A Geometry for Network Visualization in ggplot2

Sam Tyner and Heike Hofmann

1 Introduction

The three necessary elements of any network visualization are the vertices, the edges, and the layout. But once those three items are visualized, we usually find our visualization to be lacking. We may want to color the vertices (or edges) by some sort of grouping variable, or we may want to make vertices of degree 10 twice the size of vertices of degree 5. Many `R` packages already exist for network analysis and visualization such as `igraph` by Gabor Csardi, `sna` by Carter T. Butts, and `network` by Carter T. Butts et al. but we have found these packages to have unintuitive or burdensome methods for customizing the colors, sizes, etc of the vertices and edges of the network. For instance, the `igraph` package allows for coloring vertices by groups but the user must assign the colors to each vertex individually as opposed to assigning color by a grouping factor variable.

The `GGally` package by Barret Schloerke et al. contains a very useful function written by François Briatte and Moritz Marbach called `ggnet` that does allow for fairly straightforward customization of these 3 necessary graph attributes. The `ggnet` function, however, requires the graph input to be a `network` object according to the `network` package. Our goal is to build off of the `ggnet` function and present an intuitive `geom` for network visualization within the `ggplot2` framework, without the need for objects other than simple data frames.

1.1 Data Structure

In order for this geometry to work using data frames, there need to be two separate data frames given to the `geom_net` function: one for the edge information and one for the vertex information. The vertex data frame should contain all the relevant vertex information. *The only necessary variables are the vertex names/labels and the vertex degree (if undirected) or the vertex in degree and out degree (if directed). The names and degrees are stored in columns and the rows are the vertices. Other values of interest, such as grouping variables, for each vertex should be stored as columns in this dataframe, with an observation for each vertex.*

The edge data frame should contain all the relevant edge information. *The only necessary variables are the "from vertex" and "to vertex" for each edge in the network. The From and To vertices should be listed according to the names of the vertices in the vertex information data frame. Other variables may also be included for each edge, such*

as the edge weight or grouping variable. As before, the variables of interest are columns in the data frame and the rows are each edge in the network.[1]

1.2 Vertex Aesthetics

In our geometry, the possible vertex aesthetics mimic the usual aesthetics in `ggplot2` for points. The `vlabel` aesthetic is the name or number assigned to each vertex in order to uniquely identify it. This aesthetic is most useful for smaller network objects where all vertex names can be printed on or near their corresponding vertices and still be readable by the human eye. The `vfill` aesthetic can be an identity color (e.g. `color = "red"` outside of the `aes` statement) to change the fill color of all vertices, or it can take a factor variable which identifies each vertex as belonging to one of several groups, and it fills the vertices of the graph according to this grouping. It can also fill the vertices according to the different values of some numeric variable, such as degree. The colors are set to the default color palette in `ggplot2`, and the palette used can be changed in the usual way within the `ggplot2` framework, by using `scale_fill_brewer`. The `vcolor` aesthetic operates in the same way as the `vfill` aesthetic, but it changes the outline color of the vertex instead of the fill color of the vertex, which is in line with the `ggplot2` use of the `color` and `fill` aesthetics. The `vsize` aesthetic can also be set to an identity value to increase or decrease the size of all vertices in the graph, or it can be set to the degree of the vertices in undirected graphs or the in-degree (or out-degree) of directed graphs. [2] It can also be set to any other numeric variable associated with the vertices in the network. The `vshape` aesthetic is for changing the shape of the vertices from the default circle to other shapes, like square or triangle. It can also change the shape of the vertices according to some grouping variable. Finally, the `valpha` aesthetic will change the vertex transparency to a set value, like 1/10, or to some numeric variable of interest that takes on the values $[0, 1]$.

Evidently, we created all of these aesthetics to fit in exactly to the `ggplot2` grammar of graphics. We also created the edge aesthetics in a similar fashion.

1.3 Edge Aesthetics

Again, our edge aesthetics mimic the familiar `ggplot2` aesthetics, this time for lines. The `elabel` aesthetic is the name or number assigned to each edge. This aesthetic is most useful for visualizing random graphs or Markov processes, where each edge probability is of interest. The `ecolor` aesthetic changes the color of all of the edges according to the identity function or can change the color according to a grouping variable. It can also be used to color the edges in random graphs or Markov processes according to the probability of each edge. The `elinetype` aesthetic can also be used with a grouping variable to change the line type of each edge in the graph from solid to one of the many

[1] I'm not 100% certain of all of this yet, but so far it seems to me to be the best way to think about networks in terms of data frames.

[2] Sam's questions/notes: Is "degree" going to be a column in our data frame? Or will we have to write something to detect degree size?

different types of dotted lines in `ggplot2`. The `esize` aesthetic can also be set to the weight or probability of each edge, or it can be changed for all edges with the identity function. The `ealpha` aesthetic can change the edge transparency to a set value, or to some numeric variable of interest that takes on the values $[0, 1]$, such as edge weight or probability. Finally, the `directed` aesthetic is a logical value that identifies whether or not the network is directed. When this value is true, the arrows change size, color, etc. with the previous aesthetics.

1.4 Layout Aesthetics

*There are many layouts to choose from, and the default layout changes according the size of the graph. For smaller networks, with less than x vertices and y edges, the default layout is INSERT DEFAULT HERE. For larger networks with more than x vertices and y edges, the default layout is INSERT DEFAULT HERE. **Now justify these two statements with research to be done.**** The `layout` aesthetic can also be changed to any of the layouts also found in the `igraph` package, such as Kamada-Kawai, circular, or Fructerman-Reingold. Finally, the background color of the graph can be changed with the `bgfill` aesthetic. This aesthetic takes only the identity colors, not additional variables.[3]

[3]Sam's notes: I'm not sure if what I've laid out here is possible or even reasonable. I have not yet found any research into what layout is best based on number of edges, number of vertices, density of the network, or anything else.