Stable and Multiscale Topological Signatures

Mathieu Carrière, Steve Oudot, Maks Ovsjanikov

Inria Saclay – Geometrica

April 21, 2015
Shape = point cloud in $\mathbb{R}^d$ ($d = 3$)
Signature = mathematical objects used in shape analysis

Can be very different by nature:
- local/global
- intrinsic/extrinsic
- volumetric/defined on the surface
- type of information (geometry, topology...)

Satisfy 3 main properties:
- invariant to a relevant deformation class (rotation, scaling...)
- stability
- informativeness
Most common signatures:
- curvature (mean, gaussian...)
- PCA features
- spin image
- shape context
- shape diameter function
- heat kernel signature
- wave kernel signature
- geodesic features (eccentricities...)

Lack of mathematical framework to define stability

Idea: use persistent homology to build topological point signatures on shapes that are:
- provably stable
- both local and global
Persistence Diagrams

Mapping to Signatures

Shape Matching

Shape Segmentation
Persistence Diagrams (PDs) are the building blocks of the topological signature

Let \( x \in \) shape \( S \). We introduce a metric \( d_g \) on \( S \) and we compute the persistent homology of the sub-level set filtration of the distance function \( f_x(y) = d_g(x, y) \)

\( S = \) sampling from compact connected smooth manifold of dimension 2 without boundary

- no homology of dimension \( \geq 3 \)
- trivial homology of dimension 0 and 2
- only interesting dimension is 1

Let \( PD(f_x) = PD_1(F_x) \) where \( F_x = \{ f_x^{-1}([0, \alpha]) \}_{\alpha \in \mathbb{R}_+} \)

In practice, \( F_x \) has a finite index set
- $d_g$ is computed with Dijkstra’s algorithm
- Edges come from:
  - a triangulation of the shape
  - a neighborhood graph if no triangulation is given
Problem: 1D persistence is costly to compute → use symmetry

Theorem: [Cohen-Steiner, Edelsbrunner, Harer, 2009] For a real-valued function $f$ on a $d$-manifold, the ordinary dimension $r$ persistent classes of $f$ correspond to the ordinary dimension $d - r - 1$ persistent classes of $-f$

We focus on the ordinary dimension 0 persistent classes of $-f_x$

Essential dimension 1 persistent classes are lost

PDs are much easier to compute (Union-Find data structure)
Stability?

Definition: Let \((X, d_X)\) and \((Y, d_Y)\) be two metric spaces. A \textit{correspondence} between them is a subset \(C\) of \(X \times Y\) such that:

- \(\forall x \in X, \exists y \in Y \text{ s.t. } (x, y) \in C\)
- \(\forall y \in Y, \exists x \in X \text{ s.t. } (x, y) \in C\)

Definition: The \textit{metric distortion} \(\epsilon_m\) of a correspondence is:

\[
\epsilon_m = \sup_{(x,y) \in C, (x',y') \in C} |d_X(x, x') - d_Y(y, y')|
\]

Definition: Let \(f : X \to \mathbb{R}\) and \(g : Y \to \mathbb{R}\). The \textit{functional distortion} \(\epsilon_f\) of a correspondence is:

\[
\epsilon_f = \sup_{(x,y) \in C} |f(x) - g(y)|
\]
Theorem: Let $S_1$ and $S_2$ be two compact Riemannian manifolds. Let $f : S_1 \to \mathbb{R}$ and $g : S_2 \to \mathbb{R}$ be two $c$-Lipschitz functions. Let $x \in S_1$, $y \in S_2$ and a correspondence $C$ such that $(x, y) \in C$. Then, for sufficiently small $\epsilon_m$:

$$d^\infty_b(\text{PD}(f), \text{PD}(g)) \leq 19c\epsilon_m + \epsilon_f$$

Theorem: Let $S_1$ and $S_2$ be two compact Riemannian manifolds. Let $x \in S_1$, $y \in S_2$ and a correspondence $C$ such that $(x, y) \in C$. Then, for sufficiently small $\epsilon_m$:

$$d^\infty_b(\text{PD}(f_x), \text{PD}(f_y)) \leq 20\epsilon_m$$

Nearly-isometric shapes have very similar PDs for corresponding points
- $d_b^\infty$ is costly to compute in practice
- It is hard to define simple quantities like means or variance in the space of PDs
- Send the PDs to $\mathbb{R}^d$! How?
- Need to be oblivious to the points order:
  - look at the distance distribution
  - add extra distance-to-diagonal terms
  - sort the final values for stability
  - add null values to the topological signatures so they have the same dimension
\[ \forall (p, q) \in \text{PD}, \text{ we compute:} \]

\[ m(p, q) = \min(\|p - q\|_\infty, d_\Delta(p), d_\Delta(q)) \]
Stability is preserved. Let $x \in S_1$ and $y \in S_2$. If $X$ and $Y$ are the topological signatures computed from $\text{PD}(f_x)$ and $\text{PD}(f_y)$:

$$C(N)\|X - Y\|_2 \leq \|X - Y\|_\infty \leq d_b^\infty(\text{PD}(f_x), \text{PD}(f_y))$$

- $N$ is the dimension (can be 50...)

- $C(N) = \sqrt{\frac{2}{N(N-1)}}$ (can be quite small...)

- Most kernels methods in ML needs $\| \cdot \|_2$...

- Stability is preserved *whatever the number of components kept!*

- Invariant to scaling via log-scale
Visualization of the signature stability:
MDS on the signatures with $\| \cdot \|_{\infty}$:
- kNN segmentation with the signatures and $\| \cdot \|_{\infty}$:
Application: **functional maps**

Let $S_1$ and $S_2$ be two shapes. Functional maps are linear applications $L^2(S_1) \rightarrow L^2(S_2)$

Finite case: functions are vectors, linear maps are matrices

Possibility to derive a correspondence from the functional map: shape matching
Assume:

- $S_1$ and $S_2$ have the same number of points $n$
- you have $m$ functions defined on them stored in matrices $G_1$ and $G_2$ of sizes $n \times m$
- you have a diagonal $m \times m$ matrix $D$ weighting the functions

Then solve the following problem:

$$\tilde{C} = \arg\min_C \| (CG_1 - G_2)D \|_F$$

In practice, $D$ is computed over a set of training shapes.

We used the topological signature in addition to the other classical signatures.
Weights:
Effects on the correspondence quality:
Improvements on the shapes:
Application: **shape segmentation and labeling**

A `segmentation` of a shape with `n` vertices is a `n`th dimensional vector `c` giving a label to every point.

**Goal:** find the segmentation of a `test` shape given the ones of several `training` shapes.

**Supervised algorithms:**
- map every vertex `v` of label `l` from a shape to its `signature` vector `x \in \mathbb{R}^d`.
- consider the pairs `(x, l)` in the training set as realizations of pairs of random variables `(X, L)`.
Assume the test shape has \( n \) vertices. The core of supervised algorithms:

- model (with training set):
  \[
  f(c) = P(L_1 = c_1 \ldots L_n = c_n \mid X_1 = x_1 \ldots X_n = x_n)
  \]

- derive \( c^* = \arg\max_c f(c) \)

- evaluate result through comparison to ground-truth segmentation \( c^{gt} \) (often given manually) with specific comparison functions \( d \):
  \[
  \epsilon = d(c^*, c^{gt})
  \]

- Recognition rate : \( d(c^*, c^{gt}) = \frac{1}{n} \sum_{i=1}^{n} 1_{c_i^* = c_i^{gt}} \)

- Rand Index :
  \[
  d(c^*, c^{gt}) = \binom{n}{2}^{-1} \sum_{i<j}(C_{ij}P_{ij} + (1 - C_{ij})(1 - P_{ij}))
  \]

where \( C_{ij} = 1 \) iif \( c_i^* = c_j^* \) and \( P_{ij} = 1 \) iif \( c_i^{gt} = c_j^{gt} \)
Attention, there are dependencies between the labels, conditionally to the test shape!

\[
f(c) \neq \prod_{i=1}^{n} P(L_i = c_i \mid X_1 = x_1 \ldots X_n = x_n)
\]

Indeed, if all the neighbors of \( v \) have same label \( l \), it is very unlikely for \( v \)'s label to be \( \neq l \)

Instead: conditional Markov property:

\[
P(L_i = c_i \mid L_j = c_j, j \neq i, X) = P(L_i = c_i \mid L_j = c_j, j \in N_i, X)
\]

where \( N_i \) is the 1-ring neighborhood of vertex \( i \) in the mesh.
Modeling the joint conditional probability distribution $f$ with the conditional Markov property is the purpose of *probabilistic graphical models*

**Proposition:** [Hammersley, Clifford, 1971] The family of possible joint probabilities $\mathcal{F}$ is:

$$\mathcal{F} = \left\{ \frac{1}{Z} \exp \left( \sum_{i=1}^{n} f_i(c_i, x_i) + \sum_{e_{ij} \in E} g_{ij}(c_i, c_j, x_i, x_j) \right) \right\}$$

- There is no requirements for the functions $f_i$ and $g_{ij}$
- $Z$ is the *normalization factor*
GraphCut algorithm (Boykov et al., 2001) is mostly used to find \( \text{argmax } f \) when \( f \in \mathcal{F} \).

Several other algorithms exist: belief propagation, sum-product, max-product... very common in probabilistic graphical models.

Very often, \( f_i \) is a probability and \( g_{ij} \) is a *compatibility term*.

In our case, \( f_i \) is the output of a classifier (like SVM) trained on the training set.

We computed the \( f_i \)s and \( c^* \) both with and without the topological signatures.
<table>
<thead>
<tr>
<th>Object</th>
<th>SB5</th>
<th>SB5+PDs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Human</td>
<td>21.3</td>
<td>11.3</td>
</tr>
<tr>
<td>Cup</td>
<td>10.6</td>
<td>10.1</td>
</tr>
<tr>
<td>Glasses</td>
<td>21.8</td>
<td>25.0</td>
</tr>
<tr>
<td>Airplane</td>
<td>18.7</td>
<td>9.3</td>
</tr>
<tr>
<td>Ant</td>
<td>9.7</td>
<td>1.5</td>
</tr>
<tr>
<td>Chair</td>
<td>15.1</td>
<td>7.3</td>
</tr>
<tr>
<td>Octopus</td>
<td>5.5</td>
<td>3.4</td>
</tr>
<tr>
<td>Table</td>
<td>7.4</td>
<td>2.5</td>
</tr>
<tr>
<td>Teddy</td>
<td>6.0</td>
<td>3.5</td>
</tr>
<tr>
<td>Hand</td>
<td>21.1</td>
<td>12.0</td>
</tr>
<tr>
<td>Plier</td>
<td>12.3</td>
<td>9.2</td>
</tr>
<tr>
<td>Fish</td>
<td>20.9</td>
<td>7.7</td>
</tr>
<tr>
<td>Bird</td>
<td>24.8</td>
<td>13.5</td>
</tr>
<tr>
<td>Armadillo</td>
<td>18.4</td>
<td>8.3</td>
</tr>
<tr>
<td>Bust</td>
<td>35.4</td>
<td>22.0</td>
</tr>
<tr>
<td>Mech</td>
<td>22.7</td>
<td>17.0</td>
</tr>
<tr>
<td>Bearing</td>
<td>25.0</td>
<td>11.2</td>
</tr>
</tbody>
</table>
Some examples:
We introduced a provably stable topological multiscale signature for points in shapes that gives complementary information to the other classical signatures.

Directions for future work:

- Other distance functions (diffusion)?
- Other shape analysis tasks (classification, retrieval)?
- Other objects (images, point clouds of high dimension)?
Thank you! Questions?