MATH 529 – Lecture 22 (03/28/2024)

Today: * more details of pairing/persistence
* matrix algorithm for persistence

Recall: $\emptyset = K^0 \subseteq \dots \subseteq K^m = K$ a filtered simplicial complex.

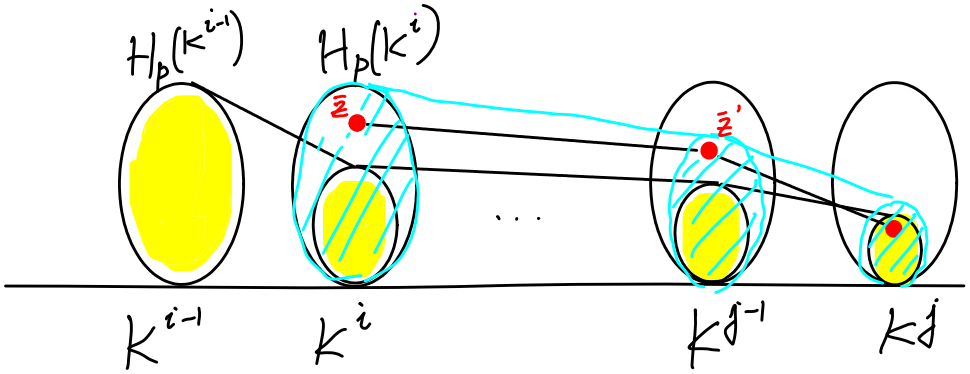
→ change of notation: we had $H_k^{l,p}$ previously; we'll need i,j,k,l,p soon!

$$H_p^{i,j} = Z_p(K^i) \Big/ \left(B_p(K^j) \cap Z_p(K^i)\right), \quad \beta_p^{i,j} = \operatorname{rank} H_p^{i,j}$$

$j > i$, usually.

$H_p^{i,j}$ captures non-trivial p-classes in K^i that stay active in K^j.
→ cycles that are not boundaries

We want to count exactly the # classes that are born in K^i and die in K^j. How do we do that? Recall the following picture (from Lecture 20):



We have $\beta_p^{i,j-1}$ = # classes in $H_p(K^i)$ that are still active in K^{j-1}

$\beta_p^{i,j}$ = # { subset of these classes that remain active in K^j }

$\beta_p^{i,j-1} - \beta_p^{i,j}$ = # classes active in K^i that die entering K^j.

Similarly, $\beta_p^{i-1,j-1} - \beta_p^{i-1,j}$ = # classes alive in K^{i-1} that die entering K^j.

Hence we get

$$\mu_p^{i,j} = \left(\beta_p^{i,j-1} - \beta_p^{i,j}\right) - \left(\beta_p^{i-1,j-1} - \beta_p^{i-1,j}\right)$$

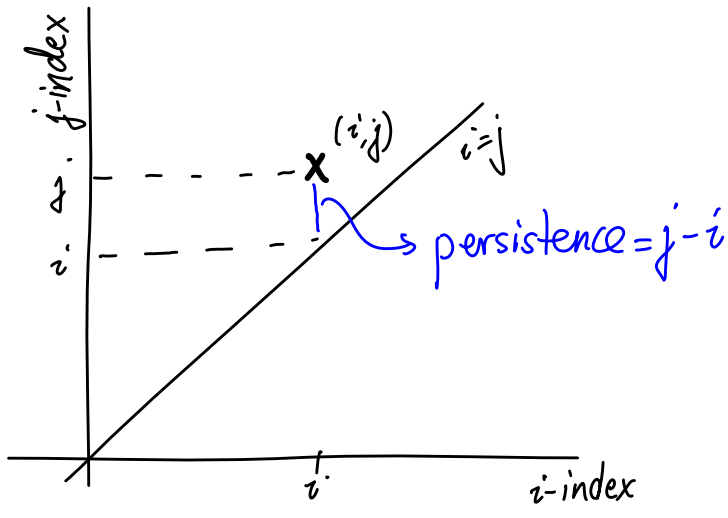
as the number of p-dimensional classes born at K^i and die entering K^j.

$\mu_p^{i,j}$ counts multiplicities (there could be more than one class born at K^i that dies entering K^j in general; but when we insist that $K^j = K^{j-1} \cup \{\sigma^j\}$, only one class will die).

p^{th}-persistence diagram

We pair σ^i a positive p-simplex with σ^j a negative $(p+1)$-simplex to capture a particular p-dimensional feature (or class). We plot the point (i,j) for each such pair on the index-index plane.

Since we assume a monotonic filtration, i.e., all proper faces are already present when simplex σ^j enters, we have that $i<j$ for all such pairs (i,j).



Hence all points are plotted above the $i=j$ ($45°$) diagonal.

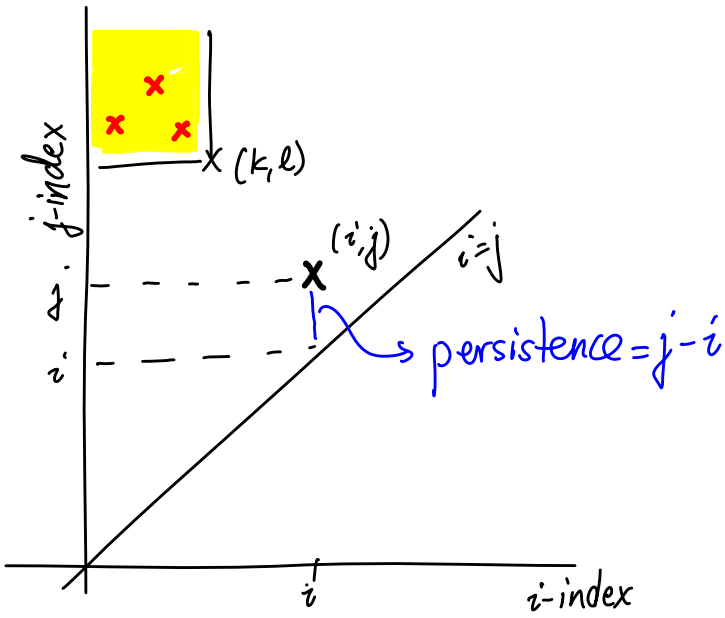
The persistence of (i,j) is then the vertical distance from the $i=j$ line ($45°$ line). In other words, the higher a point is above the diagonal line, the more persistent is the corresponding feature.

Technically, we also include $\{+\infty\}$ in both axes. Recall that unpaired positive simplices are assigned persistences of $+\infty$, and hence we plot the points $(i,+\infty)$ corresponding to such a simplex σ^i.

We can count the points in the persistence diagram to compute $\beta_p^{k,\ell}$:

$\beta_p^{k,\ell}$ = # points (with multiplicities) in the upper left quadrant with corner point (k,ℓ)

$\beta_p^{k,\ell} = 3$, as shown here using ×'s.



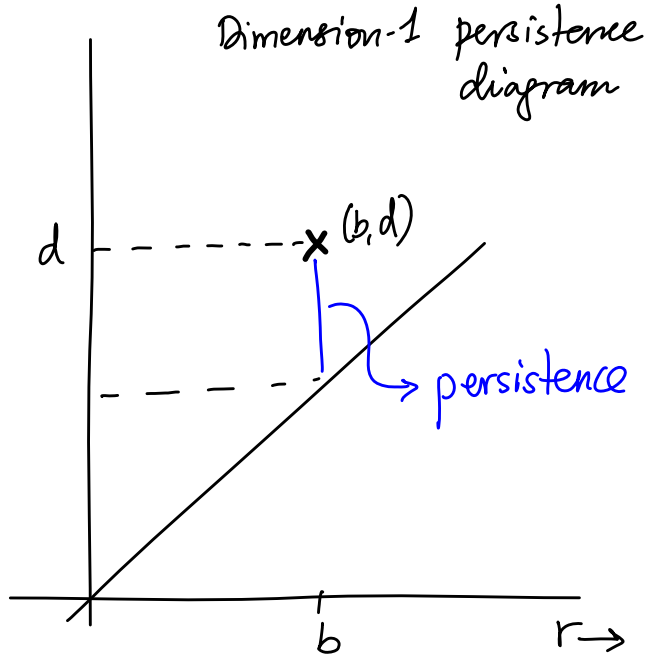
Fundamental lemma of persistent homology

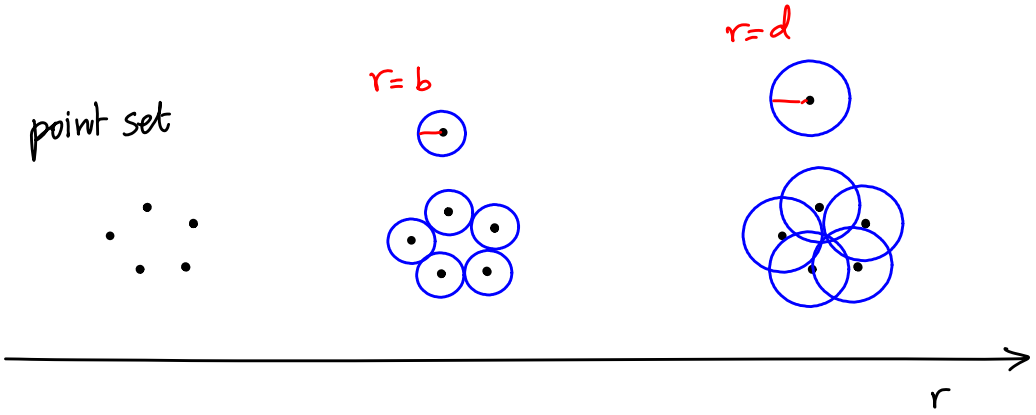
$$\beta_p^{k,\ell} = \sum_{i \le k} \sum_{j > \ell} \mu_p^{i,j} \qquad \text{for } k \le \ell.$$

$\mu_p^{i,j}$ → plotted (with multiplicities) in the persistence diagram.

Radius-based persistence diagram

While we've presented the entire pipeline of persistence for index-based filtrations, similar diagrams are obtained for points sets using, e.g., VR complexes, in terms of the radius values. We record pairs as (b, d), for instance, where the feature is born at $r = b$, and dies at $r = d$.

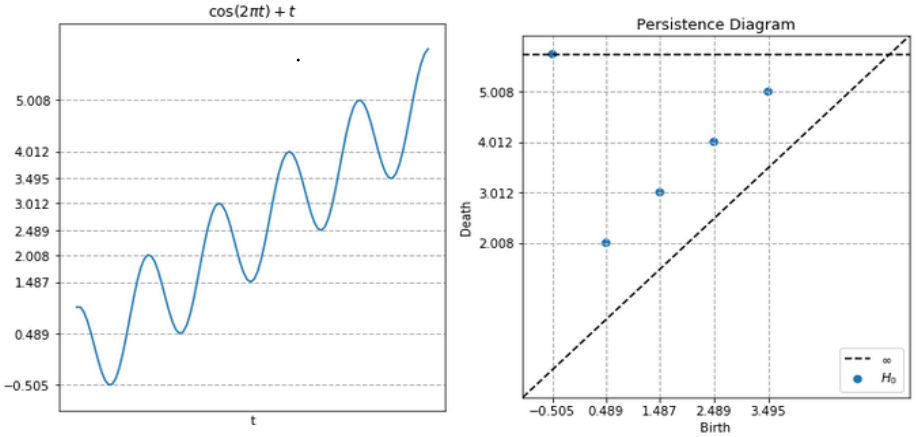




Sublevel-set persistence

Similar persistence diagrams are obtained when tracking the topology of evolving sublevel sets of functions.

(image from scikit-tda.org)



Matrix Reduction

We devise a matrix reduction algorithm to compute all pairings, i.e., compute the persistence diagram. The algorithm is motivated by the SNF algorithm to compute ranks of Z_p, B_p, H_p from $[\partial_p]$ $\forall p$. But we can also get the (σ^i, σ^j) pairings in the process!

We combine all boundary matrices with simplices ordered as in the monotonic filtration (σ^i enters in K^i, and all its faces are already present).

$$\partial[i,j] = \begin{cases} 1 & \text{if } \sigma^i \prec \sigma^j \text{ and } \dim(\sigma^i) = \dim(\sigma^j) - 1; \\ 0 & \text{otherwise} \end{cases}$$

→ in the final complex

$[\partial]$ is an $m \times m$ $\{0,1\}$-matrix (m = # simplices).

The filtration info is captured in the order of rows & column, i.e., the rows and columns are ordered the same way as the simplices come in.

We use replacement elementary column operations (over $\mathbb{Z}_2$) to "reduce" $[\partial]$ matrix to a matrix R.

→ $C_j + C_j$

new definition of reduction; not SNF!

Def Let $low(j)$ be the row index of the lowest 1 in column j. We call R **reduced** if $low(j) \neq low(j')$ whenever $j \neq j'$ specify two nonzero columns of R.

The algorithm reduces ∂ to R by adding columns from left to right ($C_j + C_i$ for $j > i$).

Here is the main block/function in the algorithm.

```
void REDUCE
    R = [∂];
    for j = 1 to m do
        while ∃ j' < j with low(j') = low(j) do
            add column j' to column j
        end
    end
```

This algorithm can be shown to run in $O(m^3)$ time.

$[\partial]$ is upper triangular to start with, and by the nature of these ECOs, we get that R is also upper triangular.

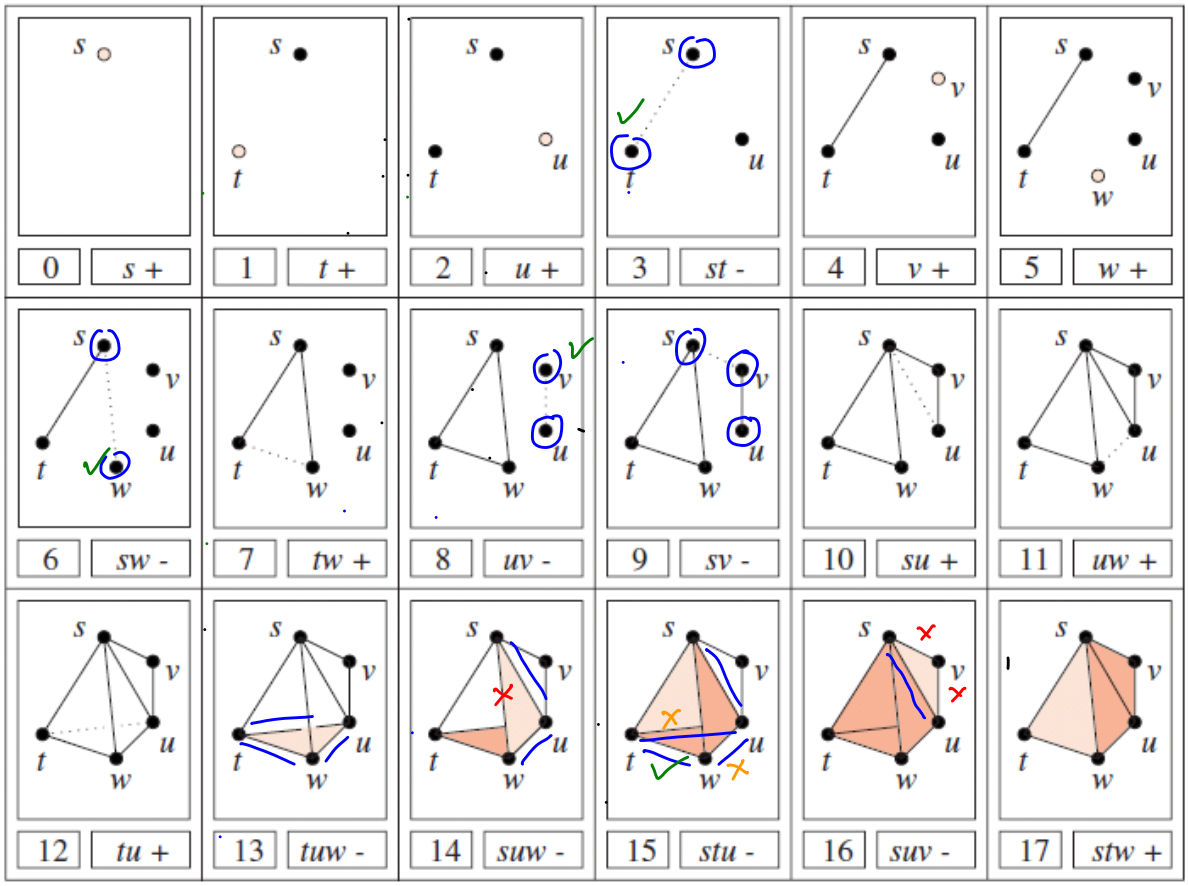
$$R = [\partial] V$$ → elementary matrix

In fact, R, $[\partial]$, and V are upper-triangular.

Note that we may not get block structure for $[\partial]$ in general. But if the filtration has the property that all p-simplices come in before any of the $(p+1)$-simplices do, then we will get a block structure — in fact, $[\partial]$ will be the block assembly of individual $[\partial_p]$ in that case.

Example

We apply matrix reduction to the complex we saw earlier. Recall the filtration and the pairings we obtained.



Pairings

(t, st)

(w, sw)

(v, uv)

(u, sv)

(tu, tuw)

(uw, suw)

(tw, stu)

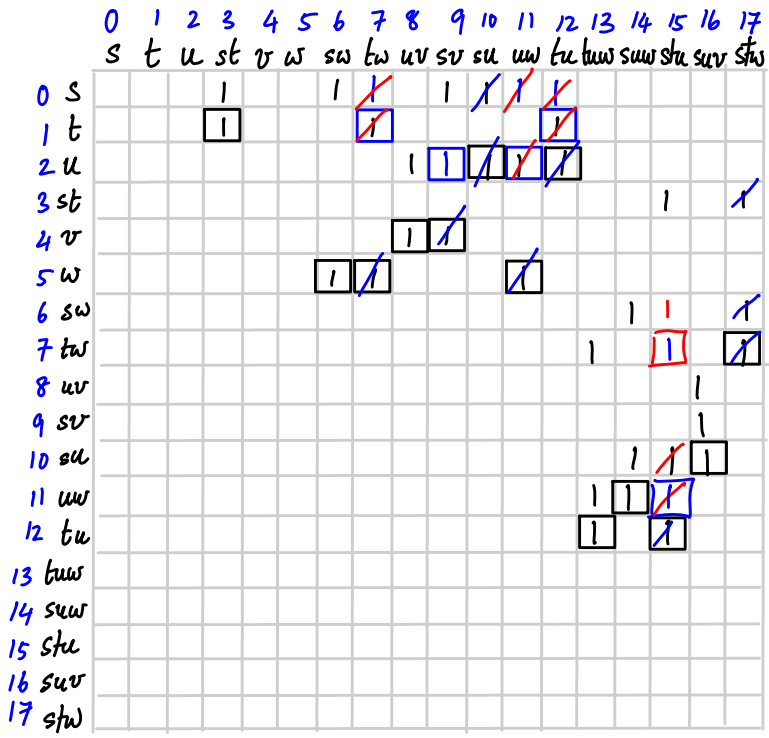
(su, suv)

	0 s	1 t	2 u	3 st	4 v	5 w	6 sw	7 tw	8 uv	9 sv	10 su	11 uw	12 tu	13 tuw	14 suw	15 stu	16 suv	17 stw
0 s				1			1			1	1							
1 t				1				1					1					
2 u									1		1	1	1					
3 st																1		1
4 v									1	1								
5 w							1	1				1						
6 sw															1			1
7 tw														1				1
8 uv																	1	
9 sv																	1	
10 su															1	1	1	
11 uw														1	1			
12 tu														1		1		
13 tuw																		
14 suw																		
15 stu																		
16 suv																		
17 stw																		

Here is the combined boundary matrix. This [∂] is not necessarily in blocks above the diagonal — depending on the filtration, columns could be mixed up (while still maintaining upper triangularity).

We use the following conventions to show steps in the reduction algorithm.

1. $\text{low}(j)$ is indicated by $\boxed{1}$ (a box on the (i,j)-th entry).
2. Changes in the first ECO (for each column) are indicated in blue.
3. Changes in the second ECO (for each column) are shown in red.



ECOs

$j=7$ $\overset{7}{tw} + \overset{6}{sw} + \overset{3}{st}$

$j=9$ $\overset{9}{sv} + \overset{8}{uv}$

$j=10$ $\overset{10}{su} + \overset{9}{sv} + uv$

$j=11$ $\overset{11}{uw} + \overset{6}{sw} + \overset{9}{sv} + uv$

$j=12$ $\overset{12}{tu} + \overset{9}{sv} + uv + \overset{3}{st}$

$j=15$ $\overset{15}{stu} + \overset{13}{tuw} + \overset{4}{suw}$

$j=17$ $\overset{17}{stw} + \overset{15}{stu} + tuw + suw$

pairings:

$\overset{1}{(t,} \overset{3}{st)}$, $\overset{5}{(w,} \overset{6}{sw)}$, $\overset{4}{(v,} \overset{8}{uv)}$, $\overset{2}{(u,} \overset{9}{sv)}$

$\overset{12}{(tu,} \overset{13}{tuw)}$, $\overset{11}{(uw,} \overset{14}{suw)}$, $\overset{7}{(tw,} \overset{15}{stu)}$, $\overset{10}{(su,} \overset{16}{suv)}$

These are the same pairings obtained by the YOUNGEST algorithm!

Details of reduction steps: For $j=7$, we first do $C_7 \overset{+}{\leftarrow}_2 C_6$, as $\text{low}(7) = \text{low}(6) = 5$ at that moment. The $(5,7)$-entry is zeroed out, $(7,0)$-entry is turned to $\underline{1}$, and $\text{low}(7)=1$ after this ECO. Now, $\text{low}(7) = \text{low}(3)$, so we do $C_7 \overset{+}{\leftarrow}_2 C_3$, which zeros out both 1's in the $j=7$ column.