Tidyverse for COVID19 Data Analysis

J. Wall

2020-04-28

# Get and clean worldwide Data

### Data from Johns Hopkins github

url\_in <- "https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse\_covid\_19\_data/csse\_covid\_19\_time\_series/"  
file\_names <- c("time\_series\_covid19\_confirmed\_global.csv",  
 "time\_series\_covid19\_deaths\_global.csv",  
 "time\_series\_covid19\_confirmed\_US.csv",  
 "time\_series\_covid19\_deaths\_US.csv")  
urls <- str\_c(url\_in,file\_names)

### Check urls

urls

## [1] "https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse\_covid\_19\_data/csse\_covid\_19\_time\_series/time\_series\_covid19\_confirmed\_global.csv"  
## [2] "https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse\_covid\_19\_data/csse\_covid\_19\_time\_series/time\_series\_covid19\_deaths\_global.csv"   
## [3] "https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse\_covid\_19\_data/csse\_covid\_19\_time\_series/time\_series\_covid19\_confirmed\_US.csv"   
## [4] "https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse\_covid\_19\_data/csse\_covid\_19\_time\_series/time\_series\_covid19\_deaths\_US.csv"

### Tidy global data

global\_confirmed <- read\_csv(urls[1]) %>%  
 pivot\_longer(cols = -c(`Province/State`, `Country/Region`, Lat, Long), names\_to = "Date",   
 values\_to = "Confirmed\_cases")  
global\_deaths <- read\_csv(urls[2]) %>%  
 pivot\_longer(cols = -c(`Province/State`, `Country/Region`, Lat, Long), names\_to = "Date",   
 values\_to = "Deaths")  
   
global <- global\_confirmed %>% full\_join(global\_deaths) %>%  
 rename(Country\_Region = `Country/Region`, Province\_State = `Province/State`) %>%  
 mutate(Date = mdy(Date))

### fix problems

knitr::kable(summary(global))

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Province\_State | Country\_Region | Lat | Long | Date | Confirmed\_cases | Deaths |
|  | Length:25608 | Length:25608 | Min. :-51.796 | Min. :-135.00 | Min. :2020-01-22 | Min. : 0 | Min. : 0 |
|  | Class :character | Class :character | 1st Qu.: 6.969 | 1st Qu.: -20.03 | 1st Qu.:2020-02-15 | 1st Qu.: 0 | 1st Qu.: 0 |
|  | Mode :character | Mode :character | Median : 23.488 | Median : 20.54 | Median :2020-03-10 | Median : 6 | Median : 0 |
|  | NA | NA | Mean : 21.317 | Mean : 22.17 | Mean :2020-03-10 | Mean : 2507 | Mean : 156 |
|  | NA | NA | 3rd Qu.: 41.166 | 3rd Qu.: 78.75 | 3rd Qu.:2020-04-03 | 3rd Qu.: 188 | 3rd Qu.: 2 |
|  | NA | NA | Max. : 71.707 | Max. : 178.06 | Max. :2020-04-27 | Max. :988197 | Max. :56259 |

# we find there are entries of -1 for Diamond Princess  
global <- global %>% filter(Confirmed\_cases >= 0, Deaths >= 0)

### Join population data to the dataset

uid\_lookup\_url <- "https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse\_covid\_19\_data/UID\_ISO\_FIPS\_LookUp\_Table.csv"  
uid <- read\_csv(uid\_lookup\_url) %>%  
 select(-c(Lat, Long\_, Combined\_Key, code3, iso2, iso3, Admin2))  
global <- global %>%   
 left\_join(uid, by = c("Province\_State", "Country\_Region")) %>%  
 select(-c(UID, FIPS)) %>%  
 select(Province\_State, Country\_Region, Date,  
 Confirmed\_cases, Deaths, Population,  
 Lat, Long)

### Get and tidy US data

US\_confirmed <- read\_csv(urls[3]) %>%  
 pivot\_longer(cols = -(UID:Combined\_Key), names\_to = "Date", values\_to = "Confirmed\_cases") %>%  
 select(Admin2:Confirmed\_cases) %>%  
 mutate(Date = mdy(Date))  
US\_deaths <- read\_csv(urls[4]) %>%  
 pivot\_longer(cols = -(UID:Population), names\_to = "Date", values\_to ="Deaths") %>%  
 select(Admin2:Deaths) %>%  
 mutate(Date = mdy(Date))

### Join deaths and cases

US <- US\_deaths %>%  
 full\_join(US\_confirmed,   
 by = c("Combined\_Key", "Date",   
 "Admin2", "Province\_State",   
 "Country\_Region")) %>%  
 rename(Long = Long\_.x, Lat = Lat.x) %>%  
 select(Admin2, Province\_State, Country\_Region,   
 Lat, Long, Population, Date, Confirmed\_cases, Deaths)

### US data so far

US %>% filter(Province\_State == "New York") %>%  
 select(Admin2, Province\_State, Confirmed\_cases, Deaths) %>%  
 head(n = 4)

## # A tibble: 4 x 4  
## Admin2 Province\_State Confirmed\_cases Deaths  
## <chr> <chr> <dbl> <dbl>  
## 1 Albany New York 0 0  
## 2 Albany New York 0 0  
## 3 Albany New York 0 0  
## 4 Albany New York 0 0

#Note that what we now have is county level data within each state. It would be nice to have data totaled for each state.

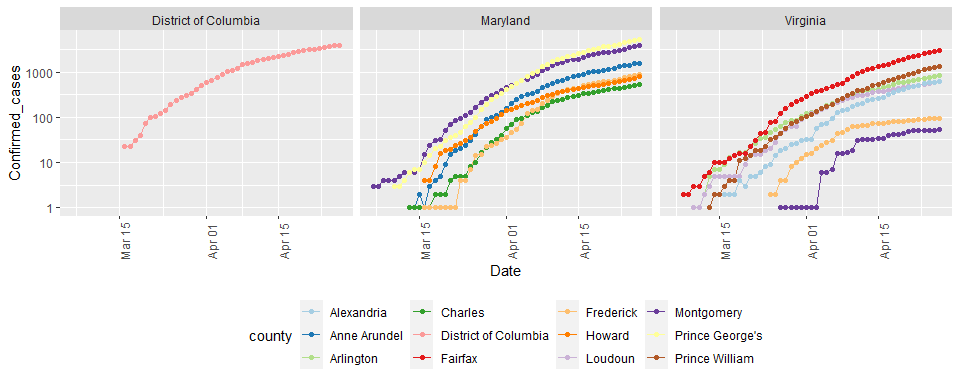
## DMV data

### get DMV data

dmv\_data <-  
 US %>%  
 mutate(state = factor(Province\_State),   
 county = factor(Admin2)) %>%  
 filter(state %in% c("Maryland", "Virginia", "District of Columbia")) %>%  
 filter(county %in%   
 c("Anne Arundel", "Montgomery", "Howard", "Frederick", "Prince George's",  
 "Charles", "District of Columbia", "Alexandria", "Arlington", "Fairfax",  
 "Loudoun", "Prince William")) %>%  
 select(-c(Admin2, Province\_State, Country\_Region)) %>%  
 mutate(Deaths\_per\_mill = 1000000 \* Deaths / Population) %>%  
 select(state, county, Date, Confirmed\_cases, Deaths,  
 Deaths\_per\_mill, Population, Lat, Long)

### Plot DMV cases by county

dmv\_data %>%  
 filter(Confirmed\_cases > 0) %>%  
 ggplot(aes(x = Date, y = Confirmed\_cases, group = county, col = county)) +  
 geom\_line() +  
 geom\_point() +  
 facet\_wrap(~ state) +  
 scale\_y\_log10() +  
 scale\_color\_brewer(palette = "Paired") +   
 guides(color = guide\_legend(ncol = 4)) +  
 theme(legend.position="bottom", axis.text.x = element\_text(angle = 90))



### Compute cases & deaths in DMV

DMV\_totals <- dmv\_data %>%   
 group\_by(Date) %>%  
 summarize(total\_cases = sum(Confirmed\_cases),   
 total\_deaths = sum(Deaths))  
  
paste0("total deaths = ", as.character(max(DMV\_totals$total\_deaths)) )

## [1] "total deaths = 953"

paste0("total cases = ", as.character(max(DMV\_totals$total\_cases) ))

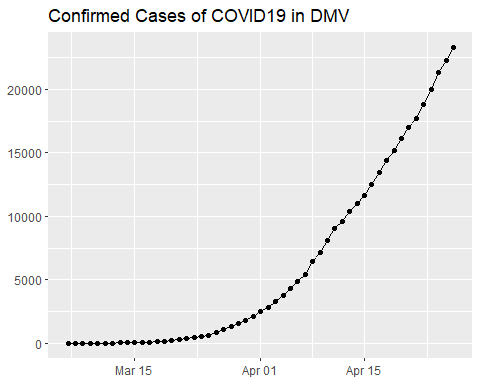
## [1] "total cases = 23309"

# Latest data from:  
paste0("latest data from: ", as.character( max(DMV\_totals$Date) ))

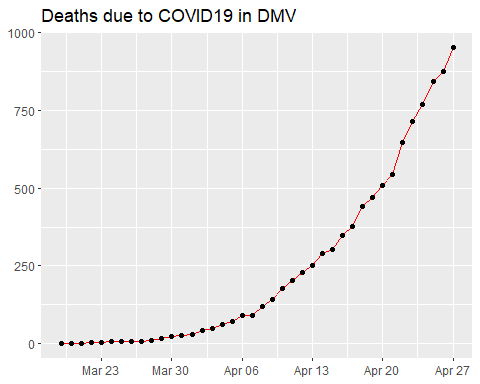
## [1] "latest data from: 2020-04-27"

### Visualize DMV cases and deaths

DMV\_totals %>%  
 filter(total\_cases > 0) %>%  
 ggplot(aes(x = Date, y = total\_cases)) +  
 geom\_line() +  
 geom\_point() +  
 labs(title = "Confirmed Cases of COVID19 in DMV", x = NULL, y = NULL)

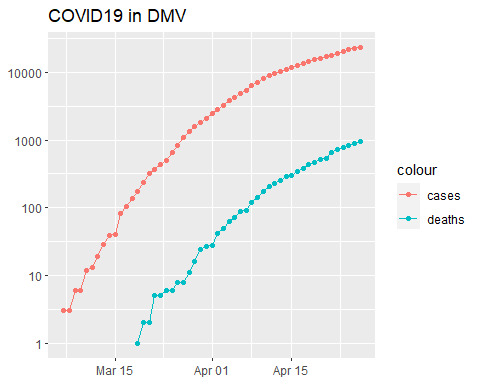


DMV\_totals %>%  
 filter(total\_deaths > 0) %>%  
 ggplot(aes(x = Date, y = total\_deaths)) +  
 geom\_line(color = "red") +  
 geom\_point() +  
 labs(title = "Deaths due to COVID19 in DMV", x = NULL, y = NULL)



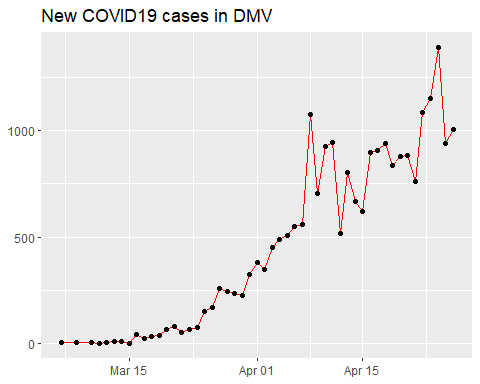
### both on one graph

DMV\_totals %>%  
 filter(total\_cases > 0) %>%  
 ggplot(aes(x = Date, y = total\_cases)) +  
 geom\_line(aes(color = "cases")) +  
 geom\_point(aes(color = "cases")) +  
 geom\_line(aes(y = total\_deaths, color = "deaths"),  
 data = DMV\_totals %>% filter(total\_deaths > 0)) +  
 geom\_point(aes(y = total\_deaths, color = "deaths"),  
 data = DMV\_totals %>% filter(total\_deaths > 0)) +  
 scale\_y\_log10() +   
 labs(title = "COVID19 in DMV", x = NULL, y = NULL)



### New cases

DMV\_totals <- DMV\_totals %>%  
 mutate(new\_cases = total\_cases - lag(total\_cases))   
DMV\_totals %>%  
 filter(new\_cases > 0) %>%  
 ggplot(aes(x = Date, y = new\_cases)) +  
 geom\_line(color = "red") +  
 geom\_point() +  
 labs(title = "New COVID19 cases in DMV", x = NULL, y = NULL)



## State analysis

### Compute state totals

US\_by\_state <- US %>%  
 group\_by(Province\_State, Country\_Region, Date) %>%  
 # add up counties and population  
 summarize(Confirmed\_cases = sum(Confirmed\_cases),   
 Deaths = sum(Deaths), Lat = median(Lat),   
 Long = median(Long), Population = sum(Population)) %>%  
 select(Province\_State, Country\_Region, Date,  
 Confirmed\_cases, Deaths, Population,  
 Lat, Long) %>%  
 ungroup()

### State data now

US\_by\_state %>% head(n = 3) %>%   
 select(Province\_State, Date, Confirmed\_cases,   
 Deaths, Population, Country\_Region)

## # A tibble: 3 x 6  
## Province\_State Date Confirmed\_cases Deaths Population Country\_Region  
## <chr> <date> <dbl> <dbl> <dbl> <chr>   
## 1 Alabama 2020-01-22 0 0 4903185 US   
## 2 Alabama 2020-01-23 0 0 4903185 US   
## 3 Alabama 2020-01-24 0 0 4903185 US

### Order deaths and cases by state

US\_state\_totals <- US\_by\_state %>%   
 group\_by(Province\_State) %>%   
 summarize(cases = max(Confirmed\_cases),  
 deaths = max(Deaths),  
 population = max(Population)) %>%  
 filter(cases > 0) %>%  
 mutate(deaths\_per\_mill = 1000000 \* deaths / population) %>%  
 arrange(desc(cases))

### View list by states

knitr::kable(US\_state\_totals %>% slice(1:10))

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Province\_State | cases | deaths | population | deaths\_per\_mill |
| New York | 291996 | 22668 | 23628065 | 959.36760 |
| New Jersey | 111188 | 6044 | 8882190 | 680.46281 |
| Massachusetts | 56462 | 3003 | 6892503 | 435.69078 |
| Illinois | 45883 | 1983 | 12671821 | 156.48895 |
| California | 44966 | 1779 | 39512223 | 45.02404 |
| Pennsylvania | 43558 | 1886 | 12801989 | 147.32086 |
| Michigan | 38210 | 3407 | 9986857 | 341.14837 |
| Florida | 32138 | 1088 | 21477737 | 50.65711 |
| Louisiana | 27068 | 1740 | 4648794 | 374.29062 |
| Connecticut | 25997 | 2012 | 3565287 | 564.33045 |

### Totals for US

# total deaths   
paste0("total US deaths = ", as.character(sum(US\_state\_totals$deaths)) )

## [1] "total US deaths = 56259"

# total cases  
paste0("total US cases = ", as.character(sum(US\_state\_totals$cases)) )

## [1] "total US cases = 988258"

# Latest data from:  
paste0("latest data from: ", as.character(max(US\_by\_state$Date)) )

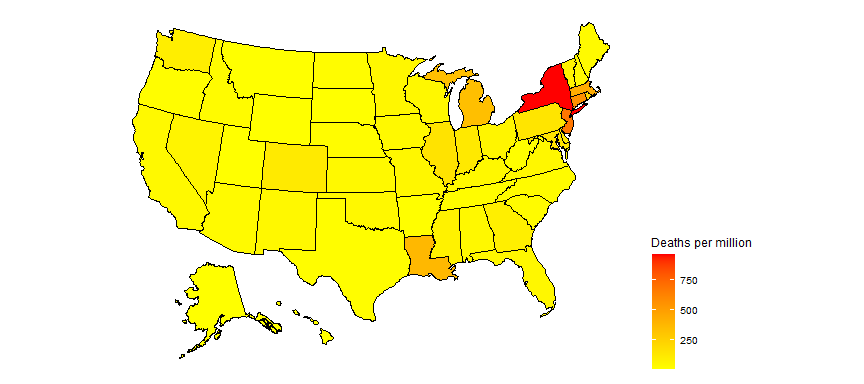
## [1] "latest data from: 2020-04-27"

## Visualizing the state data

library(usmap)  
US\_data <- US\_state\_totals %>%   
 mutate(state = Province\_State) %>%   
 filter(!is.na(deaths\_per\_mill))

### Plot states

plot\_usmap(data = US\_data,   
 values = "deaths\_per\_mill",  
 color = "black") +   
 scale\_fill\_gradient(name = "Deaths per million",  
 low = "yellow", high = "red") +  
 theme(legend.position = "right")

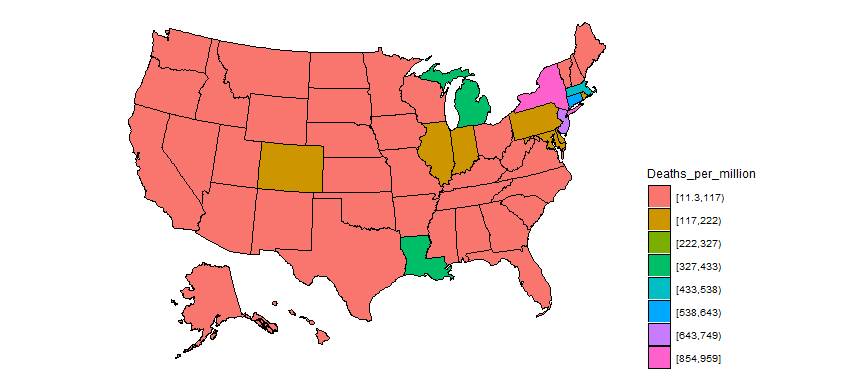


### partition deaths per million into 10 equal ranges.

US\_data <- US\_data %>%  
 filter(population > 0) %>%  
 mutate(death\_group = cut(deaths\_per\_mill,  
 breaks = seq(min(deaths\_per\_mill),  
 max(deaths\_per\_mill),  
 length.out = 10),   
 include.lowest = TRUE,  
 right = FALSE,   
 ordered\_result = TRUE) )

### Plot 10 levels

plot\_usmap(data = US\_data,   
 values = "death\_group",  
 color = "black") +   
 scale\_fill\_discrete(name = "Deaths\_per\_million") +  
 theme(legend.position = "right")



### Add states to global

#Replace the US observations in the global dataset with the US data  
exp\_global <- global %>%   
 # remove the US total data from the dataset  
 filter(Country\_Region != "US") %>%   
 # add on the totals by state  
 bind\_rows(US\_by\_state)

### Add continents

library(countrycode)  
temp <- countrycode(exp\_global$Country\_Region,  
 origin = "country.name",  
 destination = "continent")  
Confirmed <- exp\_global %>%  
 mutate(continent = temp) %>%  
 mutate(continent = case\_when(  
 Country\_Region == "Cruise Ship"~"Cruiseship",  
 Country\_Region == "Diamond Princess"~"Cruiseship",  
 Country\_Region == "MS Zaandam"~"Cruiseship",  
 Country\_Region == "Kosovo" ~ "Europe",  
 TRUE ~ continent)) %>%  
 # create a Country\_State combining Province\_State & Country\_Region  
 unite(Country\_State, c(Country\_Region, Province\_State),   
 na.rm = TRUE, remove = FALSE)  
Confirmed %>% filter(is.na(continent))

## # A tibble: 0 x 10  
## # ... with 10 variables: Country\_State <chr>, Province\_State <chr>,  
## # Country\_Region <chr>, Date <date>, Confirmed\_cases <dbl>, Deaths <dbl>,  
## # Population <dbl>, Lat <dbl>, Long <dbl>, continent <chr>

### Compute Deaths per million population

Confirmed <- Confirmed %>%  
 mutate(Deaths\_per\_mill = 1000000 \* Deaths / Population) %>%  
 select(Country\_State, Date, Confirmed\_cases,   
 Deaths, Deaths\_per\_mill, continent,   
 Population, Lat, Long, everything()) %>%  
 filter(Confirmed\_cases > 0) # leave off rows w/o cases

### Country/State’s w/ most cases

Confirmed\_totals <- Confirmed %>%   
 group\_by(Country\_State, Province\_State, Country\_Region, continent) %>%  
 summarize(Confirmed\_cases = max(Confirmed\_cases),   
 Deaths = max(Deaths),  
 Deaths\_per\_mill = max(Deaths\_per\_mill),  
 Date\_first\_case = min(Date),  
 Population = max(Population)) %>%  
 ungroup() %>%  
 select(Country\_State, Date\_first\_case, Deaths\_per\_mill,  
 Deaths, Confirmed\_cases, Population, everything())

### Worldwide totals to date

# total deaths   
paste0("total worldwide deaths = ", as.character(sum(Confirmed\_totals$Deaths)) )

## [1] "total worldwide deaths = 211169"

# total cases  
paste0("total worldwide cases = ", as.character(sum(Confirmed\_totals$Confirmed\_cases)) )

## [1] "total worldwide cases = 3041967"

# average deaths per million to date  
df <- Confirmed\_totals %>%  
 summarize(death\_rate = 1000000 \* sum(Deaths, na.rm = TRUE) / sum(Population, na.rm = TRUE))  
  
paste0("total worldwide deaths per million to date = ", as.character(sum(df$death\_rate)) )

## [1] "total worldwide deaths per million to date = 27.3897958320462"

# Latest data from:  
paste0("latest data from: ", as.character(max(US\_by\_state$Date)) )

## [1] "latest data from: 2020-04-27"

### Top 25 and top 100

Top\_25 <- Confirmed\_totals %>%   
 arrange(desc(Confirmed\_cases)) %>%  
 slice(1:25) %>%  
# Top\_25 %>%   
 select(Country\_State, continent,  
 Confirmed\_cases, Deaths,Deaths\_per\_mill)   
knitr::kable(Top\_25 %>% slice(1:8))

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Country\_State | continent | Confirmed\_cases | Deaths | Deaths\_per\_mill |
| US\_New York | Americas | 291996 | 22668 | 959.36760 |
| Spain | Europe | 229422 | 23521 | 503.07153 |
| Italy | Europe | 199414 | 26977 | 446.18234 |
| France | Europe | 164589 | 23293 | 356.85226 |
| Germany | Europe | 158758 | 6126 | 73.11663 |
| United Kingdom | Europe | 157149 | 21092 | 310.69733 |
| Turkey | Asia | 112261 | 2900 | 34.38501 |
| US\_New Jersey | Americas | 111188 | 6044 | 680.46281 |

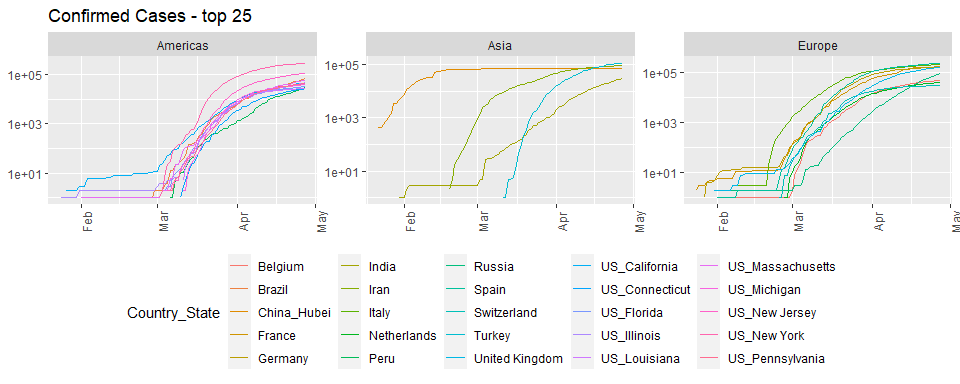
Top\_100 <- Confirmed\_totals %>%   
 arrange(desc(Confirmed\_cases)) %>%  
 slice(1:100)

### Get data for top 25

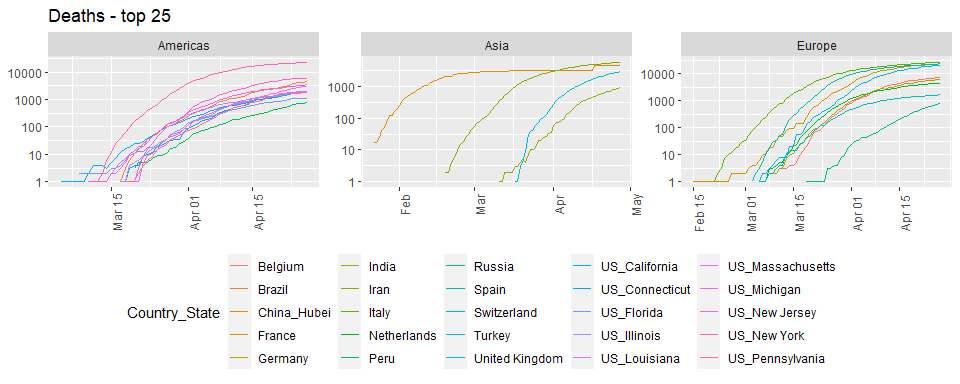
# grab top 25 country / states for graphing  
Top\_25\_states <- Top\_25$Country\_State  
Top\_25\_data <- Confirmed %>%   
 filter(Country\_State %in% Top\_25\_states) %>%  
 select(Country\_State, continent, Date, Confirmed\_cases,   
 Deaths, Deaths\_per\_mill)

### Graph top 25

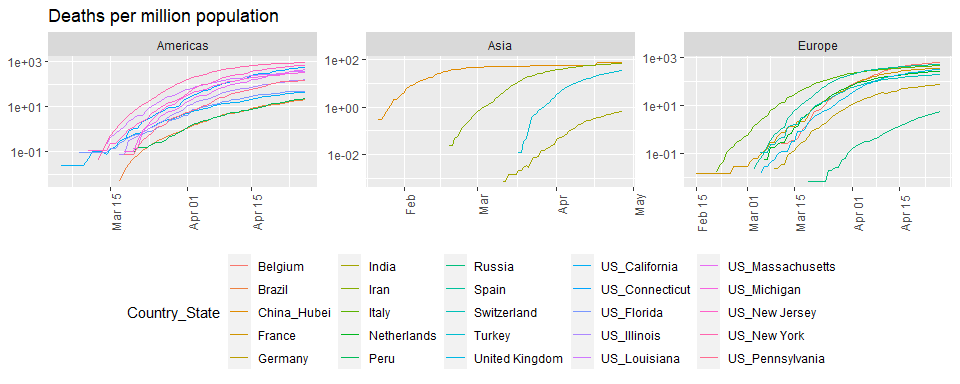
Top\_25\_data %>% filter(Confirmed\_cases > 0) %>%  
 ggplot(aes(x = Date, y = Confirmed\_cases,   
 group = Country\_State,   
 color = Country\_State)) +  
 geom\_line() +  
 facet\_wrap(~continent, scales = "free") +  
 scale\_y\_log10() +  
 labs(title = "Confirmed Cases - top 25", x = NULL, y = NULL) +  
 guides(color = guide\_legend(ncol = 6)) +  
 theme(legend.position="bottom",axis.text.x = element\_text(angle = 90))



Top\_25\_data %>% filter(Deaths > 0) %>%  
 ggplot(aes(x = Date, y = Deaths,   
 group = Country\_State,   
 color = Country\_State)) +  
 geom\_line() +  
 facet\_wrap(~continent, scales = "free") +  
 scale\_y\_log10() +  
 labs(title = "Deaths - top 25", x = NULL, y = NULL) +   
 guides(color = guide\_legend(ncol = 6)) +  
 theme(legend.position="bottom",axis.text.x = element\_text(angle = 90))

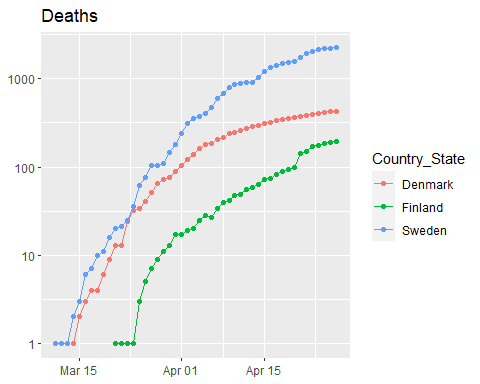


Top\_25\_data %>% filter(Deaths > 0) %>%  
 ggplot(aes(x = Date, y = Deaths\_per\_mill,   
 group = Country\_State,   
 color = Country\_State)) +  
 geom\_line() +  
 labs(title = "Deaths per million population", x = NULL,  
 y = NULL) +  
 facet\_wrap(~continent, scales = "free") +  
 scale\_y\_log10() +  
 guides(color = guide\_legend(ncol = 6)) +  
 theme(legend.position="bottom",axis.text.x = element\_text(angle = 90))

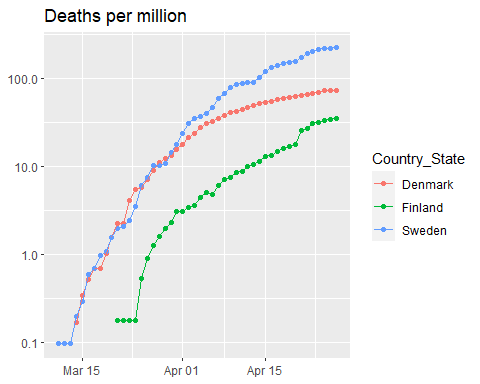


## Scandinavia analysis

# look at cases in Scandinavia since Sweden has not shut down their economy like other countries have. What impact has this had on death rates?  
Scandinavia <- Confirmed %>%   
 filter(Country\_State %in% c("Sweden", "Denmark", "Finland")) %>%  
 select(Country\_State, Date, Confirmed\_cases, Deaths, Deaths\_per\_mill, everything()) %>%  
 mutate(Country\_State = factor(Country\_State))   
  
Scandinavia %>%   
 filter(Deaths > 0) %>%  
 ggplot(aes(x = Date, y = Deaths, color = Country\_State)) +  
 geom\_point() + geom\_line() +  
 labs(title = "Deaths", x = NULL, y = NULL) +  
 scale\_y\_log10()



Scandinavia %>%   
 filter(Deaths > 0) %>%  
 ggplot(aes(x = Date, y = Deaths\_per\_mill, color = Country\_State)) +  
 geom\_point() + geom\_line() +  
 labs(title = "Deaths per million", x = NULL, y = NULL) +  
 scale\_y\_log10()



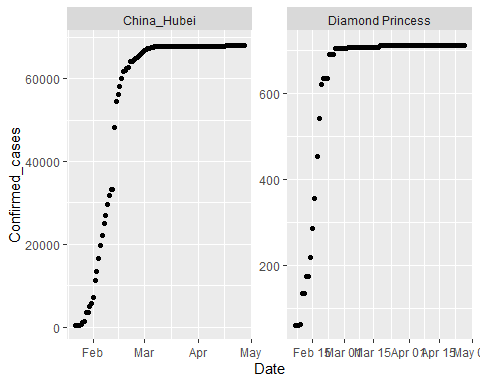
Scand\_summ <- Scandinavia %>% group\_by(Country\_State) %>%  
 summarize(Max\_Deaths\_per\_million =  
 max(Deaths\_per\_mill),  
 Total\_cases = max(Confirmed\_cases),  
 Total\_deaths = max(Deaths),  
 Population = max(Population))  
knitr::kable(Scand\_summ)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Country\_State | Max\_Deaths\_per\_million | Total\_cases | Total\_deaths | Population |
| Denmark | 73.71979 | 8698 | 427 | 5792203 |
| Finland | 34.83303 | 4695 | 193 | 5540718 |
| Sweden | 225.16479 | 18926 | 2274 | 10099270 |

# Modeling

### Hubei province of China and the Diamond Princess cruise ship.

Slowed\_cases <- Confirmed %>%   
 filter(Country\_State %in% c("China\_Hubei", "Diamond Princess"))   
Slowed\_cases %>%  
 ggplot(aes(x = Date, y = Confirmed\_cases)) +  
 geom\_point() +  
 facet\_wrap(~ Country\_State, scales = "free")

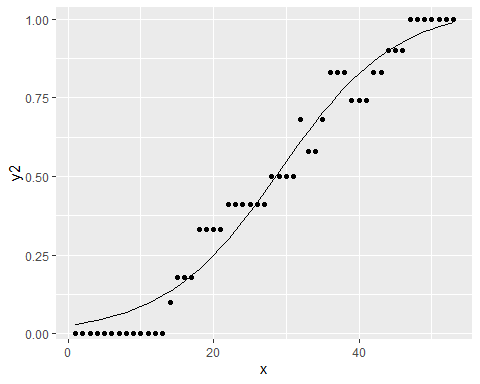


### Fitting a sigmoid function.

# thanks to http://kyrcha.info/2012/07/08/tutorials-fitting-a-sigmoid-function-in-r  
# function needed for visualization purposes  
sigmoid = function(x, params) {  
 params[1] / (1 + exp(-params[2] \* (x - params[3])))  
}  
  
x = 1:53  
y = c(0,0,0,0,0,0,0,0,0,0,0,0,0,0.1,0.18,0.18,0.18,0.33,0.33,0.33,0.33,0.41,0.41,0.41,0.41,0.41,0.41,0.5,0.5,0.5,0.5,0.68,0.58,0.58,0.68,0.83,0.83,0.83,0.74,0.74,0.74,0.83,0.83,0.9,0.9,0.9,1,1,1,1,1,1,1)  
df <- tibble(x = x, y = y)   
# fitting code  
fitmodel <- nls(y ~ a /(1 + exp(-b \* (x - c))), data = df,  
 start = list(a = 1, b = 0.5, c = 25))

### Plot model and data for sigmoid example

# visualization code  
# get the coefficients using the coef function  
params=coef(fitmodel)  
   
df$y2 <- sigmoid(x, params)  
df %>% ggplot(aes(x, y2)) + geom\_line() + geom\_point(y = y)



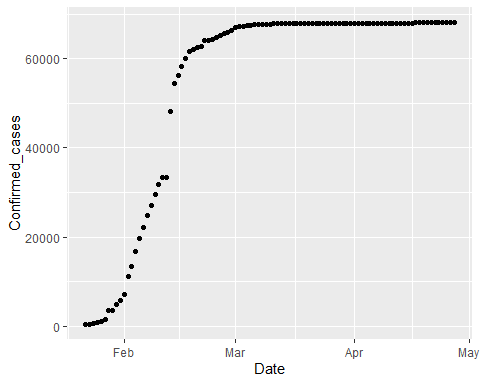
### More generalized sigmoid function

Goal: use that form for Hubei and fit a sigmoid function to that data.

### Start simply

### Get, visualize, model Hubei data

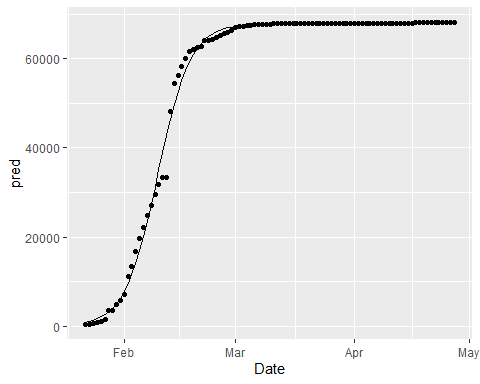
library(broom)  
Hubei\_cases <- Confirmed %>%   
 filter(Country\_State == "China\_Hubei") %>%  
 mutate(date\_int = unclass(Date))  
Hubei\_cases %>%  
 ggplot(aes(x = Date, y = Confirmed\_cases)) +  
 geom\_point()



# first with simplified sigmoid function  
sigmoid <- function(x, params) {  
 params[1] / (1 + exp(-params[2] \* (x - params[3])))  
}  
mod1 <- nls(Confirmed\_cases ~ K /(1 + exp(-B \* (date\_int - t0))),   
 data = Hubei\_cases,  
 start = list(K = 60000, B = 0.5, t0 = 18300))  
params <- coef(mod1)

### Plot Hubei model and data

Hubei\_cases <- Hubei\_cases %>%   
 mutate(pred = sigmoid(date\_int, params) ) %>%  
 select(Confirmed\_cases, pred, Date, date\_int, everything())  
  
Hubei\_cases %>%   
 ggplot(aes(Date, pred)) +   
 geom\_line() +   
 geom\_point(aes(y = Confirmed\_cases))



### Model summary

summary(mod1)

##   
## Formula: Confirmed\_cases ~ K/(1 + exp(-B \* (date\_int - t0)))  
##   
## Parameters:  
## Estimate Std. Error t value Pr(>|t|)   
## K 6.783e+04 2.018e+02 336.15 <2e-16 \*\*\*  
## B 2.341e-01 5.973e-03 39.19 <2e-16 \*\*\*  
## t0 1.830e+04 1.247e-01 146780.39 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 1590 on 94 degrees of freedom  
##   
## Number of iterations to convergence: 6   
## Achieved convergence tolerance: 6.859e-07

glance(mod1)

## # A tibble: 1 x 8  
## sigma isConv finTol logLik AIC BIC deviance df.residual  
## <dbl> <lgl> <dbl> <dbl> <dbl> <dbl> <dbl> <int>  
## 1 1590. TRUE 0.000000686 -851. 1710. 1721. 237712548. 94

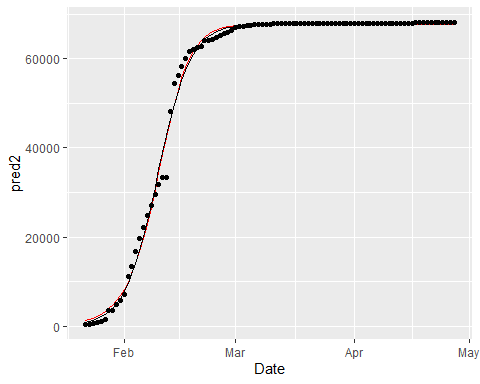
### One more step

### More complex model on Hubei

sigmoid\_gen <- function(x, params) {  
 params[1] /  
 ( (1 + exp(-params[2] \* (x - params[3]))) ) ^ (1 / params[4])  
}  
  
startK <- max(Hubei\_cases$Confirmed\_cases)  
mod2 <- nls(Confirmed\_cases ~ K /( (1 + exp(-B \* (date\_int - t0))) ) ^(1/v) ,   
 data = Hubei\_cases,  
 start = list(K = startK, B = .25, t0 = 18300, v = 1) )

### Visualize both Hubei models

# get the coefficients using the coef function  
params <- coef(mod2)  
   
Hubei\_cases <- Hubei\_cases %>%   
 mutate(pred2 = sigmoid\_gen(date\_int, params) ) %>%  
 select(Confirmed\_cases, pred2, pred, Date, date\_int, everything())  
  
Hubei\_cases %>%   
 ggplot(aes(x = Date)) +   
 geom\_line(aes(y = pred2), color = "red") +   
 geom\_line(aes(y = pred), color = "black") +  
 geom\_point(aes(y = Confirmed\_cases))



# use broom::glance to look at model results  
summary(mod2)

##   
## Formula: Confirmed\_cases ~ K/((1 + exp(-B \* (date\_int - t0))))^(1/v)  
##   
## Parameters:  
## Estimate Std. Error t value Pr(>|t|)   
## K 6.772e+04 2.032e+02 333.199 < 2e-16 \*\*\*  
## B 2.679e-01 1.911e-02 14.020 < 2e-16 \*\*\*  
## t0 1.830e+04 8.466e-01 21620.047 < 2e-16 \*\*\*  
## v 1.382e+00 2.089e-01 6.612 2.36e-09 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 1574 on 93 degrees of freedom  
##   
## Number of iterations to convergence: 11   
## Achieved convergence tolerance: 4.097e-06

glance(mod2)

## # A tibble: 1 x 8  
## sigma isConv finTol logLik AIC BIC deviance df.residual  
## <dbl> <lgl> <dbl> <dbl> <dbl> <dbl> <dbl> <int>  
## 1 1574. TRUE 0.00000410 -850. 1709. 1722. 230360130. 93

### Our plan of attack

* use a map function and nested grouped data to develop models out of this more complicated model family for the Diamond Princess and most of the China provinces
* Add the model to the nested data
* Use broom::tidy() to add the coefficients of the models to the dataset
* Use pivot\_wider to make these coefficients into columns.

### Nest China and Princess data

by\_country <- Confirmed %>%  
 filter(Country\_State != "China\_Beijing",  
 Country\_State != "China\_Shanxi",   
 Country\_State != "China\_Tibet") %>%  
 filter(Country\_Region == "China" | Country\_State == "Diamond Princess") %>%   
 #filter(Confirmed\_cases > 200) %>%  
 mutate(date\_int = unclass(Date)) %>%  
 group\_by(Country\_State, Province\_State, Country\_Region,   
 continent, Lat, Long) %>%  
 nest()  
# by\_country$data[[1]]  
by\_country

## # A tibble: 31 x 7  
## # Groups: Country\_State, continent, Lat, Long, Province\_State, Country\_Region  
## # [31]  
## Country\_State continent Lat Long Province\_State Country\_Region data   
## <chr> <chr> <dbl> <dbl> <chr> <chr> <list>   
## 1 China\_Anhui Asia 31.8 117. Anhui China <tibble ~  
## 2 China\_Chongqing Asia 30.1 108. Chongqing China <tibble ~  
## 3 China\_Fujian Asia 26.1 118. Fujian China <tibble ~  
## 4 China\_Gansu Asia 37.8 101. Gansu China <tibble ~  
## 5 China\_Guangdong Asia 23.3 113. Guangdong China <tibble ~  
## 6 China\_Guangxi Asia 23.8 109. Guangxi China <tibble ~  
## 7 China\_Guizhou Asia 26.8 107. Guizhou China <tibble ~  
## 8 China\_Hainan Asia 19.2 110. Hainan China <tibble ~  
## 9 China\_Hebei Asia 39.5 116. Hebei China <tibble ~  
## 10 China\_Heilongj~ Asia 47.9 128. Heilongjiang China <tibble ~  
## # ... with 21 more rows

### Add models to nested data

country\_mod2 <- function(df){  
 startK <- max(df$Confirmed\_cases)  
 nls(Confirmed\_cases ~ K /( (1 + exp(-B \* (date\_int - t0))) ) ^(1/v) ,   
 data = df,  
 start = list(K = startK, B = .25, t0 = 18300, v = 1),  
 control = list(maxiter = 1000, warnOnly = TRUE))  
}  
  
by\_country <- by\_country %>%  
 mutate(model = map(data, country\_mod2),  
 tm = map(model, broom::tidy)) %>%  
 unnest(tm) %>%  
 select(Country\_State, Lat, Long, term, estimate,  
 `std.error`, statistic, p.value, everything())

### Look at results

by\_country

## # A tibble: 124 x 13  
## # Groups: Country\_State, Lat, Long, continent, Province\_State, Country\_Region  
## # [31]  
## Country\_State Lat Long term estimate std.error statistic p.value  
## <chr> <dbl> <dbl> <chr> <dbl> <dbl> <dbl> <dbl>  
## 1 China\_Anhui 31.8 117. K 9.92e+2 0.808 1228. 1.47e-197  
## 2 China\_Anhui 31.8 117. B 2.74e-1 0.00545 50.2 3.90e- 69  
## 3 China\_Anhui 31.8 117. t0 1.83e+4 0.320 57223. 0.   
## 4 China\_Anhui 31.8 117. v 9.03e-1 0.0531 17.0 2.45e- 30  
## 5 China\_Chongq~ 30.1 108. K 5.79e+2 0.664 871. 1.02e-183  
## 6 China\_Chongq~ 30.1 108. B 1.81e-1 0.00407 44.5 1.84e- 64  
## 7 China\_Chongq~ 30.1 108. t0 1.83e+4 1.90 9634. 9.07e-281  
## 8 China\_Chongq~ 30.1 108. v 1.87e-1 0.0521 3.59 5.39e- 4  
## 9 China\_Fujian 26.1 118. K 3.39e+2 4.49 75.5 3.10e- 85  
## 10 China\_Fujian 26.1 118. B 1.16e-1 0.0218 5.31 7.41e- 7  
## # ... with 114 more rows, and 5 more variables: continent <chr>,  
## # Province\_State <chr>, Country\_Region <chr>, data <list>, model <list>

### Make data per province

province\_data <- by\_country %>%  
 ungroup() %>%  
 select(Province\_State, Lat, Long, term, estimate, data) %>%  
 pivot\_wider(names\_from = term, values\_from = estimate) %>%  
 mutate(t0\_date = as\_date(t0))  
province\_data %>% arrange(t0\_date) %>% print(n = Inf)

## # A tibble: 31 x 9  
## Province\_State Lat Long data K B t0 v t0\_date   
## <chr> <dbl> <dbl> <list> <dbl> <dbl> <dbl> <dbl> <date>   
## 1 Liaoning 41.3 123. <tibble~ 1.47e2 -0.0664 18126. -6.18 2019-08-17  
## 2 Shanghai 31.2 121. <tibble~ 7.26e2 0.0289 18253. 0.121 2019-12-23  
## 3 Gansu 37.8 101. <tibble~ 1.38e2 0.101 18256. 0.00901 2019-12-26  
## 4 Sichuan 30.6 103. <tibble~ 5.52e2 0.128 18259. 0.0108 2019-12-28  
## 5 Qinghai 35.7 96.0 <tibble~ 1.80e1 0.126 18261. 0.0249 2019-12-30  
## 6 Shaanxi 35.2 109. <tibble~ 2.55e2 0.134 18263. 0.0157 2020-01-01  
## 7 Yunnan 25.0 101. <tibble~ 1.80e2 0.116 18264. 0.0358 2020-01-02  
## 8 Hong Kong 22.3 114. <tibble~ 5.65e2 -0.576 18266. -5.56 2020-01-05  
## 9 Tianjin 39.3 117. <tibble~ 1.75e2 0.100 18267. 0.0285 2020-01-06  
## 10 Heilongjiang 47.9 128. <tibble~ 8.54e2 0.0397 18268. 0.113 2020-01-06  
## 11 Zhejiang 29.2 120. <tibble~ 1.25e3 0.139 18268. 0.0238 2020-01-06  
## 12 Fujian 26.1 118. <tibble~ 3.39e2 0.116 18269. 0.0432 2020-01-07  
## 13 Guangdong 23.3 113. <tibble~ 1.52e3 0.122 18269. 0.0304 2020-01-07  
## 14 Jilin 43.7 126. <tibble~ 1.02e2 0.164 18273. 0.0155 2020-01-11  
## 15 Macau 22.2 114. <tibble~ 2.92e1 -0.481 18273. -3.99 2020-01-12  
## 16 Shandong 36.3 118. <tibble~ 7.83e2 0.109 18274. 0.0818 2020-01-12  
## 17 Chongqing 30.1 108. <tibble~ 5.79e2 0.181 18283. 0.187 2020-01-21  
## 18 Guangxi 23.8 109. <tibble~ 2.54e2 0.183 18290. 0.471 2020-01-28  
## 19 Ningxia 37.3 106. <tibble~ 7.50e1 0.165 18290. 0.394 2020-01-29  
## 20 Hunan 27.6 112. <tibble~ 1.02e3 0.236 18291. 0.464 2020-01-29  
## 21 Jiangsu 33.0 119. <tibble~ 6.43e2 0.205 18291. 0.435 2020-01-30  
## 22 Henan 33.9 114. <tibble~ 1.28e3 0.237 18292. 0.526 2020-01-31  
## 23 Jiangxi 27.6 116. <tibble~ 9.38e2 0.261 18294. 0.646 2020-02-01  
## 24 Hebei 39.5 116. <tibble~ 3.23e2 0.193 18295. 0.611 2020-02-02  
## 25 Anhui 31.8 117. <tibble~ 9.92e2 0.274 18296. 0.903 2020-02-03  
## 26 Hainan 19.2 110. <tibble~ 1.68e2 0.296 18299. 1.74 2020-02-06  
## 27 Guizhou 26.8 107. <tibble~ 1.46e2 0.375 18300. 1.66 2020-02-08  
## 28 Xinjiang 41.1 85.2 <tibble~ 7.62e1 0.324 18303. 2.15 2020-02-10  
## 29 Hubei 31.0 112. <tibble~ 6.77e4 0.268 18304. 1.38 2020-02-11  
## 30 <NA> 0 0 <tibble~ 7.09e2 0.628 18310. 2.68 2020-02-18  
## 31 Inner Mongolia 44.1 114. <tibble~ 1.98e2 1.76 18375. 82.1 2020-04-23

### Join with province totals

pd <- province\_data %>% left\_join(Confirmed\_totals) %>%  
 filter(!is.na(Province\_State))  
pd %>% print(n = Inf, width = Inf)

## # A tibble: 30 x 17  
## Province\_State Lat Long data K B t0 v  
## <chr> <dbl> <dbl> <list> <dbl> <dbl> <dbl> <dbl>  
## 1 Anhui 31.8 117. <tibble [97 x 6]> 992. 0.274 18296. 0.903   
## 2 Chongqing 30.1 108. <tibble [97 x 6]> 579. 0.181 18283. 0.187   
## 3 Fujian 26.1 118. <tibble [97 x 6]> 339. 0.116 18269. 0.0432   
## 4 Gansu 37.8 101. <tibble [96 x 6]> 138. 0.101 18256. 0.00901  
## 5 Guangdong 23.3 113. <tibble [97 x 6]> 1516. 0.122 18269. 0.0304   
## 6 Guangxi 23.8 109. <tibble [97 x 6]> 254. 0.183 18290. 0.471   
## 7 Guizhou 26.8 107. <tibble [97 x 6]> 146. 0.375 18300. 1.66   
## 8 Hainan 19.2 110. <tibble [97 x 6]> 168. 0.296 18299. 1.74   
## 9 Hebei 39.5 116. <tibble [97 x 6]> 323. 0.193 18295. 0.611   
## 10 Heilongjiang 47.9 128. <tibble [96 x 6]> 854. 0.0397 18268. 0.113   
## 11 Henan 33.9 114. <tibble [97 x 6]> 1276. 0.237 18292. 0.526   
## 12 Hong Kong 22.3 114. <tibble [96 x 6]> 565. -0.576 18266. -5.56   
## 13 Hubei 31.0 112. <tibble [97 x 6]> 67716. 0.268 18304. 1.38   
## 14 Hunan 27.6 112. <tibble [97 x 6]> 1020. 0.236 18291. 0.464   
## 15 Inner Mongolia 44.1 114. <tibble [95 x 6]> 198. 1.76 18375. 82.1   
## 16 Jiangsu 33.0 119. <tibble [97 x 6]> 643. 0.205 18291. 0.435   
## 17 Jiangxi 27.6 116. <tibble [97 x 6]> 938. 0.261 18294. 0.646   
## 18 Jilin 43.7 126. <tibble [96 x 6]> 102. 0.164 18273. 0.0155   
## 19 Liaoning 41.3 123. <tibble [97 x 6]> 147. -0.0664 18126. -6.18   
## 20 Macau 22.2 114. <tibble [97 x 6]> 29.2 -0.481 18273. -3.99   
## 21 Ningxia 37.3 106. <tibble [97 x 6]> 75.0 0.165 18290. 0.394   
## 22 Qinghai 35.7 96.0 <tibble [94 x 6]> 18.0 0.126 18261. 0.0249   
## 23 Shaanxi 35.2 109. <tibble [96 x 6]> 255. 0.134 18263. 0.0157   
## 24 Shandong 36.3 118. <tibble [97 x 6]> 783. 0.109 18274. 0.0818   
## 25 Shanghai 31.2 121. <tibble [97 x 6]> 726. 0.0289 18253. 0.121   
## 26 Sichuan 30.6 103. <tibble [97 x 6]> 552. 0.128 18259. 0.0108   
## 27 Tianjin 39.3 117. <tibble [97 x 6]> 175. 0.100 18267. 0.0285   
## 28 Xinjiang 41.1 85.2 <tibble [96 x 6]> 76.2 0.324 18303. 2.15   
## 29 Yunnan 25.0 101. <tibble [97 x 6]> 180. 0.116 18264. 0.0358   
## 30 Zhejiang 29.2 120. <tibble [97 x 6]> 1255. 0.139 18268. 0.0238   
## t0\_date Country\_State Date\_first\_case Deaths\_per\_mill Deaths  
## <date> <chr> <date> <dbl> <dbl>  
## 1 2020-02-03 China\_Anhui 2020-01-22 0.0949 6  
## 2 2020-01-21 China\_Chongqing 2020-01-22 0.193 6  
## 3 2020-01-07 China\_Fujian 2020-01-22 0.0254 1  
## 4 2019-12-26 China\_Gansu 2020-01-23 0.0758 2  
## 5 2020-01-07 China\_Guangdong 2020-01-22 0.0705 8  
## 6 2020-01-28 China\_Guangxi 2020-01-22 0.0406 2  
## 7 2020-02-08 China\_Guizhou 2020-01-22 0.0556 2  
## 8 2020-02-06 China\_Hainan 2020-01-22 0.642 6  
## 9 2020-02-02 China\_Hebei 2020-01-22 0.0794 6  
## 10 2020-01-06 China\_Heilongjiang 2020-01-23 0.345 13  
## 11 2020-01-31 China\_Henan 2020-01-22 0.229 22  
## 12 2020-01-05 China\_Hong Kong 2020-01-23 0.534 4  
## 13 2020-02-11 China\_Hubei 2020-01-22 76.3 4512  
## 14 2020-01-29 China\_Hunan 2020-01-22 0.0580 4  
## 15 2020-04-23 China\_Inner Mongolia 2020-01-24 0.0395 1  
## 16 2020-01-30 China\_Jiangsu 2020-01-22 0 0  
## 17 2020-02-01 China\_Jiangxi 2020-01-22 0.0215 1  
## 18 2020-01-11 China\_Jilin 2020-01-23 0.0370 1  
## 19 2019-08-17 China\_Liaoning 2020-01-22 0.0459 2  
## 20 2020-01-12 China\_Macau 2020-01-22 0 0  
## 21 2020-01-29 China\_Ningxia 2020-01-22 0 0  
## 22 2019-12-30 China\_Qinghai 2020-01-25 0 0  
## 23 2020-01-01 China\_Shaanxi 2020-01-23 0.0776 3  
## 24 2020-01-12 China\_Shandong 2020-01-22 0.0697 7  
## 25 2019-12-23 China\_Shanghai 2020-01-22 0.289 7  
## 26 2019-12-28 China\_Sichuan 2020-01-22 0.0360 3  
## 27 2020-01-06 China\_Tianjin 2020-01-22 0.192 3  
## 28 2020-02-10 China\_Xinjiang 2020-01-23 0.121 3  
## 29 2020-01-02 China\_Yunnan 2020-01-22 0.0414 2  
## 30 2020-01-06 China\_Zhejiang 2020-01-22 0.0174 1  
## Confirmed\_cases Population Country\_Region continent  
## <dbl> <dbl> <chr> <chr>   
## 1 991 63240000 China Asia   
## 2 579 31020000 China Asia   
## 3 355 39410000 China Asia   
## 4 139 26370000 China Asia   
## 5 1587 113460000 China Asia   
## 6 254 49260000 China Asia   
## 7 147 36000000 China Asia   
## 8 168 9340000 China Asia   
## 9 328 75560000 China Asia   
## 10 939 37730000 China Asia   
## 11 1276 96050000 China Asia   
## 12 1037 7496988 China Asia   
## 13 68128 59170000 China Asia   
## 14 1019 68990000 China Asia   
## 15 199 25340000 China Asia   
## 16 653 80510000 China Asia   
## 17 937 46480000 China Asia   
## 18 110 27040000 China Asia   
## 19 146 43590000 China Asia   
## 20 45 649342 China Asia   
## 21 75 6880000 China Asia   
## 22 18 6030000 China Asia   
## 23 286 38640000 China Asia   
## 24 787 100470000 China Asia   
## 25 644 24240000 China Asia   
## 26 561 83410000 China Asia   
## 27 190 15600000 China Asia   
## 28 76 24870000 China Asia   
## 29 185 48300000 China Asia   
## 30 1268 57370000 China Asia