Data Science - COVID-19

Carlos Utrilla Guerrero

3/19/2020

COVID-19 Pandemia. On going FAIR data science pipeline.

This is an R Markdown document. It is intented to publicy illustrate data from Johns Hopkins University (https://github.com/CSSEGISandData/COVID-19) and FAIR (https://www.go-fair.org/fair-principles/) data science.

```
# This is an analysis report of the Novel Coronavirus (COVID-19)
# Aim for data processing, visualisation and statstics
# Source code: http://yanchang.rdatamining.com/
# set directory
# Data Source: 2019 Data Repository https://github.com/CSSEGISandData/COVID-19
# R Packages:
library(magrittr) # pipline operations
library(lubridate) # date operation
```

```
## Warning: package 'lubridate' was built under R version 3.6.3
```

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

```
## The following object is masked from 'package:base':
##
##     date
```

```
library(tidyverse) # data science pips
```

```
## Warning: package 'tidyverse' was built under R version 3.6.3
```

```
## -- Attaching packages ----------------------------------------------------------------- tidyv
erse 1.3.0 --
```

```
## v ggplot2 3.3.0     v purrr   0.3.3
## v tibble  2.1.3     v dplyr   0.8.4
## v tidyr   1.0.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0
```

```
## Warning: package 'tibble' was built under R version 3.6.2
## Warning: package 'tidyr' was built under R version 3.6.3
```

```
## Warning: package 'purrr' was built under R version 3.6.3
```

```
## Warning: package 'dplyr' was built under R version 3.6.3
```

```
## Warning: package 'forcats' was built under R version 3.6.3
```

```
## -- Conflicts ------------------------------------------------------------------ tidyverse_c
onflicts() --
## x lubridate::as.difftime() masks base::as.difftime()
## x lubridate::date()        masks base::date()
## x tidyr::extract()         masks magrittr::extract()
## x dplyr::filter()          masks stats::filter()
## x lubridate::intersect()   masks base::intersect()
## x dplyr::lag()             masks stats::lag()
## x purrr::set_names()       masks magrittr::set_names()
## x lubridate::setdiff()     masks base::setdiff()
## x lubridate::union()       masks base::union()
```

```
library(gridExtra) # grid based plots
```

```
## Warning: package 'gridExtra' was built under R version 3.6.3
```

```
##
## Attaching package: 'gridExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
library(kableExtra) # build HTML and LaTeX tables
```

```
## Warning: package 'kableExtra' was built under R version 3.6.3
```

```
##
## Attaching package: 'kableExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     group_rows
```

```
library(dplyr)

# Loading data
# At first, three CSV files, are downloaded and saved as local files
# and then loaded into R

# source data files
filenames <- c('time_series_19-covid-Confirmed.csv',
               'time_series_19-covid-Deaths.csv',
               'time_series_19-covid-Recovered.csv')
url.path <- paste0('https://raw.githubusercontent.com/CSSEGISandData/COVID-19/',
                   'master/csse_covid_19_data/csse_covid_19_time_series/')

#download files to local folder
download <- function(filename) {
  url <- file.path(url.path, filename)
  dest <- file.path('./data', filename)
  download.file(url, dest)
}
bin <- lapply(filenames, download)


# load data into R
data.confirmed <- read.csv('./data/time_series_19-covid-Confirmed.csv')
data.deaths <- read.csv('./data/time_series_19-covid-Deaths.csv')
data.recovered <- read.csv('./data/time_series_19-covid-Recovered.csv')

# check dimension of data confirmed
dim(data.confirmed)
```

```
## [1] 462  61
```

```
# Table:
data.confirmed[1:10, 1:10] %>%
kable(booktabs = T, caption = 'Raw Data (Confirmed, First 10 Cols') %>%
kable_styling(font_size = 6, latex_options = c('striped','hold_position','repeat_header'))
```

Raw Data (Confirmed, First 10 Cols

Province.State	Country.Region	Lat	Long	X1.22.20	X1.23.20	X1.24.20	X1.25.20	X1.26.20	X1.27.20
	Thailand	15.0000	101.0000	2	3	5	7	8	8
	Japan	36.0000	138.0000	2	1	2	2	4	4
	Singapore	1.2833	103.8333	0	1	3	3	4	5
	Nepal	28.1667	84.2500	0	0	0	1	1	1
	Malaysia	2.5000	112.5000	0	0	0	3	4	4
British Columbia	Canada	49.2827	-123.1207	0	0	0	0	0	0
New South Wales	Australia	-33.8688	151.2093	0	0	0	0	3	4
Victoria	Australia	-37.8136	144.9631	0	0	0	0	1	1

Province.State	Country.Region	Lat	Long	X1.22.20	X1.23.20	X1.24.20	X1.25.20	X1.26.20	X1.27.20
Queensland	Australia	-28.0167	153.4000	0	0	0	0	0	0
	Cambodia	11.5500	104.9167	0	0	0	0	0	1

```
# check time frame of the data
n.col <- ncol(data.confirmed) # 58 variables
# get dates from column names
dates <- names(data.confirmed)[5:n.col] %>% substr(2,8) %>% mdy()
range(dates)
```

```
## [1] "2020-01-22" "2020-03-18"
```

```
min.date <- min(dates)
max.date <- max(dates)
# last update on 16 March 2020 max.date

# Data Preparation steps:
# 1.From wide to long format
# 2.Aggregate by country
# 3. merge into a signe dataset
# cleaning and transformation
cleanData <- function(data) {
  ## remove some columns
  data %<>% select(-c(Province.State, Lat, Long)) %>% rename(country=Country.Region)
  ## convert from wide to long format
  data %<>% gather(key=date, value=count, -country)
  ## convert from character to date
  data %<>% mutate(date = date %>% substr(2,8) %>% mdy())
  ## aggregate by country
  data %<>% group_by(country, date) %>% summarise(count=sum(count)) %>% as.data.frame()
  return(data)
}
# clean the three datasets
data.confirmed %<>% cleanData() %>% rename(confirmed=count)
data.deaths %<>% cleanData() %>% rename(deaths=count)
data.recovered %<>% cleanData() %>% rename(recovered=count)

# merge above 3 datasets into one, by country and date
data <- data.confirmed %>% merge(data.deaths) %>% merge(data.recovered)

# countries/regions with confirmed cases (excl cruise ships)
countries <- data %>% pull(country) %>% setdiff('Cruise Ship')

# first 10 records when it first broke out in China
data %>% filter(country =='China')%>% head(10)
```

```
##    country       date confirmed deaths recovered
## 1    China 2020-01-22       548     17        28
## 2    China 2020-01-23       643     18        30
## 3    China 2020-01-24       920     26        36
## 4    China 2020-01-25      1406     42        39
## 5    China 2020-01-26      2075     56        49
## 6    China 2020-01-27      2877     82        58
## 7    China 2020-01-28      5509    131       101
## 8    China 2020-01-29      6087    133       120
## 9    China 2020-01-30      8141    171       135
## 10   China 2020-01-31      9802    213       214
```

```
## Cases for the Whole World
# counts for worldwide
data.world <- data %>% group_by(date) %>%
  summarise(country='World',
            confirmed=sum(confirmed),
            deaths=sum(deaths),
            recovered=sum(recovered))

data %<>% rbind(data.world)

# remaining confirmed cases
data %<>% mutate(remaining.confirmed = confirmed - deaths - recovered)

# Daily Increases and Death Rates

# rate.upper = total deaths and recovered cases
# rate.lower = total deaths and confirmed cases
# expected death rate is to be between above rates
# rate.daily =daily deaths and recovered cases

## sort by country and date
data %<>% arrange(country,date)
# daily increases of deaths and recovered cases
# set NA to increase on day1
n <- nrow(data)
day1 <- min(data$date) # set NA day1
data %<>% mutate(confirmed.inc=ifelse(date ==day1,NA, confirmed - lag(confirmed, n=1)),
                 deaths.inc=ifelse(date ==day1,NA,deaths - lag(deaths, n=1)),
                 recovered.inc=ifelse(date ==day1,NA,recovered - lag(recovered, n=1)))

# death rate base on total deaths and recovered cases
data %<>% mutate(rate.upper = (100 *deaths / (deaths + recovered)) %>% round(1))
# lower bound: death rate based on total confirmed cases
data %<>% mutate(rate.lower = (100 * deaths / confirmed) %>% round(1))
# death rate based on number ƒ death/recovered on every single day
data %<>% mutate(rate.daily = (100 * deaths.inc / (deaths.inc + recovered.inc)) %>% round(1))
```

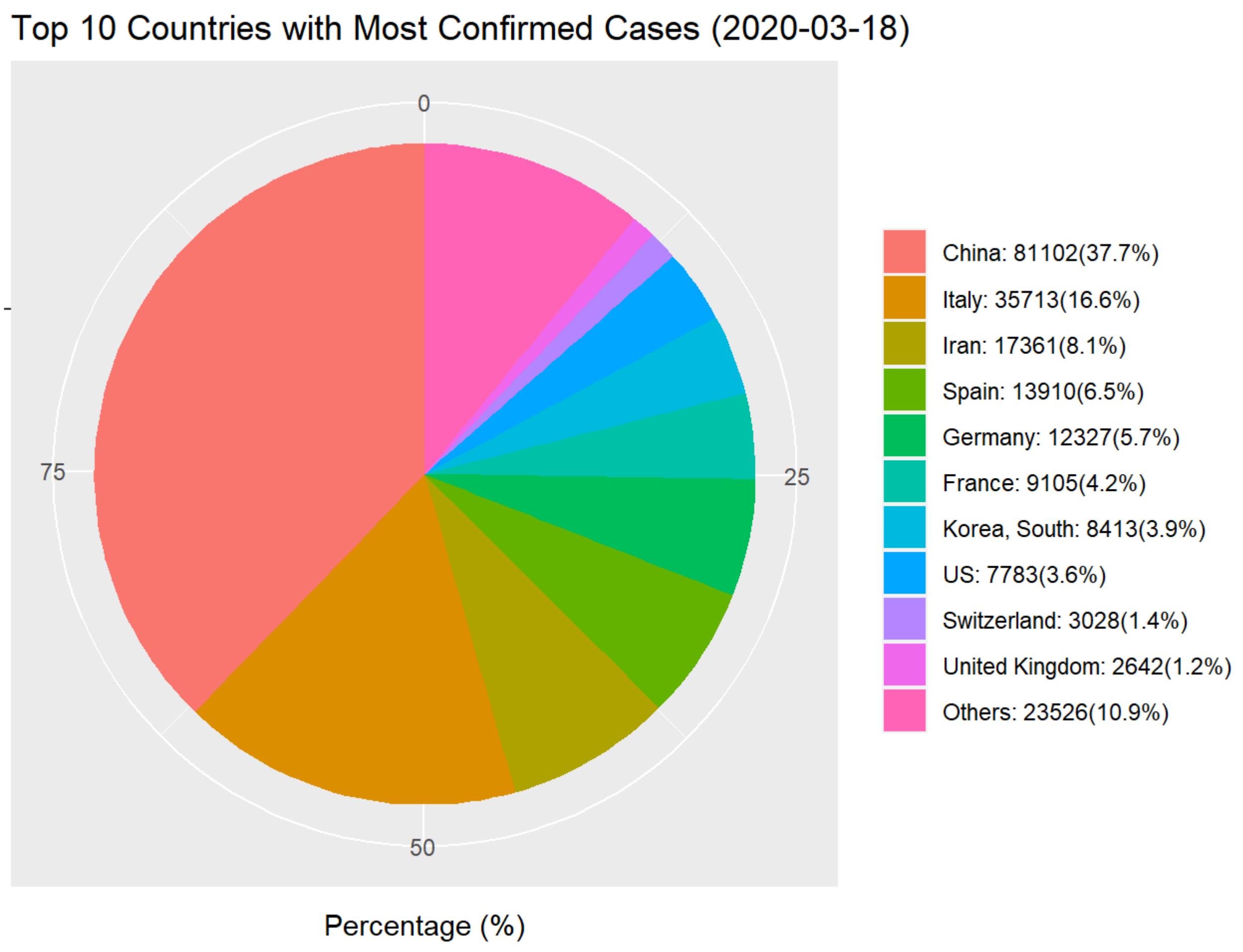
```
# Visualisation
# After preparing the data, we portrait it in various graphs

# TOP Ten Countries
# ranking by confirmed cases
data.latest <- data %>% filter(date ==max(date)) %>%
                                select(country, date, confirmed, deaths, recovered, remaining.co
nfirmed) %>%
                                mutate(ranking = dense_rank(desc(confirmed)))
# top 10 countries incl 11 World
top.countries <- data.latest %>% filter(ranking <= 11) %>%
  arrange(ranking) %>% pull(country) %>% as.character()
top.countries %>% setdiff('World') %>% print()
```

```
##  [1] "China"          "Italy"          "Iran"           "Spain"
##  [5] "Germany"        "France"         "Korea, South"   "US"
##  [9] "Switzerland"    "United Kingdom"
```

```
## add 'Others'
top.countries %<>% c('Others')
## put all others in a single group of 'Others'
df <- data.latest %>% filter(!is.na(country) & country!= 'World')%>%
  mutate(country=ifelse(ranking <= 11, as.character(country), 'Others')) %>%
  mutate(country=country %>% factor(levels = c(top.countries)))
df %<>% group_by(country) %>% summarise(confirmed=sum(confirmed))

# percentage and label
df %<>% mutate(per = (100*confirmed/sum(confirmed)) %>% round(1)) %>%
                 mutate(txt = paste0(country, ': ', confirmed, '(', per, '%)'))
df %>% ggplot(aes(fill=country)) +
  geom_bar(aes(x ='', y = per), stat= 'identity') +
  coord_polar('y', start =0) +
  xlab('') + ylab('Percentage (%)') +
  labs(title=paste0('Top 10 Countries with Most Confirmed Cases (', max.date,')')) +
  scale_fill_discrete(name='Country', labels = df$txt) +
  theme(legend.title = element_blank(), legend.text = element_text((size=7)))
```

```
## Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font family not
## found in Windows font database
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
## family not found in Windows font database
```



```
data.latest %>% filter(country %in% top.countries) %>% select(-c(date, ranking)) %>%
                                                                arrange(desc(confirmed)) %>%
                                                                kable(booktabs=T, row.names=T,
                                                                     caption = paste0('Cases in
Top Ten Countries (', max.date, '). See complete list of all infected countries at the annex A'
),
                                                                       format.args = list(big.m
ark = ',')) %>%
                                                                kable_styling(font_size = 7, lat
ex_options = c('striped', 'hold_position', 'repeat_header'))
```

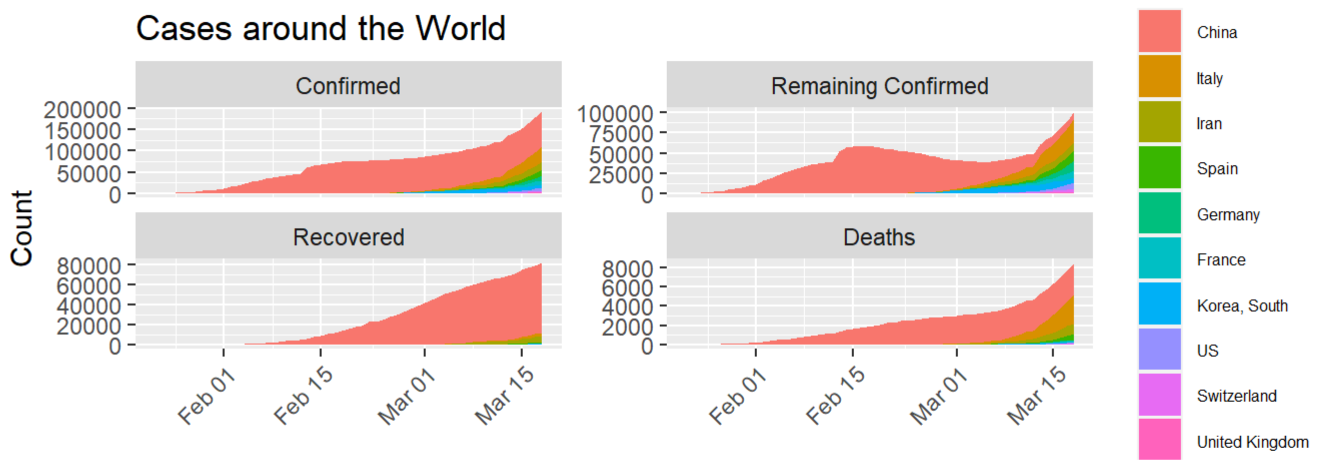
Cases in Top Ten Countries (2020-03-18). See complete list of all infected countries at the annex A

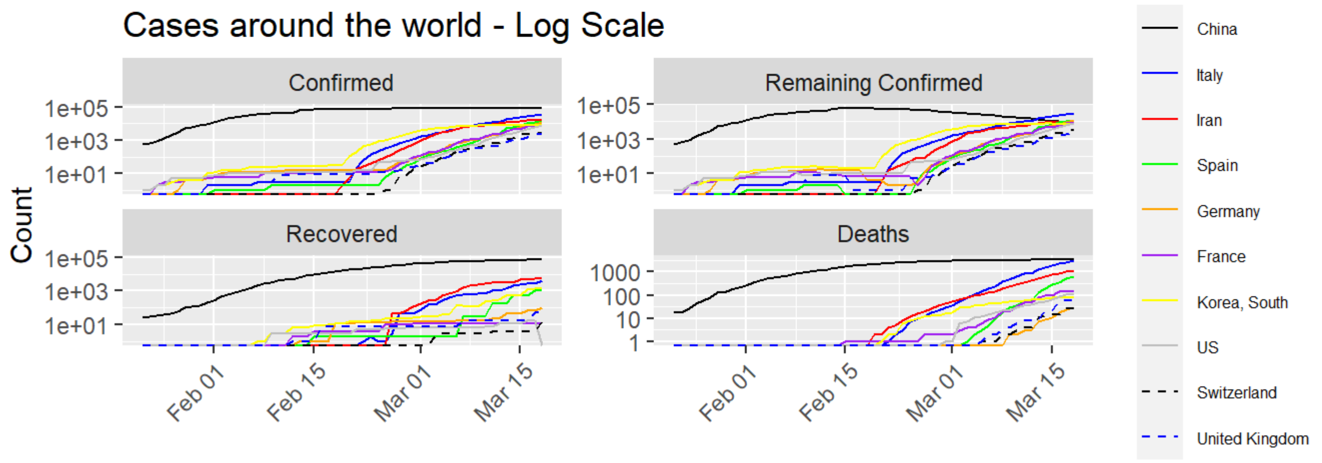
	country	confirmed	deaths	recovered	remaining.confirmed
1	World	214,910	8,733	83,207	122,970
2	China	81,102	3,241	69,755	8,106
3	Italy	35,713	2,978	4,025	28,710
4	Iran	17,361	1,135	5,389	10,837
5	Spain	13,910	623	1,081	12,206
6	Germany	12,327	28	105	12,194
7	France	9,105	148	12	8,945

	country	confirmed	deaths	recovered	remaining.confirmed
8	Korea, South	8,413	84	1,540	6,789
9	US	7,783	118	0	7,665
10	Switzerland	3,028	28	15	2,985
11	United Kingdom	2,642	72	67	2,503

```
 # Comparison across Countries
# convert from wide to long format, for drawing area plot
data.long <- data %>%
  select(c(country, date, confirmed, remaining.confirmed, recovered, deaths)) %>%
  gather(key = type, value = count, -c(country,date))
# set for factor levels to show them in a desirable order
data.long %<>% mutate(type =recode_factor(type, confirmed= 'Confirmed',
                                          remaining.confirmed = 'Remaining Confirmed',
                                          recovered= 'Recovered',
                                          deaths='Deaths'))
# plot cases by type
df <- data.long %>% filter(country %in% top.countries) %<>%
  mutate(country=country %>% factor(levels=c(top.countries)))


### CASES AROUND WORLD
p <- df%>% filter(country !='World') %>%
  ggplot(aes(x=date, y=count)) + xlab('') + ylab('Count') +
  theme(legend.title=element_blank(),
        legend.text = element_text(size=6),
        legend.key.size=unit(0.6, 'cm'),
        axis.text.x=element_text(angle = 45, hjust=1)) +
  facet_wrap(~type, ncol = 2, scale='free_y')
# area plot
plot1 <- p + geom_area(aes(fill=country)) +
  labs(title='Cases around the World')
# line plot and in log scale
linetypes <- rep(c('solid','dashed','dotted'), each=8)
colors <- rep(c('black','blue','red','green','orange', 'purple', 'yellow', 'grey'), 3)
plot2 <- p + geom_line(aes(color=country, linetype=country)) +
  scale_linetype_manual(values = linetypes) +
  scale_color_manual(values = colors) +
  labs(title = 'Cases around the world - Log Scale') +
  scale_y_continuous(trans = 'log10')
# shows two plots together
grid.arrange(plot1, plot2, ncol=1)
```

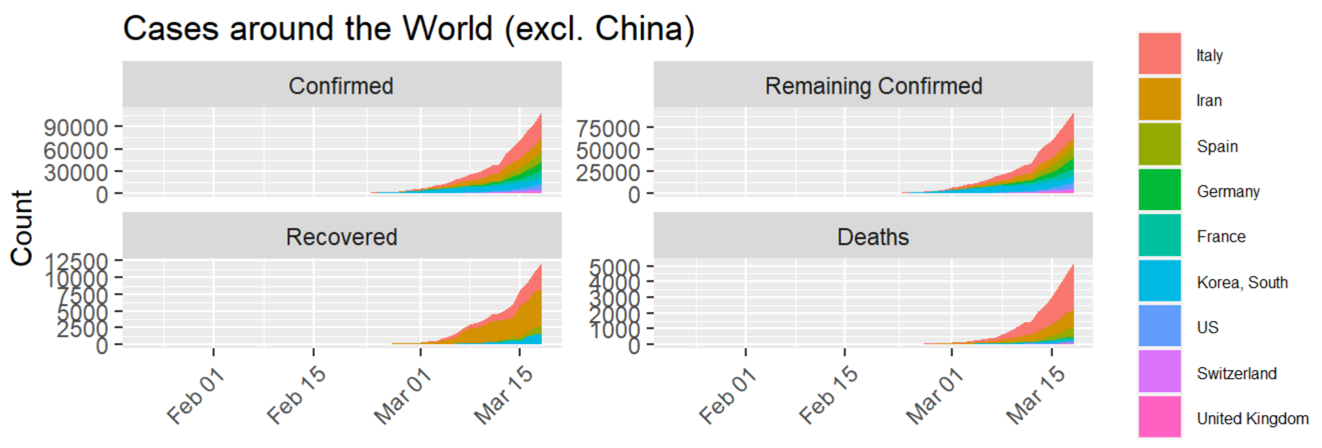
```
## Warning: Transformation introduced infinite values in continuous y-axis
```

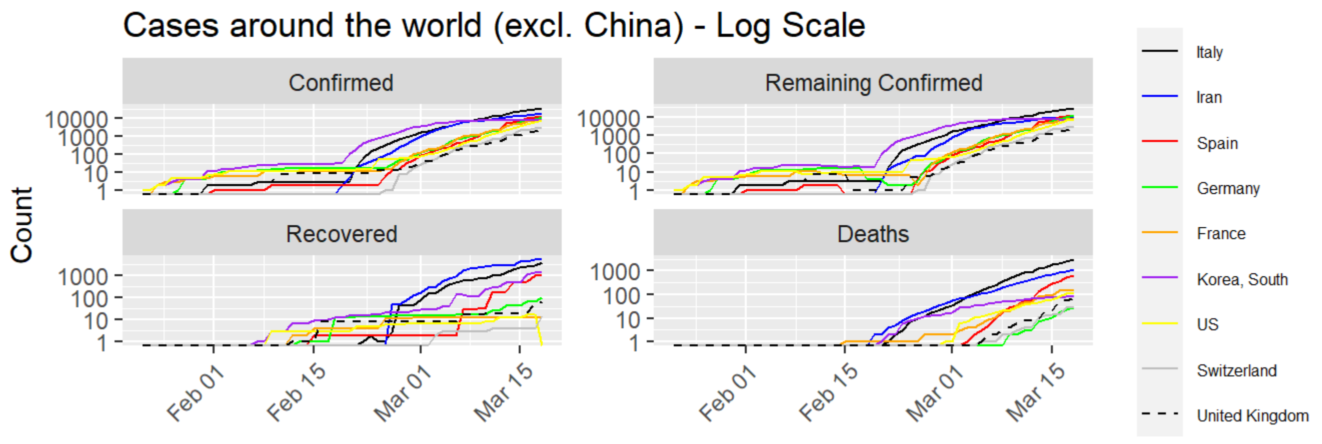




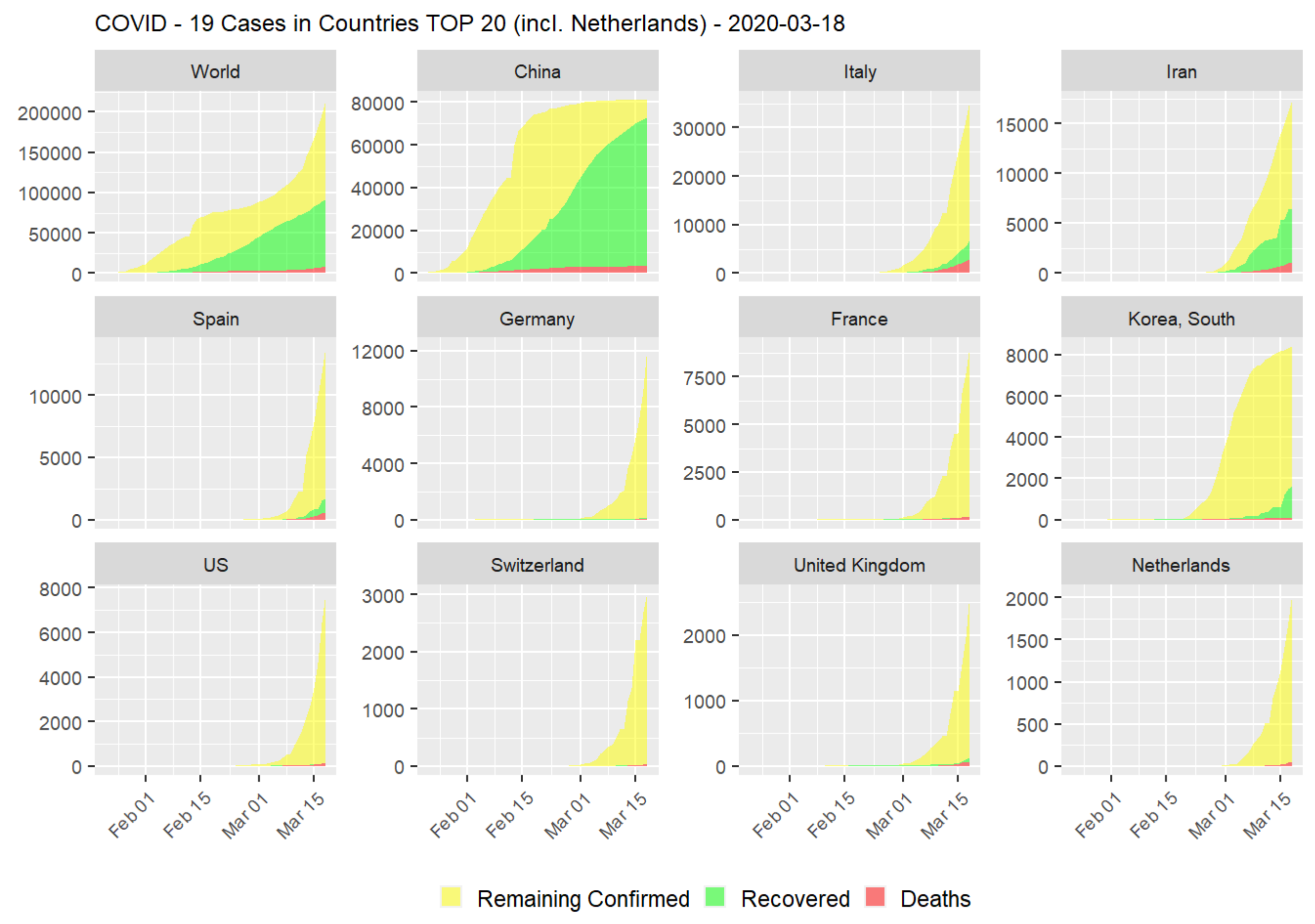
```
# Plot: excluding China
p <- df%>% filter(!(country %in% c('World', 'China'))) %>%
  ggplot(aes(x=date, y=count)) + xlab('') + ylab('Count') +
  theme(legend.title=element_blank(),
        legend.text = element_text(size=6),
        legend.key.size=unit(0.6, 'cm'),
        axis.text.x=element_text(angle = 45, hjust=1)) +
  facet_wrap(~type, ncol = 2, scale='free_y')
# area plot
plot1 <- p + geom_area(aes(fill=country)) +
  labs(title='Cases around the World (excl. China)')
# line plot and in log scale
linetypes <- rep(c('solid','dashed','dotted'), each=8)
colors <- rep(c('black','blue','red','green','orange', 'purple', 'yellow', 'grey'), 3)
plot2 <- p + geom_line(aes(color=country, linetype=country)) +
  scale_linetype_manual(values = linetypes) +
  scale_color_manual(values = colors) +
  labs(title = 'Cases around the world (excl. China) - Log Scale') +
  scale_y_continuous(trans = 'log10')
# shows two plots together
grid.arrange(plot1, plot2, ncol=1)
```

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



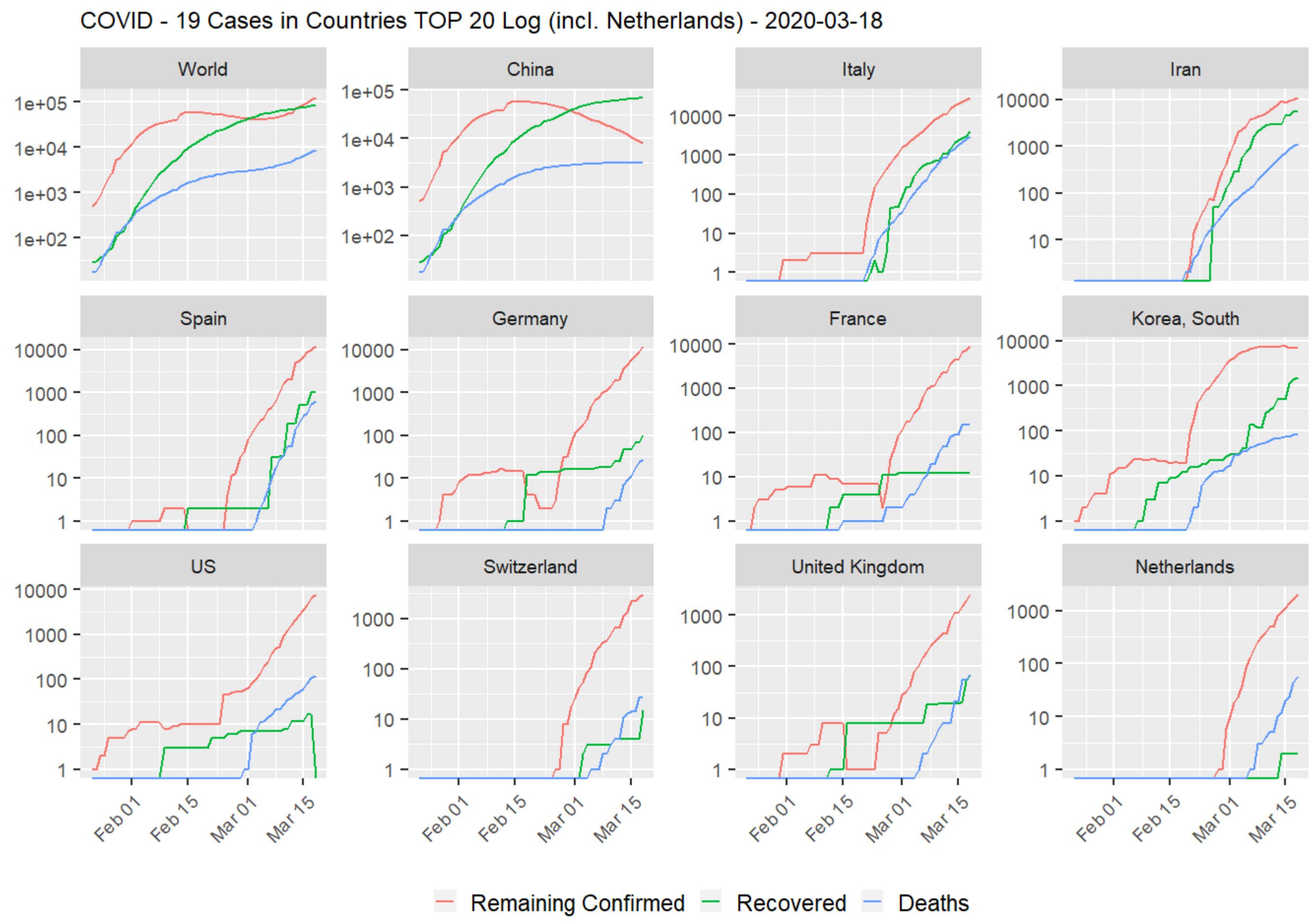


```
# # # list(countries) == 'Netherlands'

## If The Netherland is not top 20, add it in and remove 'Others'
if(!('Netherlands' %in% top.countries)) {
  top.countries %<>% setdiff('Others') %>% c('Netherlands')
  df <- data.long %>% filter(country %in% top.countries) %>%
    mutate(country=country %>% factor(levels = c(top.countries)))
}

# cases by country - area plot
df %>% filter(type != 'Confirmed') %>%
  ggplot(aes(x=date, y=count, fill=type)) +
  geom_area(alpha=0.5) +
  labs(title = paste0('COVID - 19 Cases in Countries TOP 20 (incl. Netherlands) - ', max.date))
 +
  scale_fill_manual(values=c('yellow','green','red')) +
  theme(legend.title=element_blank(), legend.position='bottom',
        plot.title= element_text(size = 9),
        axis.title.x=element_blank(),
        axis.title.y = element_blank(),
        legend.key.size = unit(0.3, 'cm'),
        strip.text.x = element_text(size=7),
        axis.text=element_text(size = 7),
        axis.text.x = element_text(angle=45, hjust=1)) +
  facet_wrap(~country, ncol=4, scale='free_y') #+ scale_y_continuous(trans = 'log10')
```



```
# cases by country - log case
df %>% filter(type != 'Confirmed') %>%
  ggplot(aes(x=date, y=count, color=type)) +
  geom_line() +
  labs(title = paste0('COVID - 19 Cases in Countries TOP 20 Log (incl. Netherlands) - ', max.dat
e)) +
  scale_fill_manual(values=c('red','green','blue')) +
  theme(legend.title=element_blank(), legend.position='bottom',
        plot.title= element_text(size = 9),
        axis.title.x=element_blank(),
        axis.title.y = element_blank(),
        legend.key.size = unit(0.3, 'cm'),
        strip.text.x = element_text(size=7),
        axis.text=element_text(size = 7),
        axis.text.x = element_text(angle=45, hjust=1)) +
  facet_wrap(~country, ncol=4, scale='free_y') + scale_y_continuous(trans = 'log10')
```

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

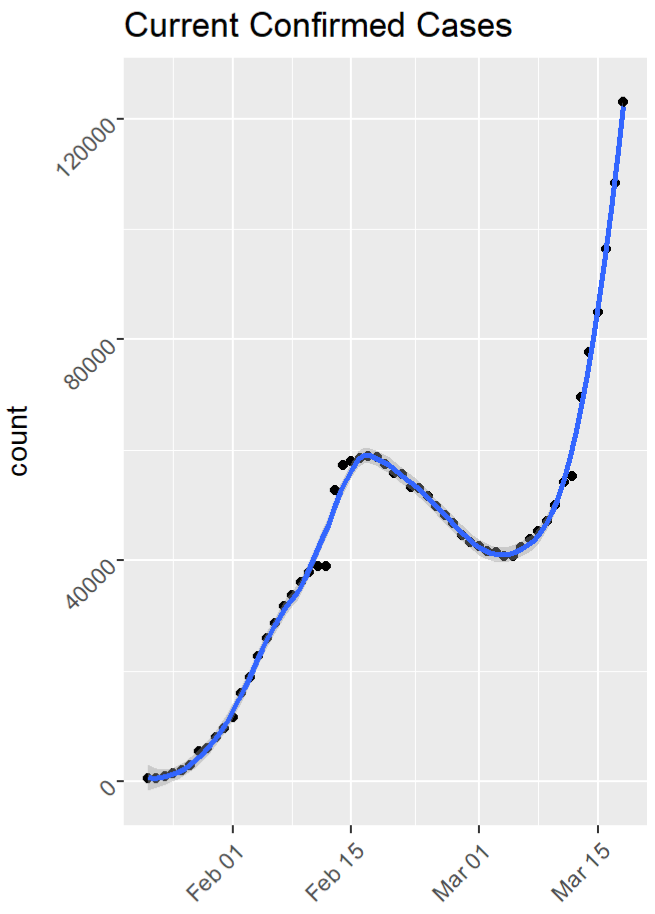


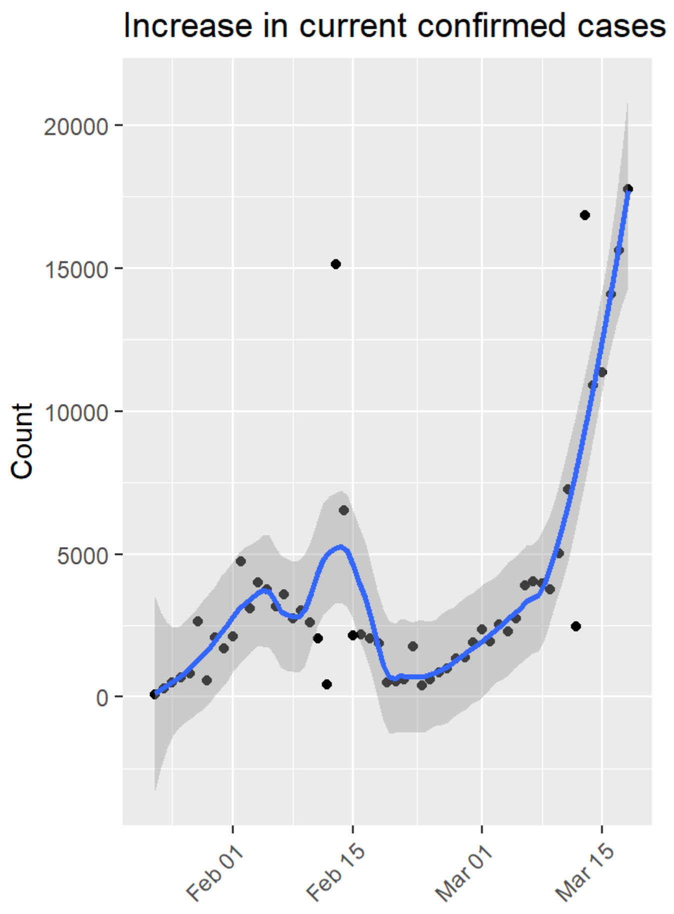
```
### Current confirmed cases:
#data.test <- data %>% filter(country %in% c('Italy', 'Spain', 'Netherlands'))
data %<>% filter(country=='World')
n <- nrow(data)
# current confirmed and it is increase with worldwide case
plot1 <- ggplot(data, aes(x = date, y=remaining.confirmed)) +
  geom_point() + geom_smooth(span=0.3) +
  xlab('') + ylab('count') + labs(title= 'Current Confirmed Cases') +
  theme(axis.text = element_text(angle = 45, hjust=1))
plot2 <- ggplot(data, aes(x =date, y=confirmed.inc)) +
  geom_point() + geom_smooth(span=0.3) +
  xlab('') + ylab('Count') + labs(title= 'Increase in current confirmed cases') +
  theme (axis.text.x = element_text(angle=45, hjust =1))
# show plot 1 and 2 side by side
grid.arrange(plot1, plot2, ncol=2)
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```

```
## Warning: Removed 1 rows containing non-finite values (stat_smooth).
```

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



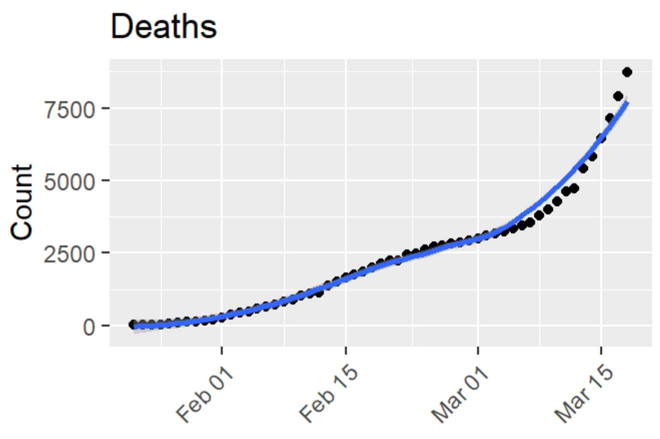


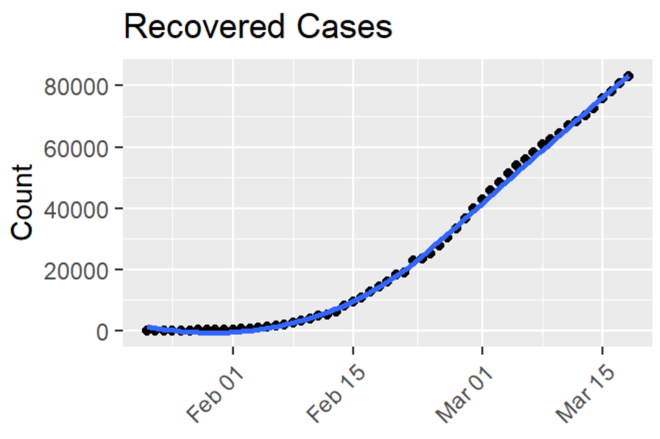
```r
# Deaths and recovery cases
plot1 <- ggplot(data,aes(x=date, y=deaths)) +
  geom_point() + geom_smooth() +
  xlab('') + ylab('Count') + labs(title = 'Deaths') +
  theme(axis.text.x = element_text(angle = 45, hjust=1))
plot2 <- ggplot(data,aes(x=date, y=recovered)) +
  geom_point() + geom_smooth() +
  xlab('') + ylab('Count') + labs(title = 'Recovered Cases') +
  theme(axis.text.x = element_text(angle = 45, hjust=1))
plot3 <- ggplot(data,aes(x=date, y=deaths.inc)) +
  geom_point() + geom_smooth() +
  xlab('') + ylab('Count') + labs(title = 'Increase in Deaths') +
  theme(axis.text.x = element_text(angle = 45, hjust=1))
plot4 <- ggplot(data,aes(x=date, y=recovered.inc)) +
  geom_point() + geom_smooth() +
  xlab('') + ylab('Count') + labs(title = 'Increase Recovered cases') +
  theme(axis.text.x = element_text(angle = 45, hjust=1))
# shows plots together
grid.arrange(plot1, plot2, plot3, plot4, nrow=2)
```

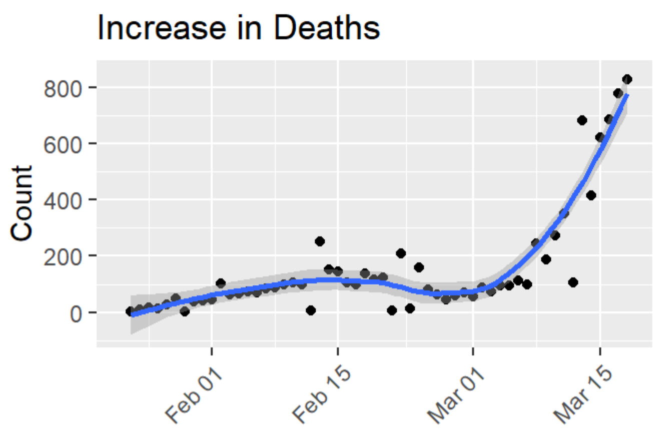
```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
## Warning: Removed 1 rows containing non-finite values (stat_smooth).

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

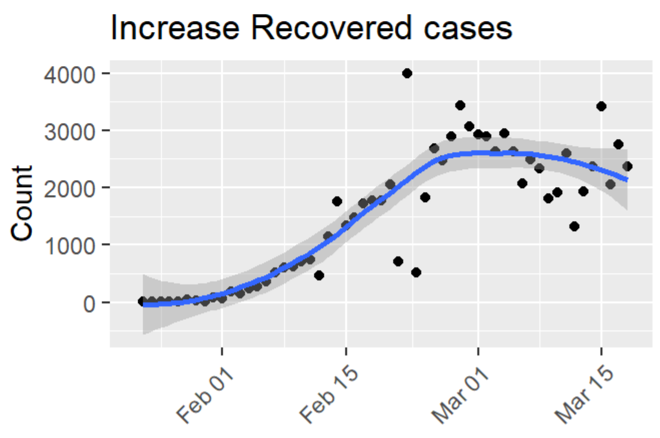
```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```

```
## Warning: Removed 1 rows containing non-finite values (stat_smooth).

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



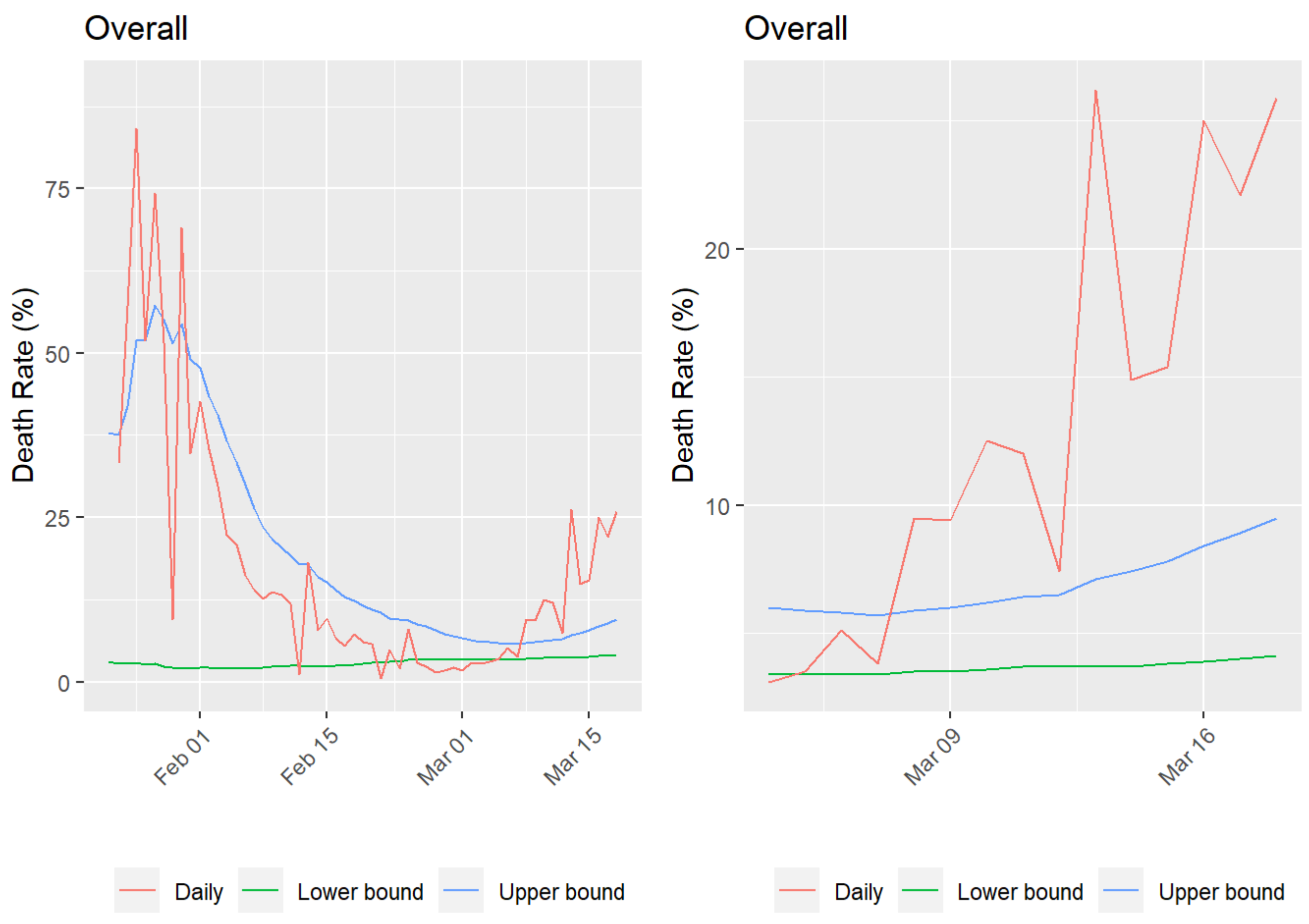






```
## Deaths rates
plot1 <- ggplot(data, aes(x=date)) +
  geom_line(aes(y=rate.upper, colour='Upper bound')) +
  geom_line(aes(y=rate.lower, colour='Lower bound')) +
  geom_line(aes(y=rate.daily, colour='Daily')) +
  xlab('') + ylab('Death Rate (%)') + labs(title='Overall') +
  theme(legend.position='bottom', legend.title=element_blank(),
        axis.text.x=element_text(angle=45, hjust=1)) +
  ylim(0,90)
## insert las two weeks
plot2 <- ggplot(data[n-(14:0),], aes(x=date)) +
  geom_line(aes(y=rate.upper, colour='Upper bound')) +
  geom_line(aes(y=rate.lower, colour='Lower bound')) +
  geom_line(aes(y =rate.daily, colour= 'Daily')) +
  xlab('') + ylab('Death Rate (%)') + labs(title = 'Overall') +
  theme(legend.position='bottom', legend.title =element_blank(),
        axis.text.x=element_text(angle=45, hjust=1))
grid.arrange(plot1, plot2, nrow=1)
```

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



```
# COVID-19 Cases Worldwide

# sort by date, descending order
data %<>% arrange(desc(date)) %>%
  select(c(date, confirmed, deaths, recovered, remaining.confirmed,
           confirmed.inc, deaths.inc, recovered.inc, rate.lower, rate.upper, rate.daily))
```

```
# Latest Cases by Country - TOP confirmed cases
data.latest %>% arrange(desc(confirmed)) %>% select(-c(date)) %>% head(15)
```

```
##           country confirmed deaths recovered remaining.confirmed ranking
## 1           World    214910   8733     83207              122970       1
## 2           China     81102   3241     69755                8106       2
## 3           Italy     35713   2978      4025               28710       3
## 4            Iran     17361   1135      5389               10837       4
## 5           Spain     13910    623      1081               12206       5
## 6         Germany     12327     28       105               12194       6
## 7          France      9105    148        12                8945       7
## 8    Korea, South      8413     84      1540                6789       8
## 9              US      7783    118         0                7665       9
## 10    Switzerland      3028     28        15                2985      10
## 11 United Kingdom      2642     72        67                2503      11
## 12    Netherlands      2058     58         2                1998      12
## 13        Austria      1646      4         9                1633      13
## 14         Norway      1550      6         1                1543      14
## 15        Belgium      1486     14        31                1441      15
```

Note that this is an developing story. Check back for updates.