AMAT 584, Lec 39 4/29/20

<u>Today</u>: Vietoris-Rips Example from last time, continued.
Remarks on Practical computation

Example, continued:

$X = \{(0,0), (2,0), (0,2), (2,2)\}$

$(0,0) = a$, $(2,0) = b$, $(0,2) = c$, $(2,2) = d$

Now we give names to the simplices of dimension at least 1:

name	simplex	birth (β)
e	$[a,b]$	1
f	$[b,c]$	1
g	$[c,d]$	1
h	$[a,d]$	1
i	$[a,c]$	$\sqrt{2}$
j	$[b,d]$	$\sqrt{2}$
k	$[a,b,c]$	$\sqrt{2}$
l	$[a,b,d]$	$\sqrt{2}$
m	$[a,c,d]$	$\sqrt{2}$
n	$[b,c,d]$	$\sqrt{2}$
o	$[a,b,c,d]$	$\sqrt{2}$

Note that for each $j \geq 0$ the alphabetical order on j-simplices is compatible with the order in which the j-simplices are born.

$\Rightarrow$ We may construct the matrix D using the alphabetical order.

Here is D:

	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
	birth 0				birth 1				birth $\sqrt{2}$						
a					1	0	0	1	1	0					
b		0			1	1	0	0	0	1		0			0
c					0	1	1	0	1	0					
d					0	0	1	1	0	1					
e											1	1	0	0	
f											1	0	0	1	
g		0					0				0	0	1	1	0
h											0	1	1	0	
i											1	0	1	0	
j											0	1	0	1	
k															1
l		0					0					0			1
m															1
n															1
o		0					0					0			0

Now we apply the standard reduction to D. Here's what we end up with:

	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
a					1	0	0	0	0	0					
b		0			1	1	0	0	0	0		0			0
c					0	1	1	0	0	0					
d					0	0	1	0	0	0					
e											1	1	1	0	
f											1	0	1	0	
g		0					0				0	0	1	0	0
h											0	1	1	0	
i											1	0	0	0	
j											0	1	0	0	
k															1
l		0					0					0			1
m															1
n															1
o		0					0					0			0

The column/pivot pairings are:

dim 0/dim 1: (b, e) (c, f), (d, g)

dim 1/dim 2: (i, k) (j, l), (h, m)

dim 2/dim 3: (n, o)

The only 0 column which does not appear as a pivot row is column a.

So the 0^{th} barcode is

$$\{\underbrace{[\beta(a),\infty)}_{=[0,\infty)},\ \underbrace{[\beta(b),\beta(e))}_{=[0,1)},\ \underbrace{[\beta(c),\beta(f))}_{=[0,1)},\ \underbrace{[\beta(d),\beta(g))}_{=[0,1)}\},$$

$$=\{[0,\infty),[0,1),[0,1),[0,1)\}.$$

The 1^{st} barcode is:

$$\{\underbrace{[\beta(i),\beta(k))}_{=[\sqrt{2},\sqrt{2})},\ \underbrace{[\beta(j),\beta(\ell))}_{=[\sqrt{2},\sqrt{2})},\ \underbrace{[\beta(h),\beta(m))}_{=[1,\sqrt{2})}\}$$

(where it is understood that elements of the form $[r,r)$ are ignored)

$$=\{[1,\sqrt{2})\}.$$

The 2^{nd} barcode is:

$$\{\underbrace{[\beta(n),\beta(o))}_{=[\sqrt{2},\sqrt{2})}\}=\{\}.$$

All higher barcodes are also empty.

<u>Remark</u>: Lots of the pairs computed in this example gave terms of the form $[r,r)$. This is quite typical in computations of Vietoris-Rips barcodes, and recent optimized algorithms exploit this to save a lot of time + memory (e.g. Bauer's Ripser software).

Remarks on Persistence Computation

Alpha/Čech filtrations.

Note: We have not explained how to algorithmically compute Čech filtrations or Delaunay/Alpha filtrations.

Both rely on computational geometry ideas will not have time to discuss in this course.

Čech filtrations are rarely used in practical computations (there are expections, and code is available, e.g. in the GUDHI library).

Alpha filtrations are very readily computable for data embedded in low dimensions (say, $\mathbb{R}^3$).

A really important question:

What size data sets can we handle?

There's no simple answer.

The answer depends on many things:

- Which barcode(s) am I trying to compute? 0^{th}? 1^{st}? 2^{nd}?
- Which filtration am I considering?

For data in $\mathbb{R}^3$, persistent homology computations using Alpha filtrations scale quite well.

Hundreds of thousands of points are feasible.

For Rips filtrations, "naive" computations of the 1^{st} persistence barcode, using state-of-the art software become difficult for, say, 5000-10,000 points on a recent consumer-grade laptop.

Memory, not, speed is almost always the issue, so using a computer with lots of memory can raise the ceiling a little bit.

- Do I need to construct the filtration for all $r \in [0,\infty)$, or can I truncate the construction for some smaller r?

If I know something in advance about the structure of my data, I may be justified in truncating. But often this is undesirable.

With some truncation, we can handle much larger data sets with the Rips construction.

- Do I need the exact barcode, or does an approximation suffice?

There are both simple and sophisticated approximation methods that can signifantly improve scalability

These methods come with theoretical guartees which are stongest when the data has low intrinsic dimensionality.

Computational cost is a serious limitation of TDA methods, for high-dimensional data!

But for many types of data, the methods scale well enough to be very useful.

And state-of-the-art methods keep getting better.