Project 1

Name: Alex Yu Michael Minniear:

2020-10-25

Contents

The following packages and libraries utilized in this project are provided for reference.

```r
knitr::opts_chunk$set(include = TRUE)
library(ggplot2)
library(dplyr)
library(maps)
library(mapproj)
library(readr)
library(ggrepel)
library(magrittr)
library(tidyr)
library(directlabels)
library(geosphere)
```

Background

The World Health Organization has recently employed a new data science initiative, *CSIT-165*, that uses data science to characterize pandemic diseases. *CSIT-165* disseminates data driven analyses to global decision makers.

CSIT-165 is a conglomerate comprised of two fabricated entities: *Global Health Union (GHU)* and *Private Diagnostic Laboratories (PDL)*. Your and your partner's role is to play a data scientist from one of these two entities.

Data

> 2019 Novel Coronavirus COVID-19 (2019-nCoV) Data Repository by John Hopkins CSSE

Data for 2019 Novel Coronavirus is operated by the John Hopkins University Center for Systems Science and Engineering (JHU CSSE). Data includes daily time series CSV summary tables, including confirmations, recoveries, and deaths. Country/region are countries/regions hat conform to World Health Organization

(WHO). Lat and Long refer to coordinates references for the user. Date fields are stored in MM/DD/YYYY format.

Project Objectives

Objective 1

The goal of objective 1 is to predict where COVID-19 originated from by reviewing reported cases on the first recorded day for confirmation, recovery, and death. The following code was applied to data from John Hopkins University Center for Systems Science and Engineering for 2019 Novel Coronavirus to help predict it's origin:

```
# Assign variable for file names
filename = "time_series_covid19_deaths_global.csv"
filename_rec = "time_series_covid19_recovered_global.csv"
filename_conf = "time_series_covid19_confirmed_global.csv"
# Read files and set variables for with the covid19 global deaths, recoveries, and confirmed cases, res
deaths <- read_delim(file = filename, delim = ",")
recovered <- read_delim(file = filename_rec, delim = ",")
confirmed <- read_delim(file = filename_conf, delim = ",")
head(deaths)
```

```
## # A tibble: 6 x 275
##   `Province/State` `Country/Region`   Lat   Long `1/22/20` `1/23/20` `1/24/20`
##   <chr>            <chr>            <dbl>  <dbl>     <dbl>     <dbl>     <dbl>
## 1 <NA>             Afghanistan       33.9  67.7          0         0         0
## 2 <NA>             Albania           41.2  20.2          0         0         0
## 3 <NA>             Algeria           28.0   1.66         0         0         0
## 4 <NA>             Andorra           42.5   1.52         0         0         0
## 5 <NA>             Angola           -11.2  17.9          0         0         0
## 6 <NA>             Antigua and Bar~  17.1 -61.8          0         0         0
## # ... with 268 more variables: `1/25/20` <dbl>, `1/26/20` <dbl>,
## #   `1/27/20` <dbl>, `1/28/20` <dbl>, `1/29/20` <dbl>, `1/30/20` <dbl>,
## #   `1/31/20` <dbl>, `2/1/20` <dbl>, `2/2/20` <dbl>, `2/3/20` <dbl>,
## #   `2/4/20` <dbl>, `2/5/20` <dbl>, `2/6/20` <dbl>, `2/7/20` <dbl>,
## #   `2/8/20` <dbl>, `2/9/20` <dbl>, `2/10/20` <dbl>, `2/11/20` <dbl>,
## #   `2/12/20` <dbl>, `2/13/20` <dbl>, `2/14/20` <dbl>, `2/15/20` <dbl>,
## #   `2/16/20` <dbl>, `2/17/20` <dbl>, `2/18/20` <dbl>, `2/19/20` <dbl>,
## #   `2/20/20` <dbl>, `2/21/20` <dbl>, `2/22/20` <dbl>, `2/23/20` <dbl>,
## #   `2/24/20` <dbl>, `2/25/20` <dbl>, `2/26/20` <dbl>, `2/27/20` <dbl>,
## #   `2/28/20` <dbl>, `2/29/20` <dbl>, `3/1/20` <dbl>, `3/2/20` <dbl>,
## #   `3/3/20` <dbl>, `3/4/20` <dbl>, `3/5/20` <dbl>, `3/6/20` <dbl>,
## #   `3/7/20` <dbl>, `3/8/20` <dbl>, `3/9/20` <dbl>, `3/10/20` <dbl>,
## #   `3/11/20` <dbl>, `3/12/20` <dbl>, `3/13/20` <dbl>, `3/14/20` <dbl>,
## #   `3/15/20` <dbl>, `3/16/20` <dbl>, `3/17/20` <dbl>, `3/18/20` <dbl>,
## #   `3/19/20` <dbl>, `3/20/20` <dbl>, `3/21/20` <dbl>, `3/22/20` <dbl>,
## #   `3/23/20` <dbl>, `3/24/20` <dbl>, `3/25/20` <dbl>, `3/26/20` <dbl>,
## #   `3/27/20` <dbl>, `3/28/20` <dbl>, `3/29/20` <dbl>, `3/30/20` <dbl>,
## #   `3/31/20` <dbl>, `4/1/20` <dbl>, `4/2/20` <dbl>, `4/3/20` <dbl>,
## #   `4/4/20` <dbl>, `4/5/20` <dbl>, `4/6/20` <dbl>, `4/7/20` <dbl>,
## #   `4/8/20` <dbl>, `4/9/20` <dbl>, `4/10/20` <dbl>, `4/11/20` <dbl>,
## #   `4/12/20` <dbl>, `4/13/20` <dbl>, `4/14/20` <dbl>, `4/15/20` <dbl>,
## #   `4/16/20` <dbl>, `4/17/20` <dbl>, `4/18/20` <dbl>, `4/19/20` <dbl>,
## #   `4/20/20` <dbl>, `4/21/20` <dbl>, `4/22/20` <dbl>, `4/23/20` <dbl>,
## #   `4/24/20` <dbl>, `4/25/20` <dbl>, `4/26/20` <dbl>, `4/27/20` <dbl>,
## #   `4/28/20` <dbl>, `4/29/20` <dbl>, `4/30/20` <dbl>, `5/1/20` <dbl>,
## #   `5/2/20` <dbl>, `5/3/20` <dbl>, ...
```

Initial review of datasets found them to all have exactly the same structure. As shown above in the deaths table, the first four columns indicate the location of the reported while the remaining columns are reported numbers (integers) by consecutive days. As shown, the first date recorded is January 22, 2020.

The next step in determining the possible origin is to sort each of the tables from highest to lowest on the initial reported date of January 22, 2020. By determining which locations had the highest initial incidents may indicate where the origin was. We begin by analyzing the table showing the first confirmed cases.

```r
# Filter table to cases greater than zero for date 1/22/20
cases <- confirmed[confirmed$`1/22/20` > 0, 1:5]
# Sort cases from highest to lowest
cases_sort <- cases[order(cases$'1/22/20', decreasing = TRUE),]
head(cases_sort)
```

```
## # A tibble: 6 x 5
##   `Province/State` `Country/Region`   Lat  Long `1/22/20`
##   <chr>            <chr>            <dbl> <dbl>     <dbl>
## 1 Hubei            China             31.0  112.       444
## 2 Guangdong        China             23.3  113.        26
## 3 Beijing          China             40.2  116.        14
## 4 Zhejiang         China             29.2  120.        10
## 5 Shanghai         China             31.2  121.         9
## 6 Chongqing        China             30.1  108.         6
```

As can be seen by the above table, Hubei Province in China has the highest initial confirmed cases of Covid-19 by a large margin. By January 22, 2020 Hubei had 444 reported cases. The next highest location with only 26 cases on January 22, 2020 was Guangdong.

Now that we've looked at where the initial confirmed cases were highest, we'll repeat that procedure for deaths and recoveries. We will create a function to automate this check and provide the top result for each data set.

```r
# Create a function to determine the max value on initial date for each file
origin <- function(df){
  print(df[which.max(df$`1/22/20`),1:5])
}
# Check to country and province for each file with highest initial occurences
origin(confirmed)
```

```
## # A tibble: 1 x 5
##   `Province/State` `Country/Region`   Lat  Long `1/22/20`
##   <chr>            <chr>            <dbl> <dbl>     <dbl>
## 1 Hubei            China             31.0  112.       444
```

```r
origin(deaths)
```

```
## # A tibble: 1 x 5
##   `Province/State` `Country/Region`   Lat  Long `1/22/20`
##   <chr>            <chr>            <dbl> <dbl>     <dbl>
## 1 Hubei            China             31.0  112.        17
```

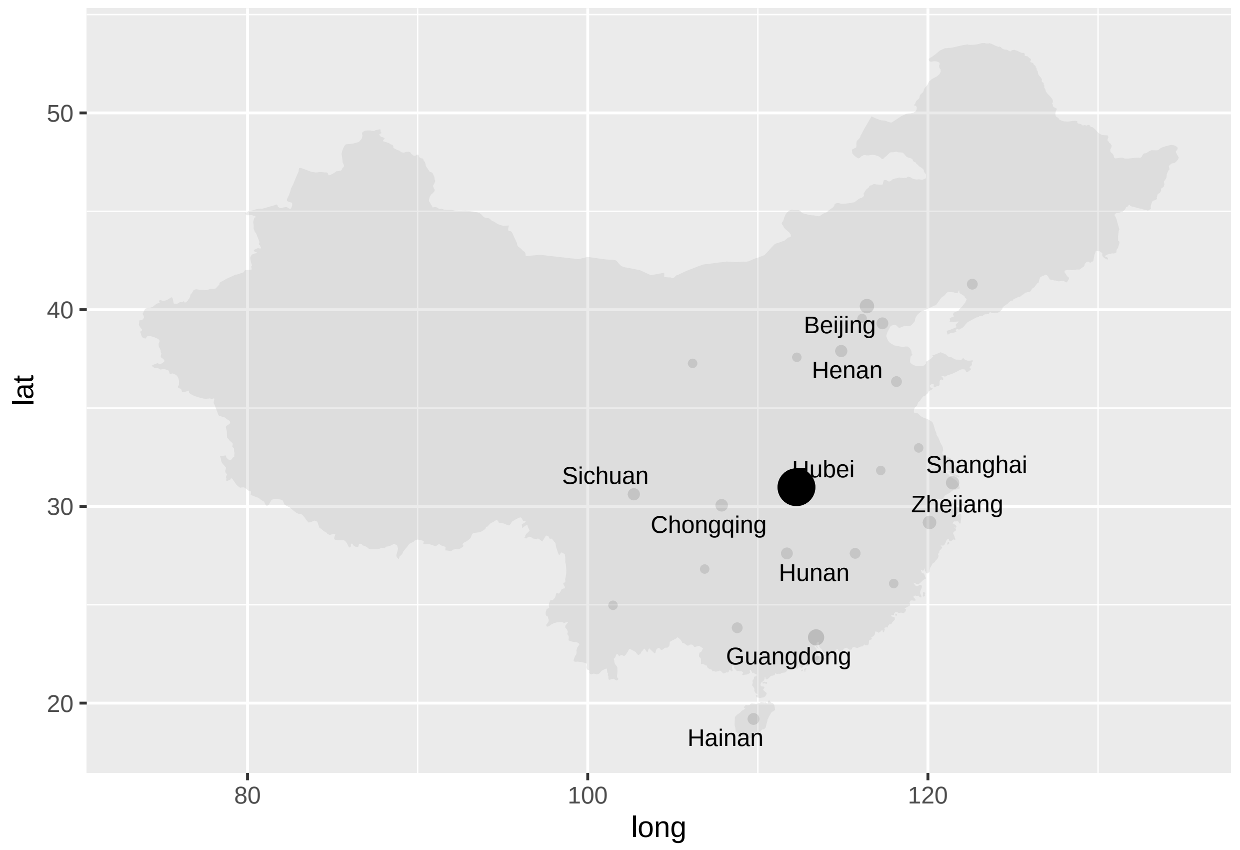
```r
origin(recovered)
```

```
## # A tibble: 1 x 5
##   `Province/State` `Country/Region`   Lat  Long `1/22/20`
##   <chr>            <chr>            <dbl> <dbl>     <dbl>
## 1 Hubei             China             31.0  112.        28
```

After running each data set through our function, we find that Hubei China had the highest initial reported confirmed cases, deaths, *and* recoveries. With the highest number of initial cases for all three categories, Hubei Province in China is predicted to be the origin of the COVID-19 virus.

To get a geographic sense of where Hubei, China is respective to the other top locations for initial confirmed cases, we created the following bubble map where the size and shade of the dots indicate the size of the initial confirmed cases by location. Location names for the top 10 locations are provided.

```
# Map the location of the intial occurences with bubble chart
CH <- map_data("world") %>% filter(region=="China")
# Create a data frame with data from China only
cases_china <- cases_sort[cases_sort$`Country/Region` == "China",]
# Change column names to make it easier to use mapping functions
newcolnames <- c("Province","Country","lat","long","cases")
colnames(cases_china) <- newcolnames
# Use ggplot to plot top locations of initial confirmed cases using cases for bubble size
ggplot() +
  geom_polygon(data = CH, aes(x=long, y = lat, group = group), fill="grey", alpha=0.3) +
  geom_point(data=cases_china, aes(x=long, y=lat, alpha=cases, size = cases)) +
  geom_text_repel(data=cases_china %>% head(10), aes(x=long, y=lat, label=Province), size=3) +
  theme(legend.position="none")
```



Objective 2

The goal of objective 2 is to located the area of the world where the most recent *first* confirmed cases of Covid-19 have been reported. In order to truly identify which areas of the globe have most recently reported an initial positive number of confirmed cases, we must compare each locations number of days without a confirmed cases. Those areas with the longest period without a confirmed case, but now have confirmed cases, are the locations of the nearest reported outbreaks.

Because each of the data set columns are in order by date, the last column of each data set represents the latest information. After refreshing the data sets to get the most current date, the following code was applied to locate what areas of the world have their first confirmed cases. We first work with the confirmed cases data set, and show the most recent date for which we have data.

```
#Get the most recent date of data (last column of dataset) and assign to variable max_date
dates <- colnames(confirmed)
max_date <- dates[length(dates)]
print(max_date)
```

```
## [1] "10/21/20"
```

The latest date we have for our data set is October 21, 2020. Review of the data set also revealed that each column is cumulative, meaning that all locations start at zero and increase for each future column where additional cases are identified. By seeing which areas of the world have gone the longest without a confirmed cases (most number of zeroes), we can determine which areas of the world have recently reported their first initial cases. The following code was applied to solve this problem using the confirmed cases data set.

```
# Add column that sums to total number of days with no confirmed cases
confirmed$zeroes <- rowSums(confirmed[,5:ncol(confirmed)]==0)
# Sort countries and states by most days without confirmed cases using max_date
recent_sort <- confirmed[order(confirmed$zeroes, decreasing = TRUE),]
# Reduce data set to top 10 with most number of consecutive days without a reported case
recent_filter <- recent_sort[0:10,]
# Show relevant columns of location and latest dates
print(recent_filter %>% select(1:4,278:ncol(recent_filter)))
```

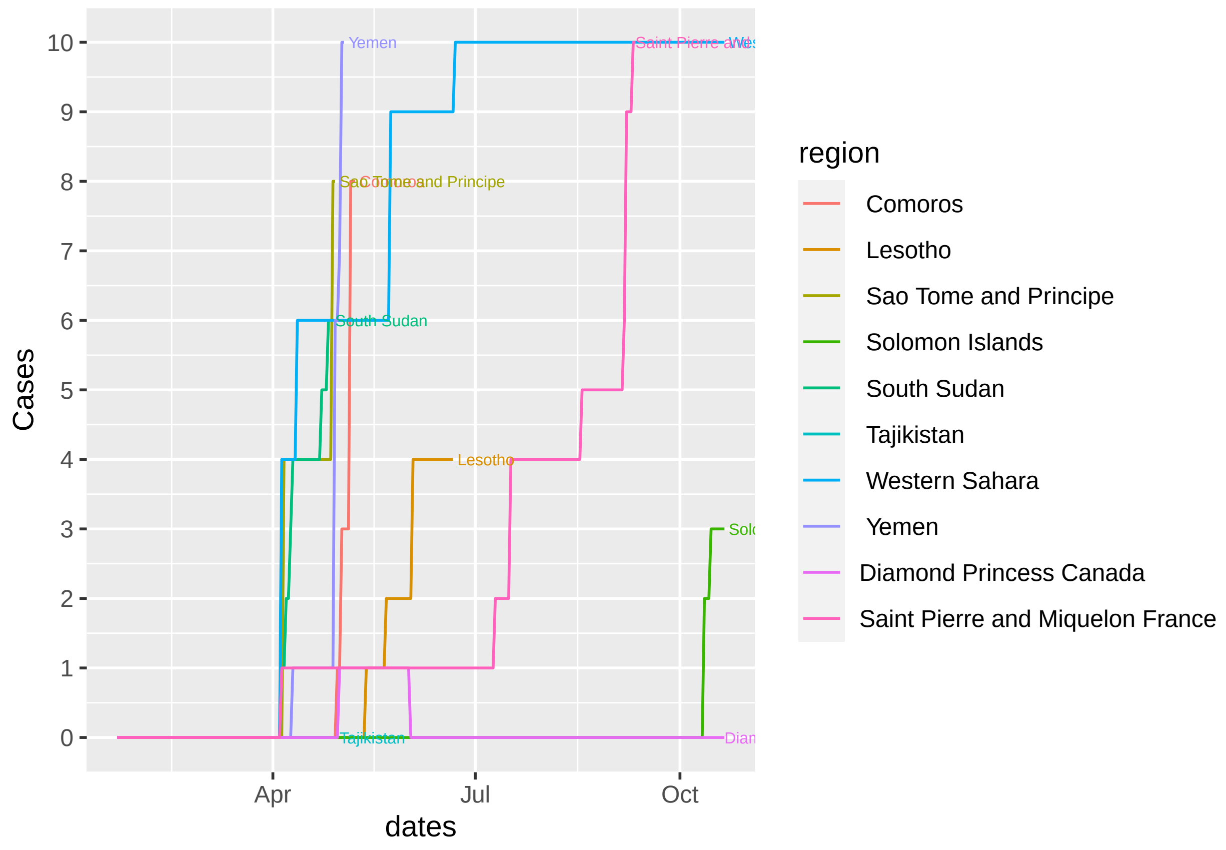
```
## # A tibble: 10 x 6
##    `Province/State`         `Country/Region`        Lat   Long `10/21/20` zeroes
##    <chr>                    <chr>                 <dbl>  <dbl>      <dbl>  <dbl>
##  1 <NA>                     Solomon Islands      -9.65  160.            3    264
##  2 Diamond Princess         Canada                0      0              0    242
##  3 <NA>                     Lesotho             -29.6    28.2        1918    112
##  4 <NA>                     Comoros             -11.6    43.3         504     99
##  5 <NA>                     Tajikistan           38.9    71.3       10613     99
##  6 <NA>                     Yemen                15.6    48.5        2057     79
##  7 <NA>                     Sao Tome and Princ~   0.186   6.61        935     75
##  8 Saint Pierre and Miquel~ France               46.9   -56.3          16     74
##  9 <NA>                     South Sudan           6.88   31.3        2870     74
## 10 <NA>                     Western Sahara       24.2   -12.9          10     74
```

As can be seen by the above table, the area of the world with the highest number of consecutive days (zeroes) without a reported case is the Solomon Islands. It's interesting to note here that the cruise ship Diamond Princess shows 242 days without a case, but shows zero cases on October 21st. After reviewing the data, we found that the Diamond Princess reported a single case on May 1st but then removed that case on June 2nd. It's best that this simply be omitted, but it's worth noting here.

Following Solomon Islands the second location with the most recent reported initial cases is in Lesotho. In order to see this more clearly, a line graph was built showing the number of confirmed cases by region by date so we can visually see the initial cases reported by these regions.

```
# Pivot the recent_filter table for plotting
recent_pvt <- recent_filter %>%
  pivot_longer(5:ncol(recent_filter), names_to = "dates", values_to = "confirmed_cases")
```

```r
# Convert column date names to actual date values
recent_pvt$dates <- as.Date(recent_pvt$dates , format = "%m/%d/%y")
#Change column names to simplify using ggplot2 library
new_cols <- c('state','country','lat','long','dates','confirmed_cases')
colnames(recent_pvt) <- new_cols
# Clarify origin name by removing NA from State if no Province/State is provided (is NA)
recent_pvt$state[is.na(recent_pvt$state)] <- ""
recent_pvt$region <- paste(recent_pvt$state,recent_pvt$country)
# Plot top 10 locations with longest streak without cases and scale for effect
ggplot(recent_pvt, mapping = aes(x = dates, y = confirmed_cases, color = region)) +
  geom_line() +
  geom_dl(aes(label = region), method = list(cex = 0.5,dl.combine("last.points"))) +
  scale_y_continuous(name = "Cases", breaks = 0:10, limits=c(0,10))
```



As can be seen by the chart, the Solomon Islands is where the most recent initial cases of Covid-19 have been reported in the world. All other locations other than those top 10 shown here reported cases well before the Solomon Islands.

Because the data sets for deaths and recoveries has the same structure, we can create functions to repeat this process for locations reporting their first deaths and their first recoveries.

```r
# Create a function that returns the date of the first confirmed case
# Take as inputs the data set(df), the state, and the country.  Defaults are NA.
confirmation <- function(df = NA, state = NA, country = NA){
# Create logic for when state is NA use country/region for data lookup
    if (is.na(state)) {
    column <- 'Country/Region'
    state <- country
    } else {
      column <- 'Province/State'
    }
# Get row number where data resides for given location
  row <- which(df[,column] == state)
# Get column number
  column <- df$zeroes[row]
# Create vector of row and column numbers and return
  result <- c(row,column)
  return(result)
}
# Create function where only dataframe is passed, but output is areas of most recent first cases
most_recent <- function(df){

  #Find most current date in date set
  max_date <- colnames(df[,ncol(df)])

  #Remove any rows where there are no reported cases
  df <- df[df[,max_date] != 0,]

  #Add column that counts number of days with no cases
  df$zeroes <- rowSums(df[,5:ncol(df)]==0)

  #Create vector of columns to specify data to pull
  columns <- c("Province/State","Country/Region","zeroes")
  location <- df[,columns]

  #Create empty vectors
  state_vec <- c()
  country_vec <- c()
  date_vec <- c()
#Create for loop that pulls state, country, and use confirmation function to return date of first cases
  for (i in 1:nrow(df)) {

    #Get state and country for each row
    col1 <- df[i,1]
    col2 <- df[i,2]
    state <- col1[[1]]
    country <- col2[[1]]

    #Run confirmation function on data set, state, and country
    location <- confirmation(df, state = state, country = country)

    #Append created vectors to store results
    state_vec <- append(state_vec, state)
    country_vec <- append(country_vec, country)

    #Get table position from return of confirmation function for date of most recent case
    row <- location[1]
    column <- location[2]

    #Retrieve resulting date and convert to date
    result <- as.Date(colnames(df[row,column+5]),"%m/%d/%y")

    #Append date vector with dates for each state/region
    date_vec <- append(date_vec,result)
  }
  # Get max date from vectored results (date of most recent reported case)
  max <- max(date_vec)

  # Find position in vector
  loc <- which(date_vec %in% max)

  # Convert final date to string for returned vector
  final_date <- format(date_vec[loc], "%m/%d/%y")

  # Create vector with state, country, and date result
  result2 <- c(state_vec[loc],country_vec[loc],final_date)

  #print result location with most recent case
  print(result2, row.names = FALSE)
}
#add_zeroes <- function(df){
#  df$zeroes <- rowSums(df[,5:ncol(df)]==0)
#  return(df)
#}
#Pass each data set to function
most_recent(confirmed)
```

```
## [1] NA                "Solomon Islands" "10/12/20"
```

```r
most_recent(deaths)
```

```
## [1] "Bonaire, Sint Eustatius and Saba" "Netherlands"
## [3] "09/16/20"
```

```r
most_recent(recovered)
```

```
## [1] NA         "Lesotho"  "05/28/20"
```

By passing each data set through the function, we can see that Solomon Islands has the most recent initial confirmed case on October 12th, the Netherland region of Bonaire, Sint Eustatius and Saba has the most recent initial deaths reported on September 16th, and Lesotho has the most recent initial recoveries reported on May 28th.

Objective 3

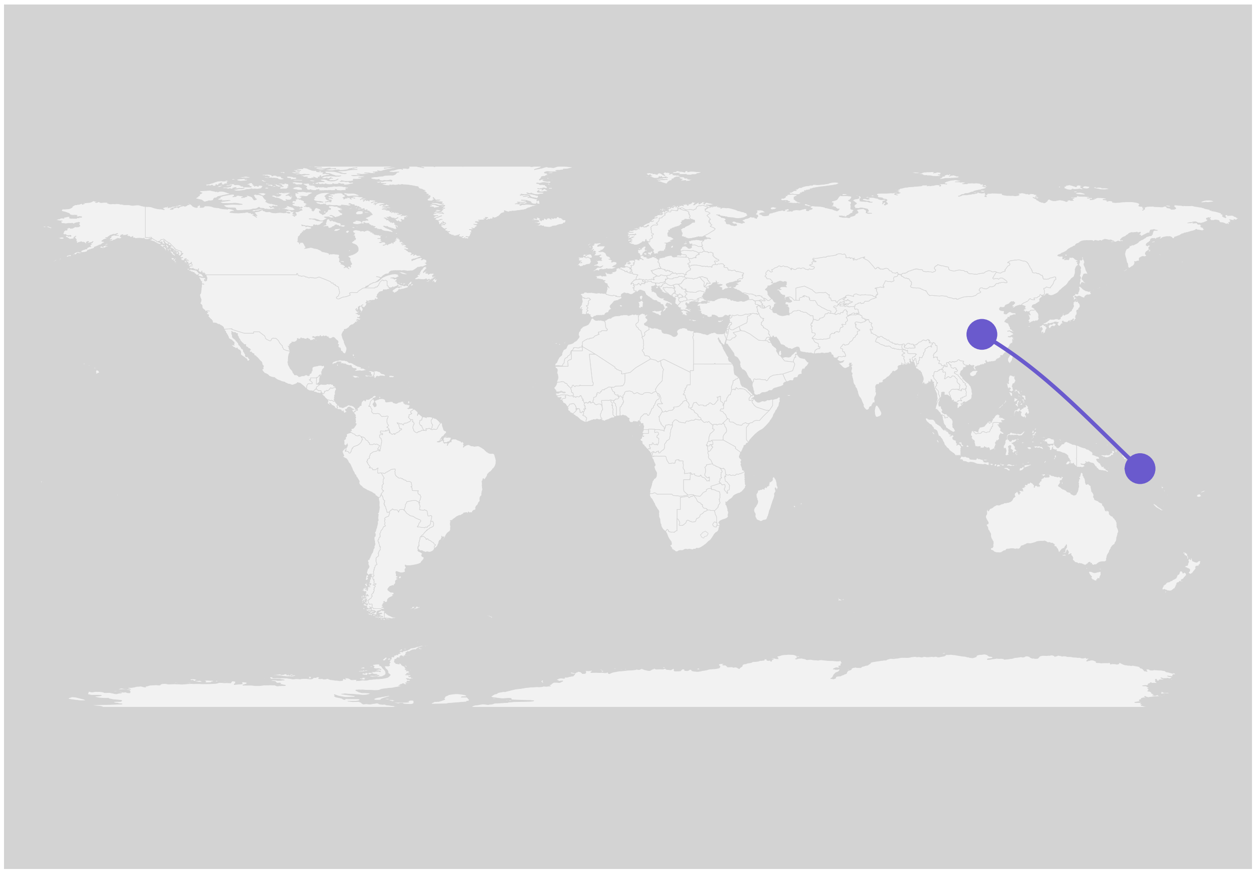
Objective 3 is to determine how far away the most recent first case reported in the Solomon Island is from where the first confirmed case was reported, predicted to be Hubei, China. The following code provides the solution in miles and plots a map showing their positions in the world.

```r
# Get latitude and longitude of Hubei from data set
hubei_lat <- confirmed[which(confirmed[,"Province/State"] == "Hubei"),"Lat"]
hubei_long <- confirmed[which(confirmed[,"Province/State"] == "Hubei"),"Long"]
# Get latitude and longitude of Solomon Islands from data set
solomon_lat <- confirmed[which(confirmed[,"Country/Region"] == "Solomon Islands"),"Lat"]
solomon_long <- confirmed[which(confirmed[,"Country/Region"] == "Solomon Islands"),"Long"]
# Create coordinate vectors for each location
hubei_coord <- c(hubei_long[[1]],hubei_lat[[1]])
```

```r
solomon_coord <- c(solomon_long[[1]], solomon_lat[[1]])
# Calculate distance in meters using geosphere::distm
meters <- geosphere::distm(x = hubei_coord, y = solomon_coord)
# Convert meters to miles
 miles <- meters * 0.00062137
# Assign variables
origin_city <- "Hubei"
distance_in_miles <- miles
origin_country <- "China"
recent_city <- "Solomon Islands"
print(paste0(recent_city, " is ",round(distance_in_miles,0), " miles away from ", origin_city, ", ",orig
```

```
## [1] "Solomon Islands is 4228 miles away from Hubei, China."
```

```r
# World map is available in the maps package
# No margin
#par(mar=c(0,0,0,0))
# World map
map('world',
    col="#f2f2f2", fill=TRUE, bg="lightgray", lwd=0.05,
    mar=rep(0,4),border=0, ylim=c(-80,80)
)
# Cities
Hubei <- c(hubei_long[[1]],hubei_lat[[1]])
Solomon_Islands <- c(solomon_long[[1]], solomon_lat[[1]])
# Data frame
data <- rbind(Hubei, Solomon_Islands) %>%
  as.data.frame()
colnames(data) <- c("long","lat")
# Dot for cities
points(x=data$long, y=data$lat, col="slateblue", cex=3, pch=20)
# Compute the connection between Hubei and Solomon Islands
inter <- gcIntermediate(Hubei,  Solomon_Islands, n=50, addStartEnd=TRUE, breakAtDateLine=F)
# Show this connection
lines(inter, col="slateblue", lwd=2)
```



Initial results seemed a little suprising, in that the map reveals that the locations are not that far apart from a global perspective. What this perhaps reveals more is that remote locations such as islands which have more capability to isolate and prohibit travel may be able to better prevent initial cases from emerging.

Objective 4

Objective 4.1 The purpose of objective 4.1 is to evaluate and compare the risk score and burden scores of locations throughout the world and provide insights into the results. Results were determined using the following code:

```r
# Extract the country, province, and lat/long labels from each of the datasets
# Remember that the Confirmed dataset has fewer rows of data than the others
dfDeaths <- deaths[, c("Province/State", "Country/Region")]
dfRecovered <- recovered[, c("Province/State", "Country/Region")]
dfConfirmed <- confirmed[, c("Province/State", "Country/Region")]
# Extracts by country (by row) a matrix containing only the final column
mTotalDeaths <- deaths[, ncol(deaths)]
mTotalRecoveries <- recovered[, ncol(recovered)]
mTotalConfirmed <- confirmed[, ncol(confirmed)]
# Adds the sum vector to the corresponding Data Frames
# The Data Frames will now hold both the four country/location labels and the total counts
dfDeaths["Total Deaths"] <- mTotalDeaths
dfRecovered["Total Recovered"] <- mTotalRecoveries
dfConfirmed["Total Confirmed"] <- mTotalConfirmed
# Merges the Deaths and Recovered tables together based on matches in the first four columns
dfCombinedRisk <- merge(dfDeaths, dfRecovered, by=c("Province/State", "Country/Region") )
# Adds a new column for Risk Score, calculated by the Total Deaths and Total Recoveries
```

```r
dfCombinedRisk["Risk Score"] <- dfCombinedRisk$"Total Deaths" / dfCombinedRisk$"Total Recovered"
# Creates a new Data Frame with the totals
dfCombinedRiskTotals <- data.frame("Province/State"="TOTALS", "Country/Region"=NA, "Total Deaths"=sum(d
dfCombinedRiskTotals["Risk Score"] <- dfCombinedRiskTotals$Total.Deaths / dfCombinedRiskTotals$Total.Re
# Creates a new Data Frame excluding rows with risk score of 0
exclzero.dfCombinedRisk <- filter(dfCombinedRisk, dfCombinedRisk$`Risk Score` > 0)
```

```
## # A tibble: 6 x 5
##   `Province/State` `Country/Region` `Total Deaths` `Total Recovere~ `Risk Score`
##   <chr>            <chr>                     <dbl>            <dbl>        <dbl>
## 1 <NA>             Singapore                    28            57807     0.000484
## 2 Zhejiang         China                         1             1279     0.000782
## 3 Jiangxi          China                         1              934     0.00107
## 4 <NA>             Qatar                       224           126406     0.00177
## 5 <NA>             Burundi                       1              497     0.00201
## 6 Curacao          Netherlands                   1              433     0.00231
```

The area of the world that has the lowest risk score is Singapore, with a risk score of 0.0005. (Inf values are ignored.)

```
## # A tibble: 6 x 5
##   `Province/State` `Country/Region` `Total Deaths` `Total Recovere~ `Risk Score`
##   <chr>            <chr>                     <dbl>            <dbl>        <dbl>
## 1 <NA>             MS Zaandam                    2                0      Inf
## 2 <NA>             Netherlands                6751                0      Inf
## 3 <NA>             Serbia                      776                0      Inf
## 4 <NA>             Sweden                     5918                0      Inf
## 5 <NA>             United Kingdom            43646                0      Inf
## 6 <NA>             Belgium                   10413            21157        0.492
```

The area of the world that has the highest risk score is Belgium, with a score of 0.49.

Both of these individual countries' scores are vastly different compared to the global risk score of 0.05.

The values for certain countries seems unreasonable. For example, the UK has a risk score of over 1700, meaning that the total number of deaths is over 1,700 times the total number of recoveries. Were this number true, it would signify that COVID-19 is fatal to anyone unfortunate to contract it. However, this figure represents an outlier, likely due to low quality of data, given that the second highest risk score in the dataset is Belgium at 0.49.

On the other end of the spectrum, there are 27 countries with a risk score of 0, indicating that there were no deaths recorded in the country, but also that there was at least 1 recovery. (If both were 0, the value would return as undefined.) This may suggest that that particular country's death tally is not being tracked, or it could also mean that the country has done a splendid job at preventing deaths due to the virus.

While the risk score can be a helpful metric, it can also be unintentionally skewed higher or lower depending on the underlying number of deaths versus confirmed cases. For example, a country that is highly successful at preventing the spread of COVID-19 but sees infected patients die at a high instance (perhaps due to poor health services) may show a high risk score even though the magnitude of total deaths is in the hundreds of people. On the other hand, a country that cannot contain the virus, but has the medical services to keep infected patients alive may show a lower risk score, even if that particular country has millions of active cases. This risk score considers deaths to be of higher importance than confirmed cases, which may be a limitation, especially for practitioners who are seeking to make forward-looking estimates.

Additionally, the risk score uses the aggregate number of deaths and recoveries to date since January 22, 2020 as part of its calculation, regardless of the trends for a particular country. Countries that may have improving recent data may warrant a lower risk score, but the risk score does not adjust for any recent patterns in data.

Objective 4.2 In objective 4.2 we create tables with the top five countries that have the most COVID-19 related confirmations, recoveries, and deaths using the following code:

```
##### Deaths
# Creates a new Data Frame that is aggregated totals based on the County/Region
# Also sorts the Data Frame descending
dfDeathsGrouped <- aggregate(dfDeaths["Total Deaths"], by=dfDeaths["Country/Region"], sum)
dfDeathsGrouped <- dfDeathsGrouped[order(-dfDeathsGrouped$`Total Deaths`),]
# Prints the top 5 countries
head( as_tibble(dfDeathsGrouped), 5 )
```

```
## # A tibble: 5 x 2
##   `Country/Region` `Total Deaths`
##   <chr>                     <dbl>
## 1 US                       219674
## 2 Brazil                   153675
## 3 India                    114610
## 4 Mexico                    86167
## 5 United Kingdom            43736
```

As we can see from the table above, aggregating all regions as of 10/18/20, the greatest number of total deaths occurred in US, with a total of 220,000, followed by Brazil, India, Mexico, and United Kingdom.

```
##### Recoveries
# Creates a new Data Frame that is aggregated totals based on the County/Region
# Also sorts the Data Frame descending
dfRecoveredGrouped <- aggregate(dfRecovered["Total Recovered"], by=dfRecovered["Country/Region"], sum)
dfRecoveredGrouped <- dfRecoveredGrouped[order(-dfRecoveredGrouped$`Total Recovered`),]
# Prints the top 5 countries
head( as_tibble(dfRecoveredGrouped), 5 )
```

```
## # A tibble: 5 x 2
##   `Country/Region` `Total Recovered`
##   <chr>                        <dbl>
## 1 India                      6663608
## 2 Brazil                     4526393
## 3 US                         3234138
## 4 Russia                     1065608
## 5 Colombia                    858294
```

Next, in this table, aggregating all regions as of 10/18/20, the greatest number of total recoveries occurred in India, with a total of 6.7 million, followed by Brazil, US, Russia, and Colombia.

```
##### Confirmed
dfConfirmedGrouped <- aggregate(dfConfirmed["Total Confirmed"], by=dfConfirmed["Country/Region"], sum)
dfConfirmedGrouped <- dfConfirmedGrouped[order(-dfConfirmedGrouped$`Total Confirmed`),]
# Prints the top 5 countries
head( as_tibble(dfConfirmedGrouped), 5 )
```

```
## # A tibble: 5 x 2
##   `Country/Region` `Total Confirmed`
##   <chr>                        <dbl>
## 1 US                         8336031
## 2 India                      7651107
## 3 Brazil                     5298772
## 4 Russia                     1438219
## 5 Argentina                  1037325
```

Finally, in this last table, aggregating all regions as of 10/21/20, the greatest number of total confirmed cases occurred in US, with a total of 8.3 million, followed by India, Brazil, Russia, and Argentina.

GitHub Log

The GitHub project page can be viewed at:

https://github.com/mminniear1520/Group-Project-1-Alex-Mike

(Git code posted on the Project assignment page is not knitting properly according to the instructions.)

We were able to print the log from the command prompt but not from RMarkdown. Below is the output for the 'main' branch.

From branch 'main'

Subject: Fix bug with Confirmed data set calculation Author: alexanderyu-palomar Date: Sun, 25 Oct 2020 14:34:04 -0700 Body: Fixed a calculation bug with the Confirmed data set that was causing the incorrect values to be displayed.

Subject: Merge pull request #2 from mminniear1520/mike Author: alexanderyu-palomar Date: Sun, 25 Oct 2020 13:46:18 -0700 Body: Add objectives 2 and 3 to RMarkdown

Subject: Update file to main Author: alexanderyu-palomar Date: Sun, 25 Oct 2020 13:45:05 -0700 Body: Merging Obj 1-3 and Obj 4, prepped file for commit to main directly.

Subject: Change output to pdf_document Author: alexanderyu-palomar Date: Sun, 25 Oct 2020 13:06:59 -0700 Body:

From branch 'mike'

Subject: Updates to objectives 2 and 3 and text Author: Michael Minniear Date: Sun, 25 Oct 2020 18:02:43 -0700 Body:

Subject: Add objectives 2 and 3 to RMarkdown Author: Michael Minniear Date: Sun, 25 Oct 2020 10:31:05 -0700 Body:

Subject: Complete Obj 4.2 code Author: alexanderyu-palomar Date: Sat, 24 Oct 2020 17:57:33 -0700 Body: Objective 4.2 code completed. Tables are not appearing properly on my machine; may need some tweaking.

Subject: Complete Obj 4.1 Author: alexanderyu-palomar Date: Sat, 24 Oct 2020 01:37:53 -0700 Body: Answer to Obj 4.1 completed. File knits but tibbles not displaying properly.

Subject: Add partially completed answer to Objective 4.1 Author: alexanderyu-palomar Date: Fri, 23 Oct 2020 01:14:35 -0700 Body: Partial answer complete for Objective 4.1. Need to embed charts between the narrative sections.