Reproducible Report on COVID-19 Data

N. A. Sackey

07/10/2021

Introduction

This report is concerned with analyzing the outbreak of COVID-19 especially with regard to Canada and its provinces. It is mainly focused on determining how well Canada fared in the face of the pandemic.

The datasets used are csv files hosted on the John Hopkins GitHub repository that record the number of cases of COVID-19 both for the US and globally from early 2020 to date and the number of deaths due to COVID-19 globally and for the US. They also record information associated with the cases and deaths such as the province, state, country, region or county as well as a variety of locational information.

Importing the Data

In order to use the COVID-19 dataset, it is necessary that the data is imported into Rstudio from the GitHub repository. Therefore, the first step is to organize the urls for the different .csv files that are needed into a vector.

```
# Create a vector of the urls for the .csv files
url1 <- "https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_
url2 <- "https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_
url3 <- "https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_
url4 <- "https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_
urls <- c(url1,url2,url3,url4)
```

After getting the urls all in one vector, the datasets are now imported and read into Rstudio.

```
# Read in the data
us_cases <- read_csv(urls[1])
```

```
## Rows: 3342 Columns: 640
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr   (6): iso2, iso3, Admin2, Province_State, Country_Region, Combined_Key
## dbl (634): UID, code3, FIPS, Lat, Long_, 1/22/20, 1/23/20, 1/24/20, 1/25/20,...
```

```
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
global_cases <- read_csv(urls[2])
```

```
## Rows: 279 Columns: 633

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr   (2): Province/State, Country/Region
## dbl (631): Lat, Long, 1/22/20, 1/23/20, 1/24/20, 1/25/20, 1/26/20, 1/27/20, ...

##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
us_deaths <- read_csv(urls[3])
```

```
## Rows: 3342 Columns: 641

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr   (6): iso2, iso3, Admin2, Province_State, Country_Region, Combined_Key
## dbl (635): UID, code3, FIPS, Lat, Long_, Population, 1/22/20, 1/23/20, 1/24/...

##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
global_deaths <- read_csv(urls[4])
```

```
## Rows: 279 Columns: 633

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr   (2): Province/State, Country/Region
## dbl (631): Lat, Long, 1/22/20, 1/23/20, 1/24/20, 1/25/20, 1/26/20, 1/27/20, ...

##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Tidying and Transforming the Data

The next step is to tidy and transform the data. After examining the dataset, it is clear that the 'Lat' and 'Long' columns are not needed for this analysis and so these 2 columns will be removed from all the datasets.

```r
# Remove Lat and Long columns
us_cases %>% select(-c(Lat, Long_)) -> us_cases
global_cases %>% select(-c(Lat, Long)) -> global_cases
us_deaths %>% select(-c(Lat, Long_)) -> us_deaths
global_deaths %>% select(-c(Lat, Long)) -> global_deaths
```

However for the US datasets, there are additional variables which are not required. These include the 'UID', 'iso2', 'iso3', 'code3' and 'FIPS' columns hence they will also be removed.

```
# Remove unneeded columns from the US data
us_cases %>% select(-c(UID, iso2, iso3, code3, FIPS)) -> us_cases
us_deaths %>% select(-c(UID, iso2, iso3, code3, FIPS)) -> us_deaths
```

For the data to be tidy, each observation must be a single row and each variable must be a single column. The various columns which have dates as the column headings all measure the same thing i.e. the number of cases or deaths on a particular day. Therefore, they can be combined and placed in their own column named 'date' such that each date is on a separate row and the values they recorded, which is either the number of cases or the number of deaths, are also be placed in a separate column named either 'cases' or 'deaths'.

```
# Combine all columns except Province and Country into one and create separate columns for cases
global_cases %>% pivot_longer(cols = -c(`Province/State`, `Country/Region`), names_to = "date", values_
global_deaths %>% pivot_longer(cols = -c(`Province/State`, `Country/Region`), names_to = "date", values
us_cases %>% pivot_longer(cols = -(Admin2:Combined_Key), names_to = "date", values_to = "cases") -> us_
us_deaths %>% pivot_longer(cols = -(Admin2:Population), names_to = "date", values_to = "deaths") -> us_
```

Next, the data for the global cases and global deaths will be joined together into a single dataset and the 'Province/State' and 'Country/Region' column headers in the global dataset will be renamed in order to ensure that they are valid variable names for R. Similarly, the US cases and US deaths data will be combined into one and the 'Admin2' column in the US dataset will be renamed to 'County' to make it clearer.

```
# Join the 2 datasets into 1
global_cases %>% full_join(global_deaths) %>% rename(Province_State = `Province/State`, Country_Region =
```

```
## Joining, by = c("Province/State", "Country/Region", "date")
```

```
us_cases %>% full_join(us_deaths) %>% rename(County = Admin2) -> us
```

```
## Joining, by = c("Admin2", "Province_State", "Country_Region", "Combined_Key", "date")
```

Lastly, the dates recorded in the 'date' column for both the US and the global dataset are of the character data type and so must be converted into a date data type.

```
# Convert dates from chr to date
us %>% mutate(date = mdy(date)) -> us
global %>% mutate(date = mdy(date)) -> global
```

After tidying the data it is now time to examine the data to check for any problems or errors by looking at a summary of the 2 datasets.

```
# Provide a summary of the data
summary(global)
```

```
##  Province_State     Country_Region          date                cases
##  Length:175491      Length:175491      Min.   :2020-01-22   Min.   :       0
##  Class :character   Class :character   1st Qu.:2020-06-27   1st Qu.:     152
##  Mode  :character   Mode  :character   Median :2020-12-01   Median :    2508
##                                        Mean   :2020-12-01   Mean   :  305655
##                                         3rd Qu.:2021-05-07   3rd Qu.:   57560
##                                         Max.   :2021-10-11   Max.   :44455949
##      deaths
##  Min.   :     0
##  1st Qu.:     1
##  Median :    40
##  Mean   :  6973
##  3rd Qu.:   951
##  Max.   :714055
```

```
summary(us)
```

```
##     County           Province_State     Country_Region     Combined_Key
##  Length:2102118     Length:2102118     Length:2102118     Length:2102118
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##       date                cases            Population          deaths
##  Min.   :2020-01-22   Min.   :      0   Min.   :       0   Min.   :    0.00
##  1st Qu.:2020-06-27   1st Qu.:     36   1st Qu.:    9917   1st Qu.:    0.00
##  Median :2020-12-01   Median :    663   Median :   24892   Median :   11.00
##  Mean   :2020-12-01   Mean   :   5267   Mean   :   99604   Mean   :   99.35
##  3rd Qu.:2021-05-07   3rd Qu.:   2876   3rd Qu.:   64979   3rd Qu.:   54.00
##  Max.   :2021-10-11   Max.   :1471645   Max.   :10039107   Max.   :26338.00
```

The summary shows the min date (earliest date) which is the 22nd of January for both datasets and the max and min for the cases and deaths as well as the population for the US dataset. From examining the data frames, it is clear that there are many observations early in the year of 2020 which do not record any cases. These are unneeded for this analysis and so it would be better to remove those and keep only the rows with at least one case.

```
# Remove rows which do not have at least 1 case
global %>% filter(cases > 0) -> global
us %>% filter(cases > 0) -> us
```

A summary of the global data shows that now the least number of cases is 1 for both datasets.

```
# Provide a summary of the data
summary(global)
```

```
##  Province_State     Country_Region          date                cases
##  Length:159434      Length:159434      Min.   :2020-01-22   Min.   :       1
##  Class :character   Class :character   1st Qu.:2020-07-29   1st Qu.:     382
##  Mode  :character   Mode  :character   Median :2020-12-25   Median :    4622
##                                        Mean   :2020-12-22   Mean   :  336438
##                                        3rd Qu.:2021-05-20   3rd Qu.:   75351
##                                        Max.   :2021-10-11   Max.   :44455949
##      deaths
##  Min.   :     0
##  1st Qu.:     3
##  Median :    69
##  Mean   :  7675
##  3rd Qu.:  1335
##  Max.   :714055
```

```
summary(us)
```

```
##     County          Province_State     Country_Region     Combined_Key
##  Length:1805533     Length:1805533     Length:1805533     Length:1805533
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##       date                cases           Population           deaths
##  Min.   :2020-01-22   Min.   :      1   Min.   :       0   Min.   :    0.0
##  1st Qu.:2020-08-20   1st Qu.:    186   1st Qu.:   11284   1st Qu.:    2.0
##  Median :2021-01-06   Median :   1038   Median :   26729   Median :   19.0
##  Mean   :2021-01-05   Mean   :   6132   Mean   :  106596   Mean   :  115.4
##  3rd Qu.:2021-05-25   3rd Qu.:   3593   3rd Qu.:   69872   3rd Qu.:   67.0
##  Max.   :2021-10-11   Max.   :1471645   Max.   :10039107   Max.   :26338.0
```

Next, it is time to ensure that the maximums shown in the summary are valid by checking for dates where the number of cases were greater than 40000000 for the global data and greater than 1000000 for the US data.

```
# Show only rows for cases greater that 40000000 and 1000000
global %>% filter(cases > 40000000)
```

```
## # A tibble: 37 x 5
##    Province_State Country_Region date          cases deaths
##    <chr>          <chr>          <date>        <dbl>  <dbl>
##  1 <NA>           US             2021-09-05 40039356 649175
##  2 <NA>           US             2021-09-06 40114538 649743
##  3 <NA>           US             2021-09-07 40391566 651886
##  4 <NA>           US             2021-09-08 40573789 654089
##  5 <NA>           US             2021-09-09 40733593 657363
##  6 <NA>           US             2021-09-10 40961390 659735
##  7 <NA>           US             2021-09-11 41025645 660495
##  8 <NA>           US             2021-09-12 41061504 660824
##  9 <NA>           US             2021-09-13 41322259 663034
## 10 <NA>           US             2021-09-14 41467287 664838
## # ... with 27 more rows
```

```
us %>% filter(cases > 1000000)
```

```
## # A tibble: 269 x 8
##    County      Province_State Country_Region Combined_Key      date         cases
##    <chr>       <chr>          <chr>          <chr>             <date>       <dbl>
##  1 Los Angeles California     US             Los Angeles, Cal~ 2021-01-16 1.00e6
##  2 Los Angeles California     US             Los Angeles, Cal~ 2021-01-17 1.01e6
##  3 Los Angeles California     US             Los Angeles, Cal~ 2021-01-18 1.02e6
##  4 Los Angeles California     US              Los Angeles, Cal~ 2021-01-19 1.03e6
##  5 Los Angeles California     US              Los Angeles, Cal~ 2021-01-20 1.04e6
##  6 Los Angeles California     US              Los Angeles, Cal~ 2021-01-21 1.05e6
##  7 Los Angeles California     US              Los Angeles, Cal~ 2021-01-22 1.05e6
##  8 Los Angeles California     US              Los Angeles, Cal~ 2021-01-23 1.07e6
##  9 Los Angeles California     US              Los Angeles, Cal~ 2021-01-24 1.07e6
## 10 Los Angeles California     US              Los Angeles, Cal~ 2021-01-25 1.08e6
## # ... with 259 more rows, and 2 more variables: Population <dbl>, deaths <dbl>
```

This shows that for the global data, the rows which are filtered are all the total number of cases in the US from the 5th of September, 2021 onwards and for the US data, they are the number of cases in the county of Los Angeles from the 16th of January, 2021. From this, it can be concluded that the maximum is valid.

Further examination of the two datasets shows that the US data contains the two variables 'Combined_Key' and 'Population' while the global data does not. However, in order to perform a comparative analysis, both variables will be required for the global data as well. Therefore, the next step is to create the 'Combined_Key' variable and add the 'Population' variable for the global dataset.

```
# Create a Combined_Key with rows containing State and Country entries attached together
global %>%  unite("Combined_Key", c(Province_State, Country_Region), sep = ", ", na.rm = TRUE, remove =
```

In order to obtain the 'Population' variable for the global dataset, another dataset containing the population for the different countries will have to be imported. This population data can be acquired from the same John Hopkins Github repository from an additional .csv file. Therefore, the earlier steps are repeated to import and read it in to Rstudio .

```
# Read dataset into Rstudio
url5 <- "https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/UID_ISO_FI
uid <- read_csv(url5)
```

```
## Rows: 4213 Columns: 12

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (7): iso2, iso3, FIPS, Admin2, Province_State, Country_Region, Combined_Key
## dbl (5): UID, code3, Lat, Long_, Population

##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Looking at the newly imported data frame, it can be seen that it contains the population for the countries in the global dataset which is what is required. However, it also contains the same unnecessary columns as the US dataset which were removed earlier when the US dataset was tidied. Therefore, the process is repeated for this new dataset and the unneeded columns are also removed.

```
# Remove unneeded columns
uid %>% select(-c(Lat, Long_, Combined_Key, code3, iso2, iso3, Admin2)) -> uid
```

Hence, only the 'UID', 'FIPS', 'Province_State', 'Country_Region' and 'Population' variables are left in the data frame. Now, all that remains is to join the uid and global data frames together into a single data frame that contains all the previous variables of the global dataset as well as the population from the uid data frame.

```
# Join uid and global for population and remove UID and FIPS after
global %>% left_join(uid, by = c("Province_State", "Country_Region")) %>% select(-c(UID, FIPS)) %>% sel
```

Finally, after joining the data frames, there is no longer any need for 'UID' and 'FIPS' in the global dataset and so they are removed as well.

Since the focus of this analysis is the cases of COVID-19 amongst the Canadian provinces, the global dataset is filtered in order to select only the COVID data about Canada and place it into a separate data frame.

```
# Select only the Canadian data
global %>% filter(Country_Region == "Canada") -> canada
```

From looking at the data frame, there are some rows for which the population data is missing and so there is a need to check exactly how many of these rows exist in the data frame and why.

```
# Select the rows with missing population data
canada %>% filter( is.na(Population))
```

```
## # A tibble: 932 x 7
##    Province_State   Country_Region date       cases deaths Population Combined_Key
##    <chr>            <chr>          <date>     <dbl>  <dbl>      <dbl> <chr>
##  1 Diamond Princess Canada         2020-05-01     1      1         NA Diamond Pri~
##  2 Diamond Princess Canada         2020-05-02     1      1         NA Diamond Pri~
##  3 Diamond Princess Canada         2020-05-03     1      1         NA Diamond Pri~
##  4 Diamond Princess Canada         2020-05-04     1      1         NA Diamond Pri~
##  5 Diamond Princess Canada         2020-05-05     1      1         NA Diamond Pri~
##  6 Diamond Princess Canada         2020-05-06     1      1         NA Diamond Pri~
##  7 Diamond Princess Canada         2020-05-07     1      1         NA Diamond Pri~
##  8 Diamond Princess Canada         2020-05-08     1      1         NA Diamond Pri~
##  9 Diamond Princess Canada         2020-05-09     1      1         NA Diamond Pri~
## 10 Diamond Princess Canada         2020-05-10     1      1         NA Diamond Pri~
## # ... with 922 more rows
```

This tibble shows that these rows refer to the cases of COVID-19 detected at locations other than the provinces of Canada such as the two cruise ships, the Diamond Princess and the Grand Princess, and therefore do not have any population data. Since this analysis is focused on the provinces of Canada, it would be better to simply remove these rows.

```
# Remove rows with NA
CAN_by_province <- na.omit(canada)
```

After ensuring that there are no more missing data, the next step is to find the total for Canada as a whole rather than according to province. Additionally. a new variable 'deaths_per_mil' will be created for the analysis.

```
# Reorganize Canada data to find totals and add new var
CAN_by_province %>% group_by(Country_Region, date) %>% summarize(cases = sum(cases), deaths = sum(death
  mutate(deaths_per_mil = deaths*1000000/Population) %>% select(Country_Region, date, cases, deaths, de
```

```
## `summarise()` has grouped output by 'Country_Region'. You can override using the `.groups` argument.
```

The end of the new ‘CAN_totals’ data frame is then examined to confirm the results of the data transformation.

```
# Display last few rows
tail(CAN_totals)
```

```
## # A tibble: 6 x 6
##   Country_Region date         cases deaths deaths_per_mil Population
##   <chr>          <date>       <dbl>  <dbl>          <dbl>      <dbl>
## 1 Canada         2021-10-06 1655380  28164           743.   37894482
## 2 Canada         2021-10-07 1659491  28195           744.   37894482
## 3 Canada         2021-10-08 1663690  28238           745.   37894482
## 4 Canada         2021-10-09 1665286  28245           745.   37894482
## 5 Canada         2021-10-10 1666882  28253           746.   37894482
## 6 Canada         2021-10-11 1667983  28263           746.   37894482
```

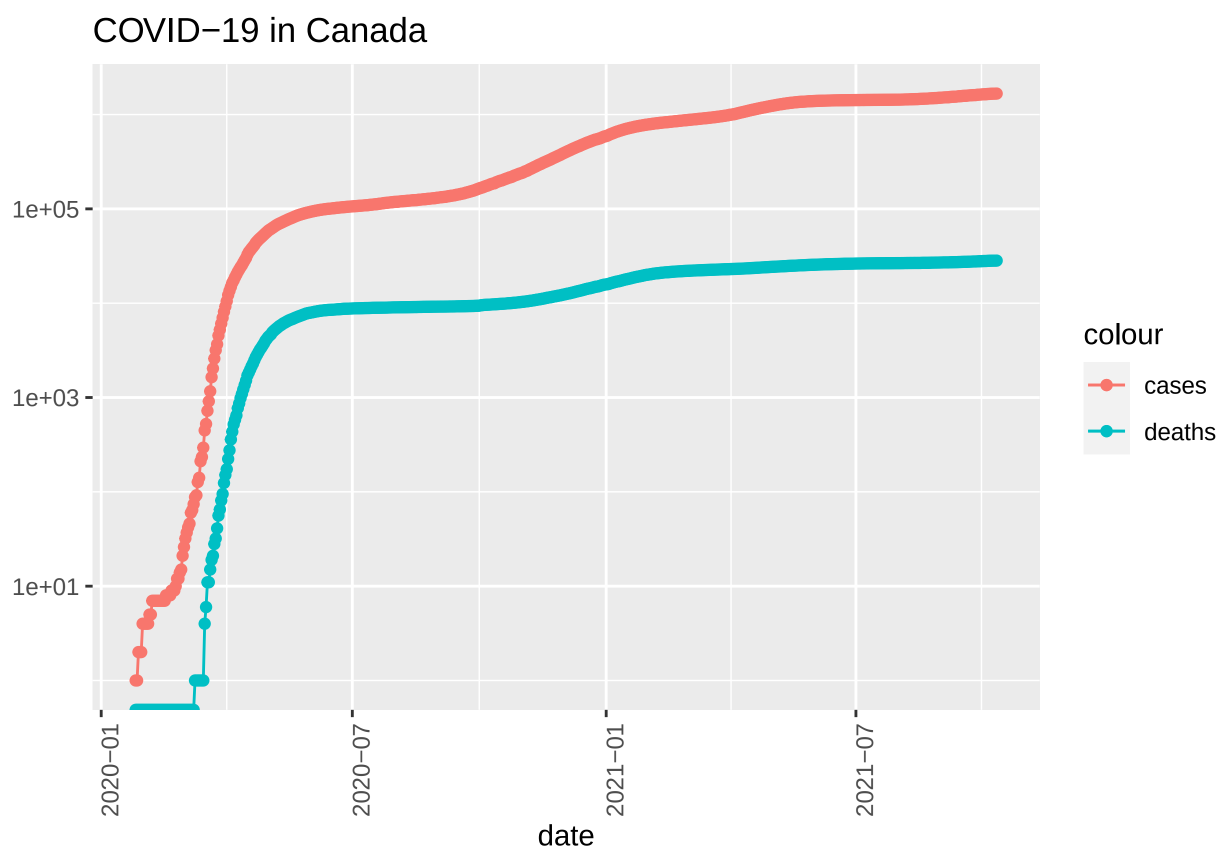
This shows that the number of cases and deaths are much higher at the tail end of the data as expected and also shows that ‘deaths_per_mil’ is around 745 for Canada as of the latest date.

Analyzing the Data

Now that the data is finally tidied and transformed, visualization and analysis can begin. In order to preserve details in the graph, the y variable will be scaled on a log scale. The first visualization will be to see how the number of cases compares to the number of deaths due to COVID-19 across the entirety of Canada and how this has changed with time.

```
# Plot cases and deaths for Canada as a whole
ggplot(CAN_totals, aes(x = date)) + geom_line(aes(y = cases, color = "cases")) + geom_point(aes(y = cas
```

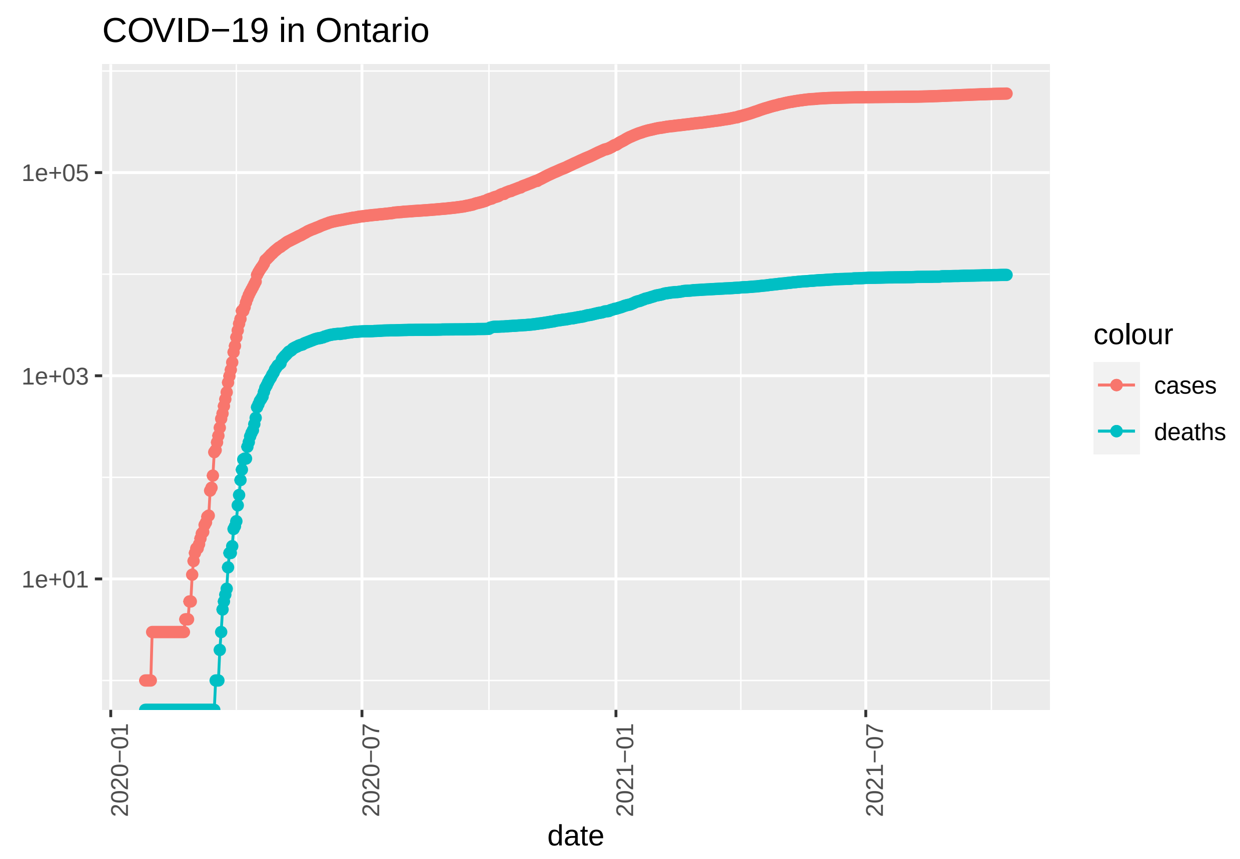
```
## Warning: Transformation introduced infinite values in continuous y-axis

## Warning: Transformation introduced infinite values in continuous y-axis
```



This shows the total number of cases and deaths in Canada from the start of the COVID-19 reports. The same kind of visualization can be performed for the individual provinces of Canada as well. For this visualization, the province of Ontario will be selected.

```
# Plot cases and deaths for Ontario
CAN_by_province %>% filter(Province_State == "Ontario") -> ontario
ggplot(ontario, aes(x = date)) + geom_line(aes(y = cases, color = "cases")) + geom_point(aes(y = cases,
```

```
## Warning: Transformation introduced infinite values in continuous y-axis

## Warning: Transformation introduced infinite values in continuous y-axis
```



From the 'CAN_totals' data, the maximum date (latest date) is the 11th of Oct.,2021 with the number of deaths on that date being 28263 which is the maximum of deaths for the entirety of Canada, however from the graph it appears that the number of COVID-19 cases have leveled off therefore raising questions about the visualization, specifically, whether the apparent leveling off is true and that there are very few new COVID-19 cases in Canada. Before any attempt at modeling, this question should be answered.

```
# Create a new variable with the new cases and new deaths
CAN_by_province %>% mutate(new_cases = cases - lag(cases), new_deaths = deaths - lag(deaths)) -> CAN_by_
CAN_totals %>% mutate(new_cases = cases - lag(cases), new_deaths = deaths - lag(deaths)) -> CAN_totals
```

In order to analyze this, there is a need to add some more variables to the data and therefore the data must be transformed yet again.

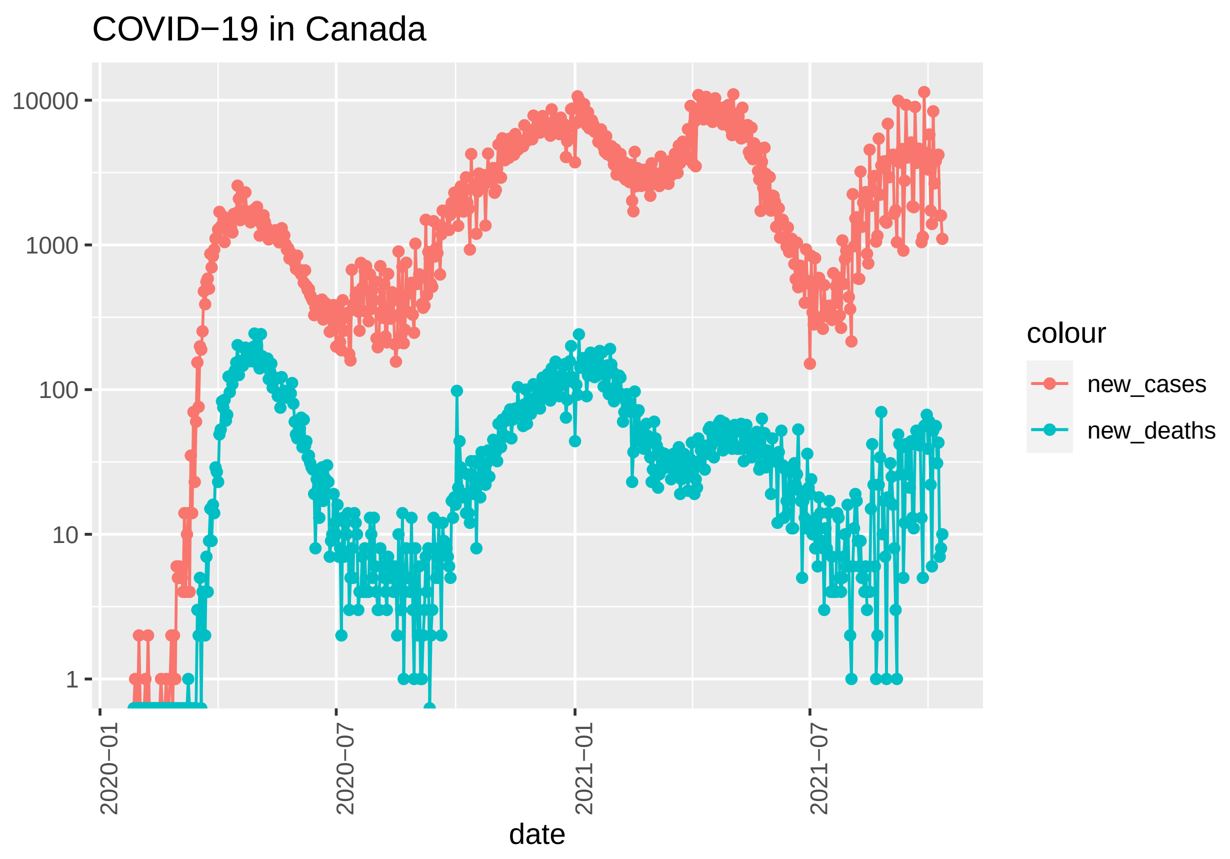
```
# Check the last few rows after adding new cases and new deaths
tail(CAN_totals %>% select(new_cases, new_deaths, everything()))
```

```
## # A tibble: 6 x 8
##   new_cases new_deaths Country_Region date          cases deaths deaths_per_mil
##       <dbl>      <dbl> <chr>          <date>        <dbl>  <dbl>          <dbl>
## 1      3803         56 Canada         2021-10-06 1655380  28164           743.
## 2      4111         31 Canada         2021-10-07 1659491  28195           744.
## 3      4199         43 Canada         2021-10-08 1663690  28238           745.
## 4      1596          7 Canada         2021-10-09 1665286  28245           745.
## 5      1596          8 Canada         2021-10-10 1666882  28253           746.
## 6      1101         10 Canada         2021-10-11 1667983  28263           746.
## # ... with 1 more variable: Population <dbl>
```

After confirming that the two new variables have been added to the data, another visualization can be performed.

```
# Plot new cases and deaths
ggplot(CAN_totals, aes(x = date)) + geom_line(aes(y = new_cases, color = "new_cases")) + geom_point(aes
```

```
## Warning: Transformation introduced infinite values in continuous y-axis

## Warning: Transformation introduced infinite values in continuous y-axis

## Warning in self$trans$transform(x): NaNs produced

## Warning: Transformation introduced infinite values in continuous y-axis

## Warning in self$trans$transform(x): NaNs produced

## Warning: Transformation introduced infinite values in continuous y-axis

## Warning: Removed 1 row(s) containing missing values (geom_path).

## Warning: Removed 1 rows containing missing values (geom_point).

## Warning: Removed 1 row(s) containing missing values (geom_path).

## Warning: Removed 2 rows containing missing values (geom_point).
```

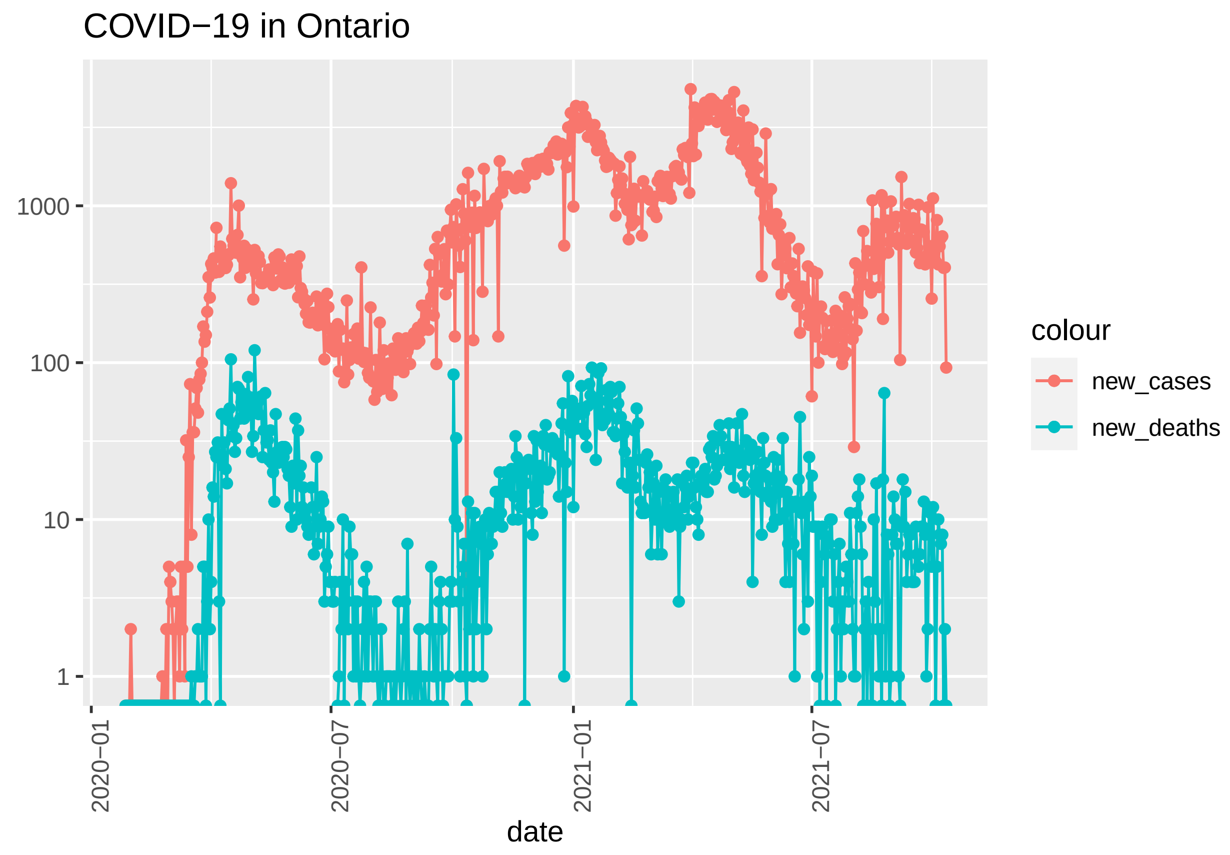


This closer inspection of the new cases and new deaths reveals that it appears to flatten out when approaching the end of the graph i.e. in the most recent days, the rate of increase appears to be declining. However, not too long before both the number of new cases and new deaths had dropped greatly but rose again sharply. It could be concluded that the drop may be due to the COVID-19 restrictions set in place earlier in the year while the rise might be related to the more recent relaxation of those same restrictions and the opening of public spaces such as schools and restaurants.

This same visualization can also be performed at the province level, specifically for Ontario.

```
# Plot new cases and new deaths for Ontario
CAN_by_province %>% filter(Province_State == "Ontario") -> ontario
ggplot(ontario, aes(x = date)) + geom_line(aes(y = new_cases, color = "new_cases")) + geom_point(aes(y
```

```
## Warning in self$trans$transform(x): NaNs produced

## Warning: Transformation introduced infinite values in continuous y-axis

## Warning in self$trans$transform(x): NaNs produced

## Warning: Transformation introduced infinite values in continuous y-axis

## Warning in self$trans$transform(x): NaNs produced

## Warning: Transformation introduced infinite values in continuous y-axis

## Warning in self$trans$transform(x): NaNs produced

## Warning: Transformation introduced infinite values in continuous y-axis

## Warning: Removed 1 row(s) containing missing values (geom_path).

## Warning: Removed 2 rows containing missing values (geom_point).

## Warning: Removed 1 row(s) containing missing values (geom_path).

## Warning: Removed 3 rows containing missing values (geom_point).
```



The visualization of the new cases and new deaths in Ontario is similar to the previous one for Canada. It shows a similar fall and rise in new cases and new deaths with what appears to be decrease in the latter days though it is still nowhere near as low as its earlier dip.

Following these basic visualizations, there may be some further questions that one may wish to answer such as which of the provinces is the best with regard to COVID-19 and which one is the worst? How can this be measured? Should the total cases be considered? Or the death rates per 1000 people? Hence, some more analysis would be required.

For further analysis, there is a need to transform the data once again. This involves the creation of new variables such as the number of cases per 1000 people and the number of deaths per 1000 people.

```
# Create new variables and calculate the cases and deaths per 1000
CAN_by_province %>% group_by(Province_State) %>% summarize(deaths = max(deaths), cases = max(cases), pop
```

Viewing the new data frame, it is clear that the province with the least deaths per thousand is Prince Edwards Island whiles Quebec has the highest deaths per thousand.

```
# Display the cases and deaths per thousand for each province
CAN_province_totals %>% slice_min(deaths_per_1k, n = 13) %>% select(cases_per_1k, deaths_per_1k, everyt
```

```
## # A tibble: 13 x 6
##    cases_per_1k deaths_per_1k Province_State            deaths  cases population
##           <dbl>         <dbl> <chr>                      <dbl>  <dbl>      <dbl>
##  1         1.91       0.00632 Prince Edward Island           1    302     158158
##  2         3.54       0.0211  Newfoundland and Labrador     11   1848     521365
##  3         6.74      0.0962  New Brunswick               75   5258     779993
##  4         7.08      0.100   Nova Scotia                 98   6918     977457
##  5        17.3       0.103   Nunavut                      4    671      38780
##  6        32.4       0.134   Northwest Territories        6   1456      44904
##  7        19.4       0.243   Yukon                       10    796      41078
##  8        37.7       0.392   British Columbia          2001 192491    5110917
##  9        61.3       0.636   Saskatchewan               751  72458    1181666
## 10        69.9       0.641   Alberta                   2830 308275    4413146
## 11        40.8       0.669   Ontario                   9840 599859   14711827
## 12        44.6       0.883   Manitoba                  1217  61385    1377517
## 13        48.8       1.34    Quebec                   11420 416266    8537674
```

The same process can also be performed to determine the best and worst cases per thousand.

```
# Display the cases and deaths per thousand for each province
CAN_province_totals %>% slice_min(cases_per_1k, n = 13) %>% select(cases_per_1k, deaths_per_1k, everythi
```

```
## # A tibble: 13 x 6
##    cases_per_1k deaths_per_1k Province_State            deaths  cases population
##           <dbl>         <dbl> <chr>                      <dbl>  <dbl>      <dbl>
##  1         1.91       0.00632 Prince Edward Island           1    302     158158
##  2         3.54       0.0211  Newfoundland and Labrador     11   1848     521365
##  3         6.74       0.0962  New Brunswick                 75   5258     779993
##  4         7.08       0.100   Nova Scotia                   98   6918     977457
##  5        17.3        0.103   Nunavut                        4    671      38780
##  6        19.4        0.243   Yukon                         10    796      41078
##  7        32.4        0.134   Northwest Territories          6   1456      44904
##  8        37.7        0.392   British Columbia            2001 192491    5110917
##  9        40.8        0.669   Ontario                     9840 599859   14711827
## 10        44.6        0.883   Manitoba                    1217  61385    1377517
## 11        48.8        1.34    Quebec                     11420 416266    8537674
## 12        61.3        0.636   Saskatchewan                 751  72458    1181666
## 13        69.9        0.641   Alberta                     2830 308275    4413146
```

This shows once again that Prince Edward Island fares the best with the least cases per thousand which makes sense considering that it had the lowest death rate. However, the province with the highest cases per thousand was Alberta and not Quebec which had the highest death rate.

Modeling

For modeling the data, the relationship between the number of cases per thousand and the number of deaths per thousand must be considered. The choice of model for this will be a linear model as a linear relationship is expected between the cases and the deaths.

```
# Use a linear model for deaths per 1000 and cases per 1000
mod <- lm(deaths_per_1k ~ cases_per_1k, data = CAN_province_totals)
summary(mod)
```

```
##
## Call:
## lm(formula = deaths_per_1k ~ cases_per_1k, data = CAN_province_totals)
```

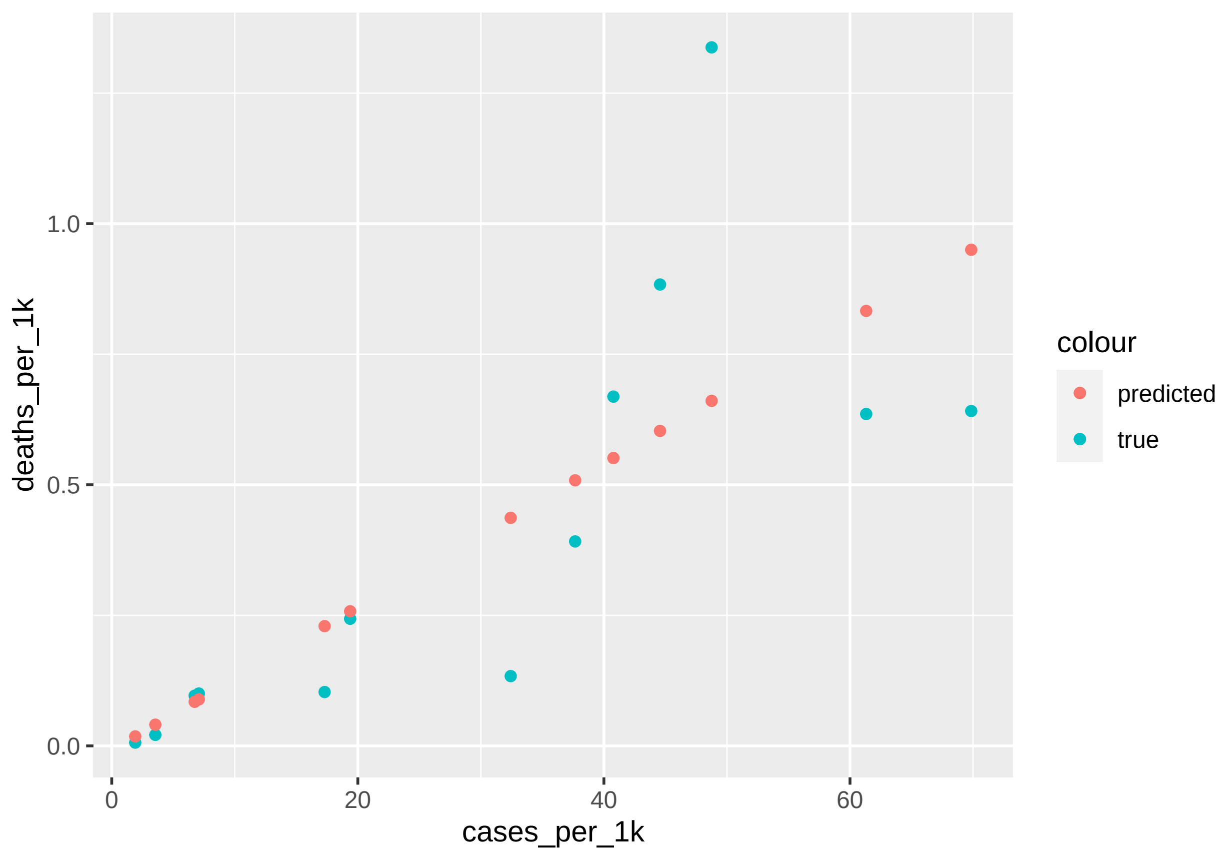
```
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.30874 -0.12613 -0.01429  0.01173  0.67694
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -0.008033   0.128343  -0.063   0.9512
## cases_per_1k  0.013715   0.003457   3.968   0.0022 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2709 on 11 degrees of freedom
## Multiple R-squared:  0.5887, Adjusted R-squared:  0.5513
## F-statistic: 15.74 on 1 and 11 DF,  p-value: 0.002205
```

The next step is to check the model. In order to do that, a new data frame containing the predictions of the model is created.

```
# Create new data frame with predictions
CAN_province_totals %>% mutate(pred = predict(mod)) -> CAN_tot_w_pred
```

Now, the predicted deaths per thousand and the actual deaths per thousand are plotted on the same graph in order to see how well the model does.

```
# Plot the predicted and true values
ggplot(CAN_tot_w_pred, aes(x = cases_per_1k)) + geom_point(aes(y = deaths_per_1k, color = "true")) + geo
```



The graph shows that the model does quite a good job of predicting the trend at the lower end but is less accurate from the middle to the higher end. Even though it can be concluded that the number of cases per thousand is an indicator of the number of deaths per thousand, the graph above raises questions such as what are the points have large residuals and why are they different compared to the other points that were modeled? These questions imply that there may be other factors and variables that should be considered and included as part of the model in order to improve prediction.

Bias Sources

The primary sources of bias identified is the choice of topic and data to analyze. The decision to work with and examine Canadian COVID-19 data was motivated by the fact that I am a resident of Canada and was quite interested in how COVID-19 had affected Canada. In the analysis, I also chose to consider Ontario over the other provinces since I live in Toronto.

Appendix

```
# Provide info about R session
sessionInfo()
```

```
## R version 4.1.1 (2021-08-10)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 20.04.3 LTS
##
```

```
## Matrix products: default
## BLAS:   /usr/lib/x86_64-linux-gnu/blas/libblas.so.3.9.0
## LAPACK: /usr/lib/x86_64-linux-gnu/lapack/liblapack.so.3.9.0
##
## locale:
##  [1] LC_CTYPE=en_CA.UTF-8       LC_NUMERIC=C
##  [3] LC_TIME=en_CA.UTF-8        LC_COLLATE=en_CA.UTF-8
##  [5] LC_MONETARY=en_CA.UTF-8    LC_MESSAGES=en_CA.UTF-8
##  [7] LC_PAPER=en_CA.UTF-8       LC_NAME=C
##  [9] LC_ADDRESS=C               LC_TELEPHONE=C
## [11] LC_MEASUREMENT=en_CA.UTF-8 LC_IDENTIFICATION=C
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] lubridate_1.8.0 forcats_0.5.1   stringr_1.4.0   dplyr_1.0.7
##  [5] purrr_0.3.4     readr_2.0.2     tidyr_1.1.4     tibble_3.1.5
##  [9] ggplot2_3.3.5   tidyverse_1.3.1
##
## loaded via a namespace (and not attached):
##  [1] Rcpp_1.0.7        assertthat_0.2.1 digest_0.6.28    utf8_1.2.2
##  [5] R6_2.5.1          cellranger_1.1.0 backports_1.2.1  reprex_2.0.1
##  [9] evaluate_0.14     httr_1.4.2       highr_0.9        pillar_1.6.3
## [13] rlang_0.4.11      curl_4.3.2       readxl_1.3.1     rstudioapi_0.13
## [17] rmarkdown_2.11    labeling_0.4.2   bit_4.0.4        munsell_0.5.0
## [21] broom_0.7.9       compiler_4.1.1   modelr_0.1.8     xfun_0.26
## [25] pkgconfig_2.0.3   htmltools_0.5.2  tidyselect_1.1.1 fansi_0.5.0
## [29] crayon_1.4.1      tzdb_0.1.2       dbplyr_2.1.1     withr_2.4.2
## [33] grid_4.1.1        jsonlite_1.7.2   gtable_0.3.0     lifecycle_1.0.1
## [37] DBI_1.1.1         magrittr_2.0.1   scales_1.1.1     cli_3.0.1
## [41] stringi_1.7.5     vroom_1.5.5      farver_2.1.0     fs_1.5.0
## [45] xml2_1.3.2        ellipsis_0.3.2   generics_0.1.0   vctrs_0.3.8
## [49] tools_4.1.1       bit64_4.0.5      glue_1.4.2       hms_1.1.1
## [53] parallel_4.1.1    fastmap_1.1.0    yaml_2.2.1       colorspace_2.0-2
## [57] rvest_1.0.1       knitr_1.36       haven_2.4.3
```