Using persistence barcodes to detect relationships in the presence of delayed oscillations

Luke Wolcott

Abstract

We give examples, using simulated data, of how topological data analysis can be used to detect relationships in a system with delayed oscillations. This Markdown file accompanies and explains the R scripts and barcode images contained in the GitHub repo github.com/lukewolcott/TDAwithSimulatedData.git.

Introduction

The GitHub repo TDAwithSimulatedData.git contains R scripts that generate simulated data sets and compute their persistence barcodes using the "TDA" package. Each is a variation on a theme: delayed oscillation. If two variables X and Y oscillate over time with a phase shift – the classic example being predator and prey populations – then in XY-space they trace a circle. If a third variable Z affects the amplitude of these oscillations, then in XYZ-space the data will make a shape that reflects this relationship. Adding additional variables pushes the data set into higher dimensions.

Topological data analysis is an excellent tool for understanding the high-dimensional shape of such a data set. Persistent homology measures the "shape" of the data set on different scales. If the data forms a bubble, or a cylinder, or a cone (or high-dimensional analogs of these) this will be detected in an output of the persistence algorithm: the barcode.

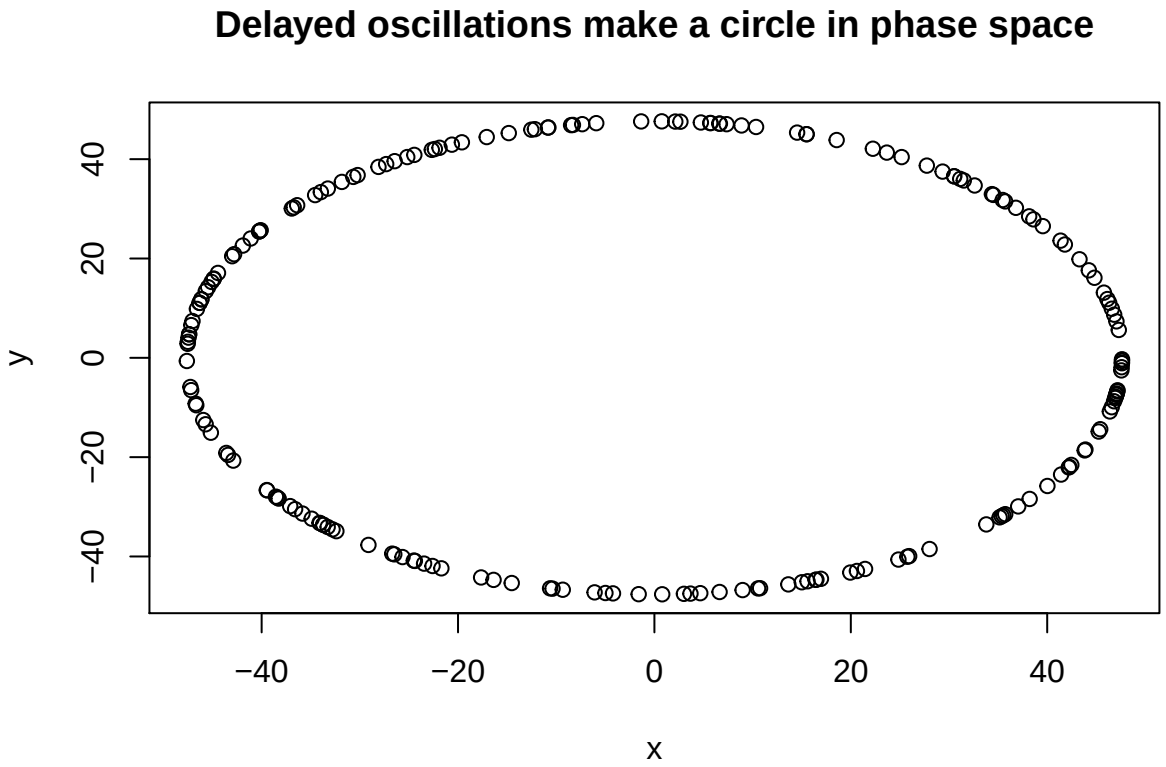
In this Markdown file we will work through several examples of such data sets and their barcodes. The goal is to show how the barcode detects relationships between variables, in the presence of delayed oscillations. The first three examples are in XYZ-space, and the last example adds a fourth parameter. We organize the examples based on how Z affects the amplitude of oscillation, and finish with some concluding comments.

1. bell_bubble: Z has a sweet spot

This example corresponds to bell_bubble.R. Imagine that variables X and Y form a system of delayed oscillation, with magnitude that depends on a third variable Z. This parameter Z has an optimum value where oscillations are at a maximum, and to either side of this the oscillations die off to zero. For example, Z could be temperature in a predator/prey system: too hot or too cold and everything dies. Or Z could be oxygen level in an aquatic ecosystem.

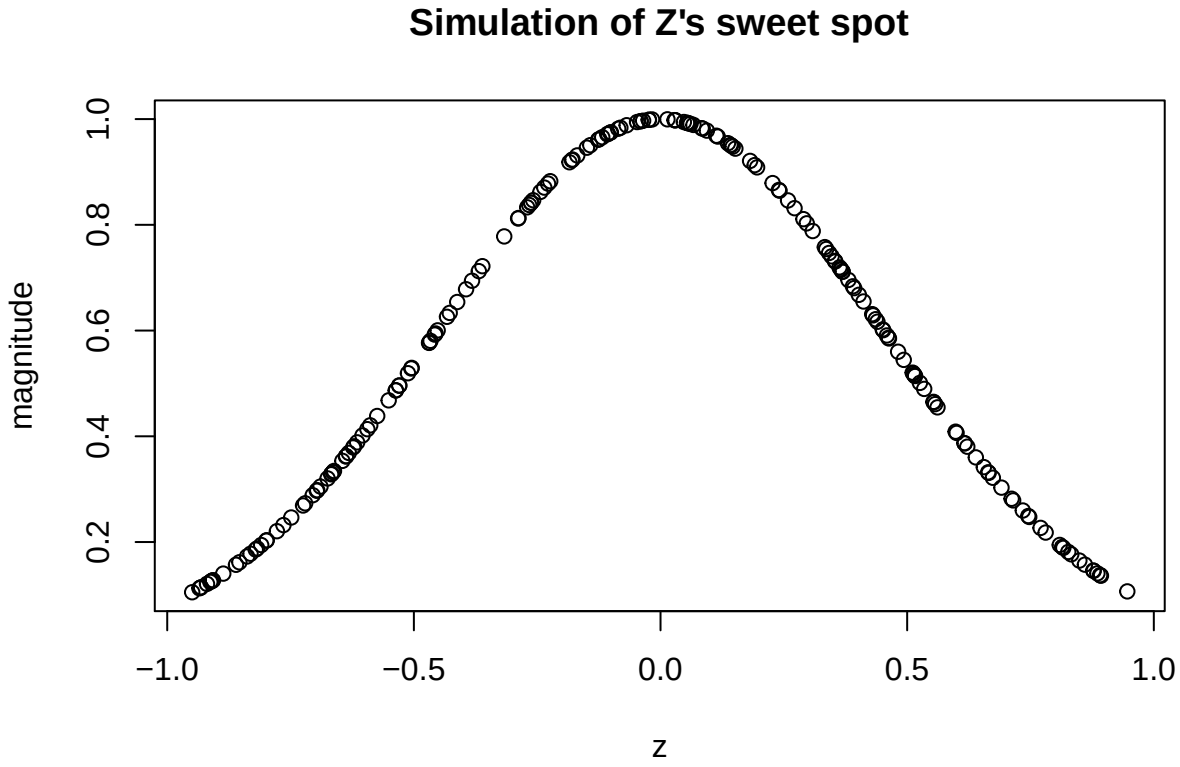
At the optimum Z value the data might fit into the XY-space as follows.

```
set.seed(137)
t <- runif(200)
x <- 100*sin(2*pi*t)/2.1
y <- 100*cos(2*pi*t)/2.1
plot(x,y,main="Delayed oscillations make a circle in phase space")
```



For simplicity we assume the optimum Z-value is at zero, and the magnitude dies off as a bell curve.

```
z <- runif(200, -0.95, 0.95)
magnitude <- exp(-(z^2)/.4)
plot(z,magnitude,main="Simulation of Z's sweet spot")
```

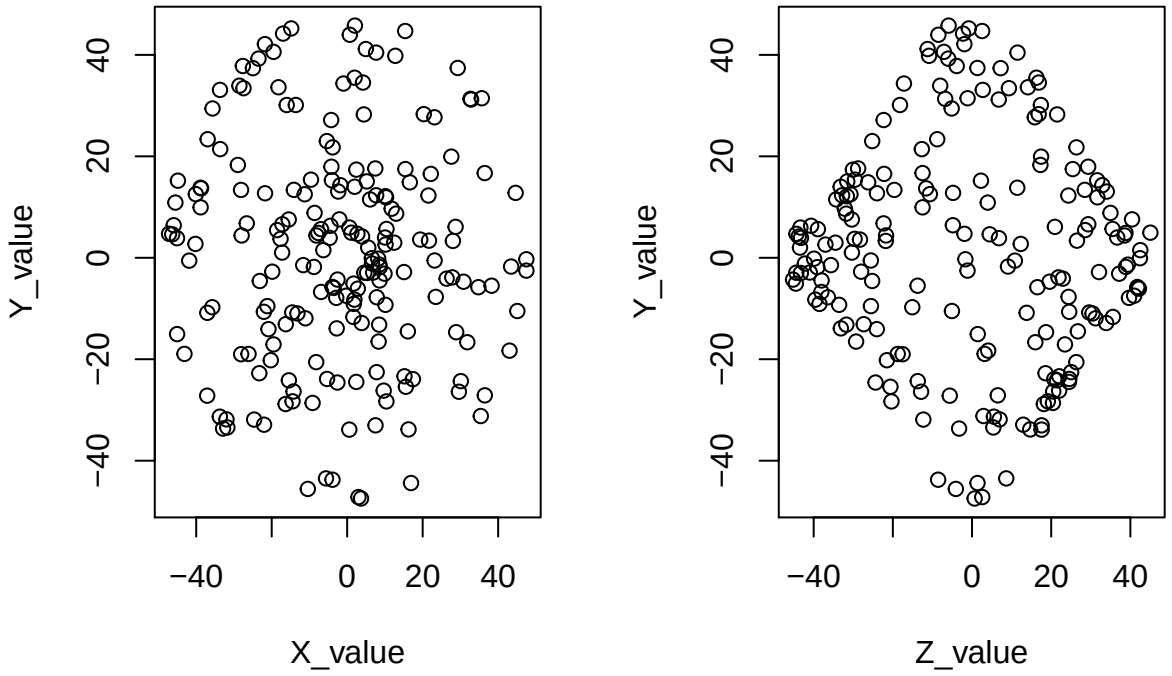


The following code generates a data set of 200 records that might arise from measuring this system. It is a 200 by 3 data frame with columns "X_value", "Y_value", and "Z_value". All variables are scaled to stay between -50 and 50.

```
x <- magnitude*100*sin(2*pi*t)/2.1
y <- magnitude*100*cos(2*pi*t)/2.1
w <- cbind(x,y,100*z/2.1)
d <- as.data.frame((w))
names(d) <- c("X_value", "Y_value", "Z_value")
```

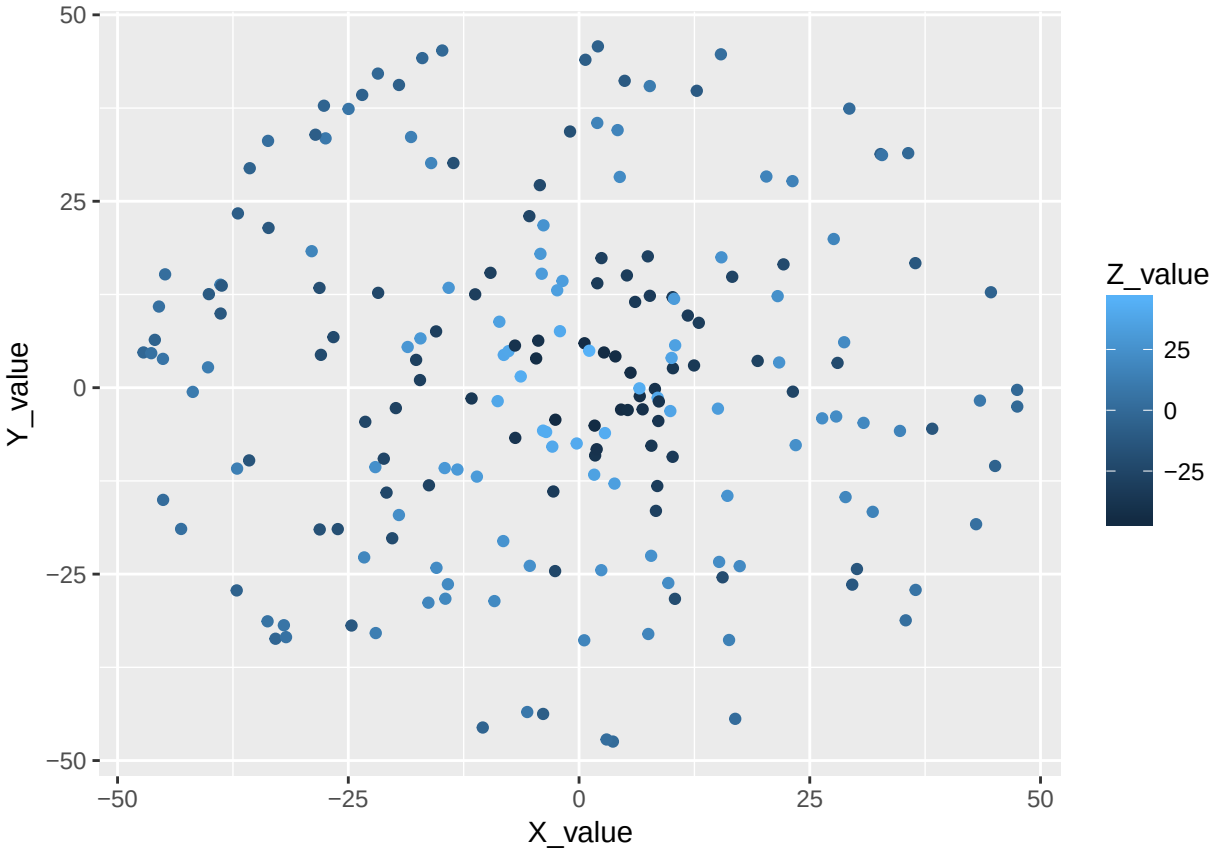
Plotting the Y_value versus X_value gives a circular mess. Plotting, say, the Y_value versus Z_value shows a bell-curved mess.

```
par(mfrow = c(1,2))
plot(d$X_value, d$Y_value,xlab="X_value",ylab="Y_value")
plot(d$Z_value, d$Y_value,xlab="Z_value",ylab="Y_value")
```



The key is that really this data lives in three dimensions, and in fact makes a nice bubble in three dimensions. We can get a sense of this if we color the points according to their Z_value. Notice that the darkest and lightest points are in the center, and the middle-blue points are towards the outside.

```
library(ggplot2)
qplot(X_value, Y_value, data=d, color=Z_value)
```



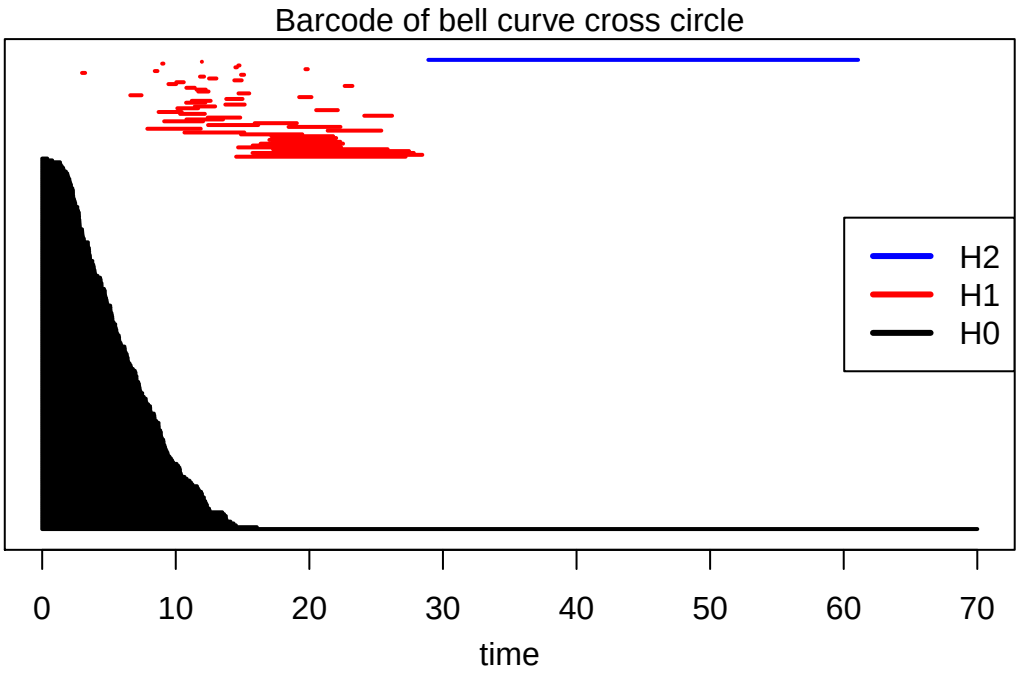
We can plot this (and move it around!) in 3D using the plotly package. (This only works if viewing in HTML, not PDF.)

```
library(plotly)
plot_ly(x = d$X_value, y = d$Y_value, z=d$Z_value, type="scatter3d", mode="markers", color = d$Z_value)
```

The barcode

But a better way to detect this bubble in 3D is to use topological data analysis. We will plot the barcode and then explain what it tells us.

```
library("TDA")
maxscale <- 70
maxdimension <- 2
Diag <- ripsDiag(X = d, maxdimension, maxscale, library = "GUDHI")
plot(Diag[["diagram"]], barcode = TRUE)
mtext("Barcode of bell curve cross circle")
legend("right", lty=c(1,1,1), lwd=c(3,3,3), col=c("blue", "red", "black"), legend=c( "H2", "H1", "H0"))
```



In this case we're mostly interested in the H1 and H2 elements. The horizontal axis is labeled "time", but this is a misnomer that the "TDA" package gives and we can't seem to change. Really this is a filtration parameter for the Vietoris-Rips complex generated by the point cloud, but we won't explain this here.

Each line of red H1 corresponds to a 1-sphere, i.e. a circle, that can be made from the points in the data set. Each line of blue H2 corresponds to a 2-sphere, i.e. a 2D bubble, like the shape formed by the surface of a globe.

Our barcode shows us a long blue H2 line, and this is indicating the data forms a 2D bubble, as expected.

Topology and topological data analysis are best at studying global or qualitative properties of a shape. For example, "How many holes, of which dimensions, does the shape have?" It is not the right tool for studying the precise shape of the magnitude curve, just that it generally has a "sweet spot" shape. Replacing the bell curve with a similar curve, for example the upper half of a circle, would yield more-or-less the same barcode.

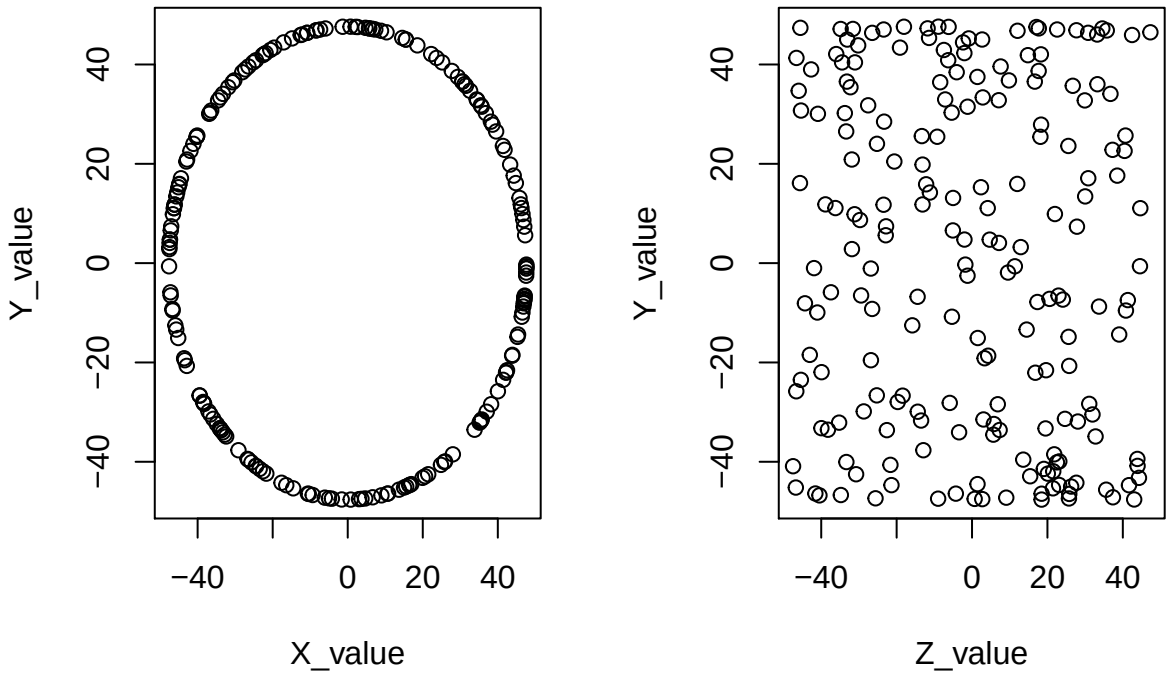
2. cyl_bubble: Z is unrelated

The next three examples will go a little quicker. This one corresponds to cyl_bubble.R. Suppose that our third parameter Z is actually unrelated to the amplitude of oscillation of X and Y. We will assume that Z is uniformly distributed between -50 and 50, but the result would be the same if its distribution were different.

```
x <- 100*sin(2*pi*t)/2.1
y <- 100*cos(2*pi*t)/2.1
w <- cbind(x,y,50*z)
d <- as.data.frame((w))
names(d) <- c("X_value", "Y_value", "Z_value")
```

Plotting Y_value versus X_value, we are looking along the Z-axis and see our circle again. Plotting Y_value versus Z_value we look along the X-axis and don't see much.

```
par(mfrow=c(1,2))
plot(d$X_value,d$Y_value,xlab="X_value",ylab="Y_value")
plot(d$Z_value,d$Y_value,xlab="Z_value",ylab="Y_value")
```

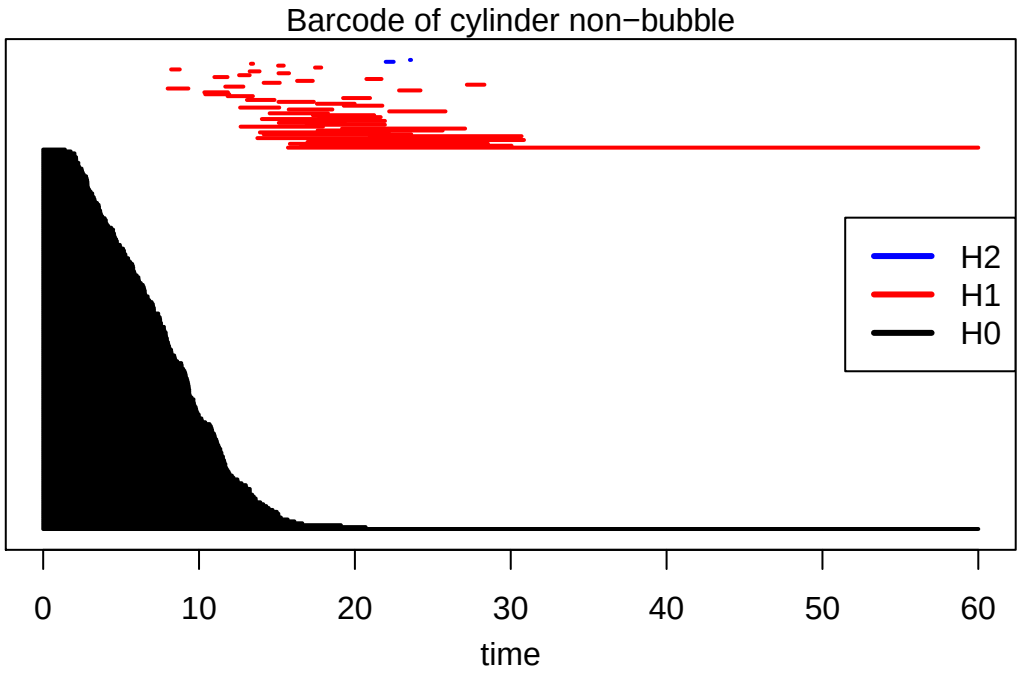


Again, we can plot this in 3D using the plotly package.

```
library(plotly)
plot_ly(x = d$X_value, y = d$Y_value, z=d$Z_value, type="scatter3d", mode="markers", color = d$Z_value)
```

The barcode

```
maxscale <- 60
maxdimension <- 2
Diag <- ripsDiag(X = d, maxdimension, maxscale, library = "GUDHI")
plot(Diag[["diagram"]], barcode = TRUE)
mtext("Barcode of cylinder non-bubble")
legend("right", lty=c(1,1,1), lwd=c(3,3,3), col=c("blue", "red", "black"), legend=c( "H2", "H1", "H0"))
```



The shape of this data set is a cylinder along the Z-axis. The persistence barcode now shows a dominant red H1 line, corresponding to the circle going around the cylinder. There are no significant blue H2 lines, which indicates no 2-spheres (soccer balls).

3. cone_bubble: Z grows from zero without bound

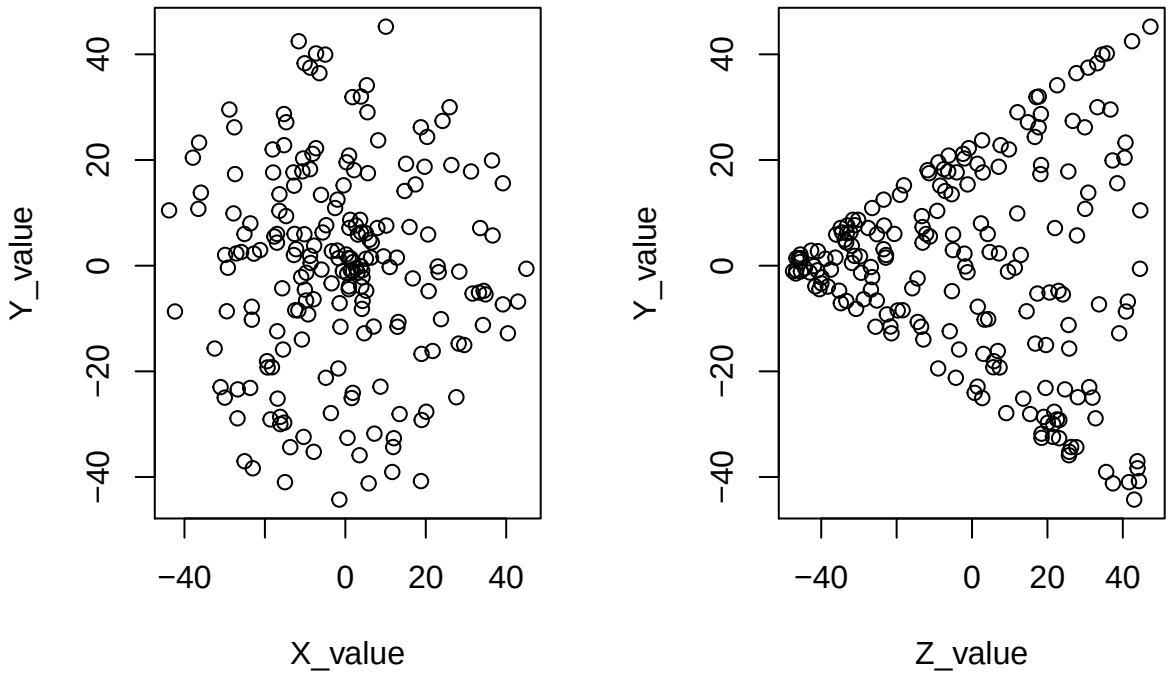
This next example corresponds to cone_bubble.R. Here we imagine that the magnitude of oscillations is determined by a variable Z, which grows from zero without bound. The exact growth curve doesn't matter; it could be linear, or perhaps exponential. For example, population oscillations might be a function of another food resource, such that more food means larger oscillations.

In our simulated data, we normalize so that the magnitude is zero around Z = -50, and grows linearly to 50 around Z=50.

```
x <- (z/2+1/2)*sin(2*pi*t)/2.1
y <- (z/2+1/2)*cos(2*pi*t)/2.1
w <- cbind(100*x,100*y,50*z)
d <- as.data.frame((w))
names(d) <- c("X_value", "Y_value", "Z_value")
```

Plotting Y_value versus X_value again gives a circular mess. But plotting Y_value versus Z_value we look at the data from the side, and see the outline of a cone.

```
par(mfrow=c(1,2))
plot(d$X_value,d$Y_value,xlab="X_value",ylab="Y_value")
plot(d$Z_value,d$Y_value,xlab="Z_value",ylab="Y_value")
```

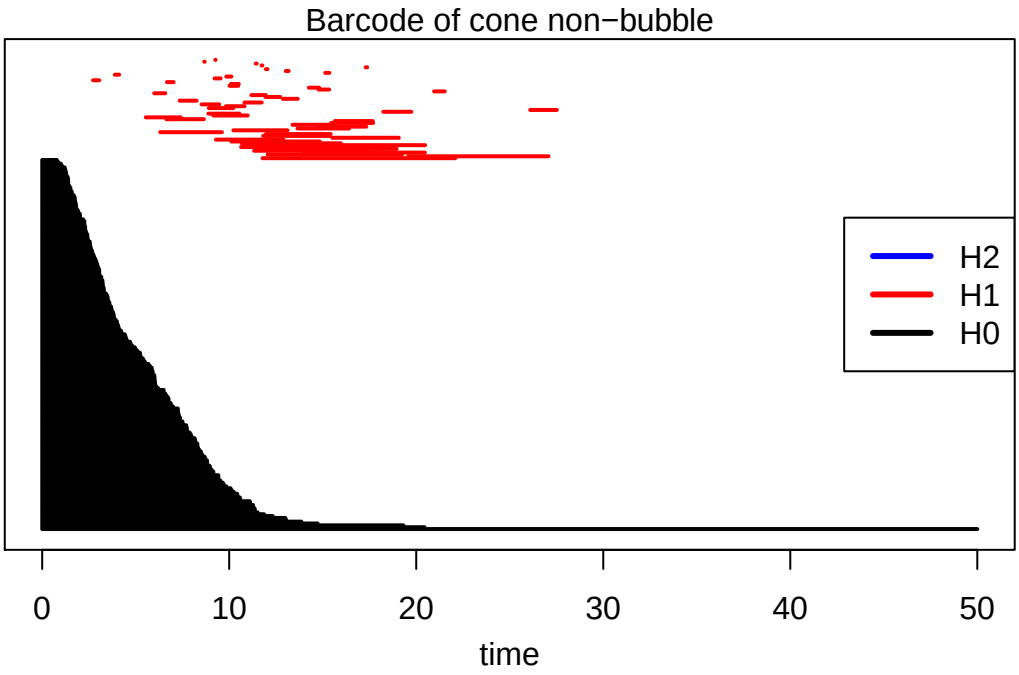


And the 3D plot.

```
library(plotly)
plot_ly(x = d$X_value, y = d$Y_value, z=d$Z_value, type="scatter3d", mode="markers", color = d$Z_value)
```

The barcode

```
maxscale <- 50
maxdimension <- 2
Diag <- ripsDiag(X = d, maxdimension, maxscale, library = "GUDHI")
plot(Diag[["diagram"]], barcode = TRUE)
mtext("Barcode of cone non-bubble")
legend("right", lty=c(1,1,1), lwd=c(3,3,3), col=c("blue", "red", "black"), legend=c( "H2", "H1", "H0"))
```



This cone doesn't have any 2-sphere bubbles, so there are no significant blue H2 lines. It also doesn't have any dominant circle (i.e. 1-sphere) structures, and you can see that all the red H1 lines are relatively short.

4. 4d_bell_bubble: Two sweet spot variables

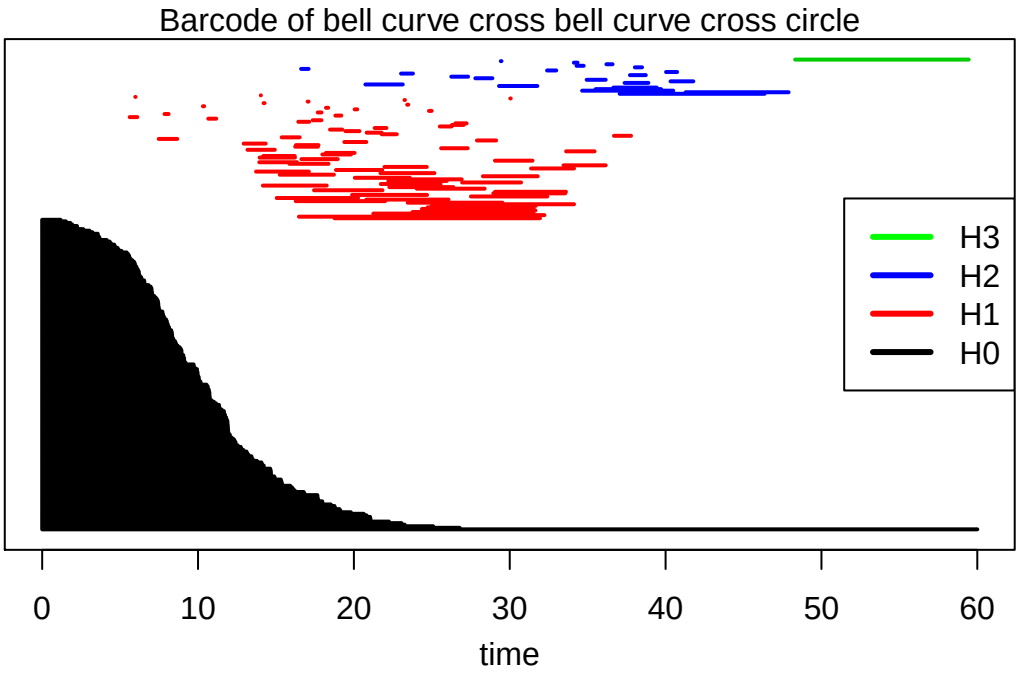
Now we go up one dimension, and show that the same analysis applies with only small changes. Imagine we are back in the first example: two variables X and Y that are in a delayed oscillation, and a third variable Z1 that tracks with the magnitude of oscillation and has a "sweet spot" structure. Suppose that there is a fourth variable Z2 that also affects the magnitude, and lets say it has a "sweet spot" affect.

Perhaps, for example, our predator and prey populations thrive at a particular temperature and water acidity, survive within some range of both, and die outside this range. We measure population values over the course of time, keeping track of both temperature and acidity. The result might be a 200 by 4 data frame as follows.

```
r <- runif(200, -0.95, 0.95)
x <- exp(-(r^2)/.4)*exp(-(z^2)/.4)*sin(2*pi*t)/2.1
y <- exp(-(r^2)/.4)*exp(-(z^2)/.4)*cos(2*pi*t)/2.1
w <- cbind(100*x,100*y,50*z, 50*r)
d <- as.data.frame((w))
names(d) <- c("X_value", "Y_value", "Z1_value", "Z2_value")
```

The barcode

```
maxscale <- 60
maxdimension <- 3
Diag <- ripsDiag(X = d, maxdimension, maxscale, library = "GUDHI")
plot(Diag[["diagram"]], barcode = TRUE)
mtext("Barcode of bell curve cross bell curve cross circle")
legend("right", lty=c(1,1,1,1), lwd=c(3,3,3,3), col=c("green", "blue", "red", "black"), legend=c("H3",
```



This barcode looks similar to the first example, bell_bubble. However, the long blue H2 feature of bell_bubble has become a long green H3 line here. The data is in the shape of a 3D bubble, living in four-dimensional space.

Closing comment: TDA as tool for data insight

In these examples, differently shaped data sets presented qualitatively different barcodes. Significant theoretical work has been done on "stability theorems", which more-or-less allow us to work backwards and use persistence barcodes as a diagnostic tool. For example, if one were to run the persistence algorithm on a 3D data set and generate a barcode that looks like the one in our first example, this would indicate that the data has the shape of a bubble. If the barcode looked as in the second example, the data likely has the shape of a cylinder. Or if the barcode had no long H1 or H2 lines, one would have evidence that the data might be related as in the third example.

Furthermore, adding parameters to make higher dimensional data sets does not add much complexity. The barcode can simultaneously detect multiple relationships among multiple parameters.

As mentioned above, topology ignores many details about specific geometries. This can be a weakness, since for example the barcode of our cone is not so easily distinguished from the barcode of a normally distributed 3D blob. But when used in the right context, these algorithms can indeed detect different classes of relationships.