Chapter 4: Persistent Homology
Topics in Computational Topology: An Algorithmic View

1 Persistent homology

1.1 Introduction to persistence

Motivation. Homology measures certain topological property of a given domain. However, in practice there are many scenarios where we wish to consider the homological features (or how they vary) across many different scales.

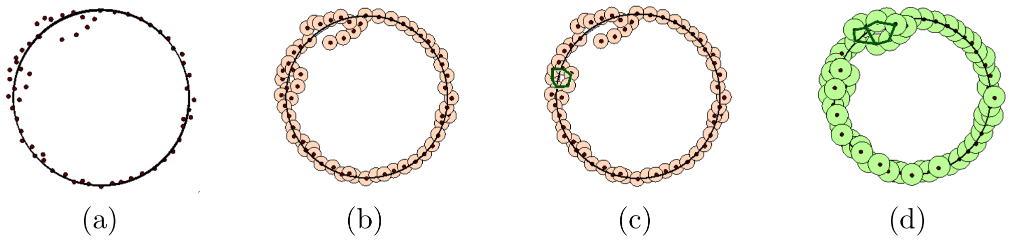


Figure 1: (a) Input points sampling a circle. (b) At a small scale, a spurious cycle is formed in the union of balls as shown in (c). At a larger scale, this cycle is killed, but a new one is created as shown in (d).

For example, impagine that we are given a set of points P sampled from a hidden domain $X \subseteq \mathbb{R}^d$. As we described earlier, we can grow a ball around each sample point in P, and take the union of these balls to approximate the hidden domain (these union of balls form a thicked version of the hidden domain). See Figure 1 for an example. However, what is the correct scale to grow this ball? Furthermore, it turns out that there is *no single good scale* in the sense that for whatever radius r we choose, the resulting union of balls could have spurious noisy features. The observation however is that, these spurious features typically are not persistent: as the radius grows, they disappear quickly (although new spurious features may be created). This suggests that perhaps, we should look at features that "persist" across a range of radius. In other words, imagine we consider a sequence of spaces formed by union of balls with growing radius. Instead of considering homological features (topological loops) at a fixed scale, we wish to track them across these scales.

In fact, it turns out that tracking homological features across growing radius also gives a description of the hidden domain in terms of its embedding. See Figure 2 for an example. Again, while homological features of a fixed space seems to provide too coarse a description of the space, by inspecting how the space changes as we continuously thickening it (via union of growing balls), we can capture geometric properties of the space. Again, in this case, we now have a sequence of growing spaces, and we wish to track topological features along this sequence. This sequence will

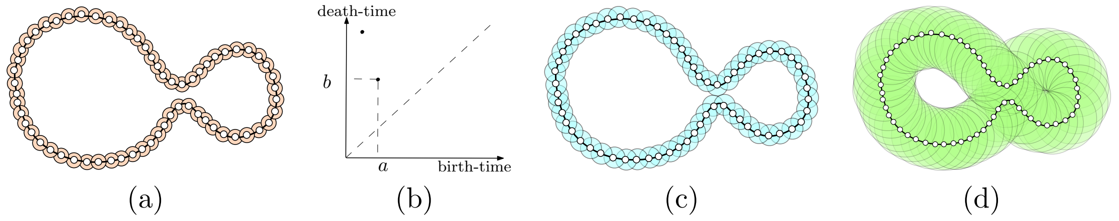


Figure 2: (a) Sample points from a dumbbell curve, and the union of balls $P^r := \bigcup_{p \in P} B(p, r)$ around them captures the thickened space of the curve. (b) The 1-dimensional persistence diagram for the filtration $P^{r_0} \subseteq P^{r_1} \subseteq \cdots \subseteq P^{r_m}$ with $r_0 < r_1 < \cdots < r_m$. The union of balls correspond to the (c) birth time P^a and (d) death time P^b for one 1-cycle.

be modeled by the so-called filtration below, and tracking homological features across this filtration is achieved by the so-called *persistent homology.*

Later, we will see that this persistent homology framework is very powerful – we can also obtain a filtration of the domain induced by any functions (or even more complex maps). This allows us to encode different information of interests through the filtration and its induced persistent homology.

Filtration. Following our intuition above, we now need to model the "growing domain". This is captured by the so-called filtration. Given a simplicial complex K, a *filtration* of K is a sequence of subcomplexes of K:

$$\emptyset = K_0 \subseteq K_1 \subseteq \cdots \subseteq K_{n-1} \subseteq K_n = K. \tag{1}$$

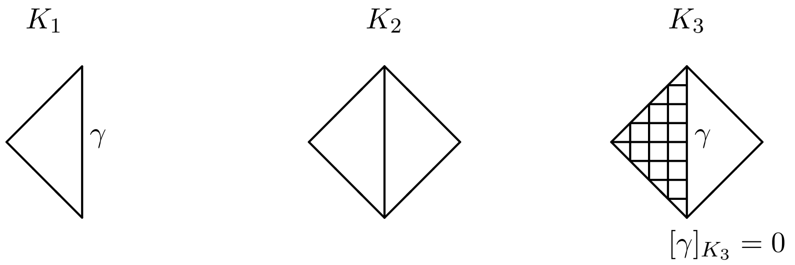
There is a natural sequence of inclusion maps $\iota^i : K_i \to K_{i+1}$, for $i \in [0, n)$. In fact, for any $i < j$, we have $\iota^{i,j} : K_i \to K_j$ which is simply $\iota^{i,j} = \iota^{j-1} \circ \cdots \circ \iota^i$.

Consider an inclusion map $\iota : X \to Y$. It is easy to check that ι will map a cycle c from X to a cycle $\iota(c)$ of Y. Similarly, it will map a boundary cycle from X to a boundary cycle in Y. These imply that it will map homologous cycles to homologous cycles (but not necessarily vice versa). Hence it induces a well-defined map from $\xi_p : \mathrm{H}_p(X) \to \mathrm{H}_p(Y)$. Indeed, given any homology class $h : \mathrm{H}_p(X)$, suppose $h = [\gamma]$. Then we simply map $\xi_p(h) = [\iota(\gamma)]$. To show that this is well-defined, we need to show that the image is the same no matter what representative cycle we take. In other words, for any $\gamma' \approx \gamma$, $\iota(\gamma') \approx \iota(\gamma)$. But this is true as it maps homologous cycles to homologous ones.

This map ξ is linear, that is: $\xi(h_1 + h_2) = \xi(h_1) + \xi(h_2)$. Indeed, let $h_1 = [\gamma_1]$ and $h_2 = [\gamma_2]$, then

$$\xi([\gamma_1] + [\gamma_2]) = \xi([\gamma_1 + \gamma_2]) = [\iota(\gamma_1 + \gamma_2)] = [\iota(\gamma_1) + \iota(\gamma_2)] = [\iota(\gamma_1)] + [\iota(\gamma_2)].$$

In other words, the inclusion $\iota : X \to Y$ incudes a *homomorphism* $\xi : \mathrm{H}_p(X) \to \mathrm{H}_p(Y)$ for any dimension p.



Example 1: Consider the inclusion $K_1 \subseteq K_2 \subseteq K_3$.
A cycle may change from non-boundary cycle to boundary cycle.

Examples 2: Consider the little pant as X, and Y is the one with one pipe closed.

Now consider the filtration in Eqn (1), the inclusion induces a sequence of homomorphism between the sequence of homology groups $\mathrm{H}_p(K_i)$s:

$$\emptyset \to \mathrm{H}_p(K_1) \to \mathrm{H}_p(K_2) \to \cdots \to \mathrm{H}_p(K_n) = \mathrm{H}_p(K). \tag{2}$$

Usually, we write the above sequence, as well as the set of homomorphism between them, as *a persistent module*:

$$\mathbb{P} := \{\mathrm{H}_*(K_i) \mid i \in [0, n]\}_{\xi_*^{i,j}:\mathrm{H}_*(K_i)\to\mathrm{H}_*(K_j)}.$$

We often write it as $\mathbb{P}(\{K_i\}_\xi$, or simply $\mathbb{P}$ when the choice of filtration and maps are clear.

Persistence homology. Eqn (2 gives us a sequence of homology groups (i.e, homological feature spaces for the sequendce of spaces in Eqn (1)), connected by a natural map induced from the inclusion. Now, we wish to talk about how "features" (homological class) evolves along this sequence: In particular, when are new features created, and when do they die (disappear).

Definition 1.1 (Persistent homology groups) *The p-th* persistent homology groups *is the image of the homomorphisms induced by inclusion, that is $\mathrm{H}_p^{i,j} := \mathrm{im}(\xi_p^{i,j})$ for any $0 \le i < j \le n$. The p-th* persistent Betti numbers *are the ranks of these groups: $\beta_p^{i,j} = \mathrm{rank}(\mathrm{H}_p^{i,j})$.*

Intuively, $\mathrm{H}_p^{i,j}$ is the set of homology classes that existed in $\mathrm{H}_p(K_i)$ which also survive till $\mathrm{H}_p(K_j)$. The persistence Betti numbers give the rank of such classes (i.e, the number of independent p-cycles that survive through). Consider the following example:

use the example where K_0 is the empty set. K_1 is an empty triangle. K_2 are two empty triangles sharing one edge. In K_3 we fill the first triangle. Now the persistence homology $\beta_1^{1,3} = 0$. Point out that even though the rank stays the same, no class survives through.

The persistent Betti numbers do not tell us the lifetimes of "features".

Definition 1.2 (Persistence, and persistence diagram) *Consider the following quantity:*

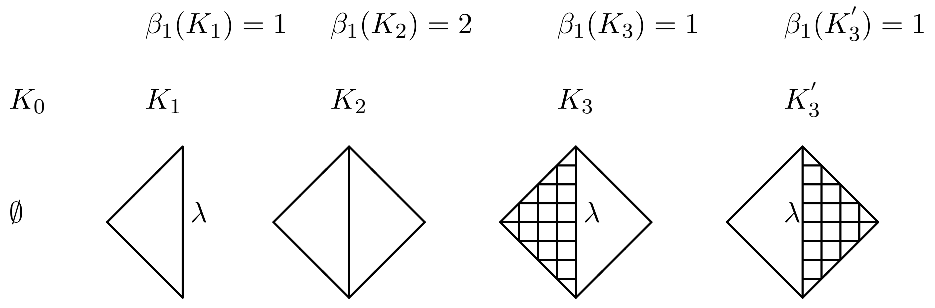
$$\mu_p^{i,j} := (\beta_p^{i,j-1} - \beta_p^{i,j}) - (\beta_p^{i-1,j-1} - \beta_p^{i-1,j}).$$

We call $\mu_p^{i,j}$ the pairing number*.*

The p-th persistence diagram *$\mathcal{D}(\mathbb{P})$ of a* persistence module *$\mathbb{P}$ is the multiset of points in the plane, where a point $(x, y) \in \mathcal{D}(\mathbb{P})$ if $\mu_p^{x,y} \neq 0$, and the multiplicity of (x, y) is $\mu_p^{i,j}$.*

Each point $(x, y) \in \mathcal{D}(\mathbb{P})$ is called a persistence pairing *or* persistence point*. The* persistence *of a persistence pairing $(x, y) \in \mathcal{D}(\mathbb{P})$ is defined as $|y - x|$, which is the lifetime of the corresponding homology class.*

Intuitively, it is the number of independent homology classes that are created by entering i and died at leaving j. If $\mu_p^{i,j} \neq 0$, then we say that i is paired with j and its persistence is defined as $j - i$. In the previous example, $\mu_1^{1,3} = 1$.



Example 2: persistent Betti number $\beta_1^{1,2} = 1$, $\beta_1^{1,3} = 0$, and $\beta_1^{2,3} = 1$.
There are two points $(1,3), (2,4)$ in the associated persistence diagram.

1.2 Persistence diagrams

In the literature, when we talk about persistence algorithm, typically, the input is a filtration. The output is the corresponding persistence diagram. It is also possible that the algorithm also output a set of cycles generating the persistence homology, which we will make more precise later. Note that the persistence diagram contains all persistent pairing numbers $\mu_p^{i,j}$. It is easy to check that one can recover the p-th persistent Betti numbers $\beta_p^{i,j}$ from the pairing numbers:

$$\beta_p^{k,l} = \sum_{i \leq k, l < j} \mu_p^{i,j}$$

Thus, $\beta_p^{k,l}$ corresponds to the left quadrant of the dimension p persistence diagram at point (k,l).

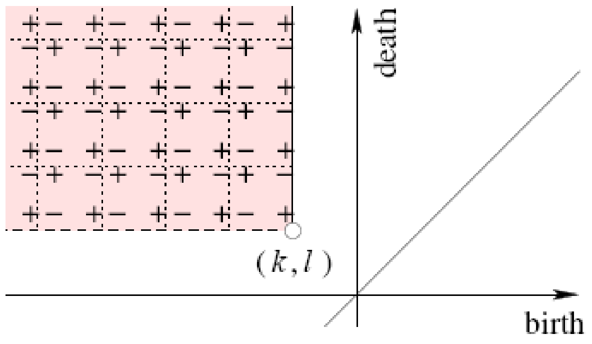


Figure 3: Persistence diagram : $\beta_p^{k,l}$ corresponds to the left quadrant.(Image courtesy of H. Edelsbrunner)

There are much information we can read off from the persistence diagram.

In general, pick any time i, j, take the horizontal and vertical line at (i,j). Persistent points in the left-top quadrant cornered at (i,j) correspond to homology classes created before i but died after j. So total number of points are $\beta_p^{i,j}$. Left slab: things created before i. Right-bottom triangle: things created after i and died before j.

Now if we pick (i,i), and look at the left-top quadrant: these are exactly the homology of K_i. If we look at (n,n), these are the homology of $K = K_n$. These are also the *essential homology.* They are homology classes that are created, but are never destroyed (or destroyed at inifity).

1.3 Alternative View

I now would like to describe an alternative way to look at the persistence pairing, to help us understand it better, in particular, to understand what the creation and death are.

For simplicity, assume that the input filtration $K_0 \subset K_1 \subset \cdot \subset K_n$ is such that $K_0 = \emptyset$, and $K_{i+1} \setminus K_i = \sigma_i$. Hence this filtration is induced by an ordering of simplices $\langle \sigma_1, \ldots, \sigma_n \rangle$.

Birth and death. Imagine that we add the p-simplex $\sigma = \sigma_i$ to K_{i-1} to obtain K_i. Since σ is of dimensin p, it will only change the p-th chain group. (Use a triangle as an example: suppose we have a tetrahedron missing one face, with an extra empty triangle hanging on it.) It turns out that when we add σ, only two cases could happen.

(Case 1): σ is a creator.

When we add σ, it is possible that σ will form a new p-cycle. If that happens, then this new p-cycle must belong to a new homology class. This is because it contains σ and there is no $p+1$-face co-facing σ. Hence no $(p+1)$-chain can have σ on the boundary. In this case, a creation is happening. In fact, it may form a family of new cycles: use the last edge of a tetrahedron as an example. However, the number of independent cycle only increases by 1 if we only add one-simplex. This can be proved algebraically, and can also be seen intuitively: all new cycles created will contain σ. Assume we have an old basis $\{\gamma_1, \ldots, \gamma_m\}$. Pick any of them as a new basis element γ_{m+1}. Then any other cycle γ can be written as this one plus some combination of old ones as $\gamma + \gamma_{m+1} \subseteq K_i$ can be written as linear combination of old basis.

In this case, $\beta_p + +$, as $z_p + +$ and no other ranks change. Hence all other betti-numbers remain the same. This is why we can σ a *creator* in this case, which creates a new family of p-homology classes.

(Case 2): σ is a destroyer.

If σ does not create a new p-cycle. Then what happens? That means that the boundary bdσ was a non-trivial $(p-1)$-cycle in K_i, and now it is filled. Why is bdσ non-trivial? If it is trivial, then adding σ will form a 2-cycle! Again, look at the tetrahedron example, both for the edge case and for the triangle case. In this case, a *death* event happens. In fact, similar to the argument above, only one independent homology class can be killed by adding a single simplex.

In this case, $\beta_{p-1} - -$, as $b_{p-1} + +$. No other betti-number changes. This is why we call σ a *destroyer* in tis case, which kills a family of $p-1$-homology classes by making them null-homologuous.

To summerize, by adding a p-simplex, either β_p ++ (in which case z_p++, and no other number change); or β_{p-1} −− (in which case b_{p-1} ++ and no other number change). The former corresponds to a birth event, and the latter corresponds to a death event. Every addition of a simplex will cause one of these two events to happend. A simplex is called a *creator* and a *destroyer* depending on its role.

Formally, what does it mean to create? That means that there is a homology class $h \in \mathrm{H}_p(K_{i+1})$ such that it does not have pre-image under ξ_p^i in $\mathrm{H}_p(K_i)$. What does it mean to destroy? There is a non-trivial homology class $h \in \mathrm{H}_p(K_i)$ whose image $\xi_p^i(h)$ is zero in $\mathrm{H}_p(K_{i+1})$.

Pairings of creators and destroyers. So this helps to explain creation and destroyment better. What are persistence pairing? Intuitively, if we have a persistence pairing (i, j), then the addition of σ_j will kill a homology class created when adding σ_i. In other words, there is a p-cycle containing σ_i that is filled only when adding σ_j (not before!). We will see in next section when we talk about algorithms to compute it, how this pairing is identified.

2 Persistent Algorithm

For simplicity, we assume that the filtration is induced by a total ordering of simplices $\langle \sigma_1, \ldots, \sigma_n \rangle$ with $K_i = \{\sigma_1, \ldots, \sigma_{i-1}\}$. Note that for this ordering to induce a valid filtration, it needs to satisfy the property that for any σ, its faces are before it in this ordering.

We now describe an algorithm where the input is $\langle \sigma_1, \ldots, \sigma_n \rangle$ (which induces a filtration $\mathcal{K}$, and the output is the set of all persistence pairings (thus the persistence diagram $\mathcal{D}(\mathcal{K})$). The algorithm is a simple matrix reduction algorithm, very similar to the Gaussian elimination we discussed earlier. It operates on the boundary matrix A.

Algorithm 1 Right-Reduction(A)

```
R = $A$;
for $j = 1 \to m$ do
  while there exists $j_0 < j$ with $lowId(j_0) = lowId(j)$ do
    add column $j_0$ of $R$ to column $j$ of $R$
  end while
end for
```

Recall Lemma 2.6 from Chapter 3:

Lemma 2.1 (Lemma 2.6 of Chap. 3) *(1) A matrix is in reduced form if all non-zero columns have unique lowest index.*
(2) Supposed a reduced matrix R is obtained from a boundary matrix ∂_p by row and column operations. Then the set of non-zero columns form a basis for B_p.

Then we note that the algorithm Right-Reduction simply scans columns of R one by one, and continues to reduce the partial matrix $A_i := \{col_A[1], \ldots col_A[i]\}$ that we have already scanned, to decide whether a new simplex is creator or destroyer. We elaborate this view further below.

Explaination of the algorithm via matrix view. Consider the matrix R_{k-1} corresponding to the matrix at iteration k before we add simplex σ_k. Now the addition of σ will add one more column and one more row to R_{k-1} to obtain R_k. We would like to know the effect of the simplex σ, that is, whether it is a creator or a destroyer. How shall we do that?

We can try to eliminate the last column col_k to see whether we can zero-out this column or not. We will only perform the following *right-additions*. That is, we can only add a column from the left of k to col_k. Suppose that after a while, we zero-out this column. What does it mean? This means that this column is a creator. Why? Imagine that we maintain the p-chain to each column again. Each time we add a left column to col_k, we add the corresponding p-chain as well. Note that σ is always in the p-chain $\rho(k)$ we associate with column-k. At the end, if col_k is zeroed out, then there exists a p-chain containing σ that is a cycle. This cycle must be new. Hence σ creates new homology classes and $\beta_p(K_k) = \beta_p(K_{k-1}) + 1$.

Now suppose that there is no way to zero out this column. What can we conclude in this case? Intuitively, to check whether this column vector is linearly dependent on all the previous column

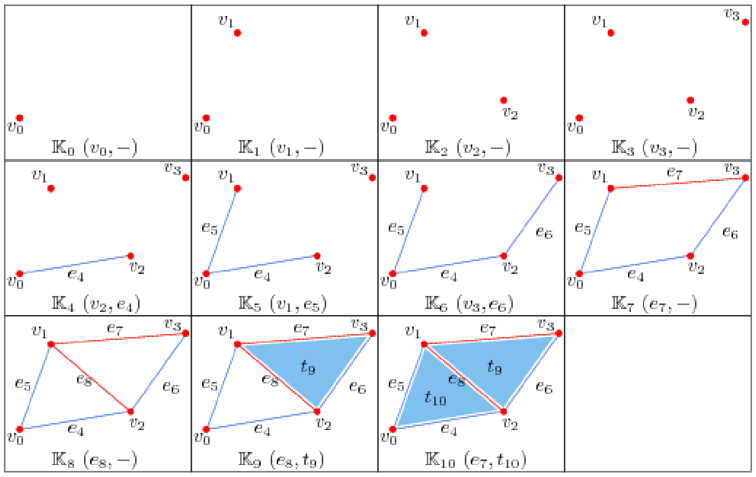


Figure 4: Red simplices are positive and blue ones are negative. The simplices are indexed to coincide with their order in the filtration. $(\cdot, \cdot)$ in each subcomplex shows the pairing between the positive and the negative. The second component missing in the parenthesis shows the introducing of a positive simplex. [Image courtesy of T. Dey]

vectors or not, (i.e, whether we can zero it out or not), we need to main a basis for all the previous columns (i.e, for R_{k-1}).

We thus perform the following strategy. At any moment, the matrix R_{k-1} remains *reduced*, which further implies that the current non-zero columns in R_{k-1} are all independent, forming a basis for B_{k-1}.

Next, after we add σ to the reduced matrix R_{k-1}, we do the cleaning of lowerest index strategy. At the end, if we zero-out this column, then it is the above case. If not, then let $j = lowId_k$. Let γ denote the $(p-1)$-cycle contained in col_k after the reduction. We know the following:

(1) $\partial\sigma$ is homologuous to γ, and γ contains σ.

(2) $\partial\sigma$ is non-trivial in K_{k-1}, but becomes a boundary cycle in K_k. That it is non-trivial in K_{k-1} follows from the fact that it cannot be written as a linear combination of basis from R_{k-1}, in which case, this column will be zeroed-out.

(3) The cycle γ is obviously created when adding σ_j, since it is the youngest in γ. In fact, the homology class $[\gamma]$ was also created when adding σ_j. This is because this is a filtration. So when σ_j was added, it has no co-face. More importantly, this class exists before time i and is only killed at time i. This follows from (2) above.

To summerize, σ kills the homology class $[\partial\sigma]$, which was created at time $j = lowId_k$.

Theorem 2.2 *Let R be the matrix after the reduction algorithm. There is a persistence pairing $(i,j) \in \mathcal{D}(\mathcal{K})$ if and only if $i = lowId_j$. Furthermore, the non-zero column $col_R[j]$ is a $(p-1)$-cycle represents a homology class that was created at time $i = lowId_j$ and killed at time j.*

General reduced form. In fact, what we have discussed does not depend on the specific reduction algorithm we use.

Definition 2.3 (Unique-LowId-Reduced form) *A matrix is in* unique-lowId reduced form *if all non-zero columns have unique lowest index.*

Now assume that we reduce a matrix to its unique-lowId reduced form using only right-additions (but in any order as one wish).

Claim 2.4 *Let R be a unique-lowId reduced form of the boundary matrix M using only right-additions. Then there is a persistence pairing (i, j) if and only if $i = lowId_j$.*

This suggests that we can come up with different persistence algorithm. For example, consider the following "add-forward" algorithm. We again inspect the matrix from left to right. At each step i, we add col_i to any column col_j such that $j > i$ and $col_j[lowId_i] = 1$. Easy to prove that after this strategy, at the end, we obtain a reduced form of the input matrix. Hence we can obtain persistence pairing this way as well.

References