

Persistent Homology References

My project report aims to give an accessible summary of persistent homology, it is available at <http://www.maths.ed.ac.uk/~aar/JaanErikPastreport.pdf>. For another introductory paper, see Gunnar Carlsson's Topological Pattern Recognition for Point Cloud Data.

For the algebraic topology background, the most popular reference text is Allen Hatcher's Algebraic Topology, partially because it is available for free online. I personally found a Youtube lecture series by *njwildberger* also called Algebraic Topology to be a great introduction.

The other works referred to in this presentation are:

- Paul Bendich, J. S. Marron, Ezra Miller, Alex Pieloch, and Sean Skwerer. Persistent homology analysis of brain artery trees
- V. Kurlin. Auto-completion of contours in sketches, maps and sparse 2d images based on topological persistence
- Gunnar Carlsson and Vin de Silva. On the local behavior of spaces of natural images.

Jaan-Eerik Past

Applications of Persistent Homology for Image Analysis

Jaan-Eerik Past

Supervised by Professor Andrew Ranicki

Summer 2015

Abstract

Topological data analysis is a field that balances the rigour of its mathematical background with the more experimental nature of statistics. Motivated by this, the paper has two aims:

- To describe the *theory* of persistent homology mathematically, highlighting the results that guarantee the methods we use for building topological structures are sound and that the analysis we perform produces meaningful information.
- To describe the *practice* of persistent homology, highlighting why the sound topological structures and true invariants are sometimes neither useful nor tractable. To illustrate this part, we use the *Javaplex* software package together with our own code to analyse images.

Acknowledgements

In addition to Professor Ranicki, I would like to thank his two PhD students Chris Palmer and Carmen Rovi for help and guidance and Professor Ramamoorthy for motivation and inspiration.

Furthermore, I would like to thank my fellow participants at the Young Topologists' Meeting at EPFL and at the LMS-CMI summer school at Oxford for many interesting and inspiring conversations, whether they were about topology or not.

Finally, this project would not have been possible without the support of the London Mathematical Society summer research scholarship.

Contents

1	Mathematics Background	3
1.1	Homology	4
1.2	Witness Complexes	13
1.3	Persistence	14
1.4	Point Clouds with Coefficients	19
2	The Computation of Persistence	21
2.1	Witness Complex Construction	21
2.2	Persistence	23
3	Image Analysis	25
3.1	Image Preprocessing	25
3.2	Discussion of Previous Results	28
3.3	Results	30
3.4	Summary	35
4	Further Developments	36
4.1	Zig-zag Persistence	36
4.2	Multidimensional Persistence	37
4.3	Computation	38
4.4	Statistics	39
4.5	Summary	40
A	Matrix calculations	41
B	Clarification of Carlsson's Results	43

Chapter 1

Mathematics Background

As a note of comfort to the reader, the application of persistent homology does not require a complete understanding of this section. The simple, intuitive explanation of persistence is enough to carry out topological data analysis, yet the understanding of its deeper formulations enables one to prove results that show its importance and to also consider possible extensions and generalisations. The reader less acquainted with algebraic topology and more interested in the computational and practical aspects of persistent homology may find it best to skip most of this section on a first read, perhaps with the exception of section 1.4.

This section does not aim to be comprehensive, but rather to expose the main results that are directly useful for us. This paper was largely inspired by an overview paper by Carlsson [1] (of which there is also a more recent version [2]). For a more thorough description, there are three wonderful books covering computational topology: one by Ghrist [3], one by Zomorodian [4] and one by Edelsbrunner and Harer [5].

Persistent homology aims to find the most relevant structures in data even if they are well-hidden. This is done in a three-step process as follows:

1. Put the data into correspondence with an algebraic structure
2. Give a classification of that structure
3. Find useful invariants from that classification

To be both usable and useful, we want invariants that have finite descriptions that are only the same for data spaces that are essentially the same (isomorphic) [6].

1.1 Homology

As our main objective is *computational* topology, we give definitions and overviews immediately usable on finite and discrete data. However, all of these concepts have rigorous general versions. The standard reference text for this is Hatcher’s Algebraic Topology [7], which is available for free online.

Topology is a study of spaces, where we are mainly interested in equivalences between spaces under continuous transformations. Simply put, we wish to know when objects can be stretched into one another without tearing. If this is the case, we call the spaces *homotopy equivalent*.

In the middle of the nineteenth century, the concept of the Euler characteristic used to describe polyhedrons paved the way for the theory of *homology*. The main feature of the theory is the description of spaces in terms of the holes present in them in various dimensions. This is very useful keeping in mind the idea of homotopy as holes in objects may stop us from being able to smoothly stretch them into other objects. Although this was later extended to a more general (manifold) setting, it is quite satisfying to know that the original description from more than 150 years ago is actually most the suitable for us.

In terms of the three-step process from before, we take our data points to be the vertices and on them we build a complex according to the structure of the data. Then, the algebraic classification will be given in terms of the homology groups of the complex. Persistence will enable us to construct the homology groups over a variety of *scales* in the data. Finally, this theory produces complete invariants in the form of persistence barcodes.

We will now begin to formally introduce all of these concepts.

Definition 1 *Two functions $f, g : X \rightarrow Y$ are homotopic if there exists a continuous map $F : X \times [0, 1] \rightarrow Y$ such that $f(x) = F(x, 0)$ and $g(x) = F(x, 1)$.*

This is just the rigorous definition of “smoothly stretching one function to another”. Note that being homotopic is an equivalence relation.

Definition 2 *Spaces X and Y are homotopy equivalent if there exist continuous maps $f : X \rightarrow Y$ and $g : Y \rightarrow X$ such that $f \circ g$ is homotopic to $\mathbb{1}_Y$ and $g \circ f$ is homotopic to $\mathbb{1}_X$.*

This is also just a rigorous way of saying “ X can be smoothly stretched into Y ”.

We could start to now look at homotopy classes of spaces and perhaps construct *homotopy groups* as our algebraic structures. When one does that, they do get a very powerful theory. However, homotopy groups, such as the *fundamental group*, are non-Abelian and thus we follow a slightly different approach to end up with a more convenient *homology* theory.

Definition 3 *An n -dimensional simplex Δ^n (an n -simplex) in $\mathbb{R}^n$ is a collection of $n+1$ linearly independent vertices and “the space between them”:*

$$\Delta^n = (t_1, \dots, t_{n+1}) \in \mathbb{R}^{n+1} : \sum_{i=1}^{n+1} t_i = 1, t_i \geq 0$$

such that

$$\sum_{i=1}^{n+1} \alpha_i t_i = \sum_{i=1}^{n+1} \beta_i t_i \Leftrightarrow \alpha_i = \beta_i \quad \forall i. \quad (1.1)$$

Note that it is important to keep track of the order of vertices in a simplex although this ordering can be chosen arbitrarily.

The lowest-dimensional simplices are familiar objects: the 0-simplex is a point, the 1-simplex a line segment, the 2-simplex a triangle and the 3-simplex a tetrahedron. For 2-dimensional surfaces, a subdivision of the surface into 2-simplices is, in fact, a *triangulation*.

Definition 4 *The face of an n -simplex is the $(n-1)$ -dimensional simplex that is left after removing one of the vertices of the original simplex.*

For tetrahedra, faces are what we would expect them to be. For other polyhedra, we would need to change the definition given above, but this one is the most convenient for calculations involving simplices.

Definition 5 *A simplicial complex X is a collection of simplices where, for all simplices $\Delta_1, \Delta_2 \subseteq X$, $\Delta_1 \cap \Delta_2$ is either $\emptyset$ or a face of both Δ_1 and Δ_2 and also $\Delta_1 \subseteq \Delta_2 \subseteq X$ implies $\Delta_1 \subseteq X$.*

This means that any two simplices can only touch at a vertex, edge, face or higher dimensional analogue and any higher-dimensional simplex has all of its lower-dimensional faces in X .

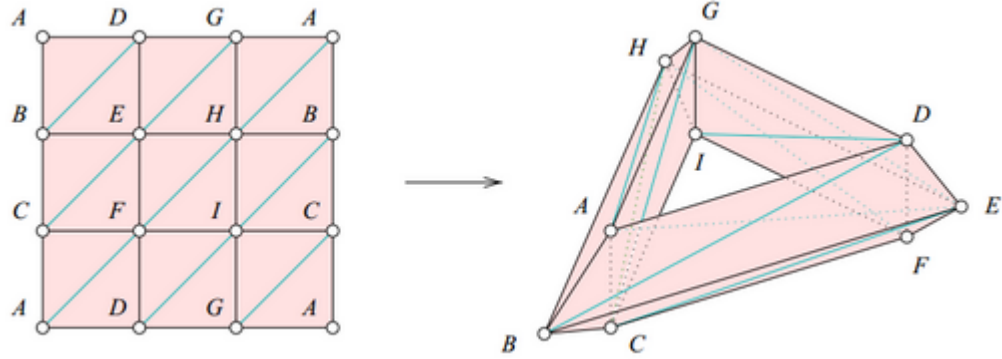


Figure 1.1: *The simplicial complex corresponding to a torus and the corresponding triangulation. Taken from the lecture notes of Herbert Edelsbrunner for Computational Topology at Duke University in the fall of 2006.*

While we think of simplicial complexes in terms of their geometric realisations in Euclidean space, they are in fact completely described by the list of simplices, the *abstract simplicial complex*, without any reference to a metric space. Indeed, a triangle is homotopy equivalent to a circle and a tetrahedron to a ball, so it does not make sense to be too concerned about the actual geometric representations of simplices.

Definition 6 An n -chain is a formal sum of n -simplices: $c_n = \sum_i \alpha_i \Delta_i$ where Δ_i are n -simplices and $\alpha_i \in \mathbb{Z}$. We create the chain group C_n by giving the set of n -chains the operation of component-wise addition: $\sum_i \alpha_i \Delta_i + \sum_i \beta_i \Delta_i = \sum_i (\alpha_i + \beta_i) \Delta_i$.

To relate the chain groups C_n and C_{n-1} , we define a homomorphism that acts on an n -simplex and gives its boundary.

Definition 7 The boundary homomorphism is a linear map $\partial_n : C_n \rightarrow C_{n-1}$ defined acting on an n -simplex by:

$$\partial_n \Delta = \partial_n [t_1, \dots, t_{n+1}] = \sum_{i=1}^{n+1} (-1)^i [t_1, \dots, \hat{t}_i, \dots, t_{n+1}], \quad (1.2)$$

where $\hat{t}_i$ means that we remove the vertex t_i from the simplex.

Recall that doing so gives us a face of the original simplex. The $(-1)^i$ is necessary to keep the orientations of the simplices correct.

We call a chains of simplices that start and end at the same point *cycles*. This means cycles are in the kernel of the boundary map. We call anything in the image of a boundary map a *boundary*. A crucial property of this homomorphism is that the boundary of a boundary is zero: if Δ is an n -simplex, then $\partial_{n-1}\partial_n\Delta = 0$.

These boundary homomorphisms enable us to define a useful sequence of structures.

Definition 8 *A chain complex is a sequence of Abelian groups with connecting maps between them such that the composition of any two consecutive maps is zero:*

$$\dots \xrightarrow{f_n} A_n \xrightarrow{f_{n-1}} A_{n-1} \xrightarrow{f_{n-2}} \dots A_1 \xrightarrow{f_1} A_0 \xrightarrow{f_0} 0, \quad (1.3)$$

where $f_{n-1} \circ f_n = 0$ for all n .

The fact that consecutive boundary maps compose to give zero means $im(\partial_{n+1})$ is a subgroup of $ker(\partial_n)$. Combined with the fact that the chain groups are Abelian (so all of their subgroups are normal), we can make the following definition:

Definition 9 *The n th homology group is the quotient group*

$$H_n = ker(\partial_n)/im(\partial_{n+1}). \quad (1.4)$$

The homology groups give information about the cycles disregarding the boundaries. Thinking about it, this means that we are counting loops, ignoring ones that are homotopic to each other. Note homology groups have a maximum dimension that is the same as the dimension of the data, after which all the homology groups are 0.

While simplicial structures are quite intuitive and easy to use for computation, when one tries to do examples by hand, it becomes clear that simplicial complexes for even simple structures can be too large to be conveniently used. For this reason, we also define a different structure that still allows for homology computation that uses *cells*. The theory is called *singular homology* and the definition of a cell, is just a continuous mapping from

the standard n -dimensional simplex to the topological space we are studying. While this definition may seem vague and very general, it can be proved that singular homology is equivalent to simplicial homology, in the sense that any cell complex can be further subdivided into a simplicial complex [7].



Figure 1.2: *The singular complex corresponding to a torus. Taken from Hatcher's Algebraic Topology [7].*

Example We now demonstrate the calculation of the homology of a simple singular complex which will be of use to us later.

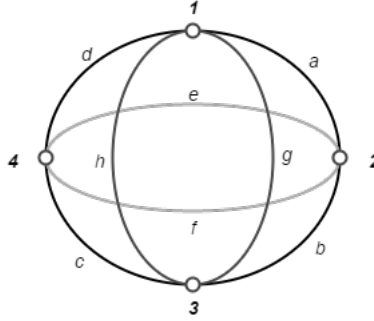


Figure 1.3: *The three-circle model, take all of the orientations clockwise*

The corresponding chain complex is $0 \xrightarrow{0} C_1 \xrightarrow{\partial_1} C_0 \xrightarrow{0} 0$, where $C_0 = \langle 1, 2, 3, 4 \rangle$ and $C_1 = \langle a, b, c, d, e, f, g, h \rangle$. From this, we get the homology groups:

$$H_0 = \ker(0)/\text{im}(\partial_1) = C_0/\langle 2 - 1, 3 - 2, 4 - 3, 4 - 1, 4 - 2, 3 - 1 \rangle, \quad (1.5)$$

$$H_1 = \ker(\partial_1)/\text{im}(0) = \langle a + b + c + d, e + f, g + h, c + d + g, e + b + c \rangle.$$

For H_1 , one could find more cycles, but these one can check that these five are the maximal generating set. For H_0 , we see that $(4 - 1) - (2 - 1) = 4 - 2$,

$(2 - 1) + (3 - 2) = 3 - 1$ and $(4 - 1) - (2 - 1) - (3 - 2) = 4 - 3$ so $\text{im}(\partial_1)$ has a maximal generating set of size three. Thus, we get $\dim(H_0) = 1$ and $\dim(H_1) = 5$.

Suppose we have a linear map between simplicial complexes $f : X_1 \rightarrow X_2$ that maps simplices to simplices of lower or equal dimension. Notice this means we get an induced map between the chain complexes $f_\# : C_n(X_1) \rightarrow C_n(X_2)$. One can also check that this map commutes with the boundary operators $\partial_n^{X_1}$ and $\partial_n^{X_2}$, that is $\partial_n^{X_1} \circ f_\# = f_\# \circ \partial_n^{X_2}$. This means $f_\#$ maps cycles to cycles and boundaries to boundaries, so there is an induced map between the homology groups as well: $f_* : H_n(X_1) \rightarrow H_n(X_2)$.

These induced maps have a structure-preserving property called *functoriality*. This means that $(\mathbb{1}_X)_* = \mathbb{1}_{H_n(X)}$ and $(f \circ g)_* = f_* \circ g_*$.

An extremely useful corollary of this is that homotopy equivalent maps between simplicial complexes induce the same map between homology groups (for more, see chapter 2 of [7]). For practical purposes, we often use the following, less general result:

Theorem 1 *If the map $f : X_1 \rightarrow X_2$ is a homotopy equivalence, then induced map $f_* : H_n(X_1) \rightarrow H_n(X_2)$ is an isomorphism for all $n \in \mathbb{Z}$.*

This means that we know the homology of a whole homotopy equivalence class by just knowing the homology of one of its members. Furthermore, if the homology groups of two spaces have no isomorphisms between them in any given dimension, we see that the spaces can not be homotopy equivalent.

Homology with Coefficients

For simplicial homology, we defined a chain complex by having each group C_n be a finite formal sum $\sum_i \alpha_i \Delta_i$ where $\alpha_i \in \mathbb{Z}$ and Δ_i are simplices. This readily generalises to the case where the coefficients $\alpha_i \in G$ for some Abelian group G . It is quite straight-forward to check that all of the notions of homology carry over to *homology with coefficients*. We denote the n^{th} homology group of space X with coefficients in group G by $H_n(X; G)$.

The usefulness of this definition is that homology with coefficients can cause some cycles or boundaries not to be noticed when compared to homology with integer coefficients. For example, when we do not care about the orientations of our simplices, homology with $\mathbb{Z}_2$ coefficients is useful as

$-1 = 1 \pmod{2}$. This clearly reduces computation times as seen in chapter 12 of [8] by making the homology groups just lists of simplices with no coefficients with each simplex being its own inverse for calculations.

Example The Klein bottle is a surface that can be defined through identifying the edges of a square to each other. While for the torus, opposite edges are identified with the same orientation, the Klein bottle has opposite orientations for two of the edges.

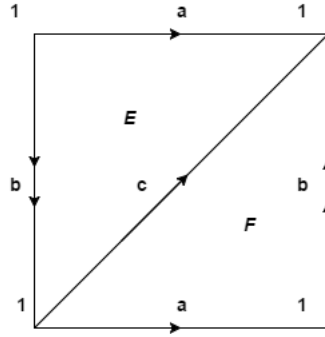


Figure 1.4: A representation of the Klein bottle by identifying the edges of a square.

To enable us to compute the *simplicial* homology¹, we must subdivide the square until we satisfy the intersection conditions of simplices. The most natural approach is to triangulate the Klein bottle using 9 vertices, 27 edges and 18 triangles. However, in the singular picture, we just have one vertex, three edges and two faces.

This gives us the following chain complex:

$$0 \xrightarrow{0} C_2 \xrightarrow{\partial_2} C_1 \xrightarrow{\partial_1} C_0 \xrightarrow{\partial_0} 0. \quad (1.6)$$

For the singular complex, we would have the following bases for the chain groups: $\langle 1 \rangle_{C_0}$, $\langle a, b, c \rangle_{C_1}$ and $\langle E, F \rangle_{C_2}$.

Carrying out the calculations of the images and kernels of the boundary maps, we get the following homology chain:

$$0 \rightarrow \langle E + F \rangle / 0 \rightarrow \langle a, b, c \rangle / \langle a - c - b, a + b - c \rangle \rightarrow \langle 1 \rangle / 0 \rightarrow 0. \quad (1.7)$$

¹The Javaplex program has an example that computes the homology of the simplicial Klein bottle for different coefficients.

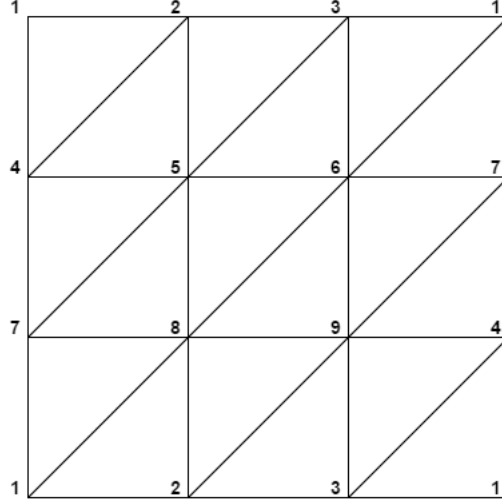


Figure 1.5: *The simplicial complex of a Klein bottle. Note all of the edges and faces are filled in. For simplicity, we have not included the orientations here, which we would need to do to calculate homology with $\mathbb{Z}$ coefficients.*

We need to do a bit of work on the middle term. We notice that $\langle a, b, c \rangle / \langle a - c - b, a + b - c \rangle = \langle a + b - c, b, c \rangle / \langle a - c - b, 2b - 2c \rangle$ Finally, this simplifies to $\langle b - c, c \rangle / \langle 2b - 2c \rangle$ Thus we get that the homology of the Klein bottle K is $H_0(K; \mathbb{Z}) = \mathbb{Z}$ and $H_1(K; \mathbb{Z}) = \mathbb{Z} \oplus \mathbb{Z}/2\mathbb{Z}$ with all other groups 0 for integral coefficients.

We can gain more insight into homology with coefficients by using the following theorem:

Theorem 2 *Universal Coefficient Theorem for Homology*

If C is a chain complex of free Abelian groups, then there are natural short exact sequences:

$$0 \rightarrow H_n(C) \otimes G \rightarrow H_n(C; G) \rightarrow \text{Tor}(H_{n-1}(C), G) \rightarrow 0$$

for all n and G [7]. Here, Tor is the torsion functor we will not be able to describe in detail.

A sequence is *exact* if $\ker(\partial_n) = \text{im}(\partial_{n+1})$ for all n , so that the corresponding quotient groups are trivial.

The Universal Coefficient Theorem will be useful for studying homology with coefficients over fields (specifically, over $\mathbb{Z}_p$). But first, we need another theorem:

Theorem 3 *The Fundamental Theorem/The Classification Theorem of Finitely Generated Abelian Groups*

Every finitely-generated Abelian group G is isomorphic to a unique direct sum of groups

$$G \simeq \mathbb{Z}^\beta \oplus \mathbb{Z}_{p_1} \oplus \dots \oplus \mathbb{Z}_{p_s}$$

where β is an integer and p_i are powers of primes [4]².

In the context of homology groups, p_i are called the *torsion coefficients* and β the *Betti number*. Note that although the Betti number is usually defined as the number of $\mathbb{Z}$ summands, the properties of the tensor product mean that for other coefficient groups G , the Betti number will be the number of G summands. Furthermore, we are allowed to use the Universal Coefficient Theorem as the homology groups are finitely generated for finite chain groups.

Since any field F is freely generated, we know $Tor(\mathbb{Z}, F) = 0$. Thus, we have $H_n(C) \otimes F \simeq H_n(C; F)$. But $F \otimes \mathbb{Z}_n = 0$ and $F \otimes \mathbb{Z} = F$ as well, which means that homology groups with field coefficients have no torsion and they have the same Betti number as homology groups with integer coefficients.

In the case of $\mathbb{Z}_2$, we get

$$H_n(C; \mathbb{Z}_2) \simeq H_n(C) \otimes \mathbb{Z}_2 \simeq \mathbb{Z}_2^\beta \oplus \mathbb{Z}_{gcd(2, p_1)} \oplus \dots \oplus \mathbb{Z}_{gcd(2, p_s)} \simeq \mathbb{Z}_2^{\beta + p_2} \quad (1.8)$$

where β is the Betti number and p_2 the number of $\mathbb{Z}_2$ summands in the decomposition of G .

Example: Looking back at our Klein bottle example, we get the homology with $\mathbb{Z}_2$ coefficients by using this theorem as $H_0(K) = \mathbb{Z}_2$, $H_1(K) = \mathbb{Z}_2 \oplus \mathbb{Z}_2$ and $H_2(K) = \mathbb{Z}_2$, with higher homology groups 0. One could also check that this is true by doing the calculations as for $\mathbb{Z}$ coefficients (in fact, it is easier due to ignoring orientations).

In *Javaplex*, we use $\mathbb{Q}$ and $\mathbb{Z}_p$ coefficients³ (note p is required to be prime to make $\mathbb{Z}_p$ a field), so we are interested in when homologies and barcodes with these two differ from each other. From the previous paragraph, it is clear that the Betti numbers with $\mathbb{Q}$ and $\mathbb{Z}$ coefficients will agree, whereas the $\mathbb{Z}_p$ can give different results when $H_n(C)$ has a torsion component, as we shall see later.

²One needs to be very careful in reading the results after the theorem 4.6 in Zomorodian, as some of the results are written in a somewhat confusing manner.

³The fact that $\mathbb{Q}$ is not finitely generated might raise some alarm for using the Fundamental Theorem, but then we note that we are using the theorem not on $\mathbb{Q}$ itself, but on the homology groups, which are finitely generated.

1.2 Witness Complexes

The Čech theorem (also known as the Nerve theorem) states that a union of balls around each point of a space (the *Čech complex*) has the same homotopy type as the underlying space. This result was first proved by Borsuk in 1948 [9] and the corresponding result for persistent homology was published by Bendich et al. in 2007 [10]. Unfortunately, this approach not only requires fairly stringent conditions to be fulfilled, but the Čech construction tends to be computationally very expensive. However, experimental approaches have shown that less rigorous constructions also give the correct homology with high confidence. Another aspect to note is that our approach is to construct a manifold structure *from* our data and then use that to infer properties of our data, so whether the nerve theorem would be useful for us is up to debate.

For these reasons, we choose to use the more efficient landmark complex construction, specifically, the *witness complex*. To find it, we select some points from our data to be *landmarks*, which we take to be our zero-simplices (vertices). We define a data point to be a witness to two landmark points based on some criterion. Two landmark points will then form a 1-simplex if and only if they have a common witness.

Definition 10 *We will be using the weak witness complex on a vertex set D , denoted $W(D)$, where there is a 1-simplex between two landmarks a and b if there exists a witness such that both landmarks are within ϵ of the closest landmark point to that witness:*

$$[ab] \in W(D) \Leftrightarrow \exists i \in D : \max(d(a, i), d(b, i)) \leq \epsilon + \delta, \quad (1.9)$$

where δ is the distance to the closest landmark.

This construction is quite similar to a Delaunay triangulation. We could also omit the δ term completely (in which case we would effectively be constructing the Vietoris-Rips complex with radius ϵ on the landmarks), or take it to be the distance to the second closest landmark, which gives the construction some robustness and reduces noise in practice [11],[12]. Also note that we can use any metric $d(\cdot, \cdot)$ to measure the distance to the landmarks.

The *lazy* witness complex is constructed by creating the largest complex that has the 0 and 1-simplices of a weak witness complex, analogously to a

Vietoris-Rips construction. This means we connect all the lower-dimensional edges into higher-dimensional faces wherever possible.

It is noteworthy that this sort of fairly crude approximation is often sufficient for gleaning the topological features of the data as we do need to reconstruct the actual *shape* of the data for that.

1.3 Persistence

Thinking in terms of the previous section, the idea of persistence is to construct complexes not only for a single radius parameter but for a wide range. This gives us a filtration of our complex, starting off with only the vertices for a small radius with higher simplices being added with increasing radius.

Then, we find the homology groups (and Betti numbers) for each of these filtrations. The resulting Betti numbers are usually given as barcodes, which represent how the homology changes through the filtration. The beginning of a barcode can be thought of as a *birth time* and the end of one as the *death time*. The significance of a homological feature is given by the length of the corresponding barcode.

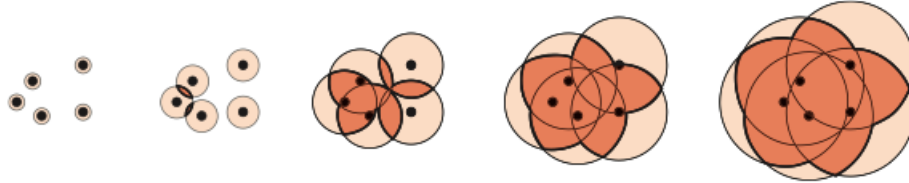


Figure 1.6: *The intersections of the balls around five vertices for various radii. If we insert an edge for every intersection, a face for every triple intersection and so on, we see we get different simplicial complexes and hence homology groups. Figure taken from Don Sheehy's (Multi)Filtering Noise in Geometric Persistent Homology [13].*

In the case of witness complexes, one can see that as ϵ increases, we construct more simplices on the same set of vertices, which means we have nested simplicial complexes for increasing values of ϵ and corresponding inclusion maps between complexes. Computationally, this is quite convenient, as for any value of ϵ , vertices σ_i with $i \in \{1 \dots p, p + 1\}$ form a p -simplex if and

only if $\max(d(\sigma_i, \sigma_j)) \leq \epsilon$, so the distances between the vertices completely describe the lazy weak witness complex.

A good theoretical overview of persistence can be found in chapter 2 of [14]. While persistence is perhaps most elegantly described in the language of category theory [15], for our needs, a more basic explanation will suffice. In this section, we prove an important theorem that gives persistent homology a strong theoretical background. For more details, see [16]. We start by giving some definitions.

Definition 11 *A principal ideal domain (PID) is a commutative ring with identity and no zero-divisors where all the ideals are principal*

One can think of a PID as a ring where we can calculate the greatest common divisor of any two elements. Thus, $\mathbb{Z}_n$ is a PID if and only if n is a prime as otherwise, it would have zero-divisors. As another example, consider $\mathbb{Z}[x, y]$, the ring of polynomials in two variables with integer coefficients. Here, the ideal (x, y) obviously can not be generated by one element, so it is not principal.

We can also add additional structure to our ring:

Definition 12 *A graded ring R is a ring that can be decomposed into a direct sum of Abelian groups $R = \bigoplus_i R_i$ such that multiplication is given by a bilinear pairing $R_i \otimes R_j \rightarrow R_{i+j}$.*

Of particular interest to us are the gradings of polynomial rings $R[t]$, such as the standard grading $R[t] = \bigoplus_n r_n t^n$ with $r_n \in R$. This means that multiplication works as expected: $t^n \cdot t^m = t^{n+m}$.

For PIDs, we have an important classification theorem:

Theorem 4 *Structure Theorem:* *A finitely generated PID R decomposes uniquely into a direct sum $M = R^\beta \bigoplus_{i=1}^n R/(r_i R)$, where $r_i \in R$ and $\beta \in \mathbb{Z}$.*

Note that the *free* part R^β may contain an infinite number of elements and gives a vector space. On the other hand the $R/(r_i R)$ are finite quotient rings, called the *torsional* portion.

To get towards our results about persistence, note we can view Abelian groups as modules over the integers. As the homology groups are Abelian groups, this will shortly become extremely important.

Consider now a simplicial complex C with a filtration given by inclusion maps

$$\emptyset \rightarrow C^0 \rightarrow C^1 \rightarrow \dots \rightarrow C^{n-1} \rightarrow C^n = C. \quad (1.10)$$

For any member C^i of this complex, we have the chain complex using boundary maps:

$$C^i = C_n^i \rightarrow C_{n-1}^i \rightarrow \dots \rightarrow C_1^i \rightarrow C_0^i \rightarrow \emptyset. \quad (1.11)$$

We call this two-dimensional sequence of chain complexes the *persistence complex*.

$$\begin{array}{ccccccc} \partial_3 \downarrow & & \partial_3 \downarrow & & \partial_3 \downarrow & & \\ \mathbf{C}_2^0 & \xrightarrow{f^0} & \mathbf{C}_2^1 & \xrightarrow{f^1} & \mathbf{C}_2^2 & \xrightarrow{f^2} & \dots \\ \partial_2 \downarrow & & \partial_2 \downarrow & & \partial_2 \downarrow & & \\ \mathbf{C}_1^0 & \xrightarrow{f^0} & \mathbf{C}_1^1 & \xrightarrow{f^1} & \mathbf{C}_1^2 & \xrightarrow{f^2} & \dots \\ \partial_1 \downarrow & & \partial_1 \downarrow & & \partial_1 \downarrow & & \\ \mathbf{C}_0^0 & \xrightarrow{f^0} & \mathbf{C}_0^1 & \xrightarrow{f^1} & \mathbf{C}_0^2 & \xrightarrow{f^2} & \dots \end{array}$$

Figure 1.7: A part of a persistence complex. Figure taken from Zomorodian's *Topology for Computing* [16].

Now, looking at the homology of this chain complex in the language of modules, we have induced *module homomorphisms* $\phi_k^i : H_k^i \rightarrow H_k^{i+1}$. The boundary maps between these give us the *persistence modules*.

To fully unleash the machinery developed here, note that the definitions and the structure theorem from before readily generalise to modules.

Definition 13 A graded module M over a graded ring R is a ring that can be decomposed into a direct sum $M = \bigoplus_i M_i$ such that the action of R on M (multiplication) is given by a bilinear pairing $R_i \otimes M_j \rightarrow M_{i+j}$.

Of particular interest to us are the gradings of polynomial rings $R[t]$, such as the standard grading $R[t] = \bigoplus_n r_n t^n$ with $r_n \in R$.

We can give the k^{th} persistence module a graded structure over the polynomial ring $R[\phi]$: $H_k = \bigoplus_{i=0}^{\infty} H_k^i$ where (in a slight abuse of notation), the grading is given by the module homomorphism acting on the homology modules. By this we mean

$$\phi \cdot \left(\sum_{i=0}^{\infty} h_k^i \right) = \left(\sum_{i=0}^{\infty} \phi_k^i h_k^i \right) = \sum_{i=0}^{\infty} h_k^{i+1}, \quad \forall h_k^i \in H_k^i. \quad (1.12)$$

That is, the polynomial structure simply shifts between the gradings.

This means that to look at the persistent homology over the entire filtration, we form the persistence module, which is a direct sum of homology modules, which themselves are vector spaces over fields.

Theorem 5 *Structure Theorem:* *A finitely generated module M over a principal ideal domain R decomposes uniquely into a direct sum*

$$M = \left(\bigoplus_{i=1}^n \phi^{\alpha_i} R \right) \oplus \left(\bigoplus_{j=1}^m \phi^{\beta_j} R / r_j R \right), \text{ where } m, n, \alpha_i, \beta_j \in \mathbb{Z}^+ \text{ and } r_j \in R.$$

Outline of proof: We apply the structure theorem from before, as any field is a principal ideal domain. This means we decompose H_k into the free part and the torsional part, just as we wish.

Looking back, we note that structure theorem for Abelian groups is a special case of this structure theorem as Abelian groups can be treated as modules over the integers.

Also note that this theorem only applies to modules over PIDs. The classification of modules over non-PID rings, for example $\mathbb{Z}[t]$, is known to generally be a difficult problem.

However, for polynomial rings of fields, the only ideals are homogeneous of the form t^j , so our structure result simplifies even further:

$$M = \left(\bigoplus_{i=1}^n \phi^{\alpha_i} \mathbb{F}[t] \right) \oplus \left(\bigoplus_{j=1}^m \phi^{\beta_j} \mathbb{F}[t] / t^{\gamma_j} \right), \quad (1.13)$$

where $m, n, \alpha_i, \beta_j, \gamma_j \in \mathbb{Z}^+$.

This gets us to our final and most important result:

Theorem 6 *There is a bijection between half-bounded intervals and finitely generated modules over a graded polynomial rings of fields.*

The half-bounded intervals can be taken to be the *persistence barcodes* and the graded modules to be our *homology groups*. The existence of the bijection means the barcodes fully classify the persistent homology groups!

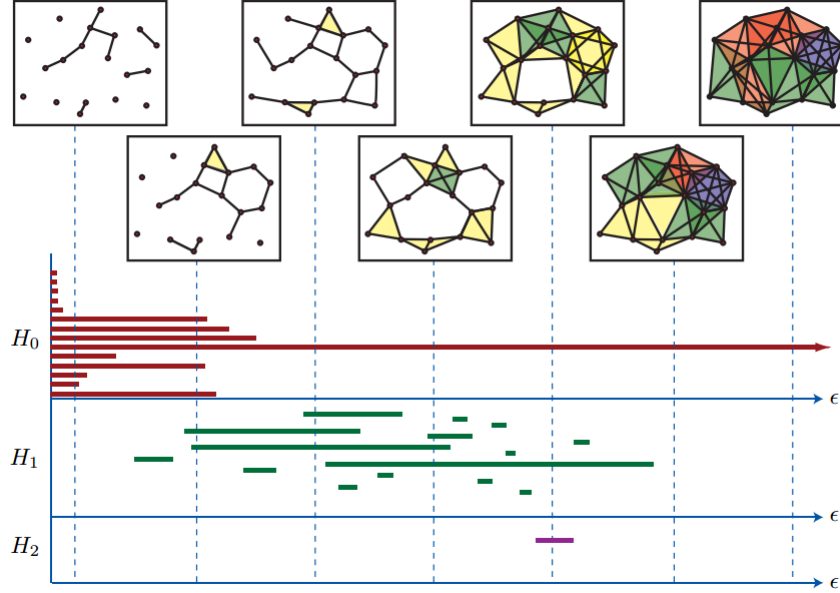


Figure 1.8: An example of a point cloud, some steps in the construction of its persistent homology and the corresponding barcodes. Figure taken from Robert Ghrist's *Barcodes: The Persistent Homology of Data* [17].

Practice

To connect the previous description with the intuitive explanation of persistence from the beginning of this section, notice that any $f_i t^n$ in the above decomposition denotes the birth of a homology class. This means that the free component only includes features which are born at some time and then persist to infinity, whereas features in the torsional component die when they are factored out by the t^{γ_j} .

The question of what represents a *long* barcode is very important, because the results of persistent homology are oftentimes difficult to read, as we shall see in our examples. However, there is no full theoretical answer to that question and it is far from clear that such an answer should exist. Not only

does the clarity of the results we wish to achieve depend directly on the type of data we are working with, but also on the scale or resolution we wish to probe.

1.4 Point Clouds with Coefficients

From our previous discussion, we see that for homology with $\mathbb{Z}_p$ and $\mathbb{Q}$ coefficients to differ, the underlying space needs to have p -torsion. For surfaces, non-zero $\mathbb{Z}_2$ torsion in H_1 means that the surface is nonorientable.

We wrote an additional class in *Javaplex* to simulate a random point cloud on the surface of a Klein bottle⁴ in $\mathbb{R}^4$. Using this, we do get the correct homologies for $\mathbb{Q}$ and $\mathbb{Z}_2$ coefficients, but to reach a clear result, we need to represent the Klein bottle by over 50,000 simplices.⁵ We suspect that this is due to the nature of the witness complex - when we reduce our data set down to a few hundred or even fewer landmarks, we are bound to miss some of the fairly subtle topology that goes into producing torsion. When we attempted using Vietoris-Rips complexes, we failed to capture any interesting features and then resulting barcodes were the same for either coefficient field.

As a heuristic argument, one can consider the fact that with too few points being used to construct higher simplices, we are very likely to miss any *twists* in our data as they tend to just get connected into a slightly thinner or thicker surface with no twisting visible. The landmarks will not be homotopy equivalent to the underlying space, for example, see section 2.3 in [1]. This seems to be in line with the arguments made by Carlsson et al. in section 2.4 of [19] for a lack of torsion in 'real' data sets.

In any case, we note that it is important to pick the correct number of landmark points as well as experimentally determining a good maximal radius in order to keep runtimes manageable. For example, sampling 6000 points we get the following results:

⁴The question how to generate random points on a manifold is actually rather complicated and often ignored. While we will not go into that discussion here, we direct the interested reader to [18]. A rectifying feature for not accounting for this issue is that we use *max-min* landmark selection which naturally picks out points further from each other.

⁵Compare this to the 500 points and 12 witnesses necessary to get the homology of a sphere in [12].

Number of landmarks	300	200	150	100
Number of simplices	53,982	24,823	16,087	8426
Runtime (<i>ms</i>)	99,754	30,931	14,227	4632

Table 1.1: *Results for witness complex construction using the max-min method for a sampling of 6000 points from a Klein bottle without noise.*

Chapter 2

The Computation of Persistence

In terms of historical development, homology was actually developed in a very computation-friendly manner in the early 1900s in terms of Betti numbers and torsion coefficients using just the boundary maps. It was not until the 1940s that a group-theoretical approach really brought singular homology to be the centrepiece for homology theory. In some senses, we will go back to make use of the earlier formulation.

There is now a plethora of computational software to carry out some or all of this process (see [20]), each of which has slightly different features. We used the *Javaplex* package due to it being able to carry out all of the steps of the pipeline and so we will describe some of the features unique to it, but we will also give the general algorithm used to compute persistence

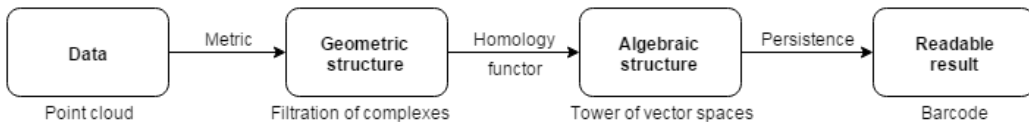


Figure 2.1: *The process of persistent homology calculations as a pipeline*

2.1 Witness Complex Construction

So far, *Javaplex* is the only package that allows for the use of witness complexes. These greatly increase the speed of computation, however, we note

that one needs to be somewhat careful about the parameter values they select for the construction. We give a brief overview here.

Landmark Selector

To initialise the construction, the complex needs a set of landmark points. The two inputs needed are the number of landmarks and some sort of point-cloud data. The number of landmarks is an arbitrary parameter, in [12], it is suggested that it should be no more than 1 in 20 data points. There is a similar experimental approach in [11]: “*In practice, we use persistence to select the number of landmarks: i.e., ... we build the complex for different ... values, calculate the homology, and choose a value at which the results stabilize.*”

The constructor then finds the selected number of landmarks based using either the *max-min* or the *random* method, the former maximises the distance to all of the existing landmarks for finding each landmark while the latter just picks random data points as landmarks. Note that an Euclidean metric space is used to give distances between the set of landmark points and data points.

Distances

To create the lazy witness *stream*, we specify a further four parameters.

As a result of the landmark selector, we get the maximum distance to a landmark point (*maxDistance*). The variable we can change is the maximum allowed distance between two connected vertices in any simplex (*max_filtration_value*). In the optical image script, this is set as *maxDistance*/4. The next directly connected variable is the number of divisions (*num_divisions*), which changes how many intervals our maximal filtration value is subdivided into i.e. how many different values of ϵ are used for homology calculations. As our aim is to find *long, persistent* features, this last variable was relatively unimportant.

Similarly, we can change the maximal dimension homology is calculated in (*max_dimension*). One viewpoint that could be taken is that the higher homology groups are similar to higher order terms in a Taylor series - while they may be useful, most information is included in the first few terms. For our image analysis purposes, although our preprocessed data input comes in 8 dimensions, our homology was only non-zero up to dimension 2.

The last variable is ν . It comes from us actually using a family of witness complexes $W(D; R, \nu)$, where we can choose between $\nu = 0, 1, 2$. When building a lazy witness complex, it's the distance to the ν^{th} closest witness that we measure our distances to in the description of witness complexes from before. We only use $\nu = 1$.

2.2 Persistence

We now provide a description of the general persistence algorithm used in Javaplex and other computational tools. A complete derivation is given in [16], but we give an outline here.

We start off with a filtration of our simplicial complex. The key point is that the filtration gives a total ordering of all the simplices in the complex, given by the birth time. For the ordering to be *compatible*, we require all the faces of a simplex to be born no later than the simplex itself.

The structure theorem from before shows that we should deal with field coefficients as homology may not be a full invariant for other rings. In Javaplex, the fields available are $\mathbb{Z}_p$, where p is an integer and $\mathbb{Q}$. We will describe the algorithm for $\mathbb{Z}_2$ coefficients, although it does generalise to other fields.

Algorithm

The homology of a complex is fully described by a boundary matrix D . The columns of D will be the simplices ordered by their birth time e.g. the radius at which they enter the complex for a witness construction. With $\mathbb{Z}_2$ coefficients, $D_{i,j} = 1$ if σ_i is a face of σ_j and $\dim(\sigma_i) = \dim(\sigma_j) - 1$ and $D_{i,j} = 0$ otherwise.

For D , we define $low(j)$ for column j to be the first row in which j has a 1 and we take $low(j) = 0$ if j consists of zeroes. The aim of our algorithm is to then transform D so that each row i only contains the $low(j)$ of at most one column j .

We do this by column addition: for each column j , for all $j' < j$ if $low(j) = low(j')$, we add j' to j . Since we are working over $\mathbb{Z}_2$, note this must *increase* $low(j)$ as the previous low entry will have 1 added to it and as such will become 0. Our algorithm finishes when each row only contains at most one $low(j)$. We can then read off the barcodes from the matrix, as

when column j is zero, it means the appearance of the corresponding simplex σ_j gives birth to a homological class of degree equal to $\dim(\sigma_j)$. Similarly, a *low* 1 in a row corresponds to the death of a homology class of degree equal to the dimension of the simplex corresponding to the row. As a worst case, this algorithm will take n^2 column operations, where n is the number of columns, which gives us $O(n^3)$ speed in terms of computing time.

There exist improved versions of this algorithm - for some complexes, column reductions are sometimes much faster than row ones, and there are developments such as the twist algorithm [21]. In general, the best results run at about the same speed as matrix multiplication.

Barcodes

One very important thing to note is that there is no particular significance in barcodes ‘going to infinity’. By ‘going to infinity’ we mean that the Betti number stays the same as the radius is increased past the cut-off radius R_{max} - it may well happen that there are longer (i.e. more significant) barcodes that happen to end before the R_{max} and shorter (i.e. less significant) barcodes that start just before the R_{max} and end just after it. To account for this, it is important to choose the correct value for R_{max} based on the data being worked with.

Also note that the plots of barcodes can be difficult to read visually, so it is sometimes more convenient to have the results displayed as the lifetime intervals of the barcodes in the form (t_{birth}, t_{death}) .

Chapter 3

Image Analysis

Although image analysis is a reasonably well-trodden area for persistent homology, it is very instructive in demonstrating the strengths, weaknesses and necessary considerations for using the theory. Here we undertake a replication of Carlsson’s original results for the local behaviour of natural images [22] and also further develop the analysis.

3.1 Image Preprocessing

Attempting to directly analyse the pixel distribution of an image is not the best approach as the amount of noise in the data makes it very difficult to get a clear picture of the underlying structure [12]. Fortunately, there are established procedures in image analysis that make our task more tenable. The steps we undertook¹ are essentially the same as described in Lee et al. [23]:

- Subdivide the given image into 3x3 pixel patches.
- *Centring*: Take logarithms of the greyscale pixel values (which are the *intensities*), find the mean intensity in each patch and subtract it from each pixel².
- *Normalisation*: find the contrast between neighbouring pixels for each patch (*d-norm*). The d-norm is found for each pixel in a patch by taking

¹The matrix calculation to achieve these steps are given in Appendix A.

²According to [23], taking the logarithm corresponds to how the human eye perceives intensities.

the square of the difference of intensities between any two adjacent pixels in a patch. Treating the 3x3 patch as a vector in $\mathbb{R}^9$, we normalise by dividing by the d-norm.

- *Contrast filtration*: Ordering the patches by d-norm (that is, local contrast in each patch), we keep the top 30% highest contrast patches (this is chosen as 20% in [23], but we keep the higher percentage). We do this to avoid our data being full of patches that are just a constant colour.
- *Density filtration*: In the space of 3x3 patches (lying in $\mathbb{R}^9$), find the Euclidean distances between all pairs of patches and then keep the ones in the lowest $p\%$ for distance to the k^{th} nearest neighbour.
- *Change of basis*: Now, from the normalisation step, our patches lie on the unit sphere in $\mathbb{R}^9$. Furthermore, we have also ignored any constant patches. Because of this, we transform to the convenient *DCT* basis described in Lee et al. [23], where each 3x3 patch is now described by a vector on the unit sphere in $\mathbb{R}^8$.

While the other steps are quite straight-forward and easy to justify, it is not immediately obvious why we can (or should) carry out the k^{th} nearest neighbour (*KNN*) density filtration. This comes in essence from the experimental observation that most of our data lie in certain high-density regions of space of patches. In this way, changing k changes the scale at which we are analysing our image - small k gives us the local behaviour of the patch space (we only look at near neighbours) whereas a large value would look at the patch space at larger scales (we look at further separated patches). Also, we note that in Carlsson’s paper, it is not clear which metric is used for *KNN*, but we used the standard Euclidean metric for $\mathbb{R}^9$ [22]. Another step that Carlsson undertakes but we did not find necessary is actually replacing each set of k nearest neighbours by their average point.

A point to notice is that if we did less preprocessing, we should not be using *max-min* landmark selection for constructing our witness complexes. The reason for this is that to maximise the distance between landmarks *max-min* is more likely to pick out extreme data points, which mostly represent noise if no density filtration is undertaken. In that case, using random landmark selectors is recommended, although they provide a less-ideal sampling from inside the more dense parts of data.

When we attempt complex construction for a 1440×900 image without any preprocessing, we run out memory, with even 1536 MB not being enough. We were able to produce results for a 188×251 image. All of the following were calculated with $k = 20, p = 30$.

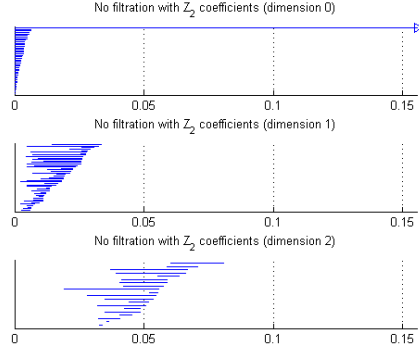


Figure 3.1: *Barcode for random landmark selection.*

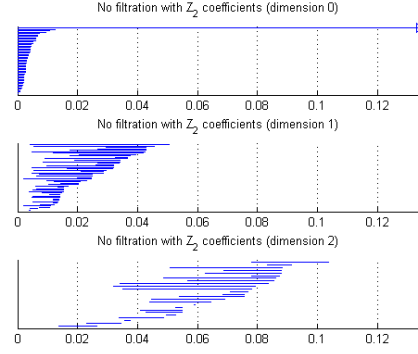


Figure 3.2: *Barcode for max-min landmark selection.*

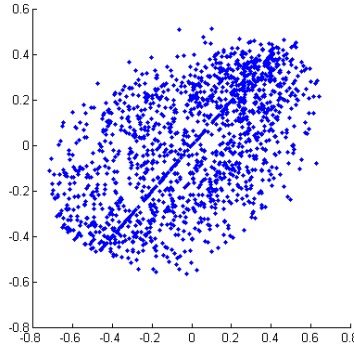


Figure 3.3: *Projection for unfiltered data.*

Here, the projections arise from taking the data in $\mathbb{R}^8$ and projecting it down to the $e_1 - e_2$ plane.

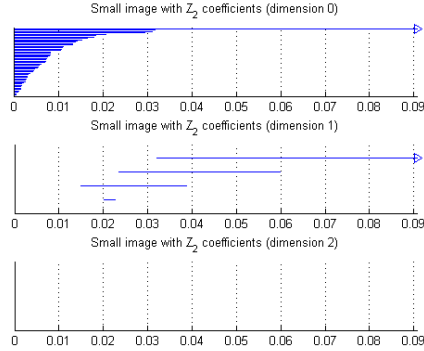


Figure 3.4: *Barcode for random landmark selection of filtered data.*

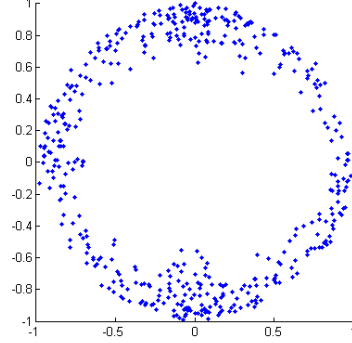


Figure 3.5: *Projection for filtered data.*

3.2 Discussion of Previous Results

When the densest parts of the patch space are inspected by Lee et al. in chapter 4 of [23], they find a concentration along linear gradients in all directions - that is, the most common types of high-contrast patches are sharp lines between dark and light areas. We verified the result of [12] that the persistence data suggest we are dealing with a surface that has homology agreeing with the three-circle model. This is in agreement with the results of Lee et al. [23]

Then, in chapter 7 of [22], Carlsson further studies the patch space by constructing a modified space. First, he notes that the *secondary circles* only appear on the barcode for lower values of k , i.e. there is more structure on a very local level in the patch space. As lowering k to less than 15 seems unwise due to noise issues, he looks at a larger set of patches ($1.25 \cdot 10^6$ patches compared to 10^4 before) and selects $k = 100$, which would have relatively corresponded to $k = 0.8$ for the smaller space. Then, he selects points in the patch data that lie closest to the purely quadratic gradients (discussed in more detail in Appendix B). He calls that small set of patches Q . The reason for adding Q to the dense dataset created by preprocessing is that while the quadratic patches are sparse in the data, they are important topologically. Now, when Q is added to the large data set, the persistent homology changes to having $b_2 = 1$ with $\mathbb{Z}_2$ coefficients. Carlsson claims this result is robust under different selection of patches.

First, we attempted the same approach on patch spaces of size 144,000

(our 1440×900 pixel images). For those spaces, with any size of Q , the main change we noticed was the addition of one or more long b_1 barcodes, indicating that we had added more circles to the space. We were only able to find a short b_2 interval for $k = 5$, but then our data was too noisy to have clear results for b_1 . We then attempted the approach on larger images on the $3 - 5$ megapixel range. As the preprocessing times were prohibitively long for our unrefined programs, we could not carry out a thorough examination, but we did analyse a few cityscapes. We did discover more structure in the $k = 10$ case, but that was in the form of up to 8 b_1 barcodes, with only very sporadic results in b_2 .

At this point, we wished to use homology with coefficients to verify which surface this corresponds to, as the only surfaces with the Betti numbers of Carlsson's results are the torus and Klein bottle. However, due to the problems with the set Q discussed above and the limitations discussed before in section 1.4, this was technically very difficult to achieve. Instead, Carlsson uses a fairly different method which is discussed in his paper with some extra clarification provided here in Appendix B. In fact, in [24], Carlsson does mention the possibility of using homology with different coefficients to determine the correct space, but it is not clear whether he has carried out that calculation on real data.

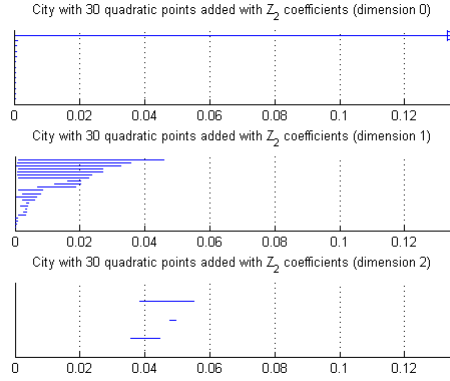


Figure 3.6: *Barcode for large cityscape with $p = 30$, $k = 10$ and 30 points added to the data.*

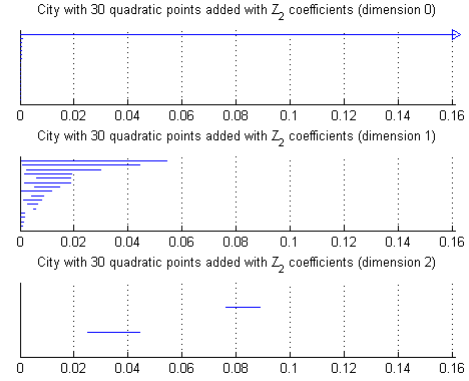


Figure 3.7: *Barcode for large cityscape with $p = 20$, $k = 5$ and 30 points added to the data.*

3.3 Results

As mentioned before, we were limited to analysing images around 1440×900 pixels in size due to using a very simple KNN algorithm. However, this was sufficient to provide us with on the order of 10,000 data points per image and then perform persistent homology on them.

Different images replicate the results of Carlsson with varying levels of success. However, it is clear that random input produces a null-result even if real data also produce noisy results. However, the three-circle model (Betti numbers 1-5-0) is quite fragile, but we do reproduce it. The Betti numbers themselves do not prove the Klein bottle model or even the three-circle model, which is just one of the many possible spaces have those Betti numbers.

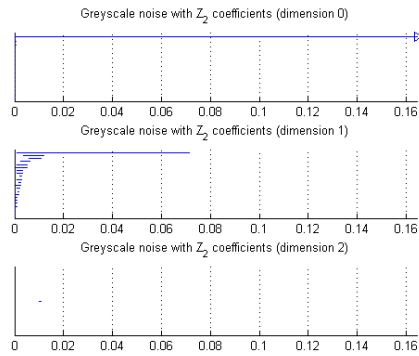


Figure 3.8: *Barcode for random noise with regular preprocessing.*

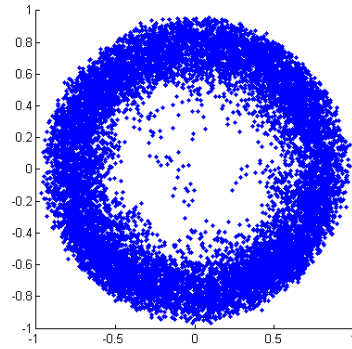


Figure 3.9: *Projection for random noise.*

For a quick visual overview of the behaviour of the patch space, a basic technique is to take the projection from the 8-dimensional space down to two dimensions. This obviously discards most of the structure in the data, but one can see the basic three-circle structure even in the projection.

As mentioned before, we should be able to tell a Klein bottle apart from a torus by using coefficients over different fields. However, we also note that the number of landmark points we use for images is far lower than the number required to see torsion in the Klein bottle point cloud.

For example, when we attempt a reconstruction from 6000 random points sampled from a Klein bottle with 50 landmarks selected, we get identical barcodes for both coefficients, with a medium length b_1 and with several

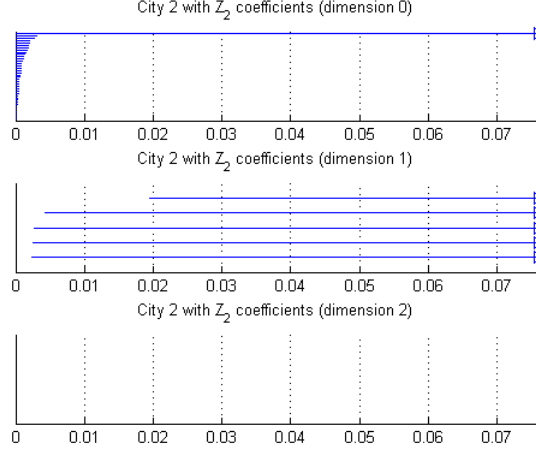


Figure 3.10: *The barcodes from preprocessing an image of a city with $k = 50$ and $p = 30$ verifying the three-circle model.*

medium b_2 intervals. For 100 landmarks, we get about 15,000 simplices. The Betti numbers are noisy, but $\mathbb{Z}_2$ clearly has b_1 and b_2 barcodes that are longer than the ones for $\mathbb{Q}$. For 200 landmarks giving about 50,000 simplices, there is no question about the results as the barcodes corresponding to the homology of the Klein bottle cover almost the entire persistence interval.

Indeed, when we attempt a construction using a high number of landmarks on the image data, our results have far too much noise. This once again shows how torsion is a very subtle phenomenon and as such requires a very clean data set.

We studied the effect of noise on the Klein bottle point cloud by writing a module to our random sampling class that adds a random error to each point, offsetting it from the actual surface of the Klein bottle. That is, we change the ideal coordinates $x_i \rightarrow x_i + \gamma\Delta$, where $\gamma \in (0, 1)$ is random and Δ is our error term. We discovered that the Betti numbers actually stay recognisable for error terms up to $\Delta = 0.5r$ and correct for up to $\Delta = 0.3r$. We do note, however, that the number of simplices in the complex increases greatly when error terms are added. This is as our landmarks no longer lie on a two-dimensional surface but form a higher-dimensional space.

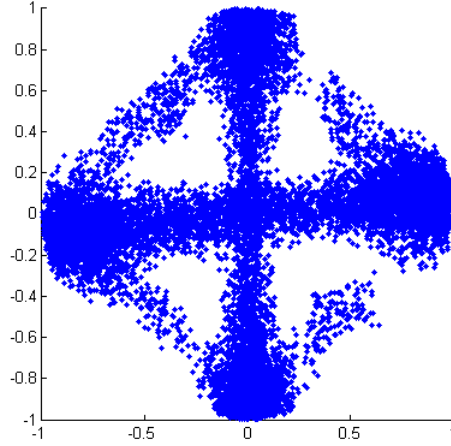


Figure 3.11: *The projection of the data from the picture as before onto the $e_1 - e_2$ plane. We see how the three-circle model comes about, but note that such a projection can only serve as a visual aid at best.*

Size of error term (units of r)	0	0.1	0.2	0.3	0.4
Number of simplices	54,000	54,000	65,000	90,000	147,000

Table 3.1: *Results of witness complex construction with 200 witness points for 6000 random points sampled with noise from a Klein bottle.*

Comparison of Barcodes

It is obvious but still surprising that the ‘complexity’ of an image is very different for Javaplex than it is in human eyes. In an attempt to find structures that differentiate between different types of images, we took a number of pictures of cities and number of pictures of faces. We carried out preprocessing and persistence procedures as described before.

The images of cities provide much clearer plots, most likely due to having mostly linear features. What is interesting is that faces have a more discernible *long-range* structure (k large), whereas for cities, the structure is there beginning at a short range. Looking at some sample patches of either image type, this is perhaps not too surprising. Furthermore, looking at sample pixel patches, one notices that perhaps patch size should be chosen according to expected feature size.

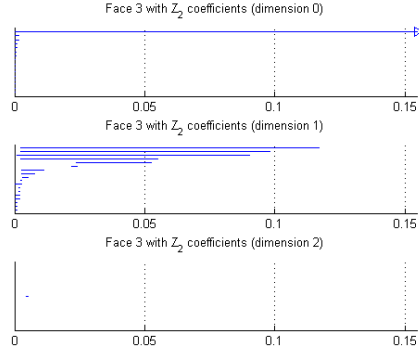


Figure 3.12: *The $p = 30$, $k = 150$ results for an image of a face.*

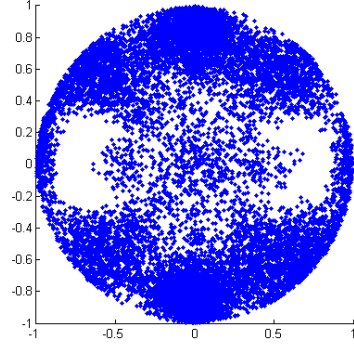


Figure 3.13: *The projection for the same image.*

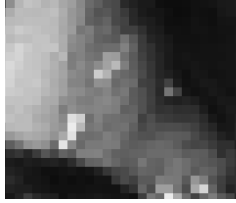


Figure 3.14: *Close-up of a face image.*

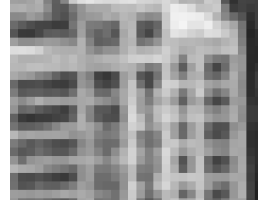


Figure 3.15: *Close-up of a city image.*

To get a quantitative picture of the differences between two pictures, we can use the *bottleneck distance*.

Definition 14 *The bottleneck distance between two persistence diagrams $dgm(f)$ and $dgm(g)$ is defined as $d_B(dgm(f), dgm(g)) = \inf_{\eta} \sup_x \|x - \eta(x)\|_{\infty}$ where η are bijections between the two persistence diagrams.*

In practice, η is any pairing of all points of two different persistence diagrams. Although one could define other metrics between barcodes, the bottleneck distance has the benefit of being stable under perturbations [25].

Procedure

When we attempt to use the bottleneck distance class in Javaplex, we notice that for persistence barcodes with a different number of persistent barcodes (barcodes that go on to the maximum filtration value), the bottleneck distance is returned as infinity. We believe this situation should give a bottleneck

distance no greater than the maximal filtration value, as in the worst case scenario, we could match a barcode persisting from 0 to the maximal radius to a point on the diagonal, which would give a bottleneck distance equal to the length of the longest barcode. After a rewriting of the bottleneck code, we calculated the mean pairwise distance inside a set of cityscapes, inside a set of faces and then between all of them mixed. While we can claim no statistical significance to our results due to a small number of samples, we did go through the process of comparison.

All of the images we used were of the same size and we carried out a $k = 20$, $p = 30$ preprocessing on them. Then, we found the bottleneck distance between one cityscape and all of the other ones. We took the average of these distances and divided it by the maximal filtration value for the image to get a relative measure of the difference. We repeated this process for each of the cityscapes and then finally took an average of all of the results.

We then repeated this for the whole group of pictures of faces and finally for all of the images mixed together. In this last approach, we took the relative measure using the cityscape or face maximal filtration values separately.

Although we only used a total of 100 pairings of image barcodes, the result is at least an indication of the ability of persistent homology to differentiate between different types of images. The mean distances between the cities were 0.46 ± 0.1 and from cities to faces 0.53 ± 0.2 ; between the faces this was 0.41 ± 0.13 and from faces to cities, 0.42 ± 0.05 .

One of the possible reasons for such small differences is the coarseness of the bottleneck measure - it only shows the one greatest distance between two barcodes. The bottleneck distance is in fact a limit of the Weierstrass metric:

$$d_{W^p}(a, b) = \inf_{\eta} \left(\sum_x (x - \eta(x))^p \right)^{\frac{1}{p}}. \quad (3.1)$$

The reason finite p values have not been as widely used is the fact that certain stability results of perturbing the barcodes only hold in the $p \rightarrow \infty$ limit. However, using a finite power of p would enable one to take into account all of the barcodes in the persistence diagrams [25].

Animation Analysis

In theory, the persistent homology component of our analysis can be used on any sort of data. A further application we attempted was using the method to study the local *space-time* behaviour of sequences of frames of an animation. This application is very attractive, as it could be used for applications ranging from motion detection in security systems to trajectory classification in robotics. To the best of our knowledge, this sort of analysis is also novel.

To achieve this, we divided each frame of a *gif* into 3×3 patches and then studied the behaviour of each patch in a sequence of three frames. Thus, our data points were vectors in $\mathbb{R}^{27}$ rather than $\mathbb{R}^9$ as before. As a filtering step, we only analysed patches that showed change during the three frames and we also attempted a density filtration by using a metric based on the relative change experienced by a pair of patches in the three-frame sequence. Unfortunately, we were unable to glean any conclusive results. The main reason for this is the fact that the larger the dimension of our data points (that is, the more complex the internal structure of any of the points we are studying), the more complicated it is to find structure in the data [26]. This issue is only exacerbated the fact that due to the limited scope of this project, we were only able to examine *gifs* with fewer than 50 frames and that we did not use extensive preprocessing on our data.

However, for future use, possible avenues to explore would be:

- Using the sliding windows approach [27] to find the series of frames which experience maximal motion.
- Exploring bigger data sets (such as film scenes).

3.4 Summary

While our description may sound rather pessimistic about the usefulness of persistent homology, applications such as a more efficient compression scheme for images [28] and a very effective texture recognition by Perea and Carlsson [29] should provide ample motivation to study the field.

Chapter 4

Further Developments

The basic idea of persistence has been quite fertile in finding applications, but there is also a great number of developments happening at quite fundamental levels to further improve persistence as a tool for data analysis.

4.1 Zig-zag Persistence

Looking at our procedure of image analysis, it is clear that the parameter k greatly influences our results. However, increasing values of k do not induce a filtration on the image patch space. This prohibits us from computing traditional persistence for k , but there is a theory called *zig-zag persistence* which enables us to look at sequences of spaces that are not necessarily nested. For example, consider the following sequence, where each map f_i could be in either direction:

$$\dots \xleftarrow{f_1} V_1 \xleftarrow{f_2} V_2 \xleftarrow{f_3} V_3 \xleftarrow{f_4} \dots \quad (4.1)$$

To give this sequence consistent directions, one could use the union sequence:

$$\dots V_1 \rightarrow V_1 \cup V_2 \leftarrow V_2 \rightarrow V_2 \cup V_3 \leftarrow V_3 \rightarrow \dots \quad (4.2)$$

Using a theoretical background of quiver theory (developed in the 1970s), Carlsson and de Silva have shown the completeness of zig-zag persistence [30], just as we included the results for regular persistence earlier. We will only mention that Gabriel's theorem is central to proving that result.

The computation of zig-zag persistence has also been extensively developed [31].

4.2 Multidimensional Persistence

While carrying out the persistence calculations for data sets, such as the image examples, picking “correct” values of various parameters, such as p , k or the number of near-quadratic patches to be added after preprocessing, can prove to be both difficult and ad hoc. Instead of choosing the values, one might wish to use persistence to determine the best values. This idea has led to the development of the theory of multidimensional persistence [6].

However, the approach has many challenges. The main issue is finding *complete invariants* - functions that always give different values over structures that are not isomorphic.

In addition to finding complete invariants, we would like them to have finite descriptions and be discrete so they would be representable and computable. Carlsson develops a multipersistence theory that is based on finitely generated graded modules, just as for traditional persistence. He also proves that this structure has a complete classification. The issue is that the invariants produced in this approach are continuous and thus not malleable for computation. We only note that this result comes from using algebraic varieties.

Carlsson also proposes some discrete invariants that are not complete in order to study the behaviour of multipersistence. In general, the approach is to not attempt a full classification of the multipersistence modules but to rather gain some insight by looking at the generators of the persistence module.

Chachólski et al. have also produced work regarding the nature of noise in multidimensional persistence [33]. Furthermore, multidimensional persistence has already been applied to biomolecular data [34] and there is quite a lot of hope it will prove useful in many more applications.

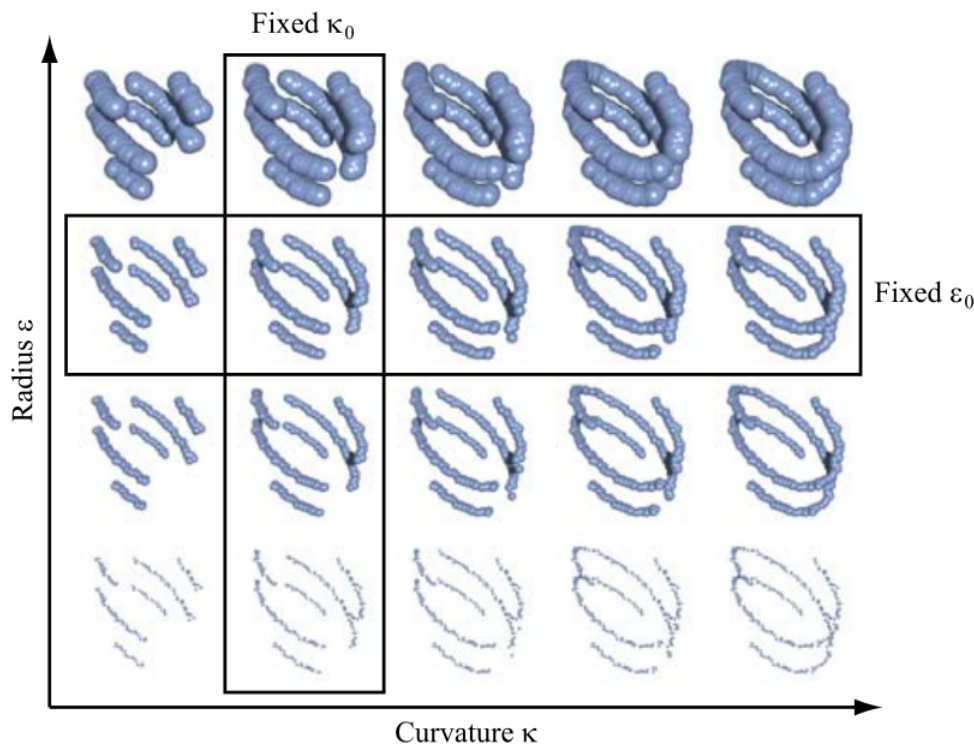


Figure 4.1: A bifiltration on a set of molecular data. The vertical filtration is given by the radius used in the construction, the horizontal one is given by including data with an increasing curvature, so both parameters provide filtrations. Figure taken from Carlsson’s *The Theory of Multidimensional Persistence* [32]

4.3 Computation

There is a plethora of different software for the calculation of persistent homology, and there is an extensive overview paper on them by Otter et al. [20]. While *Javaplex* is by now somewhat out-dated in terms of the speed of computation, we chose to use it due to it being well-documented and generally user-friendly. Furthermore, many other programs do not enable the computation of homology using different coefficient fields, which was quite important for us. While much can be said about the different packages, in this section, we will highlight two of the methods for making persistence

calculations more efficient.

The fundamental issue with topological estimation is that the complexes used to give faithful representations of the underlying space tend to become unmanageably large, which makes their construction time the limiting factor for persistent homology. Yet we have seen that the witness complex can provide us with useful information. Sheehy et al. [35] have used the very similar idea of sparse filtrations. They have proven results like the nerve theorem for constructions using landmark sets, which is exactly the kind of guarantees that we would like for witness complexes. However, their approach, called *greedy permutation*, does not directly generalise to witness complexes. They have also provided algorithms that construct a sparse filtration from greedy permutations in linear time, however, these have not yet been included in any computational software to the best of our knowledge. Their work builds upon previous results [36] that built filtrations in $O(n \log n)$ time.

A tried and functional method, which is also applied in a few of the software packages, is reduction of simplicial complexes using discrete morse theory. The method essentially removes certain pairs of simplices, which is guaranteed not to change the homology of the simplicial complex. For an overview, see for example [37].

4.4 Statistics

Persistent homology being a tool in topological *data analysis*, one would hope that the field of statistics would provided guidance and inspiration. However, the interaction between the fields is only starting to be explored.

One of the approaches that has already delivered results is using persistent homology as one step in a general data analysis process, with other steps using other statistical methods. For example, Vitaliy Kurlin has studied edge identification using persistent homology in computer vision [38]. Furthermore, he has applied this approach for currently-used methods, such as superpixels.

Another approach is using statistics to glean more information from persistence barcodes. For example, Miller et al. studied the topology of brain arteries [39] using this idea. For each of the brains analysed, they took the one hundred longest persistent barcodes and turned it into a $\mathbb{R}^{100}$ vector and then they carried out statistical analysis on the set of $\mathbb{R}^{100}$ vectors. This approach enabled them to observe structural changes that strongly correlated

with the age of the brain. Furthermore, they were able to use methods of traditional statistics to verify the significance of their results.

While persistence is meant to be robust under noise, several papers draw attention to issues in practical applications [40],[11]. One would hope that statistical methods which have been used to deal with the issues could also be applied in conjunction with persistent homology, but this is still a largely unexplored area.

4.5 Summary

We hope to have provided an overview of persistent homology that is useful as a first point of contact. Furthermore, we hope we have demonstrated the practical applications of persistence and given justification to its usefulness.

The future of the fields depends, in the authors' humble opinion, on the ability of mathematicians providing the theory and its further developments to work in conjunction with the computer scientists and statisticians whose expertise will be invaluable in helping the field reach maturity and gain more wide-spread recognition and use.

Appendix A

Matrix calculations

We present our 3x3 pixel patches as vectors $\mathbf{x} \in \mathbb{R}^9$.

We define the d-norm of a patch as $\|\mathbf{x}\|_D = \sqrt{\sum_{i \sim j} (x_i - x_j)^2}$, where $i \sim j$ if they are neighbouring pixels in a patch. We can write this as a matrix calculation: $\|\mathbf{x}\|_D = \sqrt{\mathbf{x}^T D \mathbf{x}}$, where

$$D = \begin{pmatrix} 2 & -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 3 & -1 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 2 & 0 & 0 & -1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 3 & -1 & 0 & -1 & 0 & 0 \\ 0 & -1 & 0 & -1 & 4 & -1 & 0 & -1 & 0 \\ 0 & 0 & -1 & 0 & -1 & 3 & 0 & 0 & -1 \\ 0 & 0 & 0 & -1 & 0 & 0 & 2 & -1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & -1 & 3 & -1 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & 2 \end{pmatrix}. \quad (\text{A.1})$$

The change into the convenient DCT basis is given by the matrix with the following rows

$$\begin{aligned}
\mathbf{e}_1 &= \frac{1}{\sqrt{6}}(1, 0, -1, 1, 0, -1, 1, 0), \\
\mathbf{e}_2 &= \frac{1}{\sqrt{6}}(1, 1, 1, 0, 0, 0, -1, -1, -1), \\
\mathbf{e}_3 &= \frac{1}{\sqrt{54}}(1, -2, 1, 1, -2, 1, 1, -2, 1), \\
\mathbf{e}_4 &= \frac{1}{\sqrt{54}}(1, 1, 1, -2, -2, -2, 1, 1, 1), \\
\mathbf{e}_5 &= \frac{1}{\sqrt{8}}(1, 0, -1, 0, 0, 0, -1, -1, -1), \\
\mathbf{e}_6 &= \frac{1}{\sqrt{48}}(1, 0, -1, -2, 0, 2, 1, 0, -1), \\
\mathbf{e}_7 &= \frac{1}{\sqrt{48}}(1, -2, 1, 0, 0, 0, -1, 2, -1), \\
\mathbf{e}_8 &= \frac{1}{\sqrt{216}}(1, -2, 1, -2, 4, -2, 1, -2).
\end{aligned}$$

Both of these are used and described in Lee et al. [23].

For the *KNN* algorithm, we take the Euclidean distance for $\mathbb{R}^9$ between all pairs of patches.

Appendix B

Clarification of Carlsson's Results

In [22] Carlsson uses an alternative description of the patch space by using two-variable polynomials. He does so by giving a 3×3 patch coordinates $x, y \in \{-1, 0, 1\}$ and then evaluating any possible polynomial at each of those values of x and y , giving the results of the evaluations as a vector in $\mathbb{R}^9$. A particular choice for the representation he uses is polynomials of the form $c(ax + by)^2 + d(ax + by)$.

In this notation, quadratic patches are of the form $\pm(ax + by)^2$ with $a, b \neq 0$. When converted back to the patch representation (i.e. when evaluated on $x, y \in \{-1, 0, 1\}$), they produce a nine-vector:

$$(a^2 - 2ab + b^2, b^2, a^2 + 2ab + b^2, a^2, 0, a^2, a^2 - 2ab + b^2, b^2, a^2 + 2ab + b^2). \quad (\text{B.1})$$

For our work, we then generated $|Q|$ sample patches with random values of a and b and carried out the standard preprocessing steps on them.

For a theoretical argument for the Klein bottle, Carlsson uses a map $g : S_1 \times S_1 \rightarrow K$ such that $(a, b, c, d) \mapsto c(ax + by)^2 + d(ax + by)$ for the polynomial representation of patches. Clearly, this map is not injective as $g(a, b, c, d) = g(-a, -b, c, -d)$. Thinking of $S_1 \times S_1$ as being parametrised by (θ, ϕ) , we see that the non-injectivity of g corresponds exactly to the identifications made on the unit square to create the Klein bottle (figure 1.4).

Bibliography

- [1] Gunnar Carlsson. Topology and data. *Bulletin of the American Mathematical Society*, 46(2):255–308, 2009.
- [2] Gunnar Carlsson. Topological pattern recognition for point cloud data. *Acta Numerica*, 23:289–368, 2014.
- [3] Robert Ghrist. Elementary applied topology. *Book in preperation*, 2014.
- [4] Afra J Zomorodian. *Topology for computing*, volume 16. Cambridge University Press, 2005.
- [5] Herbert Edelsbrunner and John Harer. *Computational topology: an introduction*. American Mathematical Soc., 2010.
- [6] Gunnar Carlsson and Afra Zomorodian. The theory of multidimensional persistence. 42(1):71–93.
- [7] Allen Hatcher. *Algebraic topology*. , 2002.
- [8] Afra J. Zomorodian. *Topology for Computing*. Cambridge University Press.
- [9] Karol Borsuk. On the imbedding of systems of compacta in simplicial complexes. *Fund. Math.*, 35:217–234, 1948.
- [10] P. Bendich, D. Cohen-Steiner, H. Edelsbrunner, J. Harer, and D. Morozov. Inferring local homology from sampled stratified spaces. In *48th Annual IEEE Symposium on Foundations of Computer Science, 2007. FOCS '07*, pages 536–546.

- [11] Joshua Garland, Elizabeth Bradley, and James D. Meiss. Exploring the Topology of Dynamical Reconstructions. *arXiv:1506.01128 [math]*, June 2015. arXiv: 1506.01128.
- [12] Gunnar Carlsson and Vin de Silva. Topological estimation using witness complexes.
- [13] Donald R. Sheehy. (multi)filtering noise in geometric persistent homology. In *The Fall Workshop in Computational Geometry*, 2010.
- [14] Herbert Edelsbrunner and Dmitry Morozov. Persistent homology: Theory and practice.
- [15] Peter Bubenik and Jonathan A Scott. Categorification of persistent homology. *Discrete & Computational Geometry*, 51(3):600–627, 2014.
- [16] Afra Zomorodian and Gunnar Carlsson. Computing persistent homology. 33(2):249–274.
- [17] Robert Ghrist. Barcodes: The persistent topology of data.
- [18] Persi Diaconis, Susan Holmes, and Mehrdad Shahshahani. Sampling from a manifold. In *Institute of Mathematical Statistics Collections*, pages 102–125. Institute of Mathematical Statistics.
- [19] Vin De Silva, Dmitriy Morozov, and Mikael Vejdemo-Johansson. Persistent cohomology and circular coordinates. *Discrete & Computational Geometry*, 45(4):737–759, 2011.
- [20] Nina Otter, Mason A. Porter, Ulrike Tillmann, Peter Grindrod, and Heather A. Harrington. A roadmap for the computation of persistent homology.
- [21] Chao Chen and Michael Kerber. Persistent homology computation with a twist. In *Proceedings 27th European Workshop on Computational Geometry*, volume 11, 2011.
- [22] Gunnar Carlsson and Vin de Silva. On the local behavior of spaces of natural images.
- [23] Ann B. Lee, Kim S. Pedersen, and David Mumford. The nonlinear statistics of high-contrast patches in natural images. 54(1):83–103.

- [24] Gunnar Carlsson. Topological pattern recognition for point cloud data. 23:289–368.
- [25] Herbert Edelsbrunner and John Harer. Persistent homology-a survey. *Contemporary mathematics*, 453:257–282, 2008.
- [26] Qingli Yin. On qualitative analysis of high-contrast patches in range images. In W. Eric Wong, editor, *Proceedings of the 4th International Conference on Computer Engineering and Networks*, pages 573–582. Springer International Publishing, 2015.
- [27] J. Perea and J. Harer. Sliding Windows and Persistence: An Application of Topological Methods to Signal Analysis. *ArXiv e-prints*, July 2013.
- [28] Arian Maleki, Morteza Shahram, and Gunnar Carlsson. A near optimal coder for image geometry with adaptive partitioning. In *Image Processing, 2008. ICIIP 2008. 15th IEEE International Conference on*, pages 1061–1064. IEEE, 2008.
- [29] Jose A. Perea and Gunnar Carlsson. A klein-bottle-based dictionary for texture representation. 107(1):75–97.
- [30] Gunnar Carlsson and Vin de Silva. Zigzag persistence. 10(4):367–405.
- [31] Gunnar Carlsson, Vin de Silva, and Dmitriy Morozov. Zigzag persistent homology and real-valued functions. In *Proceedings of the twenty-fifth annual symposium on Computational geometry*, pages 247–256. ACM, 2009.
- [32] Gunnar Carlsson and Afra Zomorodian. The theory of multidimensional persistence. 42(1):71–93.
- [33] Wojciech Chachlski, Anders Lundman, Ryan Ramanujam, Martina Scalamiero, and Sebastian berg. Multidimensional persistence and noise.
- [34] Kelin Xia and Guo-Wei Wei. Multidimensional persistence in biomolecular data. *arXiv preprint arXiv:1412.7679*, 2014.
- [35] Nicholas J Cavanna, Mahmoodreza Jahanseir, and Donald R Sheehy. A geometric perspective on sparse filtrations. *arXiv preprint arXiv:1506.03797*, 2015.

- [36] Donald Sheehy. Linear-size approximations to the vietoris-rips filtration. *CoRR*, abs/1203.6786, 2012.
- [37] Shaun Harker, Konstantin Mischaikow, Marian Mrozek, and Vidit Nanda. Discrete morse theoretic algorithms for computing homology of complexes and maps. *Foundations of Computational Mathematics*, 14(1):151–184, 2014.
- [38] V. Kurlin. Auto-completion of contours in sketches, maps and sparse 2d images based on topological persistence. In *Proceedings of SYNASC 2014 workshop CTIC: Computational Topology in Image Context*, pages 594–601, 2014.
- [39] Paul Bendich, J. S. Marron, Ezra Miller, Alex Pieloch, and Sean Skwerer. Persistent homology analysis of brain artery trees.
- [40] Laura Bolzano and Jordan S. Ellenberg. Understanding persistent homology and plex using a networking dataset.