Scale Free Network

Connor Lachance

4/15/2021

Use the following code to simulate a graph with 60 nodes from the Barabasi-Albert model and perform analysis

rm(list=ls())  
  
library(igraph)

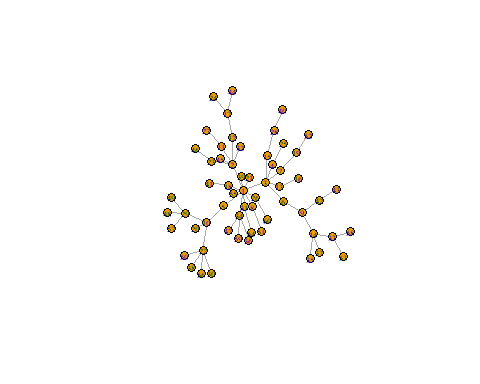
## Warning: package 'igraph' was built under R version 4.1.3

##   
## Attaching package: 'igraph'

## The following objects are masked from 'package:stats':  
##   
## decompose, spectrum

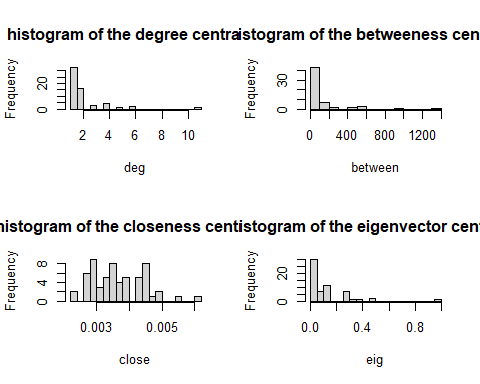
## The following object is masked from 'package:base':  
##   
## union

set.seed(1000)  
g.ba <-barabasi.game(60, directed=FALSE)  
plot(g.ba, layout=layout.kamada.kawai, vertex.size=10, vertex.label.cex=0.5)



#Calculate the degree, betweenness, closeness and eigenvetor centrality measures for each node. Plot the histograms of these centrality measures. Report the plot of histograms.

#find  
deg = degree(g.ba,mode='all')  
between= betweenness(g.ba)  
close=closeness(g.ba)  
eig=eigen\_centrality(g.ba)$vector  
all\_centrality=data.frame(deg,between,close,eig)  
  
#plot  
par(mfrow=c(2,2))  
hist(deg,breaks=15,main='histogram of the degree centrality')  
hist(between,breaks=15,main='histogram of the betweeness centrality')  
hist(close,breaks=15,main='histogram of the closeness centrality')  
hist(eig,breaks=15,main='histogram of the eigenvector centrality')

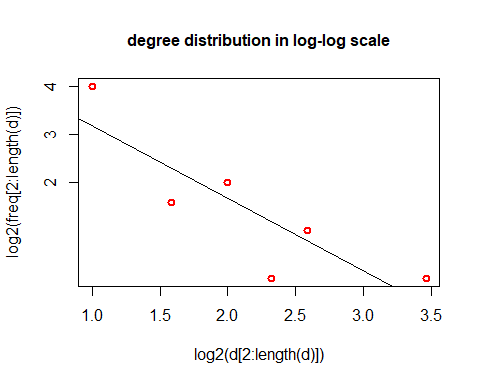


#plot the degree distribution on log scale as a scatter plot. Fit a linear model, and plot the fitted straight line on top of the scatter plot. Report the plot. Is this network scale-free?

require(igraph)  
freq= table(deg)  
d=as.numeric(names(freq))  
plot(log2(d[2:length(d)]), log2(freq[2:length(d)]), type='p', main='degree distribution in log-log scale',cex.main=1, col='red', lwd=2)  
res=lm(log2(freq[2:length(d)])~log2(d[2:length(d)]))  
res$coef

## (Intercept) log2(d[2:length(d)])   
## 4.691566 -1.510617

abline(a=res$coef[1],b=res$coef[2])

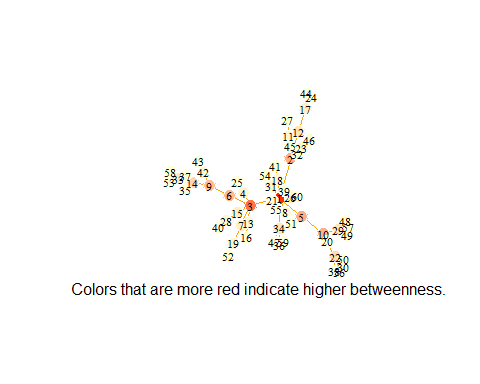
 #appears to be a pretty straight line, suggusting this network is scale-free

#Decorate the network. Use the node degrees to rescale the node size, and set the node colors using the betweenness measure (e.g., color the node with tomato if the betweenness is less than 400, gold if it is in the interval [400, 1000], and yellowgeen if it is greater than 1000). Report plot of the decorated graph.

library(igraphdata)

## Warning: package 'igraphdata' was built under R version 4.1.3

colrs <- heat.colors(length(unique(between)), alpha=1)  
fvalue <- colorRamp(c(colrs[15],colrs[3]))  
V(g.ba)$color <- rgb(fvalue(between/max(between))/255)  
  
plot(g.ba,vector.size=deg\*1.2, edge.color="orange", vertex.label.cex=.7,vertex.label.color="black", vertex.frame.color='white')  
text(x=0, y=-1.2, c('Colors that are more red indicate higher betweenness.'))



#Detect community using the Louvain method. Report the plot of the network with shade-colored communities.

commu = cluster\_louvain(g.ba)  
memb <- membership(commu)  
plot(g.ba, vertex.color = memb, layout = layout.auto, vertex.size = 10,  
 edge.width = (E(g.ba)$weight)^(1/5),   
 mark.groups=list(which(memb==1),which(memb==2), which(memb==3), which(memb==4)),  
 mark.border=NA)  
title("Communities detected by Louvain Algorithm")

