Probability and Statistics for Network Analysis MT 2023

Second Practical: Ego-networks, sampling, and Monte Carlo tests

1 Ego-networks

An ego-network is a network which is centred around a specific node and contains all its k-neighbours plus their connections. We can use as data set a set of Yeast protein interaction, available at

http://www.stats.ox.ac.uk/~reinert/dtc/networks.html,

First we load the igraph package: From the drop-down menu packages, select "load package", then select *igraph*, or use

```
library(igraph)
```

Now look at the Yeast data.

```
yeast<-read.graph("http://www.stats.ox.ac.uk/~reinert/dtc/YeastL.net", format="pajek")
```

Remove the self-loops in the data

```
yeast<-simplify(yeast, remove.multiple=TRUE, remove.loops=TRUE)
```

Find the nearest neighbours of Vertex 1

```
nb1<-neighbors(yeast, V(yeast)[1])
nb1
```

and create a vector of neighbours which contains Vertex 1 also

```
nb<-c(nb1, V(yeast)[1])
nb
```

Obtain an ego-network for Vertex 1 by using

```
sub<-graph.neighborhood(yeast, 1, nodes=V(yeast)[1])
plot(sub[[1]])
```

Compare ego-networks for two vertices with the same degree: 1983 and 1554 have both degree 28; check:

```
degree(yeast, V(yeast)[1983])
degree(yeast, V(yeast)[1554])
```

Get their ego-networks as above, and plot them

```
Sub1983<-graph.neighborhood(yeast, 1, nodes=V(yeast)[1983])
Sub1554<-graph.neighborhood(yeast, 1, nodes=V(yeast)[1554])
plot(Sub1983[[1]], layout= layout_nicely, vertex.label.cex=0.005)
plot(Sub1554[[1]], layout= layout_nicely, vertex.label.cex=0.005)
```

We can repeat the procedure for second-neighbours of Vertex 1

```
Sub2<-graph.neighborhood(yeast, 2, nodes=V(yeast)[1])
plot(Sub2[[1]] , layout= layout_nicely, vertex.label.cex=0.05)
```

Look at the 2-hop ego network for Vertex 1554

```
Sub21554<-graph.neighborhood(yeast, 2, nodes=V(yeast)[1554])
plot(Sub21554 [[1]], layout= layout_nicely, vertex.size=0.5, vertex.label.cex=0.005)
```

What happens when you increase the hop size further?

2 Subsampling

We may like to look at some samples from a network in more detail.

To sample without replacement, first draw a sample from the vertex set:

```
V<- V(yeast)
Vs<- sample(V, 10, replace = FALSE, prob = NULL)
```

Now obtain the induced subgraph for this sample

```
Vind<-induced.subgraph(yeast, Vs, impl=c("auto"))
```

and plot it

```
plot(Vind, layout= layout_nicely, vertex.label.cex=0.05)
```

How many edges does the induced subgraph have? Find a suitable igraph command.

Obtain a 1-hop snowball sample with Vs as set of seed nodes:

```
snow<-make_ego_graph(
  yeast,
  order = 1,
  nodes = Vs
)
```

The result is a collection of ego-networks. To convert this collection into a single network you can use

```
snow_list_df <- lapply(snow, as_data_frame)
snow_df <- do.call(rbind, snow_list_df)
snowball <- graph_from_data_frame(snow_df , directed = FALSE)
```

There may be multiple edges; to simplify,

```
snowball<-simplify(snowball)
```

How many edges does your snowball sample have?

You can plot it,

```
plot(snowball, layout= layout_nicely, vertex.label="", vertex.size=10)
```

3 Monte Carlo test

Suppose that we would like to base our test on the statistic T_0. We only need to be able to simulate a random sample $T_{01}, T_{02}, \ldots, T_{0n}$ from the distribution, call it F_0, determined by the null hypothesis. We assume that F_0 is continuous, and, without loss of generality, that we reject the null hypothesis H_0 for large values of T_0. Then, provided that $\alpha = m/(n+1)$ is rational, we can proceed as follows.

1. Observe the actual value t* for T_0, calculated from the data
2. Simulate a random sample of size n from F_0
3. Order the set of observed values { t*, $t_{01}, t_{02}, \ldots, t_{0n}$ }
4. Reject H_0 if the rank of t* in this set (in decreasing order) is less or equal to m.

The basis of this test is that, under H_0, the random variable T* has the same distribution as the remainder of the set and so, by symmetry,

P(t* is among the largest m value) =m/(n+1).

Use a network statistic to compare different network models. Here are some questions to consider.

1. What is the effect of choosing different parameters in the models?
2. How do the vertex degree distributions differ in the different models?
3. How does the global clustering coefficient vary across the models?

Choose one of these questions and start your own examination. Pick at least 2 different type of models and 3 different parameter choices. Test whether the statistics are statistically different in the different models using a Monte Carlo test. Be prepared to present your results in 2 min in the problem session. Below is a sample Monte Carlo test code which you can adapt to your problem.

```
p<-ecount(yeast)*2/(vcount(yeast)*(vcount(yeast) - 1))

w<-c(1:49)
for(i in (1:49)){
        N<-sample_gnp(vcount(yeast), p)
        w[i]<-transitivity(N)
        }

v<-c(w,transitivity(yeast))
 sort(v)
match(transitivity(yeast), sort(v))
```