Smoothing out the Noise

Fred Kaesmann

5/23/2020

```
library(FredsVietorisRips)
Load_FVR_Dependendencies()
Source_FVR_py(notebook = TRUE)
```

Data

We begin with a set of points, $\mathbb{X}$, and we would like to determine the number of holes present via persistent homology.

```
##### Import Data ####
df_original <- read.csv("../data/Clark_Sample_data.csv", col.names = c("x", "y"))
```

Initial Condiditons

```
##### Trials and Samples ####
# How many points are we sampling
#   and how many times are we sampling
sample_size <- 1000
ntrials <- 7

##### Size and Overlap of Pullback Covers ####
# What is the length of our pullback cover
#    and what is the overlap between them?
resolution <- 0.05
gain <- 0.75

##### Connection Distance and Cluster Requirements ####
# Points within distance epsilon are connected
#   and  only points with a minimum number of
#   connections are considered valid clusters
epsilon <- 0.1
min_connections <- 3

##### Number of Components per Cover per Trial ####
ncomponents <- vector("list", length = ntrials)
```

We run 7 iterations where we sample 1000 points from our original point cloud. Each cover beginning at 0 has resolution, or length, 0.05 and they overlap 75%. As we move up the y-axis we allocate to each cover

the points

$$\text{Cover}_j = \{(x_i, y_i) | y_i \in [y_j, y_j + r)\}$$

where y_j is the lower bound of our cover, and $y_j + r$ is its upper bound.

Run Trials

When we run each trial we process the data as normal, the only difference is that we create a list of list which contains the number of components in each cover for each of our trials. We use this list of lists to determine the average number of components in each cover and plot that.

```
for (trial in c(1:ntrials)) {
  ##### Sample Original Data ####
  df <- df_original %>%
    sample_n(sample_size)

  # Make it compatible with Pyton, start with zero
  rownames(df) <- as.numeric(rownames(df)) - 1

  ##### Filter ####
  list_of_covers <- PullBackCovers(df$x, df$y, resolution, gain)

  ##### How many components do we have? #####
  components <- FindComponents(list_of_covers, epsilon, min_connections, notebook = TRUE)

  ##### nComponents per Cover per Trial ####
  ncomponents[[trial]] <- components %>%
    lapply(length) %>%
    unlist()
}
```

Below we turn our list of lists into a table to show the minor variations in the number of components in each cover.

```
##### Matrix of Trial Results ####
trialMatrix <- matrix(unlist(ncomponents),
                      ncol = ntrials)
dimnames(trialMatrix) <- list(
  paste("Cover", c(1:nrow(trialMatrix))),
  paste("Trial", c(1:ntrials))
)

##### Average Components by Cover ####
averageComponents <- trialMatrix %>%
  rowSums() / ntrials
```

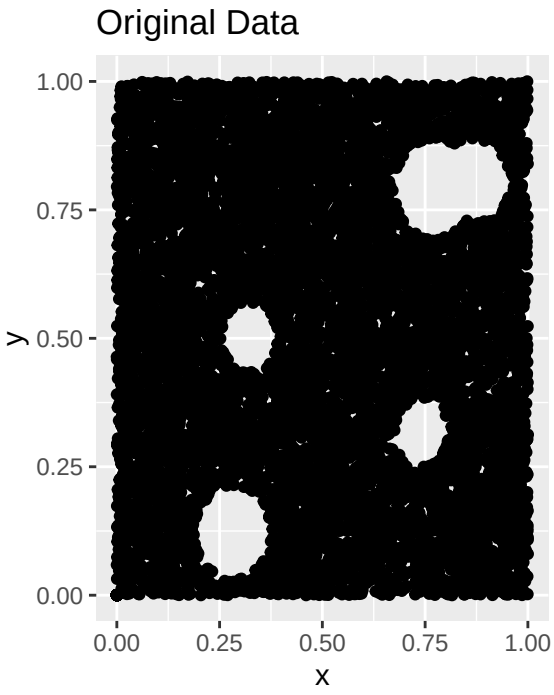
Table 1: Components in each Cover in each Trial

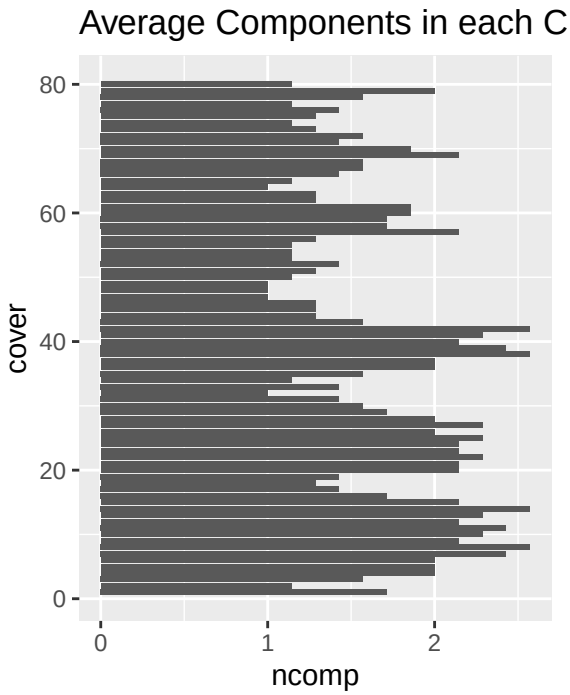
	Trial 1	Trial 2	Trial 3	Trial 4	Trial 5	Trial 6	Trial 7	averageComponents
Cover 1	1	1	3	1	2	2	2	1.714286
Cover 2	1	1	2	1	1	1	1	1.142857
Cover 3	2	1	2	2	1	2	1	1.571429

	Trial 1	Trial 2	Trial 3	Trial 4	Trial 5	Trial 6	Trial 7	averageComponents
Cover 4	2	2	2	2	2	2	2	2.000000
Cover 5	2	2	2	2	2	2	2	2.000000
Cover 6	2	2	2	2	2	2	2	2.000000
Cover 7	3	2	2	3	2	3	2	2.428571
Cover 8	3	2	3	3	2	3	2	2.571429
Cover 9	2	2	3	2	2	2	2	2.142857
Cover 10	2	2	2	2	3	2	3	2.285714
Cover 11	2	3	2	2	3	2	3	2.428571
Cover 12	2	2	2	2	3	2	2	2.142857
Cover 13	2	2	2	2	3	3	2	2.285714
Cover 14	2	3	2	2	4	3	2	2.571429
Cover 15	2	1	3	2	2	2	3	2.142857
Cover 16	2	1	2	2	1	1	3	1.714286
Cover 17	2	1	1	2	1	1	2	1.428571
Cover 18	2	1	1	2	1	1	1	1.285714
Cover 19	2	1	1	2	2	1	1	1.428571
Cover 20	3	1	2	3	3	2	1	2.142857
Cover 21	3	1	2	2	3	2	2	2.142857
Cover 22	2	2	2	2	3	2	3	2.285714
Cover 23	2	2	2	2	2	2	3	2.142857
Cover 24	2	2	2	2	2	2	3	2.142857
Cover 25	3	2	2	2	2	2	3	2.285714
Cover 26	3	2	2	2	2	1	2	2.000000
Cover 27	3	3	2	2	2	2	2	2.285714
Cover 28	3	2	1	2	2	2	2	2.000000
Cover 29	2	2	1	2	2	1	2	1.714286
Cover 30	2	1	2	2	1	1	2	1.571429
Cover 31	2	1	1	2	1	1	2	1.428571
Cover 32	1	1	1	1	1	1	1	1.000000
Cover 33	2	1	2	1	1	1	2	1.428571
Cover 34	1	1	2	1	1	1	1	1.142857
Cover 35	1	2	2	1	2	2	1	1.571429
Cover 36	2	2	2	2	2	2	2	2.000000
Cover 37	2	2	2	2	2	2	2	2.000000
Cover 38	2	2	3	2	3	3	3	2.571429
Cover 39	2	2	3	2	3	3	2	2.428571
Cover 40	2	2	3	2	2	2	2	2.142857
Cover 41	3	2	3	2	2	2	2	2.285714
Cover 42	2	3	3	3	2	2	3	2.571429
Cover 43	1	2	2	2	1	2	1	1.571429
Cover 44	1	2	1	2	1	1	1	1.285714
Cover 45	1	2	2	1	1	1	1	1.285714
Cover 46	1	2	1	1	1	1	2	1.285714
Cover 47	1	1	1	1	1	1	1	1.000000
Cover 48	1	1	1	1	1	1	1	1.000000
Cover 49	1	1	1	1	1	1	1	1.000000
Cover 50	1	1	1	2	1	1	1	1.142857
Cover 51	2	2	1	1	1	1	1	1.285714
Cover 52	3	2	1	1	1	1	1	1.428571
Cover 53	2	1	1	1	1	1	1	1.142857
Cover 54	2	1	1	1	1	1	1	1.142857
Cover 55	2	1	1	1	1	1	1	1.142857

	Trial 1	Trial 2	Trial 3	Trial 4	Trial 5	Trial 6	Trial 7	averageComponents
Cover 56	2	1	1	1	2	1	1	1.285714
Cover 57	2	2	2	2	3	2	2	2.142857
Cover 58	2	1	2	2	2	2	1	1.714286
Cover 59	1	1	2	2	2	3	1	1.714286
Cover 60	2	1	2	2	2	3	1	1.857143
Cover 61	2	1	2	1	2	4	1	1.857143
Cover 62	1	1	2	1	1	2	1	1.285714
Cover 63	1	1	2	1	1	2	1	1.285714
Cover 64	1	1	1	1	1	1	1	1.000000
Cover 65	1	1	1	2	1	1	1	1.142857
Cover 66	3	1	1	2	1	1	1	1.428571
Cover 67	2	1	1	3	2	1	1	1.571429
Cover 68	2	1	1	2	3	1	1	1.571429
Cover 69	3	2	3	3	2	1	1	2.142857
Cover 70	1	2	2	3	3	1	1	1.857143
Cover 71	2	2	1	2	1	1	1	1.428571
Cover 72	3	2	1	2	1	1	1	1.571429
Cover 73	1	1	2	2	1	1	1	1.285714
Cover 74	1	1	1	1	2	1	1	1.142857
Cover 75	1	2	2	1	1	1	1	1.285714
Cover 76	1	2	3	1	1	1	1	1.428571
Cover 77	1	1	2	1	1	1	1	1.142857
Cover 78	2	2	1	2	1	2	1	1.571429
Cover 79	3	2	1	3	1	2	2	2.000000
Cover 80	1	0	2	3	0	2	0	1.142857

Visualization of our Data and our Components





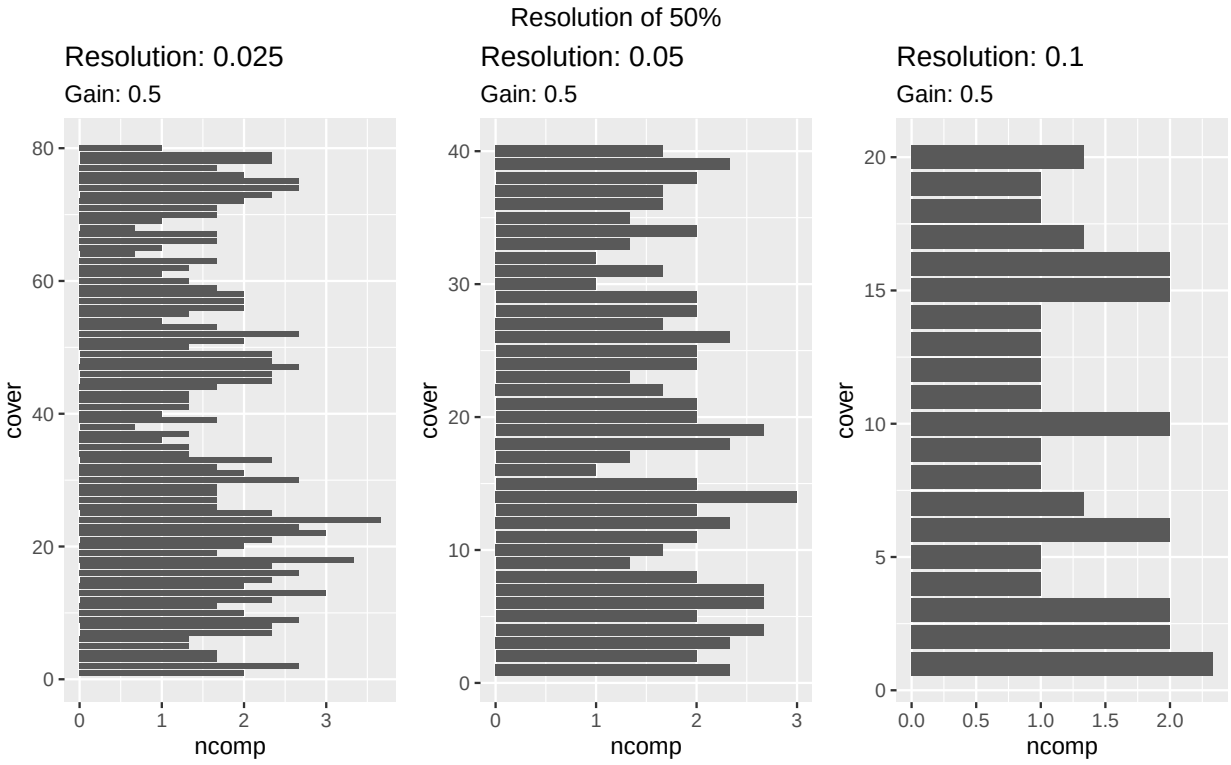
Some Additional Thoughts

Our above bar chart shows that our initial conditions appear to be well chosen to represent our specific data. This is not the case for what appears to be even minor changes in those initial conditions.

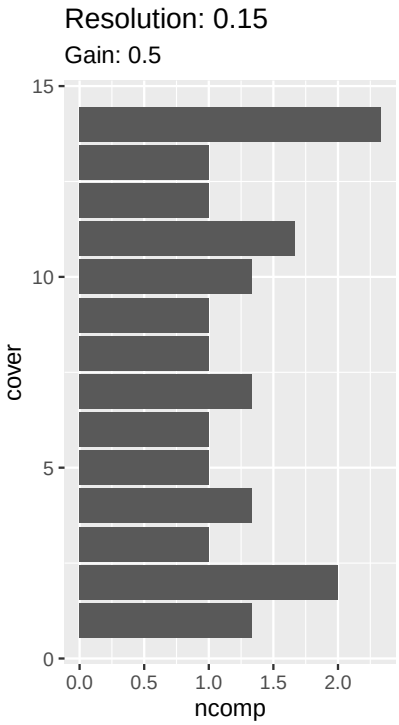
```
##### Constant Conditions ####
sample_size <- 750
ntrials <- 3
min_connections <- 3
epsilon <- 0.1



##### Varrying Condidtions
resolutionSet <- c(0.25, 0.2, 0.15, 0.1, 0.05, 0.025)
gainSet <- c(0.5, 0.75)

##### Number of Components per Cover per Trial ####
ncomponents <- vector("list", length = ntrials)
```

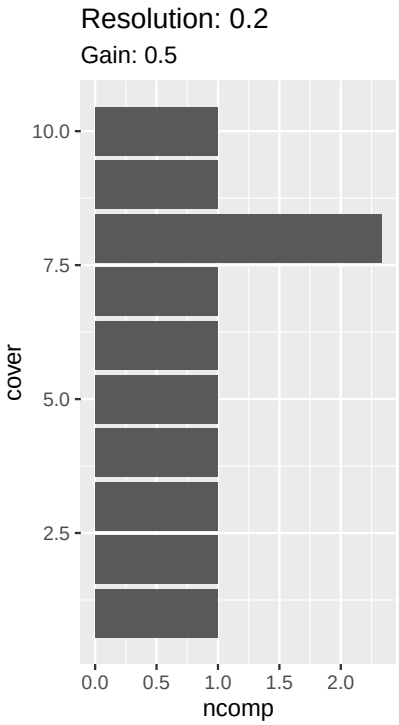
```
for (resolution in resolutionSet) {
  for (gain in gainSet) {
    for (trial in c(1:ntrials)) {
      ##### Sample Original Data ####
      df <- df_original %>%
        sample_n(sample_size)

      # Make it compatible with Pyton, start with zero
      rownames(df) <- as.numeric(rownames(df)) - 1

      ##### Filter ####
      list_of_covers <- PullBackCovers(df$x, df$y, resolution, gain)

      ##### How many components do we have? #####
      components <- FindComponents(list_of_covers, epsilon, min_connections, notebook = TRUE)

      ##### nComponents per Cover per Trial ####
      ncomponents[[trial]] <- components %>%
        lapply(length) %>%
        unlist()
    }
    ##### Matrix of Trial Results ####
    trialMatrix <- matrix(unlist(ncomponents),
                          ncol = ntrials)
    ##### Average Components by Cover ####
    averageComponents <- trialMatrix %>%
    rowSums() / ntrials


    chart <- data.frame(
      cover = c(1:nrow(trialMatrix)),
      ncomp = averageComponents) %>%
      ggplot(aes(x = cover,
```

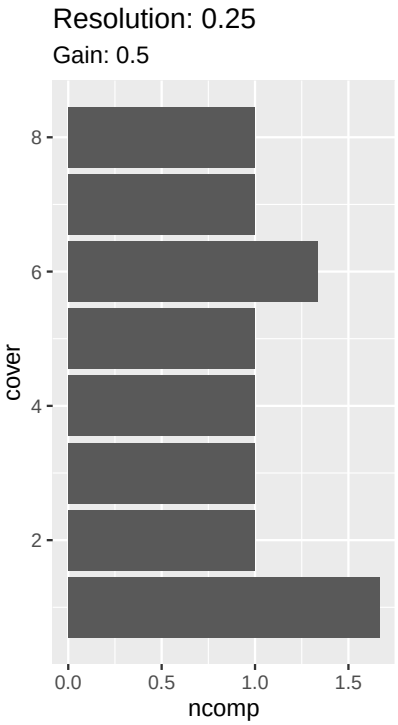
```
                  y = ncomp)) +
      geom_col() +
      coord_flip() +
      ggtitle(paste("Resolution:", resolution),
              paste("Gain:", gain))

    assign(paste0("c", resolution,gain), chart)
  }
}
```



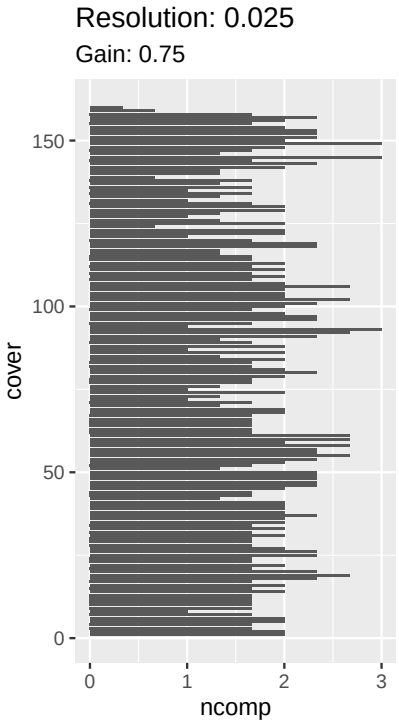
Resolution of 50%

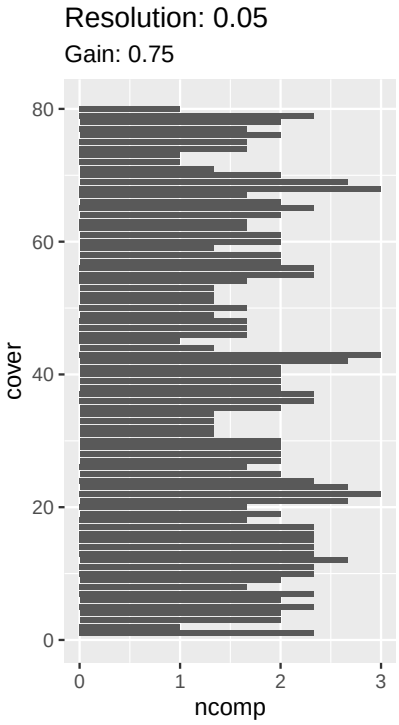


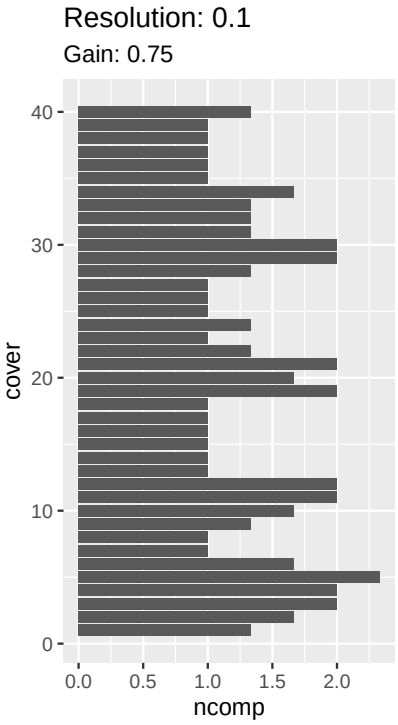




Resolution of 75%







Resolution of 75%