Tutorial: Network Description, Week 1

Introduction

This tutorial will demonstrate use of the R commands used to identify critical aspects of network data introduced in Week 1's Lecture:

- Vertex and edge counts
- Bipartite or single mode network
- Directed or undirected edges
- Weighted or unweighted edges
- Identify available vertex and edge attributes
- Access vertex and edge attributes
- Describe vertex and edge attributes

These network features are critical to identify because many network analysis techniques require a specific type of data. For example, some structural equivalence techniques require binary or unweighted edges, while Gould-Fernandez brokerage scores require directed edges and additional node attributes.

We will be using two network analysis packages throughout the course: **igraph** and **statnet**. (Statnet is technically a suite of network packages that are designed to work together, including **network** for creating statnet objects and **sna** for basic network analysis.) Each of the packages uses a slightly different network object format, and the network analysis commands in the package will only work on the correct format of network object. When possible, all tutorial examples will use commands with similar functionality from both the igraph package and the statnet suite. Generic indicators will be appended to the network object name to distinguish network objects in statnet and igraph format.

Datasets used in this tutorial are the Karate and USAirport datasets described in the igraphdata package.

- *airport.ig* - igraph network object
- *airport.stat* - statnet network object

- *karate.ig* - igraph network object
- *karate.stat* - statnet network object

Tutorial: Network Description, Week 1

Network Data in Environment

Ordinarily, the first step in any data analysis exercise is to load the data, clean it up, and transform it into the correct object type for analysis. To make the tutorial more accessible, we will be working with pre-cleaned and pre-loaded datasets. To confirm that all four of the specificed datasets are available for us, we can use the `ls()` command with no options.

Try it now. Just type the command `ls()` in the box below and then hit the `Run Code` button.

Exercise: List Objects

Code | Start Over | Solution | Run Code

```
1 #List the network objects currently available fo
2 ls()
3
```

```
[1] "airport.ig"   "airport.stat" "karate.ig"
   "karate.stat"
```

Previous Topic Next Topic

Tutorial: Network Description, Week 1

Network Size

When we first start working with network data, we may not know much about the dataset or may want to confirm that the data read in correctly by comparing it to some basic descriptive facts. Lets start by finding the size of the network, both the number of nodes and the number of edges.

```
#Count of vertices: igraph
vcount(karate.ig)
```

```
## [1] 34
```

```
#Count of edges: igraph
ecount(karate.ig)
```

```
## [1] 78
```

```
#Count of both vertices and edges: statnet
print(karate.stat)
```

```
##  Network attributes:
##   vertices = 34
##   directed = FALSE
##   hyper = FALSE
##   loops = FALSE
##   multiple = FALSE
##   bipartite = FALSE
##   name = Zachary's karate club network
##   Citation = Wayne W. Zachary. An Informati
on Flow Model for Conflict and Fission in Smal
l Groups. Journal of Anthropological Research
Vol. 33, No. 4 452-473
##   Author = Wayne W. Zachary
##   total edges= 78
##     missing edges= 0
##     non-missing edges= 0
##
##  Vertex attribute names:
##     color Faction label vertex.names
##
##  Edge attribute names:
##     weight
```

Tutorial: Network Description, Week 1

Note that the print command on a statnet graph gives a very clean summary of almost all network features we are interested in. Although igraph is typically considered easier to use, you only need to remember one command in statnet to get a complete summary of your available data. The `print()` command can be used on an igrah network, but it is much more difficult to interpret and is missing much of the information returned by `print()` on a statnet network.

To see the difference, try the `print()` command on the karate data in igraph format. Remember to hit `Run Code` after entering the command in the box below.

Code | Start Over | Solution | ▶ Run Code

```
1 #igraph equivalent of statnet::print()
2 print(karate.ig)
3
```

```
IGRAPH 4b458a1 UNW- 34 78 -- Zachary's karate
club network
+ attr: name (g/c), Citation (g/c), Author (g/
c), Faction (v/n),
| name (v/c), label (v/c), color (v/n), weight
(e/n)
+ edges from 4b458a1 (vertex names):
 [1] Mr Hi  --Actor 2  Mr Hi  --Actor 3  Mr Hi
  --Actor 4
 [4] Mr Hi  --Actor 5  Mr Hi  --Actor 6  Mr Hi
  --Actor 7
 [7] Mr Hi  --Actor 8  Mr Hi  --Actor 9  Mr Hi
  --Actor 11
[10] Mr Hi  --Actor 12 Mr Hi  --Actor 13 Mr Hi
  --Actor 14
[13] Mr Hi  --Actor 18 Mr Hi  --Actor 20 Mr Hi
  --Actor 22
[16] Mr Hi  --Actor 32 Actor 2--Actor 3  Actor
2--Actor 4
[19] Actor 2--Actor 8  Actor 2--Actor 14 Actor
2--Actor 18
+ ... omitted several edges
```

Exercise: Network Size

Now, using commands from either statnet or igraph, find the network size of the “airports” network data. Be sure to use the correct network data format (and name) for the command you choose to use.

Code | Start Over | Solution | ▶ Run Code

Tutorial: Network Description, Week 1


```
1 #Find network size (vertex and edge count)
2 vcount(airport.ig)
3 ecount(airport.ig)
4
```

```
[1] 755
```

```
[1] 23473
```

Previous Topic Next Topic

Tutorial: Network Description, Week 1

Start Over

Weighted, Directed, Bipartite?

As we saw in the last exercise, statnet `print()` command provides a description of several critical network features:

- Is this a Bipartite or single mode network?
- Are edges directed or undirected?
- Are edges weighted or unweighted?

The following igraph commands can be used to answer the same questions.

```
# Is this a Bipartite or single mode network?
is_bipartite(karate.ig)
```

```
## [1] FALSE
```

```
#  Are edges directed or undirected?
is_directed(karate.ig)
```

```
## [1] FALSE
```

```
#Are edges weighted or unweighted?
is_weighted(karate.ig)
```

```
## [1] TRUE
```

Exercise: Network Features

Now, using commands from either statnet or igraph, find the relevant features of the “airports” network data. Be sure to use the correct network data format (and name) for the command you choose to use.

Code | Start Over | Solution | Run Code

```
1 #Find network features (mode, direction, weights
2
3 is_bipartite(airport.ig)
4 is_directed(airport.ig)
5 is_weighted(airport.ig)
```

```
[1] FALSE
```

Tutorial: Network Description, Week 1


```
[1] TRUE
```

```
[1] FALSE
```

Previous Topic Next Topic

Tutorial: Network Description, Week 1

Vertex and Edge Attributes

You may have noticed that the `print()` command in both packages returned a list of vertex and edge attributes available in the dataset. These attributes are the equivalent of variables in a traditional set, with the difference being that there can be additional variables about both the vertices and the edges in a network dataset.

A **vertex attribute** is a vector (or variable) of additional information about the nodes in a network dataset, such as the name, location, or other demographic information about the actors or organizations who comprise the vertices of the network. In case you have forgotten, we can quickly get a list of available vertex attributes. (Note: the statnet command is provided but is not run for simplicity.)

```
#display vertex attributes for igraph object
vertex_attr_names(karate.ig)
```

```
## [1] "Faction" "name"    "label"   "color"
```

```
#display vertex attributes for statnet object
#network::list.vertex.attributes(karate.stat)
```

An **edge attribute** is a vector (or variable) of additional information about the edges or relationships in a network dataset. The most common edge attribute is *weight*, in a weighted dataset. Other edge attributes encountered might be *year* of the relationship (e.g., when looking at trade or alliance data), *type* of relationship (e.g., marriage or parentage in family tree data, or mutual defense or non-agression in alliance data), or other attributes that can be used to characterize the relationships in the network. We can also quickly get a list of available edge attributes.

```
#display edge attributes for igraph object
edge_attr_names(karate.ig)
```

```
## [1] "weight"
```

```
#display edge attributes for statnet object
#network::list.edge.attributes(karate.stat)
```

Tutorial: Network Description, Week 1

Exercise: List Network Attributes

Now, using commands from either statnet or igraph, list the vertex and edge attributes of the “airports” network data. Be sure to use the correct network data format (and name) for the command you choose to use.

Code | Start Over | Solution | Run Code

```
1 #List network attributes (vertex and edge)
2 igraph::vertex_attr_names(airport.ig)
3 igraph::edge_attr_names(airport.ig)
4
```

```
[1] "name"     "City"     "Position"
```

```
[1] "Carrier"    "Departures" "Seats"      "Passengers" "Aircraft"
[6] "Distance"
```

Previous Topic Next Topic

Tutorial: Network Description, Week 1

Accessing Attribute Data

When we work with an R dataframe object, you can access a specific variable in the dataframe by using the `data$varname` or `$` command. You can work with network data in a similar way to access network attributes (or variables), but the specific command varies by whether you are accessing a vertex or edge attribute, as indicated.

```
#access vertex attribute: igraph
V(karate.ig)$name
```

```
##  [1] "Mr Hi"    "Actor 2"  "Actor 3"  "Acto
r 4"  "Actor 5"  "Actor 6"
##  [7] "Actor 7"  "Actor 8"  "Actor 9"  "Acto
r 10" "Actor 11" "Actor 12"
## [13] "Actor 13" "Actor 14" "Actor 15" "Acto
r 16" "Actor 17" "Actor 18"
## [19] "Actor 19" "Actor 20" "Actor 21" "Acto
r 22" "Actor 23" "Actor 24"
## [25] "Actor 25" "Actor 26" "Actor 27" "Acto
r 28" "Actor 29" "Actor 30"
## [31] "Actor 31" "Actor 32" "Actor 33" "John
A"
```

```
#access edge attribute: igraph
E(karate.ig)$weight
```

```
##  [1] 4 5 3 3 3 3 2 2 2 3 1 3 2 2 2 2 6 3 4
5 1 2 2 2 3 4 5 1 3 2 2 2 3 3 3
## [36] 2 3 5 3 3 3 3 3 4 2 3 3 2 3 4 1 2 1 3
1 2 3 5 4 3 5 4 2 3 2 7 4 2 4 2
## [71] 2 4 2 3 3 4 4 5
```

```
#access vertex attribute: statnet
head(karate.stat %v% "vertex.names")
```

```
## [1] "Mr Hi"   "Actor 2" "Actor 3" "Actor 4"
"Actor 5" "Actor 6"
```

```
#access edge attribute: statnet
head(karate.stat %e% "weight")
```

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

Tutorial: Network Description, Week 1

The additional command, `head()`, used for the statnet example is very useful when you want to inspect the contents of a long vector of attribute data to get a sense of the content, but do not need to see a list of every value.

Exercise: Accessing Attribute Data

Now, using commands from either statnet or igraph, access the vertex attribute “City” and the edge attribute “Carrier” in the “airports” network data. Be sure to use the correct network data format (and name) for the command you choose to use. You may want to use the `head()` command to shorten the lengthy output.

Code | Start Over | Solution | Run Code

```
1 #Access "City" vertex attribute
2 head(V(airport.ig)$City)
3 #Access "Carrier" edge attribute
```

```
[1] "Bangor, ME"    "Boston, MA"    "Anchorag
e, AK" "New York, NY"
[5] "Las Vegas, NV" "Miami, FL"
```

Previous Topic Next Topic

Tutorial: Network Description, Week 1

Describing Network Attributes

Any R commands that are ordinarily used to work with vector objects can also be used to work with network attribute data accessed using the techniques in the last step. This means that we can do useful things like sort lists of names using `sort()` or summarize numeric data using `summary()`. Here is an example of how to summarize numeric network attributes.

```
#summarize numeric network attribute: igraph
summary(E(karate.ig)$weight)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.
Max.
##   1.000   2.000   3.000   2.962   4.000
7.000
```

```
#summarize numeric network attribute: statnet
summary(karate.stat %e% "weight")
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.
Max.
##   1.000   2.000   3.000   2.962   4.000
7.000
```

Exercise: Summarize Network Attributes

Using the airport data, summarize the numeric edge attribute "Passengers."

Code | Start Over | Solution | Run Code

```
1 #summarize numeric network attribute: igraph
2 summary(E(airport.ig)$Passengers)
3
4 #summarize numeric network attribute: statnet
5 summary(airport.stat %e% "Passengers") #summariz
6
7
```

Tutorial: Network Description, Week 1


```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Ma
x.
      1      77     721    2238    2735    7215
2
```

```
Length  Class   Mode
     0   NULL   NULL
```

Congratulations. You have completed the Week 1 Tutorial on Network Description.

Previous Topic