The phom package: User's manual

Version 1.0.0

Andrew Tausz

February 13, 2012

The phom package is an R package that computes the persistent homology of geometric datasets. Persistent homology is an algebraic tool that allows one to understand the topological characteristics of a given dataset across all spatial scales. It may be thought of as an extension of clustering to higher-dimensional homological properties. The purpose of this package is to make these tools available to the statistical community.

The phom package may be cited as follows:

> Andrew Tausz, *phom: Persistent Homology in R*, Version 1.0.0, 2011. Available at CRAN http://cran.r-project.org.

Contents

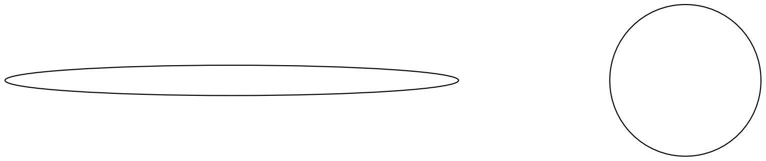
1 Introduction to Persistent Homology

Given a set of points, X, we are interested in understanding the shape of this dataset. A recently developed technique which addresses this problem is known as *persistent homology*, which allows one to compute a multi-scale decomposition of the topological features of the dataset. The theory of persistent homology originates in standard (simplicial) homology from algebraic topology. Roughly speaking, simplicial homology defines a set of algebraic invariants of simplicial complexes which describe the topological characteristics of the shape.

We refer the reader to [Car09] for a very readable introduction to the field of topological data analysis as well as the computational tasks involved. Further computational details are available in [ZC05]. Standard references on algebraic topology and its algebraic foundations include [Hat02, Mun93, Wei95, ML75, Rot08].

For example, if $X = S^1$, the unit circle, then X has homology groups $H_0(X) = H_1(X) = \mathbb{Z}$, and $H_i(X) = 0$ for $i > 1$. Informally speaking, H_0 counts the number of connected components of a topological space, and H_1 counts the number of 1-dimensional holes. Similarly, the higher homology groups encode information about higher-dimensional voids. On slightly more exotic spaces, other topological invariants such as (un)orientability may be reflected in the torsion components of the homology groups. We refer to the rank of the i-th homology group (the number of $\mathbb{Z}$ factors) as the i-th Betti number, denoted by β_i.

Persistent homology computes topological invariants of filtered sequences of simplicial complexes. The relevance of this to data-analysis is the following: Conventional topological invariants are unsuitable for many application purposes since they are simultaneously too sensitive, and too weak. They are too "weak" because of their homotopy invariance. This means that the underlying shape can be stretched in any way (as long as it is not ripped) without changing its homology groups. For example, since the two shapes below are homeomorphic, there is no topological invariant that will distinguish them, even though there are visible geometric differences.



Similarly, the following figure illustrates what is meant by (conventional) homology being unstable. An arbitrarily small perturbation may have the effect of changing the homology. The figure on the right has $\beta_1 = 1$, whereas the figure on the left has $\beta_1 = 2$.

Persistent homology, on the other hand, is an invariant of *filtered* spaces. By this, we mean a collection of topological spaces (simplicial complexes in our case)

$$K_1 \hookrightarrow K_2 \hookrightarrow \ldots \hookrightarrow K_n$$

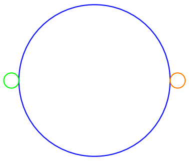
which are nested. An example of such a filtered simplicial complex is seen in Figure 1. In this figure we can see that the complex starts out as a discrete set of points, and points are joined together as we increase a scale parameter. Section 2 describes such constructions in more detail.

Fixing a field, $\mathbb{F}$, we may apply the p-dimensional homology functor, $H_p(-, \mathbb{F})$ to get the sequence of $\mathbb{F}$-vector spaces with associated maps:

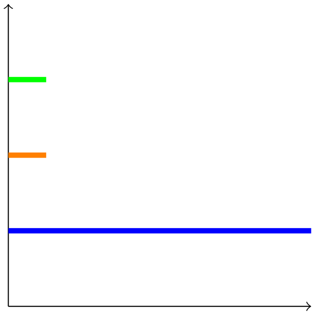
$$H_p(K_1, \mathbb{F}) \to H_p(K_2, \mathbb{F}) \to \ldots \to H_p(K_n, \mathbb{F})$$

It turns out that algebraically, such sequences may be decomposed into a multiset of intervals, $\{[a_i, b_i) \subseteq \{1, \ldots, n, \infty\}\}$. We do not delve into the algebraic details here, but merely assure the reader that significant effort has gone into placing the above statements on sound mathematical theory. The presence of the interval $[a, b)$ in dimension p indicates the presence of a p-dimensional homology class (something we informally call a topological feature) that is born at index a, and that dies at index b. For example, in dimension 0 this may be a connected component that is present at a given length scale, but that ends once it is merged with other connected components as we increase the length-scale. Intuitively, long intervals correspond with significant features, whereas short intervals may be considered to be noise. We refer to this multiset of intervals as a barcode.

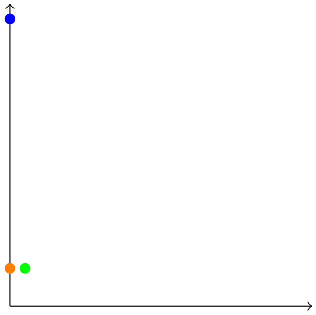
For example, computing the persistent homology of the figure



gives the interval decomposition which may be represented by the following barcode plot. Note that the vertical positioning does not have any relevance.



Alternatively, we may draw a *persistence diagram* which is a set of points in the plane in which the x-coordinate represents the start of an interval, and the y-coordinate represents the ending point.



2 Generating Filtered Complexes from Geometric Data

The computation of persistent homology is a two-stage process. The two main steps are:

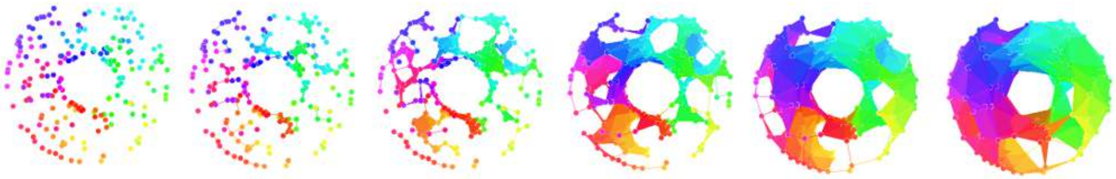
- Construct a filtered simplicial complex given a geometric dataset
- Compute the persistent homology of the filtered complex

The `phom` package exposes two ways of constructing a filtered complex on a metric space, $(\mathcal{X}, d)$.

2.1 The Vietoris-Rips Construction

We define the filtered complex $\mathrm{VR}(\mathcal{X}, r)$ as follows. Suppose that the points of $\mathcal{X}$ are $\{x_1, ...x_N\}$, where $N = |\mathcal{X}|$. The Vietoris-Rips complex is constructed as follows:

- **Add points:** For all points $x \in \mathcal{X}$, $x \in \mathrm{VR}_0(\mathcal{X}, 0)$



Figure˜1: Example of a lazy-witness complex generated from randomly sampled points on a torus. This is an example of a filtered simplicial complex. Note that the complex starts out as a discrete set of points, and as we increase the scale parameter, points are joined together with nearby points. Higher simplices arise when groups of points are mutually close together.

- **Add 1-skeleton:** The 1-simplex $[x_i, x_j]$ is in $\mathrm{VR}_1(\mathcal{X}, r)$ iff $d(x_i, x_j) \leq r$
- **Expansion:** We define $\mathrm{VR}(\mathcal{X}, r)$ to be the maximal simplicial complex containing $\mathrm{VR}_1(\mathcal{X}, r)$. That is, a simplex $[x_0, ..x_k]$ is in $\mathrm{VR}(\mathcal{X}, r)$ if and only if all of its edges are in $\mathrm{VR}_1(\mathcal{X}, r)$.

An extensive discussion on algorithms for computing the Vietoris-Rips complex can be found in [Zom10]. The phom implementation is based on the results of this paper.

2.2 The Lazy-Witness Construction

The fundamental idea behind the lazy-witness construction is that a relatively small subset of a point cloud can accurately describe the shape of the dataset. This construction has the advantage of being more resistant to noise than the Vietoris-Rips construction. An extensive discussion about it can be found in [dSC04].

The lazy-witness construction starts with a selection of landmark points, $\mathcal{L} \subset \mathcal{X}$ with $|\mathcal{L}| = L$. One possibility is to simply choose a random subset of $\mathcal{X}$. Another possibility is to perform a sequential max-min selection: An initial point l_0 is selected, and then we inductively select the point l_k which maximizes the minimum distance to all previously generated points. This max-min construction tends to produce more evenly spaced points than the random selection. Again we refer the reader to [dSC04] for a more detailed discussion, as well as empirical results supporting these claims.

This construction is parameterized by a value ν, which most commonly takes the values 0, 1, or 2. We also define the distance matrix D to contain the pairwise distances between the points in $\mathcal{X}$.

- **Define m_i:** If $\nu = 0$, let $m_i = 0$, otherwise, define m_i to be the ν-th smallest entry in the i-th column of D
- **Add points:** For all points $l \in \mathcal{L}$, $l \in \mathrm{LW}_0(\mathcal{X}, 0, \nu)$
- **Add 1-skeleton:** The 1-simplex $[l_i, l_j]$ is in $\mathrm{LW}_1(\mathcal{X}, r, \nu)$ iff there exists an $x \in \mathcal{X}$ such that $\max(d(l_i, x), d(l_j, x)) \leq r + m_i$.
- **Expansion:** We define $\mathrm{LW}(\mathcal{X}, r, \nu)$ to be the maximal simplicial complex containing $\mathrm{LW}_1(\mathcal{X}, r, \nu)$.

An example of a lazy-witness complex on random points on a torus may be found in Figure 1. We remind the reader that this is a set of snapshots of a continuously varying set of nested spaces.

3 Homology Computation

At the core of the phom library is the set of algorithms that actually compute the homology of a filtered chain complex. Key references to background material regarding these algorithms can be found in [ZC05, dSMVJ10]. Although we do not describe them in detail here, we note that the algorithms for computing persistent homology can be formulated as a matrix decomposition problem. The fundamental reason for this is the equivalence of category of persistent vector spaces of finite type, and the category finitely generated graded modules over $\mathbb{F}[t]$. This correspondence is described in [ZC05].

The homology algorithms are built in a way that is optimized for chain complexes implemented as *streams*. By this we mean that a filtered chain complex is represented by a sequence of basis elements that are produced in increasing order of their filtration indices. Enforcing the constraint that all complexes must be implemented this way allows `phom` to perform the matrix decomposition operations in an efficient online fashion.

4 Methods

The `phom` package contains a main method for computing persistent homology, `pHom`, as well as two for visualizing the results of such a computation, `plotPersistenceDiagram` and `plotBarcodeDiagram`.

4.1 `pHom`

Description

This function computes persistent homology on a given dataset. This is a two-step process which involves: (1) creating a filtered simplicial complex on the dataset, and (2) computing the persistent homology of the filtered complex.

It outputs a matrix with three columns. Each row in the output matrix corresponds to a persistence interval. The first column stores the dimension of the interval, the second stores the starting point, and the third stores the ending point.

The method provides two ways to construct a filtered simplicial complex on the data points: the Vietoris-Rips construction, and the lazy-witness construction.

Note that the Vietoris-Rips construction can produce complexes that are very large. To circumvent this, the witness construction may be used. This method produces complexes that are significantly smaller.

The argument `mode` is used to select between the two different constructions, by either specifying `"vr"` or `"lw"`.

Usage

`pHom`*(X, dimension, max_filtration_value, mode = "vr", metric = "euclidean", p = 2, landmark_set_size = 2 * ceiling(sqrt(length(X))), maxmin_samples = min(1000, length(X)))*

Arguments

X

A matrix which has one of the two following interpretations. In the case where `metric = "distance_matrix"`, `X` is required to be a square matrix whose entries indicate the distance between two points. The number of rows (and columns) equals the number of points in the dataset. In the case where `metric` is something other than `"distance_matrix"`, `X` is required to be a matrix in which the rows are points in Euclidean space. The number of columns is the dimensionality of the dataset. Note that for different choices of the `metric` argument, the points will be endowed with different metrics.

dimension

The maximum dimension to compute persistent homology to.

max_filtration_value

The maximum filtration value to use in constructing the filtered complex.

mode

This indicates the type of filtration to use. The two possible choices are `"vr"` (default) and `"lw"`. The choice `"vr"` indicates that the Vietoris-Rips filtration will be used, and the choice `"lw"` indicates that the lazy-witness construction will be used. For Vietoris-Rips filtrations, the parameters `landmark_set_size` and `maxmin_samples` are ignored.

metric

This indicates the type of metric that will be used. Valid choices include the following: `"distance_matrix"`, `"euclidean"`, `"maximum"`, `"manhattan"`, `"canberra"`, `"binary"`, `"minkowski"`. When `"distance_matrix"` is specified, the paramter `"X"` is interpreted as a distance matrix. Otherwise, `"X"` is regarded as a set of points, where each row is one point.

p

This is the value of the power to use in the minkowski metric.

landmark_set_size

The number of points to include in the landmark set. A sensible value for this is in between 20 and 100. The default value is taken to be $2\sqrt{|X|}$. This parameter is only relevant for the lazy-witness filtration.

maxmin_samples

The number of samples to use when performing the maxmin selection. The default value is taken to be $\min(|X|, 1000)$. This parameter is only relevant for the lazy-witness filtration.

4.2 `plotPersistenceDiagram`

Description

This function plots a persistence diagram from a given set of intervals.

Usage

`plotPersistenceDiagram`*(intervals, max_dim, max_f, title="Persistence Diagram")*

Arguments

intervals

A matrix with three columns that specifies the persistence intervals. Entries in the first column indicate the dimension of an interval. The entries in the second and third columns indicate the start and end points of the intervals, respectively. The function `pHom` produces outputs that are in this form.

max_dim

The maximum dimension to plot.

max_f

The maximum filtration value to use in the persistence diagram.

title

The title on the persistence diagram.

4.3 `plotBarcodeDiagram`

Description

This function plots a set of intervals as a set of line-segments. In comparison the function `plotPersistenceDiagram` described in section 4.2 plots a set of intervals as points in the plane corresponding to their start and end points.

Usage

`plotBarcodeDiagram`*(intervals, dimension, max_f, title="Persistence Diagram")*

Arguments

intervals
A matrix with three columns that specifies the persistence intervals. Entries in the first column indicate the dimension of an interval. The entries in the second and third columns indicate the start and end points of the intervals, respectively. The function pHom produces outputs that are in this form.
dimension
The dimension to plot intervals for. Unlike plotPersistenceDiagram, this function only plots intervals for one dimension, and not all of them.
max_f
The maximum filtration value to use in the diagram.
title
The title on the barcode diagram.

5 Examples

5.1 Mixture of Two Gaussians

This example computes the 0-dimensional persistent homology for a set of points generated by a mixture of two Gaussians. We expect that the persistent homology will indicate two clusters. Note that the l_1 (manhattan) norm is used in this example.

```
> library(phom)
> N <- 50
> x1 <- rnorm(N) * 0.1
> y1 <- rnorm(N) * 0.1
> X1 <- t(as.matrix(rbind(x1, y1)))
> x2 <- rnorm(N) * 0.1 + 0.5
> y2 <- rnorm(N) * 0.1 + 0.5
> X2 <- t(as.matrix(rbind(x2, y2)))
> x <- cbind(x1, x2)
> y <- cbind(y1, y2)
> X <- as.matrix(rbind(X1, X2))
> max_dim <- 0
> max_f <- 0.8
> intervals <- pHom(X, max_dim, max_f, metric = "manhattan")
```

To plot the dataset, we use the following command. The resulting plot is shown in Figure 2.

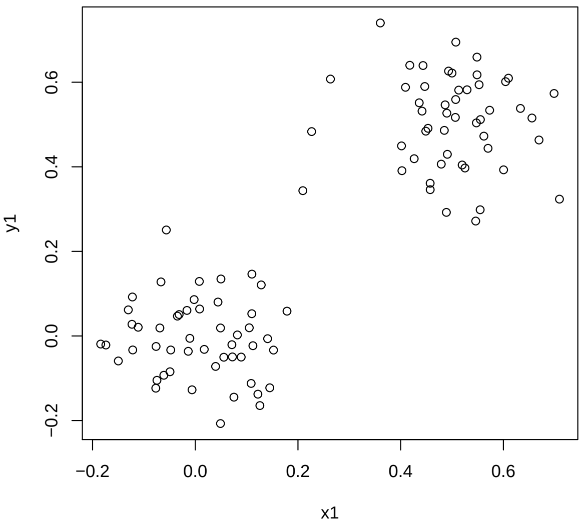
```
> plot(X)
```

To plot, we use the function plotBarcodeDiagram as follows. The resulting plot is shown in Figure 3. In this figure, we can see that there are two prominent intervals, suggesting that there are two clusters.

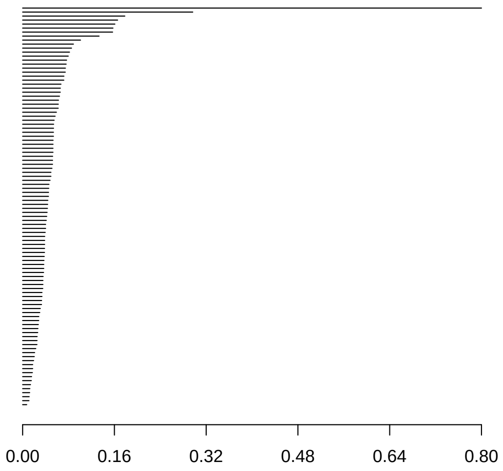
```
> plotBarcodeDiagram(intervals, max_dim, max_f, title = "")
```

5.2 Rips Filtration on Points on a Circle

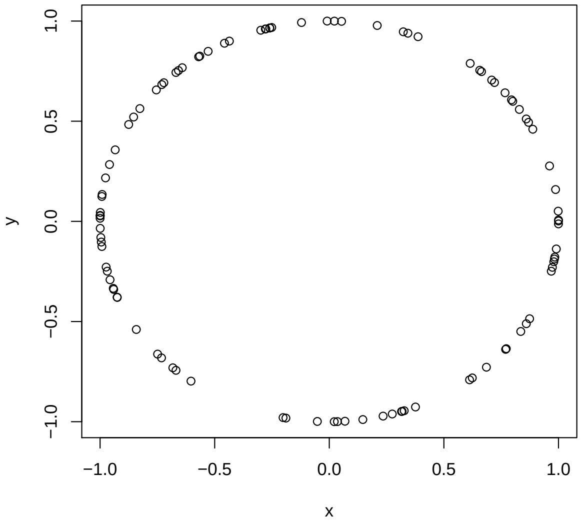
As a first example, let us consider generating uniformly distributed points on a circle. The default Euclidean norm is used in this example.



Figure˜2: Mixture of two Gaussians dataset



Figure˜3: Dimension 0 intervals for mixture of Gaussians example



Figure~4: Uniform points on S^1

```
> library(phom)
> t <- 2 * pi * runif(100)
> x <- cos(t)
> y <- sin(t)
> X <- t(as.matrix(rbind(x, y)))
> max_dim <- 1
> max_f <- 0.6
> intervals <- pHom(X, max_dim, max_f)
```

To plot the dataset, we use the following command. The resulting plot is shown in Figure 4.

```
> plot(X)
```

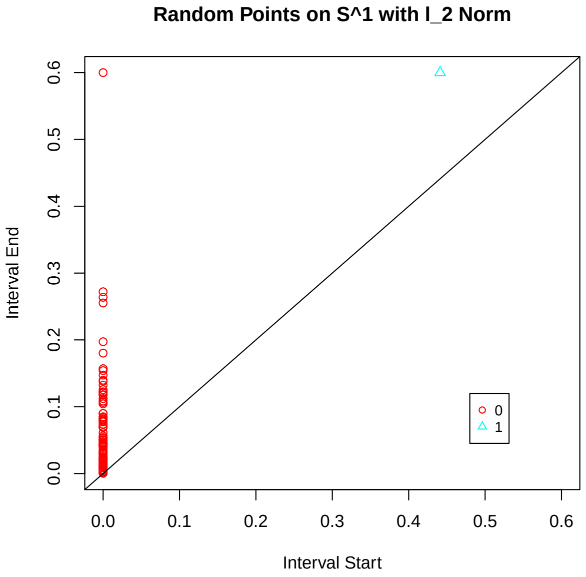
For this example, we use the function `plotPersistenceDiagram` to visualize the intervals. The resulting plot is shown in Figure 5. Note that we can see that there are two significant generators - one in dimension 0 and one in dimension 1.

```
> plotPersistenceDiagram(intervals, max_dim, max_f, title = "Random Points on S^1 with l_2 Norm")
```

5.3 Lazy-Witness Filtration for Points on S^3

In this example, we generate points on S^3. The lazy-witness construction is used to reduce the initial dataset of size 5,000 to one of size 40.

```
> library(phom)
> sphere_points <- function(n, d) {
+     points <- matrix(rnorm(n * d), nrow = n, ncol = d)
```



Figure~5: Persistence Diagram for uniformly chosen points on S^1

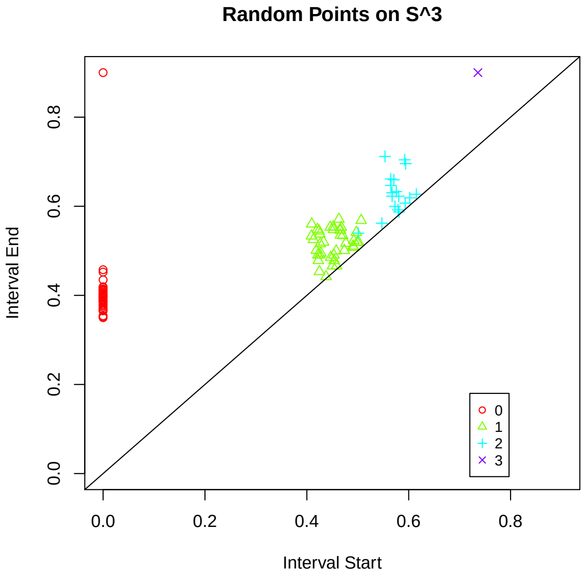
```
+     L <- apply(points, MARGIN = 1, FUN = function(x) {
+         sqrt(sum(x * x))
+     })
+     D <- diag(1/L)
+     U <- D %*% points
+     U
+ }
> n <- 5000
> d <- 4
> X <- as.matrix(sphere_points(n, d))
> max_dim <- d - 1
> max_f <- 0.9
> landmark_set_size <- 40
> maxmin_samples <- 1000
> intervals <- pHom(X, max_dim, max_f, mode = "lw", metric = "euclidean",
+     landmark_set_size = landmark_set_size, maxmin_samples = maxmin_samples)
```

Of course, since it is not possible to visualize the 3-sphere we do not include a plot of the original dataset. The resulting persistence diagram may be found in Figure 6.

```
> plotPersistenceDiagram(intervals, max_dim, max_f, title = "Random Points on S^3")
```

References

[Car09] Gunnar Carlsson, *Topology and data*, Bulletin of the American Mathematical Society **46** (2009), no.~2, 255–308.



Figure˜6: Persistence Diagram for lazy-witness filtration for points on S^3

[dSC04] Vin de˜Silva and Gunnar Carlsson, *Topological estimation using witness complexes*, Eurographics Symposium on Point-Based Graphics (M.˜Alexa and S.˜Rusinkiewicz, eds.), The Eurographics Association, 2004.

[dSMVJ10] Vin de˜Silva, Dmitriy Morozov, and Mikael Vejdemo-Johansson, *Dualities in persistent (co)homology*, Unpublished manuscript, 2010.

[Hat02] Allen Hatcher, *Algebraic topology*, Cambridge University Press, Cambridge, 2002. MR MR1867354

[ML75] Saunders Mac˜Lane, *Homology*, 3rd ed., Springer, October 1975.

[Mun93] James˜R. Munkres, *Elements of algebraic topology*, Westview Press, December 1993.

[Rot08] Joseph˜J. Rotman, *An introduction to homological algebra*, 2nd ed., Springer, October 2008.

[Wei95] Charles˜A. Weibel, *An introduction to homological algebra*, Cambridge University Press, October 1995.

[ZC05] Afra Zomorodian and Gunnar Carlsson, *Computing persistent homology*, Discrete Comput. Geom **33** (2005), 249–274.

[Zom10] A.˜Zomorodian, *Fast construction of the Vietoris-Rips complex*, Computers & Graphics **34** (2010), no.˜3, 263 – 271.