Social Network Analysis Home Assignment 1

{rashid ali}

due date - 08.05.2016 23:59

Table of Contents

# Power law. Descriptive network analysis

Please send your reports to [network.hse.2016@gmail.com](mailto:network.hse.2016@gmail.com) with the subject of of the following structure: *[MAGOLEGO SNA 2016] {LastName} {First Name} HA{Number}*

Late submission policy: -1 point per day

Use this file as a template for your report.  
Support your computations with figures and comments. Send ONLY .Rmd versions of your report.

##   
## Attaching package: 'igraph'

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

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

## Problem 1

Recall from the lecture that probability density function (PDF) for power law distributed variable is:

Take logarithm of both sides:

Now you can use standard R functions like lm() to calculate coefficient via linear regression. However you might find that it is a bad idea.

Alternatively, you can compute cumulative density function (CDF)

of power law distribution. Good things about CDF of the power law are:

* It still has form of power law
* On log-log plot it looks more like a line

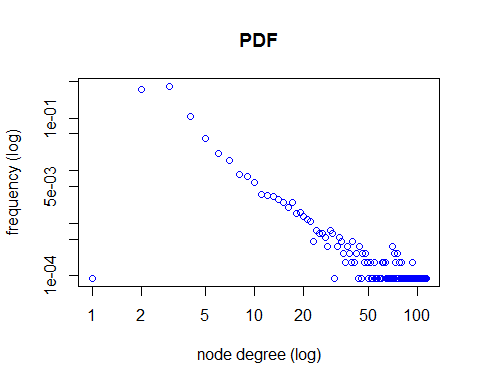
##### 1. Derive the formula for CDF function of power law . It means you should calculate cumulative distribution function by integration of PDF function.

\*write your answer here

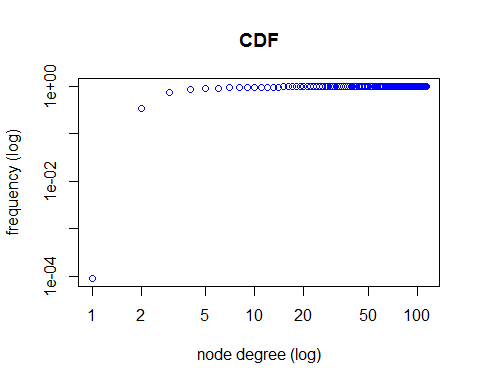
F(+???)= ???\_xmin^(+???)??????p(x)dx= ???\_xmin^(+???)??????Ct^(-??? ) ??? dt???  
   
F(x)=C t^(1-???)/(1-???) ???\_xmin^(+???)???   
F(x)=c (0+ ???xmin???^(1-???)/(???-1))   
 => F(x) = 1  
1=c (0+ ???xmin???^(1-???)/(???-1))  
=>C= (???-1)/???xmin???^(???-1)

##### 2. Download [Internet Network](https://sites.google.com/site/cxnets/research222)  and plot PDF and CDF of the degree distribution in log-log scale

## Put your code here  
network= read.graph("Internet\_AS.dat", format = "edgelist")  
xmax=max(degree(network))  
degree\_range= as.numeric(1:xmax)  
degree\_dist <- degree.distribution(network, cumulative = FALSE)[degree\_range]  
nonzero.position <- which(degree\_dist > 0)  
degree\_dist <- degree\_dist[nonzero.position]  
degree\_range <- degree\_range[nonzero.position]  
cum\_degree\_range <- degree\_range+1  
plot(degree\_dist, log = "xy", main = "PDF", xlab = "node degree (log)", ylab = "frequency (log)", col= "blue")



cum\_degree\_dist <- cumsum(degree\_dist)  
plot(cum\_degree\_dist, log = "xy", main = "CDF", xlab = "node degree (log)", ylab = "frequency (log)", col="blue")

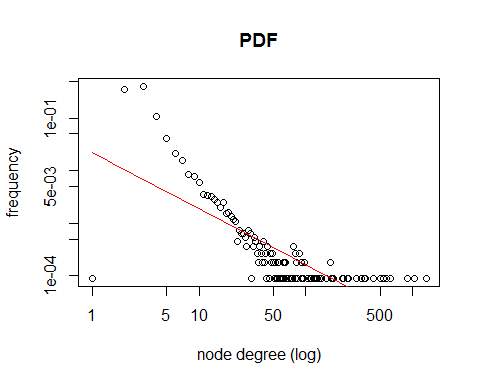


##### 3. Fit linear regression model to PDF and CDF to estimate . Plot fitted models along with data

## fit linear regression for PDF  
reg <- lm(log(degree\_dist) ~ log(degree\_range))  
coefficients <- coef(reg)  
func\_powerlaw <- function(x) exp(coefficients[[1]] + coefficients[[2]] \* log(x))  
alpha <- -coefficients[[2]]  
print(alpha)

## [1] 1.067228

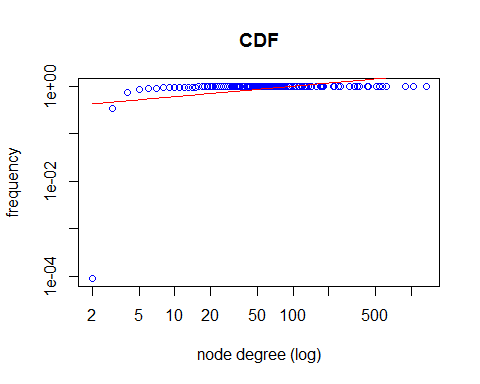
# plot PDF and its estimation  
plot(degree\_dist ~ degree\_range, log = "xy", main = "PDF", xlab = "node degree (log)", ylab = "frequency")  
curve(func\_powerlaw, col = "red", add = TRUE, n = length(degree\_range))



# fit linear regression for CDF  
reg <- lm(log(cum\_degree\_dist) ~ log(cum\_degree\_range))  
coefficients <- coef(reg)  
func\_powerlaw <- function(x) exp(coefficients[[1]] + coefficients[[2]] \* log(x))  
alpha2 <- -coefficients[[2]]  
alpha2

## [1] -0.2199214

# plot cDF and its estimation  
plot(cum\_degree\_dist ~ cum\_degree\_range, log = "xy", main = "CDF", xlab = "node degree (log)", ylab = "frequency", col= "blue")  
curve(func\_powerlaw, col = "red", add = TRUE, n = length(cum\_degree\_range))



## Problem 2

# Kolmogorov-Smirnov test describes how similar are two distributions. In our case, when we have fitted model and #original data, we can calculate their CDFs and Kolmogorov-Smirnov test shows us how well model approximates original data. In other words, it shows us the goodness-of-fit of our model. where and are theoretical and empirical CDFs respectively.

# KS illustration

To estimate of the fitted power-law model we can use KS test:

* Pick some value
* Fit power-law distribution to data (that is estimation of ) -- now we have
* Perform KS test -- compute statistic
* Finnaly, choose that provides minimal value of statistic among all KS tests run above.

In R all this stuff can be done in one line of code.

Again, use [Internet Network](https://sites.google.com/site/cxnets/research222)  
Properly load it into R and do following tasks:

##### 1. Using power.law.fit find xmin value and corresponding alpha

## Put your code here  
d <- degree(network)  
fit <- power.law.fit(d, NULL, implementation = "plfit")  
alpha <- fit$alpha  
xmin <- fit$xmin  
C <- (alpha-1)\*xmin^(alpha-1)  
alpha

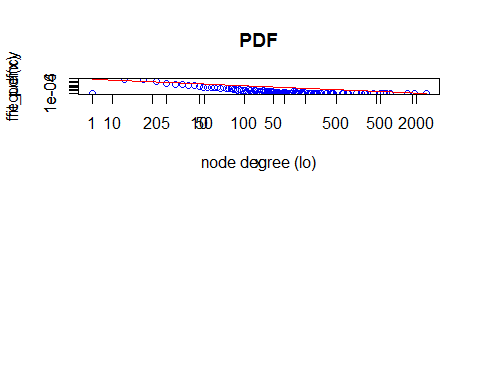
## [1] 2.08306

xmin

## [1] 7

##### 2. Put fitted model along with empirical PDF (CDF)

## Put your code here  
par(mfrow=c(2,1))  
fit\_pdf <- function(x) return(C\*x^(-alpha))  
plot(degree\_range,degree\_dist, log = "xy", main = "PDF", xlab = "node degree (lo)", ylab = "frequency", col= "blue")  
#lines(x, x^(-alpha), col="green")  
par(new=TRUE)  
curve(fit\_pdf,from=xmin,to=xmax, log="xy", col = "red", add = FALSE, n = length(degree\_range))



## Problem 3.

For [Wikipedia vote network](http://snap.stanford.edu/data/wiki-Vote.html) (clear up comments in the begging of the file) derive the following characteristics:  
1. The number of vertices and edges  
2. The number of loops (edges that start and end at the same vertex)  
3. The number of symmetrical edges  
4. Degree distribution (without considering the direction)  
5. The number of nodes with a degree greater than 1 and with a degree greater than 15  
6. Find strongly connected components and their sizes.  
7. Take subgraph of the original graph, which consists of the first 80 vertices and set color into red for those nodes in which the number of incoming edges is greater than the number of outgoing edges.Otherwise, set color in blue. For nodes with the same number of incoming and outgoing edges set color into green. Besides that, increase the size of vertices with a maximum value of transitivity (for example, you may set size into 10 for these nodes and 1 for others).  
8.Take subgraph from the previous task and find maximal connected component. For this component highlight any way that corresponds to the diameter of the subgraph. How many such paths are in this graph?  
9. Make average neighbor degree vs node degree scatter plot (one point on the plot per node) and aggregated plot, averaging over all nodes with the same degree (aggregated average vs degree, one value per degree). Explain your observations.  
10. Make local clustering coefficient vs node degree scatter plot (one point on the plot per node) and aggregated, averaging over all nodes with the same degree (aggregated average vs degree, one value per degree). Explain your observations.

##### 1. The number of vertices and edges.

## Put your code here  
wiki\_network = read.graph("Wiki-Vote.txt", format = "edgelist")  
vcount(wiki\_network)

## [1] 8298

ecount(wiki\_network)

## [1] 103689

##### 2. The number of loops (edges that start and end at the same vertex)

## Put your code here  
sum(is.loop(wiki\_network), na.rm = TRUE)

## [1] 0

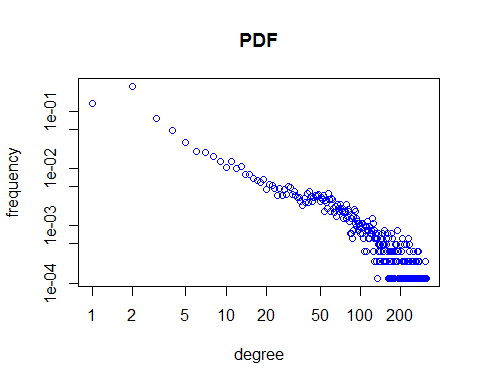
##### 3. The number of symmetrical edges

## Put your code here  
reciprocity(wiki\_network)\*ecount(wiki\_network)/2

## [1] 2927

##### 4. Degree distribution

## Put your code here  
degree\_dist2 <- degree.distribution(wiki\_network, mode='all')  
plot(degree.distribution(wiki\_network, mode= 'all')[which(degree\_dist2>0)], log= 'xy', main = "PDF", xlab = "degree", ylab = "frequency", col="blue")



##### 5. The number of nodes with a degree greater than 1 and with a degree greater than 15

sum(degree(wiki\_network)>1)

## [1] 4800

sum(degree(wiki\_network)>15)

## [1] 2389

##### 6. Find strongly connected components and thier sizes.

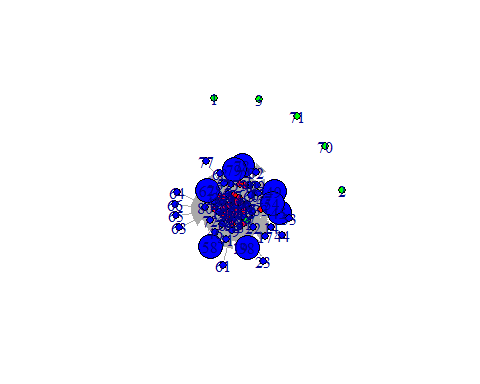
## Put your code here  
components <- clusters(wiki\_network, mode = "strong")  
components$no

## [1] 6999

#compoenets$csize

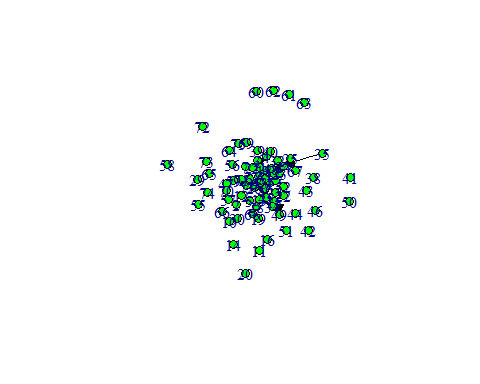
##### 7. Take subgraph of the original graph, which consists of the first 80 vertices and set color into red for those nodes in which the number of incoming edges is greater than the number of outgoing edges.Otherwise, set color in blue. For nodes with the same number of incoming and outgoing edges set color into green. Besides that, increase the size of vertices with a maximum value of transitivity (for example, you may set size into 10 for these nodes and 1 for others).

subgraph1 <- induced.subgraph(wiki\_network, V(wiki\_network)[1:80])  
dgin <- degree(subgraph1, mode = "in")  
dgout <- degree(subgraph1, mode = "out")  
V(subgraph1)[which(dgin>dgout)]$color <- "red"  
V(subgraph1)[which(dgin==dgout)]$color <- "green"  
V(subgraph1)[which(dgin<dgout)]$color <- "blue"  
tr <- transitivity(subgraph1, type = "local", isolates = "zero")  
V(subgraph1)$size <- ifelse(tr==1, 30, 10)  
plot(subgraph1)



##### 8.Take subgraph from the previous task and find maximal connected component. For this component highlight any way that corresponds to the diameter of the subgraph. HHow many such paths are in this graph?

## Put your code here  
con\_com <- clusters(subgraph1, mode = "weak")  
V(subgraph1)$membership <- clusters(subgraph1, mode = "weak")$membership  
subg\_new <- induced.subgraph(subgraph1, V(subgraph1)[membership==4])  
dpath <- get.diameter(subg\_new)  
E(subg\_new, path = dpath)$color <- "black"  
V(subg\_new)$color <- " green"  
plot(subg\_new, vertex.size= 10)

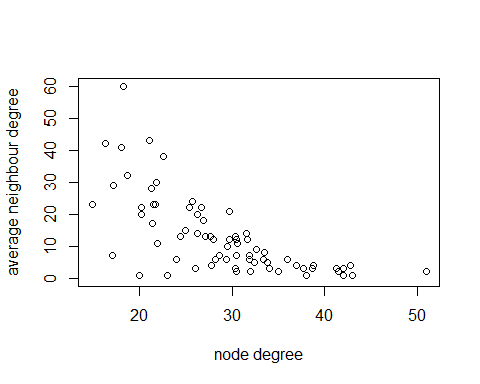


path.length.hist(subg\_new, directed = TRUE)

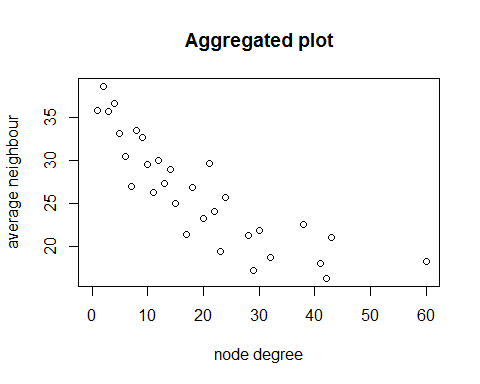
## $res  
## [1] 461 916 274 36 3  
##   
## $unconnected  
## [1] 3860

##### 9. Make average neighbour degree vs node degree scatter plot (one point on the plot per node) and aggregated plot, averaging over all nodes with the same degree (aggregated average vs degree, one value per degree). Explain your observations.

## Put your code here  
# get average neighbour degree  
avnd <- graph.knn(subgraph1)$knn  
# get node degree  
nd <- degree(subgraph1)  
# scatter plot  
plot(avnd, nd, xlab = "node degree", ylab = "average neighbour degree")

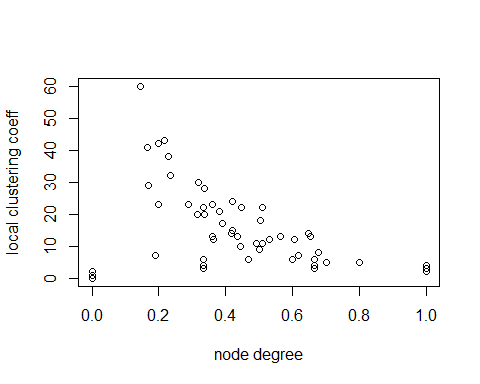


agg\_avnd <- c()  
index = 1  
for (d in unique(nd)) {  
agg\_avnd[index] <- sum(avnd[which(nd==d)])/length(which(nd==d))  
index = index + 1  
}  
plot(unique(nd), agg\_avnd, main = "Aggregated plot", xlab = "node degree", ylab = "average neighbour")



##### 10. Make local clustering coeff vs node degree scatter plot (one point on the plot per node) and aggregated, averaging over allnodes with the same degree (aggregated average vs degree, one value per degree). Explain your observations.

## Put your code here  
# get local clustering coeff  
lcc<- transitivity(subgraph1, type = "local", isolates = "zero")  
# get node degree  
nd <- degree(subgraph1)  
# scatter plot  
plot(lcc, nd, xlab = "node degree", ylab = "local clustering coeff")



agg\_lcc <- c()  
index = 1  
for (d in unique(nd)) {  
agg\_lcc[index] <- sum(lcc[which(nd==d)])/length(which(nd==d))  
index = index + 1  
}  
plot(unique(nd), agg\_lcc, main = "Aggregated plot", xlab = "node degree", ylab = "average local clustering coeff")

