Chapter 1

Introduction

In many applications in data science, the data is given to us as a point cloud in some (potentially high-dimensional) space. Interesting data about which we can actually make statements using tools from data analysis usually has some underlying shape, and this shape conveys information: think about points sampled from a sphere or a torus. In both cases the point cloud will "look like" a 2-dimensional object, but the objects look different. In order to describe and compare different shapes, in particular in higher dimensions, we need some mathematical language. Luckily this language has already been developed under the name of *topology*. In topological data analysis we use the classical tools and language from topology to detect and describe the notion of "shape" in data sets. The main tool we use for this is *homology*, which can be regarded as "counting holes". Let us illustrate this with some toy examples.

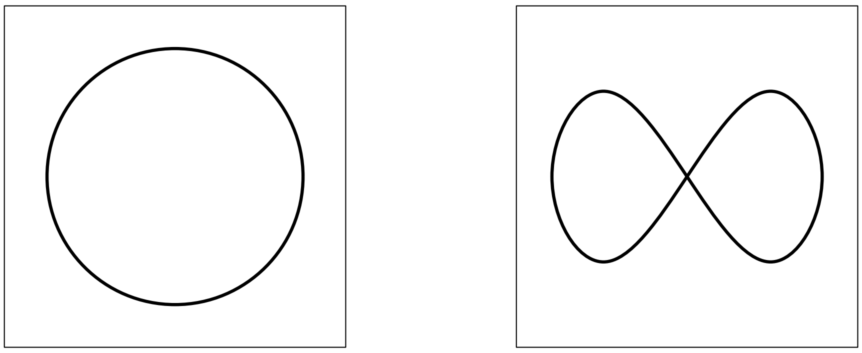

Figure 1.1: *A circle (left) and a Figure-8 (right)*

Consider the two shapes in Figure 1.1. On the left, you see a circle, on the right a Figure-8, both in $\mathbb{R}^2$. If you had to count how many "holes" these shapes have, you would probably argue along the following lines: the circle has one hole, as removing it from $\mathbb{R}^2$ we are left with one bounded connected component. Similarly, the Figure-8 has two holes. In particular, you would say that the two shapes are "different", as they have a different number of holes. This type of intuition is indeed correct, and topology provides us with the language to make this mathematically precise.

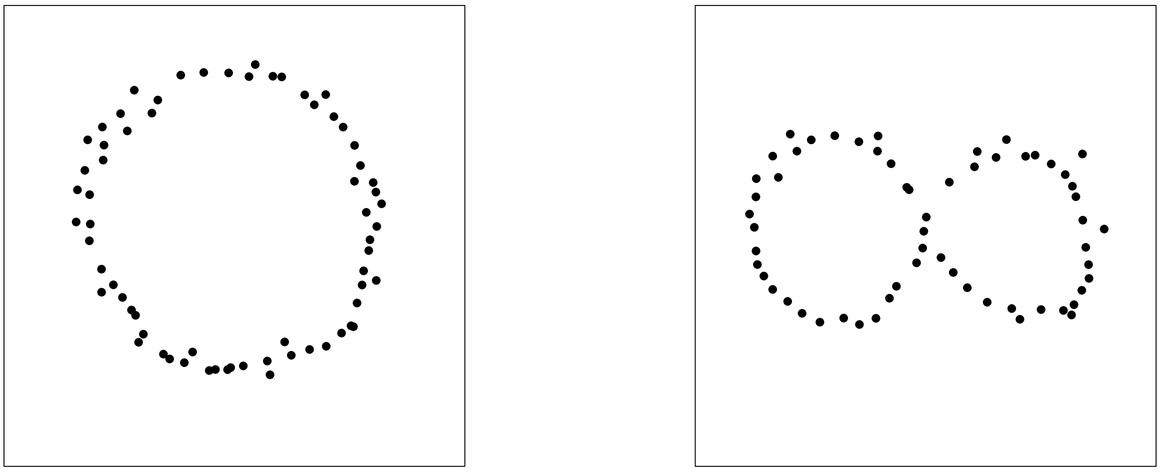

Figure 1.2: *Points sampled from a circle (left) and a Figure-8 (right).*

Let us now look at the 2-dimensional data sets in Figure 1.2. For a human, it is immediately visible that the data sets are (noisy) samples of the shapes of Figure 1.1. How do we get a computer to "see" this? Note that we cannot count the number of holes as we did above: as we are just given finitely many points, there are no holes in both data sets. However, by squinting our eyes a bit, we clearly see the different shapes from which we sampled. A mathematical analogue of squinting one's eyes could be to enlarge the points to disks with some small radius. This is depicted in Figure 1.3.

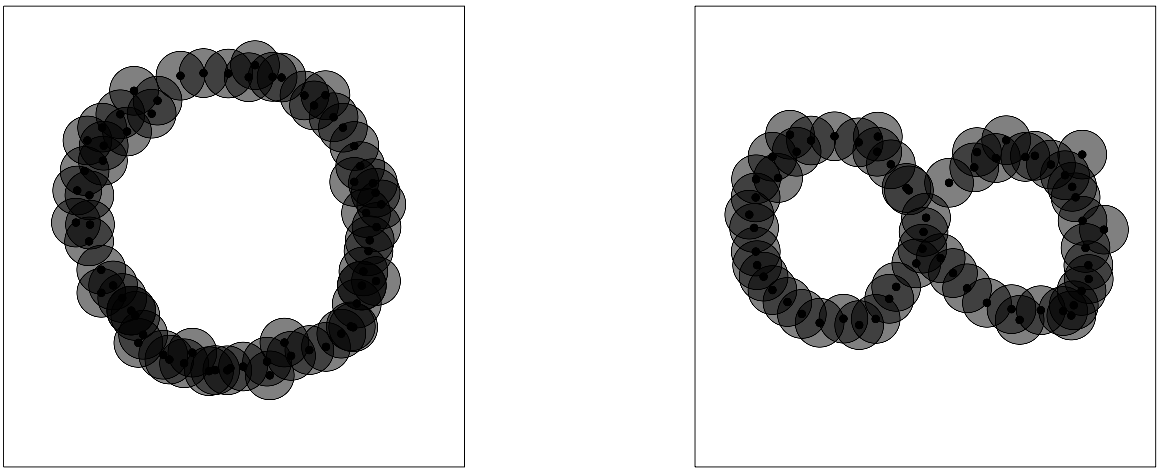

Figure 1.3: *Enlarging the points to disks shows the shape.*

Indeed, choosing a large enough radius, by considering the union of the disks we again get shapes that look like the two shapes we sampled from. In particular, on the left we again have one hole whereas on the right we have two holes. Unfortunately, it is not at all clear what radius we should choose. Indeed, by choosing a radius that is too small, we might not create all the holes of the original shape. On the other hand, by choosing a radius that is too large, we might fill in some of the holes[1]. This is depicted in Figure 1.4.

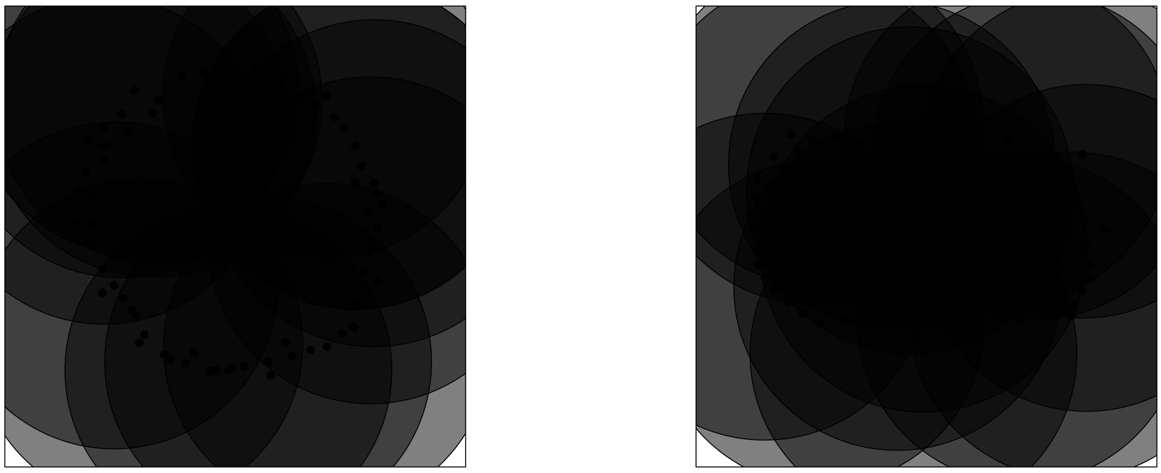
Figure 1.4: *Enlarging the disks too much, we lose the shapes again.*

Of course we could now try to somehow estimate a good radius. However, this is quite a difficult task in general. The key idea of *persistent homology* is to continuously grow the disks and keep track of the number of holes and how long they live. During this process of growing the disks, many holes will be created. However, many of them are filled in shortly after they are created, see Figure 1.5 for an example. In persistent homology, we keep track of all holes created in the process, together with timestamps of when they are created and when they are filled in. This gives a *lifetime* for each hole. The intuition behind this is that holes with a short lifetime are just a result of the process, whereas holes with a long lifetime convey information about the shape of the underlying data.

Let us consider yet another point cloud, this time sampled from two nested circles, see Figure 1.6. Two circles give us two holes, so in the process of growing disks we expect to see two holes with a long lifetime.

One way to visualize the lifetimes of holes is through *barcodes*: for each hole, we draw an interval whose startpoint and endpoint correspond to the time of creation of the hole, and when it got filled in. Doing this for the point cloud in Figure 1.6, we get the barcodes depicted in Figure 1.7. Indeed, while there are many intervals, only two of them are long, implying that there are two holes inherent to the underlying data, which agrees with the fact that the points were sampled from two circles.

[1]The same is true for squinting your eyes: if you squint them too much, you close them and do not see anything...

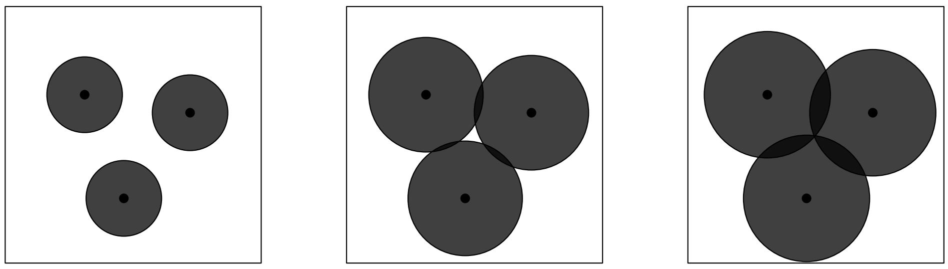

Figure 1.5: *In the growing disks process, many holes get filled in shortly after they are created.*

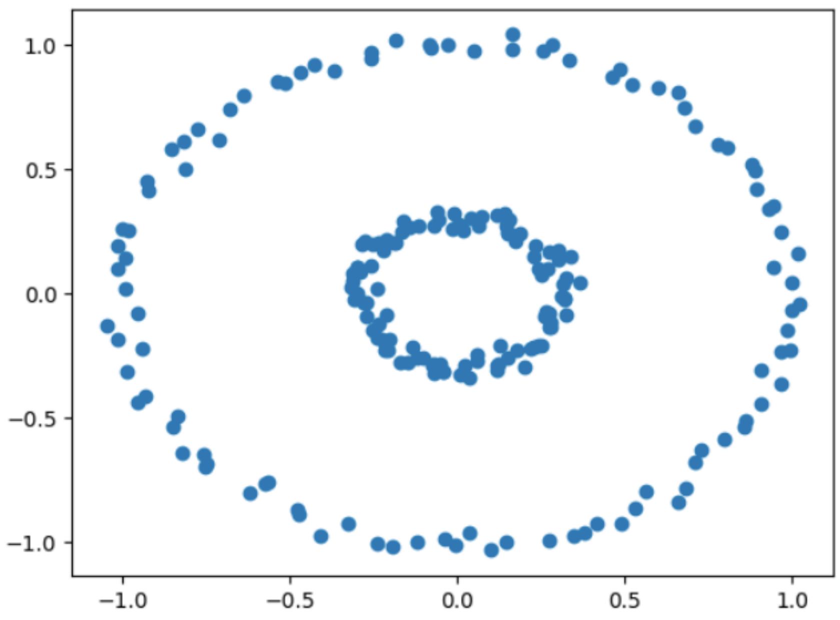

Figure 1.6: *Points sampled from two nested circles.*

The main work we will do in these lecture notes is to formalize this process of growing disks and keeping track of holes, as we sketched above. For this we first introduce some essential background of homology theory in Chapter 3. This requires some mathematical background that goes above linear algebra. This background will be introduced in Chapter 2. The growing disks process is modeled via nested *simplicial complexes*, and there are several different such complexes that can be defined. Some of these are discussed in Chapter 5. Keeping track of holes created and filled in is done via the theory of *persistent homology*, which we introduce in Chapter 4. In this chapter we also discuss algorithms to compute persistent homology. Once we have computed persistent

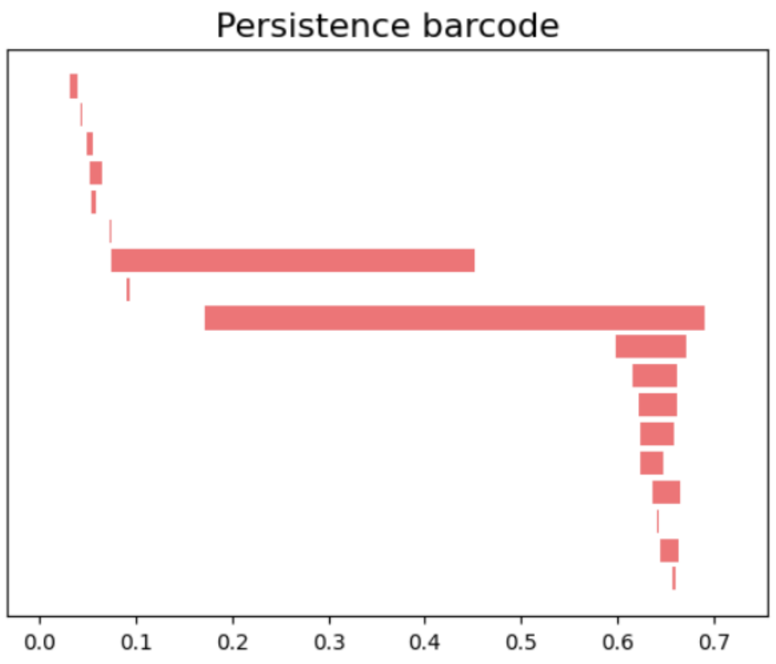

Figure 1.7: *The barcodes from two nested circles.*

homology and have the output, for example in the form of barcodes, we might want to compare different such outputs with each other. For this, there are several distance measures, which we discuss in Chapter 6. There we also mathematically prove stability results stating that if the data is perturbed only a little, then also the output cannot change too much.

Persistent homology is not the only application of topology to the analysis of data. Another widely used tool in topological data analysis is *Mapper*, which we discuss in Chapter 7. Finally, we discuss the computational problem of finding nice representatives of holes in Chapter 8 as well as persistence over multiple parameters in Chapter 9.

These lecture notes focus a lot on the mathematical theory behind topological data analysis. Topological data analysis has however seen many successful applications in recent years. We highlight some of them in Chapter 10. There are also powerful programming libraries available that implement many of the concepts discussed in these lecture notes. For the coding aspects of the topics in these lecture notes there are interactive notebooks on the course webpage.