Clustering Example 7: Network Clustering of Dolphin Data

Amy Wagaman, Nathan Carter

July 2020

Load necessary libraries.

```
library(igraph)
library(ape)
```

This requires you to have access to the `dolphins.gml` file. It is available in the book's GitHub repository at the following URL.

https://github.com/ds4m/ds4m.github.io/tree/master/chapter-5-resources/dolphins.gml

If you run this R code, place the data file in the same folder as the code file.

How many vertices and edges does the graph have?

```
g <- read_graph("dolphins.gml", format = "gml")
vcount(g)
```

```
## [1] 62
```

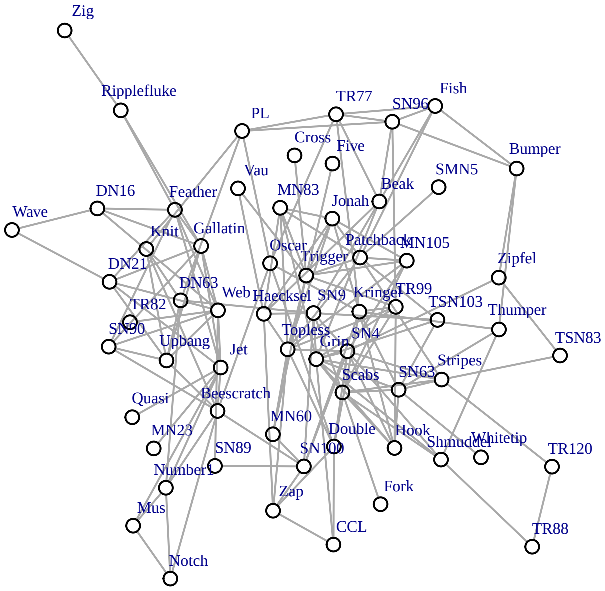
```
ecount(g)
```

```
## [1] 159
```

Plot the graph

Some layout algorithms use randomness, so we set a seed, for reproducibility.

```
set.seed(540)
mylayout3 <- layout_(g, with_gem())
igraph.options(vertex.size=5, vertex.color = "NA", vertex.shape = "circle",
               vertex.label.cex = 0.5, vertex.label.dist = 1.3)
plot(g, layout = mylayout3)
```



We can ask questions about network density.

```
degree(g)
```

```
##  [1]  6  8  4  3  1  4  6  5  6  7  5  1  1  8 12  7  6  9  7  4  9  6  1  3  6
## [26]  3  3  5  5  9  5  1  3 10  5  1  7 11  8  2  8  5  6  7  4 11  2  6  1  2
## [51]  7 10  4  2  7  2  2  9  1  5  1  3
```

```
summary(degree(g))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   3.000   5.000   5.129   7.000  12.000
```

Apply the `fastgreedy.community` clustering algorithm.

```
set.seed(240)
fg1 <- fastgreedy.community(g)
length(fg1) # How many clusters?
```

```
## [1] 4
```

```
sizes(fg1) # How large was each cluster?
```

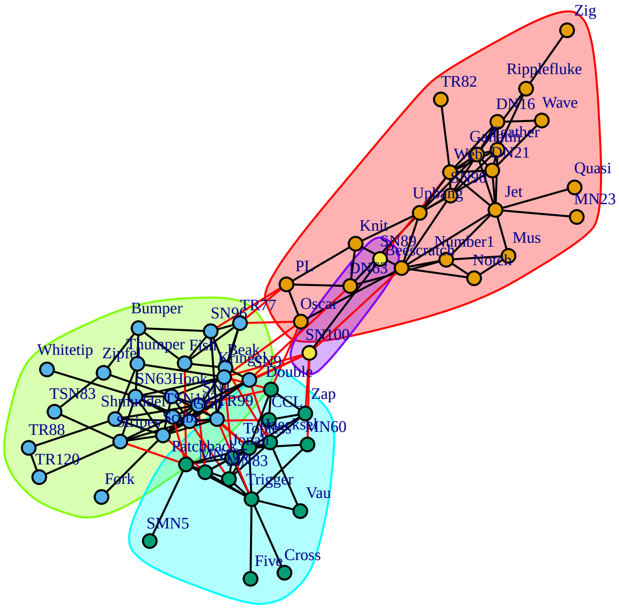
```
## Community sizes
##  1  2  3  4
## 22 23 15  2
```

```
membership(fg1) # the familiar clustering vector
```

```
##  [1] 2 1 2 3 3 1 1 1 3 1 2 3 2 1 2 3 2 1 3 1 2 3 1 3 3 1 1 1 1 3 1 1 1 2 2 3 4 2
## [39] 2 4 2 1 2 2 2 3 2 2 1 2 2 3 2 2 1 3 1 1 2 3 1 2
```

Plot the network again, with clusters shown.

```
plot(fg1, g)
```



Measure the clustering quality.

The `fastgreedy.community` algorithm seeks to maximize modularity, which is a measure of how separated different vertex types are from one another. We can ask for the modularity of the result.

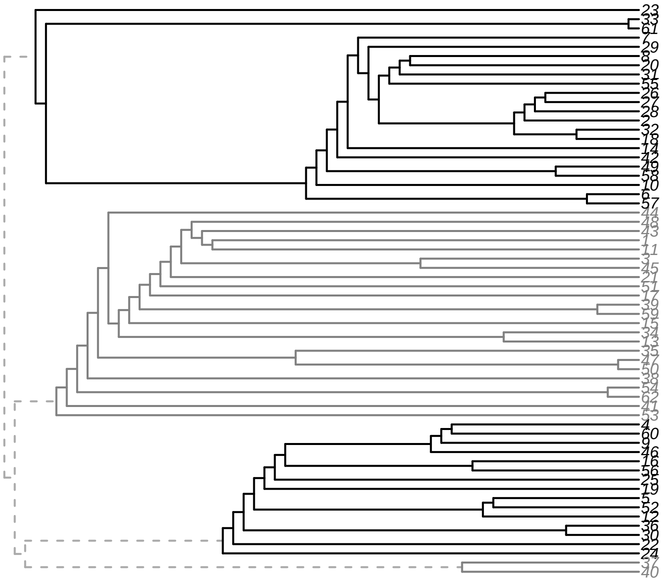
```
modularity(g, fg1$membership)
```

```
## [1] 0.4954907
```

Plot the corresponding dendrogram.

Because the `fastgreedy.community` algorithm is hierarchical, it also produces a dendrogram.

```
dendPlot(fg1, mode="phylo", palette = gray(c(0, 0.5, 0, 0.5)),
         cex = 0.5, label.offset = 0.1)
```



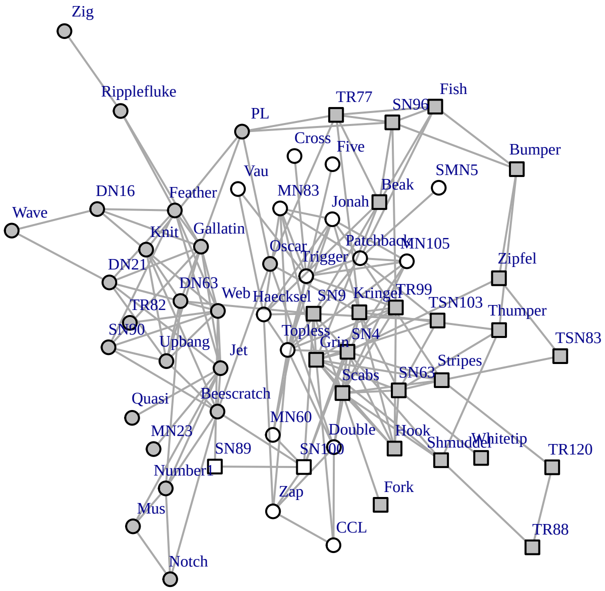
Improve the clustering plot.

The plot that moves vertices near to one another is a bit busy and messy. Let's assign vertex shapes to indicate the clustering solution and plot without the colored blobs shown above.

```
shapes <- rep("circle", 62)
colors <- rep("NA", 62)

for (i in 1:62) {
  if (fg1$membership[i]==2) { shapes[i] = "square" }
  if (fg1$membership[i]==4) { shapes[i] = "square" }
  if (fg1$membership[i]==1) { colors[i] = "gray"   }
  if (fg1$membership[i]==2) { colors[i] = "gray"   }
}

igraph.options(vertex.size=5, vertex.color = colors, vertex.label.cex = 0.5,
               vertex.label.dist = 1.3, vertex.shape = shapes)
plot(g, layout = mylayout3)
```



Apply the `leading.eigenvector.community` clustering algorithm.

```
sp1 <- leading.eigenvector.community(g)
length(sp1) # How many clusters?
```

```
## [1] 5
```

```
sizes(sp1) # How large was each cluster?
```

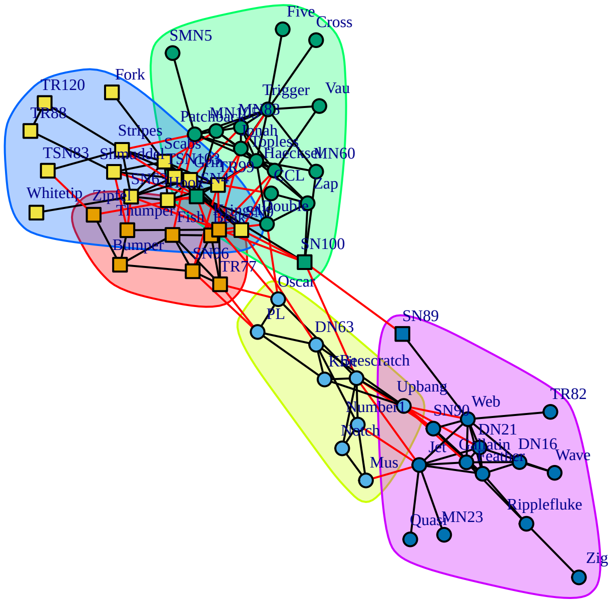
```
## Community sizes
##  1  2  3  4  5
##  8  9 17 14 14
```

```
membership(sp1) # the familiar clustering vector
```

```
##  [1] 1 2 1 3 3 5 5 2 3 5 1 3 4 5 4 3 4 5 3 2 1 3 5 3 3 2 2 2 2 3 2 5 5 4 4 3 3 3
## [39] 4 5 4 5 1 4 1 3 4 1 5 4 4 3 4 4 2 3 5 5 4 3 5 1
```

Plot the network again, with clusters shown.

```
plot(sp1, g)
```



Measure the clustering quality.

See comments above about the `modularity` metric.

```
modularity(g, sp1$membership)
```

```
## [1] 0.4911989
```

Apply the `walktrap.community` clustering algorithm.

```
wt1 <- walktrap.community(g)
length(wt1) # How many clusters?
```

```
## [1] 4
```

```
sizes(wt1) # How large was each cluster?
```

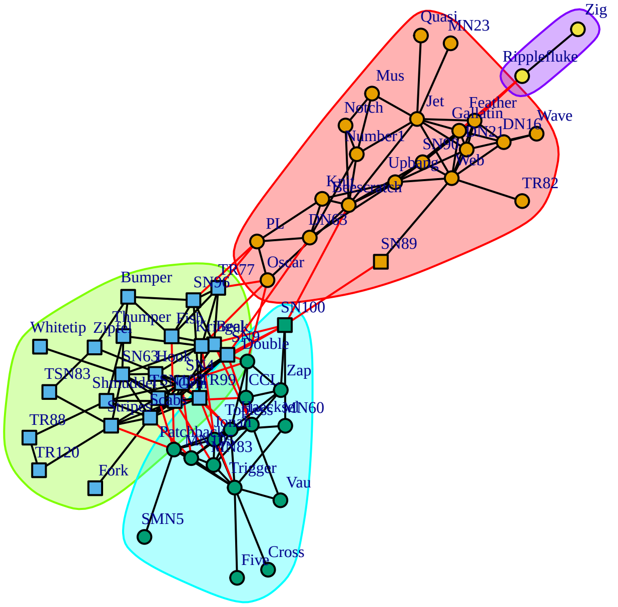
```
## Community sizes
##  1  2  3  4
## 21 23 16  2
```

```
membership(wt1) # the familiar clustering vector
```

```
##  [1] 2 1 2 3 3 1 1 1 3 1 2 3 2 1 2 3 2 1 3 1 2 3 1 3 3 1 1 1 1 3 1 1 4 2 2 3 3 2
## [39] 2 1 2 1 2 2 2 3 2 2 1 2 2 3 2 2 1 3 1 1 2 3 4 2
```

Plot the network again, with clusters shown.

```
plot(wt1, g)
```



Measure the clustering quality.

See comments above about the `modularity` metric.

```
modularity(g, wt1$membership)
```

```
## [1] 0.4888454
```