CS233: The Shape of Data Handout # 2
Geometric and Topological Data Analysis
Stanford University Wednesday, 25 April 2018

Homework #2: Computational Topology - Simplicial Complexes, Homology, Persistence, Bottleneck Distances and the Mapper Algorithm [100 points total]
Due Date: Wednesday, 9 May 2018

Problem 1. [10 points]

A simplicial complex is a combinatorial structure used to encode topological information on point-cloud data, typically based on proximity information. A simplicial complex can also be used to encode the topology of various spatial partitions.

Consider the decomposition of San Francisco into 11 districts as shown below. Create an abstract simplicial complex to represent the 11 regions show1 and their topological adjacencies. Each region will become a vertex (0-simplex) of the complex, pairs of regions that share a common boundary give rise to edges (1-simplices), and triplets of regions that have a common point become triangles (2-simplices of the complex).

Draw a schematic diagram of your simplicial complex embedded in the plane. Orient all simplices according to the map layout: edges go from west to east and all triple region intersections go around counterclockwise.

Write down the boundary matrices corresponding to your complex: ∂_1 is a matrix with rows the 0-simplices and columns the 1-simplices; ∂_2 is a matrix with rows the 1-simplices and columns the 2-simplices. Verify that $\partial_1 \partial_2 = 0$, as claimed.

Extend the numbering of the regions (e.g., complex vertices) given in the image to a full filtration of the complex by assigning to each simplex the largest value of its vertices and ordering simplices of the same value by dimension. Write down the sequence of simplices in the way they would be added one-by-one in the filtration.

Problem 2. [10 points]

Prove that the homology groups H_k for the p-simplex are trivial in dimensions $1 \leq k \leq p$. In other words, show that for any k, $1 \leq k < p$, *any* cycle of k-simplices in the p simplex is in fact a boundary of some chain of $k+1$-simplices (so $Z_k = B_k$). For simplicity you can assume that homology is taken over $\mathbb{Z}_2$ (so each simplex is present with a coefficient of 0 or 1 only).

Problem 3. [20 points]

Extraction of topological information from point cloud data is the main focus of this homework; we will explore this in two scenarios:

1. Points sampled from a shape in Euclidean space.
2. The above, with a film of background noise added.

In the problem we will use the computational topology software *Javaplex*. The home page for Javaplex is:

http://javaplex.github.io/

Begin by working through sections 4.1 to 5.2 of the provided Javaplex tutorial by Henry Adams. The tutorial wiki is also available in the link below:

https://github.com/appliedtopology/javaplex/wiki/Tutorial

This will explain how to obtain a barcode for a set of points in Euclidean space, based on a construction of their Rips complex filtration. Use this knowledge to work through the following exercises.

• Generate a sampling of a unit circle in the plane (100 sample points) and verify that the Rips barcode has a single long interval in the 1-dimensional homology.

• Generate a pair of circles (with some Gaussian noise). Experiment with overlapping and non-overlapping circles. Do you recover the expected topology?

• Generate a noisy circle lying in a square film of background noise. (Say $100+100$ points.) How can you filter out the background noise and focus on the topological signal?

Pick an integer $1 \leq k \ll n$. The $k-$codensity $\text{cod}_k(x)$ of a point x is the distance from x to its k^{th} nearest neighbor. If $\text{cod}_k(x)$ is small, then x is a crowded region of the data. This is implemented for you in `codensity.m`, which takes as input the $n \times n$ matrix D which contains pairwise distances between all points.

• How successfully can you use this codensity to extract the topology of the circle?

We now extend the idea of measuring topology on a circle to a sphere of a general dimension k. The goal is to calculate the homology of the $k-$sphere in all dimensions.

• Generate N points uniformly sampled on the surface of a $k-$sphere in $\mathbf{R}^{k+1}$ (for example, generate points in a spherically symmetric normal distribution and the normalize the corresponding vectors to unit length). To what extent can you robustly recover the Betti profile $[1,0,...,0,1,0,....]$? How large can we make k so that reproducible results can no longer be obtained?

• Apply a background film of uniform Gaussian noise, with mean -2 and standard deviation 4 as follows:

```
Y = -2 + 4 * rand(k+1, M);
Z = [X, Y];
```

Here, X is the set of points sampled on the k-sphere in the previous problem. Using a co-density estimator to filter out the noise, can you still recover the correct Betti sequence for the sphere?

Problem 4. [30 points]

Shape features used to perform shape analysis tasks can be extrinsic or intrinsic. Extrinsic features are features that depend on the embedding of the shape in 3D, while intrinsic features are dependent only measurements made on the surface of the object. To see the the distinction, consider articulations of humans or mammals. These are near-isometric deformations in the sense that they preserve geodesic distances on the shape surface, which are intrinsic features. By contrast, extrinsic features, such as Euclidean distances or shape poses can drastically change under such deformations.

Using geodesic distances, we can view shapes as just metric spaces. The Gromov-Hausdorff distance is a metric that measures how far two compact metric spaces are from being isometric. For near-isometric shapes we expect it to be very close to zero. We first define the Hausdorff distance, and then use this notion to define the Gromov-Hausdorff distance.

Hausdorff distance: Let X and Y be two non-empty subsets of a metric space (M,d). We define their Hausdorff distance $d_H(X,Y)$ to be

$$d_H^M(X,Y) = \max\{\sup_{x\in X}\inf_{y\in Y} d(x,y), \sup_{y\in Y}\inf_{x\in X} d(x,y)\} \tag{1}$$

Gromov-Hausdorff distance: The Gromov-Hausdorff distance between compact metric spaces (X,d_X) and (Y,d_Y) is:

$$d_{GH}((X,d_X),(Y,d_Y)) = \inf_{Z,\gamma_X,\gamma_Y} d_H^Z(\gamma_X(X),\gamma_Y(Y)), \tag{2}$$

where γ_X,γ_Y range over all isometric embeddings of X,Y into some shared metric space (Z,d_Z).

An alternate definition of the Gromov-Hausdorff distance is given as follows:
For sets A and B, a subset $R \subset A\times B$ is a *correspondence* (between A and B) if and only if
(1) $\forall a \in A$, *there exists* $b \in B$, *s.t.* $(a,b) \in R$
(2) $\forall b \in B$, *there exists* $a \in A$, *s.t* $(a,b) \in R$

Let $\mathscr{R}(A,B)$ denote the set of all possible correspondences between sets A and B.

Consider metric spaces (X,d_X) and (Y,d_Y). Let $\Gamma : X \times Y \times X \times Y \to \mathbf{R}^+$ be given by $\Gamma(x,y,x',y') = |d_X(x,x') - d_Y(y,y')|$. Then, the Gromov-Hausdorff distance between X and Y can be rewritten as

$$d_{GH}(X,Y) := \frac{1}{2}\inf_{R\in\mathscr{R}(A,B)}\max_{(x,y)\in R,(x',y')\in R}\Gamma(x,y,x',y')\,. \tag{3}$$

The above definition provides a view of the Gromov-Hausdorff distance as obtaining the specific correspondence that minimizes the maximum discrepancy of distance differences between corresponding points in the two shapes. To read more about the Gromov-Hausdorff distance metric and for a proof showing the equivalence of the two definitions, you can check the book by D. Burago, Y. Burago and S. Ivanov at the link below:

http://www.math.psu.edu/petrunin/papers/akp-papers/bbi.pdf

In the following problem, we use a Euclidean space as (Z,d_Z). Even in this setting, computing Gromov-Hausdorff distances can be computationally demanding. We begin with a couple of toy examples to become familiar with these distances.

• Compute the Hausdorff distances between two circles in the Euclidean plane of radii r_1, r_2 and centered at $(0,0)$ and $(d,0)$ respectively. Compute an upper bound to the Gromov-Hausdorff distance of the two circles by computing the Hausdorff distance between them after a translation that makes both these circles concentric. What do you observe as $r_1 \to r_2$?

You have learned about Rips complexes of finite metric spaces in class, as well as about persistence diagrams and their bottleneck distances. An interesting fact is that the bottleneck distance between the persistence diagrams of the Rips filtrations of two finite metric spaces can be proven to be a lower bound on their Gromov-Hausdorff distance. Thus the bottleneck distance can become a "poor man's" substitute for the expensive GH distance. This is shown in the paper "Gromov Hausdorff Stable Signatures for Shapes using Persistence" by Chazal, Cohen-Steiner, Guibas, Memoli and Oudot. The link to this paper is provided below.

http://geometry.stanford.edu/papers/ccgmo-ghsssp-09/ccgmo-ghsssp-09.pdf

• Compute (by hand) the bottleneck distance between 0-dimensional persistence diagrams of the Rips complex on a k-sided regular polygon inscribed in a circle of radius r_1, and the Rips complex on a k-sided regular polygon inscribed within a circle of radius r_2. How does this bottleneck distance vary in the limit as $k \to \infty$? Is the lower bound assertion satisfied?

• Consider X, a set of two points at distance 2, and Y, a set of two points at distance $2+2\epsilon$ in the plane. What is the Gromov-Hausdorff distance between the two sets? Plot the $0-$dimensional persistence diagrams of the two point sets, and compute the bottleneck distance

between them. Is the lower bound to the Gromov-Hausdorff distance tight?

You can use the Javaplex software to obtain the persistence intervals and to compute the bottleneck distances. Shape data in the form of point clouds will be provided to you in `shapes.mat`. `shapes.mat` consists of both the sampled point collections of 6 shapes, and the inter-nodal Dijkstra distances for the sampled points in the shapes with full resolution. You can use this data for the following problems. (**Note:** You can perform all the computations without explicitly using the point clouds, and using just the Dijkstra matrix if you use the Explicit Metric Space feature in Javaplex)

• Consider the point clouds in `shapes.mat`. These point clouds are representative of two different sets of isometric shapes. Use the persistence algorithm to compute the lower bound on the Gromov-Hausdorff distances between these shape spaces. Depending on the values of the bottleneck distances between these shape spaces, classify these point clouds into two different classes of isometric shapes.

Problem 5. [30 points]

The Mapper algorithm is a tool that can be used to perform topological analysis of geometric data such as point clouds. The algorithm filters the data using a user-provided "lens" function and generates a graph skeleton structure that helps visualize the data, based on both point proximity and the function values. The algorithm will be discussed in class on Tuesday, 26 April, and you will be provided with a skeleton of the code implementing the Mapper algorithm. In this problem you will first complete an implementation of the Mapper algorithm, and then apply the algorithm to a set of shapes to perform a shape analysis task.

Mapper implementation: The input you are provided with is a shape mesh S, with set of vertices V. The lens function you use to compute the Mapper algorithm can be viewed as a function $f : V \to [a,b] \subset \mathbf{R}$. There are a couple of parameters which vary the Mapper graph; these are the number of intervals n_{ints} and the interval size g. You can vary the variables `ints` and `gap` to modify the values of n_{ints} and g respectively in the MATLAB implementation. The defaults are set to `ints = 10` and `gap = 2/ints - 1e-5`. The start points of the intervals are $a, a+h, a+2h, ..., a+(n-1)h$, where $h = \frac{b-a}{n_{ints}}$, and the end points are $a+g, a+h+g, ..., a+(n-1)h+g$ respectively. If $a+ih+g > b$, then the end point is just g. The default interval size considered is such that the length of intersection between consecutive intervals is almost half of the interval size — you can vary this as long the intersection between consecutive intervals is non-empty. You could also modify the code to have intervals of unequal sizes.

The output of your algorithm is the labeled mapper graph, with nodes labeled from 1 to n_{ints}. Each node of the graph is a connected component on the mesh, which is obtained as follows. Assuming the Mapper graph G has been built using the first $i-1$ intervals for a positive

integer i, **compute** $S_i = f^{-1}([a+ih, a+ih+g])$, where f^{-1} is the pre-image of the Mapper function f. **Compute** the connected components of S_i, given by $\Omega_{i,1}, \Omega_{i,2}, ..., \Omega_{i,n_i}$. For this you can use the function `function_components` provided to you. Now, you will add n_i new vertices to your graph, all labeled i, each corresponding to a certain $\Omega_{i,j}$, for $j = 1, 2, .., n_i$. **Add an edge** between the node corresponding to $\Omega_{i-1,k}$ ($k = 1, 2, ..., n_{i-1}$) and node corresponding to $\Omega_{i,j}$, if $\Omega_{i-1,k} \cap \Omega_{i,j}$ is non-empty. To do this, simply modify the graph adjacency matrix `An` where specified. The graph plot is implemented.

Remark: Feel free to use any feature function over the shape as the lens function to produce the mapper output. One potential feature function that can be used is the height of every point in the mesh.

You are provided with 12 human shape meshes. In all cases the z-coordinate corresponds to the normal upright direction.

To-do:

• Complete the parts in **bold** letters in the description of the Mapper implementation, in the MATLAB skeleton provided to you.

• Plot the graph for the shape `victoria0`, with the z-coordinate as the lens function. Notice that the label here is 1 for shape vertices having the lowest value and n_{ints} for shape vertices having the highest value of the lens function.

• In which of these shapes do the Mapper graphs contain a cycle? Why?

• Do you observe a pattern in the graphs unique to the shapes with a standing pose? Why? Shapes `victoria2` and `victoria24`, are both shapes in the reclining pose. Plot their Mapper graphs with the z coordinate as the feature function. Why do they have a different number of components with the 1 label?

• **Bonus question:** Is it possible to infer the pose of the shapes from the Mapper graphs alone? Select one or more feature functions that you believe can help in this task. Play around with all parameters to perform this task — interval sizes, interval overlaps, feature functions etc.