Statistical Network Analysis of global flight infrastructure

Seán McMahon

22 May 2020

Loading in the data and making it igraph suitable

```
load("flights.RData") #Loads in the matrix Y

Yb = ifelse(Y>0,1,0) #Creates the binary version of Y with a value of 1 if it is above 0

X = Y + t(Y)
X = X/2      #Creates the undirected matrix X


Xb = ifelse(X>0,1,0) #Creates the binary version of X

library(igraph)

##
## Attaching package: 'igraph'

Ygra = graph.adjacency(Y, mode="directed", weighted = TRUE)  #Creates Igraph objects of the 4 matrices
Ybgra = graph.adjacency(Yb, mode="directed")
Xgra = graph.adjacency(Yb, mode="undirected", weighted = TRUE)
Xbgra = graph.adjacency(Yb, mode="undirected")
```

What countires are the most connected?

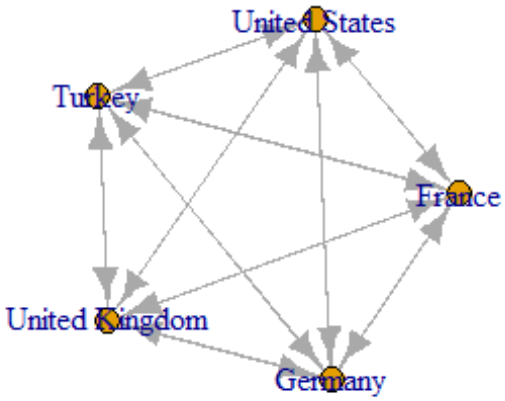
```
Ybdeg = degree(Ybgra, mode="total") #Creates an object with the total degree of all nodes
Ybdeg[order(Ybdeg, decreasing = TRUE)[1:5]] #Prints the 5 nodes with the largest degree in descending order.

##         France United Kingdom        Germany         Turkey  United States
##            227            198            196            186            184
```

Plotting the conections

```
plot(Ybgra, vertex.size= log(Ybdeg), vertex.label = ifelse(Ybdeg > 183, V(Ybgra)$name , NA), edge.arrow.size=.1,edge.width=0.2, vertex.label.font = 2, vertex.label.cex = 1, layout = layout.graphopt)
```

It is a bit too messy plotting everything, lets try just plotting the top 5

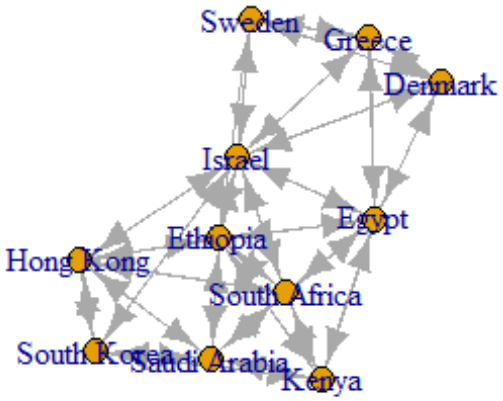
```
Ybdeg2 <-order(igraph::degree(Ybgra, mode = "all", normalized = T),
               decreasing = TRUE)
TopYdeg = induced_subgraph(Ybgra, Ybdeg2[1:5])
plot(TopYdeg)
```



We don't really gain much insight here, other than that the top 5 Countres are all connected to one another in both directions.

Instead lets take a look at a lower degree subgroup - the 20th-30th countries with the highest degree

```
Ybdeg3 <-order(igraph::degree(Ybgra, mode = "all", normalized = T),
               decreasing = TRUE)
MidYdeg = induced_subgraph(Ybgra, Ybdeg3[20:30])
plot(MidYdeg)
```



We can see that within this cohort, Israel has the most mutual connections with other countries. We can see that European countries such as Sweden and Denmark,can use Israel as a way to connect to further away countries such as Hong Kong and South Africa.

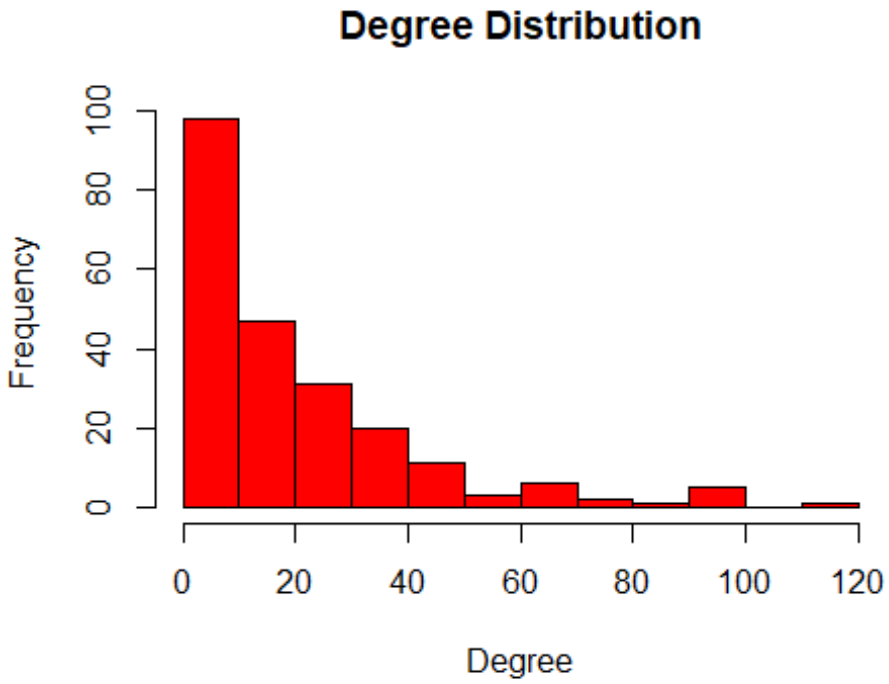
Let us lastly look at some countries with very low degree - the 120th-130th countries.

```
Ybdeg4 <-order(igraph::degree(Ybgra, mode = "all", normalized = T),
               decreasing = TRUE)
LowYdeg = induced_subgraph(Ybgra, Ybdeg4[120:130])
plot(LowYdeg)
```

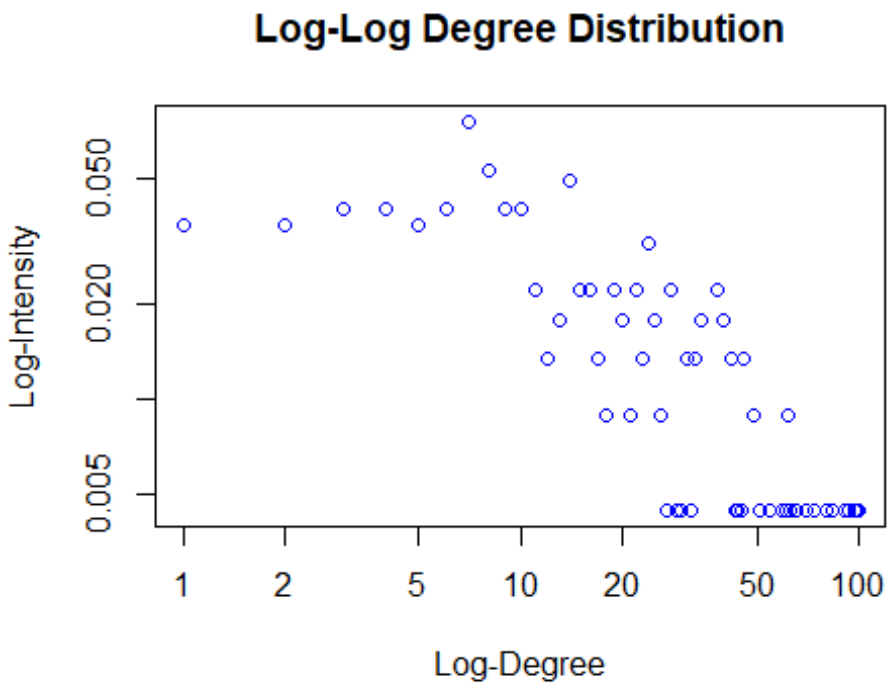
We can see that only the Republic of the Congo and the Democratic Republic of the Congo are connected within this cohort- the others are completely isolated, there is no way to travel between one country to another in one flight.

Is the powerlaw property displayed in this network?

```
hist(degree(Xgra, mode="total"),col="red", xlab="Degree", ylab="Frequency", main="Degree Distribution") #Creates a graph of the degree distribution in the natural scale
```




```
#Creates a graph of the log-log degree distribution
dd.Y = degree.distribution(Xgra)
d = 1:max(degree(Xgra))-1
ind = (dd.Y != 0)
plot(d[ind], dd.Y[ind], log="xy", col="blue",xlab=c("Log-Degree"), ylab=c("Lo
g-Intensity"), main="Log-Log Degree Distribution")
```



There is slight evidence for powerlaw behaviour, the degree distributions on the normal scale are concentrated on the left hand side - meaning that the vast majority of nodes have a very low number of connections, and it is a small number of nodes that are highly connected. When transformed to a log-log distribution, a straight line can be seen from 10 Log-degree onwards, indicating that there is some powerlaw behaviour, but it is not perfect.

Network Summary statistics

```
transitivity(Ybgra) #Clustering Coefficient
```

```
## [1] 0.4326141
```

```
mean(degree(Ybgra)) #Average total degree
```

```
## [1] 40.51556
```

```
reciprocity(Ybgra) #Reciprocity coefficient
```

```
## [1] 0.983326
```

```
mean_distance(Ybgra) #Average path length
```

```
## [1] 2.34929
```

```
mean(knn(Ybgra)$knn) #Average degree of the neighbours
```

```
## [1] 80.28593
```

From this last result you can see that the network is highly connected, with the average degree of a given nodes neighbors being 80.286

What nodes are the most central? Is there a difference between page rank and betweenness centrality?

```
Ybpage = page.rank(Ybgra)$vector       #Creates a vector with all the page ra
nk values
Ybpage[order(Ybpage, decreasing = TRUE)[1:5]]   #Shows the top 5 values and n
ode names
```

```
##         France  United States United Kingdom        Germany         Turkey
##     0.02131363     0.02112168     0.01882248     0.01791224     0.01702057
```

```
Ybcentre = centr_betw(Ybgra)$res   #Creates a vector with all the betweenness
centrality values
Ybcentre[order(Ybcentre, decreasing=TRUE)[1:5]] #Shows the top 5 values
```

```
## [1] 8712.975 6554.087 3577.687 3056.025 2925.455
```

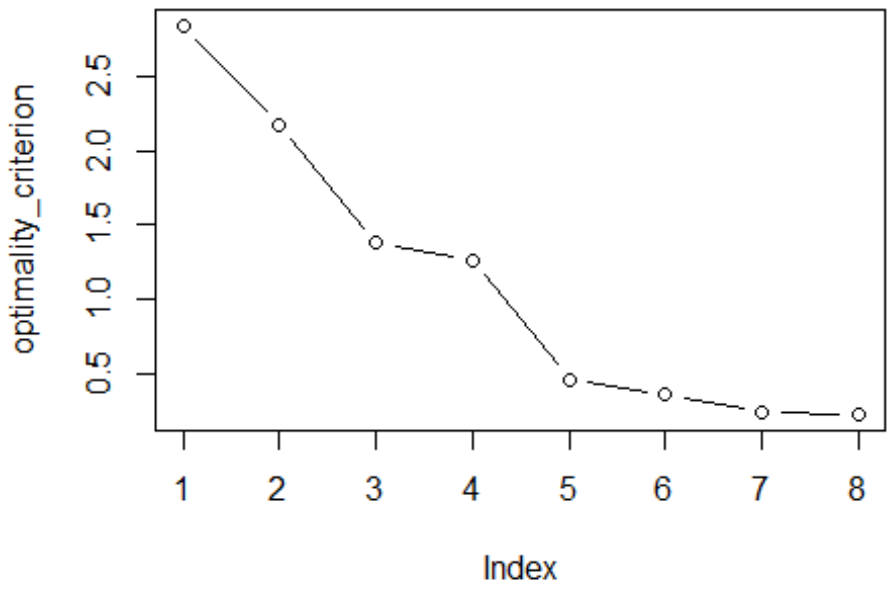
```
V(Ybgra)[order(Ybcentre, decreasing = T)[1:5]] #Shows the top 5 node names
```

```
## + 5/225 vertices, named, from 10ce749:
## [1] United States        France               United Kingdom
## [4] Australia            United Arab Emirates
```

It can be seen that for page-rank, the top 5 nodes are France, the United States, the United Kingdom, Germany and then Turkey. For the betweeness centrality values, the top 5 are the United States, France, the United Kingdom, Australia, and the UAE. Betweenness centrality is measuring the number of times a node lies on the shortest path between other nodes. The page rank takes into account a nodes connections and thier neighbors connections, as well as taking the nodes degree into account. The reason for the difference could be that Australia and the UAE are important airports that provide a link between different areas that other nodes cant supply, whereas Germany and Turkey has central and strong connectios beyond what they directly connect with.

Clustering the network

```
embedding = eigen(X)$vectors[, 1:3] #Takes the top 3 eigenvectors
K_max = 8 #Sets the max number of clusters to examine
res_kmeans = list()
optimality_criterion = rep(NA,K_max)
for (k in 1:K_max){
  res_kmeans[[k]] = kmeans(embedding, k)
  optimality_criterion[k] = res_kmeans[[k]]$tot.withinss
}
plot(optimality_criterion, type = "b")  #The elbow on the plot is the number
of clusters to use
```




```
memberships = res_kmeans[[4]]$cluster
res = make_clusters(Xgra, memberships)
V(Xgra)[res[[1]]] #First group is all in Western Europe
```

```
## + 5/225 vertices, named, from 10cec3b:
## [1] France         Germany        Italy          Spain          United Kin
gdom
```

```
V(Xgra)[res[[2]]] #Second group is North America
```

```
## + 18/225 vertices, named, from 10cec3b:
##  [1] Austria        Belgium        Croatia        Czech Republic Denmark
##  [6] Greece         Ireland        Morocco        Netherlands    Norway
## [11] Poland         Portugal       Romania        Russia         Sweden
## [16] Switzerland    Tunisia        Turkey
```

```
V(Xgra)[res[[3]]] #Group 3 is east asia
```

```
## + 196/225 vertices, named, from 10cec3b:
##   [1] Afghanistan                    Albania
##   [3] Algeria                        American Samoa
##   [5] Angola                         Anguilla
##   [7] Antigua and Barbuda            Argentina
##   [9] Armenia                        Aruba
##  [11] Australia                      Azerbaijan
##  [13] Bahamas                        Bahrain
##  [15] Bangladesh                     Barbados
##  [17] Belarus                        Belize
```

```
##  [19] Benin                                Bermuda
## + ... omitted several vertices
```

The spectral clustering was able to quite accuratly cluster 3 different geographical regions into 3 different clusters, with Western Europe being in the first cluster, North America in the second, and the 3rd cluster being East Asia. The final cluster however, contained all other remaining countries and no significant inference about these can be made.

Let us now try fitting a stochastic block model using the blockmodels package to see how its communities compare to the spectral clusters.

```
library(blockmodels)
```

```
## Loading required package: Rcpp
```

```
## Loading required package: parallel
```

```
## Loading required package: digest
```

```
sbm = BM_poisson(membership_type = "SBM",  #Creates the stochastic block mode
l
                    adj = as.matrix(X),
                    plotting = "",
                    explore_min = 2,
                    explore_max = 8)
```

```
sbm$estimate()
```

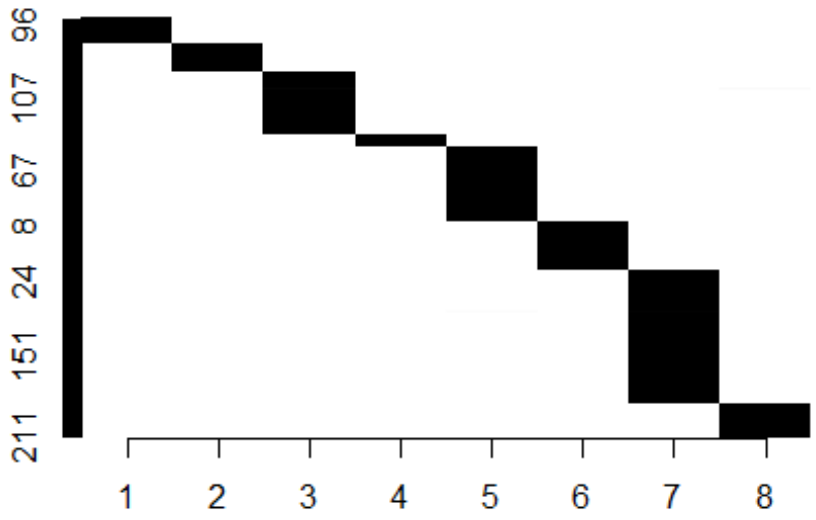
```
## -> Estimation for 1 groups
##                  -> 1 initializations provided
##                  -> 0 initializations already used
##              -> Estimation with 1 initializations
##
```

------OUTPUT TRIMMED TO REDUCE DOCUMENT SIZE ---------

```
##                -> Useless, no better ICL criterion found
##                      -> better ICL found: -44268.0603662582
##                      -> old ICL: -44145.184063665
##     -> With descending number of groups
##         -> For 7 groups
##             -> Selecting intializations
##                 -> Init from merging groups from 8 groups
##             -> Already done
##         -> For 6 groups
##             -> Selecting intializations
##                 -> Init from merging groups from 7 groups
##             -> Already done
##         -> For 5 groups
##             -> Selecting intializations
##                 -> Init from merging groups from 6 groups
```

```
##             -> Already done
##         -> For 4 groups
##             -> Selecting intializations
##                 -> Init from merging groups from 5 groups
##             -> Already done
##         -> For 3 groups
##             -> Selecting intializations
##                 -> Init from merging groups from 4 groups
##             -> Already done
##         -> For 2 groups
##             -> Selecting intializations
##                 -> Init from merging groups from 3 groups
##             -> Already done
```

```
K_star = which.max(sbm$ICL) # Finds that the best model according to ICL valu
e is with 8 groups

# Clusters the nodes according to the block model
soft_clustering = sbm$memberships[[K_star]]$Z
hard_clustering =apply(soft_clustering, 1, which.max) #

sbm$memberships[[K_star]]$plot() # A plot of the size of each group
```




```
#Looks at the node names in each group
Countries = colnames(X)
Countries[which(hard_clustering ==1)]
```

```
##  [1] "Australia"            "China"                "Egypt"
##  [4] "Hong Kong"            "India"                "Japan"
##  [7] "Malaysia"             "Russia"               "Saudi Arabia"
## [10] "Singapore"            "South Korea"          "Taiwan"
## [13] "Thailand"             "United Arab Emirates"
```

```
Countries[which(hard_clustering ==2)]
```

```
##  [1] "Austria"     "Belgium"     "Canada"      "Denmark"     "Greece"
##  [6] "Ireland"     "Mexico"      "Morocco"     "Netherlands" "Norway"
## [11] "Poland"      "Portugal"    "Sweden"      "Switzerland" "Turkey"
```

```
Countries[which(hard_clustering ==3)]
```

```
##  [1] "Afghanistan" "Armenia"     "Azerbaijan"  "Bahrain"     "Bangladesh"
##  [6] "Belarus"     "Brunei"      "Burma"       "Cambodia"    "Georgia"
## [11] "Guam"        "Indonesia"   "Iran"        "Iraq"        "Jordan"
## [16] "Kazakhstan"  "Kuwait"      "Kyrgyzstan"  "Laos"        "Lebanon"
## [21] "Macau"       "Maldives"    "Nepal"       "New Zealand" "Oman"
## [26] "Pakistan"    "Philippines" "Qatar"       "Sri Lanka"   "Sudan"
## [31] "Tajikistan"  "Uzbekistan"  "Vietnam"     "Yemen"
```

```
Countries[which(hard_clustering ==4)]
```

```
## [1] "France"         "Germany"        "Italy"          "Spain"
## [5] "United Kingdom" "United States"
```

```
Countries[which(hard_clustering ==5)]
```

```
##  [1] "Angola"              "Benin"               "Botswana"
##  [4] "Burkina Faso"        "Burundi"             "Cameroon"
##  [7] "Chad"                "Comoros"             "Congo (Brazzaville)"
## [10] "Congo (Kinshasa)"    "Cote d'Ivoire"       "Djibouti"
## [13] "Equatorial Guinea"   "Ethiopia"            "Gabon"
## [16] "Ghana"               "Guinea"              "Kenya"
## [19] "Liberia"             "Madagascar"          "Malawi"
## [22] "Mali"                "Mauritius"           "Mayotte"
## [25] "Mozambique"          "Namibia"             "Niger"
## [28] "Nigeria"             "Reunion"             "Rwanda"
## [31] "Senegal"             "Sierra Leone"        "Somalia"
## [34] "South Africa"        "South Sudan"         "Tanzania"
## [37] "Togo"                "Uganda"              "Zambia"
## [40] "Zimbabwe"
```

```
Countries[which(hard_clustering ==6)]
```

```
##  [1] "Albania"              "Antigua and Barbuda"  "Argentina"
##  [4] "Aruba"                "Bahamas"              "Brazil"
##  [7] "Cape Verde"           "Chile"                "Colombia"
## [10] "Costa Rica"           "Cuba"                 "Dominican Republic"
## [13] "Ecuador"              "El Salvador"          "Guatemala"
## [16] "Honduras"             "Iceland"              "Jamaica"
```

```
## [19] "Jersey"               "Netherlands Antilles" "Panama"
## [22] "Peru"                 "Puerto Rico"          "Trinidad and Tobago"
## [25] "Venezuela"            "Virgin Islands"
```

```
Countries[which(hard_clustering ==7)]
```

```
##  [1] "American Samoa"                  "Anguilla"
##  [3] "Barbados"                        "Belize"
##  [5] "Bermuda"                         "Bhutan"
##  [7] "Bolivia"                         "Bosnia and Herzegovina"
##  [9] "British Virgin Islands"          "Cayman Islands"
## [11] "Central African Republic"        "Christmas Island"
## [13] "Cocos (Keeling) Islands"         "Cook Islands"
## [15] "Dominica"                        "East Timor"
## [17] "Eritrea"                         "Falkland Islands"
## [19] "Faroe Islands"                   "Fiji"
## [21] "French Guiana"                   "French Polynesia"
## [23] "Gambia"                          "Gibraltar"
## [25] "Greenland"                       "Grenada"
## [27] "Guadeloupe"                      "Guernsey"
## [29] "Guinea-Bissau"                   "Guyana"
## [31] "Haiti"                           "Isle of Man"
## [33] "Kiribati"                        "Lesotho"
## [35] "Macedonia"                       "Marshall Islands"
## [37] "Martinique"                      "Mauritania"
## [39] "Micronesia"                      "Moldova"
## [41] "Mongolia"                        "Montenegro"
## [43] "Nauru"                           "New Caledonia"
## [45] "Nicaragua"                       "Niue"
## [47] "Norfolk Island"                  "North Korea"
## [49] "Northern Mariana Islands"        "Palau"
## [51] "Papua New Guinea"                "Paraguay"
## [53] "Saint Kitts and Nevis"           "Saint Lucia"
## [55] "Saint Pierre and Miquelon"       "Saint Vincent and the Grenadines"
## [57] "Samoa"                           "Sao Tome and Principe"
## [59] "Seychelles"                      "Slovakia"
## [61] "Slovenia"                        "Solomon Islands"
## [63] "Suriname"                        "Swaziland"
## [65] "Tonga"                           "Turkmenistan"
## [67] "Turks and Caicos Islands"        "Tuvalu"
## [69] "Uruguay"                         "Vanuatu"
## [71] "Wallis and Futuna"               "Western Sahara"
```

```
Countries[which(hard_clustering ==8)]
```

```
##  [1] "Algeria"        "Bulgaria"       "Croatia"        "Cyprus"
##  [5] "Czech Republic" "Estonia"        "Finland"        "Hungary"
##  [9] "Israel"         "Latvia"         "Libya"          "Lithuania"
## [13] "Luxembourg"     "Malta"          "Romania"        "Serbia"
## [17] "Tunisia"        "Ukraine"
```

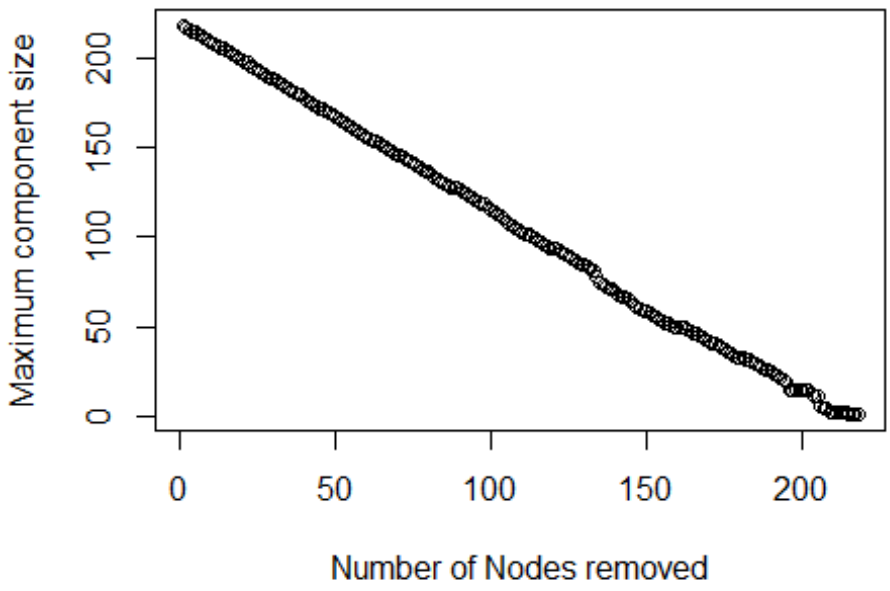
The Integrated Completed Likelihood criterion showed that 8 groups was the best to use. From these 8 groups it can be seen that the majority within each share geographical features.

- Group 1 is primarily composed of countries within Asia.
- Group 2 is primarily composed of smaller western European and North American countries.
- Group 3 is composed of countries primarily in central and and western Asia.
- Group 4 is composed of high population countries in Western Europe and North America that have many connections.
- Group 5 is composed mainly of African Countries that connect to one another primarily
- Group 6 is composed primarily with South American and Carribean countries
- Group 7 is composed of very small nations and islands, primarily those located in the pacific that mainly connect to one another and very few other nodes
- Group 8 is composed primarily of Eastern European Countries

---

How resistant is our airport network to attacks? What happens to global connectivity if certain countries are locked down?

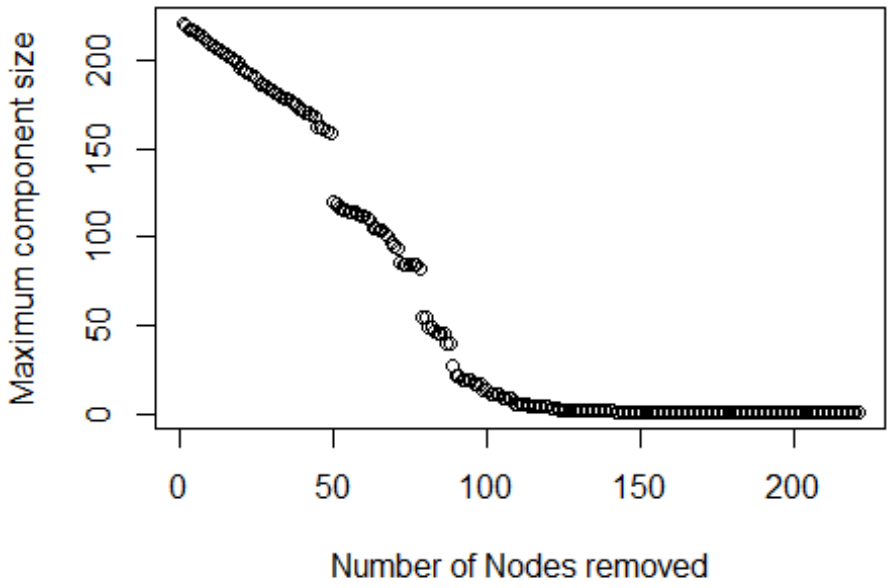
```
tester = Xbgra   #Creates a copy of the Xb matrix so the deletions are not pe
rmanent
set.seed(1)      #Sets a seed so that the outcomes are repeatable
max=225
tracker = list()
maxsize = list()

#A loop that removes a random node from the network and checks to see if new
components emerge. Monitors the maximum size of the biggest component as the
nodes are deleted
for(i in 1:224){

  tester = delete.vertices(tester, sample(1:max,1))
  if(components(tester)$no > 1){
    tracker = append(tracker, i)
    maxsize = append(maxsize, max(components(tester)$csize))
  }

  max = max-1

}
tracker[1] #Shows how many deletions were needed for a second component to be
created for the first time

## [[1]]
## [1] 6

plot(as.numeric(maxsize), xlab = "Number of Nodes removed", ylab = "Maximum c
omponent size") #Plots the maximum size of the biggest component over time
```



From the plot you can see that as more nodes get deleted, the maximum size of the component gets smaller at a near perfect linear rate. The fact that deleting nodes does not cause a rapid decline in the component sign indicates that this network, and thus our airport connectivity, is rather robust and able to handle node attacks.

```
####The process is now repeated but this time the node with the higest degree
is removed each step
tester = Xbgra
set.seed(1)
max=225
tracker = list()
maxsize = list()
highestdegree = 0

for(i in 1:224){

  highestdegree = degree(tester) #Creates a list of the each nodes degree

  tester = delete.vertices(tester, order(highestdegree, decreasing =TRUE)[1])
#Removes the node with the highest degree
  if(components(tester)$no > 1){
    tracker = append(tracker, i)
    maxsize = append(maxsize, max(components(tester)$csize))
  }
}
```

```r
tracker[1] #Shows how many deletions were needed for a second component to be created for the first time

## [[1]]
## [1] 3

plot(as.numeric(maxsize), xlab = "Number of Nodes removed", ylab = "Maximum component size") #Plots the maximum size of the biggest component over time
```



From this plot you can see that when the node with the highest degree is removed each step, the max component size decereases linearly until about 50 nodes are removed, then it drops rapidly, which is very different to when they were removed at random. It can be seen that from between 50 and 80 nodes being removed, the maximum component size drops rapidly indicating that the network becomes fractured and broken up. This more severe reaction is to be expected when the most connected node is removed each step.
