Social Network Analysis in R

CORE Lab

Department of Defense Analysis

2019-06-11

Background

Social network analysis (SNA) is commonly used to understand groups and social formations. **The focus of this methodology is the relationships among individuals, which influence a person's behavior above and beyond the influence of his or her individual attributes (Valente 2010).** As such, SNA enables analysts to understand how social ties help to define, enable, and constrain the knowledge, reach, and capacities of actors within groups (Cunningham, Everton, and Murphy 2016).

While social network research is **not** exclusively dependent on software applications, these do increase the efficiency of researchers. Here we will focus on **R**.

Goals

Here we will explore some of the key SNA features of the open-source programming language **R**, and a variety of packages - primarily **igraph** (https://igraph.org/r/) and **visNetwork** (https://datastorm-open.github.io/visNetwork/) - developed to work with relational data. Specifically, we will:

- Explore, structure, visualize, and analyze relational data with the **igraph** library.
- Build processes in **R** with **igraph** to streamline analysis.
- Create interactive visualizations with the **visNetwork** package.
- Please note that **statnet** (http://www.statnet.org/) is another commonly used package among social network analysts.

Supplemental Packages

- here - A package to make it easier to find your files by constructing paths to your project's files.
- tidyverse - A collection of packages designed for data science. Users get packages such as dplyr, ggplot, purr, etc.
- purr - A useful tool for working with vectors.
- DT - A "wrapper" of the JavaScript Library "DataTables". We will use it to build interactive data tables.
- kableExtra - A package to help us build tables using HTML.
- emo - A package that allows users to insert emoji into RMarkdown documents (like this one!).
- xaringan - You're looking at it!

Getting Started: Loading Data

Let's bring in data from an edgelist:

```
read.csv(here::here("data/Familial.csv"), header = TRUE)
#familial ← as.data.frame(read.csv(file="Familial.csv", header=TRUE))
```

Source	Target	Relationship
Adonis	Boxcar	Familial
Adonis	Ghost	Familial

Previous 1 2 3 Next

Relationships Codebook

- Familial: (person-to-person; i.e., one-mode) - Defined as any family connection through blood, adoption, or marriage.
- Financial: person-to-person) - Defined as two actors, in reporting or intelligence, who are explicitly stated as transferring funds between one another for any purpose, legal or illegal.
- Friendship: (person-to-person) - Defined as two individuals who are explicitly stated as friends, or who are explicitly known as trusted confidants in reports or in intelligence documentation.
- Hierarchy: (person-to-person) - Defined as relationships between immediate superiors and subordinates in an organization.

Relationships Import

Familial:

```
familial ← read.csv(here::here("data/Familial.csv"), header = TRUE)
```

Financial:

```
financial ← read.csv(here::here("data/Financial.csv"), header = TRUE)
```

Friendship:

```
friendship ← read.csv(here::here("data/Friendship.csv"), header = TRUE)
```

Hierarchy:

```
hierarchy ← read.csv(here::here("data/Hierarchy.csv"), header = TRUE)
```

Building Networks

First, install the package:

```
install.packages("igraph")
```

Now load the package:

```
library(igraph)
```

Create an **igraph** graph:

```
familialNet ← igraph::graph_from_data_frame(familial,
                                            directed = FALSE)
```

Familial Network Object

```
familialNet
```

```
## IGRAPH 41fe87c UN-- 22 30 --
## + attr: name (v/c), Relationship (e/c)
## + edges from 41fe87c (vertex names):
##  [1] Adonis   --Boxcar    Adonis   --Ghost     Adonis   --Jelly
##  [4] Bananas  --Blue Eyes Bananas  --Brains    Bananas  --Slingshot
##  [7] Bananas  --Gremlin   Bat G.   --Big G.    Bat G.   --Blaze
## [10] Big G.   --Blaze     Boots    --Fat Boy   Boots    --Freckles
## [13] Boots    --Icepick   Boots    --Repo Girl Boxcar   --Ghost
## [16] Boxcar   --Jelly     Brains   --Slingshot Brains   --Gremlin
## [19] Fat Boy  --Freckles  Fat Boy  --Icepick   Fat Boy  --Repo Girl
## [22] Slingshot--Gremlin   Freckles --Icepick   Freckles --Repo Girl
## + ... omitted several edges
```

- `name` listed as `(v/c)`, which denotes a vertex-level character attributes
- `Relationship` listed as `(e/c)` or edge-level character attributes

Familial Network Edges

```
E(familialNet)
```

```
## + 30/30 edges from 41fe87c (vertex names):
##  [1] Adonis    --Boxcar      Adonis    --Ghost       Adonis    --Jelly
##  [4] Bananas   --Blue Eyes   Bananas   --Brains      Bananas   --Slingshot
##  [7] Bananas   --Gremlin     Bat G.    --Big G.      Bat G.    --Blaze
## [10] Big G.    --Blaze       Boots     --Fat Boy     Boots     --Freckles
## [13] Boots     --Icepick     Boots     --Repo Girl   Boxcar    --Ghost
## [16] Boxcar    --Jelly       Brains    --Slingshot   Brains    --Gremlin
## [19] Fat Boy   --Freckles    Fat Boy   --Icepick     Fat Boy   --Repo Girl
## [22] Slingshot --Gremlin     Freckles  --Icepick     Freckles  --Repo Girl
## [25] Ghost     --Jelly       Goldie    --Pookey      Goldie    --O.G.
## [28] Pookey    --O.G.        Icepick   --Repo Girl   Snake     --Sonny Black
```

```
ecount(familialNet)
```

```
## [1] 30
```

Familial Network Nodes

```
V(familialNet)
```

```
## + 22/22 vertices, named, from 41fe87c:
##  [1] Adonis      Bananas     Bat G.      Big G.      Boots
##  [6] Boxcar      Brains      Fat Boy     Slingshot   Freckles
## [11] Ghost       Goldie      Pookey      Icepick     Snake
## [16] Jelly       Blue Eyes   Gremlin     Blaze       Repo Girl
## [21] O.G.        Sonny Black
```

```
vcount(familialNet)
```

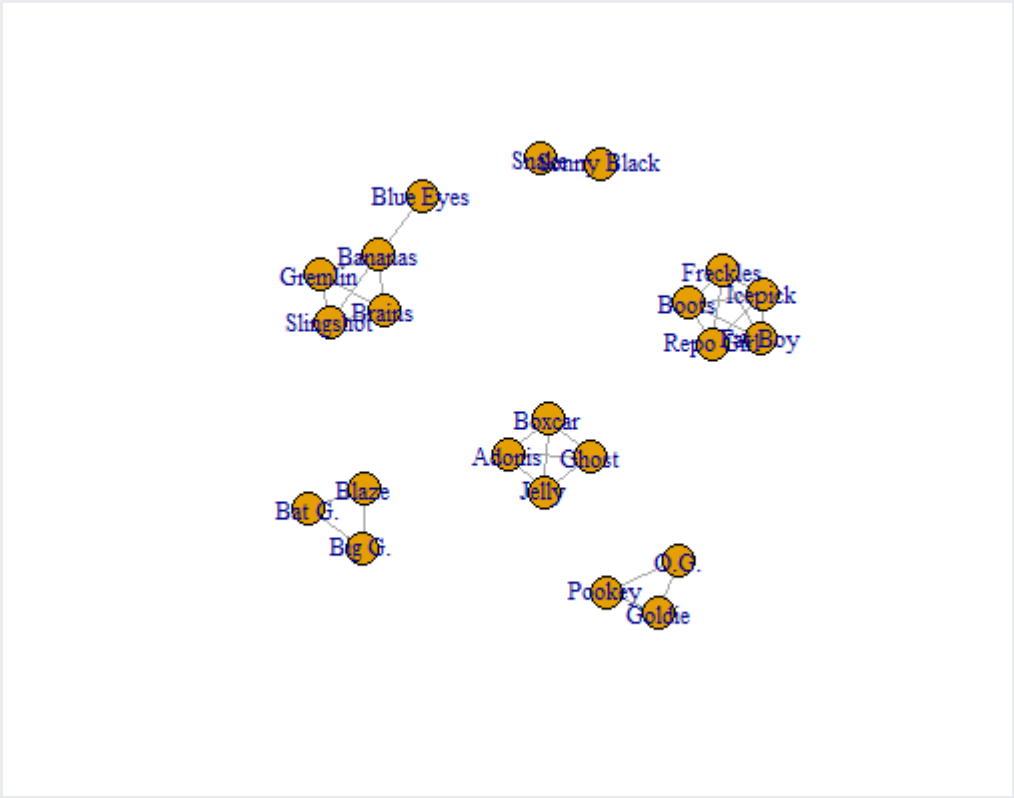
```
## [1] 22
```

```
V(familialNet)$name
```

```
##  [1] "Adonis"      "Bananas"     "Bat G."      "Big G."      "Boots"
##  [6] "Boxcar"      "Brains"      "Fat Boy"     "Slingshot"   "Freckles"
## [11] "Ghost"       "Goldie"      "Pookey"      "Icepick"     "Snake"
## [16] "Jelly"       "Blue Eyes"   "Gremlin"     "Blaze"       "Repo Girl"
## [21] "O.G."        "Sonny Black"
```

Familial Network Visualization

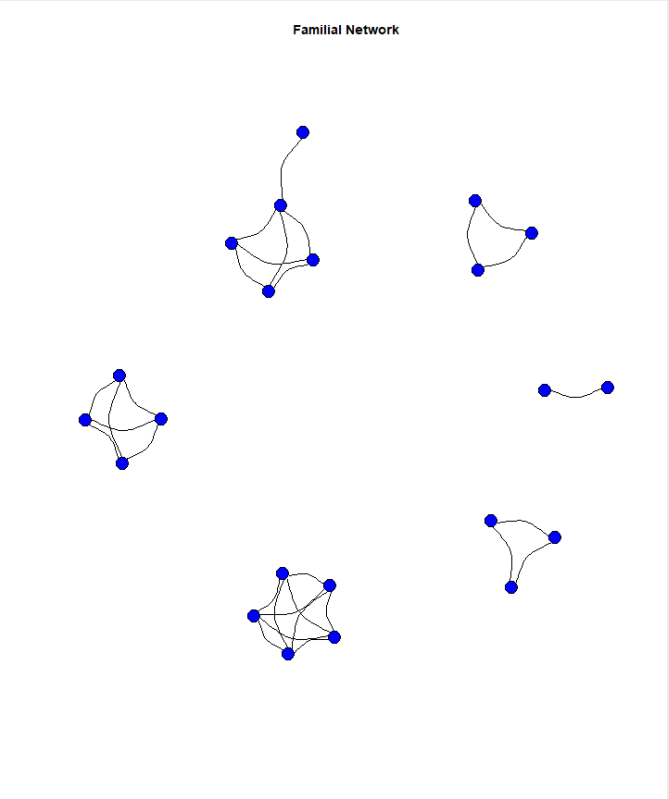
```
plot.igraph(familialNet)
```



Familial Network Visualization

```
plot.igraph(familialNet,
            # Nodes ====
            vertex.label = NA,
            vertex.color = "blue"
            vertex.size = 5,
            # Edge ====
            edge.color = "black",
            edge.arrow.size = 0,
            edge.curved = TRUE,
            # Other ====
            margin = .01,
            frame = FALSE,
            main = "Familial Netw
            )
```

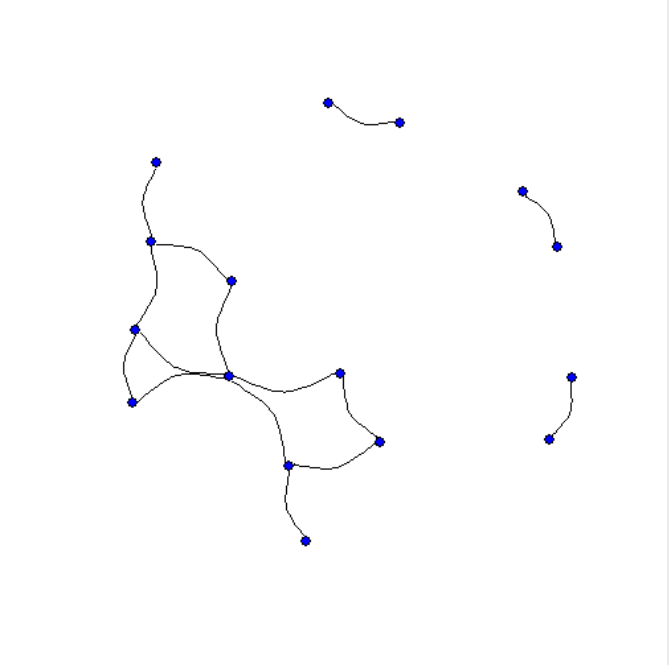
Arguments = 😄



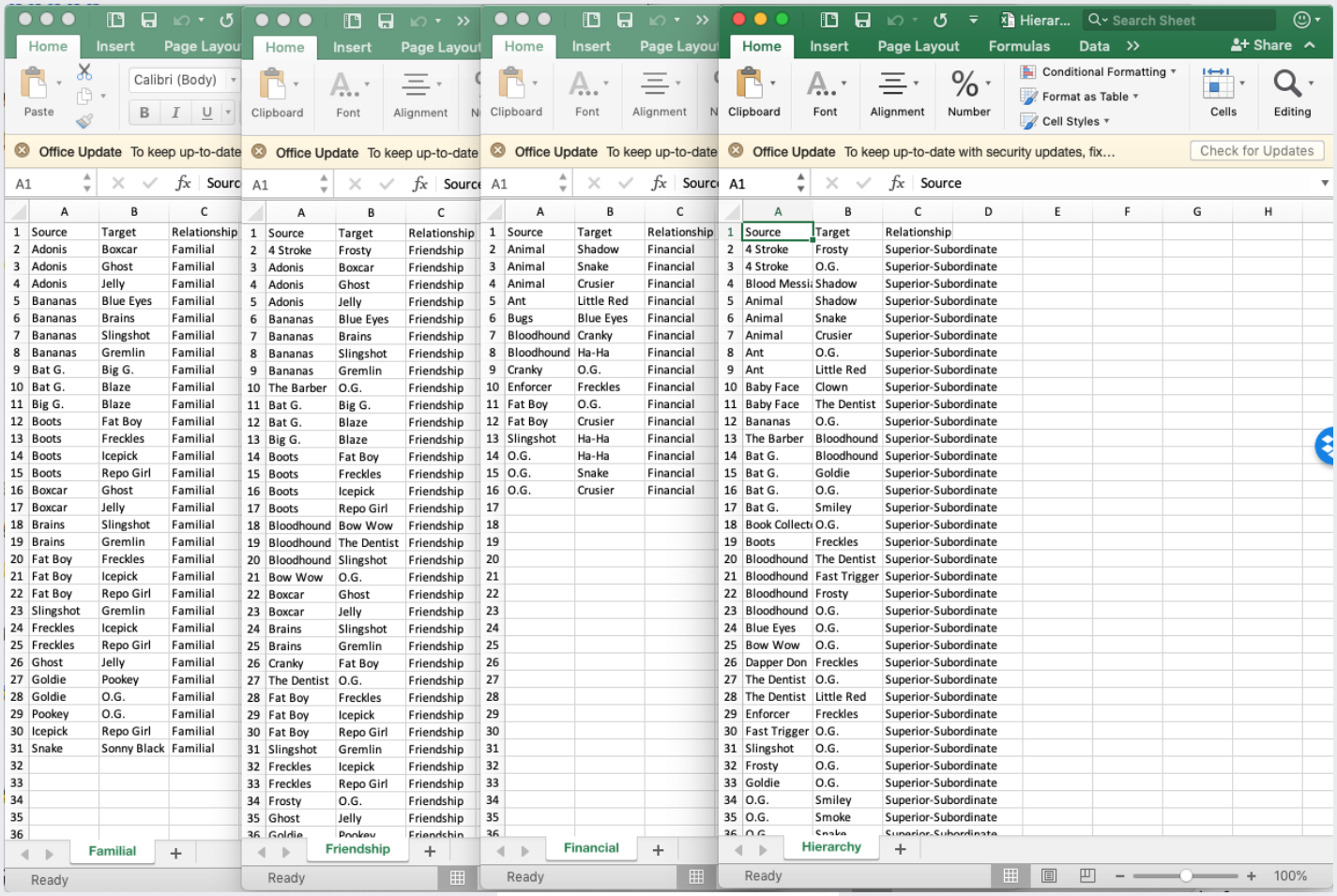
So What?

Automation! Automation! Automation!

```
read.csv(here::here("data/Financia
  igraph::graph_from_data_frame(d
  igraph::plot.igraph(vertex.label
            vertex.color="blue",
            vertex.size=5,
            edge.color="black",
            edge.arrow.size=0,
            edge.curved=TRUE,
            margin = .01,
            frame = FALSE
            )
```



Automation! Automation! Automation!



Source	Target	Relationship
Adonis	Boxcar	Familial
Adonis	Ghost	Familial
Adonis	Jelly	Familial
Bananas	Blue Eyes	Familial
Bananas	Brains	Familial
Bananas	Slingshot	Familial
Bananas	Gremlin	Familial
Bat G.	Big G.	Familial
Bat G.	Blaze	Familial
Big G.	Blaze	Familial
Boots	Fat Boy	Familial
Boots	Freckles	Familial
Boots	Icepick	Familial
Boots	Repo Girl	Familial
Boxcar	Ghost	Familial
Boxcar	Jelly	Familial
Brains	Slingshot	Familial
Brains	Gremlin	Familial
Fat Boy	Freckles	Familial
Fat Boy	Icepick	Familial
Fat Boy	Repo Girl	Familial
Slingshot	Gremlin	Familial
Freckles	Icepick	Familial
Freckles	Repo Girl	Familial
Ghost	Jelly	Familial
Goldie	Pookey	Familial
Goldie	O.G.	Familial
Pookey	O.G.	Familial
Icepick	Repo Girl	Familial
Snake	Sonny Black	Familial

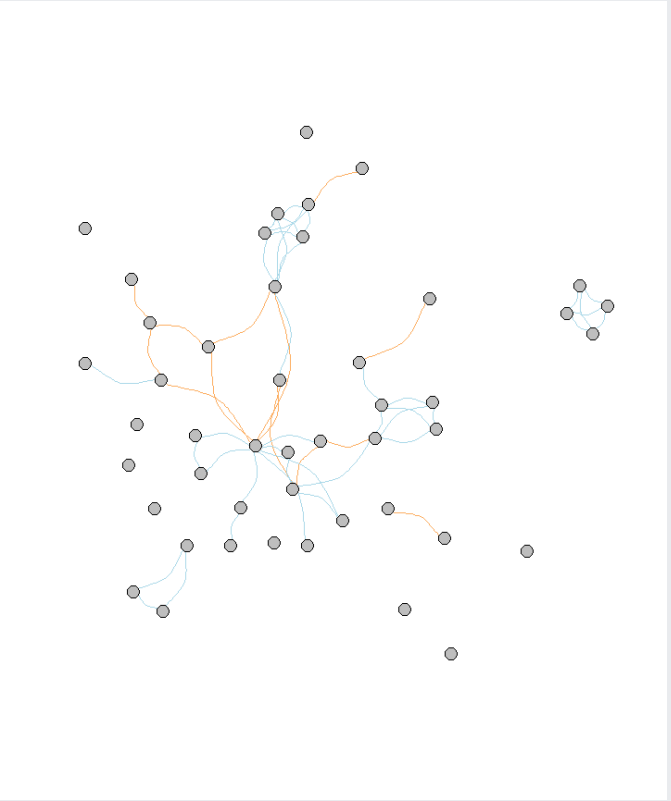
Source	Target	Relationship
4 Stroke	Frosty	Friendship
Adonis	Boxcar	Friendship
Adonis	Ghost	Friendship
Adonis	Jelly	Friendship
Bananas	Blue Eyes	Friendship
Bananas	Brains	Friendship
Bananas	Slingshot	Friendship
Bananas	Gremlin	Friendship
The Barber	O.G.	Friendship
Bat G.	Big G.	Friendship
Bat G.	Blaze	Friendship
Big G.	Blaze	Friendship
Boots	Fat Boy	Friendship
Boots	Freckles	Friendship
Boots	Icepick	Friendship
Boots	Repo Girl	Friendship
Bloodhound	Bow Wow	Friendship
Bloodhound	The Dentist	Friendship
Bloodhound	Slingshot	Friendship
Bow Wow	O.G.	Friendship
Boxcar	Ghost	Friendship
Boxcar	Jelly	Friendship
Brains	Slingshot	Friendship
Brains	Gremlin	Friendship
Cranky	Fat Boy	Friendship
The Dentist	O.G.	Friendship
Fat Boy	Freckles	Friendship
Fat Boy	Icepick	Friendship
Fat Boy	Repo Girl	Friendship
Slingshot	Gremlin	Friendship
Freckles	Icepick	Friendship
Freckles	Repo Girl	Friendship
Frosty	O.G.	Friendship
Ghost	Jelly	Friendship

Source	Target	Relationship
Animal	Shadow	Financial
Animal	Snake	Financial
Animal	Crusier	Financial
Ant	Little Red	Financial
Bugs	Blue Eyes	Financial
Bloodhound	Cranky	Financial
Bloodhound	Ha-Ha	Financial
Cranky	O.G.	Financial
Enforcer	Freckles	Financial
Fat Boy	O.G.	Financial
Fat Boy	Crusier	Financial
Slingshot	Ha-Ha	Financial
O.G.	Ha-Ha	Financial
O.G.	Snake	Financial
O.G.	Crusier	Financial

Source	Target	Relationship
4 Stroke	Frosty	Superior-Subordinate
4 Stroke	O.G.	Superior-Subordinate
Blood Messi	Shadow	Superior-Subordinate
Animal	Shadow	Superior-Subordinate
Animal	Snake	Superior-Subordinate
Animal	Crusier	Superior-Subordinate
Ant	O.G.	Superior-Subordinate
Ant	Little Red	Superior-Subordinate
Baby Face	Clown	Superior-Subordinate
Baby Face	The Dentist	Superior-Subordinate
Bananas	O.G.	Superior-Subordinate
The Barber	Bloodhound	Superior-Subordinate
Bat G.	Bloodhound	Superior-Subordinate
Bat G.	Goldie	Superior-Subordinate
Bat G.	O.G.	Superior-Subordinate
Bat G.	Smiley	Superior-Subordinate
Book Collecti	O.G.	Superior-Subordinate
Boots	Freckles	Superior-Subordinate
Bloodhound	The Dentist	Superior-Subordinate
Bloodhound	Fast Trigger	Superior-Subordinate
Bloodhound	Frosty	Superior-Subordinate
Bloodhound	O.G.	Superior-Subordinate
Blue Eyes	O.G.	Superior-Subordinate
Bow Wow	O.G.	Superior-Subordinate
Dapper Don	Freckles	Superior-Subordinate
The Dentist	O.G.	Superior-Subordinate
The Dentist	Little Red	Superior-Subordinate
Enforcer	Freckles	Superior-Subordinate
Fast Trigger	O.G.	Superior-Subordinate
Slingshot	O.G.	Superior-Subordinate
Frosty	O.G.	Superior-Subordinate
Goldie	O.G.	Superior-Subordinate
O.G.	Smiley	Superior-Subordinate
O.G.	Smoke	Superior-Subordinate

Automation! Automation! Automation!

```
files ← list.files(path="data/one
                   pattern = "*.(
                   full.names = T
files %>%
  purrr::map_dfr(., ~.x %>% read_(
  igraph::graph_from_data_frame(d:
  igraph::set_edge_attr("color",
                        value =
                          E(.)$I
                          E(.)$I
                          E(.)$I
                          E(.)$I
                        )) %>%
  igraph::plot.igraph(vertex.labe
                      vertex.colo
                      vertex.size
                      edge.arrow.s
                      edge.curved
                      margin = .0
                      )
```



Metrics in Igraph

Q: Now that you have data, how can you analyze it?

Metric	Explanation	Command
Density	Number of observed ties divided by possible number of ties	`edge_density()`
Average Degree	Sum of ties divided by number of actors	`mean(degree())`
Global Clustering	Sum of each actor's clustering divided by number of actors	`transitivity()`

Quick Note on Commands

```
g ← list.files(path="data/onemode/",
                    pattern = "*.csv",
                    full.names = TRUE) %>%
  purrr::map_dfr(read_csv) %>%
  igraph::graph_from_data_frame(directed = FALSE)
```

```
edge_density(g, loops = FALSE) # Simple Command
```

```
## [1] 0.1188406
```

```
g_density←edge_density(g, loops = FALSE) # Assigning object
```

```
g_density# Calling "g_density" object
```

```
## [1] 0.1188406
```

Network-Level Measures

```
g ← list.files(path="data/onemode/",
                    pattern = "*.csv",
                    full.names = TRUE) %>%
  purrr::map_dfr(read_csv) %>%
  igraph::graph_from_data_frame(directed = FALSE) %>%
  igraph::set.graph.attribute("density", edge_density(.)) %>%
  igraph::set.graph.attribute("avg_degree", mean(degree(.))) %>%
  igraph::set.graph.attribute("avg_clu_coef", transitivity(., "average")
```

```
graph_attr(g, "density")
```

```
## [1] 0.1188406
```

```
graph_attr(g, "avg_degree")
```

```
## [1] 5.347826
```

```
graph_attr(g, "avg_clu_coef")
```

```
## [1] 0.6436332
```

Network-Level Measures Report

```
data.frame(
  "Density" = graph_attr(g, "density"),
  "Avg. Degree" = graph_attr(g, "avg_degree"),
  "Avg. Clustering Coefficient" = graph_attr(g, "avg_degree")
) %>%
  knitr::kable(format = "html", digits = 3, caption = "Demo Table") %>%
  kableExtra::kable_styling(bootstrap_options = c("striped", "condensed"
  kableExtra::add_footnote(label = "table footnote", notation = "number"
```

Demo Table

Density	Avg..Degree	Avg..Clustering.Coefficient
0.119	5.348	5.348

[1] table footnote

Metrics in Igraph

Q: Now that I've looked at network-level measures, what do I do?

Metric	Explanation	Command
Degree	Count of actor's ties	`degree()`
Eigenvector	Weights an actor's centrality by the centrality of its neighbors	`evcent()`
Betweenness	How often each actor lies on the shortest path between all other actors	`betweenness()`

Vertex-Level Measures

```
g %>%
  igraph::set.vertex.attribute("degree", value=degree(.)) %>%
  igraph::set.vertex.attribute("eigenvector", value=round(evcent(.)$vect
  igraph::set.vertex.attribute("betweenness", value=round(betweenness(.)
  igraph::get.data.frame("vertices") %>%
  DT::datatable(rownames = F,
                options = list(
                  pageLength=10,
                  dom="tp"
                ))
```

<!iv id="htmlwidget-76e5bd3935494abeed7a" style="width:100%;height:auto;" class="datatables html-widget">

<!

Interactive Visuals

First, install the package:

```
install.packages("visNetwork")
```

Now load the package:

```
library(visNetwork)
```

visNetwork with Igraph

```
visNetwork::visIgraph(g)
```

.center[<!iv id="htmlwidget-ea32a0d1ef6c6cb527c9" style="width:504px;height:504px;" class="visNetwork html-widget">

<!

visNetwork with Igraph Visualization Arguments

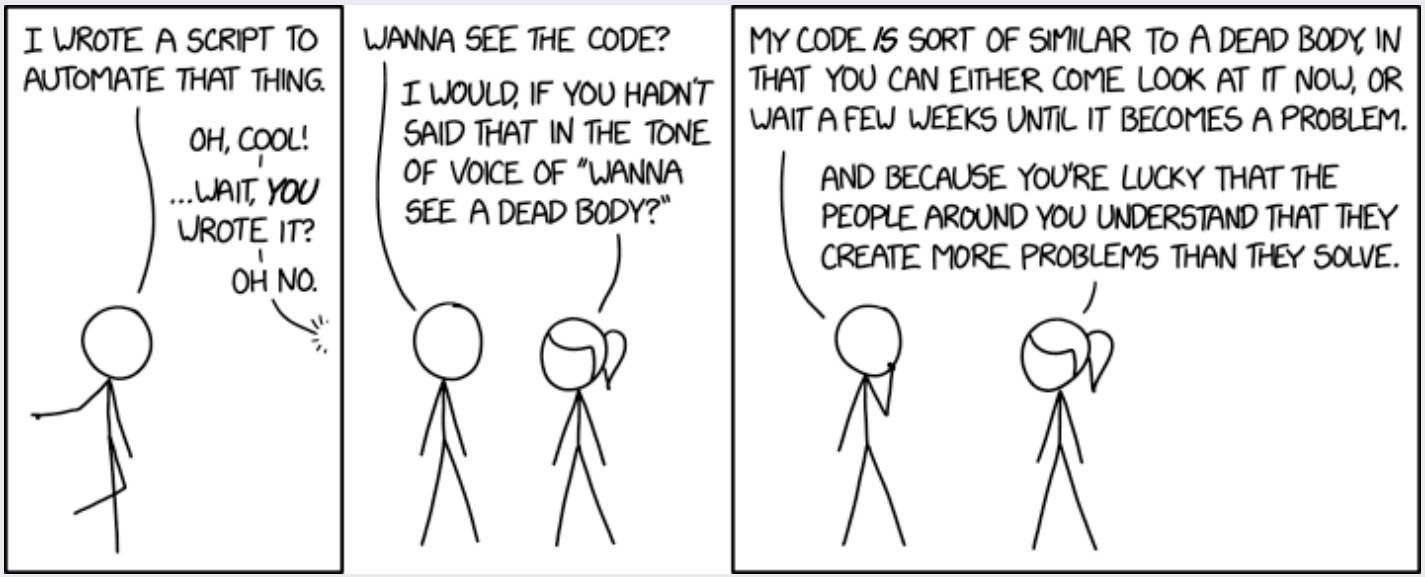
```
g ← g %>%
  # Node attributes ====
  set.vertex.attribute("color.background", value = "grey") %>%
  set.vertex.attribute("color.border", value = "black") %>%
  set.vertex.attribute("borderWidth", value = 2) %>%
  set.vertex.attribute("size", value = degree(.)) %>%
  set.vertex.attribute("label", value = V(.)$name) %>%
  # Edge attributes ====
  set.edge.attribute("width", value = scales::rescale(edge_betweenness(.
  set.edge.attribute("color", value = "slategrey") %>%
  set.edge.attribute("smooth", value = FALSE) %>%
  set.edge.attribute("shadow", value = TRUE)
```

visNetwork with Igraph Visualization

.center[<!iv id="htmlwidget-fe3960ecf0ec13583780" style="width:504px;height:504px;" class="visNetwork html-widget">

<!

Questions?



Dan Cunningham - dtcunnin@nps.edu

Christopher Callaghan - cjcallag@nps.edu