Pacote Visões do Coronavírus em R

Humberto Bezerra

08/12/2020

Bom dia!

Basicamente trata-se de exploração de dados do pacote coronavirus disponível no R, criado por Rami Krispin, autor do livro "Hands-On Time Series Analysis with R" .

https://github.com/RamiKrispin/coronavirus

https://github.com/RamiKrispin/coronavirus_dashboard

https://ramikrispin.github.io/

Primeira Etapa, instalação do pacote através do Github.

Outra opção: instalar através do CRAN e como a base é atualizada mensalmente, utilize o comando update_dataset() para obtenção de uma atualização.

```
##
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

## Updates are available on the coronavirus Dev version, do you want to update? n/Y
```

Para ver o conjunto de dados:

```
data(coronavirus)
head(coronavirus)
```

```
##         date province     country      lat     long      type cases
## 1 2020-01-22          Afghanistan 33.93911 67.70995 confirmed     0
## 2 2020-01-23          Afghanistan 33.93911 67.70995 confirmed     0
```

```
## 3 2020-01-24          Afghanistan 33.93911 67.70995 confirmed     0
## 4 2020-01-25          Afghanistan 33.93911 67.70995 confirmed     0
## 5 2020-01-26          Afghanistan 33.93911 67.70995 confirmed     0
## 6 2020-01-27          Afghanistan 33.93911 67.70995 confirmed     0
```

Exploração do conjunto de dados para ver as informações disponíveis. Note que há dados de latitude e longitude também.

Uma boa forma de vislumbre dos dados utilizando a função Glimpse:

```
glimpse(coronavirus)
```

```
## Rows: 255,360
## Columns: 7
## $ date     <date> 2020-01-22, 2020-01-23, 2020-01-24, 2020-01-25, 2020-01-2...
## $ province <chr> "", "", "", "", "", "", "", "", "", "", "", "", "", "", ""...
## $ country  <chr> "Afghanistan", "Afghanistan", "Afghanistan", "Afghanistan"...
## $ lat      <dbl> 33.93911, 33.93911, 33.93911, 33.93911, 33.93911, 33.93911...
## $ long     <dbl> 67.70995, 67.70995, 67.70995, 67.70995, 67.70995, 67.70995...
## $ type     <chr> "confirmed", "confirmed", "confirmed", "confirmed", "confi...
## $ cases    <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
```

Se usar o pacote dplyr para fazer um resumo dos dados de total confirmados por país. Pode-se concentrar nos top 20 (usando head(20)). Certifique-se de que você sabe o que faz cada comando abaixo. Note que estamos usando o operador pipe para facilitar.

```
library(dplyr)

summary_df <- coronavirus %>%
  filter(type == "confirmed") %>%
  group_by(country) %>%
  summarise(total_cases = sum(cases)) %>%
  arrange(-total_cases)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
summary_df %>% head(20)
```

```
## # A tibble: 20 x 2
##    country        total_cases
##    <chr>                <int>
##  1 US                14757000
##  2 India              9677203
##  3 Brazil             6603540
##  4 Russia             2439163
##  5 France             2345648
##  6 Italy              1728878
##  7 United Kingdom     1727751
##  8 Spain              1684647
##  9 Argentina          1463110
## 10 Colombia           1371103
## 11 Germany            1194550
## 12 Mexico             1175850
## 13 Poland             1063449
## 14 Iran               1040547
## 15 Peru                972688
## 16 Ukraine             834913
## 17 Turkey              828295
```

```
## 18 South Africa        814565
## 19 Belgium             591756
## 20 Indonesia           575796
```

Agora, vamos ver os novos casos durante as últimas 24 horas por país e por tipo.

```
coronavirus %>%
  filter(date == max(date)) %>%
  select(country, type, cases) %>%
  group_by(country, type) %>%
  summarise(total_cases = sum(cases))%>%
  head(20)
```

```
## `summarise()` regrouping output by 'country' (override with `.groups` argument)
```

```
## # A tibble: 20 x 3
## # Groups:   country [7]
##    country             type      total_cases
##    <chr>               <chr>           <int>
##  1 Afghanistan         confirmed         234
##  2 Afghanistan         death              10
##  3 Afghanistan         recovered         292
##  4 Albania             confirmed         840
##  5 Albania             death              16
##  6 Albania             recovered         331
##  7 Algeria             confirmed         750
##  8 Algeria             death              15
##  9 Algeria             recovered         529
## 10 Andorra             confirmed          45
## 11 Andorra             death               0
## 12 Andorra             recovered          67
## 13 Angola              confirmed          55
## 14 Angola              death               0
## 15 Angola              recovered           3
## 16 Antigua and Barbuda confirmed           0
## 17 Antigua and Barbuda death               0
## 18 Antigua and Barbuda recovered           0
## 19 Argentina           confirmed        3278
## 20 Argentina           death             138
```

Para ver o total de casos confirmados por país (vamos focar nos top 10, usando print(n=10)):

```
coronavirus %>%
   filter(type == "confirmed") %>%
   group_by(country) %>%
   summarise(total = sum(cases)) %>%
   arrange(-total) %>%
   print(n = 10)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 191 x 2
##    country           total
##    <chr>             <int>
##  1 US             14757000
##  2 India           9677203
##  3 Brazil          6603540
```

```
##  4 Russia          2439163
##  5 France          2345648
##  6 Italy           1728878
##  7 United Kingdom  1727751
##  8 Spain           1684647
##  9 Argentina       1463110
## 10 Colombia        1371103
## # ... with 181 more rows
```

Agora, da mesma forma, o total de mortes por país:

```
coronavirus %>%
   filter(type == "death") %>%
   group_by(country) %>%
   summarise(total = sum(cases)) %>%
   arrange(-total) %>%
   print(n = 10)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 191 x 2
##    country         total
##    <chr>           <int>
##  1 US             282299
##  2 Brazil         176941
##  3 India          140573
##  4 Mexico         109717
##  5 United Kingdom  61342
##  6 Italy           60078
##  7 France          55247
##  8 Iran            50310
##  9 Spain           46252
## 10 Russia          42675
## # ... with 181 more rows
```

E agora o total de recuperados por país:

```
coronavirus %>%
   filter(type == "recovered") %>%
   group_by(country) %>%
   summarise(total = sum(cases)) %>%
   arrange(-total) %>%
   print(n = 10)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 191 x 2
##    country     total
##    <chr>       <int>
##  1 India     9139901
##  2 Brazil    5866657
##  3 US        5624444
##  4 Russia    1920744
##  5 Argentina 1294692
##  6 Colombia  1257410
##  7 Italy      913494
##  8 Peru       907654
```

```
##  9 Germany    868285
## 10 Mexico     866186
## # ... with 181 more rows
```

Vamos selecionar os casos confirmados para o Brasil e ver o total.

```
confirmadosBrasil=coronavirus %>%
   filter(type == "confirmed", country=="Brazil")%>%
   summarise(total = sum(cases))
confirmadosBrasil
```

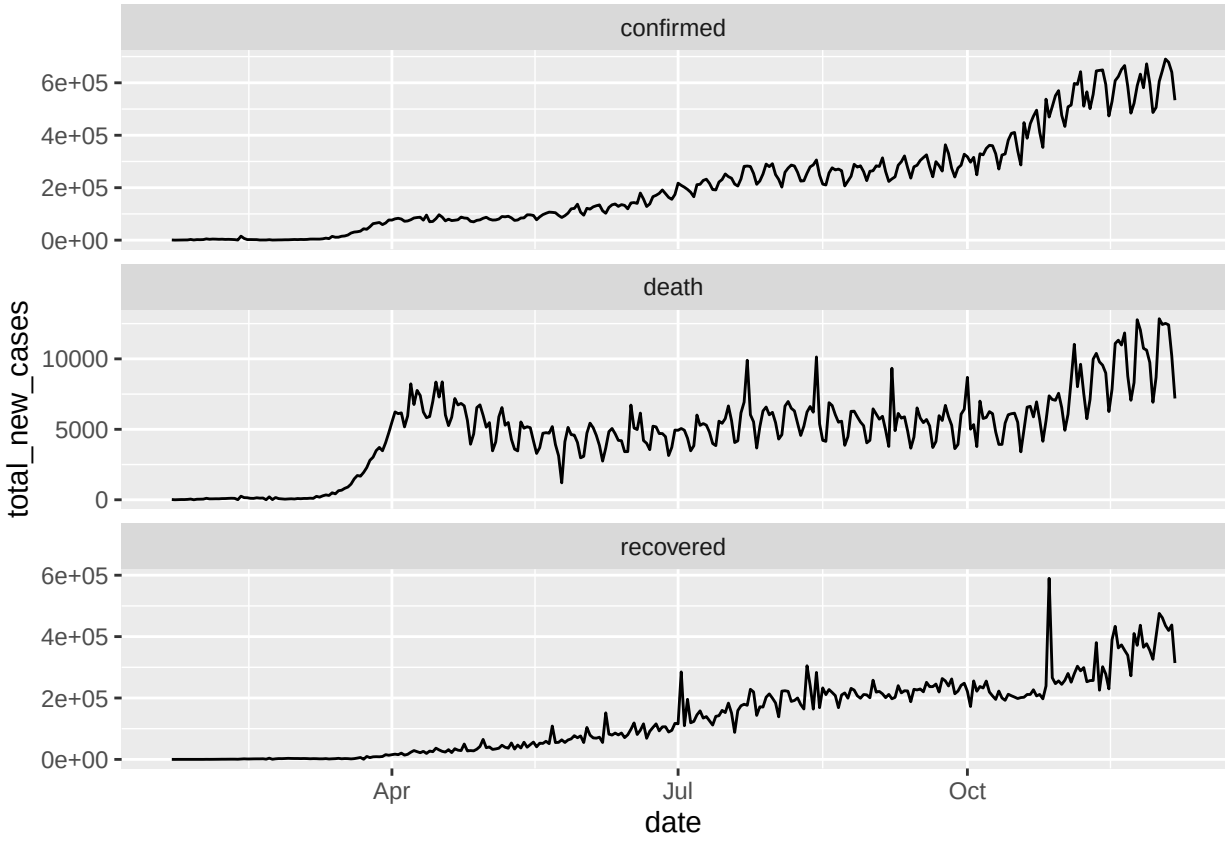
```
##     total
## 1 6603540
```

Vamos fazer um gráfico da evolução do número total de novos casos confirmados, óbitos e recuperados, a nível mundial:

```
totals = coronavirus %>% group_by(date, type) %>%
  summarise(
    total_new_cases = sum(cases)
  )
```

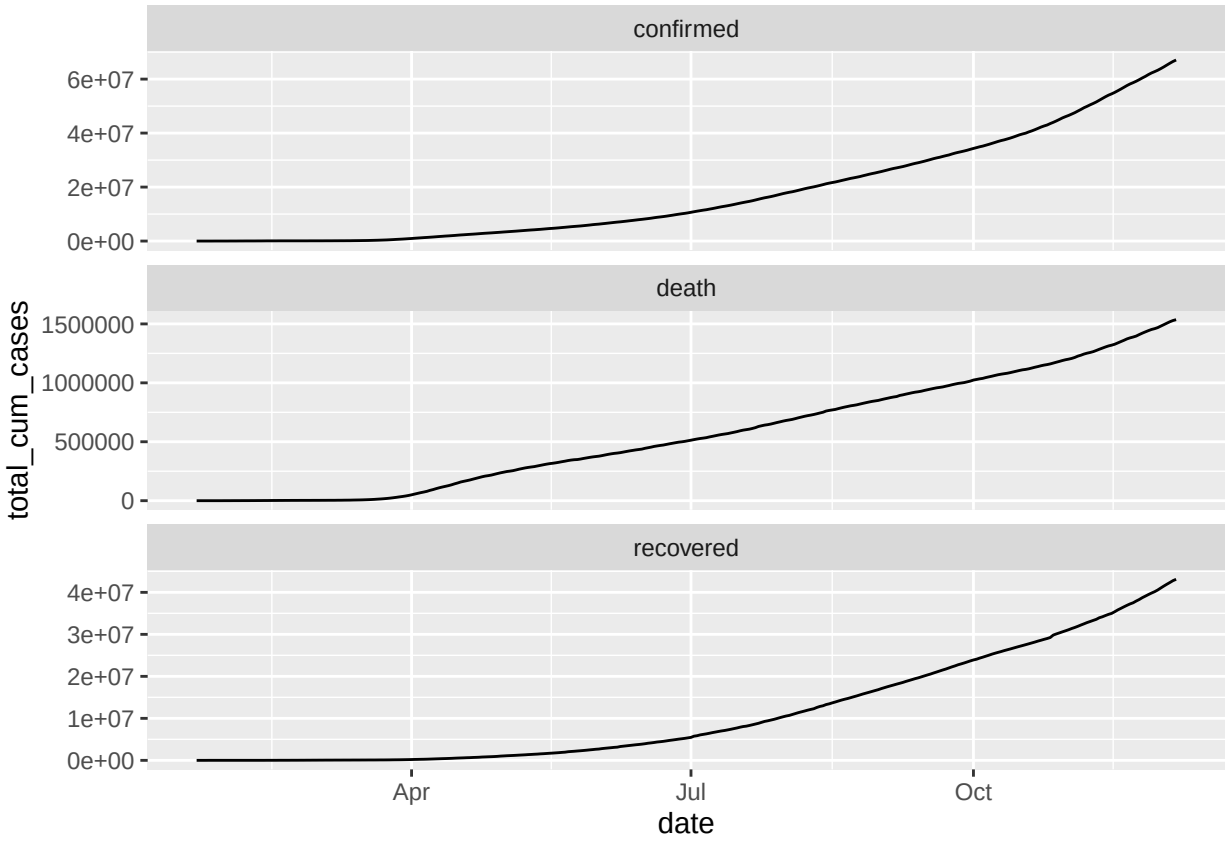
```
## `summarise()` regrouping output by 'date' (override with `.groups` argument)
```

```
totals %>% ggplot(aes(x=date, y=total_new_cases)) +
                    geom_line() +
                    facet_wrap(~type, ncol=1, scales="free_y")
```



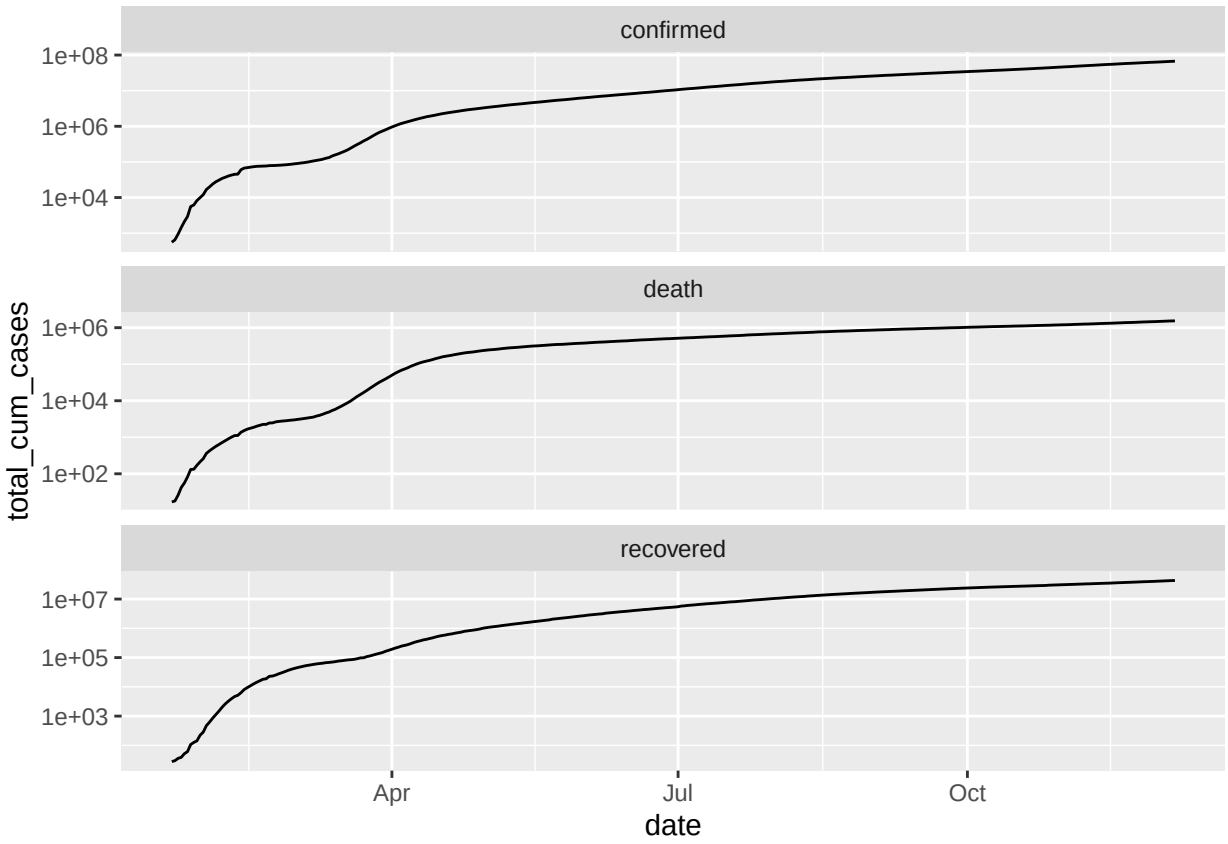
E agora para os valores acumulados:

```
totals = totals %>% group_by(type) %>%  mutate(
    total_cum_cases = cumsum(total_new_cases)
  )

totals %>% ggplot(aes(x=date, y=total_cum_cases)) +
                    geom_line() +
                    facet_wrap(~type, ncol=1, scales="free_y")
```



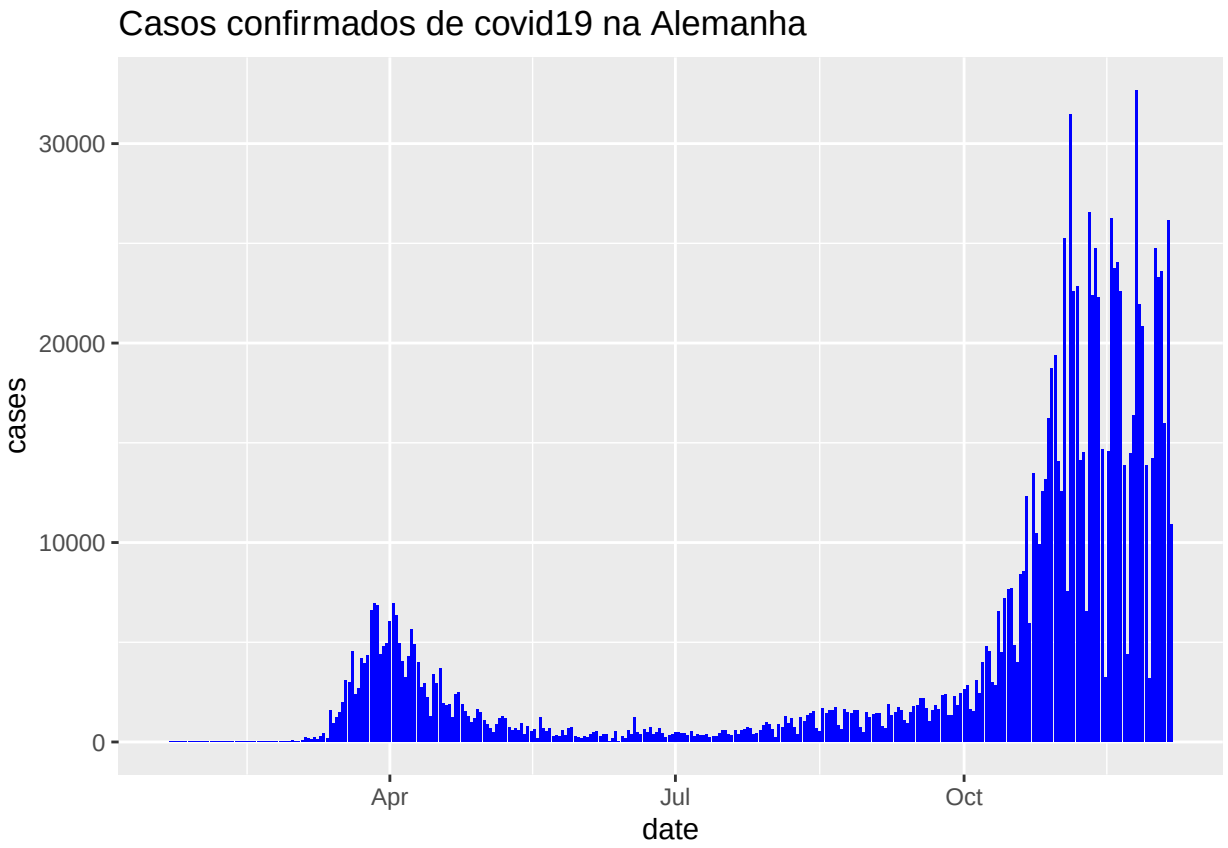
Em escala logarítmica:

```
totals %>% ggplot(aes(x=date, y=total_cum_cases)) +
                    geom_line() +
                    facet_wrap(~type, ncol=1, scales="free_y") +
                  scale_y_log10()
```



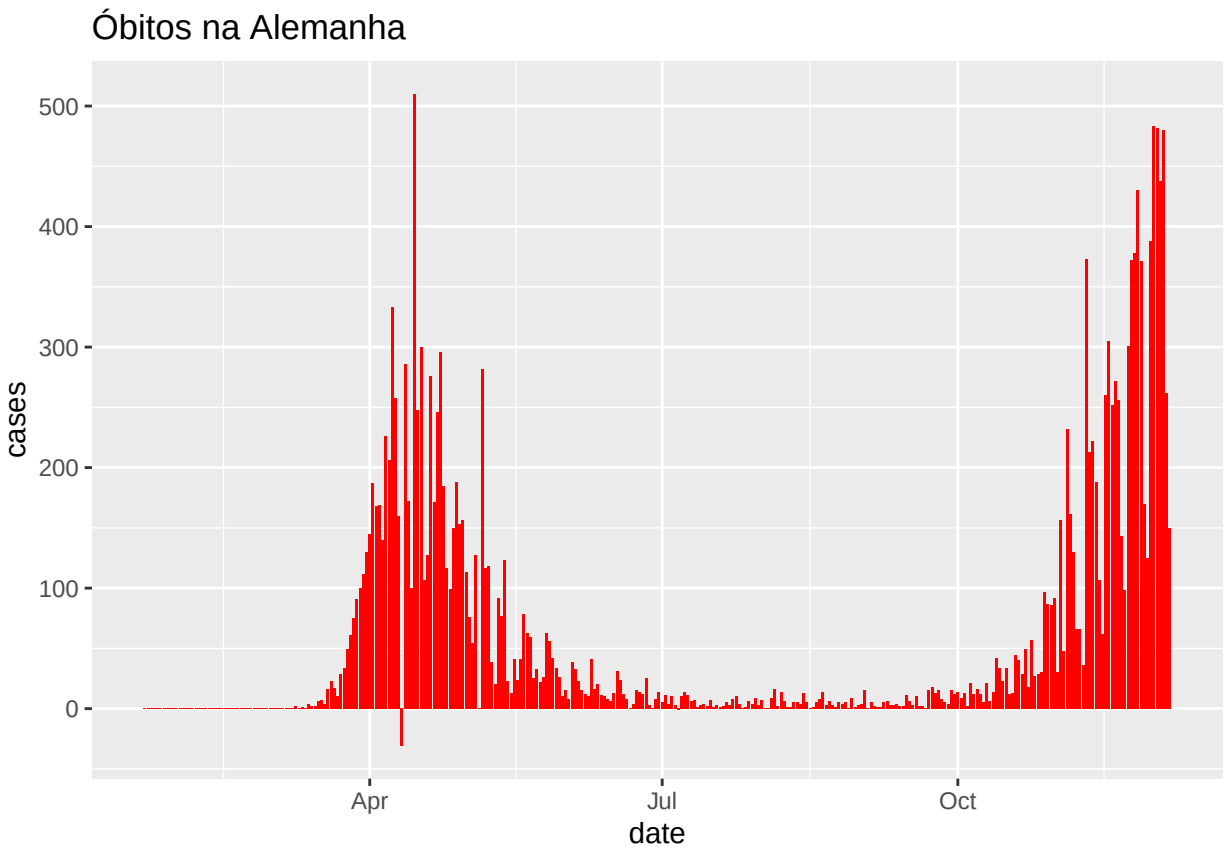
No próximo passo, vamos selecionar um país, no caso a **Alemanha** (Germany) e verificar a evolução do total de confirmados.

```
teste1 = coronavirus%>%filter(country=="Germany",
                      type == "confirmed")

                      plot1=ggplot(teste1)+
                      geom_col(mapping = aes(x=date,
                                             y = cases),
          fill = "blue") +
  labs(title="Casos confirmados de covid19 na Alemanha")
  plot1
```



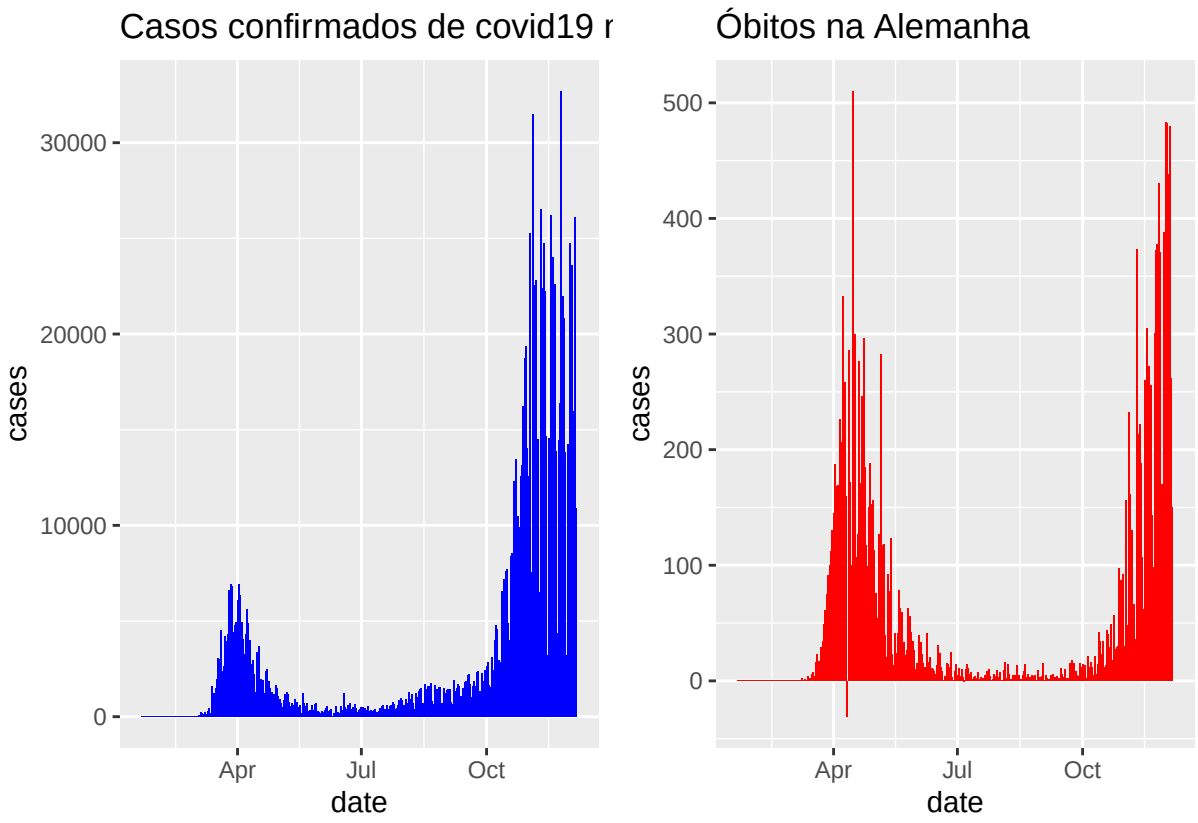
Podemos fazer o mesmo para avaliar a evolução do total de óbitos.

```
teste2 = coronavirus%>%filter(country=="Germany",
                      type == "death")

                      plot2 = ggplot(teste2)+
                      geom_col(mapping = aes(x=date,
                                             y = cases),
          fill = "red") +
  labs(title="Óbitos na Alemanha")
                      plot2
```



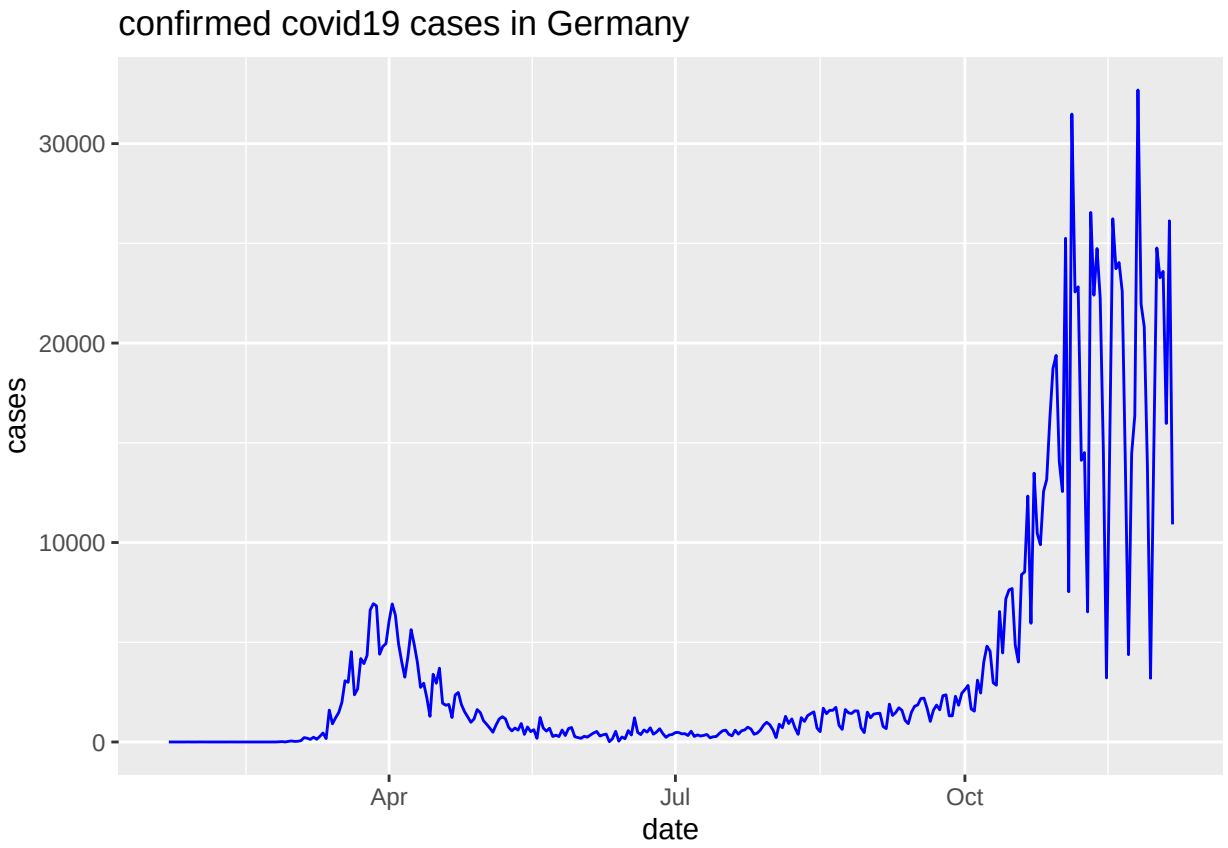
Colocando lado a lado:

```
(plot1 + plot2)
```



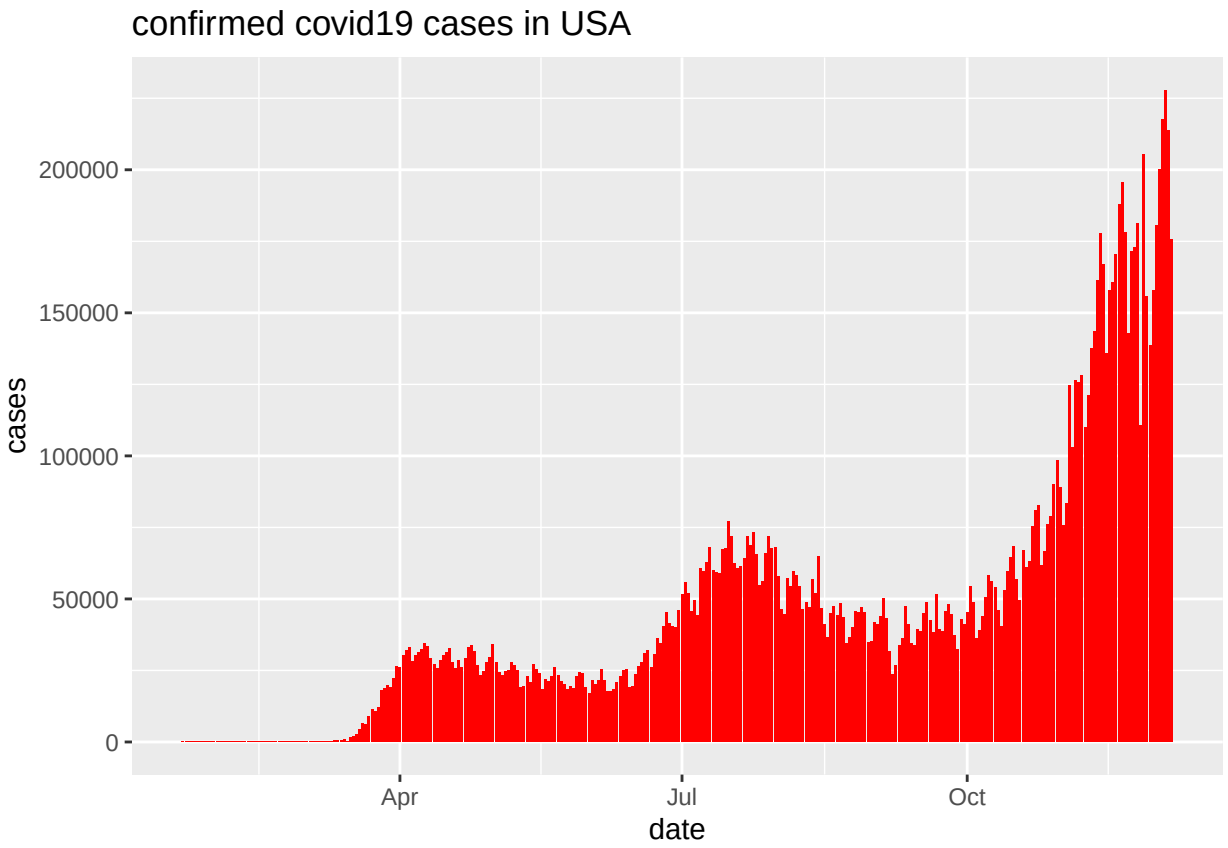
Em vez de geom_col poderíamos usar geom_line também. Vejamos.

```
ggplot(teste1)+
                    geom_line(mapping = aes(x=date,
                                            y = cases),
          color = "blue") +
  labs(title="confirmed covid19 cases in Germany")
```



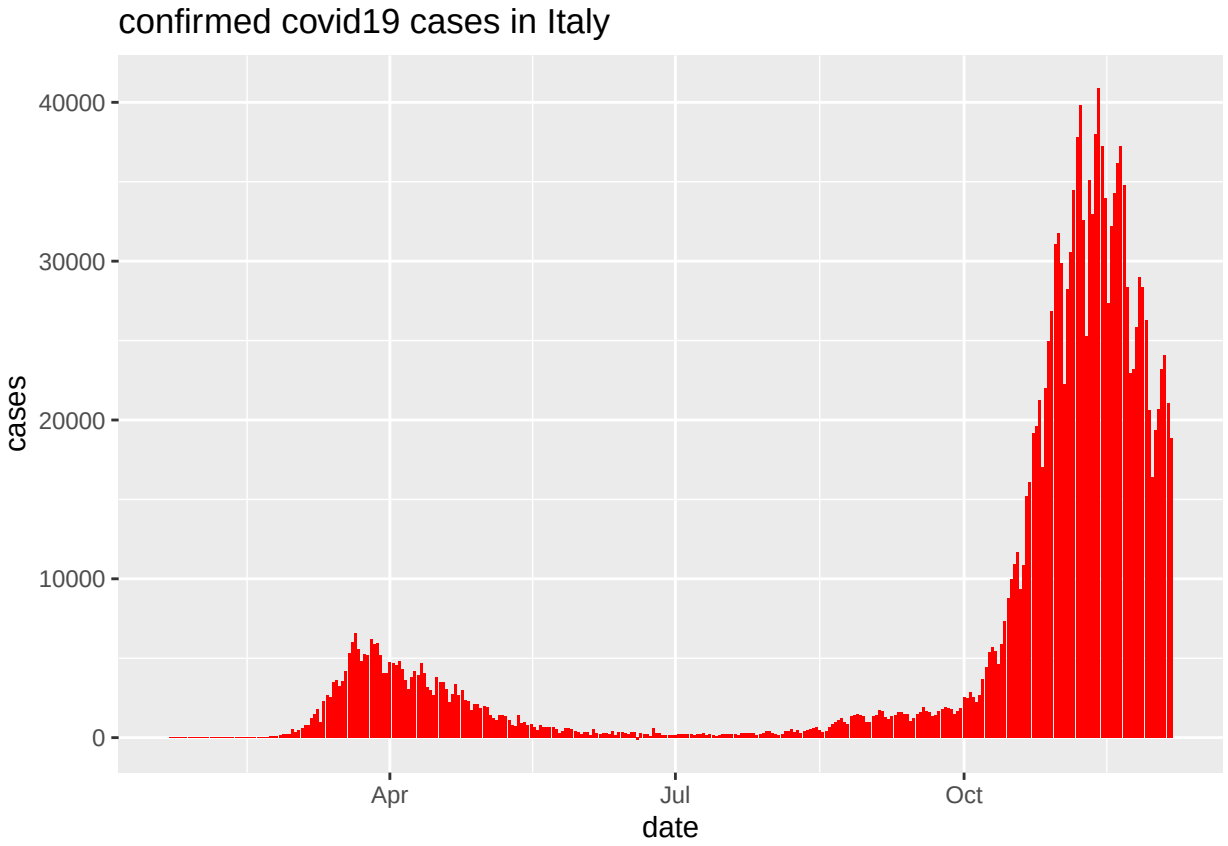
Façamos o mesmo para os casos confirmados dos **EUA** (USA).

```
testeUS = coronavirus%>%filter(country=="US",
                      type == "confirmed")

                      ggplot(testeUS)+
                      geom_col(mapping = aes(x=date,
                                             y = cases),
           fill = "red") +
  labs(title="confirmed covid19 cases in USA")
```



E agora, para a **Itália**:

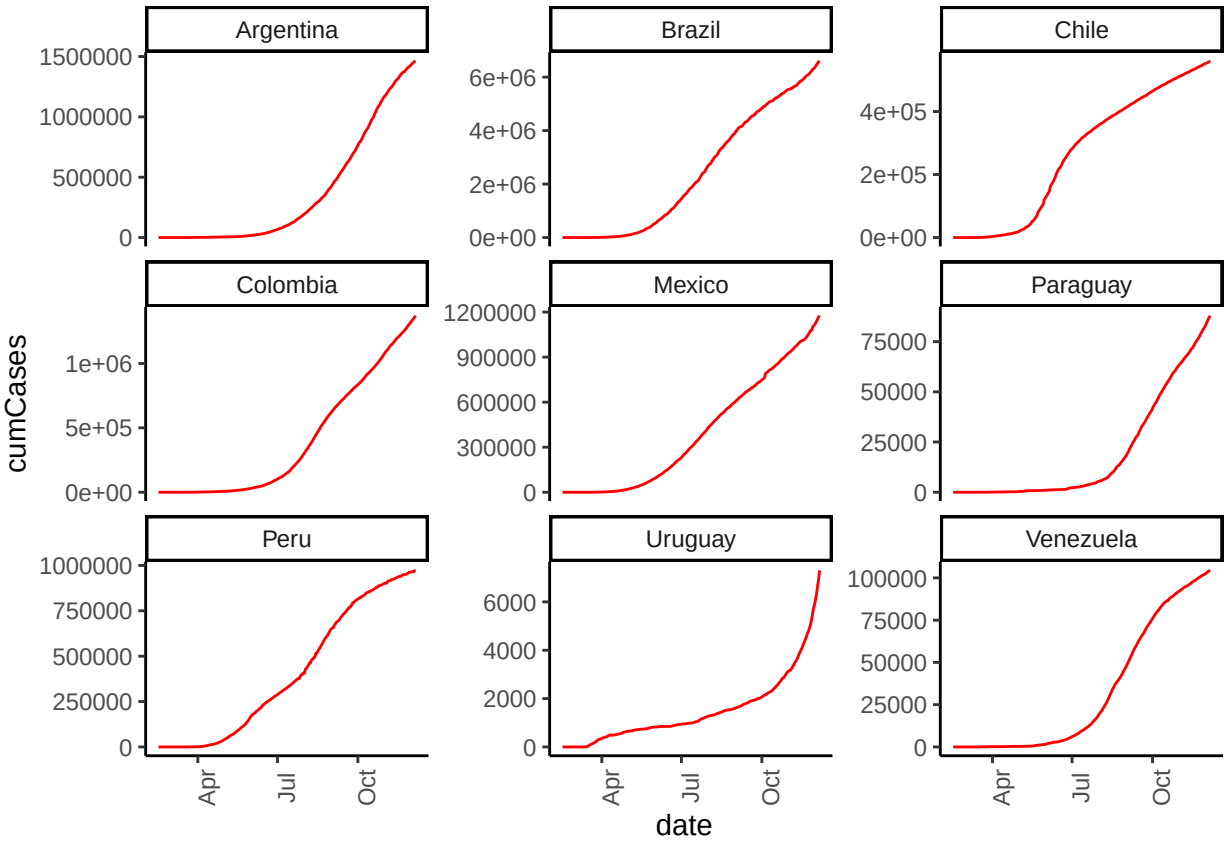
```
testeItalia = coronavirus%>%filter(country=="Italy",
                      type == "confirmed")

                      ggplot(testeItalia)+
                      geom_col(mapping = aes(x=date,
                                             y = cases),
           fill = "red") +
  labs(title="confirmed covid19 cases in Italy")
```



Na **América Latina**, vamos selecionar um grupo de países e fazer um gráfico do número de casos confirmados acumulado usando o recurso facet_wrap do ggpplot.

Vejamos a situação. Considerando essas informações, em qual desses países você gostaria de estar agora?

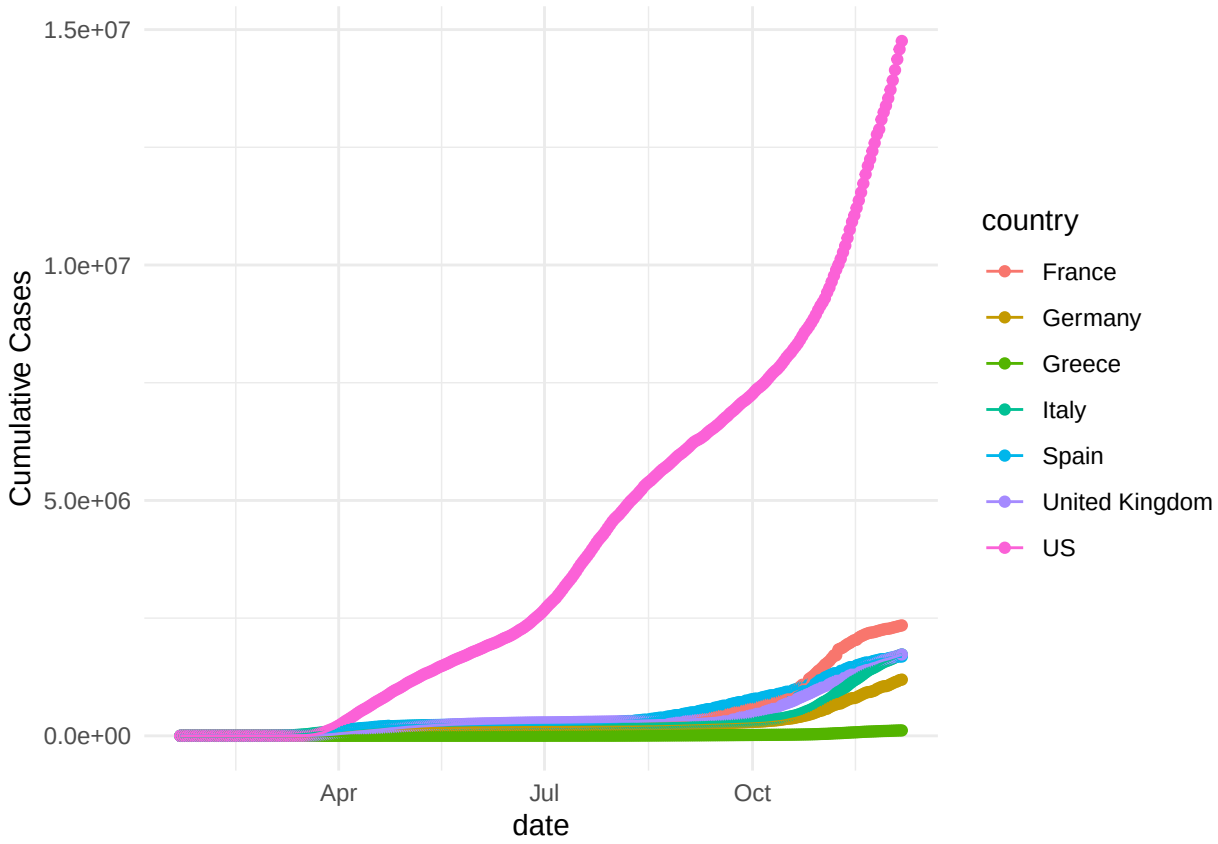
```
LatinAmerica = coronavirus %>% filter(country %in% c("Brazil","Uruguay","Chile","Argentina", "Mexico",
  cumCases = cumsum(cases))



ggplot(LatinAmerica, aes(x=date, y=cumCases)) +
   geom_line(color="red") +   facet_wrap(~country, ncol=3, scales="free_y") +
   theme_classic() +
    theme(axis.text.x = element_text(angle = 90))
```



Vamos agora fazer um gráfico dos casos acumulados para países selecionados em um mesmo gráfico.

```
coronavirus%>%mutate(date=as.Date(date))%>%
  filter(country %in% c("Italy","US","Greece","Spain", "France", "United Kingdom", "Germany"), type=="c

  group_by(date, country)%>%summarise(Daily_Cases=sum(cases))%>%
  group_by(country)%>%arrange(date)%>%
  mutate(Agg_Cases=cumsum(Daily_Cases))%>%
  ggplot(aes(x=date, y=Agg_Cases, col=country))+geom_point()+geom_line()+ylab("Cumulative Cases")+theme_
```

```
## `summarise()` regrouping output by 'date' (override with `.groups` argument)
```



Na tabela abaixo fazemos alguns cálculos para países selecionados em relação ao número de mortes.

```
death_tb<-coronavirus%>%mutate(date=as.Date(date))%>%
  filter(country %in% c("Italy","US","Greece","Spain", "France", "United Kingdom", "Germany"), type=="d
  group_by(date, country)%>%summarise(Daily_Cases=sum(cases))%>%group_by(country)%>%arrange(date)%>%
  mutate(Agg_Cases=cumsum(Daily_Cases), Diff=Daily_Cases/lag(Daily_Cases)-1)%>%arrange(desc(date))%>%sli
```

```
## `summarise()` regrouping output by 'date' (override with `.groups` argument)
```

```
death_tb
```

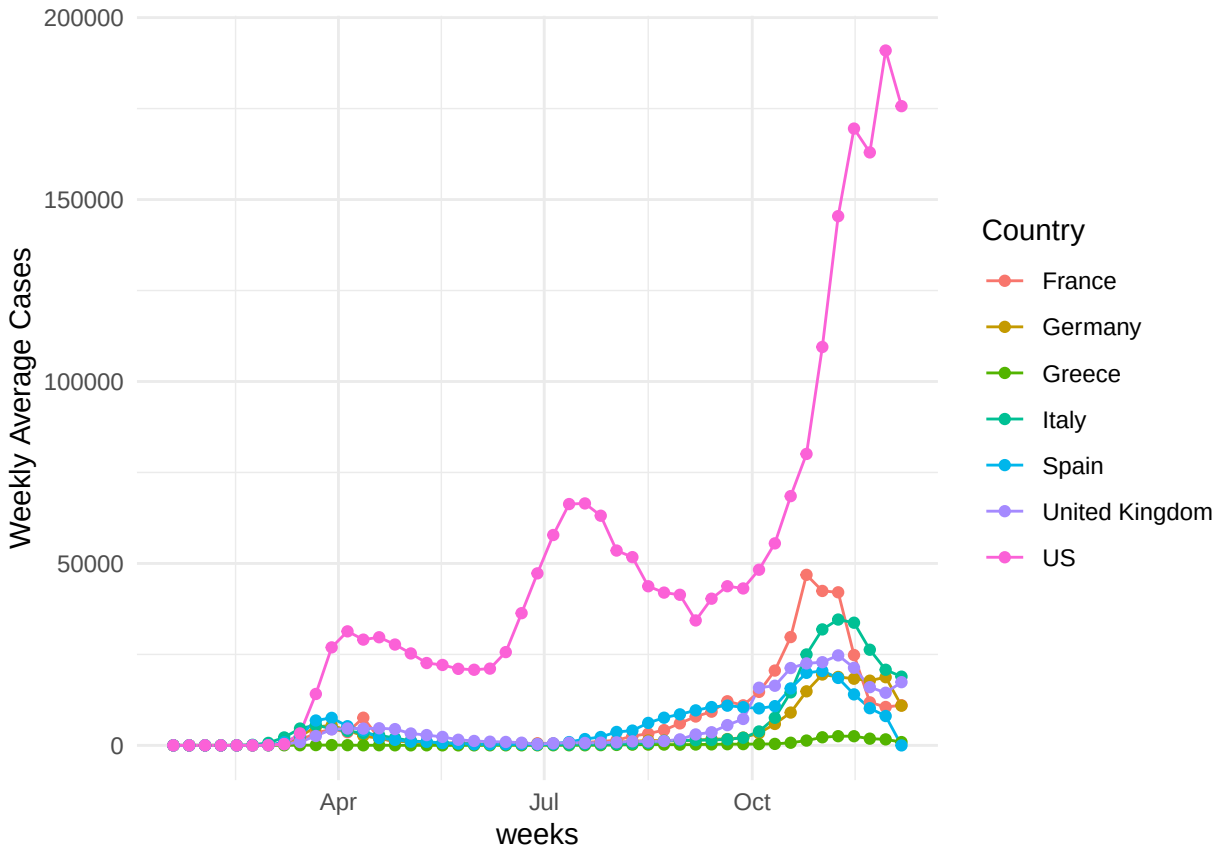
```
## # A tibble: 7 x 5
## # Groups:   country [7]
##   date       country        Agg_Deaths Yestrday_Deaths Change_in_Daily_Deaths
##   <date>     <chr>               <int>           <int>                  <dbl>
## 1 2020-12-06 France              55247             174                 -0.187
## 2 2020-12-06 Germany             18989             150                 -0.427
## 3 2020-12-06 Greece               3003             101                  0.0306
## 4 2020-12-06 Italy               60078             564                 -0.148
## 5 2020-12-06 Spain               46252               0                NaN
## 6 2020-12-06 United Kingdom      61342             231                 -0.418
## 7 2020-12-06 US                 282299            1113                 -0.506
```

E agora um gráfico para a média semanal de casos:

```
weekly=coronavirus%>%filter(type=="confirmed", country %in% c("Italy","US","Greece","Spain", "France",
  mutate(date=as.Date(date), weeks = floor_date(date, "weeks"))%>%group_by(country,weeks)%>%
  summarise(weekly_cases=sum(cases), avg_daily=round(sum(cases)/length(unique(date))))%>%rename(Country=
```

```
## `summarise()` regrouping output by 'country' (override with `.groups` argument)
```

```
ggplot(data = weekly, aes(x=weeks, y=avg_daily, col=Country))+geom_line()+geom_point()+ylab("Weekly Ave
```

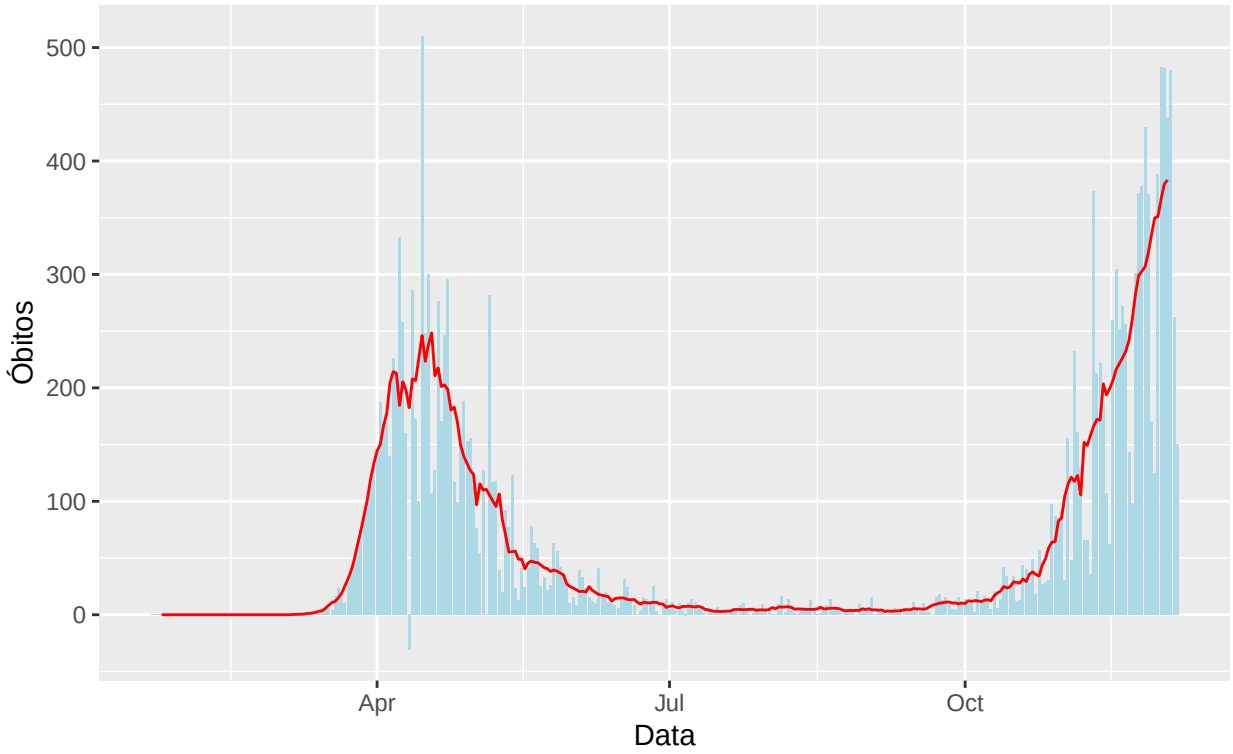


Média móvel

Vejamos através do exemplo abaixo como podemos inserir média móvel usando o R. Para tanto, vamos usar o pacote zoo e a função **rollmean()**. O argumento dessa função é um número inteiro que indica a janela da média móvel (usaremos aqui 7 dias). Vamos usar o exemplo da Alemanha para óbitos (armazenado em teste2) e também para confirmados (armazenado em teste1).

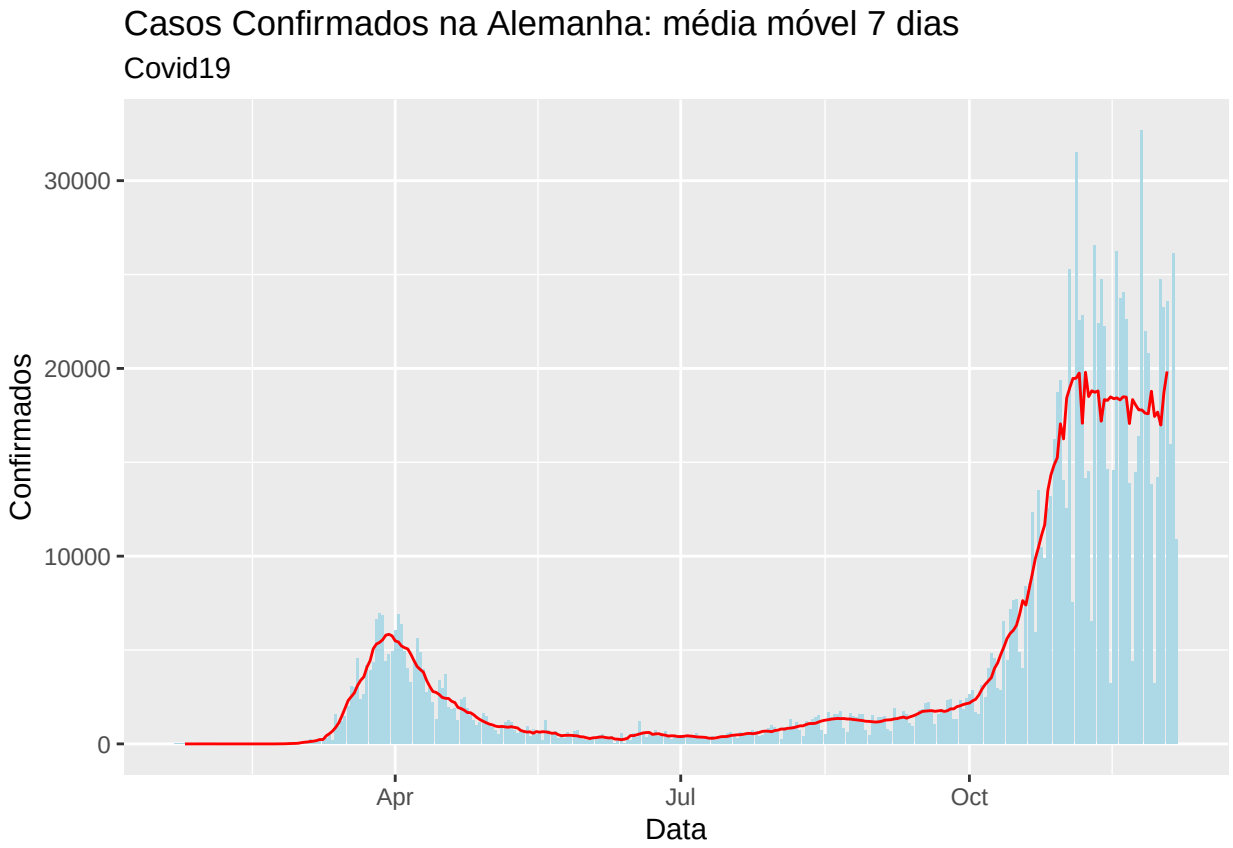
```
##Primeiro óbitos:
móvel=teste2%>%mutate(média_móvel7 = zoo::rollmean(cases, k = 7, fill = NA))

g1=ggplot(data =móvel, aes(x=date)) +
  geom_bar(aes(y = cases), stat = "identity", fill = "lightblue") +
              geom_line(mapping=aes(x = date,
                        y = média_móvel7), color="red")+
      labs(title = "Óbitos na Alemanha: média móvel 7 dias",
                   subtitle = "Covid19",
                   y = "Óbitos",
                   x = "Data")
g1
```

```
## Warning: Removed 6 row(s) containing missing values (geom_path).
```




```
##Agora confirmados:

móvel2=teste1%>%mutate(média_móvel7 = zoo::rollmean(cases, k = 7, fill = NA))

g2=ggplot(data =móvel2, aes(x=date)) +
  geom_bar(aes(y = cases), stat = "identity", fill = "lightblue") +
              geom_line(mapping=aes(x = date,
                        y = média_móvel7), color="red")+
      labs(title = "Casos Confirmados na Alemanha: média móvel 7 dias",
                  subtitle = "Covid19",
                  y = "Confirmados",
                  x = "Data")
g2
```

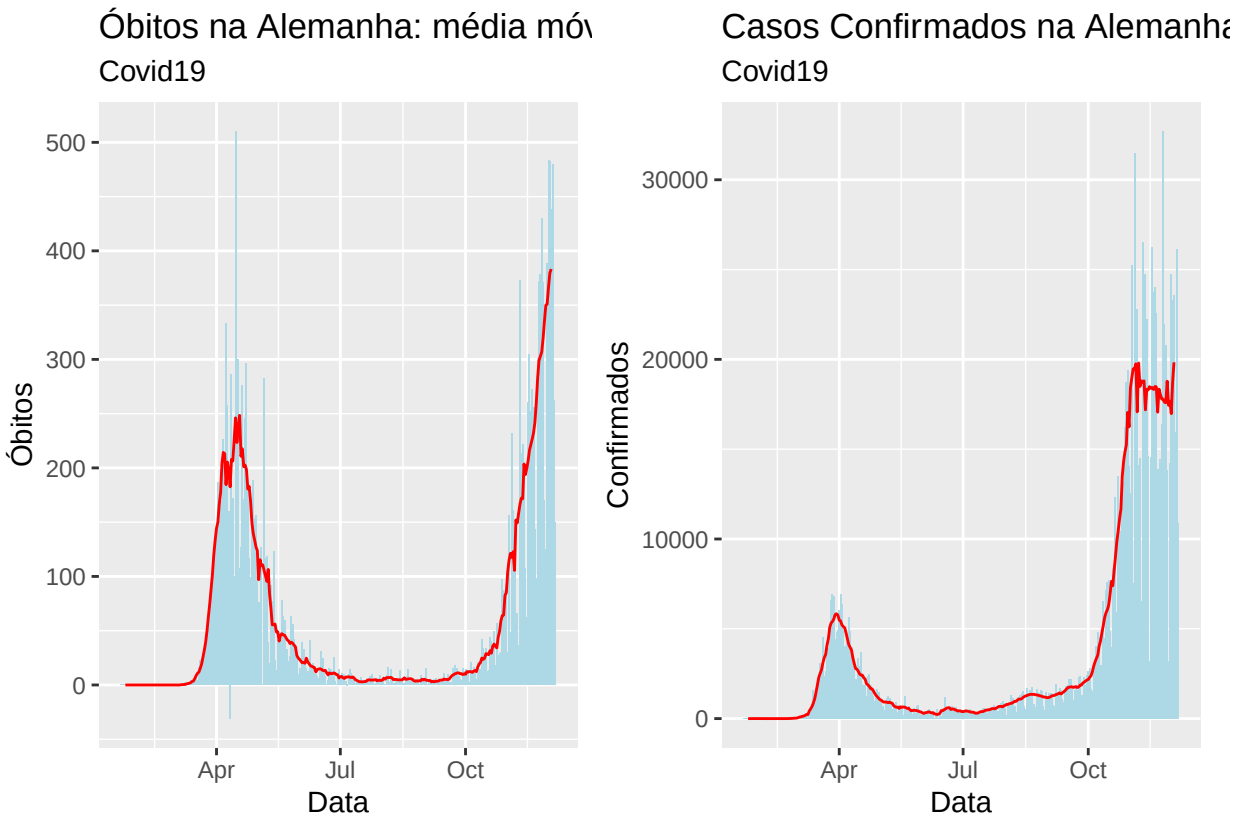
```
## Warning: Removed 6 row(s) containing missing values (geom_path).
```




```
(g1+g2)
```

```
## Warning: Removed 6 row(s) containing missing values (geom_path).
```

```
## Warning: Removed 6 row(s) containing missing values (geom_path).
```

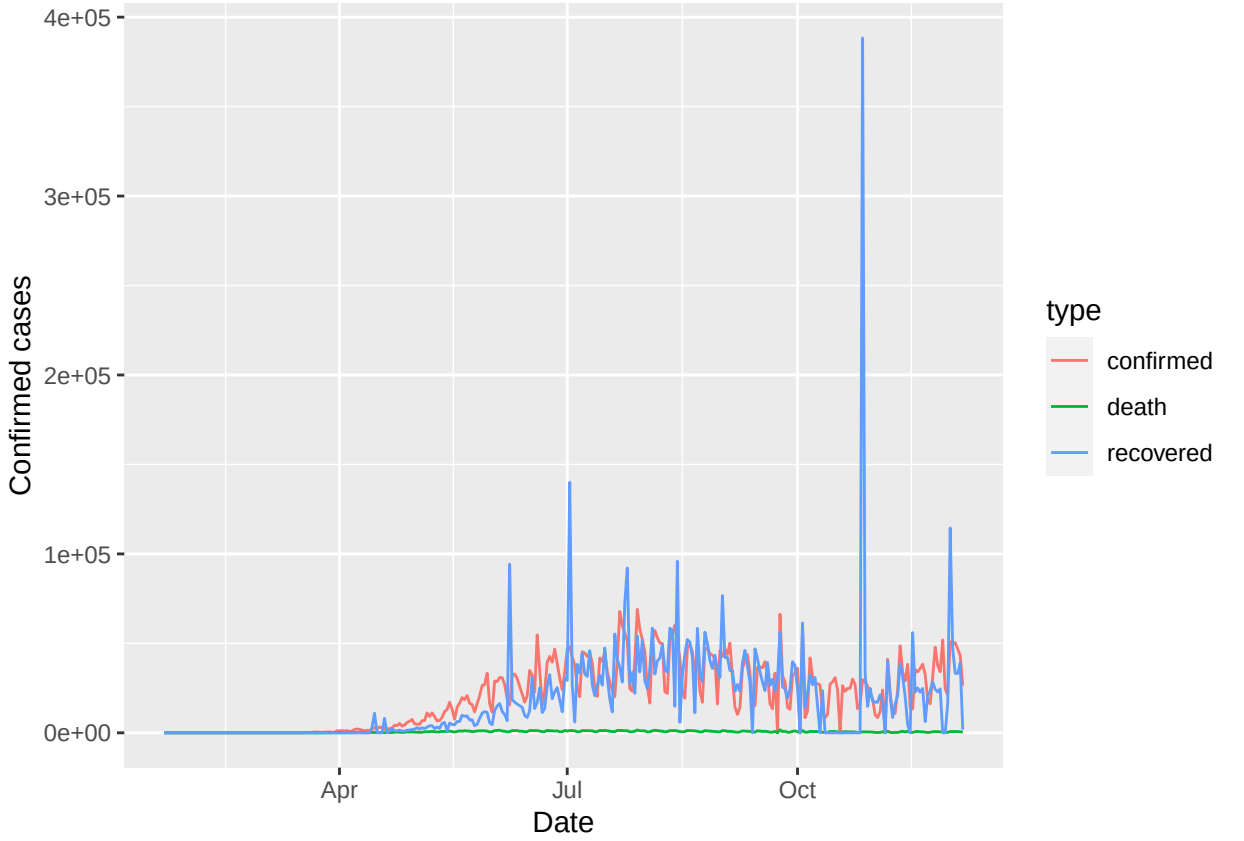


Para finalizar, vamos fazer um gráfico para o **Brasil** dos confirmados, óbitos e recuperados. Para isso, vamos precisar do pacote tidyr e da função pivot_Wider, que nos permitirá acrescentar colunas e diminuir linhas a fim de construir nosso novo conjunto de dados. Acompanhe os comandos abaixo.

```
library(tidyr)

#Primeiro, vamos selecionar apenas o Brasil

cases_brazil <- coronavirus %>%
  filter(country =='Brazil') %>%
  select(date, country, type, cases) %>%
  group_by(date)
#Agora, vamos calcular as variáveis de interesse
Brazil_taxas <- cases_brazil %>%
  filter(date >= '2020-03-01') %>%
  group_by (date, type) %>%
  summarize (total = sum(cases)) %>%
  pivot_wider (names_from = type,
               values_from = total) %>%
  arrange (date) %>%
  ungroup() %>%
  mutate(active = confirmed - death,
         cum_active = cumsum(active),
         cum_confirm = cumsum(confirmed),
         cum_death = cumsum(death),
         cum_recovered = cumsum(recovered))
```

```
## `summarise()` regrouping output by 'date' (override with `.groups` argument)
```

```
df <- as.data.frame(cases_brazil)

#agora o gráfico:

ggplot(cases_brazil) +
  geom_line(aes(x=date, y=cases, group = type, color = type)) +
  ylab("Confirmed cases")+
  xlab("Date")
```

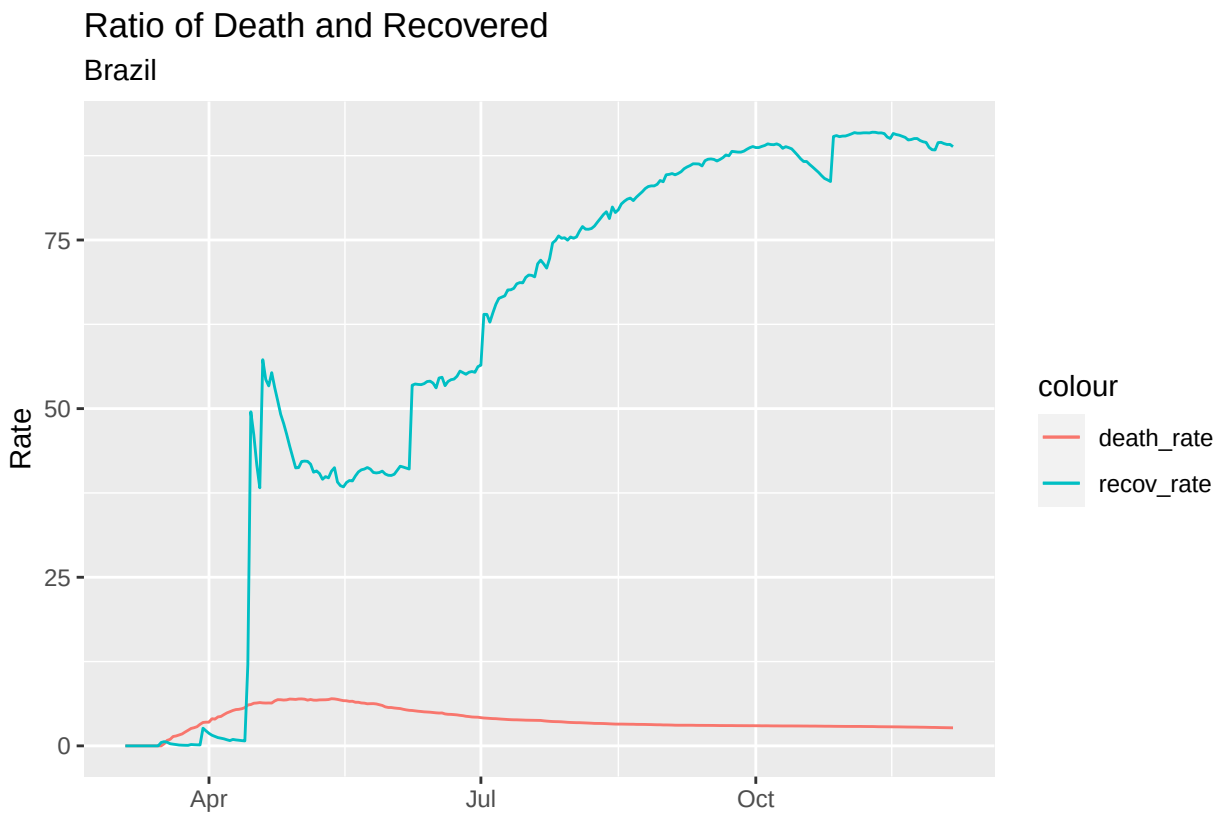


E, por fim, vamos apresentar um gráfico com as taxas de óbitos e recuperados. Vamos calcular as variáveis necessárias e usar ggplot para colocar no gráfico.

```
ratio <- Brazil_taxas %>%
              group_by(date) %>%
              summarise(death = sum(cum_death), confirmed = sum(cum_confirm),recovered = sum(cum_recove
              mutate(recov_rate = 100*(recovered/confirmed))%>%
              mutate(death_rate = 100*(death/confirmed))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
Ratio <- na.omit(ratio)
ggplot(Ratio) +
    geom_line(aes(x=date, y= death_rate, color = 'death_rate')) +
    geom_line(aes(x=date, y= recov_rate, color = 'recov_rate')) +
    labs(x = "", y = 'Rate', title = 'Ratio of Death and Recovered',
       subtitle = 'Brazil')
```


Ratio of Death and Recovered
Brazil
Rate
75
50
25
0
Apr
Jul
Oct
colour
death_rate
recov_rate