Network Analytics Amazon Products

Christine Lim

October 14, 2019

1. Download data

Data is from Amazon, where the nodes in this network are Amazon products, including books, movies, and music. The edges in this network represent hyperlinks from a given product's landing page to the landing pages of those products most frequently co-purchased with the given product.

```r
#set working directory
setwd("C:/Users/cvlim/Desktop/NYU Coursework/Network Analytics/data/data")

#import tables
graph_complete<-read.table("graph_complete.txt")
graph_subset_rank1000<-read.table("graph_subset_rank1000.txt")
graph_subset_rank1000_cc<-read.table("graph_subset_rank1000_cc.txt")
id_to_titles<-read.table("id_to_titles.txt", header=TRUE, fill=TRUE)
```

2. Network Structure visualization

```r
#Download and install igraph

#install.packages("igraph")
library(igraph) #igraph is a library and R package for network analysis.
```

1) Plot the network using the information in the file graph_subset_rank1000.txt. Note that this is not the complete network, but only a subset of edges between top-ranked products. By visualizing the graph, you get an idea of the structure of the network you will be working on. In addition to plotting, comment on anything interesting you observe.

```r
#Turn network into igraph object
net<-graph_from_data_frame(graph_subset_rank1000, directed=F) #undirected network for purposes of visua


#as_data_frame(net, what="edges") #edges
#as_data_frame(net, what="vertices") #nodes
V(net) #nodes
```

```
## + 1355/1355 vertices, named, from b0fe5fb:
##    [1] 411653 68951  236897 265343 472765 153184 424919 469074 480433
##   [10] 220748 42974  491768 291610 105110 67371  120884 212405 355494
##   [19] 29203  20195  444150 285340 5039   325446 530653 265598 216323
##   [28] 45094  96553  342416 445728 267451 287962 172503 265468 88060
##   [37] 401507 353676 443284 33304  95210  124568 374639 48316  535849
##   [46] 367733 83582  468264 84492  572    87559  488883 209067 97692
##   [55] 35109  531651 108364 108187 13198  25341  65731  117196 30695
##   [64] 54800  178132 458306 30639  324938 337679 214842 197455 61751
##   [73] 397309 97533  429060 100805 313421 334110 420909 478494 514999
##   [82] 281633 178948 496408 32865  270504 483960 23117  511354 232354
## + ... omitted several vertices
```

```
E(net) #edges
```

```
## + 2611/2611 edges from b0fe5fb (vertex names):
##  [1] 411653--94292  68951 --478494 236897--265343 236897--265343
##  [5] 236897--472765 153184--172503 424919--172503 469074--48638
##  [9] 480433--48638  220748--42974  220748--42974  491768--105110
## [13] 291610--105110 491768--105110 67371 --78848  120884--390297
## [17] 212405--390297 355494--325446 29203 --349384 20195 --349384
## [21] 444150--349384 285340--349384 5039  --246337 325446--21728
## [25] 530653--21728  265598--21728  355494--21728  216323--354795
## [29] 45094 --96553  45094 --96553  342416--49527  445728--363808
## [33] 267451--30639  287962--20466  153184--172503 153184--265468
## [37] 153184--88060  153184--401507 153184--424919 153184--353676
## + ... omitted several edges
```

```
vcount(net)
```

```
## [1] 1355
```

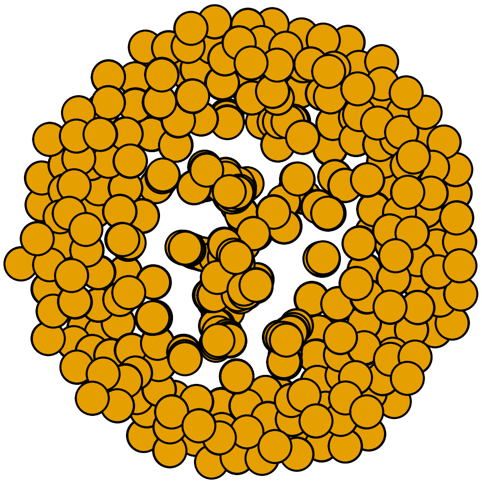
```
ecount(net)
```

```
## [1] 2611
```

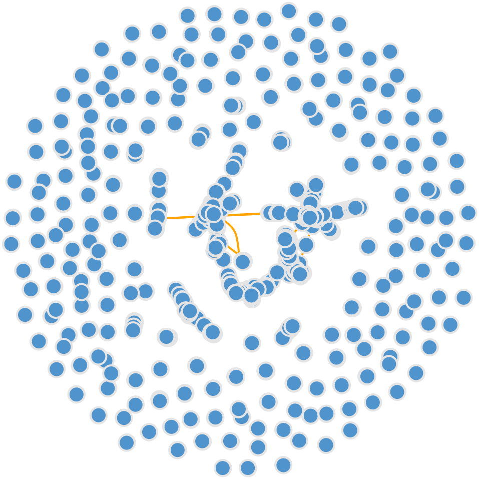
There are 1,355 nodes and 2,611 edges

1) Plotting the network of subset of edges between top-ranked products

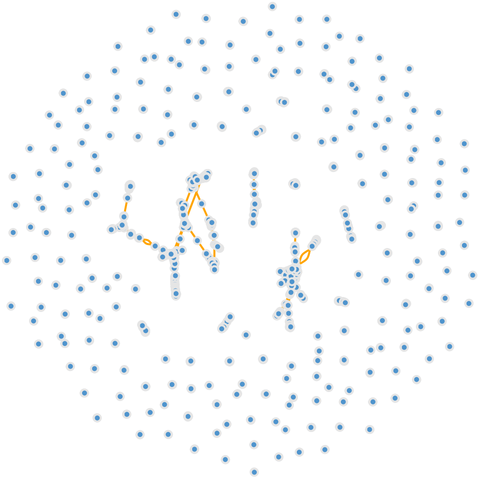
```
#plot object auto layout

#original plot
plot(net, directed=F,
     vertex.label=NA,
     layout=layout.auto)
```



```
#changing vertex size & color
plot(net, directed=F,
     vertex.size=7,
     vertex.label=NA,
     edge.arrow.size=5,
     layout=layout.auto, edge.color="orange",
     vertex.color="steelblue3", vertex.frame.color="grey90")
```



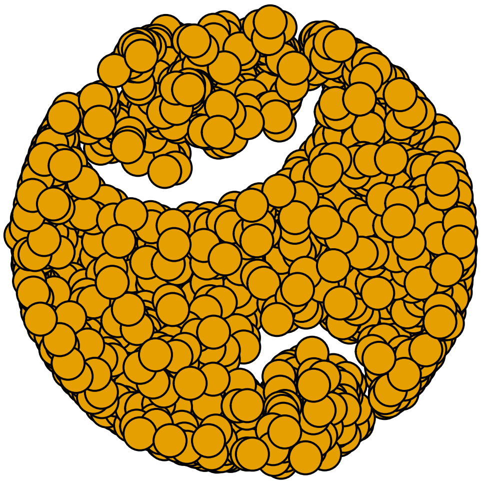
```
#decreasing vertex size to make edges more apparent
plot(net, directed=F,
     vertex.size=3,
     vertex.label=NA,
     edge.arrow.size=5,
     layout=layout.auto, edge.color="orange",
     vertex.color="steelblue3", vertex.frame.color="grey90")
```



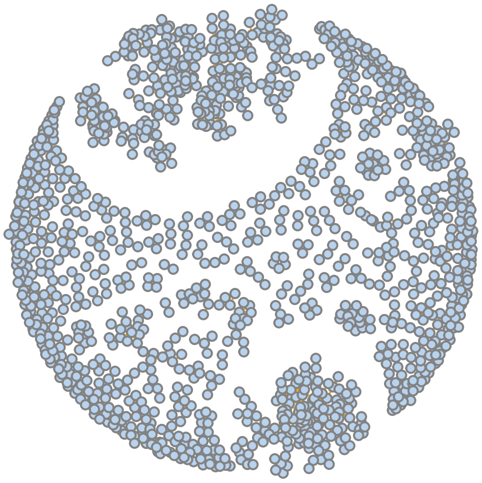
For the auto layout, it looks like many of the nodes are self-referring on the outer sides of the network, while there are a few large clusters in the center.
In terms of visualization, changing the colors and vertex size makes it easier to identify the edges, since it is difficult to see them when the node size is bigger.
Further reducing the size of the vertices gave the plot a "zooming out" effect and made it much clearer easier it is to identify structures and shapes in the network.

Plot object using kamada kawai layout

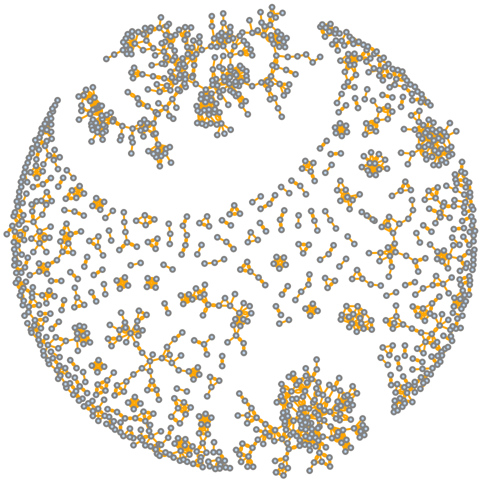
```r
#original size
plot(net, directed=F,
     vertex.label=NA,
     edge.arrow.size=2,
     layout=layout.kamada.kawai)
```



```r
#reducing vertex size to 4
plot(net, directed=F,
     vertex.size=4,
     vertex.label=NA,
     edge.arrow.size=2,
     layout=layout.kamada.kawai,
     edge.color="orange",
     vertex.color="slategray2",
     vertex.frame.color="grey51")
```



```
#further reducing vertex size to 2
plot(net, directed=F,
     vertex.size=2,
     vertex.label=NA,
     edge.arrow.size=2,
     layout=layout.kamada.kawai,
     edge.color="orange",
     vertex.color="slategray2",
     vertex.frame.color="grey51")
```

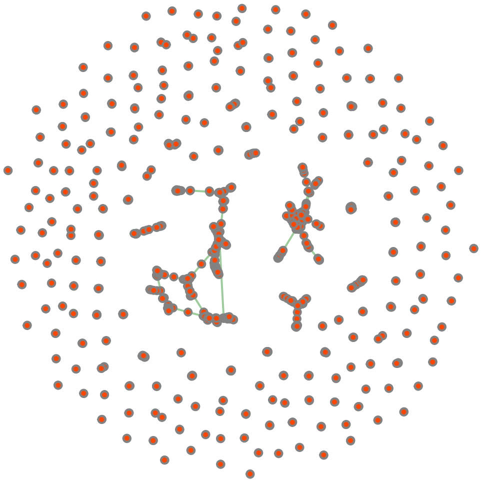


Here the clusters are much more distinct, and seem drawn together to form larger "shapes". Instead of having large clusters concentrated in the center, there looks like two major clusters on the top and botton of the network with nodes that are highly connected, while nodes with fewer edges are gravitated towards the center.

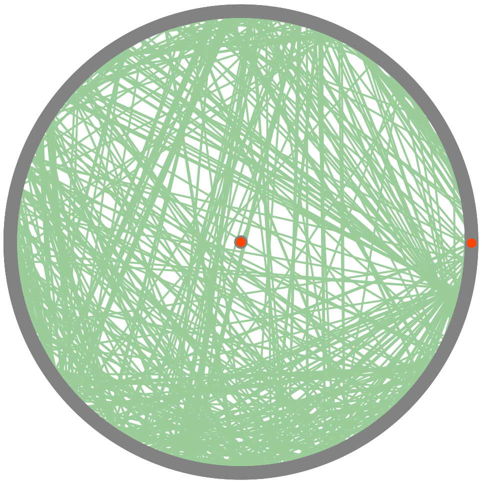
Again, reducing the size of the vertices made viewing the edges more apparent and allows us to visualize individual network structures more easily.

Playing with other layouts:

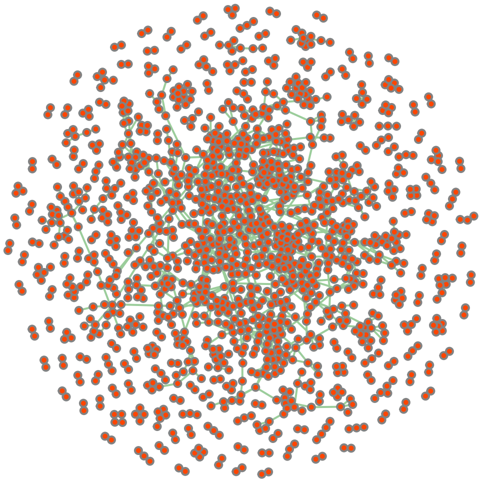
```
#remove nodes that only refer to itself

net2<-simplify(net, remove.multiple = T, remove.loops=T,
               edge.attr.comb = igraph_opt("edge.attr.comb") )

plot(net2, directed=F, vertex.size=3, vertex.label=NA,
     edge.color="darkseagreen3", vertex.color="orangered",  vertex.frame.color="grey51")
```



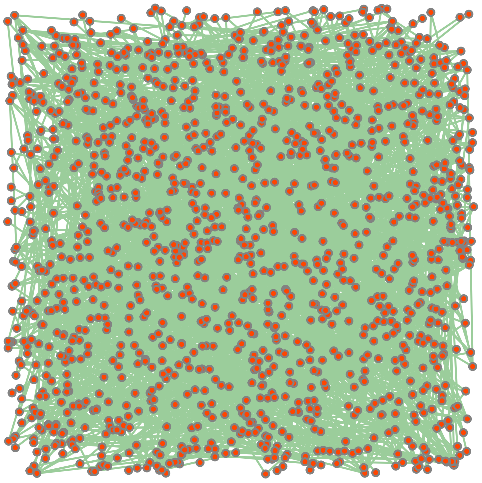
```r
plot(net2, directed=F, vertex.size=5, vertex.label=NA, edge.arrow.size=2,
     layout=layout_as_star,
     edge.color="darkseagreen3", vertex.color="orangered",  vertex.frame.color="grey51") #star layout
```



```
plot(net2, directed=F, vertex.size=3, vertex.label=NA, edge.arrow.size=2,
     layout=layout.graphopt,
     edge.color="darkseagreen3", vertex.color="orangered",  vertex.frame.color="grey51")  #graphopt lay
```



```
plot(net2, directed=F, vertex.size=3, vertex.label=NA, edge.arrow.size=2,
     layout=layout_with_lgl,
     edge.color="darkseagreen3", vertex.color="orangered",  vertex.frame.color="grey51")  #graphopt lay
```



2) Now, use the file graph subset rank1000 cc.txt to plot only the largest connected component in the above network.

```
net_cc<-graph_from_data_frame(graph_subset_rank1000_cc, directed=F)


# as_data_frame(net_cc, what="edges") #edges
# as_data_frame(net_cc, what="vertices") #nodes
V(net_cc) #nodes
```

```
## + 292/292 vertices, named, from e495421:
##   [1] 415305 297722 182411 539734 267500 539367 538546 540230 539964 516279
##  [11] 540153 542414 171675 538785 447339 82836  48611  73768  95554  73437 
##  [21] 75527  312068 135171 241213 381095 541914 198466 423769 1299   136432
##  [31] 529269 262229 128602 273044 111494 52374  539257 539854 419575 340252
##  [41] 485290 111591 63986  150174 420056 201516 440768 55147  190050 92686 
##  [51] 510545 313436 11731  287328 77548  233148 345354 454037 398931 463542
##  [61] 178396 273273 541902 135011 113827 372908 188833 139989 541371 338680
##  [71] 370635 330960 541281 51791  540248 243066 539088 265550 480146 539872
##  [81] 178718 327275 538640 239251 70526  111255 139837 541406 213826 538249
##  [91] 42374  333341 247731 319311 96529  274842 480014 542075 54049  25224 
## + ... omitted several vertices
```

```r
E(net_cc) #edges
```

```
## + 604/604 edges from e495421 (vertex names):
##  [1] 415305--112822 297722--116802 182411--116802 539734--273044
##  [5] 267500--273044 539367--538785 538546--538785 540230--540153
##  [9] 539964--540153 516279--540153 540230--540153 540230--542414
## [13] 540230--539964 171675--541064 539367--538785 415305--539367
## [17] 539367--447339 539367--82836  539367--48611  73768 --274142
## [21] 95554 --73437  540153--542414 540230--542414 542414--73437
## [25] 539964--542414 542414--95554  75527 --426430 312068--128602
## [29] 135171--128602 241213--128602 381095--12274  541914--12274
## [33] 198466--12274  423769--12274  1299  --12274  136432--12274
## [37] 529269--12274  262229--12274  312068--135171 135171--128602
## + ... omitted several edges
```

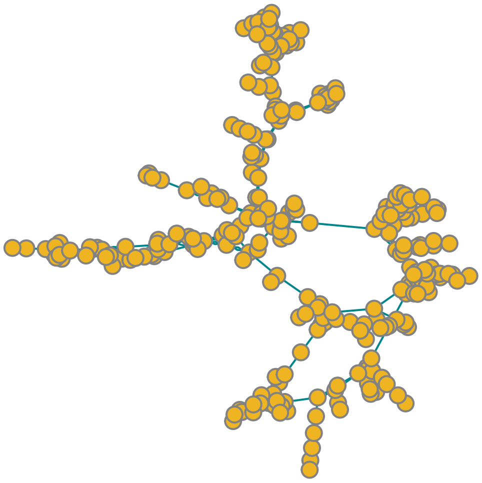
```r
vcount(net_cc)
```

```
## [1] 292
```

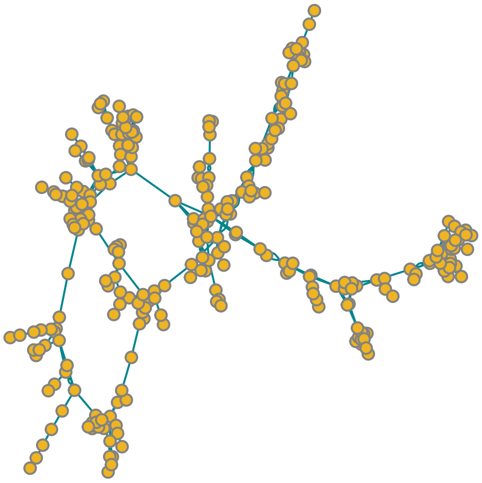
```r
ecount(net_cc)
```

```
## [1] 604
```

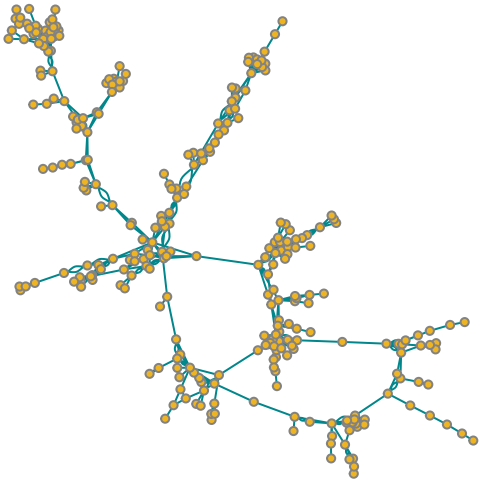
```r
#plot object auto layout
plot(net_cc, directed=F,vertex.size=7,
     vertex.label=NA,
     edge.arrow.size=2,
     layout=layout.auto,
     edge.color="turquoise4", vertex.color="goldenrod2", vertex.frame.color="grey51")
```



```
#reducing vertex size
plot(net_cc, directed=F,
     vertex.size=5,
     vertex.label=NA,
     edge.arrow.size=2,
     layout=layout.auto,
     edge.color="turquoise4", vertex.color="goldenrod2", vertex.frame.color="grey51")
```



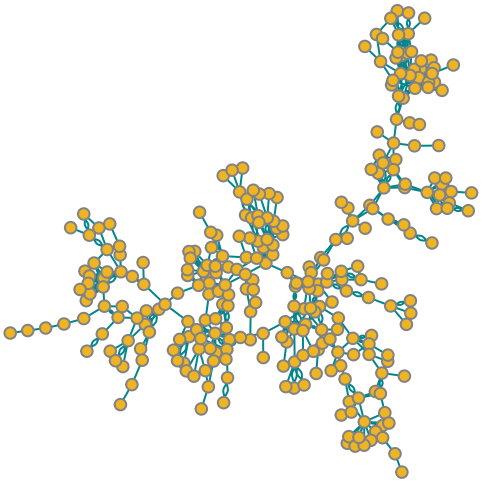
```
#further reducing vertex size
plot(net_cc, directed=F,
     vertex.size=3.5,
     vertex.label=NA,
     edge.arrow.size=2,
     layout=layout.auto,
     edge.color="turquoise4", vertex.color="goldenrod2", vertex.frame.color="grey51")
```



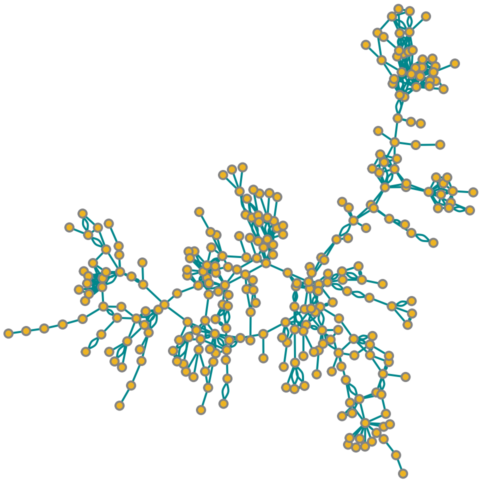
Here we can see that all of the nodes are connected in the same “path”, unlike the larger dataset where there were some completely unlinked clusters. Even though all the nodes are linked, they form sub-clusters where some edges are longer than others, giving an effect of branches off a main path of edges.

Plot subset using kamada kawai layout:

```
#plot object kamada kawai
plot(net_cc, directed=F,
     vertex.size=5,
     vertex.label=NA, edge.arrow.size=2,
     layout=layout.kamada.kawai,
     edge.color="turquoise4",
     vertex.color="goldenrod2",
     vertex.frame.color="grey51")
```



```
plot(net_cc, directed=F,
     vertex.size=3.5,
     vertex.label=NA, edge.arrow.size=2,
     layout=layout.kamada.kawai,
     edge.color="turquoise4",
     vertex.color="goldenrod2",
     vertex.frame.color="grey51")
```



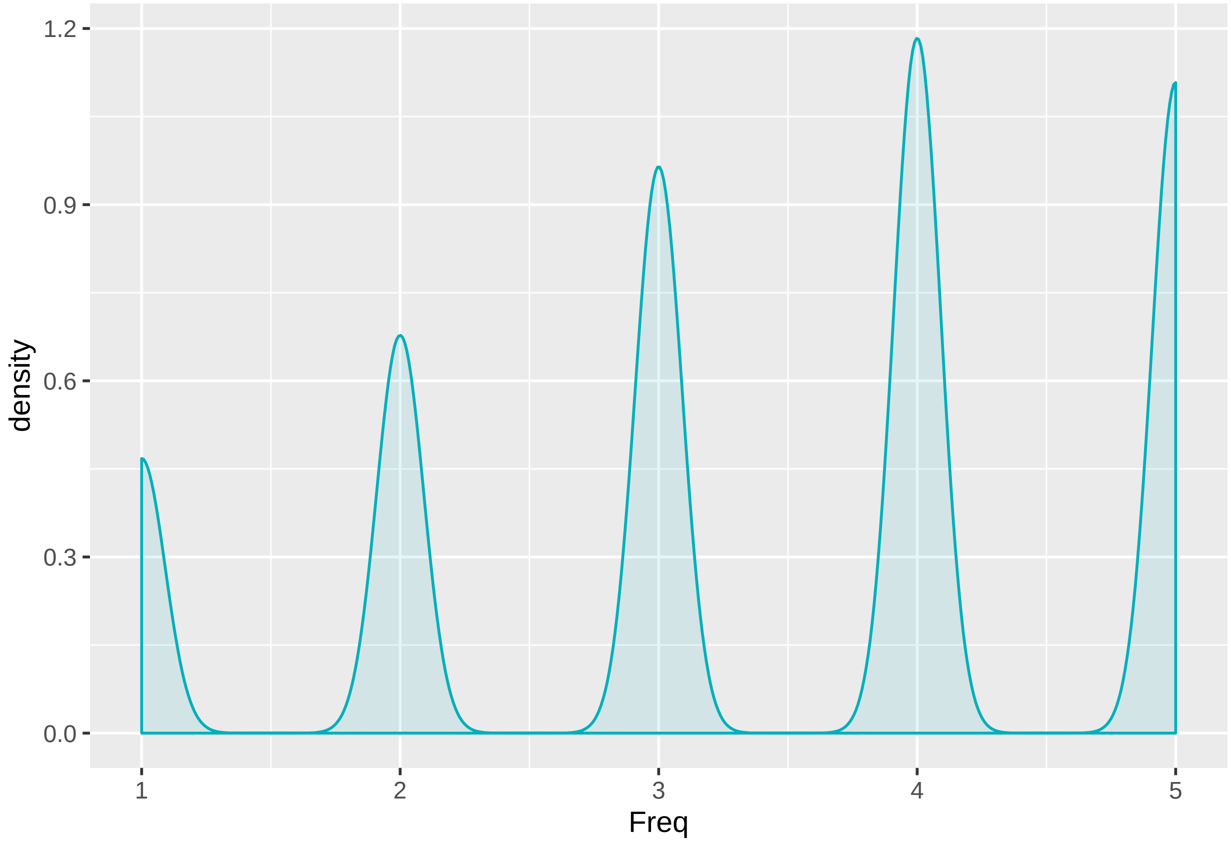
We can see that using kamada kawai, the clusters are again more distict. There are also branches off a main "path", but the nodes on the "branches" seem more spread out.

3. Data Analysis

1) Plot the out-degree distribution of our dataset (x-axis number of similar products, y-axis number of nodes)

```
#compute out-degree for each product
out_degree<-as.data.frame(table(graph_complete$V1))
#counting number of outgoing links from product a to product page b

#out_degree2<-as.data.frame(table(graph_subset_rank1000$V1)) #using subset data
```

```
#plotting the distribution

library(ggplot2)

ggplot(out_degree, aes(x=Freq)) +
  geom_density(fill = "#00AFBB", colour = "#00AFBB", alpha = 0.1)
```



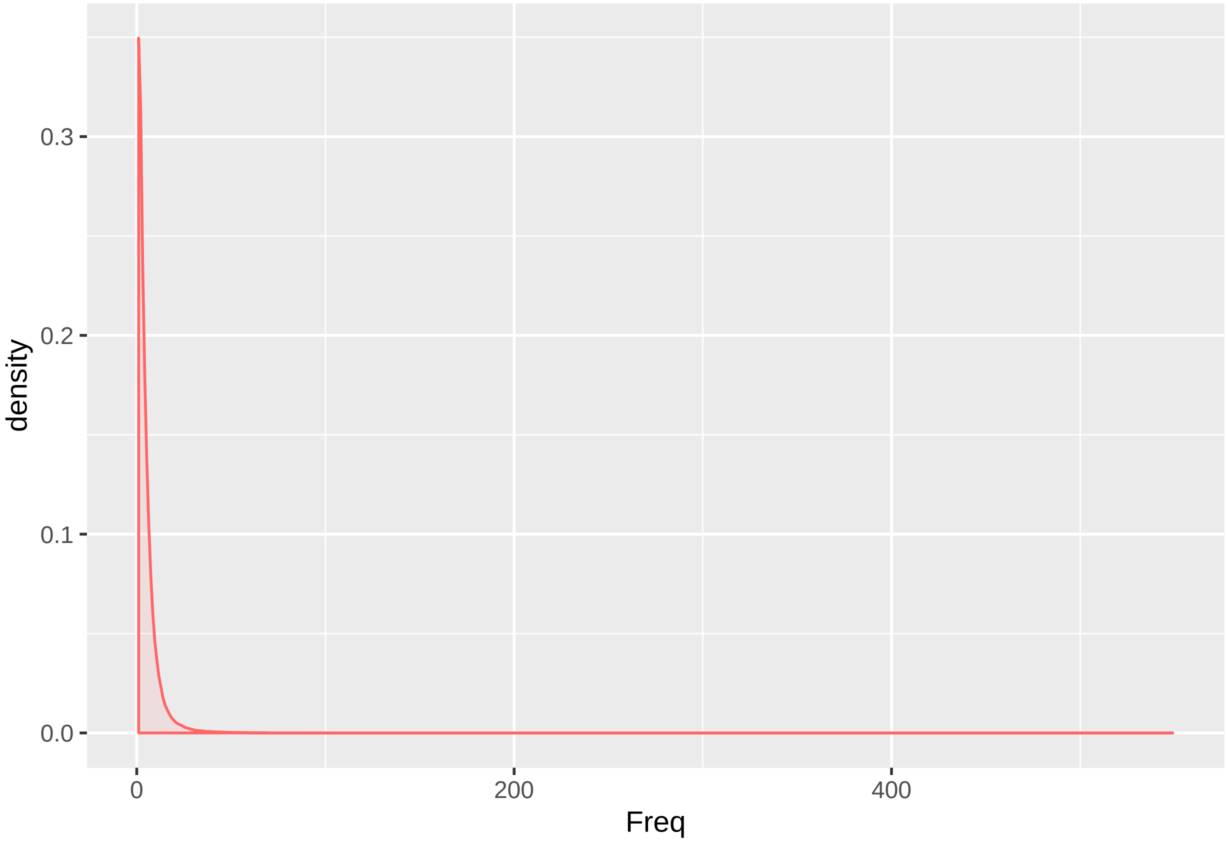
```
#ggplot(out_degree2, aes(x=Freq)) + geom_density() #using subset data
```

There is a maximum of five outbound links showing that Amazon links on an outbound node page referring to other pages are limited to 5, and the most frequent occurrence of links is 4.

2) Plot the in-degree distribution of our dataset (x-axis number of similar products, y-axis number of nodes).

```
in_degree<-as.data.frame(table(graph_complete$V2)) #counting number of products(nodes)
```

```
#plotting the in-degree distribution

ggplot(in_degree, aes(x=Freq)) +
  geom_density(fill = "#FF6666", colour = "#FF6666", alpha = 0.1)
```

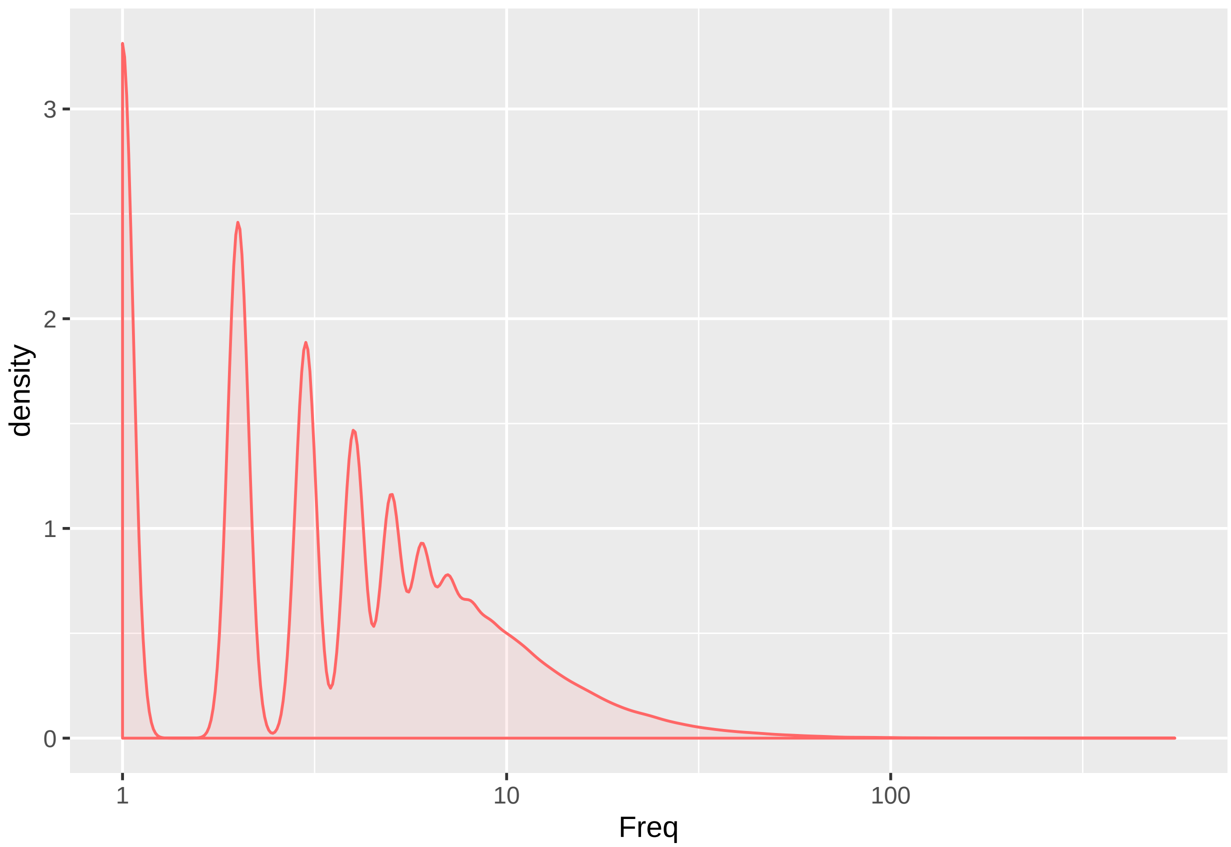


The in-degree number of page references are much more skewed with a long tail, ranging from 0 to >500 references.

3) Plotting in-degree on log scale

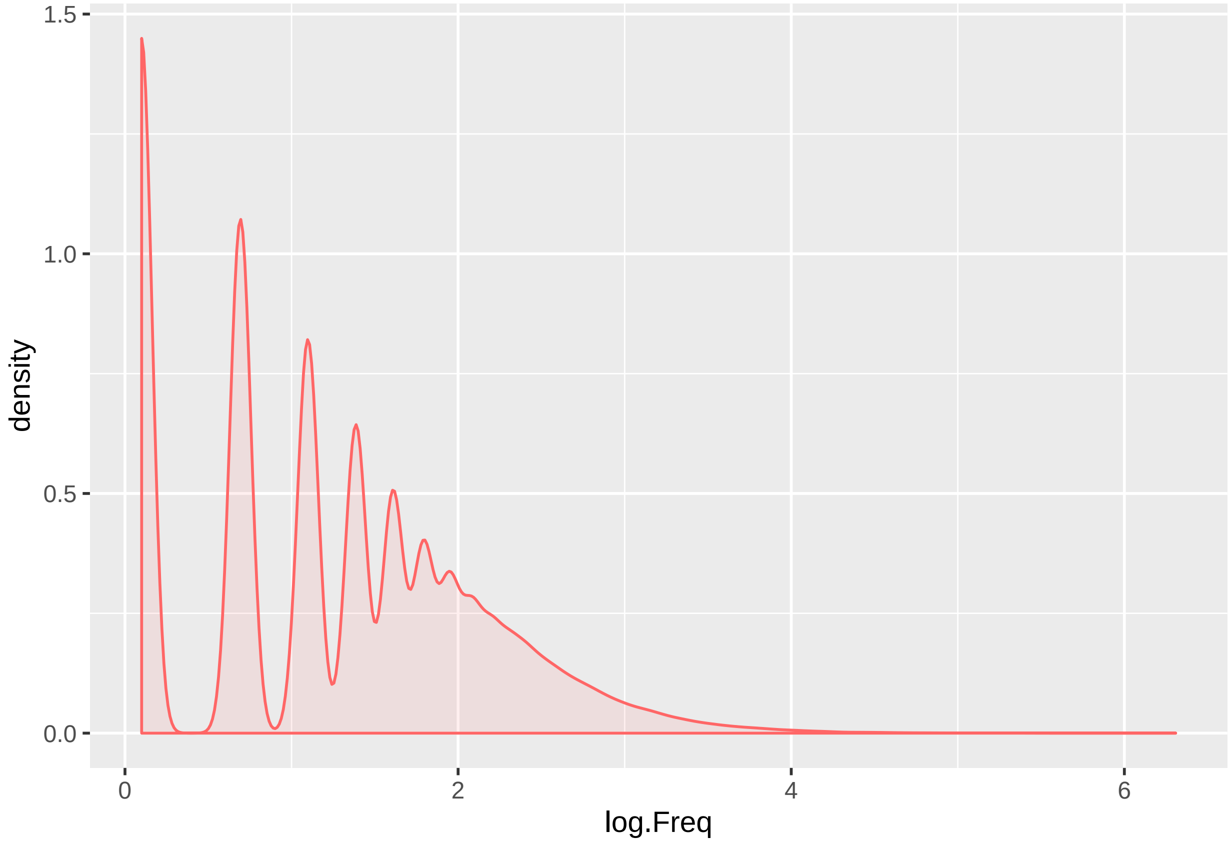
Transforming x-axis to log scale

```
#Plotting transformation of x-axis to log scale
ggplot(in_degree, aes(x=Freq)) +
  geom_density(fill = "#FF6666", colour = "#FF6666", alpha = 0.1) +
  scale_x_log10()
```



Replacing data with 0 to 0.1 and graphing

```
in_degree$log.Freq<-log(in_degree$Freq) #convert Frequency to log scale
in_degree$log.Freq[in_degree$log.Freq==0]<-0.1 #replace 0 with .1

ggplot(in_degree, aes(x=log.Freq)) +
  geom_density(fill = "#FF6666", colour = "#FF6666", alpha = 0.1)
```



A node(page) can have a range from 0 to >500 amount of references to it, but from scaling the original graph to the log graph, it can be seen that the majority of nodes have under 10 inbound links.

4) Compute the average number of inbound co-purchase links, the standard deviation, and the maximum. Comment on the result.

```
library(psych)

describe(in_degree)
```

```
##           vars      n      mean       sd   median   trimmed      mad min
## Var1*        1 237258 118629.50 68490.63 118629.5 118629.50 87939.68 1.0
## Freq         2 237258      5.19     6.76      3.0      3.95     2.97 1.0
## log.Freq     3 237258      1.23     0.87      1.1      1.17     1.03 0.1
##                 max     range skew kurtosis     se
## Var1*     237258.00 237257.00 0.00    -1.20 140.61
## Freq         549.00    548.00 9.17   299.03   0.01
## log.Freq       6.31      6.21 0.43    -0.34   0.00
```

Average number of inbound co-purchase links: 5.19

Standard deviation: 6.76

Maximum: 549

There is a large range in the frequency of links in the inbound co-purchase links. Since the max is so much greater than the mean, and the standard deviation is higher than the mean, there is a highly skewed distribution.

5. Report the names of the 10 products with the most inbound co-purchase links.

```
#names of the top 10 products with most inbound co-purchase links

library(dplyr)

top_10<-head(in_degree[order(in_degree$Freq, decreasing=T),], n=10)
names(top_10)[1]<-"id"

top_10_products<-merge(top_10, id_to_titles, by="id")
top_10_products<-top_10_products[order(-top_10_products$Freq),]
top_10_products$title
```

```
##  [1] Laura
##  [2] Diagnostic and Statistical Manual of Mental Disorders DSM-IV-TR (Text Revision) (Diagnostic and
##  [3] Publication Manual of the American Psychological Association, Fifth Edition
##  [4] The Great Gatsby
##  [5] 1001 Most Useful Spanish Words (Beginners' Guides)
##  [6] It Works
##  [7] Brown Bear, Brown Bear, What Do You See?
##  [8] Easy Spanish Phrase Book: Over 770 Basic Phrases for Everyday Use
##  [9] The Prince
## [10] The TEMPEST
## 336523 Levels:  ... Zzz ...: The Most Interesting Book You'll Ever Read About Sleep (Mysterious You)
```