Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Institute of Theoretical Computer Science
Patrick Schnider

Introduction to Topological Data Analysis **Scribe Notes 19** **FS23**

Scribe notes by Simon Weber. Please contact me for corrections.
Lecture date: May 5, 2023
Last update: Friday 5th May, 2023, 14:00

Reeb Graphs

The idea of Reeb graphs is that given some topological space X, and some function $f : X \to \mathbb{R}$, we consider the preimage of f for some fixed value $a \in \mathbb{R}$. We place one point per connected component of the preimage. We do this for some values in $\mathbb{R}$, and connect the points corresponding to neighboring connected components in adjacent preimages. More formally,

Definition 1. *Let X be some topological space, and f a function $f : X \to \mathbb{R}$. Two points x, y are called equivalent ($x \sim y$), iff $f(x) = f(y) = \alpha$ and x and y are in the same connected component of $f^{-1}(\alpha)$. The* Reeb graph R_f *is the quotient space $X/\sim$.*

To make sure that nothing weird happens due to some things being infinite, we assume all of our functions to be levelset tame:

Definition 2. *A function $f : X \to \mathbb{R}$ is* levelset tame *if*

- *each levelset $f^{-1}(\alpha)$ has finitely many connected components, all of which are path-connected, and*
- *the homology groups of the levelsets only change at finitely many critical values.*

The Reeb graph itself is just a (continuous) topological space. We call it a graph, since it is 1-dimensional. To arrive at a graph as we know it in combinatorics, we will need to discretize it. To discretize the Reeb graph, we need to define vertices and edges. There are many different possibilities of defining vertices and edges to discretize the Reeb graph, but we wish to define some type of minimal one.

Let us look at the neighborhood of some point p in the Reeb graph (as a topological space). We look at how many ways there exist to go from p towards the direction of higher f-value (we call this number the up-degree u), and how many ways to go towards the direction of lower f-value (we call this the down-degree l). Depending on u and l, we classify p as in Table 1.

Table 1: Classifications of points in the Reeb graph.

u	l	Classification
1	1	regular
0	> 0	maximum
> 0	0	minimum
≥ 2	l	up-fork
u	≥ 2	down-fork

Note that a point can fall into multiple of these classes, for example it can be a maximum and a down-fork simultaneously, or an up-fork and a down-fork simultaneously. We call the minima, maxima, up-forks, and down-forks *critical points*. Our discretization places vertices at the critical points.

Note that the graph we get through this process is not necessarily simple, we may have multi-edges.

We next consider merge trees and split trees, which are variants of the Reeb graph, where instead of levelsets, we look at sub-level sets or super-level sets.

Definition 3. *Let X be some topological space, and f a function $f : X \to \mathbb{R}$. We have $x \sim_M y$ for two points x, y, iff $f(x) = f(y) = \alpha$ and x and y are in the same connected component of $f^{-1}((-\infty, \alpha])$. The* merge tree T_M *is the quotient space $X/\sim_M$.*

Note that in the merge tree, since we only increase the space under consideration, we never have a connected component that splits. We can only have new connected components appearing, and connected components merging. This also tells us that the Merge tree (or its discretization) is always a tree.

Definition 4. *Let X be some topological space, and f a function $f : X \to \mathbb{R}$. We have $x \sim_S y$ for two points x, y, iff $f(x) = f(y) = \alpha$ and x and y are in the same connected component of $f^{-1}([\alpha, \infty))$. The* split tree T_S *is the quotient space $X/\sim_S$.*

In computers, we do not like working with arbitrary topological spaces. We thus now look more at Reeb graphs in the context of simplicial complexes. We consider a simplicial complex K and a function $f : K \to \mathbb{R}$, which is piecewise linear (linear on each simplex). We observe that the Reeb graph then only depends on the 2-skeleton of K. This is the case since looking at a levelset is the same as cutting through the simplicial complex. When we cut through a simplex, we generally get a simplex of one dimension lower. In a simplicial complex, connectivity is completely determined by the 1-skeleton. Thus, before cutting, the 2-skeleton suffices. Furthermore, we can see that the critical points are images of the vertices of K. This happens since a connected component can only appear, disappear, split, or merge at some local maximum or minimum of the connected component. Since the function is linear, the maximum or minimum of every simplex is also attained at some vertex.

We define the *augmented Reeb graph* of a simplicial complex with a PL-function, by just taking all the images of the vertices as our graph vertices.

How can we compute this augmented Reeb graph? We can do a discrete sweep (or scan) through the simplicial complex in the order given by f, only stopping at values a such that $f(v) = a$ for some vertex v. In this sweep, we want to keep track of the connected components. The levelset $f^{-1}(\alpha)$ of the 2-skeleton of K is just a graph G_α: vertices and edges of K induce vertices of G_α, triangles induce edges. We can now go through our vertices in order, look at these graphs, and update the connected components.

The runtime of this algorithm is given by the data structure used to manage the connected components. We want a data structure that can update the connected components under insertion and deletions of edges and vertices. There are such data structures that can do each update in amortized time $O(\log m)$, where m is the size of the graph. The size of the graph is bounded by the sum m of vertices, edges, and triangles in K. Each such feature appears at one point, and disappears at one point, and we thus have at most $2m$ insertions and deletions in total, giving an $O(m \log m)$ algorithm.

Homology of Reeb graphs

Since the Reeb graph R_f is a graph (a 1-dimensional object), we have $H_p(R_f) = 0$ for $p \geq 2$.

Observation 5. *For a levelset tame* $f : X \to \mathbb{R}$*, we have* $\beta_0(X) = \beta_0(R_f)$*.*

In other words, the Reeb graph captures the 0-homology of the input space X perfectly, no matter which levelset tame function f we use.

Sadly, the same does not hold for the 1-homology. Let us consider a torus, as in Figure 1. In general, it can be that the choice of function f determines whether we capture a hole or not, consider e.g. a cylinder. Note that for the torus, it is actually the case that no matter which function f we choose, we cannot capture its 1-homology (this is non-trivial to show).

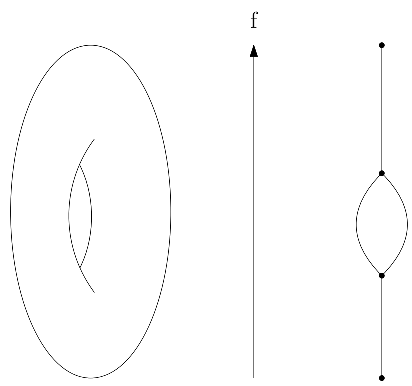

Figure 1: The torus and its Reeb graph.

On the other hand, we can see that every cycle in the Reeb graph is indeed also a cycle in the topological space X, and it cannot be filled in, so it is indeed a hole. Thus we also get the following observation:

Observation 6. *For a levelset tame* $f : X \to \mathbb{R}$*, we have* $\beta_1(X) \geq \beta_1(R_f)$*.*

Can we somehow formalize which holes we lose? To do this, we split up homology into "horizontal" and a "vertical" parts, where horizontal and vertical are of course relative to f.

Definition 7. *A* p*-th homology class* $h \in H_p(X)$ *is called* horizontal *if there is a finite set of values* $A = \{a_1, \ldots, a_k\}$ *such that* h *has a pre-image under the map* $H_p(\bigcup_{a \in A} X_a) \to H_p(X)$ *induced by inclusion, where* $X_a = f^{-1}(a)$*.*

This definition means that we need to be able to find a finite set of levelsets, such that we can find cycles contained in these levelsets, which are in the homology class h in $H_p(X)$.

One now wonders whether the set of horizontal homology classes forms a group. Let this set be $\overline{H_p}(X)$. It turns out that it is indeed a group.

Lemma 8. $\overline{H_p}(X)$ *is a subgroup of* $H_p(X)$*.*

Proof. First, we see that the identity element 0 is in $\overline{H_p}(X)$. We can take an arbitrary set A, and we can always map the 0 element of $H_p(\bigcup_{a \in A} X_a)$ to 0.

Next, we show that the set is closed under addition. Let $p, q \in \overline{H_p}(X)$, and we show that $p + q \in \overline{H_p}(X)$. p has a pre-image in some levelset A_p, and q has a pre-image in some levelset A_q. $p + q$ must have a pre-image in $A_p \cup A_q$.

Finally, we show that the inverse of every element is contained in the group, but since every element is self-inverse in $\mathbb{Z}_2$-homology, we get that for every element its inverse is also contained in $\overline{H_p}(X)$. □

Since the horizontal homology is a sub-group, we can now easily define *vertical homology* by taking quotient groups.

Definition 9. *The* vertical homology group *of* X *with respect to* f *is the group* $\overset{v}{H}_p(X) := H_p(X)/\overline{H_p}(X)$*.*

Observation 10. $\mathrm{rank}(H_p(X)) = \mathrm{rank}(\overline{H_p}(X)) + \mathrm{rank}(\overset{v}{H}_p(X))$.

Fact 11. *The surjection* $\phi : X \to R_f$ *induces an isomorphism* $\overset{v}{\Phi} : \overset{v}{H}_1(X) \to H_1(R_f)$*.*

In other words, when we go from a space X to its Reeb graph, we keep the vertical homology classes, and lose the horizontal ones.

Corollary 12. *Given* X *an orientable connected compact 2-manifold, and a Morse function* $f : X \to \mathbb{R}$*, then* $\beta_1(R_f) = \beta_1(X)/2$*.*

Here, a 2-manifold is a space that locally at every point looks like $\mathbb{R}^2$. Orientable means that there is an inside and an outside side. A Morse function is a "nice enough" function defined in terms of some derivatives, which we do not need to specify here.