

Analyse de la pandémie de COVID-19 aux États-Unis

Ahmed Osman

Présentation du Projet

Dans ce projet, on utilisera plusieurs données liées au COVID-19 aux États-Unis . L'objectif principal est de répondre à certaines questions qui seront posées (section 4) en explorant ces données réelles et en appliquant les techniques de visualisation de données.

Pour faire cela, nous devrons d'abord rendre les données au format “tidy”.

On trouvera dans ces données, la population pour chaque État, le nombre de personnes infectées, décédées,
...

Packages Utilisés

Ces packages sont les packages initiales qu'on utilisera tout au long de notre projet, il se peut qu'on chargera d'autre package dans la suite pour utiliser quelques fonctions...

```
library(tidyverse)

## -- Attaching packages ----- tidyverse 1.3.1 --

## v ggplot2 3.3.5     v purrr    0.3.4
## v tibble   3.1.4     v dplyr    1.0.7
## v tidyr    1.1.4     v stringr  1.4.0
## v readr    2.0.1     vforcats  0.5.1

## -- Conflicts ----- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()   masks stats::lag()

library(lubridate)

##
## Attaching package: 'lubridate'

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

# pour la lecture des fichiers excel
library(readxl)
```

1 - Importations des données et modifications

COVID-19 :

Ce jeu de données contient des informations sur le nombre de personnes infectées, décédées et guéries, ... du COVID-19 aux États-Unis.

source des données : https://github.com/CSSEGISandData/COVID-19/tree/master/csse_covid_19_data/csse_covid_19_daily_reports_us
page web : <https://coronavirus.jhu.edu/us-map>

```
# on récupère les fichiers csv
COVID_data_path <- "csse_covid_19_daily_reports_us/"
COVID_data_files <- list.files(path = "csse_covid_19_daily_reports_us/")

COVID_data <- tibble()
date <- c()
for (i in 1:length(COVID_data_files)) {

  # on crée notre dataframe
  file <- paste0(COVID_data_path, COVID_data_files[i])
  newfile <- read_csv(file, show_col_types = FALSE)
  COVID_data <- COVID_data %>% bind_rows(newfile)

  # on récupère la date (le nom des fichiers)
  n <- nrow(newfile)
  # on se débarrasse des ".csv"
  date <- c(date, rep(str_sub(COVID_data_files[i], 1, 10), n))
}

# on ajoute "date" à la table "COVID_data"
date <- tibble(date)
COVID_data <- bind_cols(date, COVID_data)
# On enlève les variables dont on en a plus besoin
rm(COVID_data_files, COVID_data_path, file, newfile, n, date)

head(COVID_data)

## # A tibble: 6 x 21
##   date Province_State Country_Region Last_Update      Lat  Long_ Confirmed
##   <chr> <chr>        <chr>       <dttm>      <dbl> <dbl>    <dbl>
## 1 01-0~ Alabama      US          2021-01-02 05:30:44  32.3 -86.9    365747
## 2 01-0~ Alaska       US          2021-01-02 05:30:44  61.4 -152.     47019
## 3 01-0~ American Samoa US          2021-01-02 05:30:44 -14.3 -170.      0
## 4 01-0~ Arizona      US          2021-01-02 05:30:44  33.7 -111.     530267
## 5 01-0~ Arkansas     US          2021-01-02 05:30:44  35.0 -92.4    229442
## 6 01-0~ California   US          2021-01-02 05:30:44  36.1 -120.    2437211
## # ... with 14 more variables: Deaths <dbl>, Recovered <dbl>, Active <dbl>,
## #   FIPS <dbl>, Incident_Rate <dbl>, Total_Test_Results <dbl>,
## #   People_Hospitalized <dbl>, Case_Fatality_Ratio <dbl>, UID <dbl>,
## #   ISO3 <chr>, Testing_Rate <dbl>, Hospitalization_Rate <dbl>,
## #   People_Tested <dbl>, Mortality_Rate <dbl>
```

Après avoir importer notre jeu de données, on “nettoie” (sélectionne, filtre, corrige..) les données, pour rendre cette table au format **tidy**.

```
# correction des types des colonnes qui nous intéresse
COVID_data <- COVID_data %>%
  # correction du type de la variable date
  mutate(date = mdy(date)) %>%
  # correction des types des colonnes
  mutate(Confirmed = as.integer(Confirmed),
        Deaths = as.integer(Deaths),
        Recovered = as.integer(Recovered),
        Active = as.integer(Active)) %>%
  # renommage des colonnes
  rename(state = Province_State, confirmed = Confirmed,
         deaths = Deaths, recovered = Recovered, active = Active) %>%
  # On sélectionne les colonnes qui nous intéresse
  select(date, state, confirmed, deaths, recovered, active)

# On remarque que "Recovered" est dans la colonne State par erreur, on souhaite donc
# l'enlever
COVID_data <- COVID_data %>%
  filter(state != "Recovered")

head(COVID_data)

## # A tibble: 6 x 6
##   date      state     confirmed  deaths recovered  active
##   <date>    <chr>     <int>     <int>     <int>     <int>
## 1 2021-01-01 Alabama     365747    4872    202137   158738
## 2 2021-01-01 Alaska      47019     206     7165    39615
## 3 2021-01-01 American Samoa     0       0       NA       0
## 4 2021-01-01 Arizona      530267    9015    76934   444318
## 5 2021-01-01 Arkansas     229442    3711   199247   26484
## 6 2021-01-01 California    2437211   26168      NA  2309915
```

Population des États-Unis :

Ce jeu de données contient des informations sur la population pour chaque État aux États-Unis.

source des données : <https://data.ers.usda.gov/reports.aspx?ID=17827>

```
# importations des données
pop_data <- read_xlsx("PopulationReport.xlsx")

pop_data <- pop_data %>%
  # On sélectionne les colonnes qui nous intéresse
  select(state = Name, population = `Pop. 2020`) %>%
  # les deux dernières lignes n'ont aucun rapport avec la population aux US
  filter(population != is.na(population))

head(pop_data)

## # A tibble: 6 x 2
```

```

##   state      population
##   <chr>      <dbl>
## 1 United States  331449281
## 2 Alabama        5024279
## 3 Alaska         733391
## 4 Arizona        7151502
## 5 Arkansas       3011524
## 6 California     39538223

```

Vaccinations :

Ce jeu de données contient des informations sur le nombre de personnes vaccinées “v” au jour “j” pour chaque État aux États-Unis.

source des données : <https://ourworldindata.org/us-states-vaccinations>

```

vacc_data <- read_csv("us_state_vaccinations.csv", show_col_types = FALSE)

# On corrige le nom de l'État "New York State" par "New York" pour empêcher les
# erreurs de jointures qu'on fera aux étapes suivantes.

vacc_data$location[vacc_data$location == "New York State"] <- "New York"

vacc_data <- vacc_data %>%
  # correction
  mutate(date = ymd(date),
         daily_vaccinations = as.numeric(daily_vaccinations)) %>%
  # sélection
  select(state = location,
         date = date,
         vaccinations = daily_vaccinations)

head(vacc_data)

## # A tibble: 6 x 3
##   state    date    vaccinations
##   <chr>   <date>      <dbl>
## 1 Alabama 2021-01-12      NA
## 2 Alabama 2021-01-13     5906
## 3 Alabama 2021-01-14     7083
## 4 Alabama 2021-01-15     7478
## 5 Alabama 2021-01-16     7498
## 6 Alabama 2021-01-17     7509

```

Régions et noms des Etats-Unis :

Cet ensemble de données contient des informations sur les noms des États-Unis et de leurs régions. On va se référer sur cette base de données et y faire des jointures dessus, nous avons donc besoin de ces données.

source des données : <https://www.kaggle.com/omer2040/usa-states-to-region>

```

# importation
states_data <- read.csv("states.csv")

# On sélectionne les colonnes qui nous intéresse
states_data <- states_data %>%
  select(state = State, region = Region)

head(states_data)

##          state region
## 1      Alaska    West
## 2   Alabama    South
## 3  Arkansas    South
## 4   Arizona    West
## 5 California    West
## 6 Colorado    West

```

2 - pré-Analyse des données

On veut vérifier l'ensemble des données principale et on s'intéresse aux variables principales ("confirmed", "deaths", "recovered", "active"). On veut donc savoir si les données sont cohérentes au niveau des États-Unis et à un niveau pour un État sélectionné.

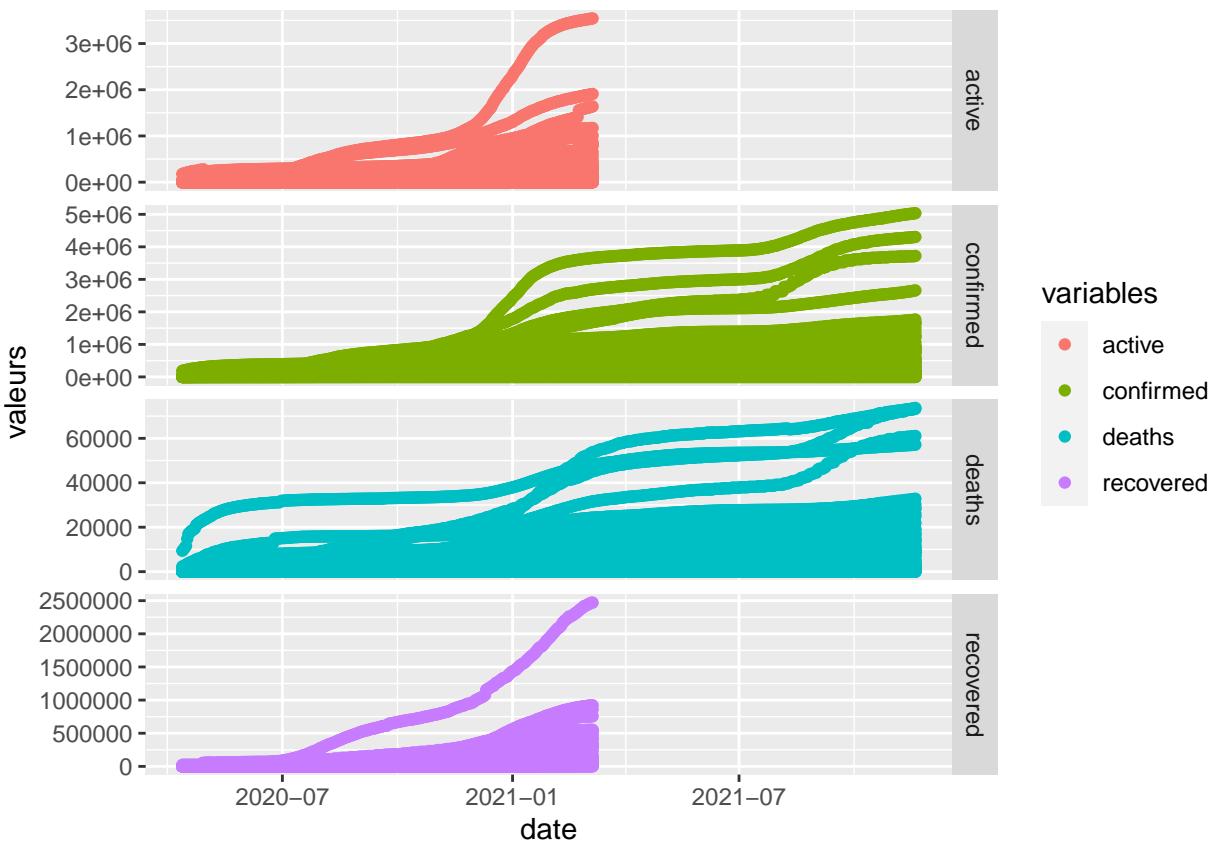
Au niveau des États-Unis :

```

COVID_data %>%
  # sélection des variables principales
  select(state,date, confirmed:active) %>%
  # on transforme la table du format "wide" au format "long"
  pivot_longer(cols = c("confirmed", "deaths", "recovered", "active"),
               names_to = "variables",
               values_to = "valeurs") %>%
  # graphique des valeurs en fonction de la date
  ggplot(aes(x = date,
             y = valeurs,
             color = variables)) +
  geom_point() +
  facet_grid(variables ~ .,
             scales = "free")

```

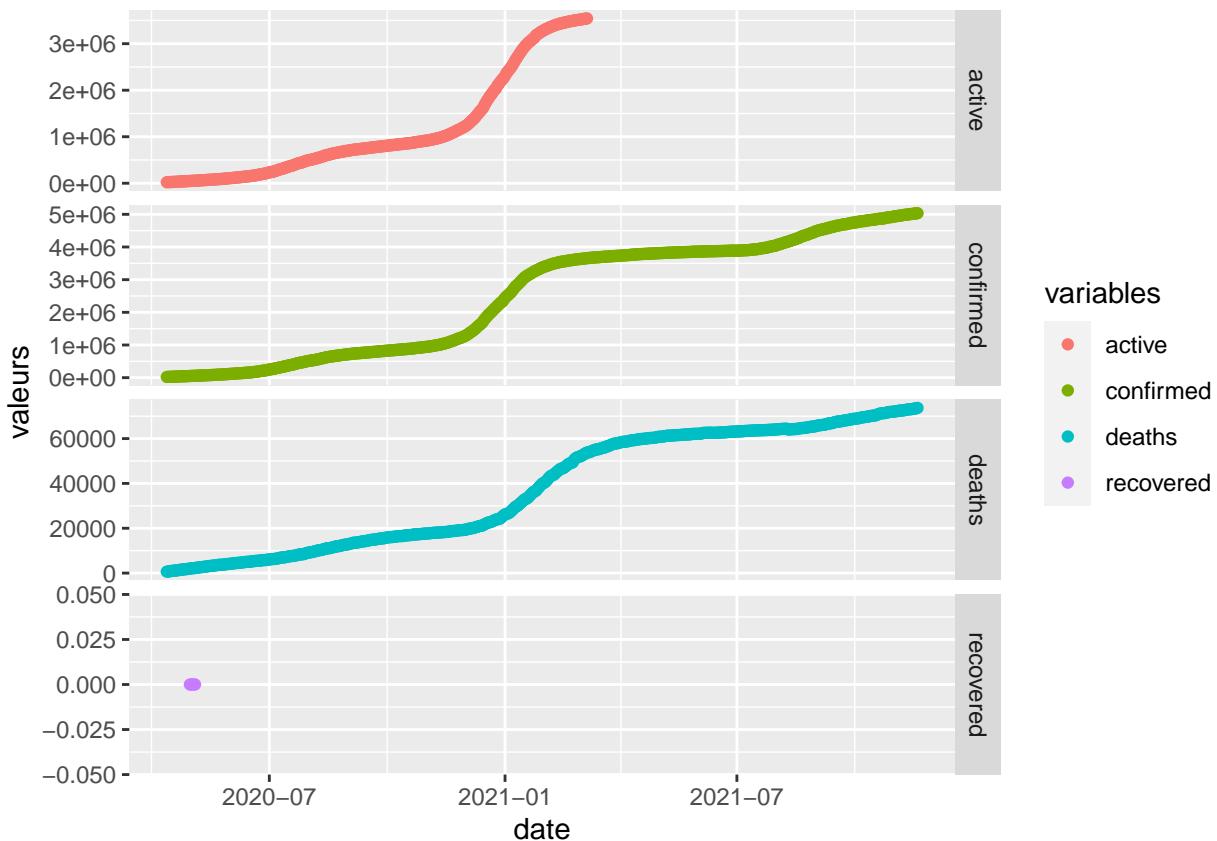
Warning: Removed 33923 rows containing missing values (geom_point).



Au niveau de l'État Californie :

```
COVID_data %>%
  # sélection des variables principales
  select(state, date, confirmed:active) %>%
  # on choisit que les observations de "California"
  filter(state == "California") %>%
  # on transforme la table du format "wide" au format "long"
  pivot_longer(cols = c("confirmed", "deaths", "recovered", "active"),
               names_to = "variables",
               values_to = "valeurs") %>%
  # graphique des valeurs en fonction de la date
  ggplot(aes(x = date,
             y = valeurs,
             color = variables)) +
  geom_point() +
  facet_grid(variables ~ .,
             scales = "free")
```

Warning: Removed 840 rows containing missing values (geom_point).



On remarque, d'après les graphiques précédents, que les données des variables **active** et **recovered** sont incomplètes (données manquantes : on a pas des données suffisantes pour ces variables). On s'intéressera donc qu'aux variables **confirmed** et **deaths**.

De plus, on voit que la colonne **confirmed** est le nombre total de personnes infectées.

3 - Création de la table principale

On va maintenant créer une seule table. Cette table contiendra les colonnes qui nous intéressent seulement. Chaque observations aura comme identifiant une date unique pour chaque État, ainsi l'ID de cette table sera composé de l'ensemble (**date**, **state**).

3.1 - Jointures

Pour créer cette table, on a besoin d'effectués des jointures. On va donc se référer sur la table **states_data** et on effectuera les jointures dessus.

```
# table states_data
head(states_data)
```

```
##           state region
## 1      Alaska    West
## 2   Alabama     South
## 3 Arkansas     South
## 4 Arizona     West
```

```

## 5 California    West
## 6 Colorado     West

data <- states_data %>%
  # on ordonne la table states_data
  arrange() %>%

  # on a joute un id pour chaque state
  mutate(state_id = row_number()) %>%

  # on effectue les jointures
  left_join(COVID_data, by = "state") %>%
  left_join(pop_data, by = "state") %>%
  left_join(vacc_data, by = c("state", "date")) %>%

  # sélection des colonnes
  select(state_id, state, region, date:active, vaccinations, population) %>%

  # on renomme les colonne
  rename(`confirmed total` = confirmed,
         `deaths total` = deaths,
         `daily vaccine doses` = vaccinations)

head(data)

##   state_id state region      date confirmed total deaths total recovered
## 1          1 Alaska  West 2021-01-01      47019     206    7165
## 2          1 Alaska  West 2021-01-02      47821     215    7165
## 3          1 Alaska  West 2021-01-03      48118     215    7165
## 4          1 Alaska  West 2021-01-04      48382     218    7165
## 5          1 Alaska  West 2021-01-05      48582     218    7165
## 6          1 Alaska  West 2021-01-06      48922     220    7165
##   active daily vaccine doses population
## 1 39615             NA 733391
## 2 40421             NA 733391
## 3 40718             NA 733391
## 4 40980             NA 733391
## 5 41177             NA 733391
## 6 41514             NA 733391

```

3.2 - Opérations sur la table principales

Ce que l'on voudrait faire :

- ajouter une variable qui compte la population en millions
- calculer le nombre quotidien de cas “confirmed”, “deaths”.
- vérifier les valeurs manquantes et les corriger si possible
- obtenir la première date de vaccinations pour chaque État (date de sortie du vaccin)
- vérifier si on a des valeurs négatives

Valeurs manquantes :

```
# valeurs manquantes pour les variables "confirmed total" et "deaths total"
data %>% filter(is.na(`confirmed total`)) %>% nrow()
```

```
## [1] 0
```

```
data %>% filter(is.na(`deaths total`)) %>% nrow()
```

```
## [1] 0
```

```
# valeurs manquantes pour les vaccinations
```

```
data %>% filter(is.na(`daily vaccine doses`)) %>% nrow()
```

```
## [1] 14621
```

La vaccination a commencé beaucoup plus tard après le début du COVID-19, nous nous attendions donc à des valeurs manquantes. On peut donc remplacer ces valeurs manquantes par 0. (pas de vaccinations au jour “j”)

Première date de vaccinations :

```
# 1ère date de vaccination pour chaque État
date.min.vacc.state <- data %>%
  filter(!is.na(`daily vaccine doses`)) %>%
  group_by(state) %>%
  summarise(min_date = min(date)) %>%
  ungroup()
```

```
head(date.min.vacc.state)
```

```
## # A tibble: 6 x 2
##   state      min_date
##   <chr>     <date>
## 1 Alabama   2021-01-13
## 2 Alaska    2021-01-13
## 3 Arizona   2021-01-13
## 4 Arkansas  2021-01-13
## 5 California 2021-01-13
## 6 Colorado   2021-01-13
```

On voit d'après la table précédente que les personnes ont commencé à prendre le vaccin le **"13-01-2021"** pour la plupart des States.

Opérations restantes :

```

data <- data %>%
  # population de chaque État en millions
  mutate(`population in millions` = round(population / 10**6, 2)) %>%

  # on remplace les NA par 0
  mutate(`daily vaccine doses` = replace_na(`daily vaccine doses`, 0)) %>%

  # calcul quotidiens des variables : total(date actuelle) - total(date précédente)
  group_by(state) %>%
  mutate(`confirmed daily cases` = `confirmed total` - lag(`confirmed total`, 1),
        `deaths daily cases` = `deaths total` - lag(`deaths total`, 1)) %>%

  # calcul du nombre total au fil du temps : somme du premier jusqu'au dernier
  mutate(`vaccine doses total` = cumsum(`daily vaccine doses`)) %>%
  ungroup() %>%

  # rearrange columns
  select(state_id:date,
        `confirmed total` , `confirmed daily cases` ,
        `deaths total`     , `deaths daily cases` ,
        `vaccine doses total` , `daily vaccine doses` ,
        population, `population in millions` ,
        everything())

head(data)

## # A tibble: 6 x 14
##   state_id state  region date      `confirmed total` `confirmed daily cases`
##       <int> <chr> <chr> <date>          <int>                  <int>
## 1         1 Alaska West 2021-01-01        47019                   NA
## 2         1 Alaska West 2021-01-02        47821                   802
## 3         1 Alaska West 2021-01-03        48118                   297
## 4         1 Alaska West 2021-01-04        48382                   264
## 5         1 Alaska West 2021-01-05        48582                   200
## 6         1 Alaska West 2021-01-06        48922                   340
## # ... with 8 more variables: deaths total <int>, deaths daily cases <int>,
## #   vaccine doses total <dbl>, daily vaccine doses <dbl>, population <dbl>,
## #   population in millions <dbl>, recovered <int>, active <int>
```

Valeurs négatives

```

data %>% filter(`confirmed daily cases` < 0) %>% head()

## # A tibble: 6 x 14
##   state_id state  region date      `confirmed total` `confirmed daily cases`
##       <int> <chr> <chr> <date>          <int>                  <int>
## 1         1 Alaska West 2020-04-12        272                  -64510
## 2         1 Alaska West 2020-04-13        277                  -64903
## 3         1 Alaska West 2020-04-14        285                  -65051
## 4         1 Alaska West 2020-04-15        293                  -65243
```

```

## 5      1 Alaska West  2020-04-16          300        -65430
## 6      1 Alaska West  2020-04-17          309        -65612
## # ... with 8 more variables: deaths total <int>, deaths daily cases <int>,
## #   vaccine doses total <dbl>, daily vaccine doses <dbl>, population <dbl>,
## #   population in millions <dbl>, recovered <int>, active <int>

# On fait pareil pour les autres variables
# data %>% filter(`deaths daily cases` < 0)
# data %>% filter(`daily vaccine doses` < 0)

# on remplace ces valeurs négatives par 0
data$`confirmed daily cases`[data$`confirmed daily cases` < 0] <- 0
data$`deaths daily cases`[data$`deaths daily cases` < 0] <- 0
data$`daily vaccine doses`[data$`daily vaccine doses` < 0] <- 0

```

4 - Questions

Cette section représente les questions qu'on peut se poser.

1 - Trouver le nombre d'états par région et représenter les graphiquement.

```

data %>%
  group_by(region) %>%
  summarise(states = n_distinct(state)) %>%
  ungroup()

## # A tibble: 4 x 2
##   region     states
##   <chr>      <int>
## 1 Midwest      12
## 2 Northeast     9
## 3 South       17
## 4 West        13

# on ajoute le nom des états en minuscule pour faire une jointure et obtenir
# des données géographiques
data <- data %>%
  mutate(state_ = tolower(state))

# dernière date dans les données (la plus récente)
max.date <- data %>% pull(date) %>% max()

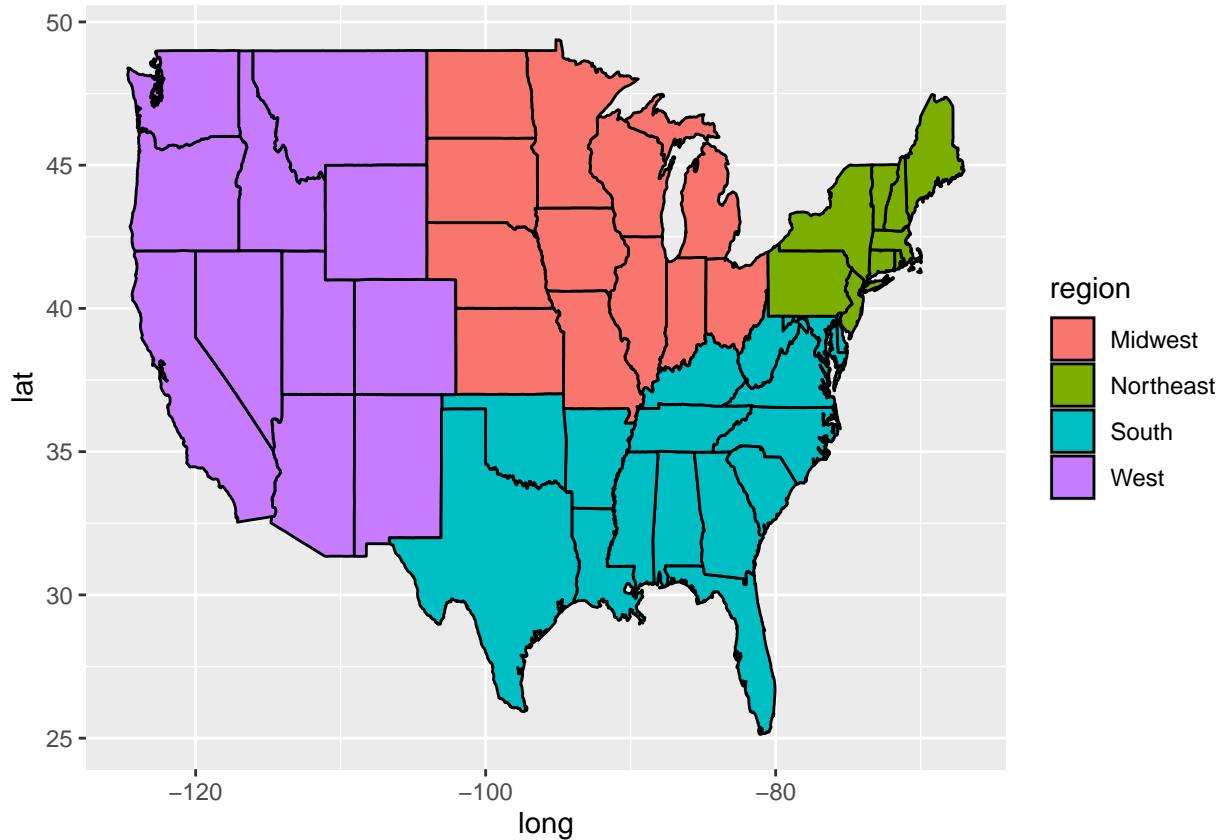
# afficher les états sur la carte (map) selon la dernière date
# (couleur de l'état définie par région)
data %>%
  filter(date == ymd(max.date)) %>%
  # obtenir les données de longitude et de latitude
  left_join(x = .,
            y = map_data("state"),

```

```

    by = c("state_" = "region")) %>%
ggplot(aes(x = long, y = lat,
           group = group)) +
geom_polygon(aes(fill = region),
             color = "black")

```



2 - Que se passe-t-il avec le nombre total de personnes infectées/décédées (nombre absolu/relatif) au fil du temps au niveau de l'État ?

Le nombre relatif s'obtient en calculant le pourcentage calculé à partir de la population de l'État. (le nombre total divisé par la population)

```

# On ajoute donc le nombre de personne infectées/décédées
data <- data %>%
  mutate(`confirmed total %` = `confirmed total` / population,
        `deaths total %` = `deaths total` / population)

```

On va s'intéresser à une région pour ne pas avoir beaucoup de graphiques, donc par exemple la région du **Midwest**.

Pour les États de la région du Midwest :

On va donc tracer les graphiques des États pour chaque région du Midwest.

```

# nombre absolu de cas inféctés
p11 <- data %>% filter(region == "Midwest") %>% distinct() %>%
  ggplot(aes(x = date,
              y = `confirmed total`,
              group = state ,
              color = state)) +
  geom_line(show.legend = F) +
  xlab("Date") +
  ylab("Nombre total de cas inféctés")

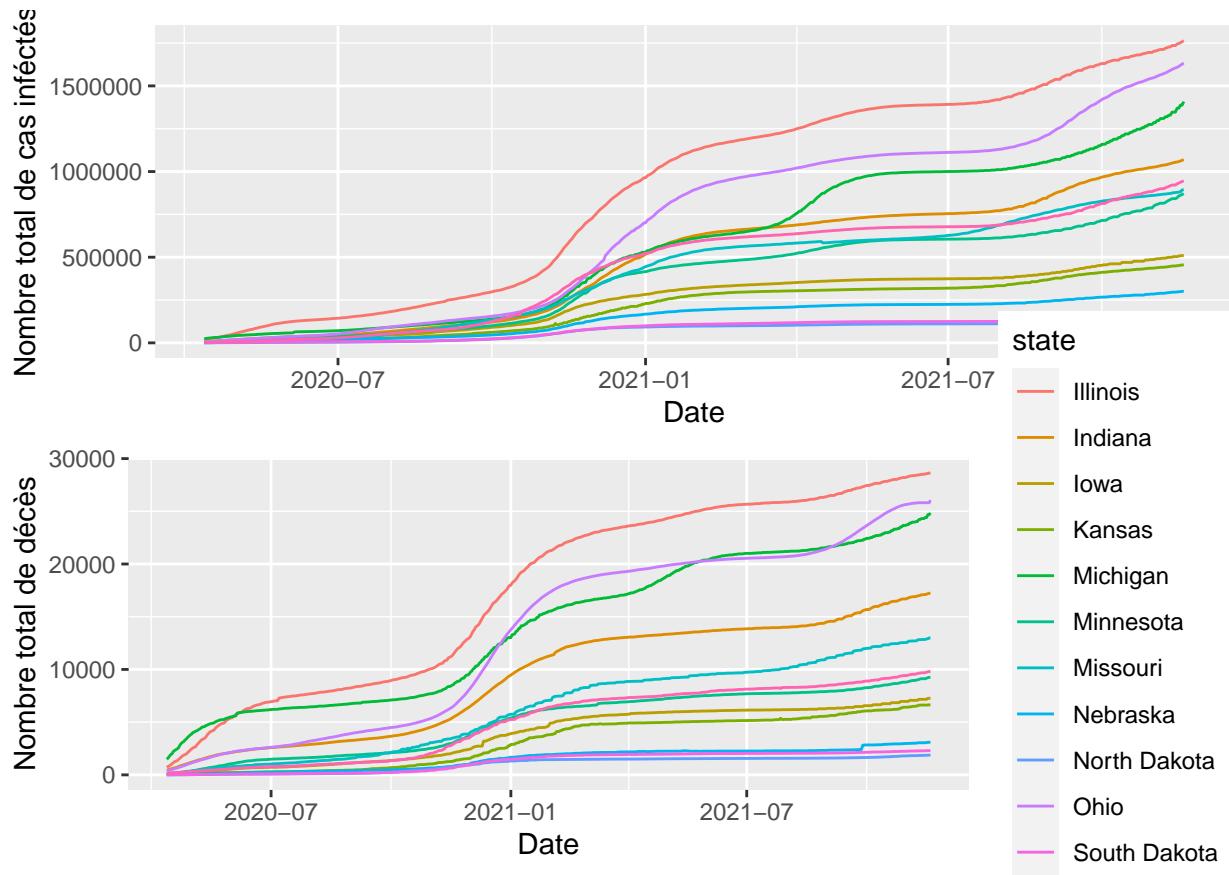
# nombre relatif de cas inféctés
p21 <- data %>% filter(region == "Midwest") %>% distinct() %>%
  ggplot(aes(x = date,
              y = `confirmed total %`,
              group = state,
              color = state)) +
  geom_line(show.legend = F) +
  xlab("Date") +
  ylab("% de cas confirmés au total")

# Nombre absolu de décès
p12 <- data %>% filter(region == "Midwest") %>% distinct() %>%
  ggplot(aes(x = date,
              y = `deaths total` ,
              group = state,
              color = state)) +
  geom_line() +
  xlab("Date") +
  ylab("Nombre total de décès")

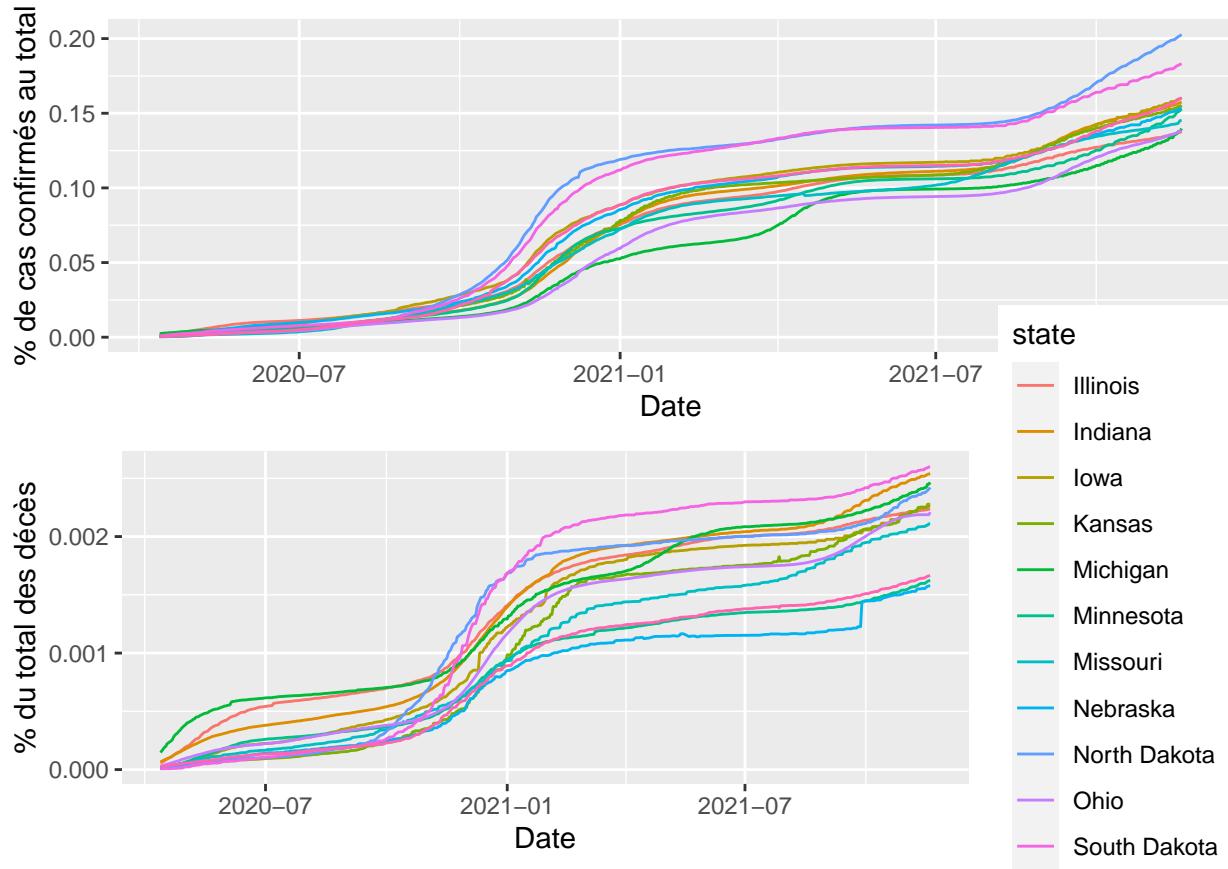
# nombre relatif de décès
p22 <- data %>% filter(region == "Midwest") %>% distinct() %>%
  ggplot(aes(x = date,
              y = `deaths total %` ,
              group = state,
              color = state)) +
  geom_line() +
  xlab("Date") +
  ylab("% du total des décès")

#p11;p12;p21;p22
#cowplot::plot_grid(p11, p12, p21, p22)
gridExtra::grid.arrange(p11, p12, nrow = 2)

```



```
gridExtra::grid.arrange(p21, p22, nrow = 2)
```



Avec % qui signifie le pourcentage... (% de cas confirmés = pourcentage de cas confirmés)

On fait de même avec les autres régions. Même code que précédemment mais on change "Midewest" par la nouvelle région. (**South** par exemple)

Pour tous les États-Unis :

On va maintenant tracer un graphique à barres pour le nombre total relatif de personnes infectées/décédées au fil du temps au niveau des États-Unis.

Graphique à barres

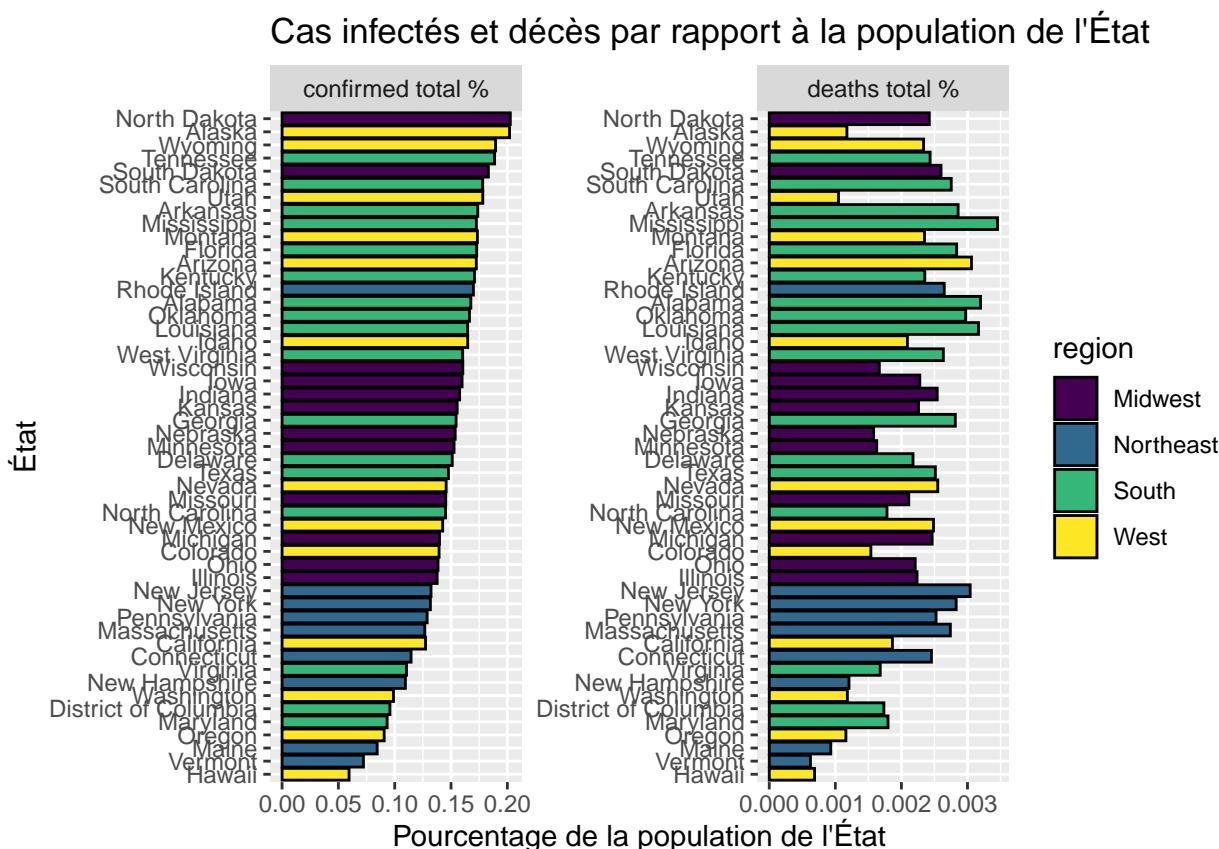
Pour les nombres relatifs :

```
# Graphique à barres - Nombre relatifs
data %>%
  # la date la plus récente
  filter(date == max.date) %>%
  # colonnes qui nous intéressent
  select(region, state, `confirmed total %`, `deaths total %`) %>%
  # transformation des données au format longer
  pivot_longer(cols = c("confirmed total %", "deaths total %"),
               names_to = "count",
               values_to = "value") %>%
  # on trie les états
  group_by(state) %%
```

```

mutate(tot_percentage = sum(value)) %>%
ungroup() %>%
arrange(tot_percentage, state) %>%
mutate(state = as.factor(state),
       state = fct_inorder(state)) %>%
# création du plot
ggplot(aes(y = state,
            x = value,
            fill = region)) +
geom_col(color = "black") +
facet_wrap(count ~ .,
           scales = "free") +
xlab("Pourcentage de la population de l'État") +
ylab("État") +
ggtitle("Cas infectés et décès par rapport à la population de l'État") +
scale_fill_viridis_d()

```



Maintenant on fait pareil pour le nombre total absolu de personnes infectées/décédées au fil du temps au niveau des États-Unis.

Pour les nombres absolus :

```

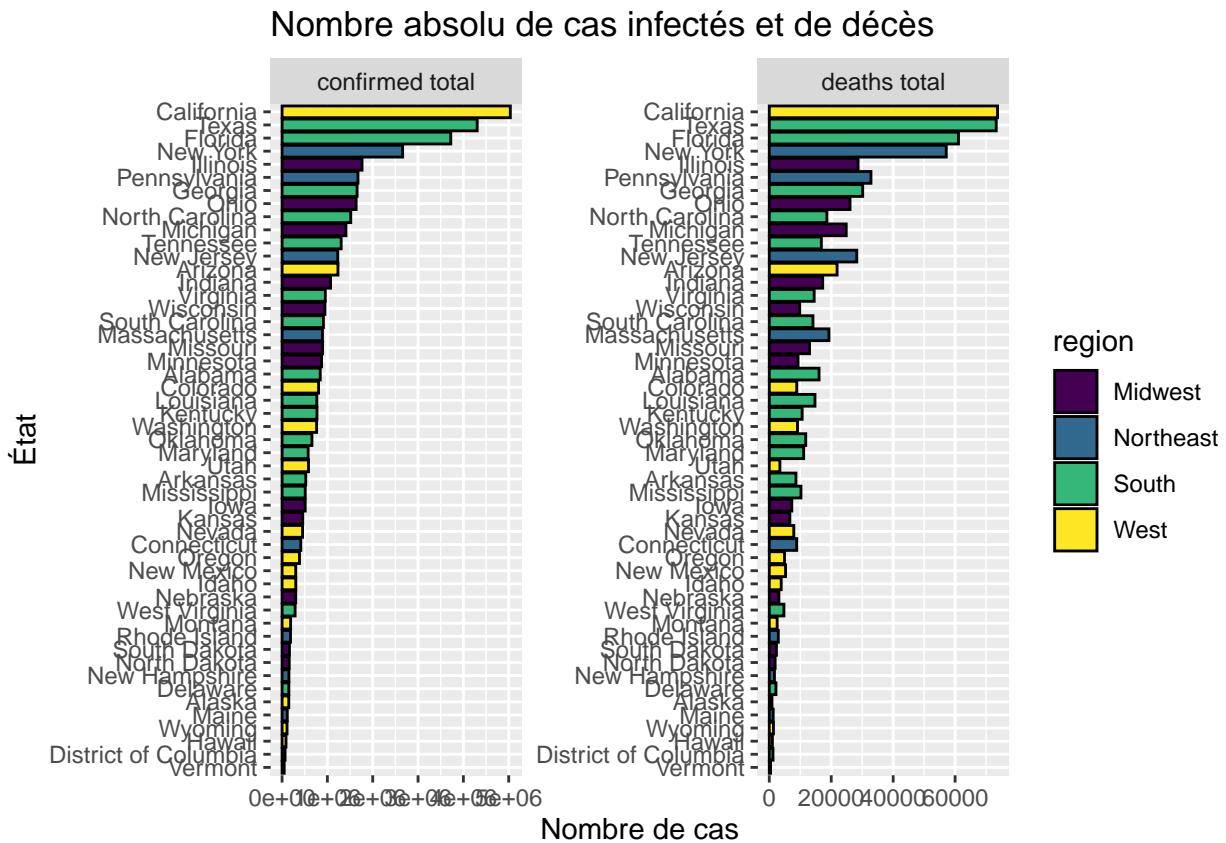
# Graphique à barres - Nombre absolu
data %>%
  # la date la plus récente
  filter(date == max.date) %>%
  # colonnes qui nous intéressent

```

```

select(region, state, `confirmed total`, `deaths total`) %>%
# transformation des données au format long
pivot_longer(cols = c("confirmed total", "deaths total"),
             names_to = "count",
             values_to = "value") %>%
# on trie les états
group_by(state) %>%
mutate(tot_percentage = sum(value)) %>%
ungroup() %>%
arrange(tot_percentage, state) %>%
mutate(state = as.factor(state),
       state = fct_inorder(state)) %>%
# création du plot
ggplot(aes(y = state,
           x = value,
           fill = region)) +
geom_col(color = "black") +
facet_wrap(count ~ .,
           scales = "free") +
xlab("Nombre de cas") +
ylab("État") +
ggtitle("Nombre absolu de cas infectés et de décès") +
scale_fill_viridis_d()

```



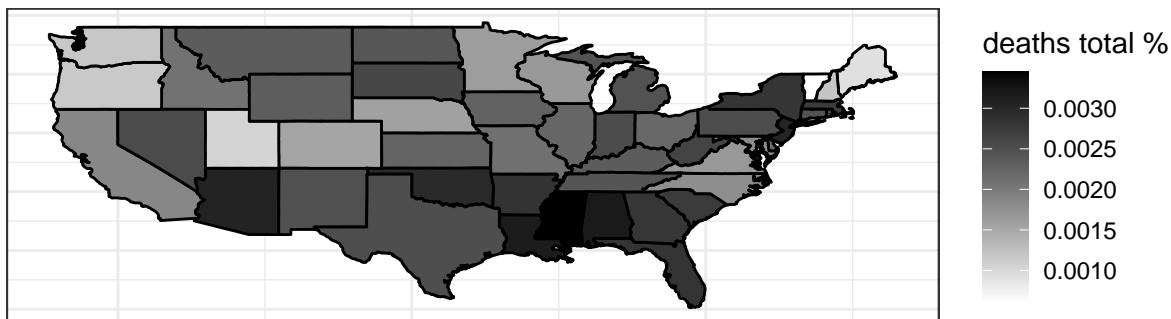
Graphique Map

```
# création de la map - nombre relatifs
p1 <- data %>%
  filter(date == max.date) %>%
  # on sélectionne les colonnes qui nous intéressent
  select(region, state, `confirmed total %`, `deaths total %`) %>%
  # convertir les noms d'états en minuscules
  mutate(state_ = tolower(state)) %>%
  # obtenir la longitude et la latitude
  left_join(x = .,
            y = map_data("state"),
            by = c("state_" = "region")) %>%
  ggplot(aes(x = long, y = lat,
             group = group)) +
  geom_polygon(aes(fill = `deaths total %`),
               color = "black") +
  xlab("") +
  ylab("") +
  ggtitle("Pourcentage de décès par rapport à la population de l'État") +
  scale_fill_gradient(low = "white", high = "black") +
  theme_bw() +
  theme(axis.ticks = element_blank(),
        axis.text = element_blank())

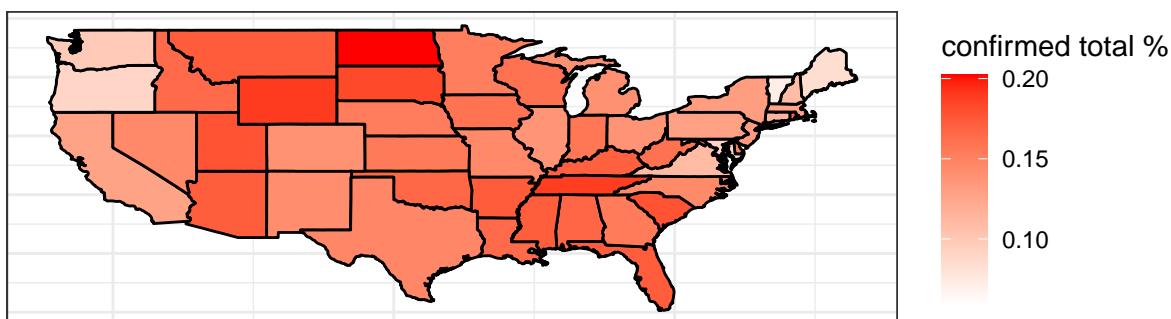
p2 <- data %>%
  filter(date == max.date) %>%
  # on sélectionne les colonnes qui nous intéressent
  select(region, state, `confirmed total %`, `deaths total %`) %>%
  # convertir les noms d'états en minuscules
  mutate(state_ = tolower(state)) %>%
  # obtenir la longitude et la latitude
  left_join(x = .,
            y = map_data("state"),
            by = c("state_" = "region")) %>%
  ggplot(aes(x = long, y = lat,
             group = group)) +
  geom_polygon(aes(fill = `confirmed total %`),
               color = "black") +
  xlab("") +
  ylab("") +
  ggtitle("Pourcentage de personnes infectées par rapport à la population de l'État") +
  scale_fill_gradient(low = "white", high = "red") +
  theme_bw() +
  theme(axis.ticks = element_blank(),
        axis.text = element_blank())

cowplot::plot_grid(p1, p2, nrow = 2)
```

Pourcentage de décès par rapport à la population de l'État



Pourcentage de personnes infectées par rapport à la population de l'État



3 - Afficher sur la carte comment le nombre de cas de COVID a changé au fil du temps

On va créer des aperçus mensuels (tous les 30 jours un aperçu), ensuite on va afficher les subplots pour chaque aperçus mensuel en montrant le nombre de cas de COVID pour chaque État.

```
data <- data %>%
  arrange(state, date) %>%
  # on ajoute un id. de date pour chaque état
  group_by(state) %>%
  mutate(date_id = row_number()) %>%
  ungroup() %>%
  # ajouter un indicateur de l'aperçu de date pour chaque 30ème date
  # inclure la première date
  mutate(`date each 30d` = case_when(date_id == 1 ~ TRUE,
                                      # inclure la dernière date
                                      date == max.date ~ TRUE,
                                      # inclure la 30ème date
                                      date_id %% 30 == 0 ~ TRUE,
                                      T ~ FALSE))
```

Nombre total de cas au fil du temps

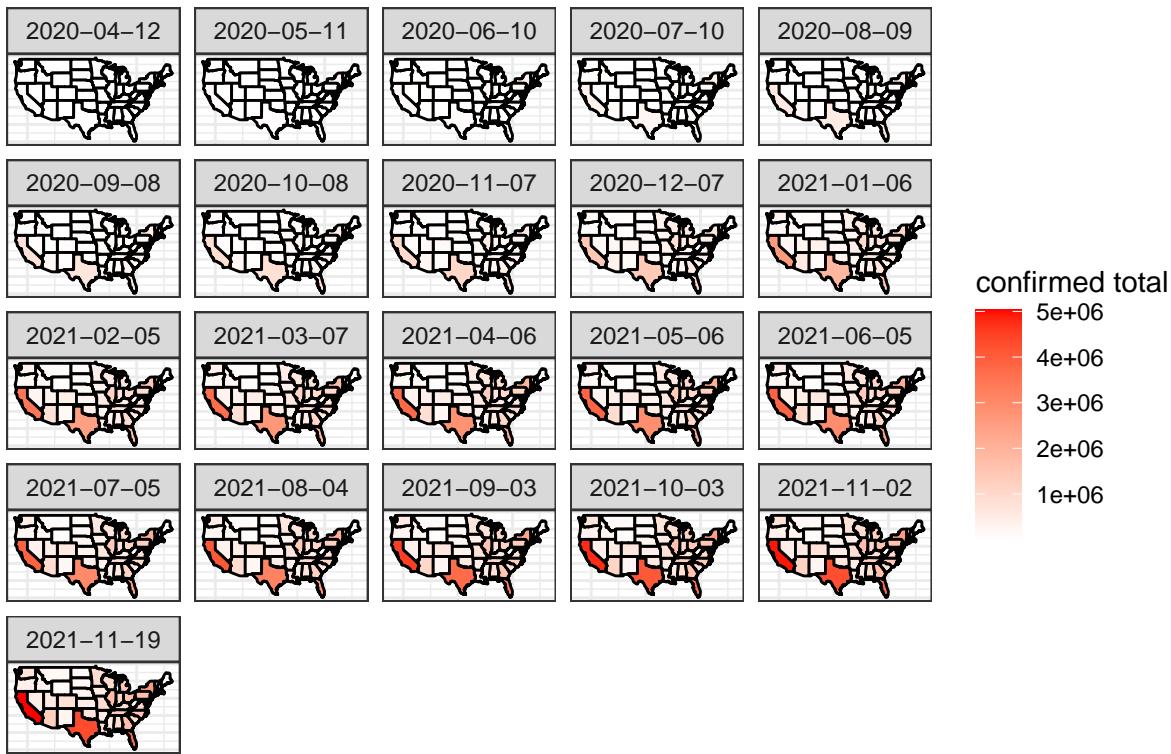
```
# création de la map
data %>%
  filter(`date each 30d`) %>%
```

```

# sélection des colonnes qui nous intéressent
select(region, state, date, `confirmed total`) %>%
# convertir les noms d'états en minuscules
mutate(state_ = tolower(state)) %>%
# obtenir la longitude et la latitude
left_join(x = .,
          y = map_data("state"),
          by = c("state_" = "region")) %>%
ggplot(aes(x = long, y = lat,
           group = group)) +
geom_polygon(aes(fill = `confirmed total`),
             color = "black") +
facet_wrap(. ~ date) +
xlab("") +
ylab("") +
ggtitle("Nombre total de cas au fil du temps") +
scale_fill_gradient(low = "white", high = "red") +
theme_bw() +
theme(axis.ticks = element_blank(),
      axis.text = element_blank())

```

Nombre total de cas au fil du temps



4 - Afficher sur la carte comment le nombre total de doses de vaccination a augmenté au fil du temps

On va créer des aperçus mensuels (tous les 30 jours un aperçu), ensuite on va afficher les subplots pour chaque aperçus mensuels en montrant le nombre total de doses de vaccin pour chaque État.

```

# création de la map
data %>%
  filter(`date each 30d`) %>%
  # sélection des colonnes qui nous intéressent
  select(region, state, date, `vaccine doses total`) %>%
  # convertir les noms d'états en minuscules
  mutate(state_ = tolower(state)) %>%
  # obtenir la longitude et la latitude
  left_join(x = .,
            y = map_data("state"),
            by = c("state_" = "region")) %>%
  ggplot(aes(x = long, y = lat,
             group = group)) +
  geom_polygon(aes(fill = `vaccine doses total`),
               color = "black") +
  facet_wrap(. ~ date) +
  xlab("") +
  ylab("") +
  ggtitle("Nombre de doses de vaccin au fil du temps") +
  scale_fill_gradient(low = "white", high = "green") +
  theme_bw() +
  theme(axis.ticks = element_blank(),
        axis.text = element_blank())

```

Nombre de doses de vaccin au fil du temps

