# Ecological Networks
Exercise sheet

## Problem 1: Network Science

### Objectives

You will gain an understanding of the key measures in network science and how to compute them. Moreover, you will learn how to analyze real biological interaction networks with respect to these measures.

(a) Familiarize yourself with Prof. Bascompte's web of life (www.web-of-life.es). What information is available? For this exercise, we use the pollination network "M_PL_052" from southwestern Greenland. Check the R-code in the starting script and try to understand what it does.

(b) Each interaction matrix contains a certain number of rows and columns. In this pollination network, for instance, the rows stand for plants, the columns for pollinators. Let's use R to investigate certain properties of the network.

Since we will need them frequently, store both the number of rows and the number of columns as variables, called *n_row* and *n_col*, respectively. We could look at the interaction matrix in the console by simply typing *data* and hitting Enter, but that's not very nice to look at. Instead, let's create a graphical representation of the interaction matrix using *heatmap*.

Hints:

1. Use *?heatmap* if you have trouble or want to learn about *heatmap*'s different parameters.
2. To get a consistent colormap for the whole matrix set *scale='none'* when calling the heatmap function.
3. You might also need *as.matrix(data)* to transform the data frame into a matrix.
4. If your downloaded interaction matrix is not binary but weighted (i.e., it contains weights greater 1), think of a way to make it binary.

You can also try to visualize the network using different approaches, such as *plotweb(data)*. Feel free to explore further options.

What do you observe? Which species is the most interconnected one? Do you think the matrix is highly nested?

(c) We can also graph the interactions as a network. Use the function *graph_from_adjacency_matrix* from the igraph package to get a network representation of the interaction matrix, which we call *net* and which you can plot.

Hints:

1. You will get an error if you use the original variable *data* since the input of *graph_from_adjacency_matrix* needs to be a square matrix. We need first to create an extended interaction matrix.
2. You can try fixing this error on your own or use the following function. What does each line achieve?

```r
fulladjm <- function(data, n_row, n_col) {
  fulladjm <- matrix(0, n_row + n_col, n_row + n_col)
  fulladjm[(n_row + 1):(n_row + n_col), 1:n_row] = t(as.matrix(data > 0))
  fulladjm[1:n_row, (n_row + 1):(n_row + n_col)] = as.matrix(data > 0)
  return(full_adjm)
}
```

By using

```
adjm <- fulladjm(data, n_row, n_col)
net <- graph_from_adjacency_matrix(adjm, mode = "undirected")
colrs <- c("tomato", "gold")
V(net)$color <- colrs[as.numeric(V(net) > n_row) + 1]
V(net)$label <- append(row.names(data), colnames(data))
```

we can color rows and columns differently. You can now play around with the graph and try to display various features. Have a look at http://kateto.net/network-visualization (Section 4) for ideas of what you can do.

(d) Let's now investigate different properties of our network. Using the R functions *rowSums* and *colSums*, we can look at the degree distribution of rows and columns. Compute and plot the degree distribution for the rows and the columns of the matrix. Think of an appropriate way to visualize the degree distribution.

Hints:

1. One way to solve this problem is to compute each degree distribution by writing a function that takes as input a matrix and outputs the resepective degree distribution of the matrix. The first entry of this vector would correspond to the number of rows/columns that have only one connection.

Are there any 'hubs' in either distribution? What does this mean for biology, which species would you be most worried about going extinct? Which has the potentially largest cascade effect?

(e) We can also look at the 'density', which is defined as the ratio of interactions over all possible interactions. Write a function called *density* that takes as input a matrix of any size and outputs the density of the matrix.

How dense is your interaction matrix? Do you think extinction of one species has a bigger impact in a matrix of high or low density? Can we even make such general statements?

Can you think of reasons why density is biologically not perfectly suited to compare different networks?

(f) The nestedness is a more complicated function. We will rely on the implementation of the library *bipartite*. Write a function that takes as input a matrix of any size, calculates the temperature T using bipartite's function *nested(matrix, "binmatnest")*, transforms the temperature via $(100-T)/100$ to scale the nestedness into 0 (no nestedness) to 1 (full nestedness), and returns the nestedness. What is the nestedness of your interaction matrix?

(g) We now know the nestedness of our interaction matrix but what does this tell us? Is the nestedness higher than we could have expected by chance? To find this out, we have to perform a statistical analysis. We will create many 'similar' matrices, calculate their nestednesses and compare them to the nestedness of our real interaction matrix.

In the 2003 PNAS paper, Prof. Bascompte uses two 'null models'. You can think of them as recipes to create similar matrices to the one under investigation. Look at the paper and write a function *nullmodel1* that implements null model 1. If time allows, write another function *nullmodel2* that implements null model 2. Both functions take as input a matrix of any size and output a similar matrix based on the specifics described in the paper.

Hints:

1. You will need to use some of R's random number generating functions like *runif*, *rbinom*, or *sample.int*.
2. It's not explicitly written in the paper but don't you agree that it would make sense to ensure that each species has at least one connection (*rowSums* and *colSums* are helpful here)

(h) We now have all the necessary functions to test whether the nestedness of our interaction matrix is significantly higher than could be expected by chance. Create *Nsim=100* null models (the more the better). Calculate and record the nestedness of each null model. How can we use these values to evaluate whether the nestedness of our interaction matrix was significantly higher than expected by chance?

(i) Now that you have the whole pipeline set up, look at Table 1 of Prof. Bascompte's 2003 PNAS paper and download (from the web of life) one network that has supposedly highly significant nestedness (**) and one that has not significant nestedness (NS). Let's see whether we can recreate the same results.

# Problem 2: Scale-free networks

## Objectives

You will learn how scale free networks emerge naturally.

(a) In a breakthrough 1999 paper, Barabasi and Albert found that a large variety of networks share a universal property: Their degree distribution is 'scale-free'. That means, they possess many nodes with only one or a few connections but also a few hubs with many connections. At the same time, they introduced preferential attachment as a mechanism that gives rise to a scale-free distribution. What is a logical explanation for the scale-free distribution of, for example, the network of the internet?

(b) Starting with only two connected nodes, grow an undirected network to size $N = 100$ according to the principal of preferential attachment. Each new node shall connect to m existing nodes (in the easiest case $m = 1$), based on the degree distribution of the existing nodes. For example, a hub with already 10 connections is 10 times as likely to become connected to a newly added node as a node with only one connection. Use igraph's function *make_graph(edgelist, directed=FALSE)* to show the network at different steps of the creation process.

Hints:

1. It will be helpful for you to initialize a vector of node pairs as well as a vector keeping track of how many other nodes each node is connected to:

```
edgelist <- c(1, 2)
degreelist <- c(1, 1)
```

*edgelist* is initialized with the first edge, from 1 to 2. *degreelist* here shows that node 1 is attached to 1 other node, as is node 2.

2. *sample.int(number_of_existing_nodes, m, replace = F, prob = degreelist)* might prove helpful in selecting m existing nodes for the connections to a newly added node
3. Remember to use R's *help* function if you want to learn more about a function like for example *make_graph*.
4. Don't forget to update *degreelist*!
5. Once you've created the edgelist, you can use the following for nice plotting

```
A <- make_graph(edgelist, directed = FALSE)
V(A)$label <- ""
V(A)$size <- degreelist + 2
plot(A)
```