javaPlex – persistent homology in Java

Andrew Tausz   Mikael Vejdemo-Johansson

June 1, 2011

Persistent homology

In a large variety of areas we encounter the following fundamental problem:

Problem

Estimate topological features of a shape based on a family of sample points (*point clouds*).

Applications

- The space of natural 3×3 pixel patches is a Klein bottle.
- This Klein bottle yields new techniques for analyzing textures.
- Shape matching and shape databases.
- Filtering materials databases for high CO_2 adsorbitivity.
- Verifying sensor coverage for plane regions.

Persistent homology

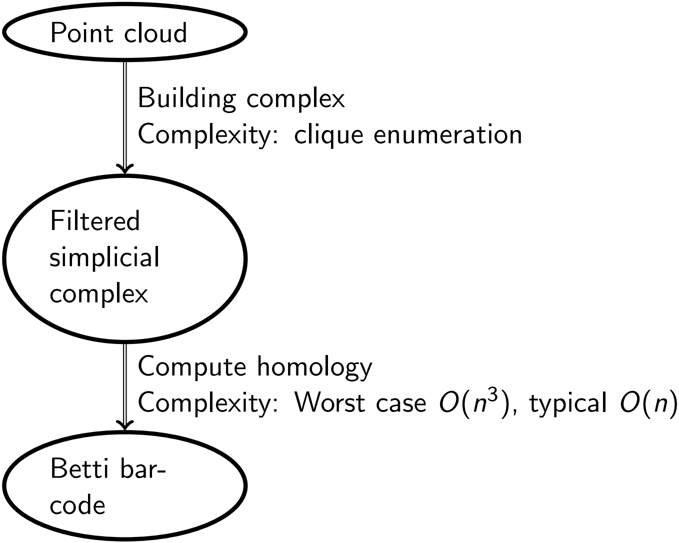
Our favourite solution is due to Edelsbrunner, Letscher and Zomorodian.

Solution

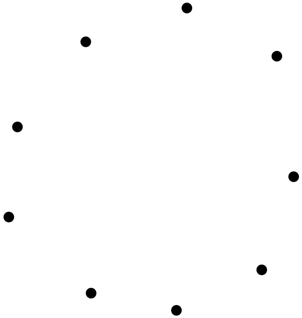
Čech complexes, Vietoris-Rips complexes, α-shapes and several other constructions capture by a filtered simplicial complex stemming from intersection relations of disks centered on the sample points

Persistent homology tracks the changes in topological features, and picks out such features that remain present over a long time.

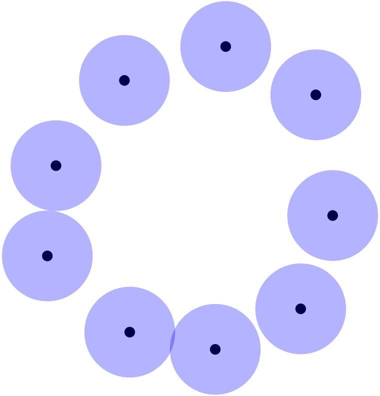
Persistent homology



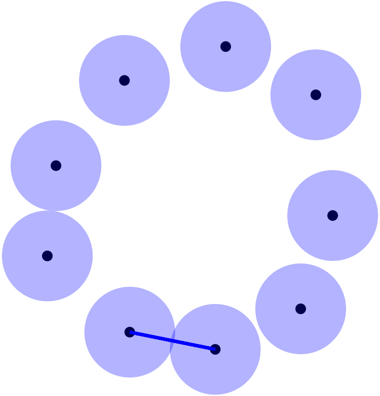
Vietoris-Rips complex



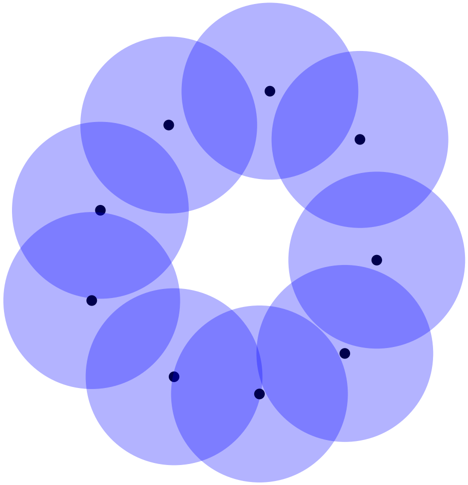
Vietoris-Rips complex



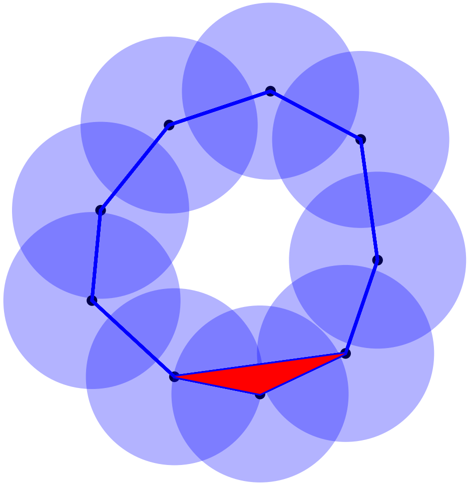
Vietoris-Rips complex



Vietoris-Rips complex



Vietoris-Rips complex



History of Plex

Early developments in the theory of persistent homology took place at Stanford, as did the development of software to compute with point clouds.

1. Plex
2. jPlex

History of Plex

Early developments in the theory of persistent homology took place at Stanford, as did the development of software to compute with point clouds.

1. Plex – C++, Matlab through MEX, memory hog
2. jPlex – Java, Matlab natively, BeanShell, highly optimized

Design properties for javaPlex

javaPlex grew out of dissatisfaction with both Plex and jPlex.

Disadvantages

Plex

- Dependent on exact Matlab version and mex dialect
- Design carrying *very* many pointers: high memory consumption

jPlex

- Too highly optimized: no real capacity to extend or modify
- Not actually competitive enough to motivate this

Usage examples

This is joint work with David Eklund, Jonathan Hauenstein, Martina Scolamiero, and Chris Peterson.

Application

Consider as a linkage the configurations of cyclo-octane (C_8H_{16}). The carbon bonds in this have fixed length, and an angle of $\arccos(-1/3) \simeq 109.47$. By fixing one carbon atom at the origin, and it's closest neighbours at $(1, 0, 0)$ and $(-\frac{1}{3}, \frac{2}{3}\sqrt{2}, 0)$, we may study the distribution of the remaining atoms. The variety of possible configurations is a surface in $\mathbb{R}^{15}$.

This variety was first described by *Martin, Thompson, Coutsias, and Watson* in TOPOLOGY OF CYCLO-OCTANE ENERGY LANDSCAPE. J. Chem. Phys. 132, 234115 (2010).

Cyclo-octane

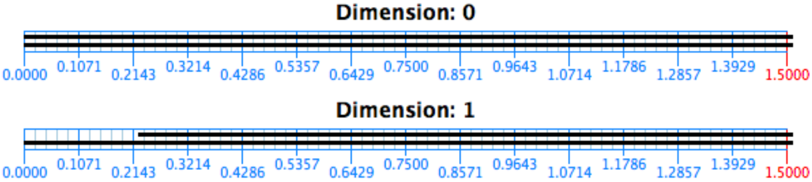
We sample, using BERTINI, points on the real singular locus of the cyclo-octane variety.
Using MATLAB and JAVAPLEX, we create a witness complex on 100 out of 1 606 sampled points:

Cyclo-octane

```
>> load Distances.out;
>> m_space = metric.impl.ExplicitMetricSpace(Distances);
>> diam = metric.utility.MetricUtility.estimateDiameter(m_space);
>> r_max = diam/2
ans = 3.1017
>> landmarks = api.Plex4.createMaxMinSelector(m_space,100);
>> stream = api.Plex4.createWitnessStream(landmarks,3,r_max/2);
>> stream.getSize()
ans = 11418
>> persistence = api.Plex4.getDefaultSimplicialAlgorithm(3);
>> fii = persistence.computeIntervals(stream);
>> fvi = stream.transform(fii);
>> api.Plex4.createBarcodePlot(fvi,'witness',r_max/2);
```

Cyclo-octane

From the Matlab code above, we acquire the Betti barcodes:

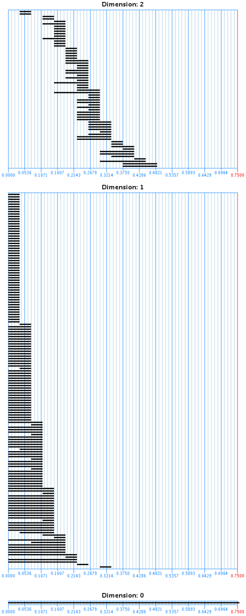


Which is strongly consistent with the structure of the singular set from *Martin, Thompson, Coutsias, and Watson* as a pair of disjoint circles.

The entire computation runs in 6 seconds. (mean running time for 10 runs, std.dev. 0.8s)

Cyclo-octane

- Computation with coefficients in $\mathbb{F}_2$ and $\mathbb{F}_3$ to look for torsion.
- Witness complex with 500 landmarks.
- Total complex 26M simplices.
- Computation time: 3 days.
- Severe memory issues.
- Surprising lack of difference between $\mathbb{F}_2$ and $\mathbb{F}_3$: torsion killed by particular intersection of sphere and Klein bottle.



Access to the package

Software package

Published on `http://code.google.com/p/javaplex`

JavaDoc documentation

Linked from software homepage.

Usage tutorial

Linked from software homepage. Thanks to Henry Adams for adapting the jPlex tutorial to javaPlex.

Questions or comments?