Introduction to Topological Data Analysis

Persistent Homology

Norm Matloff
University of California, Davis

Presentation to Levenson Lab
Dept. of Pathology
UC Davis School of Medicine
October 23, 2019

Mathematical Theory Not Presented

Norm Matloff
University of California, Davis

Mathematical Theory Not Presented

- We'll skip the abstract, math grad student-level theory. :-)
- Much of it of limited relevance to the simple method I'll use here as my running example.

Broad Overview

- Determine "what is connected to what" in dataset.
- Definition of *connected* depends on the application and possibly the ingenuity of the analyst.
- Do this in each of a sequence of steps.
- Each step produces some kind of data summarizing connectivity in that step. The data is collectively called a *filtration*.
- Use that output data as features, e.g. to do classification.

Image Classification Example

Image Classification Example

- TDA can be applied to many kinds of data, but we'll focus mainly on image classification here.
- The famous MNIST data, hand-drawn digits. Predict what digit it is, by analyzing the pixels (28×28).
- Not just greyscale, but mainly black-and-white. Here I'll look only at pixels > 192 level.

Filtration Method in This Example

Filtration Method in This Example

- For simplicity, I'll first use a somewhat nonstandard TDA method, which I'll call the *Sweep method*.
- May or may not be better than other methods.
- But is simple, easy to explain and draw.
- **Just an example**.

Crucial Need for Dimension Reduction

Crucial Need for Dimension Reduction

- In MNIST case, we are predicting digit from $28^2 = 784$ features.
- 784 way too large: (a) Overfitting. (b) Horrendous computation needs.
- So, we need to convert the existing 784 features to a smaller number (*dimension reduction*). But how?

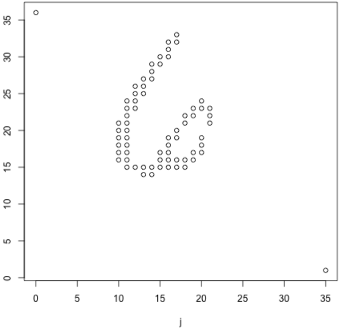
Dimension Reduction Methods for Images

Dimension Reduction Methods for Images

- Principal Components Analysis (PCA)
    - A traditional approach. Project the data from R^{784} to, say, R^{50}, using eigenanalysis.
    - Tacitly linear, but can form polynomial terms before compute PCA.
- Convolutional Neural Networks (CNNs)
    - Currently most fashionable.
    - Not new! The "C" part of CNN is just **traditional image smoothing**: break image into tiles, and then finding the median or max pixel intensity in each tile. In MNIST, take 4×4 tiles, stride 4, so now have $7^2 = 49$ predictors.
- Geometric methods:
    - Runs statistics (RLRN): counts of how many consecutive vertical, horizontal or diagonal pixels are black, etc.
    - TDA.

A '6'

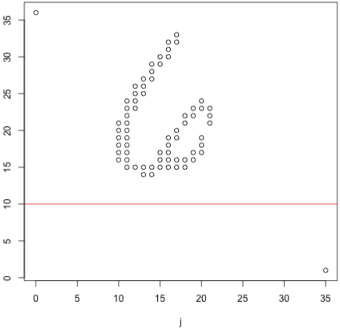
A '6'



Filtration plan:

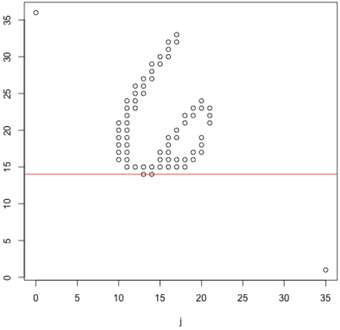
- Draw a series of horizontal lines.
- See how many components are formed in the figure by a line.

A '6'



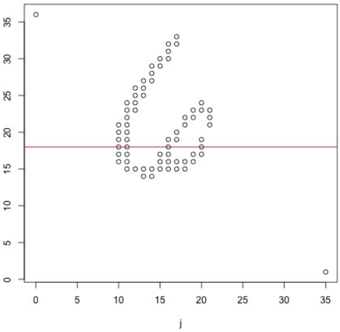
E.g. line at row 10. 0 components

A '6'



Line at row 14: 1 component.

A '6'

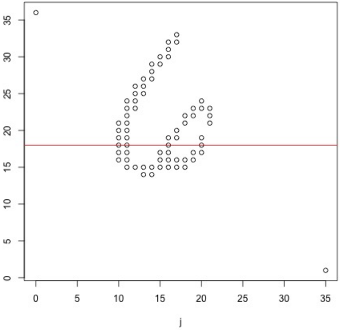


Line at row 18: 3 components

Norm Matloff
University of California, Davis

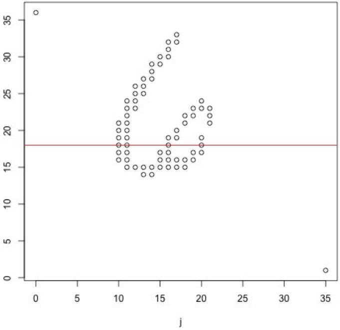
Birth, Death Times

Birth, Death Times



We talk about *birth* and *death* times. E.g. the first 3-component line is "born" at row 17 and "dies" at row 25.

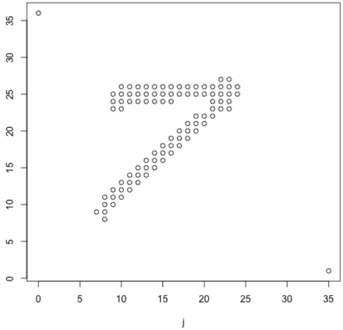
Birth, Death Times



We talk about *birth* and *death* times. E.g. the first 3-component line is "born" at row 17 and "dies" at row 25. Those pairs, e.g. (17,25), form our dimension-reduced data.

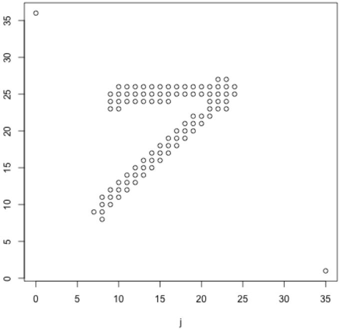
A '7'

A '7'



A 1-component line will be born at row 8, then die at row 23.

A '7'



A 1-component line will be born at row 8, then die at row 23. Then we get a 2-component birth, not long-lived, rows 23-24 only.

'6' vs. '7'

'6' vs. '7'

digit	typical pattern
'6'	3-comps., then 1-comps.
'7'	1-comps., then 2-comps.

'6' vs. '7'

digit	typical pattern
'6'	3-comps., then 1-comps.
'7'	1-comps., then 2-comps.

- So, easy to distinguish '6' and '7' via birth-death (BD) data, right?

'6' vs. '7'

digit	typical pattern
'6'	3-comps., then 1-comps.
'7'	1-comps., then 2-comps.

- So, easy to distinguish '6' and '7' via birth-death (BD) data, right?
- But what if the top bar of a '7' is angled slightly up, not down?

'6' vs. '7'

digit	typical pattern
'6'	3-comps., then 1-comps.
'7'	1-comps., then 2-comps.

- So, easy to distinguish '6' and '7' via birth-death (BD) data, right?
- But what if the top bar of a '7' is angled slightly up, not down? Then only have 1-comps.

A Second Opinion

A Second Opinion

Solution: "Get a second opinion": Collect vertical-bar BD data.

digit	typical pattern
'6'	mainly 3-comps.
'7'	mainly 2-comps.

A Second Opinion

Solution: "Get a second opinion": Collect vertical-bar BD data.

digit	typical pattern
'6'	mainly 3-comps.
'7'	mainly 2-comps.

So, our new features could be the two sets of BD data, horizontal and vertical sweeps.

BD Data as Features

BD Data as Features

- All dimension-reduction methods replace the original data with the dimension-reduced data.

BD Data as Features

- All dimension-reduction methods replace the original data with the dimension-reduced data.
- The latter are then used as features in your favorite ML method, SVM, NNs, RFs, logistic, whatever.

BD Data as Features

- All dimension-reduction methods replace the original data with the dimension-reduced data.
- The latter are then used as features in your favorite ML method, SVM, NNs, RFs, logistic, whatever.
- E.g. MNIST. Orig. data matrix is 65000×784. If do the above 4×4 tiling, new data matrix is 65000×49.

BD Data as Features

- All dimension-reduction methods replace the original data with the dimension-reduced data.
- The latter are then used as features in your favorite ML method, SVM, NNs, RFs, logistic, whatever.
- E.g. MNIST. Orig. data matrix is 65000×784. If do the above 4×4 tiling, new data matrix is 65000×49.
- In essence, the "C" part of CNN feeds the latter matrix is into the "NN" part, i.e. the dense NN layers.

BD Data as Features

- All dimension-reduction methods replace the original data with the dimension-reduced data.
- The latter are then used as features in your favorite ML method, SVM, NNs, RFs, logistic, whatever.
- E.g. MNIST. Orig. data matrix is 65000×784. If do the above 4×4 tiling, new data matrix is 65000×49.
- In essence, the "C" part of CNN feeds the latter matrix is into the "NN" part, i.e. the dense NN layers.
- But that 65000×49 matrix could instead be fed into SVM, RF, logit or whatever. See e.g. Zhou (2017) and Miller (2017).

BD Data as Features

- All dimension-reduction methods replace the original data with the dimension-reduced data.
- The latter are then used as features in your favorite ML method, SVM, NNs, RFs, logistic, whatever.
- E.g. MNIST. Orig. data matrix is 65000×784. If do the above 4×4 tiling, new data matrix is 65000×49.
- In essence, the "C" part of CNN feeds the latter matrix is into the "NN" part, i.e. the dense NN layers.
- But that 65000×49 matrix could instead be fed into SVM, RF, logit or whatever. See e.g. Zhou (2017) and Miller (2017).
- In the CNN case, the tiles have weights, which vary over the iterations. But these weights have direct analogs in the other ML methods, e.g. coefficients in logit.

Notes

Notes

- For color images, calculate BD data for R, G, B.
- Various *persistence diagrams* to plot BD data.
- Have lots of non-image applications too. E.g. Kurlin *et al*, (2018) use moisture data at various altitudes to predict atmospheric rivers.

Not Out of the Woods Yet

Not Out of the Woods Yet

Not so simple. For instance:

Not Out of the Woods Yet

Not so simple. For instance:

- **Anomalous BDs:** Sometimes have fainter pixels than our 192 threshold. E.g. row 20 in the '6' had a gap. Causes an incorrect birth/death.

Not Out of the Woods Yet

Not so simple. For instance:

- **Anomalous BDs:** Sometimes have fainter pixels than our 192 threshold. E.g. row 20 in the '6' had a gap. Causes an incorrect birth/death.
- **Vectorization:** Different images for the same digit have different numbers of BD pairs. But ML methods require the feature vector to have a constant number of features from one data point to another (in this case one image to another), the 49 in our 65000×49 example above.

Not Out of the Woods Yet

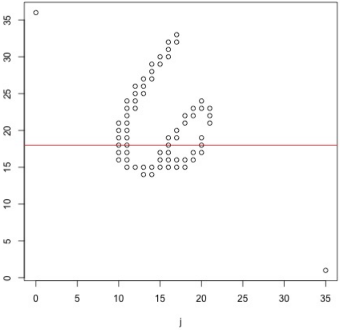
Not so simple. For instance:

- **Anomalous BDs:** Sometimes have fainter pixels than our 192 threshold. E.g. row 20 in the '6' had a gap. Causes an incorrect birth/death.
- **Vectorization:** Different images for the same digit have different numbers of BD pairs. But ML methods require the feature vector to have a constant number of features from one data point to another (in this case one image to another), the 49 in our 65000×49 example above.
- **Orientation:**The above filtration scheme largely assumed:
  - Mainly black-and-white image, not really greyscale.
  - Image has a notion of left-right, up-down.

  Not true for, e.g., histology slides.

Possible Solutions: Anomalous BDs

Possible Solutions: Anomalous BDs



- Ignore row 20 in the BD calculation (e.g. D - B = 1).
- But what if the row is real? Are there 1-row comps. in most images of this digit?
- Maybe do BD at each of several pixel intensity thresholds, e.g. 64, 128, 192.

Possible Solutions: Vectorization

Possible Solutions: Vectorization

- **Grid counts:**
  - Say have 35-row, 35-column images, so that (B,D) pairs are in the range 1 to 35.
  - Construct a "2-D histogram" (counts only, not graphed), with bin size, say, 5×5.
  - So we have a grid, $1 \leq i \leq j \leq 7$), a total of 28 points.
  - For each image, calculate the count of (B,D) pairs at each grid point. Do the same for the red vertical lines.
  - Our new data matrix is 65000×56. Feed that into SVM, NN, RF, logit, whatever.

Possible Solutions: Vectorization

- **Grid counts:**
  - Say have 35-row, 35-column images, so that (B,D) pairs are in the range 1 to 35.
  - Construct a "2-D histogram" (counts only, not graphed), with bin size, say, 5×5.
  - So we have a grid, $1 \leq i \leq j \leq 7$), a total of 28 points.
  - For each image, calculate the count of (B,D) pairs at each grid point. Do the same for the red vertical lines.
  - Our new data matrix is 65000×56. Feed that into SVM, NN, RF, logit, whatever.
- **Ad hoc:**
  - For a large, detailed images, the above method may need voluminous computation and/or lead to overfitting.
  - Some analysts devise their own *ad hoc* method.
  - E.g. Garside (2019) compute a vector consisting of the number of pixels, average lifetime, area under the persistence function, and four measures based on polygons drawn in the persistence graph.

Possible Solutions: Orientation

Possible Solutions: Orientation

Most filtration methods for images don't assume the image has a left and right, up and down. (More on this shortly.)

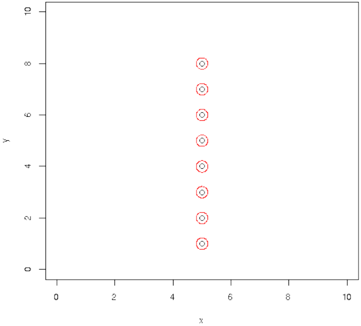
(Vietoris-)Rips Filtrations

(Vietoris-)Rips Filtrations

- We've seen one filtration method so far (my Sweep method). Here's another, very widely used.
- Draw a red circle around each data point, same radius for all.
- The filtration consists of drawing an increasing sequence of radii.
- Points in overlapping circles are considered to be in the same component.
- Use with any metric, not just Euclidean distance and not just on image data.

An 'I'

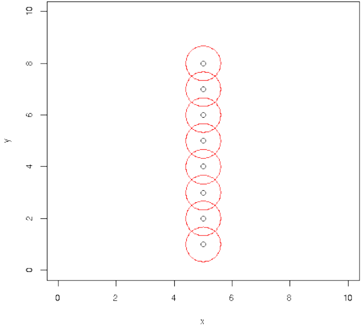
An 'I'



- radius 0.2
- 8 1-components

An 'I'

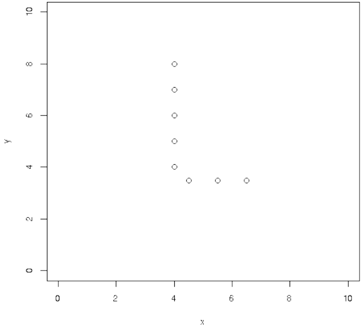
An 'I'



- radius 0.6
- 1 8-component
- the 8 1-components died at 0.5, the 1-component was born at 0.5

An 'L'

An 'L'

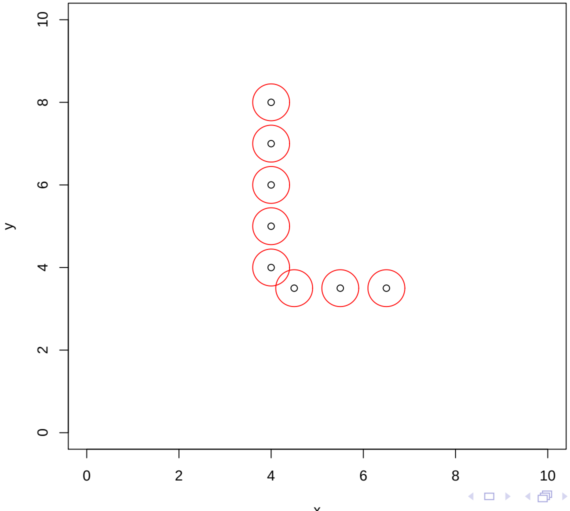


- I took the 'I' and just bent it; linear distance between points still 1.0
- but now there will be a birth at $0.5(0.5\sqrt{2}) = 0.35$, not 0.5
- originally 8 1-components, then 7 components (1 2-comp., 6 1-comps.), then 1 8-comp. — different pattern than for 'I'

An 'L'

An 'L'

Radius 0.4:



Rips Senses Angles!

Rips Senses Angles!

The point:

> *Rips filtration does more than topology; it does geometry. (Math: curvature)*

An Approach to General Images Using Rips

An Approach to General Images Using Rips

E.g. do this:

- Do a separate analysis at each of several (or many) threshold values.
- In each one, do a Rips filtration.
- Combine (i.e. concatenate) the BD data for the various thresholds to get the feature vector for an image.

A "Topographic" Method

A "Topographic" Method

- Consdider, e.g., the image classification context.
- Analogous to a topographic map. At each altitude, have contours at that height.
- Treat the (row,col) data as being the X-Y plane, and the pixel intensities as the Z-axis.
- Instead of a moving red line, now have a red plane, moving in the Z direction but always parallel to the X-Y plane.
- Pixels that survive the red-plane cutoff op and that are adjacent then considered in the same component.
- Computer BD data as before.

Nongeometric Applications

Nongeometric Applications

- Association of diet with cancer.
- Data on age, gender, race, various dietary measures.
- Still can do, say, Rips, using distance in the same way as k-Nearest Neighbors.

Betti Numbers

Betti Numbers

- Most of the math theory behind persistent homology centers around this concept.
- β_i is data on "holes" at dimension $i = 0, 1, 2, ...$
- Dimension 0 is number of components (with the holes being gaps between components in a line).
- Dimension 1 is holes in a plane, e.g. the 2 holes in an '8'.
- Dimension 2 is holes in 3-dimensional objects. In the image classification context, this would be the volume of spheres in voxel data.
- Higher dimensions not visualizable, but e.g. can use Rips on data (not images, just regular data) of d columns, with the holes being considered dimension $d - 1$.
- Software typically gives the user a choice of maximum dimension to use. Too high a value may result in overfitting.

Betti Numbers (cont'd.)

Betti Numbers (cont'd.)

- TDA software will typically report BD data separately for each dimension.
- E.g. for a 2-dim. image, the dimension 0 BD will be reported, followed by dimension 1. But one will probably use all of it.
- Where does the Sweep method fit into this? Since have both horizontal and vertical sweeps, it is inherently 2-D (Betti dimension 1), so Betti-type separation wouldn't make sense.

R Software

R Software

- **TDA**
- **TDAstats**
- **TDAsweep** (mine, in preparation)