1 Computing Persistence Homology

There are multiple libraries available for computing persistent homology and conducting TDA in general, such as DIPHA, Dionysus, GUDHI, and JavaPlex. Most of these are C++ libraries with python bindings. I have tried several of these. On small datasets, their performances are not very different, and their syntaxes, or their structures, are similar, too. [1] lists detailed benchmarking results on these libraries for larger realistic datasets.

JavaPlex[2] is a library written in Java developed by Stanford computational topology group. It also offers examples and tutorials in Matlab. Since I'm most familiar with Matlab and Java, I decided to first get familiar with this library, to develop a general sense on computing persistent homology. However, according to [1], JavaPlex is one of the slowest libraries in the benchmark tests. My past experience with scientific computing also shows that C++ is probably the best choice for performing a large amount of computation. Therefore, a C++ library such as Risper (the fastest according to [1]) will likely be used for the project.

I followed the tutorial with JavaPlex available on `https://github.com/appliedtopology/javaplex/wiki/Tutorial`.

1.1 Some General Issues

After experiments with some toy examples such as torus ($S^1 \times S^1$ embedded in $\mathbb{R}^4$), $n-$circles, the figure 8, $n-$cube, and other simple geometric shapes, I found the following problems with computing PH.

Noise Contrary to what I have assumed, PH is not that resistant or stable to noise. In many examples, it is hard to set the threshold for "short" meaningless intervals in the barcode, especially when the desired features or the true features have different scales, or resolutions. However, the exact influence of noise and how to deal with that largely depends on the data set.

One interesting thing to notice is that the sparsity of data points sometimes lessen the effect of noise. For example, in the figure 8 example, 75 points with noise produce unwanted intervals in the barcode, whereas 50 points with noise do not. This could mean that we do not necessarily need to sample a very large amount of data points to get accurate results.

Counterintuitive Results In the cube example, point clouds $(\pm 1, \pm 1, \pm 1)$ with Vietoris-Rips simplex produce a barcode of 8 0-dimensional generators, 5 1-dimensional generators, 0 2-dimensional generator, and 1 3-dimensional generator. However conterintuitive it is, this is indeed the correct result. For 1-dimensional "holes", among the 6 "faces" of the cube, one is actually a linear combination of the rest five. As for the 3-dimensional generator, when the filtration value is large enough, $t \geq \sqrt{2}a$, all the diagonal vertices will be connected by an edge, which makes the simplicial complex a cross-polytope, which is homeomorphic to a 3-sphere.

However, if we adequately sample more points from the cube, "correct" results will be produced. In particular, even if we do not include the 8 vertices in the point cloud, or we add noise to the points, as long as we sample enough points, we can still get satisfying results.

The practical implication of this issue could be that

1. in 3D space, if we sample insufficient points, some small local structure could be misrepresented
2. some structure might be a linear combination of others, which could result in miscounting the total number
3. this summary is only intended to remind myself of these possible sources of "error", what exactly happens to realistic data might be totally different or complicated.

Parameter Choice

- maximum dimension: often just the dimension of the underlying space. Interesting results could be produced if we actually include dimensions higher than the underlying space. In terms of real higher dimensional data, usually only interested in dimensions less than 3.
- maximum filtration value: for example, the maximal "distance" to draw an edge between points in VR complex. Usually we have some empirical ways to determine this value, e.g. from domain knowledge. Another way to determine this value is embedded in landmark selection.
- division number: since we cannot continuously increase the filtration value, we have to choose a number of divisions, or correspondingly a step size, for the increase. This seems to be a less important factor in computing PH, both because it does not significantly affect the produced barcode, and because increase in this value does not require additional computation resource.
- landmark number: intuitively speaking, the more landmarks the better. However, in the real data examples with lazy witness stream, only 50 landmarks are used in a data set of more than 10000 points, and the result is still very satisfying.

On the one hand, for all the parameters above, we want them as small as possible to reduce computational cost. On the other hand, increase in these values do not necessarily improve the performance of the algorithm or help our understanding of the data.

Computational Cost A very small change could double the time of computation. For example, increase from 1.9 to 2.1 for the maximum filtration value could result in 10 times computational time required. A major reason for this is that the number of simplices produced is, in the worst case of VR filtration, exponential with respect to the maximum filtration value. VR filtration, perhaps the simplest filtration, is entirely based on distance, and includes each simplex as long as the simplex's faces have all been included. Other types of filtrations such as witness and lazy witness described below have been proposed to add extra criteria for what kind of simplices should be included in the complex.

1.2 Landmark selection

Real point cloud data is usually too large to be entirely included in computation. Therefore, we need to select.

In JavaPlex, the problem is addressed by constructing "witness" stream (or filtration, stream seems to be a term preferred by JavaPlex) and selecting landmarks. Two common choices are randomly selected landmarks, and sequentially maxmin landmarks. Maxmin tends to cover the dataset and be spread apart, and is therefore susceptible to outliers. The tutorial claims such problem can be solved by selecting a dense subset.

The issue of how to select "representative" points from data seems only relevant to point cloud data. In the case of pixel/voxel imaging data, grey scale value instead of distance is usually used as filtration value, and therefore perhaps we can just sample at equidistant pixels/voxels. In the case of density distribution data, perhaps just sample by the distribution itself. In addition, for the point cloud data, perhaps some interpolation could also be used to sample equidistantly?

1.3 Witness Stream & Lazy Witness Stream

The witness stream (W) and lazy witness stream (W_ν) are only implemented by JavaPlex and GUDHI, according to [1], and JavaPlex does not provide other common choices such as Čech filtration and alpha filtration. W and W_ν are based on landmark selection.

In addition, they are proposed to deal with curse of dimensionality and reduce the number of simplexes produced. Essentially, a simplex exists in W only when a point "witnesses" it. Details omitted. Lazy witness stream is even "simpler". "Lazy" refers to the fact that it's a "flag" complex: the 1-skeleton determines all higher dimensional simplices, and therefore less computation is involved. The parameter ν, in some sense, controls how hard to find a witness, and therefore how many simplexes are identified.

Performance Experiments with the toy models described above show that, compared to VR filtration, they produce considerably less simplices, and thus require less computation resources. Moreover, the resulting barcodes contain less noise interval bars, and are clearer to understand. However, this is only the toy model case. Real data are typically more complicated and noisy, and therefore further investigation is required to thoroughly determine the practical differences between the classic VR and the somewhat novel witness streams.

1.4 Real Data Experiments

The tutorial includes 3 real data experiments, the range image patches, the optical image patches, and the cyclo-octane molecule conformations. The tutorial uses lazy witness stream on all of them. In fact, witness, and lazy witness, perform similarly well on these 3 data sets. Vietoris-Rips filtration is simply too heavy to be performed on such data set, and if we attempt to use the same landmarks as with witness stream, without the extra simplex selection criterion based on all the points, VR filtration produces nonsense. However, the three data sets are more or less not so "real", esp. the cyclo-octane molecule conformations. They are all very low dimensional data specifically intended to produce topological implications, and therefore we should not be surprised at the perfect barcodes the algorithm produces.

1.5 Keep track of the features

Regarding out potential need for accurately identify and locate certain structures either in cell nuclei density data, or cryoEM images, one important problem with JavaPlex and its algorithms is that it's not good at keeping track of how a feature appears and how it dies. For example, at certain filtration value, two half circles merge to one circle, and JavaPlex could not provide such information as which two half circles give rise to which circle. However, JavaPlex could produce a "representative circle" for each generator. Thus, I assume it's not particularly difficult to modify the code to keep track of more detailed information. Nonetheless, among the numerous births and deaths of features, among the countless merges, which ones are important to keep track of, and which ones are simply transient changes caused by noise, seem to remain an issue to address. In the real data examples provided by the tutorial, they are only interested in the general topological implication provided by PH, instead of local details.

2 Notes on Cell/Nuclei Data

I also took some notes concerning the cell data, so as to get a general idea on how such data are treated and how people develop the ideas. It's largely based on the review paper [3]. This paper describes state-of-the-art traditional methods for cell nuclei detection and segmentation. After some search on the internet, I feel like 2D and 3D data are not very different, since many methods are originally developed in 2D and can be adapted to 3D. Also, it seems like all the problems concerning nucleus/cells center at detection of nucleus/cells.

Basically, all the methods are guided by some empirical understanding of the geometric properties of the nucleus/cells, from the relevant domain knowledge. For example, since most cells are elliptical (or in fact quite close to circular), we can use Gaussian distribution to fit the shape. A lot of methods are based on the gray scale intensity difference between the cells and surrounding fluids.

Some current challenges include:

1. Touching/Overlapping cells: they are difficult to deal with since it requires multiple types of information to distinguish touching/overlapping cells, such as shape, intensity, cell type and etc. Many extant algorithms depend on specific datasets, and therefore are not generalizable. Segmentation with shape preserving is very important, and also very difficult at the same time.
2. Scalability to Large Number: currently, analysis is usually performed on individual patches of a whole slide image, due to constraint of computational cost. Although it's typically unnecessary to achieve cell segmentation on the whole slide images, parameter choosing etc. are more accurate if whole image considered.

Below are detailed notes.

2.1 Detection

2.1.1 Distance Transform

Mechanism Local maxima = centroids of nuclei or cells. Often paired with "Watershed Segmentation".

Disadvantage Only effective on regular shapes in a binary image. Susceptible to small changes. Complex image → variations → over-detection.

Improvement Gaussian filter, then trace gradient vector field. Accumulated pixels threshold to distinguish b/w local and non-local maxima.

Further Lin et al. gradient weighted-distance transform for 3D fluorescence image.

2.1.2 Morphology Operation

Mechanism Binary morphological filtering for images w/ certain structure element, circle, square, cross... Examining the geometrical and topological structures of objects w/ predefined shape. Four basic shift-invariant operators:

1. Erosion
2. Dilation
3. Opening
4. Closing

The four can be used to generate more basic morphological operations, boundary, hole, skeletonizing... Binary morphology can be extended to gray-scale morphology. Widely used operators: top-hat, bottom-hat transforms. For example, UE (Ultimate Erosion). Erosion until can't.

Advantage Can be used to basic image enhancement, preparing for further analysis. UE: can separate touching or overlapping cells.

Disadvantage UE: can produce multiple marker for each cell.

Improvement

1. Improved UE, Park et al. noise robust stopping criterion. Perform until convex. However, binarization.
2. Conditional Erosion: Yang et al. Coarse erosion preserves shapes, and fine erosion avoids under-segmentation

Further Hodneland 3D fluorescence images. Adaptive threshold for ridge extraction, then link gaps. Plissiti gray-scale, not converting.

2.1.3 H-minima/maxima Transform

Mechanism Based on morphology operation, used in local minima detection. Image A, depth value h, $H(A, h) = R^{\epsilon}(A + h)$, where R^{ϵ} is reconstruction by erosion. Some regional minima are suppressed. Initially connected parts can be split in terms of the detected minima, h leads to under/over-segmentation. Usually used to generate markers for watershed transform based segmentation.

Advantage Compared with DT (EDT), all minima → H-minima. Very popular in biomedical images.

Disadvantage Suppress minima, so needs enhancement beforehand. Properly defined h value is needed.

Improvement

1. Adaptive HIT., iteratively increase h until a region merging. Ignores nucleus size.
2. Jung and Kim, adaptively choose h to minimize segmentation distortion.
3. Variance in cell areas.

2.1.4 LoG, Laplacian of Gaussian

Mechanism In medical image analysis, LoG is one of the most popular for small blobs.

Disadvantage

1. Might fail in touching / overlapping objects.
2. Scale issue.

Improvement

1. Lindeberg introduces normalizing factor for multiscale LoG blob detector.
2. Kong generalized LoG, for elliptical structures (oblique elliptical Gaussian)
3. Hessian analysis to identify optimal scale
4. Unsupervised GMM can be used to refine blobs

2.1.5 Maximally Stable Extremal Regions

Mechanism Maximally Stable Extremal Regions. Set of nested extremal regions based on level sets in the intensity landscape. Local intensity minimum-based criterion.

1. Generate sufficient number of extremal regions.
2. Recognize those regions corresponding to real nuclei or cells.
   (a) Eccentricity
   (b) Blob appearance + shape properties
   (c) Arteta formulates an optimization problem, candidates -¿ scores -¿ DP for maximal total score
   (d) Multilevel thresholding

2.1.6 Hough Transformation

Mechanism Circular/elliptical nuclei in pathological images. From xy-plane transform to circular a, b, r parameter space. Discrete voting strategy? Most votes corresponding to parameter? Locate the targets by seeking peaks in parameter space (e.g. gradient descent).

Disadvantage False peaks due to noise, incorrect edge extraction, touching objects. Further analysis is needed.

Improvement Gaussian smoothing to denoise, morphology operations to reconstruct. SVM classifier. Optimization problem can be solved by some ILP.

Further

1. Can deal with arbitrary shapes.
2. 3D transformation can be done.
3. Randomized version

2.1.7 Radial Symmetry Based Voting

Mechanism Locate the centroids of nuclei or cells. High radial symmetry points highlighted.

Disadvantage High computational complexity. False peaks due to clustered nuclei. Radius range. What if not circular?

Improvement FRST. Candidates, thresholding. Affine transform to deal with non-circular.

2.1.8 DNN, esp. CNN

Mechanism

Ciregan Mitotic cell detection in breast cancer histology images.

1. Probability map of being centroid of a mitotic cell.
2. Smoothed w/ disk kernel
3. Non-maxima suppression

Alternatively, can be formulated into an optimization problem.

1. Candidates by LoG, MSER, iterative voting, etc
2. Score by CNN
3. Best subset of candidates

Disadvantage Computationally expensive for large-scale images.

2.2 Segmentation

Methodologies:

1. Separate fore and back grounds, and then splits

2. Markers, then expand
3. Generate candidates, then select

Algorithms:

1. Thresholding
2. Morphology Operation
3. Watershed transform
4. Deformable models
5. Clustering
6. Graph-based models
7. Supervised learning

2.2.1 Intensity Thresholding

First and simplest method.

Mechanism Assumption: intensity distributions for fore- and back- grounds are sufficiently and consistently distinct. Convert to binary with global threshold, or locally adaptive threshold. Usually empirical. Can also be some optimization problem. Inter-variance for example.

Disadvantage How to choose threshold

Improvement Dividing into sub-images. However, introduce other need-to-defined parameters.

Further Convert RGB to gray-scale, Callau

2.2.2 Morphology Operation

Mechanism Top down erosion and bottom up dilation. Erosion until markers are obtained. Grows the markers w/ dilation to reconstruct, while preventing merging.

Disadvantage Under-segmentation in dense cell clumps.

Improvement Modeling w/ shapes

Further Always used to facilitate subsequent segmentation.

2.2.3 Watershed Transformation

Mechanism most popular region accumulation method. Seed points, then add pixels. Flood water in the regional minima, while preventing merging building dams. Highest point. Boundaries -¿ watershed lines, slitting the landscape into regions.

Advantage Gradient magnitude images, also gray intensity images, distance transform maps, and other gray scale images.

Disadvantage Over segmentation.

Improvement Merge false segmentations based on real nuclei or cells.

2.2.4 Deformable Models

Mechanism One of the most popular. Pre-specified region, active contour evolves to boundary by minimizing energy functional, achieves segmentation when reaches boundary. $EG(v) = E_{int}(v) + E_{image}(v) + E_{con}(v)$. Internal smoothness, image energy encouraging towards feature, constraint for interaction with user. Two common types of deformable models are geodesic models and parametric models.

Advantage Great tradeoff b/w efficiency and flexibility.

2.2.5 Clustering

Mechanism Different levels of similarity, internal, and outside. Often followed by edge extraction. Many metrics to choose, Euclidean distance, correlation, 0-1 error, etc.

1. K-means: iterative descent $argmin \sum_{i=1}^{N} \sum_{k=1}^{K} r_{ik} \|x_i - \mu_k\|^2$
2. Fuzzy C-means: Not hard as K-means, one object to plural clusters, membership degree.
3. Expectation-Maximization: also soft. Mixture of Gaussians: $N(x_i|\mu_k, \Sigma_k)$ Maximize log likelihood over the Gaussian parameters and weights.

2.2.6 Graph-Based Methods

Model one image as a weighted graph, each node is a pixel, or subpixel, edge weights = similarity b/w pixels. By certain criterion, partitioned into multiple sets, representing segmentation.

1. Max-flow/Min-cut: Graph-cut algorithm minimizes an energy function. $EG(L) = \sum_{v \in V} D_v(L_v) + \sum_{(v,u) \in N} S_{v,u}(L_v, L_u)$, N penalty plus interaction potential that controls the spatial smoothness.

   Favoring partitioning out small sets of nodes, which are undesired.

2. Normalized cut: Avoid unnatural bias. Somehow normalizes the cut.

3. Conditional Random Field: Variant of Markov random field, set of random variables represented by a graph.

4. Random Walk: graph edge weight represents the likelihood that a random walker will cross the edge.

References

[1] N. Otter, M. A. Porter, U. Tillmann, P. Grindrod, and H. A. Harrington, "A roadmap for the computation of persistent homology," *EPJ Data Science*, vol. 6, p. 17, Aug 2017.

[2] A. Tausz, M. Vejdemo-Johansson, and H. Adams, "JavaPlex: A research software package for persistent (co)homology," in *Proceedings of ICMS 2014* (H. Hong and C. Yap, eds.), Lecture Notes in Computer Science 8592, pp. 129–136, 2014. Software available at `http://appliedtopology.github.io/javaplex/`.

[3] F. Xing and L. Yang, "Robust nucleus/cell detection and segmentation in digital pathology and microscopy images: A comprehensive review," *IEEE Reviews in Biomedical Engineering*, vol. 9, pp. 234–263, 2016.