COVID19 assignment for STAT 413/613

J. Wall

2020-04-06

We will use this assignment to look at COVID-19 data

# First we will look at data in the US by county

The first portion of this work is thanks to Max Kuhn of RStudio.

theme\_set(theme\_bw())  
  
dat <-  
 read\_csv("https://raw.githubusercontent.com/nytimes/covid-19-data/master/us-counties.csv")  
deaths\_by\_state <- dat %>%  
 group\_by(state,date) %>%  
 summarize(total\_deaths = sum(deaths)) %>%  
 ungroup() %>%  
 filter(date == max(date)) %>%  
 arrange(desc(total\_deaths))  
  
# or  
  
deaths\_by\_state <- dat %>%  
 count(state,date, wt = deaths) %>%  
 filter(date == max(date)) %>%  
 rename(total\_deaths = n) %>%  
 arrange(desc(total\_deaths))  
  
deaths\_by\_state

## # A tibble: 55 x 3  
## state date total\_deaths  
## <chr> <date> <dbl>  
## 1 New York 2020-04-05 4161  
## 2 New Jersey 2020-04-05 917  
## 3 Michigan 2020-04-05 617  
## 4 Louisiana 2020-04-05 477  
## 5 California 2020-04-05 351  
## 6 Washington 2020-04-05 344  
## 7 Illinois 2020-04-05 283  
## 8 Massachusetts 2020-04-05 231  
## 9 Florida 2020-04-05 220  
## 10 Georgia 2020-04-05 219  
## # ... with 45 more rows

# total deaths   
sum(deaths\_by\_state$total\_deaths)

## [1] 9661

# Latest data from:  
max(dat$date)

## [1] "2020-04-05"

* **Exercise 1:** (1/2 point) Produce a tibble that has both deaths and total cases per state, arranged by the total number of cases in descending order

## # A tibble: 1,884 x 4  
## state date total\_deaths total\_cases  
## <chr> <date> <dbl> <dbl>  
## 1 New York 2020-04-05 4161 122911  
## 2 New York 2020-04-04 3568 114996  
## 3 New York 2020-04-03 2935 102945  
## 4 New York 2020-04-02 1958 92770  
## 5 New York 2020-04-01 1652 83890  
## 6 New York 2020-03-31 1282 75832  
## 7 New York 2020-03-30 1062 67216  
## 8 New York 2020-03-29 897 59568  
## 9 New York 2020-03-28 782 53364  
## 10 New York 2020-03-27 535 44636  
## # ... with 1,874 more rows

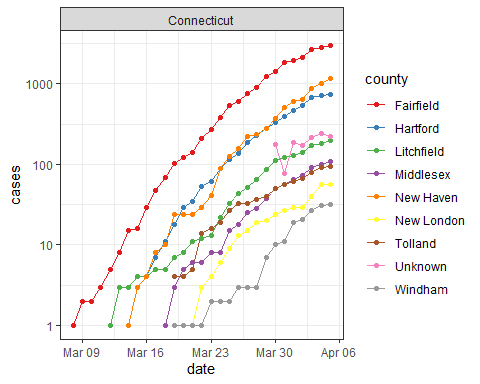
## [1] "total deaths = 9661"

## [1] "total cases = 336386"

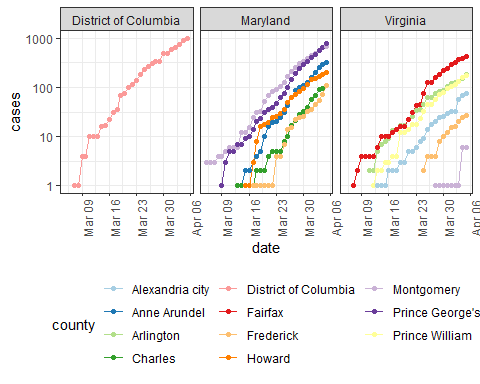
## [1] "latest data from: 2020-04-05"

Now we will choose just a few states to look at

dat\_small <-  
 dat %>%  
 filter(state %in% c("South Carolina", "Connecticut")) %>%  
 mutate(county = factor(county))  
  
dat\_small %>%  
 filter(state == "Connecticut") %>%  
 ggplot(aes(x = date, y = cases, group = county, col = county)) +  
 geom\_line() +  
 geom\_point() +  
 facet\_wrap(~ state) +  
 scale\_y\_log10() +  
 scale\_color\_brewer(palette = "Set1")



* **Exercise2:** (1 point) Do the analysis like above for the DMV area (DC, MD and VA) for both deaths and total cases. Use <http://www.theusgenweb.org/dcgenweb/geography/counties.shtml> for the counties in the DMV area. Be sure you get them all (hint: look at spelling in the dataset versus spelling you are entering)

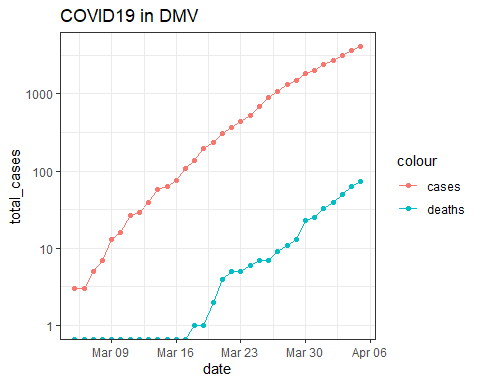
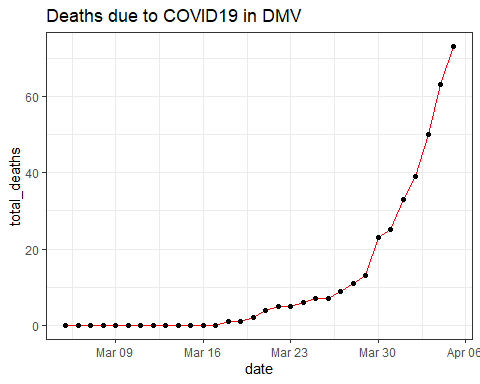
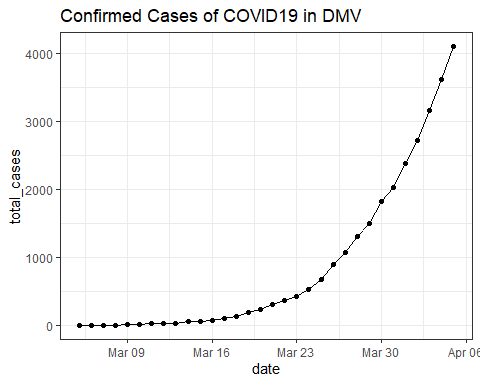


* **Exercise 3:** (1 point) Graph the total number of cases by date in the DMV area and the total deaths by date.

## [1] "total deaths = 73"

## [1] "total cases = 4106"

## [1] "latest data from: 2020-04-05"



# Get and clean worldwide Data

* **Exercise 4:** (1/2 point) Get data from (<https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/archived_data/archived_time_series/time_series_19-covid-Confirmed_archived_0325.csv>). After you have read in the dataset, create a new variable called Country/State that combines the country with the province/state and tidy the dataset.

## # A tibble: 6 x 7  
## `Country/State` `Province/State` `Country/Region` Lat Long Date   
## <chr> <chr> <chr> <dbl> <dbl> <date>   
## 1 Thailand <NA> Thailand 15 101 2020-01-22  
## 2 Thailand <NA> Thailand 15 101 2020-01-23  
## 3 Thailand <NA> Thailand 15 101 2020-01-24  
## 4 Thailand <NA> Thailand 15 101 2020-01-25  
## 5 Thailand <NA> Thailand 15 101 2020-01-26  
## 6 Thailand <NA> Thailand 15 101 2020-01-27  
## # ... with 1 more variable: Confirmed\_Cases <dbl>

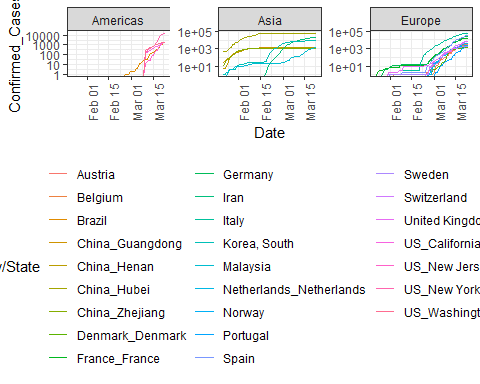
* **Exercise 5:** (1/2 point) Add a new variable called continent to the dataset. Be sure there are no NA’s for continent.

## # A tibble: 6 x 8  
## `Country/State` `Province/State` `Country/Region` Lat Long Date   
## <chr> <chr> <chr> <dbl> <dbl> <date>   
## 1 Thailand <NA> Thailand 15 101 2020-01-22  
## 2 Thailand <NA> Thailand 15 101 2020-01-23  
## 3 Thailand <NA> Thailand 15 101 2020-01-24  
## 4 Thailand <NA> Thailand 15 101 2020-01-25  
## 5 Thailand <NA> Thailand 15 101 2020-01-26  
## 6 Thailand <NA> Thailand 15 101 2020-01-27  
## # ... with 2 more variables: Confirmed\_Cases <dbl>, continent <chr>

* **Exercise 6:** (1/2 point) Find the 25 Country/State’s with the most Confirmed Cases (Hint: use group\_by, summarize and ungroup?)

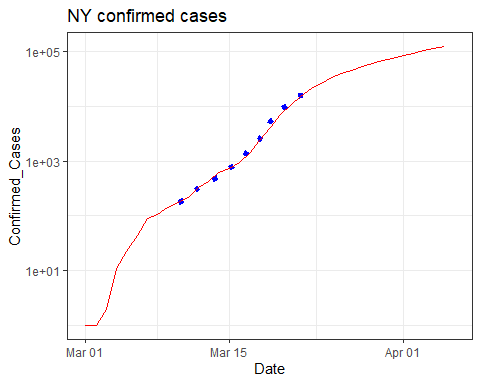
## # A tibble: 25 x 4  
## `Country/State` `Country/Region` continent ttl  
## <chr> <chr> <chr> <dbl>  
## 1 China\_Hubei China Asia 67800  
## 2 Italy Italy Europe 59138  
## 3 Spain Spain Europe 28768  
## 4 Germany Germany Europe 24873  
## 5 Iran Iran Asia 21638  
## 6 France\_France France Europe 16018  
## 7 US\_New York US Americas 15793  
## 8 Korea, South Korea, South Asia 8897  
## 9 Switzerland Switzerland Europe 7245  
## 10 United Kingdom\_United Kingdom United Kingdom Europe 5683  
## 11 Netherlands\_Netherlands Netherlands Europe 4204  
## 12 Belgium Belgium Europe 3401  
## 13 Austria Austria Europe 3244  
## 14 Norway Norway Europe 2383  
## 15 US\_Washington US Americas 1996  
## 16 Sweden Sweden Europe 1934  
## 17 US\_New Jersey US Americas 1914  
## 18 US\_California US Americas 1642  
## 19 Portugal Portugal Europe 1600  
## 20 Brazil Brazil Americas 1593  
## 21 China\_Guangdong China Asia 1413  
## 22 Denmark\_Denmark Denmark Europe 1395  
## 23 Malaysia Malaysia Asia 1306  
## 24 China\_Henan China Asia 1274  
## 25 China\_Zhejiang China Asia 1238

* **Exercise 7:** (1/2 point) Plot the number of cases over time for the top 25 country/states faceting by continent. Use appropriate scales for the axes.



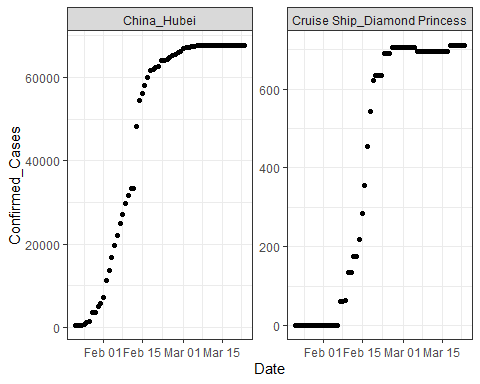
* **Exercise 8:** (1 point) Get the data for the state of NY from this csse dataset and compare the growth of cases from this dataset to the growth shown in the nytimes dataset used for exercises 1-3. Does the data match? If not, how is it different?

## # A tibble: 36 x 3  
## Date Confirmed\_Cases n  
## <date> <dbl> <dbl>  
## 1 2020-03-01 NA 1  
## 2 2020-03-02 NA 1  
## 3 2020-03-03 NA 2  
## 4 2020-03-04 NA 11  
## 5 2020-03-05 NA 22  
## 6 2020-03-06 NA 44  
## 7 2020-03-07 NA 89  
## 8 2020-03-08 NA 106  
## 9 2020-03-09 NA 142  
## 10 2020-03-10 173 173  
## 11 2020-03-11 220 217  
## 12 2020-03-12 328 326  
## 13 2020-03-13 421 421  
## 14 2020-03-14 525 610  
## 15 2020-03-15 732 732  
## 16 2020-03-16 967 950  
## 17 2020-03-17 1706 1374  
## 18 2020-03-18 2495 2382  
## 19 2020-03-19 5365 4152  
## 20 2020-03-20 8310 7102  
## 21 2020-03-21 11710 10356  
## 22 2020-03-22 15793 15168  
## 23 2020-03-23 15793 20875  
## 24 2020-03-24 NA 25666  
## 25 2020-03-25 NA 33067  
## 26 2020-03-26 NA 38988  
## 27 2020-03-27 NA 44636  
## 28 2020-03-28 NA 53364  
## 29 2020-03-29 NA 59568  
## 30 2020-03-30 NA 67216  
## 31 2020-03-31 NA 75832  
## 32 2020-04-01 NA 83890  
## 33 2020-04-02 NA 92770  
## 34 2020-04-03 NA 102945  
## 35 2020-04-04 NA 114996  
## 36 2020-04-05 NA 122911



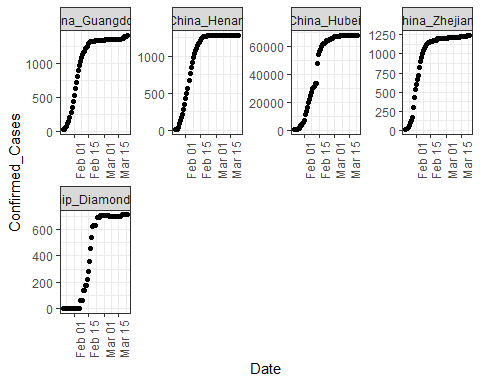
Let’s take a look at situations where the growth appears to have stopped. In particular, we start by visualizing the cases over time in the Hubei province of China and the Diamond Princess cruise ship.

Slowed\_cases <- Confirmed %>%   
 filter(`Country/State` %in% c("China\_Hubei", "Cruise Ship\_Diamond Princess"))   
Slowed\_cases %>%  
 ggplot(aes(x = Date, y = Confirmed\_Cases)) +  
 geom\_point() +  
 facet\_wrap(~ `Country/State`, scales = "free") #+



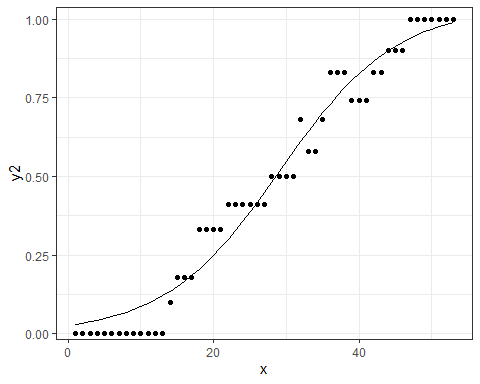
# scale\_y\_log10()

* **Exercise 9:** (1/2 point) Visualize cases from China that were in the top 25 country/states in terms of the number of cases and the Diamond Princess over time.



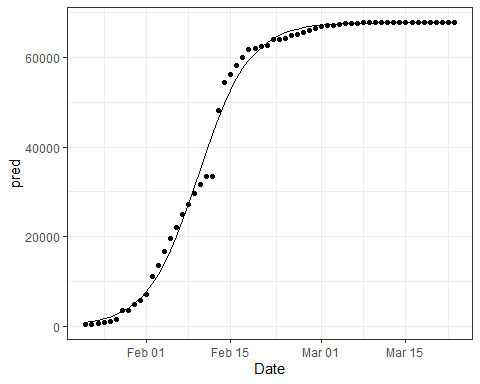
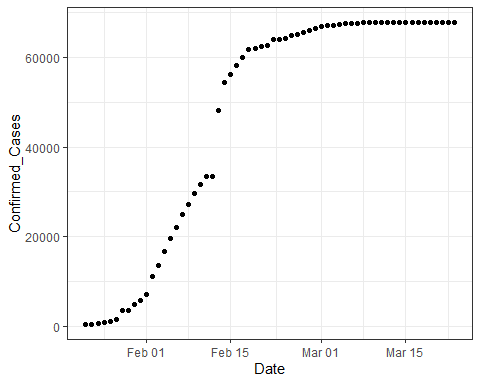
These graphs look like sigmoid functions. Here is a demo of fitting a sigmoid function.

# 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))  
   
# 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)



A more generalized sigmoid function is of the form

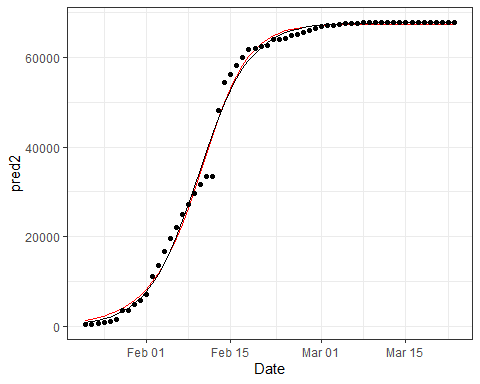
We will try using that form for the Hubei province and fitting a sigmoid function to that data. -**Exercise 10:** (1 point) Model the Hubei data with the simple form demonstrated above. Plot your results on top of the data points. Use broom::glance() and summary to look at model results.



##   
## Formula: Confirmed\_Cases ~ K/(1 + exp(-B \* (date\_int - t0)))  
##   
## Parameters:  
## Estimate Std. Error t value Pr(>|t|)   
## K 6.773e+04 3.839e+02 176.44 <2e-16 \*\*\*  
## B 2.348e-01 7.916e-03 29.66 <2e-16 \*\*\*  
## t0 1.830e+04 1.660e-01 110241.51 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 2002 on 59 degrees of freedom  
##   
## Number of iterations to convergence: 6   
## Achieved convergence tolerance: 2.132e-06

## # A tibble: 1 x 8  
## sigma isConv finTol logLik AIC BIC deviance df.residual  
## <dbl> <lgl> <dbl> <dbl> <dbl> <dbl> <dbl> <int>  
## 1 2002. TRUE 0.00000213 -558. 1123. 1132. 236424289. 59

-**Exercise 11:** (1 point) Now develop a second model and apply it to the same Hubei data. Graph your model results on top of the data and use summary and glance to look at your results. This model family will be



##   
## Formula: Confirmed\_Cases ~ K/((1 + exp(-B \* (date\_int - t0))))^(1/v)  
##   
## Parameters:  
## Estimate Std. Error t value Pr(>|t|)   
## K 6.743e+04 3.909e+02 172.490 < 2e-16 \*\*\*  
## B 2.799e-01 2.746e-02 10.196 1.50e-14 \*\*\*  
## t0 1.830e+04 1.018e+00 17983.849 < 2e-16 \*\*\*  
## v 1.490e+00 2.885e-01 5.164 3.09e-06 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 1976 on 58 degrees of freedom  
##   
## Number of iterations to convergence: 11   
## Achieved convergence tolerance: 9.751e-06

## # A tibble: 1 x 8  
## sigma isConv finTol logLik AIC BIC deviance df.residual  
## <dbl> <lgl> <dbl> <dbl> <dbl> <dbl> <dbl> <int>  
## 1 1976. TRUE 0.00000975 -556. 1123. 1133. 226420388. 58

* **Exercise 12:** (2 points) Now use a map function and nested grouped data to develop models out of this more complicated model family for the Diamond Princess and each of the China provinces except Beijing. Add the model to the nested data. Use broom::tidy() to add the coefficients of the models to your dataset. Use pivot\_wider to make these coefficients into columns.

## # A tibble: 5 x 7  
## # Groups: Country/State, Province/State, Country/Region, Lat, Long, continent  
## # [5]  
## `Country/State` `Province/State` `Country/Region` Lat Long continent data   
## <chr> <chr> <chr> <dbl> <dbl> <chr> <lis>  
## 1 China\_Hubei Hubei China 31.0 112. Asia <tib~  
## 2 China\_Guangdong Guangdong China 23.3 113. Asia <tib~  
## 3 China\_Henan Henan China 33.9 114. Asia <tib~  
## 4 China\_Zhejiang Zhejiang China 29.2 120. Asia <tib~  
## 5 Cruise Ship\_Dia~ Diamond Princess Cruise Ship 35.4 140. Asia <tib~

## # A tibble: 20 x 13  
## # Groups: Province/State, Lat, Long, Country/State, Country/Region, continent  
## # [5]  
## `Province/State` Lat Long term estimate std.error statistic p.value  
## <chr> <dbl> <dbl> <chr> <dbl> <dbl> <dbl> <dbl>  
## 1 Hubei 31.0 112. K 6.74e+4 391. 1.72e+2 2.52e- 80  
## 2 Hubei 31.0 112. B 2.80e-1 0.0275 1.02e+1 1.50e- 14  
## 3 Hubei 31.0 112. t0 1.83e+4 1.02 1.80e+4 2.37e-197  
## 4 Hubei 31.0 112. v 1.49e+0 0.288 5.16e+0 3.09e- 6  
## 5 Guangdong 23.3 113. K 1.36e+3 2.57 5.29e+2 1.51e-108  
## 6 Guangdong 23.3 113. B 2.11e-1 0.00600 3.52e+1 7.88e- 41  
## 7 Guangdong 23.3 113. t0 1.83e+4 1.38 1.32e+4 1.19e-189  
## 8 Guangdong 23.3 113. v 2.86e-1 0.0640 4.48e+0 3.59e- 5  
## 9 Henan 33.9 114. K 1.28e+3 1.71 7.45e+2 3.77e-117  
## 10 Henan 33.9 114. B 2.38e-1 0.00511 4.65e+1 1.28e- 47  
## 11 Henan 33.9 114. t0 1.83e+4 0.577 3.17e+4 1.25e-211  
## 12 Henan 33.9 114. v 5.26e-1 0.0511 1.03e+1 1.04e- 14  
## 13 Zhejiang 29.2 120. K 1.23e+3 41.0 3.00e+1 5.48e- 37  
## 14 Zhejiang 29.2 120. B 1.07e-1 0.0340 3.13e+0 2.72e- 3  
## 15 Zhejiang 29.2 120. t0 1.83e+4 249. 7.34e+1 6.67e- 59  
## 16 Zhejiang 29.2 120. v 2.62e-2 0.659 3.98e-2 9.68e- 1  
## 17 Diamond Princess 35.4 140. K 7.01e+2 2.84 2.47e+2 2.26e- 89  
## 18 Diamond Princess 35.4 140. B 6.32e-1 0.0700 9.03e+0 1.18e- 12  
## 19 Diamond Princess 35.4 140. t0 1.83e+4 0.361 5.07e+4 1.95e-223  
## 20 Diamond Princess 35.4 140. v 2.48e+0 0.407 6.09e+0 9.68e- 8  
## # ... with 5 more variables: `Country/State` <chr>, `Country/Region` <chr>,  
## # continent <chr>, data <list>, model <list>

## # A tibble: 5 x 8  
## `Province/State` Lat Long K B t0 v t0\_date   
## <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <date>   
## 1 Hubei 31.0 112. 67430. 0.280 18304. 1.49 2020-02-11  
## 2 Guangdong 23.3 113. 1362. 0.211 18287. 0.286 2020-01-26  
## 3 Henan 33.9 114. 1276. 0.238 18292. 0.526 2020-01-31  
## 4 Zhejiang 29.2 120. 1228. 0.107 18259. 0.0262 2019-12-29  
## 5 Diamond Princess 35.4 140. 701. 0.632 18310. 2.48 2020-02-18

Now you would be free to look at the t0\_date in comparison to when the first n cases appeared in that province or look at K in relation to the total population of the province. If you find a pattern there, perhaps see if it applies to those datasets that have not yet flattened out.

-**Extra Credit** (1 point) Use any of the COVID19 data to analyze anything of particular interest to you. Explain what you are doing and why and what did you find out.

I hope you had fun with this timely dataset!!