CS 6170 Computational Topology: Topological Data Analysis **University of Utah**
Spring 2017 **School of Computing**

Lecture 11: Feb 14, 2017

Lecturer: Prof. Bei Wang <beiwang@sci.utah.edu> *Scribe: Kevin Childers*

11.1 Examples of persistent homology

Let K denote a simplicial complex. Two things we want to do with K are

1. compute the homology of K, and
2. compute the **persistent homology** of K.

For persistent homology, we need a filtration

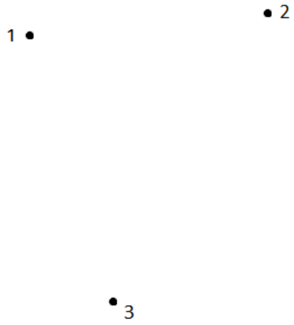
$$\emptyset = K_0 \subseteq K_1 \subseteq \cdots \subseteq K_n = K$$

of our complex K.

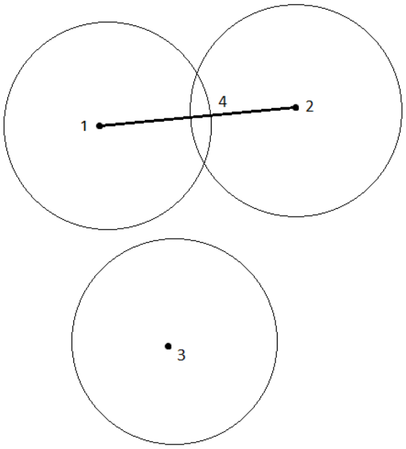
11.1.1 Example: Sensor networks

Lets suppose we have a sensor network with three sensors, which we label 1, 2, and 3. We begin sensing with radius $r = 0$, and slowly increase r. At any point, we can form a Čech complex for the given r.

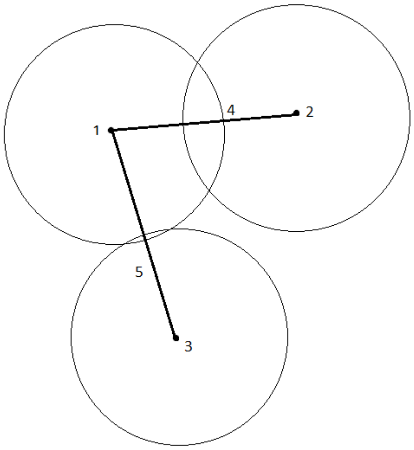
- When $r = 0$, the complex consists of the three vertices, and no higher simplexes.



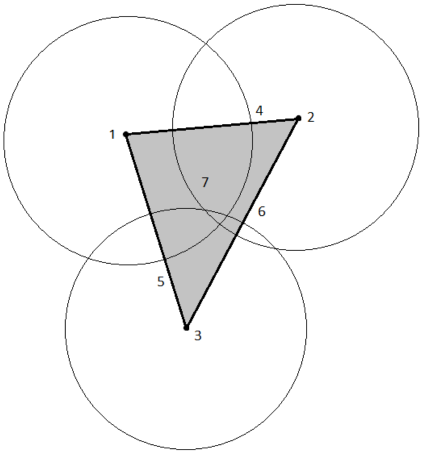
- Say that at $r = 1$, we get our first overlap, between sensors 1 and 2. We now introduce a 1-simplex, which we call 4, which connects 1 and 2.



- Say that at $r = 2.5$, we get another overlap, between sensors 1 and 3. Then we introduce another 1-simplex, which we call 5, which connects 1 and 3.



- Suppose that when $r = 3$ we pick up a triple overlap. In this case we need to introduce two new simplexes: a new 1-simplex between 2 and 3, which we call 6, and a 2-simplex bounded by 4, 5, and 6, which we call 7. Let $K = \{1, 2, 3, 4, 5, 6, 7\}$.



We can use this process to order our simplexes, by order of their appearance. Since 1, 2, and 3 all appear simultaneously, we arbitrarily order them. To avoid confusion, we order them 1, 2, 3 (i.e., 1 appears first, then 2, then 3). Similarly 6 and 7 appear simultaneously, but in this case we must order 6 ahead of 7, because 6 is a boundary of 7. This illustrates the general principle that the boundary on a given simplex should appear before the simplex itself.

The ordering we have given to the simplexes in K now gives rise to a filtration

$$\emptyset = K_0 \subseteq K_1 \subseteq K_2 \subseteq \cdots \subseteq K_7 = K$$

where $K_i = \{1, 2, \ldots, i\}$, the subset of the first i simplexes. The following table gives the Betti numbers at each step in the filtration, along with summary of changes.

Subcomplex	β_0	β_1	Summary
K_0	0	0	Empty complex
K_1	1	0	New component created
K_2	2	0	New component created
K_3	3	0	New component created
K_4	2	0	Component of 2 dies
K_5	1	0	Component of 3 dies
K_6	1	1	Loop created
K_7	1	0	Loop dies

Remark. When components are merged, we keep the older component and say that the younger component "dies."

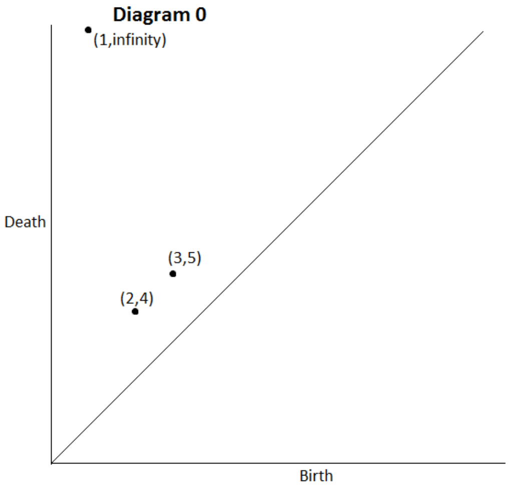
Let t_i denote the "time" of K_i. A component was created at t_1, which never died. It persists on the interval $[t_1, \infty)$. Components were also created at t_2 and t_3, but each died at t_4 and t_5 respectively. These components persisted on the intervals $[t_2, t_4]$ and $[t_3, t_5]$ respectively. A loop was created at t_6, and died at t_7, so this loop persisted on the interval $[t_6, t_7]$.

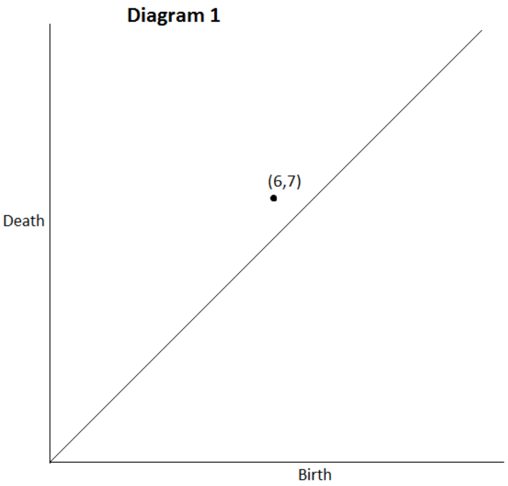
We can record this information in a **persistence diagram**, by plotting birth time against death time. We include a diagonal line to symbolize birth time = death time, and define the **persistence** of an element of homology to be

$$\text{persistence} = \text{death time} - \text{birth time},$$

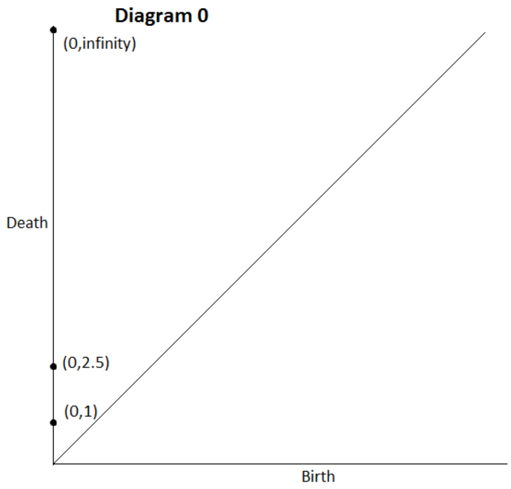
i.e. the vertical distance from the corresponding point to the diagonal.

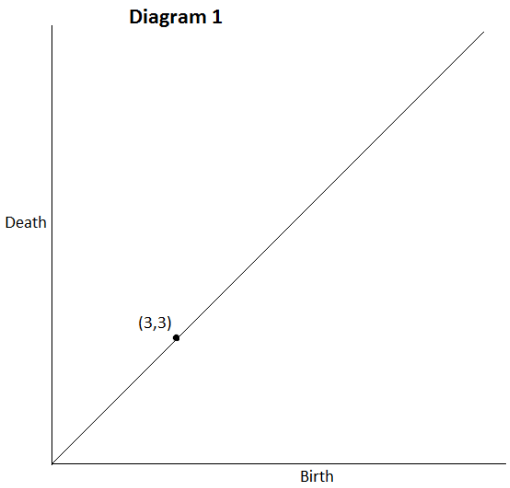
If we say that $t_i = i$, then the persistence diagrams for our example are





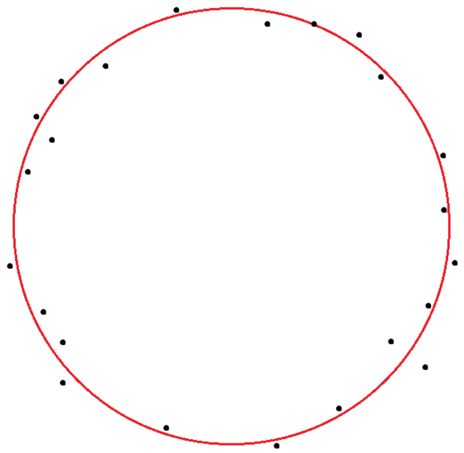
We could instead use radius in place of time. In this case we have components on intervals $[0, \infty)$, $[0, 1]$, and $[0, 2.5]$, and we have a loop on the interval $[3, 3]$. Note that the loop is created and killed at the same instance, which appears graphically by this point lying on the diagonal.



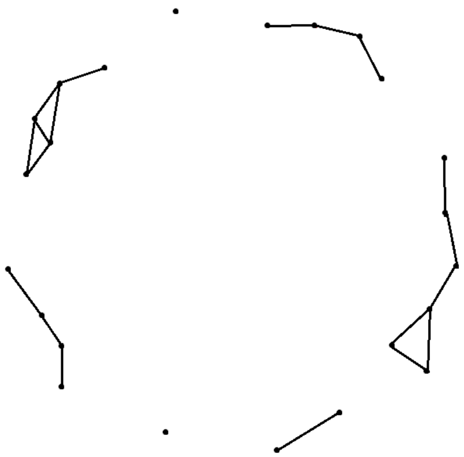


11.1.2 Meaning of Persistence

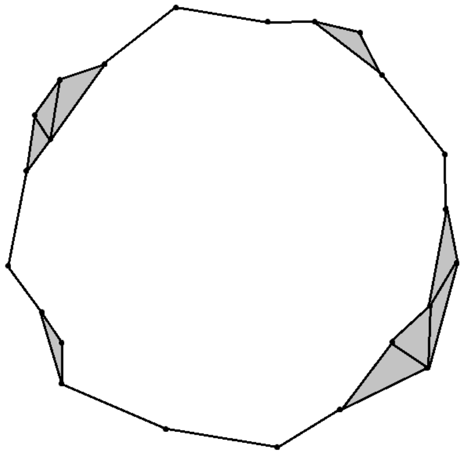
Persistence is a measure of how long a certain homological feature lasts. Longer persistence means a greater likelihood of a given feature being significant. For example, suppose the true shape of our space is a circle, and our data closely resembles a circle, with some noise.



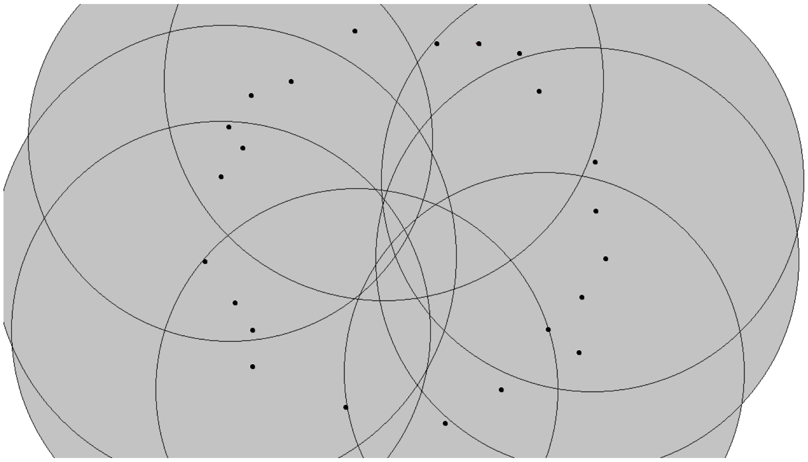
With very small radius, there will be a number of components, and possibly some loops appearing. But these features should not persist very long.



At a relatively small radius, we should arrive at a single component, and find a loop corresponding to the inside of the circle.



This loop persists for a much longer time, until we test at a large enough radius to fill in the center of the circle entirely.



We conclude that the components and loops appearing early are most likely not significant, but that the long persisting loop is likely significant to the shape of our data.

11.2 An algorithm to compute persistent homology

Let K denote a simplicial complex, and $f : K \to \mathbb{R}$ a monotonic function. Recall that a function is **monotonic** if whenever $\sigma \leq \tau$ (meaning σ is a face of τ), $f(\sigma) \leq f(\tau)$. A **compatible ordering** of K is an ordering $K = \{\sigma_1, \ldots, \sigma_m\}$ such that $i < j$ if $f(i) < f(j)$ or if $\sigma_i \leq \sigma_j$.

Once K is ordered, we define the **boundary matrix**, denoted ∂ by

$$\partial[i, j] = \begin{cases} 1 & \text{if } \sigma_i \text{ is a codimension 1 face of } \sigma_j \\ 0 & \text{otherwise.} \end{cases}$$

The **codimension** of a face is just the dimension of the simplex minus the dimension of the face. So a codimension 1 face of a k-simplex is a face which is a $(k-1)$-simplex.

The algorithm will make use of the following function.

```
    low(j): row index of the lowest 1 in column j.
```

Recall that our definition of homology uses $\mathbb{Z}/2$ coefficients, so we can write ∂ with all entries either 0 or 1. We say that a matrix R is **reduced** if $\texttt{low}(j) \neq \texttt{low}(j_0)$ whenever $j \neq j_0$.

The algorithm to reduce ∂ using column operations is as follows

```
R = boundary matrix
for j=1 to m
    while there exists j0 < j with low(j0) = low(j)
        add col j0 to col j
    end while
end for
```

We will use a couple examples to illustrate the application of this reduction to persistent homology.

11.2.1 An example

Let's return to our sensory network example and use the monotonic function which sends the simplex we called i to the real number i, i.e. $f(1) = 1$, $f(2) = 2$, etc. Note that the numbering we have given the simplexes is a compatible ordering. We create the matrix ∂ by recording the boundary pieces of each simplex j in the jth column. For example, 4 has boundary formed by 1 and 2, so in column 4 there is a 1 in the 1st and 2nd rows, and 0's elsewhere. The 0's have been ommitted.

$$\partial = \begin{array}{c|ccccccc} & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ \hline 1 & & & & 1 & 1 & & \\ 2 & & & & 1 & & 1 & \\ 3 & & & & & 1 & 1 & \\ 4 & & & & & & & 1 \\ 5 & & & & & & & 1 \\ 6 & & & & & & & 1 \\ 7 & & & & & & & \end{array}$$

Let's run the algorithm on this matrix.

- Nothing happens with columns 1, 2, and 3.
- Column 4 has lowest 1 in row 2, and because there are no 1's to the left of this entry, we move on to column 5.
- The lowest 1 in column 5 is in row 3, and there are no 1's to the left of this entry, so we move on to column 6.
- This is the first interesting column. The lowest 1 is in row 3. Since column 5 is the first column with a 1 in row 3, we add column 5 to column 6. Now the lowest 1 in column 6 is in row 2, but column 4 is the first column with a 1 in row 2, so we add column 4 to column 6 as well. This results in column 6 consisting of all 0's so we move on to column 7.
- The lowest 1 in column 7 is in row 6. Since there are no 1's to the left of this entry, we are finished.

The result is the matrix

$$R = \begin{array}{c|ccccccc} & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ \hline 1 & & & & 1 & 1 & & \\ 2 & & & & \boxed{1} & & & \\ 3 & & & & & \boxed{1} & & \\ 4 & & & & & & & 1 \\ 5 & & & & & & & 1 \\ 6 & & & & & & & \boxed{1} \\ 7 & & & & & & & \end{array}$$

We can keep track of the column operations we performed in another matrix V, which for our example is given by

$$V = \begin{array}{c|ccccccc} & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ \hline 1 & 1 & & & & & & \\ 2 & & 1 & & & & & \\ 3 & & & 1 & & & & \\ 4 & & & & 1 & & 1 & \\ 5 & & & & & 1 & 1 & \\ 6 & & & & & & 1 & \\ 7 & & & & & & & 1 \end{array}$$

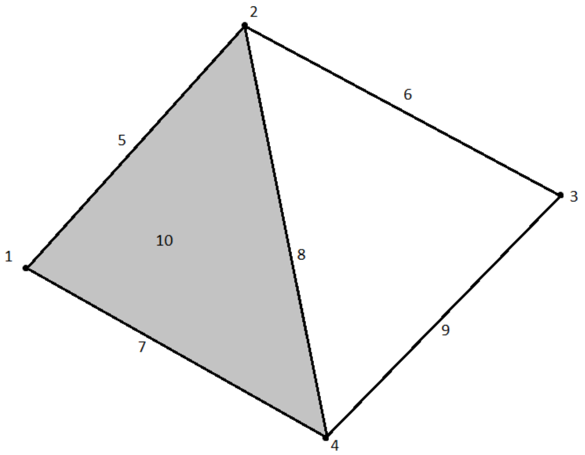
The way to interpret this matrix is the following. The jth column of R is obtained by adding columns of ∂ corresponding to nonzero entries in the jth column of V (e.g. column 6 of R is the sum of columns 4, 5, and 6 of ∂). The algebraic relationship between R, ∂, and V is

$$R = \partial V.$$

Consider the boxed entries in R. These are the values which are the lowest 1's in a column. They are in positions $(2, 4)$, $(3, 5)$, and $(6, 7)$. These pairs of numbers should look familiar. The first entry corresponds to a homological feature being created, and the second entry records when that feature died. The first entry also tells us what dimension of homology the feature belongs to. Since 2 and 3 are 0-simplexes, the 1's in positions $(2, 4)$ and $(3, 5)$ correspond to components, and are reflected in β_0. Since 6 is a 1-simples, the 1 in position $(6, 7)$ corresponds to a loop, which is reflected in β_1.

11.2.2 Another example

Let's run our algorithm on another simplicial complex:



The matrix ∂ is given by

$$\partial = \begin{array}{c|cccccccccc} & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \\ \hline 1 & & & & & 1 & & 1 & & & \\ 2 & & & & & 1 & 1 & & 1 & & \\ 3 & & & & & & 1 & & & 1 & \\ 4 & & & & & & & 1 & 1 & 1 & \\ 5 & & & & & & & & & & 1 \\ 6 & & & & & & & & & & \\ 7 & & & & & & & & & & 1 \\ 8 & & & & & & & & & & 1 \\ 9 & & & & & & & & & & \\ 10 & & & & & & & & & & \end{array}$$

Let's reduce to R.

- Nothing happens for the first 7 columns.
- We add column 7 to column 8, then column 5 to column 8.
- We add columns 7, 6, and 5 to column 9.
- Nothing happens to column 10.

The result is

$$R = \begin{array}{c|cccccccccc} & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \\ \hline 1 & & & & & 1 & & 1 & & & \\ 2 & & & & & \boxed{1} & 1 & & & & \\ 3 & & & & & & \boxed{1} & & & & \\ 4 & & & & & & & \boxed{1} & & & \\ 5 & & & & & & & & & & 1 \\ 6 & & & & & & & & & & \\ 7 & & & & & & & & & & 1 \\ 8 & & & & & & & & & & \boxed{1} \\ 9 & & & & & & & & & & \\ 10 & & & & & & & & & & \end{array}$$

The 1's in (2,5), (3,6), and (4,7) tell us that new components were created at stages 2, 3, and 4, and each persisted 3 steps. The 1 in (8,10) tells us that a loop was created at stage 8, which persisted for 2 steps.

Now look back at our simplicial complex. What about the homology with infinite persistence? There was a loop created at stage 9 which is unaccounted for, and we should also be able to track the component which was created at stage 1. How does the matrix R encode this information? Look at the columns consisting of all 0's. They are columns 1, 2, 3, 4, 8, and 9. The numbers 2, 3, 4, and 8 appeared as birth times of features which died. The remaining indexes for 0-columns (1 and 9) correspond to features which did not die. Since 1 is a 0-simplex, a new component was created at stage 1 which did not die. Since 9 is a 1-simplex, a new loop was created at stage 9 which did not die either.

More examples will be given next lecture.