Creating Networks

The goal

The goal of this assignment is very simple - learn how to create a network in R.

We will talk about three main ways to do this, but we will start with the hardest first.

Networks from two spreadsheets

I think it's really useful to think of networks as two spreadsheets. The first spreadsheet is a list of edges and their attributes (a.k.a. an edgelist) and the second is a list of nodes and their attributes.

We'll start by creating a simple edgelist, using made up names.

Exercise

In Excel (or LibreOffice or Google Sheets) open a new spreadsheet, and create a spreadsheet that looks like this:

From	To
John	Ida
Elwyn	Ida
Okan	Axelle
John	Axelle
Axelle	Ida

Save the spreadsheet to your computer as a .xlsx file and then:

1. Click `File > Import Dataset > Excel`
2. Browse to the file
3. Under Import options, change the name to `edgelist`
4. Copy and paste the code in the "Code Preview" window here (delete my code which is given as an example).
5. *If you are using a Mac or Linux, you may have to add '~/' to the beginning of the file path.

```
library(readxl)
#edgelist <- read_excel("~/Teaching/communication_and_networks/week_4/edgaelist.xlsx")
#View(edgelist)
```

You might get an error here that says something like `there is no package called 'readxl'`. If you get that, then you need to run `install.packages('tidyverse')` in the console (this is the bottom left window). It will take a while to install, but if it works correctly your code above should work.

To test if everything worked right, uncomment and this code to print out all of the edges in R.

(Note: the "#" symbol "comments out" code - this means that anything after the # will be ignored by R)

```
#print(edgelist)
```

Excercise

Next, we will create a table of attributes for the people in our network, and import that, too.

Do the same thing again - open a new spreadsheet, and this time make it look like this:

Name	Age	Gender	Major
John	22	M	COM
Elwyn	23	M	SOC
Okan	25	F	COM
Ida	19	F	SOC
Axelle	20	T	COM

Make sure that the spelling (including capitalization) is the same as in your edgelist file.

Again, save it and import it. This time, import it as node_atts.

Here is my import code, which you should delete and replace with your own code:

```
#library(readxl)
#node_atts <- read_excel("~/Teaching/communication_and_networks/week_4/node_atts.xlsx")
#View(node_atts)
```

Uncomment and run the following code to check that it worked.

```
#print(node_atts)
```

Importing into R

Finally, we'll create a network "object" using these files. Right now it's just two "dataframes" (what R calls spreadsheets). Creating a network objects tells R that these represent a network, so that it can create visualizations, get network statistics, etc.

We do this by running `graph_from_data_frame`. This is a function that takes in the arguments `d`, `vertices`, and `directed`.

`d` is the edgelist `vertices` is the node attributes `directed` is whether this is a directed or undirected graph. `G <-` saves the network to the variable `G` and then `plot(G)` makes a basic plot of it.

If you've done everything right, then when you uncomment the code below, it should produce a network plot.

```
#G <- graph_from_data_frame(d = edgelist,
#                           vertices = node_atts, directed = T)
#plot(G)
```

Finally, to make things even more confusing, for lots of the cool things we'll be doing in future lessons, we need to take one more step, and turn this from a normal network object into a `tidygraph` network object.

To do this, we take `G` (the network object we just created) and run `as_tbl_graph()` on it.

Uncomment the code below and run it. If it worked, then you should see a text output with two tables:

`Node Data` and `Edge Data`

```
#G <- as_tbl_graph(G)

#G
```

Exercise

See if you can figure out how to add another person to the network (hint: the easiest way to do this is in Excel). Give them node attributes and at least one edge. Use `graph_from_data_frame` to create a network object and `as_tbl_graph` to change it into a `tidygraph` network object.

Finally, create a new network plot with the new person.

```
## YOUR CODE HERE
```

Loading data from R packages

There are also R packages which have network data that you can load right into R. One of the biggest fairly simple ones is the `networkdata` package.

Unfortunately, it's a little bit tricky to do that for this package. Try to run `install.packages("remotes")` in the Console (at the bottom left), and when that is done run `remotes::install_github("schochastics/networkdata")`, also in the console.

If it works, then you should be able to load the package when you uncomment the code below.

```
#library(networkdata)
```

Now, you will have access to the networks from the package. Run `data(package='networkdata')` to see all of the options. This code shows an advice network from a high-tech firm:

```
# This code turns the ht_advice network object into a tidygraph object, and saves it as G
#G <- as_tbl_graph(ht_advice)
#plot(G)
```

Exercise

Figure out how to load the `ffe_elite` network as a `tidygraph` object, and plot it.

```
# YOUR CODE HERE
```

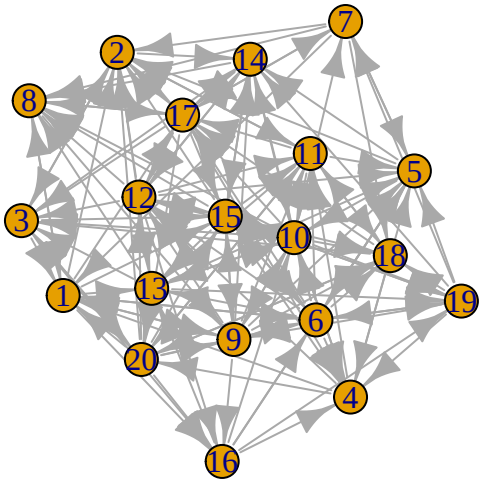
Random networks

Finally(!), the last thing you can do is to create random networks in R. There are a ton of options, so I'll just show you one.

This is a Erdos-Renyi graph, where each edge is created randomly.

```
G = play_erdos_renyi(20, .4)

plot(G)
```



Excercise

The `play_erdos_renyi()` function above has to parameters, which have the values `20` and `.4`. Try changing the values and observing the result. What do you think each of these parameters do?

YOUR ANSWER HERE

Exercise

A list of a bunch of possible random graphs is listed at https://tidygraph.data-imaginist.com/reference/index.html under "Graph Creation". See if you can figure out how to make a ring graph with 20 nodes.