. Note that the resulting map does not have the large patch discontinuities present in the original one. We also evaluated the quality of the initial and optimized maps with respect to the ground truth map. Figure 5 shows the percent of point-to-point

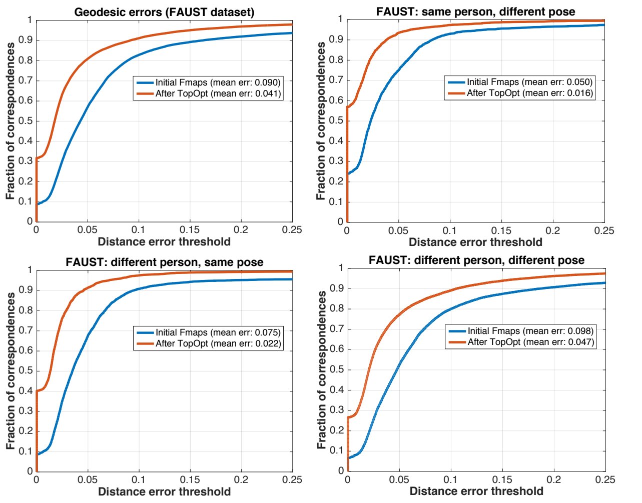


Figure 7: *Quantitative evaluation of correspondences computed on 100 random pairs of shapes from FAUST dataset [BRLB14], using the basic functional maps pipeline in [NO17] before and after our topological functional map optimization.*

correspondences below a certain threshold, following the evaluation protocol introduced in [KLF11]. We also plot the errors of in the ground truth map, which are caused by its representation as a reduced-size functional map. Note the significant improvement of the map after topological optimization, especially with respect to large errors, which are eliminated by the topological optimization.

Finally, to illustrate the effect of the optimization, we consider the indicator function of a connected region on the source shape, and its associated persistence diagram, shown in Figure 6a. We then show its image on the target shape via the initial functional map in Figure 6b, which contains multiple significant local maxima. Finally, we show the image of the same function onto the target but via the functional map after our optimization in Figure 6c. Note the intuitive "connectivity" of this function and its unique prominent local maximum.

7.4. Improvement of Computed Functional Maps

We also use our topological optimization procedure to improve functional maps computed using descriptor preservation constraints, as described in [NO17].

For this, we consider 100 pairs of shapes, taken at random from the FAUST dataset [BRLB14], and for each shape pair introduce one landmark point (in the middle of the right leg). We then compute descriptor functions, which consist of Wave Kernel Signatures [ASC11] and Wave Kernel Map (to enforce the landmark) sampled at 20 different times. Finally, we compute the functional map using a linear system of equations based on multipliciative operators described in [NO17]. We then optimize the resulting functional maps using our topological optimization approach by minimizing Eq. 11. During topological optimization, we again use an iterative procedure where we sample 5 points and compute their intrinsic Voronoi diagram. We then optimize Eq. 11 for 12 iterations of L-BFGS and convert the optimized functional map to a point-to-

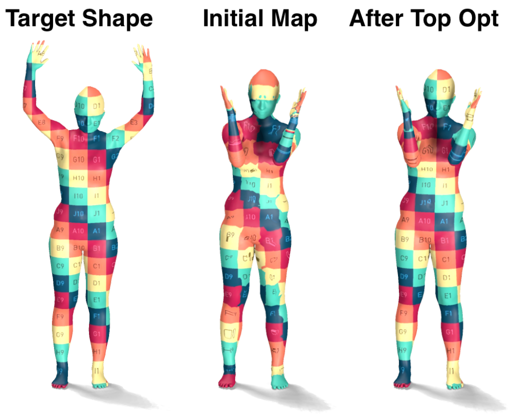


Figure 8: *Texture transfer from a target shape (left) onto the source using a functional map converted to a point-to-point one before (middle) and after (right) topological functional map optimization.*

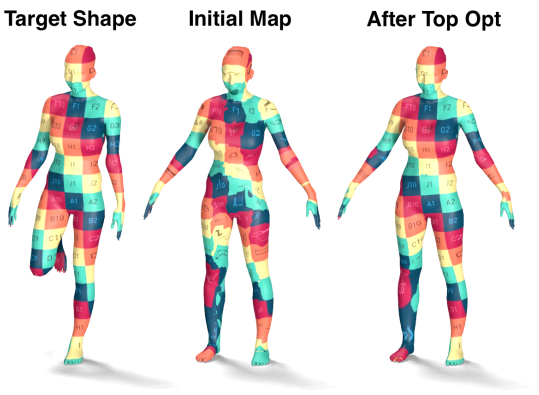


Figure 9: *Texture transfer from a target shape (left) onto the source using a functional map converted to a point-to-point one before (middle) and after (right) topological functional map optimization.*

point one. We found that the last step helps to improve the quality of the map, as it restricts the optimization from drifting too far from point-wise maps. An alternative would be, e.g., to enforce preservation of pointwise products of functions, as suggested in [NO17].

Figure 7 shows the quantitative evaluation of the functional maps converted to a point-to-point maps (using the basic procedure introduced in [OBCS*12]) on 100 shape pairs before and after topological optimization. Note the significant improvement in quality across all categories in the dataset.

We also illustrate the quality of the computed functional maps converted to point-to-point maps in Figures 8, 9 and 10 via texture transfer. Note the improvement in the continuity in the maps after topological optimization. We stress that unlike previous works, such as [MCSK*17, VLR*17], we never enforce continuity of point-to-point maps. Instead, the optimization is done purely in the "functional domain" by minimizing Eq. 11 directly. Moreover, we do not enforce any prior on the metric distortion, such as requesting the maps to be isometric, conformal or area-preserving, but instead

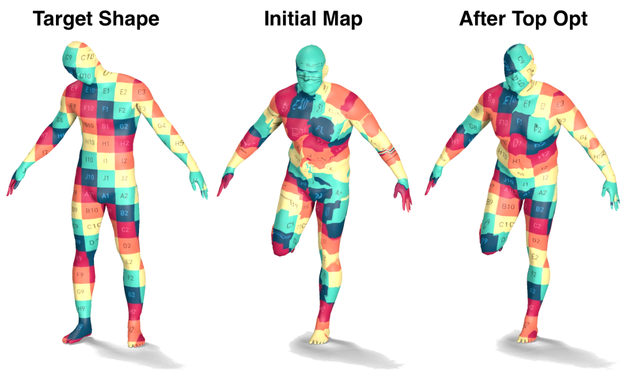


Figure 10: *Texture transfer from a target shape (left) onto the source using a functional map converted to a point-to-point one before (middle) and after (right) topological optimization.*

only promote continuity, while operating with functional maps in a reduced basis. Since we use a basic approach for converting functional maps to point-to-point ones, some high-frequency noise can still be present in the resulting correspondence. Nevertheless, post-processing of these maps using techniques such as [MCSK*17] can certainly improve the results further.

8. Conclusion, Limitations & Future Work

In this paper we presented an approach for optimizing real-valued functions defined on shapes, based on a wide variety of topological criteria. Our main observation is that previously proposed persistence diagrams can be differentiated with respect to the changes in function values, and as such optimized for using continuous optimization techniques. We use this procedure for both aligning diagrams of functions defined on possibly different domains, i.e., reducing their bottleneck or Wasserstein distances, and also for simplifying a given function to remove undesired topological features. Finally, we show how this analysis can be used in the context of functional map computations, first by characterizing *continuous* point-to-point maps directly in the functional domain and then presenting an optimization scheme that helps to promote continuity of functional maps in the reduced basis.

In the future, it would be very interesting to provide rigorous stability guarantees and bounds on how our characterization behaves for *approximately* continuous maps, especially when expressed as functional maps in the reduced basis. It would also be interesting to explore other functionals on persistence diagrams both in shape analysis and in the more broad data analysis applications, for example on images or graphs. Moreover, our functional map improvement procedure is not well-adapted to partial shapes and it would be interesting to see how it can be used jointly with objectives such as promoting bijectivity or partiality, or with existing approaches that promote continuity of pointwise maps, e.g. [MCSK*17, VLR*17]. In addition, it would be interesting to study questions of convergence of our optimization problems, depending on the choice and size of the basis, as well as stability with respect to changes in the shape, taking advantage of the theoretical stability guarantees available for persistence diagrams.

Finally, we believe that the interaction between topological data analysis techniques, including persistence diagrams, with function-based approaches, including the functional maps framework, is a very fruitful and largely unexplored area for future work in general.

Acknowledgements The authors would like to thank Frederic Chazal for helpful discussions. Parts of this work were supported by the Jean Marjoulet chair from Ecole Polytechnique, a Google Focused Research Award, ERC Starting Grant No. 758800 (EXPROTEA), ARRS project N1-0058 (TOPREP) and H2020-MSCA-RISE-2015-6911 (RENOIR).

References

[AGH*09] ATTALI D., GLISSE M., HORNUS S., LAZARUS F., MOROZOV D.: Persistence-sensitive simplification of functions on surfaces in linear time. *Presented at TOPOINVIS 9* (2009), 23–24. 2

[ASC11] AUBRY M., SCHLICKEWEI U., CREMERS D.: The Wave Kernel Signature: A Quantum Mechanical Approach to Shape Analysis. In *Proc. ICCV Workshops* (2011), IEEE, pp. 1626–1633. 10

[ASK*05] ANGUELOV D., SRINIVASAN P., KOLLER D., THRUN S., RODGERS J., DAVIS J.: SCAPE: Shape Completion and Animation of People. In *ACM Transactions on Graphics (TOG)* (2005), vol. 24, ACM, pp. 408–416. 9

[BCBB16] BIASOTTI S., CERRI A., BRONSTEIN A., BRONSTEIN M.: Recent trends, applications, and perspectives in 3d shape similarity assessment. In *Comp. Graph. Forum* (2016), vol. 35, pp. 87–119. 1

[BDM15] BOISSONNAT J.-D., DEY T. K., MARIA C.: The compressed annotation matrix: An efficient data structure for computing persistent cohomology. *Algorithmica 73*, 3 (2015), 607–619. 4

[BLW12] BAUER U., LANGE C., WARDETZKY M.: Optimal topological simplification of discrete functions on surfaces. *Discrete & Computational Geometry 47*, 2 (2012), 347–377. 2

[BRLB14] BOGO F., ROMERO J., LOPER M., BLACK M. J.: FAUST: Dataset and Evaluation for 3d Mesh Registration. In *Proc. CVPR* (2014), pp. 3794–3801. 9, 10

[Car09] CARLSSON G.: Topology and data. *Bulletin of the AMS 46*, 2 (2009), 255–308. 1, 3

[CCL03] CAZALS F., CHAZAL F., LEWINER T.: Molecular shape analysis based upon the morse-smale complex and the connolly function. In *Proceedings of the nineteenth annual symposium on Computational geometry* (2003), ACM, pp. 351–360. 2

[CCSG*09] CHAZAL F., COHEN-STEINER D., GLISSE M., GUIBAS L. J., OUDOT S. Y.: Proximity of persistence modules and their diagrams. In *Proceedings of the twenty-fifth annual symposium on Computational geometry* (2009), ACM, pp. 237–246. 1, 3, 7

[CK13] CHEN C., KERBER M.: An output-sensitive algorithm for persistent homology. *Computational Geometry 46*, 4 (2013), 435–447. 1

[CSEH07] COHEN-STEINER D., EDELSBRUNNER H., HARER J.: Stability of persistence diagrams. *Discrete & Computational Geometry 37*, 1 (2007), 103–120. 1, 3, 4, 5, 7

[CSEM06] COHEN-STEINER D., EDELSBRUNNER H., MOROZOV D.: Vines and vineyards by updating persistence in linear time. In *Proceedings of the twenty-second annual symposium on Computational geometry* (2006), ACM, pp. 119–126. 2

[Cut13] CUTURI M.: Sinkhorn distances: Lightspeed computation of optimal transport. In *Advances in neural information processing systems* (2013), pp. 2292–2300. 2

[CW17] CANG Z., WEI G.: Topologynet: Topology based deep convolutional and multi-task neural networks for biomolecular property predictions. *PLOS Computational Biology 13*, 7 (2017), e1005690. 2

[CZCG05] Carlsson G., Zomorodian A., Collins A., Guibas L. J.: Persistence barcodes for shapes. *International Journal of Shape Modeling 11*, 02 (2005), 149–187. 1, 2, 3, 5

[DLL*10] Dey T. K., Li K., Luo C., Ranjan P., Safa I., Wang Y.: Persistent heat signature for pose-oblivious matching of incomplete models. In *Computer Graphics Forum* (2010), vol. 29, Wiley Online Library, pp. 1545–1554. 2

[EBC17] Ezuz D., Ben-Chen M.: Deblurring and denoising of maps between shapes. In *Computer Graphics Forum* (2017), vol. 36, Wiley Online Library, pp. 165–174. 2

[EH10] Edelsbrunner H., Harer J.: *Computational topology: an introduction*. American Mathematical Soc., 2010. 2, 3

[EHZ01] Edelsbrunner H., Harer J., Zomorodian A.: Hierarchical morse complexes for piecewise linear 2-manifolds. In *Proceedings of the seventeenth annual symposium on Computational geometry* (2001), ACM, pp. 70–79. 2

[ELZ00] Edelsbrunner H., Letscher D., Zomorodian A.: Topological persistence and simplification. In *Foundations of Computer Science, 2000. Proceedings. 41st Annual Symposium on* (2000), IEEE, pp. 454–463. 2, 4

[GBCB16] Goodfellow I., Bengio Y., Courville A., Bengio Y.: *Deep learning*, vol. 1. MIT press Cambridge, 2016. 4

[GHO16] Gameiro M., Hiraoka Y., Obayashi I.: Continuation of point clouds via persistence diagrams. *Physica D: Nonlinear Phenomena 334* (2016), 118–132. 2

[HO17] Huang R., Ovsjanikov M.: Adjoint map representation for shape analysis and matching. In *Computer Graphics Forum* (2017), vol. 36, Wiley Online Library, pp. 151–163. 2

[KBBV15] Kovnatsky A., Bronstein M. M., Bresson X., Vandergheynst P.: Functional correspondence by matrix completion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2015), pp. 905–914. 2, 6

[KLF11] Kim V. G., Lipman Y., Funkhouser T.: Blended Intrinsic Maps. In *ACM Transactions on Graphics (TOG)* (2011), vol. 30, ACM, p. 79. 10

[LJY16] Liu J.-Y., Jeng S.-K., Yang Y.-H.: Applying topological persistence in convolutional neural network for music audio signals. *arXiv preprint arXiv:1608.07373* (2016). 2

[LOC14] Li C., Ovsjanikov M., Chazal F.: Persistence-based structural recognition. In *Proc. CVPR* (2014), pp. 1995–2002. 1, 4

[LRB*16] Litany O., Rodolà E., Bronstein A. M., Bronstein M. M., Cremers D.: Non-rigid puzzles. In *Computer Graphics Forum* (2016), vol. 35, pp. 135–143. 2, 6

[LRBB17] Litany O., Rodolà E., Bronstein A. M., Bronstein M. M.: Fully spectral partial shape matching. *Computer Graphics Forum 36*, 2 (2017), 247–258. 2

[MCSK*17] Mandad M., Cohen-Steiner D., Kobbelt L., Alliez P., Desbrun M.: Variance-minimizing transport plans for inter-surface mapping. *ACM Trans. on Graph. 36* (2017), 14. 2, 10, 11

[MDSB03] Meyer M., Desbrun M., Schröder P., Barr A. H.: Discrete Differential-Geometry Operators for Triangulated 2-Manifolds. In *Visualization and mathematics III*. Springer, 2003, pp. 35–57. 8

[MMS11] Milosavljević N., Morozov D., Skraba P.: Zigzag persistent homology in matrix multiplication time. In *Proceedings of the twenty-seventh annual symposium on Computational geometry* (2011), ACM, pp. 216–225. 1

[NMR*18] Nogneng D., Melzi S., Rodolà E., Castellani U., Bronstein M., Ovsjanikov M.: Improved functional mappings via product preservation. In *Computer Graphics Forum* (2018), vol. 37. 2

[NO17] Nogneng D., Ovsjanikov M.: Informative descriptor preservation via commutativity for shape matching. *Computer Graphics Forum 36*, 2 (2017), 259–267. 2, 7, 10

[OBCS*12] Ovsjanikov M., Ben-Chen M., Solomon J., Butscher A., Guibas L.: Functional Maps: A Flexible Representation of Maps Between Shapes. *ACM Transactions on Graphics (TOG) 31*, 4 (2012), 30. 1, 2, 6, 7, 9, 10

[OCB*17] Ovsjanikov M., Corman E., Bronstein M., Rodolà E., Ben-Chen M., Guibas L., Chazal F., Bronstein A.: Computing and processing correspondences with functional maps. In *ACM SIGGRAPH 2017 Courses* (2017), pp. 5:1–5:62. 1, 2, 6

[OPT*17] Otter N., Porter M. A., Tillmann U., Grindrod P., Harrington H. A.: A roadmap for the computation of persistent homology. *EPJ Data Science 6*, 1 (2017), 17. 4

[PP93] Pinkall U., Polthier K.: Computing Discrete Minimal Surfaces and their Conjugates. *Experimental mathematics 2*, 1 (1993), 15–36. 8

[RCB*17] Rodolà E., Cosmo L., Bronstein M. M., Torsello A., Cremers D.: Partial functional correspondence. In *Computer Graphics Forum* (2017), vol. 36, pp. 222–236. 2

[ROA*13] Rustamov R. M., Ovsjanikov M., Azencot O., Ben-Chen M., Chazal F., Guibas L.: Map-based exploration of intrinsic shape differences and variability. *ACM Transactions on Graphics (TOG) 32*, 4 (2013), 72. 2

[SDGP*15] Solomon J., De Goes F., Peyré G., Cuturi M., Butscher A., Nguyen A., Du T., Guibas L.: Convolutional wasserstein distances: Efficient optimal transportation on geometric domains. *ACM Transactions on Graphics (TOG) 34*, 4 (2015), 66. 2

[SM93] Singh R. K., Manhas J. S.: *Composition Operators on Function Spaces*, vol. 179. Elsevier, 1993. 13

[SOCG10] Skraba P., Ovsjanikov M., Chazal F., Guibas L.: Persistence-based segmentation of deformable shapes. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on* (2010), IEEE, pp. 45–52. 1, 2, 4

[SOG09] Sun J., Ovsjanikov M., Guibas L.: A Concise and Provably Informative Multi-Scale Signature Based on Heat Diffusion. In *Computer graphics forum* (2009), vol. 28, pp. 1383–1392. 9

[SPKS16] Solomon J., Peyré G., Kim V. G., Sra S.: Entropic metric alignment for correspondence problems. *ACM Transactions on Graphics (TOG) 35*, 4 (2016), 72. 2

[STY17] Skraba P., Thoppe G., Yogeshwaran D.: Randomly weighted $d-$ complexes: Minimal spanning acycles and persistence diagrams. *arXiv preprint arXiv:1701.00239* (2017). 2

[Tan55] Tanaka T.: On the family of connected subsets and the topology of spaces. *Journal of the Mathematical Society of Japan 7*, 4 (1955), 389–393. 13

[TCL*13] Tam G. K., Cheng Z.-Q., Lai Y.-K., Langbein F. C., Liu Y., Marshall D., Martin R. R., Sun X.-F., Rosin P. L.: Registration of 3D point clouds and meshes: a survey from rigid to nonrigid. *IEEE TVCG 19*, 7 (2013), 1199–1217. 1

[VKZHCO11] Van Kaick O., Zhang H., Hamarneh G., Cohen-Or D.: A survey on shape correspondence. In *Computer Graphics Forum* (2011), vol. 30, pp. 1681–1707. 1

[VLB*17] Vestner M., Lähner Z., Boyarski A., Litany O., Slossberg R., Remez T., Rodola E., Bronstein A., Bronstein M., Kimmel R., Cremers D.: Efficient deformable shape correspondence via kernel matching. In *Proc. 3DV* (2017). 2

[VLR*17] Vestner M., Litman R., Rodolà E., Bronstein A., Cremers D.: Product manifold filter: Non-rigid shape correspondence via kernel density estimation in the product space. In *Proc. CVPR* (2017), pp. 6681–6690. 2, 10, 11

[WZWW17] Wu K., Zhao Z., Wang R., Wei G.-W.: Topp-s: Persistent homology based multi-task deep neural networks for simultaneous predictions of partition coefficient and aqueous solubility. *arXiv preprint arXiv:1801.01558* (2017). 2

Appendix A: Proof of Lemma 1

As all critical points in a super-level set filtration occur at vertex values, there must be a unique vertex v corresponding to each homological critical value (i.e. any function value where the topology changes). Since the coordinates of each point in the persistence diagram correspond to a critical value the map $\pi(b_i, d_i) = (v_b, v_d)$ is unique.

Appendix B: Proof of Lemma 2

Let ε denote the minimum separation between vertex function values. First note that by the assumption of distinct function values, there is a total ordering on the vertices. For any function within a $\varepsilon/2$ ball, i.e.

$$|f(x) - f'(x)| < \varepsilon/2 \qquad \forall x \in \mathcal{M}$$

this ordering remains the unchanged. This immediately implies that the map η from each simplex to a vertex is constant in this neighborhood.

We first formally define the δ-neighborhood of a persistence diagram P_f. Recall that by Eq.1, a persistence diagram is a map from a topological space endowed with a real valued function (X, f) to a multi-set of points. Note that the map is completely determined by the space and the function. Therefore, we define a δ-neighborhood of P_f as all P_g of the form (X, g) such that $||f - g||_\infty$.

We next show that ζ is constant in a small enough neighborhood. Although ζ is defined per point in the diagram, this induces a map from a diagram to some collection of simplicies in the space. If the map is constant per point, it is constant in the neighborhood defined above.

In an upper-star filtration, the total ordering on the vertices can also be extended to the remaining simplices. Each vertex v has a set of simplices for which it determines the function value when the simplex enters the filtration. This is given by the preimage $\eta^{-1}(v)$. Generically, the preimages are disjoint sets of simplices, i.e. no simplex maps to two vertices. As η is constant in some neighborhood, this implies that the map ζ can also be chosen so that it is constant in this neighborhood. Each preimage can be extended to a total order independently. As the sets of preimages do not change, this extension can be kept constant. Finally, we note that the composition of two constant maps is again constant completing the proof.

Appendix C: Proof of Lemma 3

Assuming generic P_f and P_g all pairwise distances between points in P_f and P_g are unique. This implies that the maximum of any matching is unique. This proves the first part of the lemma. As the values are distinct, it follows that there exists a neighborhood such that the pair of points achieving the maximum is constant. Finally, the continuity of persistence diagrams implies that any point which is sufficiently close to to the diagonal cannot achieve the bottleneck distance and so cannot change the matching.

Appendix D: Proof of Theorem 1

Given a continuous bijection between two topological spaces, the induced pull-back of real-valued functions must clearly preserve products of functions. Similarly it must preserve persistence diagrams of real-valued functions since both the topological structure of the space and function values are preserved.

Conversely, consider any invertible linear functional map T_F. It is well-known that if T_F preserves pointwise products of functions, then it must correspond to a pull-back by a point-to-point map (e.g. corollary 2.1.14 in [SM93]). Moreover, since T_F is invertible, its inverse must also satisfy this property, meaning that the underlying point-to-point map is a bijection.

Finally, by assumption both T_F and its inverse preserve persistence diagrams of real-valued functions. Now, consider an indicator function of a region. By definition, its persistence diagram will have a unique local maximum, if and only if the region is connected. Thus preservation of persistence diagrams implies that the underlying point-to-point correspondence (and its inverse) maps connected regions to connected regions. Finally, a bijective map between two topological spaces is continuous if and only if both it and its inverse map connected sets to connected sets (see e.g. [Tan55]).