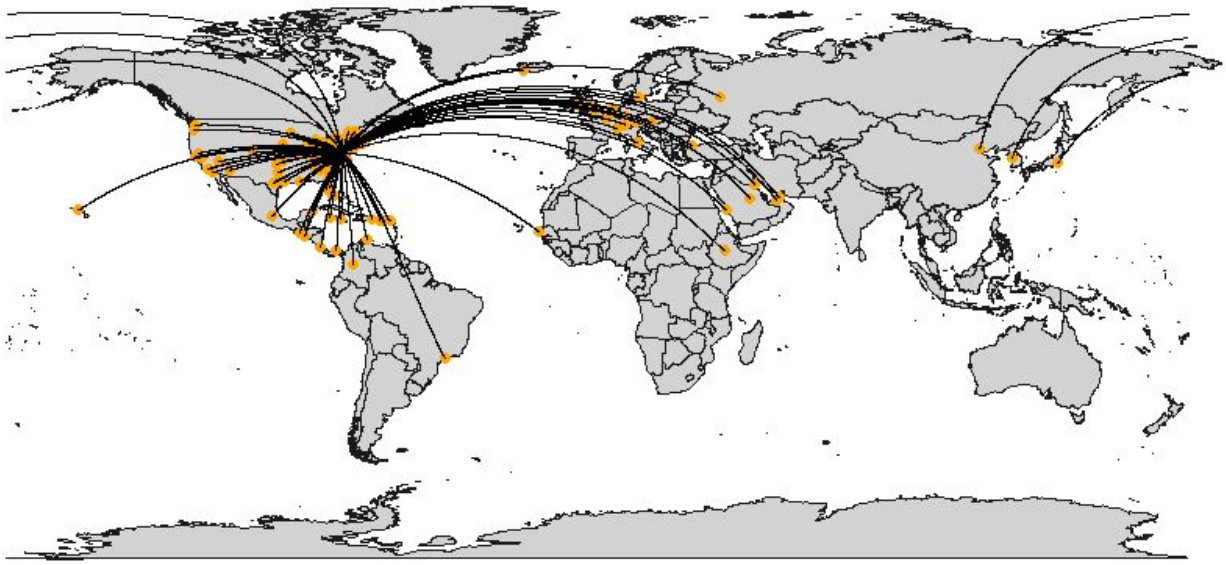
Jisu Kim (jk5nc), Timothy Kim (tsk8va), Olivia Ryu (hr2ad)
STAT 3280-001
Professor Tianxi Li
12 April 2020

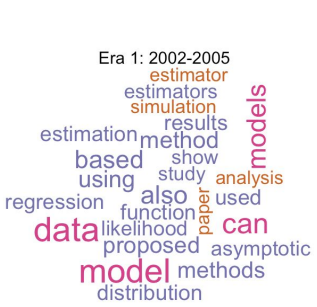
Homework 3

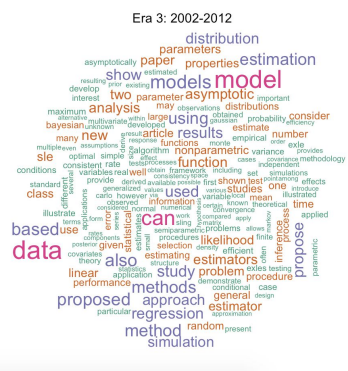
Problem 1



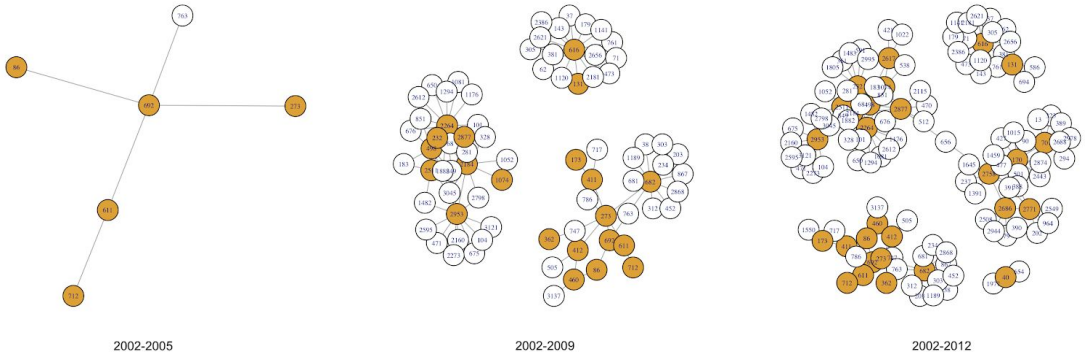
Note: We plotted all air routes flying to or from Washington Dulles International Airport (IATA code “IAD”), located in Dulles, Virginia. This was done by subsetting routes.dat to only entries where either the source or destination were IAD, extracting all airports along the routes and plotting, and then plotting all of the routes themselves in arc shapes.

Problem 2.1

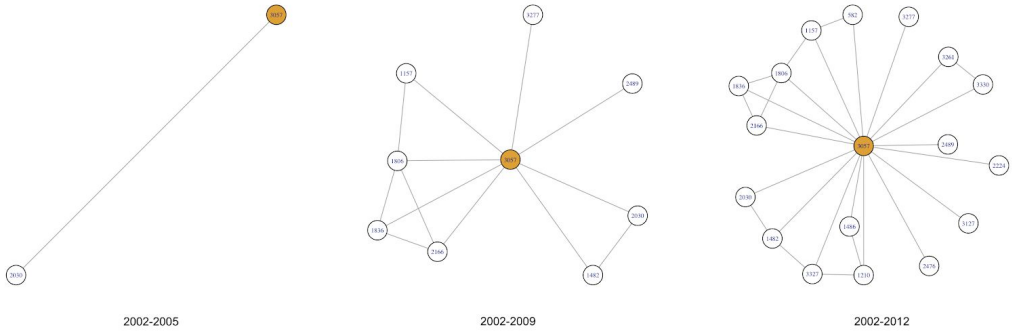




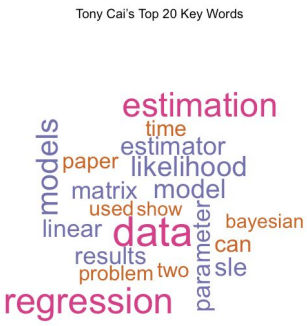
Problem 2.2.a



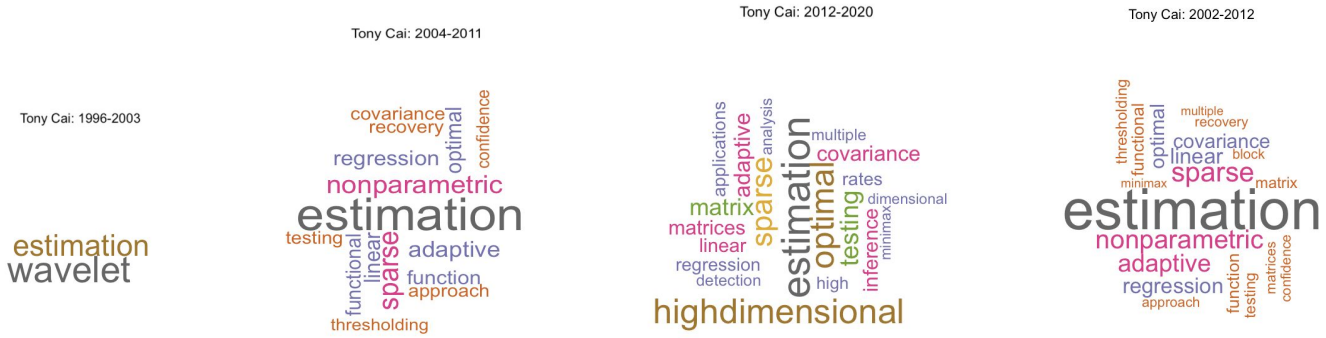
Problem 2.2.b



Problem 2.2.c



Problem 2.2.d



Problem 2.1

For this problem, we split the 11-year span into three separate "eras", in which the next era is an accumulation of the previous eras, in order to observe the trend in abstract keywords over time. It is clear that the words "data" and "model" are consistently the most frequently appearing words in the abstracts.

Problem 2.2a

We observe Tony Cai's citation network rapidly increases in size as he produces more publications, and as more other researchers cite his work. Of interesting note is the presence of distinct clusters: In the third graph, representing all of Tony Cai's work up to 2012, there are four disconnected clusters of nodes, indicating that papers in these clusters did not cite others. This perhaps represents papers in different fields of study that have no need to cite one another.

Problem 2.2b

Tony Cai's collaboration network follows a similar trend in expansion and increase in complexity. Because this graph focuses strictly on researchers, it is less "complex" as the citation network. There are no "clusters" present in these graphs, as all researchers share an edge with Tony Cai if they were collaborators.

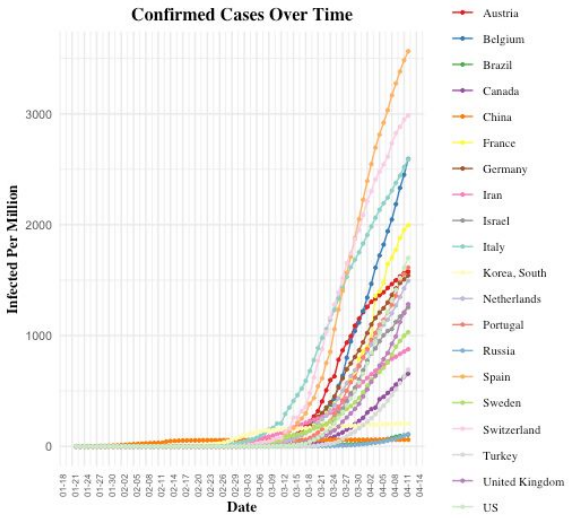
Problem 2.2c

Yes. Tony Cai's keywords from his articles are similar to the global trend of keywords from abstracts in that he also most frequently uses the words "data" and "model."

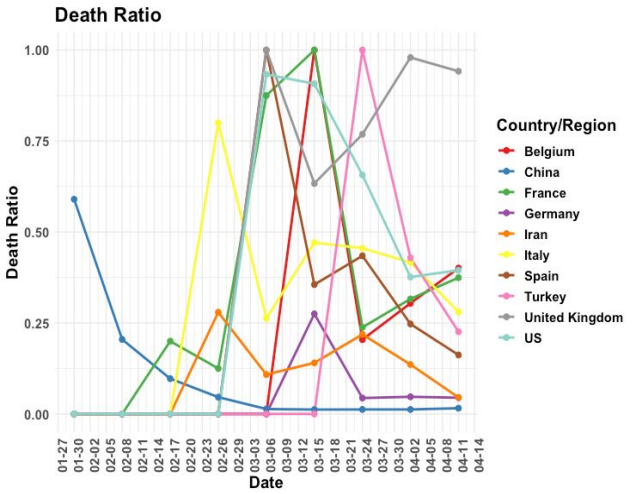
Problem 2.2d

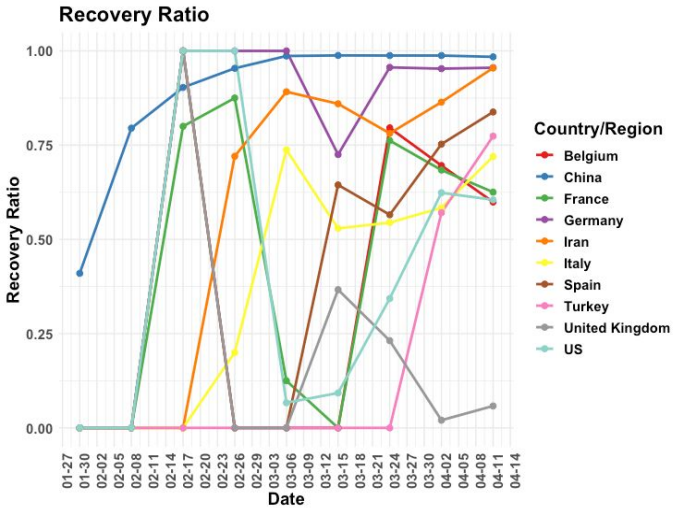
After retrieving Tony Cai's paper titles from Google Scholar, we generated both 1.) his key words from titles over time for his whole academic career and 2.) his key words from titles for the period 2002-2012. In comparing his word cloud for the period 2002-2012 to the word cloud in part c, we see that they are similar in that the word "estimation" is among the most frequently appearing words. However, we also see a bit of variation in the words from his articles and the words from his paper titles - frequently used words in his paper titles such as "nonparametric", "sparse", and "adaptive" are words that we do not see in his word cloud for keywords from articles in part c.

Problem 3.1

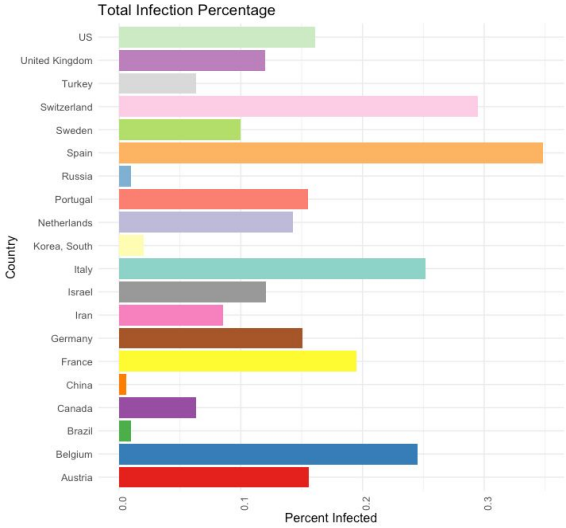


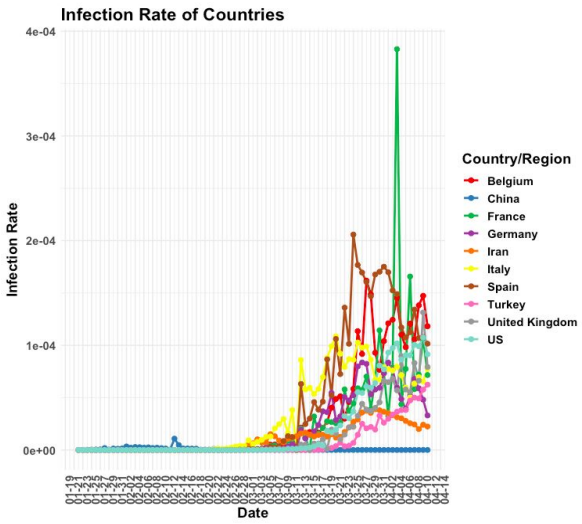
Problem 3.2





Problem 3.4





Problem 3.2

At present, it is impossible to calculate the death to recovery ratio, as these numbers are not finalized and we must take into account those who are currently battling the virus. In other words, the current death numbers are relevant to the confirmed numbers of the past. Therefore, to account for the time gap in between these numbers, we used an alternative method to compute the death and recovery ratios. We used the formula $\frac{new\ deaths}{new\ deaths + new\ recovered}$, where "new deaths" represents the number of new deaths (as opposed to cumulative deaths provided by the dataset) aggregated over a period of 9 days and "new recovered" represents the number of new recoveries made within the same 9 days. This method ensures that we are not neglecting those who are still battling the virus, as we are only considering only those who either recovered or died from the virus.

Problem 3.4

Looking at the total infection percentage, it seems that Russia, South Korea, China, and Brazil are doing significantly better than other countries, while countries like Spain and Switzerland are struggling currently.

To examine and understand the measures that are helping out these countries, we can look at South Korea for example. They have established numerous *FREE* testing centers easily accessible by everyone. In addition to this measure, they have taken extensive steps to isolate and trace the paths of the infected. They have taken aggressive preparatory measures even prior to the actual hit.

China has also demonstrated exemplary quarantine procedures, as seen in the rapidly declining death ratio and equally-rapidly increasing recovery ratio. Using strict and immediate quarantine of hotspots such as Wuhan, China was able to aggressively nullify the infectious transmission of COVID-19 and is now well on the path to recovery: Wuhan lifted its quarantine on April 7th, and the graphs indicate a resurgence in infection has not occurred.

Just a week before Spain's shutdown on March 8th, sports events, political party conferences, and massive demonstrations to celebrate International Women's Day all took place. Spain simply reacted too late. In addition, the country lacked essential equipment, like ventilators and protective clothing.

The spread of the virus is clearly out of control in the United States, as indicated by the continual upward trend in infection rate according to the second graph. We recognize that it is too late for the US to take more preparatory measures as South Korea has done. However, we can learn from this exemplary country's method of establishing free testing centers for all. Despite the lockdown, it is inevitable to leave home to tend to business; when the infected, unaware of their status, go out and interact with objects, there is a great likelihood that this virus will spread further. Therefore, quick testing centers would be of great benefit to this nation.

Appendix

Problem 1

```
# Problem 1

# Load in packages
library(maps)
install.packages("geosphere")
library('geosphere')

# Importing data
airports <- read.delim("airports.txt", sep=",")
colnames(airports) <- c("id", "name", "city", "country", "IATA", "ICAO",
                        "latitude", "longitude", "altitude", "timezone",
                        "DST", "database.timezone")

airlines <- read.delim("airlines.dat", sep=",")
colnames(airlines) <- c("id", "name", "alias", "IATA", "ICAO", "callsign",
                        "country", "active")

routes <- read.delim("routes.dat", sep=",")
colnames(routes) <- c("airline", "airline.id", "source", "source.id",
                      "destination", "destination.id", "codeshare", "stops",
                      "equipment")

# Have to repair source.id and destination.id to be numeric class
routes$source.id <- as.numeric(as.character(routes$source.id))
routes$destination.id <- as.numeric(as.character(routes$destination.id))
routes <- na.omit(routes)

# Selecting Washington Dulles International Airport
sel <- rbind(routes[which(routes$source == "IAD"),],
             routes[which(routes$destination == "IAD"),])

# Pulling the nodes from this df of routes
sel.unique <- c(unique(sel$source.id), unique(sel$destination.id))
sel.airports <- rbind(airports[airports$id %in% sel.unique,], airports[3584,])

sel.source.index <- which(unlist(lapply(sel$source.id,function(x){
  any(airports$id==x)
})))
sel.destination.index <- which(unlist(lapply(sel$destination.id,function(x){
  any(airports$id==x)
})))
sel <- sel[intersect(sel.source.index,sel.destination.index),]

# Plotting airports over a world map
world <- map("world", col="gray85",  border="gray10", fill=TRUE, bg="white")
```

```
points(x=sel.airports$longitude, y=sel.airports$latitude, pch=20,
       cex=sel.airports$Visits/80, col="orange")

# Generating arcs with shortest path
for(i in 1:nrow(sel))
{
  node1 <- airports[airports$id == sel[i,]$source.id,]
  node2 <- airports[airports$id == sel[i,]$destination.id,]
  arc <- gcIntermediate( c(node1[1,]$longitude, node1[1,]$latitude),
                         c(node2[1,]$longitude, node2[1,]$latitude),
                         n=1000, addStartEnd=TRUE, breakAtDateLine = F ) #1000
points along the arc
  longitude_diff <- abs(node1[1,]$longitude) + abs(node2[1,]$longitude)
  arc = data.frame(arc)
  if(longitude_diff>180){
    lines(arc[arc$lon>=0,])
    lines(arc[arc$lon<0,])
  }
  else{
    lines(arc)
  }
}
```

Problem 2 (Set Up)

```
# Problem 2
########### Run once ###########
install.packages("tm")
install.packages("SnowballC")

dir.create("./Era1Abstracts/")
dir.create("./Era2Abstracts/")
dir.create("./Era3Abstracts/")
################################

authors <- read.delim("authorList.txt", sep="\n")
paper.biadj <- read.table("authorPaperBiadj.txt")
cit.adj <- read.table("paperCitAdj.txt")
# papers <- read.csv("paperList.txt", header=TRUE)
# abstracts <- read.csv("paperList_Abstracts_Keyword.txt", header=TRUE)

#########################################################################

papers1 <- scan("paperList.txt",sep="\n",what="")
papers <- scan("paperList_Abstracts_Keyword.txt",sep="\n",what="")
papers <- papers[-1]

dt <- scan("paperList_Abstracts_Keyword.txt",what="",sep="\n")
```

```
length(dt)
dt <- dt[-1]

tmp1 <- lapply(dt,function(s)unlist(strsplit(s,'"')))

years <- unlist(lapply(tmp1,function(x) return(x[3])))
years <- gsub(",","",years)
years <- as.numeric(years)

abss <- unlist(lapply(tmp1,function(x) return(x[4])))
abss <- unlist(lapply(abss,function(x)gsub('\\{','',x)))
abss <- unlist(lapply(abss,function(x)gsub("}","",x)))
abss <- unlist(lapply(abss,function(x)gsub("#","",x)))
abss <- unlist(lapply(abss,function(x)gsub("&","",x)))
abss <- unlist(lapply(abss,function(x)gsub("<","",x)))
abss <- unlist(lapply(abss,function(x)gsub(">","",x)))
abss <- unlist(lapply(abss,function(x)gsub("/","",x)))
abss <- unlist(lapply(abss,function(x)gsub(")","",x)))
abss <- unlist(lapply(abss,function(x)gsub("\\(","",x)))
abss <- unlist(lapply(abss,function(x)gsub("amp","",x)))
```

Problem 2.1

```
# Problem 2.1
by.year <- data.frame(abss, years)
set1 <- by.year[which(by.year$years == 2002 |
                        by.year$years == 2003 |
                        by.year$years == 2004 |
                        by.year$years == 2005) ,]
set2 <- rbind(set1, by.year[which(by.year$years == 2006 |
                        by.year$years == 2007 |
                        by.year$years == 2008 |
                        by.year$years == 2009) ,])
set3 <- rbind(set2, by.year[which(by.year$years == 2010 |
                        by.year$years == 2011 |
                        by.year$years == 2012) ,])

era1 <- as.character(set1$abss)
era2 <- as.character(set2$abss)
era3 <- as.character(set3$abss)

for(k in 1:length(abss))
{
  write(abss[k], file=paste("./AllAbstracts/", k, sep=""))
}

for(k in 1:length(era1))
{
```

```
  write(era1[k], file=paste("./Era1Abstracts/", k, sep=""))
}

for(k in 1:length(era2))
{
  write(era2[k], file=paste("./Era2Abstracts/", k, sep=""))
}

for(k in 1:length(era3))
{
  write(era3[k], file=paste("./Era3Abstracts/", k, sep=""))
}

library(tm)
library(SnowballC)

cname <- "./AllAbstracts/"
cname1 <- "./Era1Abstracts"
cname2 <- "./Era2Abstracts"
cname3 <- "./Era3Abstracts"

## read in all the documents as a Corpus
docs <- Corpus(DirSource(cname))
docs1 <- Corpus(DirSource(cname1))
docs2 <- Corpus(DirSource(cname2))
docs3 <- Corpus(DirSource(cname3))

document.names <- unlist(meta(docs,"id"))
document.names1 <- unlist(meta(docs1,"id"))
document.names2 <- unlist(meta(docs2,"id"))
document.names3 <- unlist(meta(docs3,"id"))

## Removing punctuation:
docs <- tm_map(docs, removePunctuation)
docs1 <- tm_map(docs1, removePunctuation)
docs2 <- tm_map(docs2, removePunctuation)
docs3 <- tm_map(docs3, removePunctuation)

## Removing numbers
docs <- tm_map(docs, removeNumbers)
docs1 <- tm_map(docs1, removeNumbers)
docs2 <- tm_map(docs2, removeNumbers)
docs3 <- tm_map(docs3, removeNumbers)

## To lower and remove stopwords
docs <- tm_map(tm_map(docs, tolower), removeWords, stopwords("english"))
docs1 <- tm_map(tm_map(docs1, tolower), removeWords, stopwords("english"))
docs2 <- tm_map(tm_map(docs2, tolower), removeWords, stopwords("english"))
docs3 <- tm_map(tm_map(docs3, tolower), removeWords, stopwords("english"))
```

```
## To remove extra whitespaces
docs <- tm_map(docs, stripWhitespace)
docs1 <- tm_map(docs1, stripWhitespace)
docs2 <- tm_map(docs2, stripWhitespace)
docs3 <- tm_map(docs3, stripWhitespace)

dtm <- DocumentTermMatrix(docs)
tdm <- TermDocumentMatrix(docs)
freq <- colSums(as.matrix(dtm))
ord <- order(freq)
dtms <- removeSparseTerms(dtm, 0.99)
DTM <- as.matrix(dtms)
doc.index <- as.numeric(document.names)
final.DTM <- matrix(0,nrow=nrow(DTM),ncol=ncol(DTM))
final.DTM[doc.index,] <- DTM
colnames(final.DTM) <- colnames(DTM)

dtm1 <- DocumentTermMatrix(docs1)
tdm1 <- TermDocumentMatrix(docs1)
freq1 <- colSums(as.matrix(dtm1))
ord1 <- order(freq1)
dtms1 <- removeSparseTerms(dtm1, 0.99)
DTM1 <- as.matrix(dtms1)
doc.index1 <- as.numeric(document.names1)
final.DTM1 <- matrix(0,nrow=nrow(DTM1),ncol=ncol(DTM1))
final.DTM1[doc.index1,] <- DTM1
colnames(final.DTM1) <- colnames(DTM1)

dtm2 <- DocumentTermMatrix(docs2)
tdm2 <- TermDocumentMatrix(docs2)
freq2 <- colSums(as.matrix(dtm2))
ord2 <- order(freq2)
dtms2 <- removeSparseTerms(dtm2, 0.99)
DTM2 <- as.matrix(dtms2)
doc.index2 <- as.numeric(document.names2)
final.DTM2 <- matrix(0,nrow=nrow(DTM2),ncol=ncol(DTM2))
final.DTM2[doc.index2,] <- DTM2
colnames(final.DTM2) <- colnames(DTM2)

dtm3 <- DocumentTermMatrix(docs3)
tdm3 <- TermDocumentMatrix(docs3)
freq3 <- colSums(as.matrix(dtm3))
ord3 <- order(freq3)
dtms3 <- removeSparseTerms(dtm3, 0.99)
DTM3 <- as.matrix(dtms3)
doc.index3 <- as.numeric(document.names3)
final.DTM3 <- matrix(0,nrow=nrow(DTM3),ncol=ncol(DTM3))
final.DTM3[doc.index3,] <- DTM3
colnames(final.DTM3) <- colnames(DTM3)
```

```
par(mfrow=c(1,1))

library(wordcloud)
wordcloud(
  words = colnames(final.DTM),
  freq = colSums(final.DTM>0),
  scale = c(2, 0.25),min.freq = 200,colors=brewer.pal(4, "Dark2"))

wordcloud(
  words = colnames(final.DTM1),
  freq = colSums(final.DTM1>0),
  scale = c(2, 0.25),min.freq = 200,colors=brewer.pal(4, "Dark2"))
text(x=0.5, y=1, "Era 1: 2002-2005")

wordcloud(
  words = colnames(final.DTM2),
  freq = colSums(final.DTM2>0),
  scale = c(2, 0.25),min.freq = 200,colors=brewer.pal(4, "Dark2"))
text(x=0.5, y=1, "Era 2: 2002-2009")

wordcloud(
  words = colnames(final.DTM3),
  freq = colSums(final.DTM3>0),
  scale = c(2, 0.25),min.freq = 200,colors=brewer.pal(4, "Dark2"))
text(x=0.5, y=1, "Era 3: 2002-2012")
```

Problem 2.2.a

```
# Problem 2.2a
new.docs <- tm_map(new.docs, stripWhitespace)
docs <- new.docs
lapply(docs[1:2], as.character)
dtm <- DocumentTermMatrix(docs)
dtms <- removeSparseTerms(dtm, 0.99)  ### remove terms that appear in no more than
1% entries
DTM <- as.matrix(dtms)

final.DTM <- matrix(0,nrow=nrow(DTM),ncol=ncol(DTM))
final.DTM[doc.index,] <- DTM

A2P <- read.table("authorPaperBiadj.txt")
A2P <- as.matrix(A2P)
A.DTM <- A2P%*%final.DTM

### Now get author names
authors <- read.table("authorList.txt",stringsAsFactors=FALSE)
head(authors)
authors <- unlist(lapply(authors,function(x)gsub(" ","_",x)))
```

```
head(authors)
rownames(A.DTM)  <- authors

### final coauthorship
Coauthor.Adj <- A2P%*%t(A2P)
dim(Coauthor.Adj)

summary(rowSums(Coauthor.Adj))
authors[which(rowSums(Coauthor.Adj)>50)]

### most collaborative statisticians (2002-2012 on top journals)
summary(rowSums(A2P))
authors[which(rowSums(A2P)>30)]

#statistician selected = V13057 “T_Tony_Cai”

# Isolating only Tony Cai's info
A2P.coords <- data.frame(which(A2P==1, arr.ind = TRUE))
colnames(A2P.coords) <- c("author", "paper")
tonycai.coords <- A2P.coords[which(A2P.coords$author == 3057), ]
tonycai.DOI <- tonycai.coords$paper
tonycai.df <- data.frame(tonycai.DOI, years[tonycai.DOI])
colnames(tonycai.df) <- c("paper.doi", "year")
tonycai <- papers[tonycai.df$paper.doi]

tony1 <- tonycai.df[which(tonycai.df$year %in% c("2002", "2003", "2004", "2005")),]
tony2 <- rbind(tony1, tonycai.df[which(tonycai.df$year %in% c("2006", "2007",
"2008", "2009")),])
tony3 <- rbind(tony2, tonycai.df[which(tonycai.df$year %in% c("2010", "2011",
"2012")),])

# Subsetting the citation adjacency matrix for only Tony Cai, by era
library("igraph")
cit.adj.coords <- data.frame(which(cit.adj==1, arr.ind = TRUE))
tony1.adj <- unique(rbind(cit.adj.coords[which(cit.adj.coords$row %in%
tony1$paper.doi),],
                   cit.adj.coords[which(cit.adj.coords$col %in%
tony1$paper.doi),]))
tony2.adj <- unique(rbind(cit.adj.coords[which(cit.adj.coords$row %in%
tony2$paper.doi),],
                   cit.adj.coords[which(cit.adj.coords$col %in%
tony2$paper.doi),]))
tony3.adj <- unique(rbind(cit.adj.coords[which(cit.adj.coords$row %in%
tony3$paper.doi),],
                   cit.adj.coords[which(cit.adj.coords$col %in%
tony3$paper.doi),]))

tony1.graph <- graph_from_data_frame(tony1.adj, directed = FALSE)
tony2.graph <- graph_from_data_frame(tony2.adj, directed = FALSE)
tony3.graph <- graph_from_data_frame(tony3.adj, directed = FALSE)
```

```
V(tony1.graph)$color <- V(tony1.graph)$name %in% tonycai.DOI
V(tony2.graph)$color <- V(tony2.graph)$name %in% tonycai.DOI
V(tony3.graph)$color <- V(tony3.graph)$name %in% tonycai.DOI

par(mfrow=c(1,3))

plot(tony1.graph, vertex.size=15, vertex.label.cex=0.8,
     layout = layout_nicely)
plot(tony2.graph, vertex.size=15, vertex.label.cex=0.8,
     layout = layout_nicely)
plot(tony3.graph, vertex.size=15, vertex.label.cex=0.8,
     layout = layout_nicely)
```

Problem 2.2.b

```
# Part 2.2b
tony1.collab <- A2P.coords[which(A2P.coords$paper %in% tony1$paper.doi),]
tony2.collab <- A2P.coords[which(A2P.coords$paper %in% tony2$paper.doi),]
tony3.collab <- A2P.coords[which(A2P.coords$paper %in% tony3$paper.doi),]

tony1.collab.flip <- data.frame(tony1.collab$paper, tony1.collab$author)
colnames(tony1.collab.flip) <- c("paper", "author")
tony2.collab.flip <- data.frame(tony2.collab$paper, tony2.collab$author)
colnames(tony2.collab.flip) <- c("paper", "author")
tony3.collab.flip <- data.frame(tony3.collab$paper, tony3.collab$author)
colnames(tony3.collab.flip) <- c("paper", "author")

tony1.collab <- merge.data.frame(tony1.collab, tony1.collab.flip, by="paper")
tony2.collab <- merge.data.frame(tony2.collab, tony2.collab.flip, by="paper")
tony3.collab <- merge.data.frame(tony3.collab, tony3.collab.flip, by="paper")

tony1.collab <- unique(subset(tony1.collab, select = -c(paper)))
tony2.collab <- unique(subset(tony2.collab, select = -c(paper)))
tony3.collab <- unique(subset(tony3.collab, select = -c(paper)))

tony1.collab.graph <- graph_from_data_frame(tony1.collab, directed = FALSE)
tony1.collab.graph <- simplify(tony1.collab.graph, remove.loops = TRUE)
tony2.collab.graph <- graph_from_data_frame(tony2.collab, directed = FALSE)
tony2.collab.graph <- simplify(tony2.collab.graph, remove.loops = TRUE)
tony3.collab.graph <- graph_from_data_frame(tony3.collab, directed = FALSE)
tony3.collab.graph <- simplify(tony3.collab.graph, remove.loops = TRUE)

V(tony1.collab.graph)$color <- V(tony1.collab.graph)$name == "3057"
V(tony2.collab.graph)$color <- V(tony2.collab.graph)$name == "3057"
V(tony3.collab.graph)$color <- V(tony3.collab.graph)$name == "3057"

plot(tony1.collab.graph, vertex.size=15, vertex.label.cex=0.8,
     layout = layout_nicely, edge.arrow.size=.4)
plot(tony2.collab.graph, vertex.size=15, vertex.label.cex=0.8,
```

```
     layout = layout_nicely, edge.arrow.size=.4)
plot(tony3.collab.graph, vertex.size=15, vertex.label.cex=0.8,
     layout = layout_nicely, edge.arrow.size=.4)
```

Problem 2.2.c

```
# Problem 2.2c
A2P.coords <- data.frame(which(A2P==1, arr.ind = TRUE))
colnames(A2P.coords) <- c("author", "paper")
subset1 <- A2P.coords[which(A2P.coords$author == 3057), ]
tonycai <- papers[subset1$paper]

# Code to convert a character list into the document subfolder
dir.create("./TonyCai/")
for(k in 1:length(tonycai))
{
  write(abss[k], file=paste("./TonyCai/", k, sep=""))
}
cname.tonycai <- "./TonyCai/"
docs.tonycai <- Corpus(DirSource(cname.tonycai))
document.names.tonycai <- unlist(meta(docs.tonycai,"id"))
docs.tonycai <- tm_map(docs.tonycai, removePunctuation)
docs.tonycai <- tm_map(docs.tonycai, removeNumbers)
docs.tonycai <- tm_map(tm_map(docs.tonycai, tolower), removeWords,
stopwords("english"))
docs.tonycai <- tm_map(docs.tonycai, stripWhitespace)

dtm.tonycai <- DocumentTermMatrix(docs.tonycai)
tdm.tonycai <- TermDocumentMatrix(docs.tonycai)
freq.tonycai <- colSums(as.matrix(dtm.tonycai))
ord.tonycai <- order(freq.tonycai)
dtms.tonycai <- removeSparseTerms(dtm.tonycai, 0.99)  ### remove terms that appear
in no more than 1% entries
DTM.tonycai <- as.matrix(dtms.tonycai)
doc.index.tonycai <- as.numeric(document.names.tonycai)
final.DTM.tonycai <- matrix(0,nrow=nrow(DTM.tonycai),ncol=ncol(DTM.tonycai))
final.DTM.tonycai[doc.index.tonycai,] <- DTM.tonycai
colnames(final.DTM.tonycai) <- colnames(DTM.tonycai)

wordcloud(
  words = colnames(final.DTM.tonycai),
  freq = freq.tonycai,
  scale = c(2, 0.25), min.freq=10, max.words=20, colors=brewer.pal(4, "Dark2"))
text(x=0.5, y=0.90, "Tony Cai's Top 20 Key Words")
```

Problem 2.2.d

```
# Problem 2.2d

library(tm)
library(dplyr)
library(wordcloud)

tony_cai <- read.csv("T Tony Cai Profile Search.csv")

# The word cloud for Tony Cai's entire academic career
text <- tony_cai$Title
docs <- Corpus(VectorSource(text))

docs <- docs %>%
  tm_map(removeNumbers) %>%
  tm_map(removePunctuation) %>%
  tm_map(stripWhitespace)
docs <- tm_map(docs, content_transformer(tolower))
docs <- tm_map(docs, removeWords, stopwords("english"))

dtm <- TermDocumentMatrix(docs)
matrix <- as.matrix(dtm)
words <- sort(rowSums(matrix),decreasing=TRUE)
df <- data.frame(word = names(words),freq=words)

wordcloud(words = df$word, freq = df$freq, min.freq = 10,
          max.words=20, random.order=FALSE, rot.per=0.35,
          colors=brewer.pal(8, "Dark2"))
text(x=0.5, y=1, "title")

# The word cloud for Tony Cai's early academic years, from 1996-2003
tony_cai1 <- tony_cai[which(tony_cai$Year == 1996 |
                        tony_cai$Year == 1997 |
                        tony_cai$Year == 1998 |
                        tony_cai$Year == 1999 |
                        tony_cai$Year == 2000 |
                        tony_cai$Year == 2001 |
                        tony_cai$Year == 2002 |
                        tony_cai$Year == 2003) ,]

text1 <- tony_cai1$Title
docs1 <- Corpus(VectorSource(text1))

docs1 <- docs1 %>%
  tm_map(removeNumbers) %>%
  tm_map(removePunctuation) %>%
  tm_map(stripWhitespace)
docs1 <- tm_map(docs1, content_transformer(tolower))
```

```
docs1 <- tm_map(docs1, removeWords, stopwords("english"))

dtm1 <- TermDocumentMatrix(docs1)
matrix1 <- as.matrix(dtm1)
words1 <- sort(rowSums(matrix1),decreasing=TRUE)
df1 <- data.frame(word = names(words1),freq=words1)

wordcloud(words = df1$word, freq = df1$freq, min.freq = 5,
          max.words=20, random.order=FALSE, rot.per=0.35,
          scale = c(2, 0.25),
          colors=brewer.pal(8, "Dark2"))
text(x=0.5, y=1, "Tony Cai: 1996-2003")

# The word cloud for Tony Cai's academic years from 2004-2011
tony_cai2 <- tony_cai[which(tony_cai$Year == 2004 |
                              tony_cai$Year == 2005 |
                              tony_cai$Year == 2006 |
                              tony_cai$Year == 2007 |
                              tony_cai$Year == 2008 |
                              tony_cai$Year == 2009 |
                              tony_cai$Year == 2010 |
                              tony_cai$Year == 2011) ,]

text2 <- tony_cai2$Title
docs2 <- Corpus(VectorSource(text2))

docs2 <- docs2 %>%
  tm_map(removeNumbers) %>%
  tm_map(removePunctuation) %>%
  tm_map(stripWhitespace)
docs2 <- tm_map(docs2, content_transformer(tolower))
docs2 <- tm_map(docs2, removeWords, stopwords("english"))

dtm2 <- TermDocumentMatrix(docs2)
matrix2 <- as.matrix(dtm2)
words2 <- sort(rowSums(matrix2),decreasing=TRUE)
df2 <- data.frame(word = names(words2),freq=words2)

wordcloud(words = df2$word, freq = df2$freq, min.freq = 5,
          max.words=20, random.order=FALSE, rot.per=0.35,
          scale = c(2.5, 0.5),
          colors=brewer.pal(8, "Dark2"))
text(x=0.5, y=1, "Tony Cai: 2004-2011")

# The word cloud for Tony Cai's recent academic years, 2012-2020
tony_cai3 <- tony_cai[which(tony_cai$Year == 2012 |
                 tony_cai$Year == 2013 |
                 tony_cai$Year == 2014 |
                 tony_cai$Year == 2015 |
                 tony_cai$Year == 2016 |
```

```
                tony_cai$Year == 2017 |
                tony_cai$Year == 2018 |
                tony_cai$Year == 2019 |
                tony_cai$Year == 2020) ,]

text3 <- tony_cai3$Title
docs3 <- Corpus(VectorSource(text3))

docs3 <- docs3 %>%
  tm_map(removeNumbers) %>%
  tm_map(removePunctuation) %>%
  tm_map(stripWhitespace)
docs3 <- tm_map(docs3, content_transformer(tolower))
docs3 <- tm_map(docs3, removeWords, stopwords("english"))

dtm3 <- TermDocumentMatrix(docs3)
matrix3 <- as.matrix(dtm3)
words3 <- sort(rowSums(matrix3),decreasing=TRUE)
df3 <- data.frame(word = names(words3),freq=words3)

wordcloud(words = df3$word, freq = df3$freq, min.freq = 5,
          max.words=20, random.order=FALSE, rot.per=0.35,
          scale = c(2, 0.25),
          colors=brewer.pal(8, "Dark2"))
text(x=0.5, y=1, "Tony Cai: 2012-2020")

# The word cloud from 2002-2012:
tony_cai4 <- tony_cai[which(tony_cai$Year == 2002 |
                              tony_cai$Year == 2003 |
                              tony_cai$Year == 2004 |
                              tony_cai$Year == 2005 |
                              tony_cai$Year == 2006 |
                              tony_cai$Year == 2007 |
                              tony_cai$Year == 2008 |
                              tony_cai$Year == 2009 |
                              tony_cai$Year == 2010 |
                              tony_cai$Year == 2011 |
                              tony_cai$Year == 2012),]

text4 <- tony_cai4$Title
docs4 <- Corpus(VectorSource(text4))

docs4 <- docs4 %>%
  tm_map(removeNumbers) %>%
  tm_map(removePunctuation) %>%
  tm_map(stripWhitespace)
docs4 <- tm_map(docs4, content_transformer(tolower))
docs4 <- tm_map(docs4, removeWords, stopwords("english"))

dtm4 <- TermDocumentMatrix(docs4)
```

```
matrix4 <- as.matrix(dtm4)
words4 <- sort(rowSums(matrix4),decreasing=TRUE)
df4 <- data.frame(word = names(words4),freq=words4)

wordcloud(words = df4$word, freq = df4$freq, min.freq = 5,
          max.words=20, random.order=FALSE, rot.per=0.35,
          scale = c(3, 0.25),
          colors=brewer.pal(8, "Dark2"))
text(x=0.5, y=1, "Tony Cai: 2002-2012")
```

Problem 3.1

```
# Problem 3.1

### Load packages
install.packages("reshape2")
library(reshape2)
library(plyr)
library(ggplot2)
install.packages("wbstats")
library(wbstats)
library(dplyr)
install.packages("gganimate")
library(gganimate)
install.packages("gifski")
library(gifski)
library(dplyr)

### Read in data
confirmed_world <-
read.csv("https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_cov
id_19_data/csse_covid_19_time_series/time_series_covid19_confirmed_global.csv",
stringsAsFactors = FALSE, check.names =  FALSE)

death_world <-
read.csv("https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_cov
id_19_data/csse_covid_19_time_series/time_series_covid19_deaths_global.csv",strings
AsFactors = FALSE, check.names =  FALSE)

recovered_world <-
read.csv("https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_cov
id_19_data/csse_covid_19_time_series/time_series_covid19_recovered_global.csv",stri
ngsAsFactors = FALSE, check.names =  FALSE)

confirmed_world <- reshape2::melt(confirmed_world, id.vars = c("Province/State",
"Country/Region", "Lat", "Long"), variable.name = "Date", value.name = "Confirmed")

death_world <- reshape2::melt(death_world, id.vars = c("Province/State",
```

```
"Country/Region", "Lat", "Long"), variable.name = "Date", value.name = "Death")

recovered_world <- reshape2::melt(recovered_world, id.vars = c("Province/State",
"Country/Region", "Lat", "Long"), variable.name = "Date", value.name = "Recovered")

### Create one table with confirmed, death, and recovered data
world_history_data <- dplyr::left_join(confirmed_world, death_world, by =
c("Province/State", "Country/Region", "Lat", "Long", "Date"))

world_history_data <- dplyr::left_join(world_history_data, recovered_world, by =
c("Province/State", "Country/Region", "Lat", "Long", "Date"))

### Reformat date
world_history_data$Date <- as.Date(as.character(world_history_data$Date), format =
c("%m/%d/%y"))

# Reformat column name
colnames(world_history_data) <- make.names(colnames(world_history_data))

# Sum columns to have cumulative data
world.summary.data <- ddply(world_history_data,.(Country.Region, Date),function(x){
  colSums(x[,c("Confirmed","Death","Recovered")])
})

### Too many countries; cut down to 20 countries with most infected
lastday <- max(world.summary.data$Date)

world.summary.data <- world.summary.data[world.summary.data$Date<=lastday,]

yesterday.data <- world.summary.data[world.summary.data$Date==lastday,]

# Sort yesterday's data from greatest to least number of confirmed cases
sort.index <- sort(yesterday.data$Confirmed,decreasing=TRUE,index.return=TRUE)$ix

# Take only the 20 countries with the most confirmed cases
yesterday.data.major  <- yesterday.data[sort.index[1:20],]

# Take just the country names
yesterday.data.major  <-
data.frame(Country.Region=yesterday.data$Country.Region[sort.index[1:20]])

yesterday.data.major$Country.Region  <-
as.character(yesterday.data.major$Country.Region )

# Grab all data about 20 relevant countries from all dates
major.summary.data <- dplyr::inner_join(world.summary.data,yesterday.data.major,by
= "Country.Region")

### Defining colors
cols <- matrix(c(brewer.pal(9,"Set1"),brewer.pal(11,"Set3")),ncol=1)
```

```
# Map each color to a country
rownames(cols) <- unique(major.summary.data$Country.Region)

### Sort according to count
major.summary.data$Country.Region <- factor(major.summary.data$Country.Region,
                                            levels =
rev(yesterday.data.major$Country.Region))

### Population Data (2018 seems to be most recent)
pop_data <- wb(country =
c('AUT','BEL','BRA','CAN','CHN','FRA','DEU','IRN','ISR','ITA','KOR','NLD','PRT','RU
S','ESP','SWE','CHE','TUR','GBR','USA'),
               indicator = "SP.POP.TOTL", startdate = 2018, enddate = 2018)

# Reformatting stuff
pop.data <- pop_data[,c("country","value")]
pop.data$ppm <- pop.data$value/1000000
pop.data$country[pop.data$country=='Iran, Islamic Rep.'] <- "Iran"
pop.data$country[pop.data$country=='Korea, Rep.'] <- "Korea, South"
pop.data$country[pop.data$country=='United States'] <- "US"
pop.data$country[pop.data$country=='Russian Federation'] <- 'Russia'
colnames(pop.data) <- c("Country.Region","population","ppm")
major.summary.data <- dplyr::left_join(major.summary.data, pop.data, by =
c("Country.Region"))
major.summary.data$infected.per.mil <-
major.summary.data$Confirmed/major.summary.data$ppm


### Part 3.1 Graphical Representation
problem3.1 <- ggplot(major.summary.data,
       aes(x = Date, y = infected.per.mil, col = Country.Region)) +
  geom_line(lwd = 0.5) + geom_point(size = 0.75) +
  scale_color_manual(values = cols) + theme_minimal() +
  labs(title = "Confirmed Cases Over Time", x="Date", y="Infected Per Million",
color = "Country/Region") +
  scale_x_date(date_labels = "%m-%d", date_breaks = "1 day") +
  theme(text = element_text(size=10),
        plot.title=element_text(hjust=0.5, family="Times", face="bold"),
        axis.title.x=element_text(family="Times", face="bold"),
        axis.title.y=element_text(family="Times", face="bold"),
        axis.text.x = element_text(angle = 90, size=6),
        legend.title=element_text(family="Times", face="bold"),
        legend.text=element_text(family="Times"),
        legend.position = "right")
problem3.1

# The animation for this problem
animated_problem3.1 <- problem3.1 + geom_point(aes(group = seq_along(Date)),
size=0.75) + transition_reveal(Date)
```

```
animate(animated_problem3.1, nframes=30, renderer =
gifski_renderer("problem3.1.gif"), height=700, width=700)
```

Problem 3.2

```
### Part 3.2

#only keep the 10 most affected country names because graph too busy with 20
yesterday.data.major2  <- yesterday.data[sort.index[1:10],1]
major.summary.data2 <- major.summary.data[(major.summary.data$Country.Region %in%
yesterday.data.major2),]

# Death to Recovery Ratio
daily.death <- major.summary.data2 %>% group_by(Country.Region) %>%
mutate(new.death = Death - lag(Death, default=first(Death)))
daily.death <- daily.death %>% group_by(Country.Region) %>% mutate(new.recovery =
Recovered - lag(Recovered, default=first(Recovered)))


new.death.aggregate <- colSums(matrix(daily.death$new.death, nrow=9))
new.recovery.aggregate <- colSums(matrix(daily.death$new.recovery, nrow=9))
points <- new.death.aggregate/ (new.death.aggregate+new.recovery.aggregate)
points2 <- new.recovery.aggregate/(new.death.aggregate+new.recovery.aggregate)
points[is.nan(points)]<-0
points2[is.nan(points2)]<-0
ind <- seq(from=9,to=810,by=9)
graph.points <- daily.death[ind,]
graph.points$points <- points
graph.points$points2 <- points2

# Death Graphical Representation
ggplot(graph.points,
       aes(x = Date, y = points, col = Country.Region)) + geom_line(lwd = 1) +
geom_point(size = 2) +
  scale_color_manual(values = cols) + theme_minimal() + ylab("Death Ratio") +
  xlab("Date") + labs(title = "Death Ratio" , color = "Country/Region") +
scale_x_date(date_labels = "%m-%d",date_breaks = "3 day") + theme(text =
element_text(size = 14, face = "bold"),axis.text.x = element_text(angle = 90),
legend.position = "right")

# Recovery Graphical Representation
ggplot(graph.points,
      aes(x = Date, y = points2, col = Country.Region)) + geom_line(lwd = 1) +
  geom_point(size = 2) +
  scale_color_manual(values = cols) + theme_minimal() + ylab("Recovery Ratio") +
  xlab("Date") + labs(title = "Recovery Ratio" , color = "Country/Region") +
scale_x_date(date_labels = "%m-%d",date_breaks = "3 day") + theme(text =
element_text(size = 14, face = "bold"),axis.text.x = element_text(angle = 90),
legend.position = "right")
```

Problem 3.4

```
# Problem 3.4

# Which countries show the lowest total infected proportion?
ggplot(major.summary.data[major.summary.data$Date==lastday,], aes(x =
Country.Region, y = (Confirmed/population)*100, fill = Country.Region)) +
  geom_col() + scale_fill_manual(values = cols) + theme_minimal() +
  ylab("Percent Infected") + xlab("Country") + labs(title= "Total Infection
Percentage", color = "Country/Region") + coord_flip() +
  theme(text = element_text(size = 12), axis.text.x = element_text(angle = 90),
        legend.position = "none")

# Infection Rates
daily.infected <- major.summary.data %>% group_by(Country.Region) %>%
mutate(new.inf = Confirmed - lag(Confirmed, default=Confirmed[1]))
#only keep the 10 most affected country names because graph too busy with 20
yesterday.data.major2  <- yesterday.data[sort.index[1:10],1]


daily.infected <- daily.infected[(daily.infected$Country.Region %in%
yesterday.data.major2),]
ggplot(daily.infected,
       aes(x = Date, y = new.inf/population, col = Country.Region)) + geom_line(lwd
= 1) + geom_point(size = 2) +
  scale_color_manual(values = cols) + theme_minimal() + ylab("Infection Rate") +
  xlab("Date") + labs(title = "Infection Rate of Countries" , color =
"Country/Region") + scale_x_date(date_labels = "%m-%d",date_breaks = "2 day") +
theme(text = element_text(size = 14, face = "bold"),axis.text.x =
element_text(angle = 90), legend.position = "right")
```