Post01

Rahul Misra

10/31/2017

Network Visualization with ggplot2

What is network visualization:

Networks and graphs are an extremely important way of understanding data. Graphs are mathematical structures used to model pairwise relations between objects. Typically, a graph consists of vertices (also called nodes and points) which are connected by edges (also known as arcs or lines). In network and computer science, a network can be defined as : a graph in which nodes and/or edges have attributes. Furthermore, the edges in a network are used to show relations between objects. Networks are everywhere around us and form an integral parts of our lives, whether we know it or not. From a transportation map to a social network to the internet, networks can be seen all around us and understanding these networks can give us a lot of information. The physical mapping of a network is known as network visualization. Here is an anecdote from AT&T Labs Infoviz research group about network graphs There is information in the connections. A glance is enough to identify nodes with the most links, nodes straddling different subgroups, and nodes isolated by their lack of connections. Corporations might look at a graph to verify that marketing and sales are communicating, urban planners to monitor the interconnectedness, or isolation, of neighborhoods, biologists to discover interactions between genes, and network analysts to monitor security.

What is ggplot2:

Ggplot2 is a plotting system for R and a grammar of graphics, which according to Hadley Wickham tries to take the good parts of base and lattice graphics and none of the bad parts. Grammar of graphics refers to a general scheme for data visualization which breaks up graphs into semantic components such as scales and layers. Released in 2005, ggplot2 is one of the most popular R packages today and has several extensions.

Implementations of Network Visualizations:

Network visualization can be implemented in a number of different ways in ggplot2, however, for in this post I will be focusing on two main ways. First, we can implement this using a wrapper function known as ggnet2. For those unaware of what a wrapper function is, a wrapper function can de defined as a subroutine in a software library or a computer program whose main purpose is to call a second subroutine or a system call with little or no additional computation. Another way to implement network visualizations is using geomnet which wraps all networks structures, including vertices, edges and vertex labels into a single geom.

Ggnet2:

Introduction: The ggnet2 function is one whose purpose is to plot network objects such as vertices, nodes and edges as ggplot2 objects. Keeping in mind the idea of coercion, inputs can be any objects that can be coerced into the network class ie. adjacency matrices, adjacency lists etc.

Installation:

```
ggnet2 is available through the GGally package:

  install.packages("GGally")
  library(GGally)

It is also available in its own standalone package :

    devtools::install_github("briatte/ggnet")
           library(ggnet)
```

Additionally for ggnet2 to work, the following libraries/dependencies are needed :

```
 library(network)
```

library(sna)

library(ggplot2)

More about ggnet2:Ggnet2 offers a large number of network visualization functionalities which can be accessed in a single function call and can be easily understood by users who might not be extremely experienced in using ggplot2. As mentioned before the input object for ggnet2 can be an object of the network class or an object that can be coerced into network class objects.

Color and size: GGnet2 offers an easy way to modify and alter node sizes, node colors, edge sizes and edge colors as can be seen in the code below :

Code :

ggnet2(net, node.size = 6, node.color = “black”, edge.size = 1, edge.color = “grey”)

(image 1)

Additionally it is possible to pass in multiple node colors or a vector of node colors:

Code :

ggnet2(net, size = 6, color = rep(c(“tomato”, “steelblue”), 5))

(image 2)

Another interesting way to color nodes is by providing a grouping. An example of this is through the assigning of a vertex attribute phono which indicates whether the name of the vertex is a vowel of consonant :

Code :

net %v% “phono” = ifelse(letters[1:10] %in% c(“a”, “e”, “i”), “vowel”, “consonant”)

If the user passes the name of the vertex attribute into the color argument, the argument will map the colors of the nodes as can be seen below:

Code:

ggnet2(net, color = “phono”)

(image 3)

In a similar manner, node sizes can also be varied based on an input argument. However rather than using the color argument of ggnet2, the size argument of ggnet2 will have to be used:

Code :

```
ggnet2(net, size = "phono")

ggnet2(net, size = "phono", size.palette =   c("vowel" = 10, "consonant" = 1))
```

(image 4)

Node placement and the Mode argument :The Mode argument in ggnet2 controls how the nodes in the graph are placed. By default the nodes are placed with the Fruchterman-Reingold force-directed algorithm, however this can be changed and all node placements in the sna package can be used:

Code :

ggnet2(net, mode = “circle”)

Node labels: Using the label argument, nodes in a ggnet2 network can be labeled using a vertex attribute, vertex name or any other such label.

Code:

```
ggnet2(net, label = TRUE)

ggnet2(net, label = "xlabs")

ggnet2(net, label = 0:5)
```

These node labels whose size is set to the half of the node can also have their sizes varied:

Code :

ggnet2(net, size = 12, label = TRUE, label.size = 5)

Likewise the labels color and alpha can also be modified using label.color and label.alpha respectively.

Node Shapes: Much like in ggplot2, node shapes can be altered in ggnet2 as well.

Code :

ggnet2(net, color = “phone”, shape = 15)

(image 5)

Real Life example of ggnet2 of French MPs on Twitter :

The following example contains a dataset of 339 French Members of Parliament (MPs), and the ties that they formed by following each other on Twitter.

Code :

```
r = "https://raw.githubusercontent.com/briatte/ggnet/master/"

# read nodes
v = read.csv(paste0(r, "inst/extdata/nodes.tsv"), sep = "\t")
names(v)

# read edges
e = read.csv(paste0(r, "inst/extdata/network.tsv"), sep = "\t")
names(e)

# network object
net = network(e, directed = TRUE)

# party affiliation
x = data.frame(Twitter = network.vertex.names(net))
x = merge(x, v, by = "Twitter", sort = FALSE)$Groupe
net %v% "party" = as.character(x)

# color palette
y = RColorBrewer::brewer.pal(9, "Set1")[ c(3, 1, 9, 6, 8, 5, 2) ]
names(y) = levels(x)

# network plot
ggnet2(net, color = "party", palette = y, alpha = 0.75, size = 4, edge.alpha = 0.5)
```

(Image 6)

Geomnet:

Introductions: As mentioned before geomnet is a package built on top of ggplot2 that provides a geom called geom_net to visualize graphics and networks. geomnet implements network visualization in a single ggplot2 layer.

Dependencies/Imports:

Ggplot2,

sna,

network,

dplyr,

tidyr,

readr

plotly

Layout Parameter : The layout.alg parameter in geomnet takes a character value corresponding to the possible network layouts in the sna package much like ggnet2. Unlike ggnet2 the default layout for geomnet is kamadakawai.

Code :

geom_net(mapping = NULL, data = NULL, stat = “net”, position = “identity”, show.legend = NA, na.rm = FALSE, inherit.aes = TRUE, layout.alg = “kamadakawai”)

Vertice Aesthetics: Much like in ggplot2 the aesthetics of a vertice can be changed using standard parameters such as colour, size, shape, alpha, x, and y.

Labels: The labelon argument is a logical parameter which if set to true will label nodes with their ID???s. Furthermore much like in ggplot2 the aes option can be used to label nodes. Likewise labelcolor, fontsize, vjust, hjust and other such options avaibale in ggplot2 are also available in geomnet.

Example - Blood donation diagram:

Code :

```
library(geomnet)

data(blood)

ggplot(data = blood$edges, aes(from_id = from, to_id = to)) +

  geom_net(colour = "darkred", layout.alg = "circle",    labelon = TRUE, size = 15, directed = TRUE, vjust = 0.5,
labelcolour   = "grey80", arrowsize = 1.5, linewidth = 0.5, arrowgap = 0.05 + selfloops = TRUE, ecolour = "grey40"
) +
theme_net()
```

(Image 7)

Conclusion: The aim of this post is to introduce students of Stats 133 to a possible application of ggplot2 ie. network visualization. Additionally this post aims to educate students about the usage of two extensions namely geomnet and ggnet2 to implement network visualizations. With the provided descriptions, images and code I hope the readers of this post are able to gain a good understanding of network visualization using ggplot2. Please note the images for this file are located in the folder Post01_Images

References:

https://en.wikipedia.org/wiki/Graph_theory

https://journal.r-project.org/archive/2017/RJ-2017-023/RJ-2017-023.pdf

https://flowingdata.com/2010/11/17/why-network-visualization-is-useful/

https://briatte.github.io/ggnet/#node-labels

https://www.rstudio.com/wp-content/uploads/2015/03/ggplot2-cheatsheet.pdf

https://github.com/sctyner/geomnet

https://www.rdocumentation.org/packages/geomnet/versions/0.2.0/topics/geom_net

http://ggplot2.org