

COVID-19 patients, deaths, and death rate

22/07/2020

Source of the data

The data is compiled from <https://github.com/CSSEGISandData/COVID-19>.

The python script *script.py* is a modification of <https://github.com/JacopoPan/JHU-2019nCoV-to-pandas-DF>

Here I avoided any data modeling and am just showing the raw number of diagnosed patients, deaths, death rate, new cases by day, and new deaths by day.

Top countries by cases and deaths

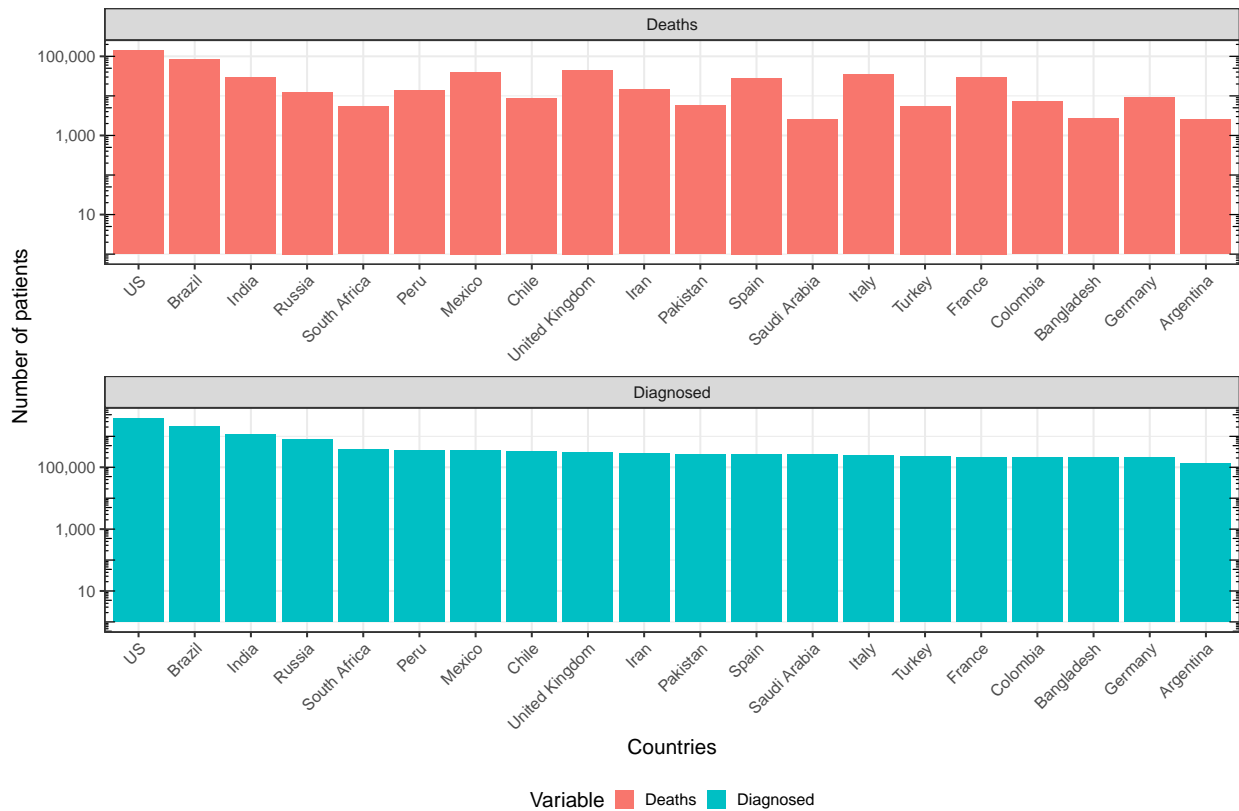
```
deathmat = read.csv("deaths.tsv", header=T,
                    sep="\t", check.names=F, row.names=1)
countdeath = apply(deathmat[nrow(deathmat),1:(ncol(deathmat) - 1)], 2, sum)

casemat = read.csv("confirmed.tsv",
                  header=T, sep="\t", check.names=F, row.names=1)
countcases = apply(casemat[nrow(casemat),1:(ncol(casemat) - 1)], 2, sum)

totaldf = data.frame(Countries=c(names(countdeath), names(countcases)),
                    Values=as.numeric(c(countdeath, countcases)),
                    Variable=c(rep("Deaths", length(countdeath)),
                               rep("Diagnosed", length(countcases))))
select_countries = names(sort(countcases, TRUE))[1:20]

totaldf = totaldf[totaldf$Countries %in% select_countries, ]
totaldf$Countries = factor(totaldf$Countries, levels=select_countries)

P = ggplot(totaldf, aes(x=Countries, y=Values, fill=Variable)) +
  geom_bar(stat="identity", position="dodge") +
  facet_wrap(~Variable, nrow=2, scales="free") +
  theme_bw(base_size=18) +
  scale_y_log10("Number of patients", labels=comma) +
  theme(legend.position="bottom",
        axis.text.x=element_text(angle=45, hjust=1)) +
  annotation_logticks(sides="lr")
plot(P)
```



Calculate doubling rate for each country since June 15th

- Red line will show daily increase of twice.
- Purple line will show daily increase of 10%
- Blue line will show daily increase of 1%

```
get_expected = function(datadf, rate=2, init=100){
  list.data = lapply(unique(datadf$variable), function(country){
    tempdf = datadf[datadf$variable==country, ]
    # idx_st = min(which(tempdf$value >= init))
    idx_st = min(which(tempdf$date >= as.Date("2020-06-15")))
    init_val = tempdf$value[idx_st]
    advals = sapply(1:nrow(tempdf), function(x){
      if(x > idx_st){
        return(init_val * (rate**(x - idx_st)))
      }else{
        return(0)
      }
    })
    ad_df = data.frame(variable=country, Rate=rate,
                      date=tempdf$date, value=advals)
    return(ad_df)
  })
  out_df = do.call("rbind", list.data)
```

```

    return(out_df)
}

```

Total morbidity

```

deathmat = read.csv("deaths.tsv", header=T,
                    sep="\t", check.names=F, row.names=1)
countries = c("China", "Iran", "Italy",
              "US", "Russia", "Spain",
              "France", "Canada", "Germany",
              "Korea, South", "United Kingdom", "Israel")
deathmat = deathmat[,c(countries, "date")]

moltendf = melt(deathmat)

## Using date as id variables

moltendf$date = as.Date(moltendf$date)

doubling_df = get_expected(moltendf, rate=2, init=100)
onepercent_df = get_expected(moltendf, rate=1.01, init=100)
tenpercent_df = get_expected(moltendf, rate=1.1, init=100)

P2 = ggplot(moltendf[moltendf$value > 0,],
            aes(x=date, y=(value),
                group=variable)) +
  geom_line() +
  geom_line(data=doubling_df[doubling_df$value > 0,], colour="red3", linetype=2) +
  geom_line(data=onepercent_df[onepercent_df$value > 0,], colour="blue", linetype=3) +
  geom_line(data=tenpercent_df[tenpercent_df$value > 0,], colour="purple", linetype=4) +
  theme_bw(base_size=14) +
  facet_wrap(~variable) +
  scale_y_log10(
    name="Total deaths caused by SARS-CoV-2", labels=comma, limits=c(1, max(moltendf$value))) +
  xlab("Date as of outbreak") +
  ggtitle(paste("Last update on", Sys.time())) +
  annotation_logticks()

plot(P2)

```

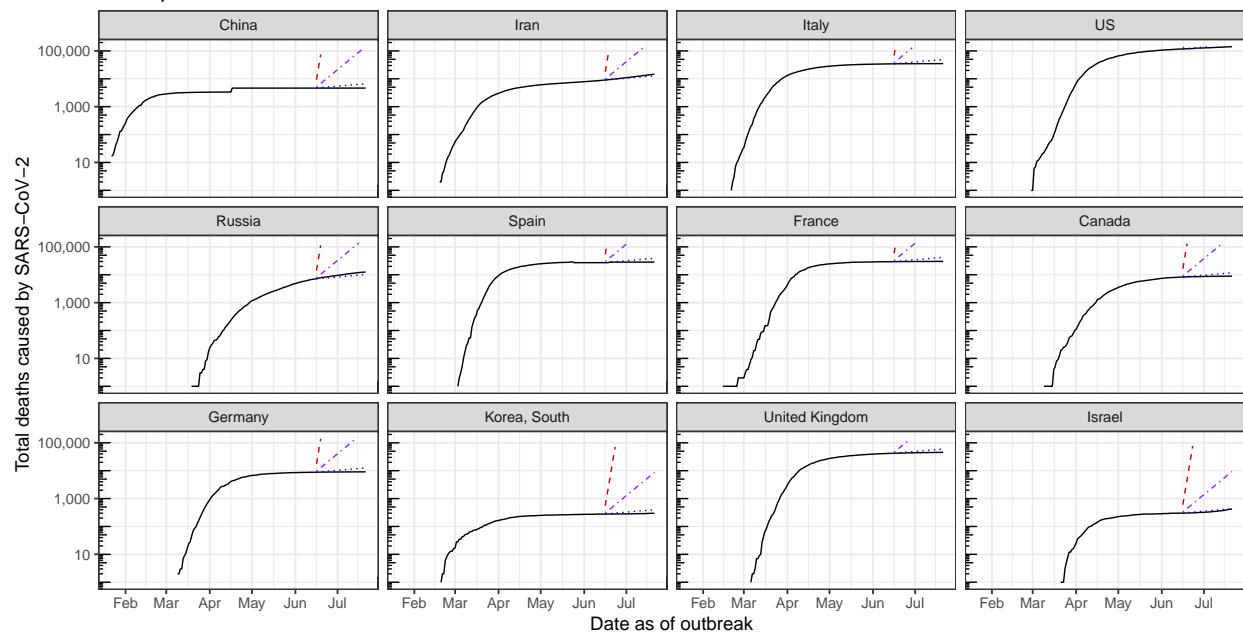
```
## Warning: Removed 390 row(s) containing missing values (geom_path).
```

```
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
```

```
## Warning: Removed 17 row(s) containing missing values (geom_path).
```

```
## Warning: Removed 146 row(s) containing missing values (geom_path).
```

Last update on 2020-07-22 14:34:15



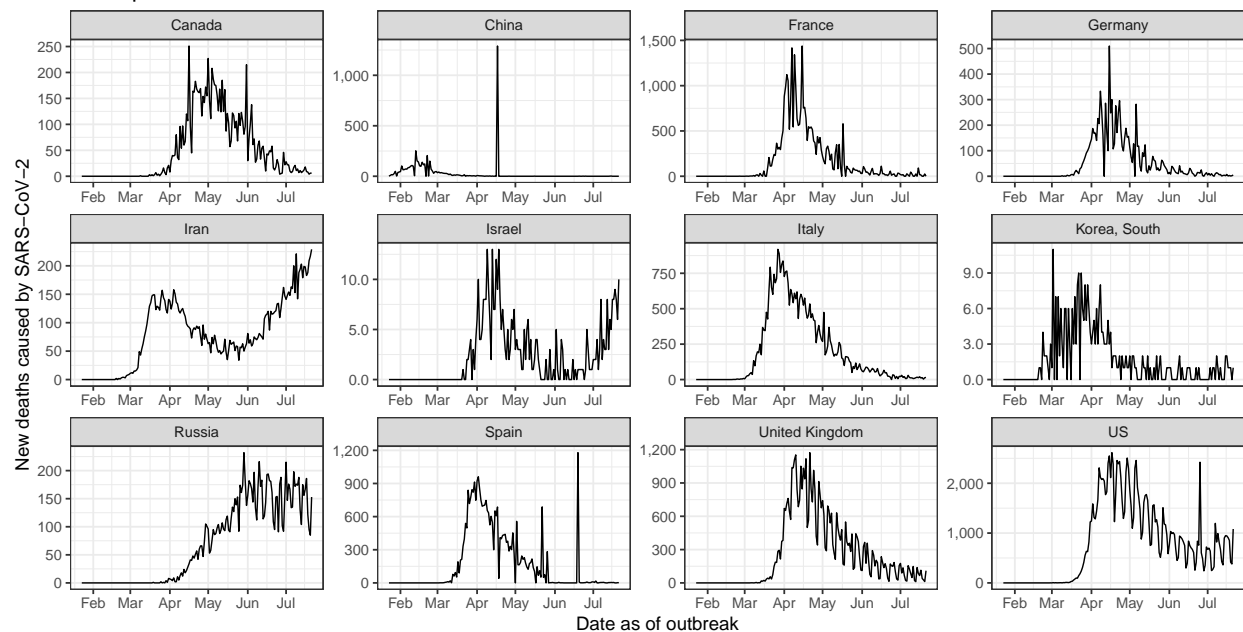
New deaths by day

```
list.data = lapply(colnames(deathmat)[1:(ncol(deathmat)-1)], function(country){
  new_cases = diff(deathmat[,country])
  addf = data.frame(date=deathmat$date[2:nrow(deathmat)],
                    variable=country,
                    value=new_cases)
})
# Some countries (e.g. Spain on 5/25/20) have corrected their stats
# Enforce 0 on the plot
moltendf = do.call("rbind", list.data)
moltendf$value[moltendf$value < 0] = 0

moltendf$date = as.Date(moltendf$date)

P = ggplot(moltendf, aes(x=date, y=value, group=variable)) +
  geom_line() +
  theme_bw(base_size=14) +
  facet_wrap(~variable, scales="free") +
  scale_y_continuous(
    name="New deaths caused by SARS-CoV-2", labels=comma) +
  ggtitle(paste("Last update on", Sys.time())) +
  xlab("Date as of outbreak")
plot(P)
```

Last update on 2020-07-22 14:34:16



Total COVID-19 patients

```
casemat = read.csv("confirmed.tsv",
                  header=T, sep="\t", check.names=F, row.names=1)

casemat = casemat[,c(countries, "date")]

moltendf = melt(casemat)

## Using date as id variables

moltendf$date = as.Date(moltendf$date)

doubling_df = get_expected(moltendf, rate=2, init=100)
onepercent_df = get_expected(moltendf, rate=1.01, init=100)
tenpercent_df = get_expected(moltendf, rate=1.1, init=100)

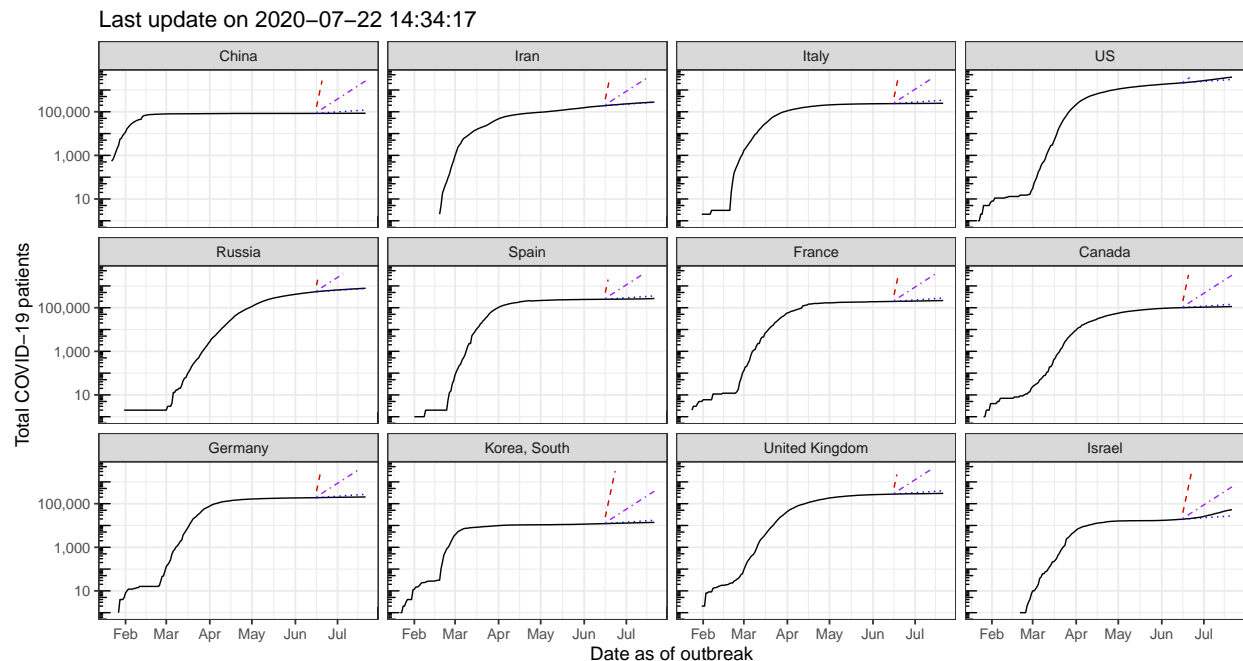
P2 = ggplot(moltendf[moltendf$value > 0, ], aes(x=date, y=(value), group=variable)) +
  geom_line() +
  theme_bw(base_size=14) +
  facet_wrap(~variable) +
  geom_line(data=doubling_df[doubling_df$value > 0,], colour="red3", linetype=2) +
  geom_line(data=onepercent_df[onepercent_df$value > 0,], colour="blue", linetype=3) +
  geom_line(data=tenpercent_df[tenpercent_df$value > 0,], colour="purple", linetype=4) +
  scale_y_log10(
    name="Total COVID-19 patients", labels=comma, limits=c(1, max(moltendf$value))) +
  xlab("Date as of outbreak") +
```

```
ggtitle(paste("Last update on", Sys.time())) +
  annotation_logticks()
```

```
plot(P2)
```

```
## Warning: Removed 383 row(s) containing missing values (geom_path).
```

```
## Warning: Removed 84 row(s) containing missing values (geom_path).
```



New COVID-19 patients by day

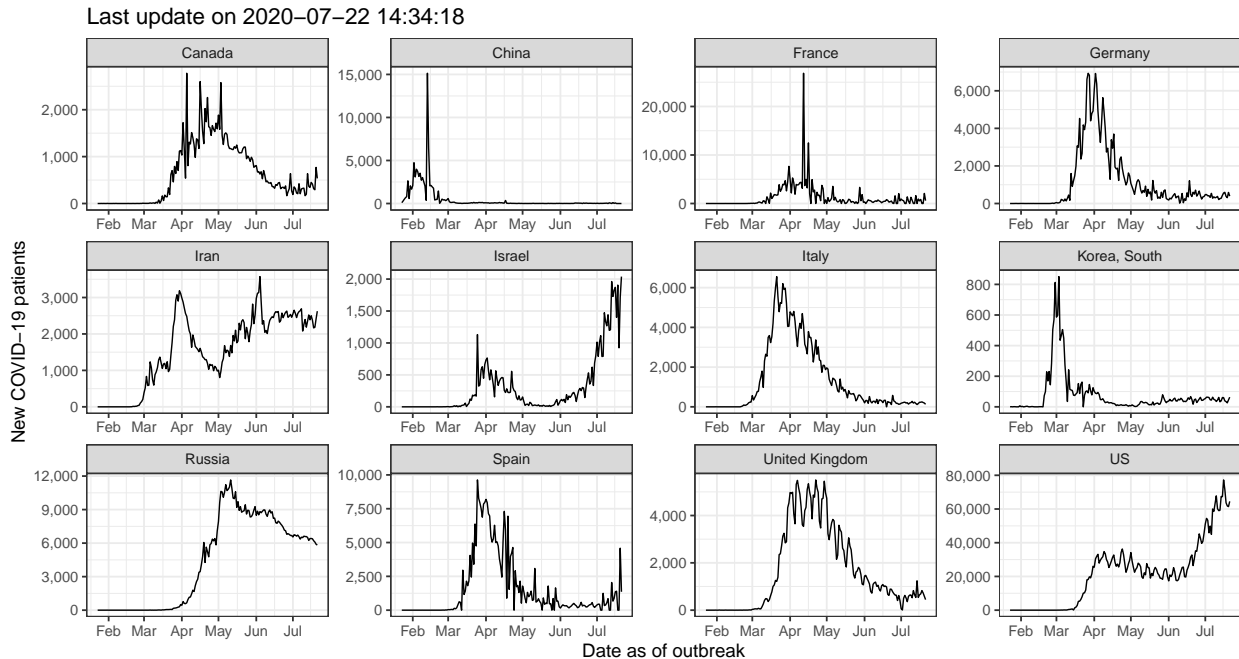
```
list.data = lapply(colnames(casemat)[1:(ncol(casemat)-1)], function(country){
  new_cases = diff(casemat[,country])
  addf = data.frame(date=casemat$date[2:nrow(casemat)],
                    variable=country,
                    value=new_cases)
})
# Some countries (e.g. Spain on 5/25/20) have corrected their stats
# Enforce 0 on the plot
moltendf = do.call("rbind", list.data)
moltendf$value[moltendf$value < 0] = 0
moltendf$date = as.Date(moltendf$date)

P = ggplot(moltendf, aes(x=date, y=value, group=variable)) +
  geom_line() +
  theme_bw(base_size=14) +
  facet_wrap(~variable, scales="free") +
  scale_y_continuous(
```

```

name="New COVID-19 patients", labels=comma) +
ggtitle(paste("Last update on", Sys.time())) +
xlab("Date as of outbreak")
plot(P)

```



PCA of countries by total patients

Principal component analysis identifies non-correlating variables (components) from the data and here can identify outlier countries according to a single variable.

In the following plots, I show the first three principal components for total diagnosed patients by day, total deaths by day, and also death rate by day.

```

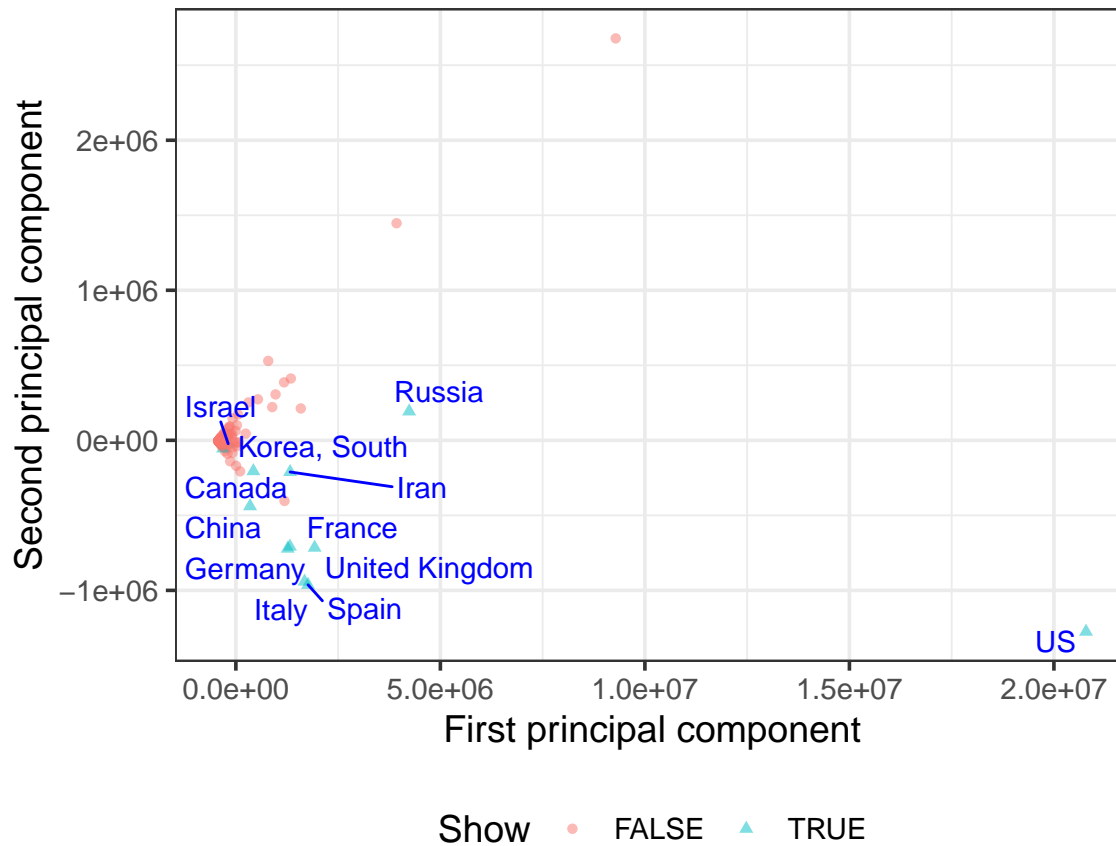
casemat = read.csv("confirmed.tsv", header=T, sep="\t", check.names=F, row.names=1)

pcamat = casemat[,1:(ncol(casemat) - 1)]
PCA = prcomp(t(pcamat))
pcadf = as.data.frame(PCA$x)
pcadf$Country = colnames(casemat)[1:nrow(pcadf)]
pcadf$Show = ifelse(pcadf$Country %in% countries, TRUE, FALSE)

P = ggplot(pcadf, aes(x=PC1, y=PC2)) +
  geom_point(aes(colour=Show, shape=Show), alpha=0.5) +
  geom_text_repel(data=pcadf[pcadf$Show,], aes(label=Country), colour="blue", size=4) +
  theme_bw(base_size=14) +
  xlab("First principal component") +
  ylab("Second principal component") +
  ggtitle(paste("Last update on", Sys.time())) +
  theme(legend.position="bottom")
plot(P)

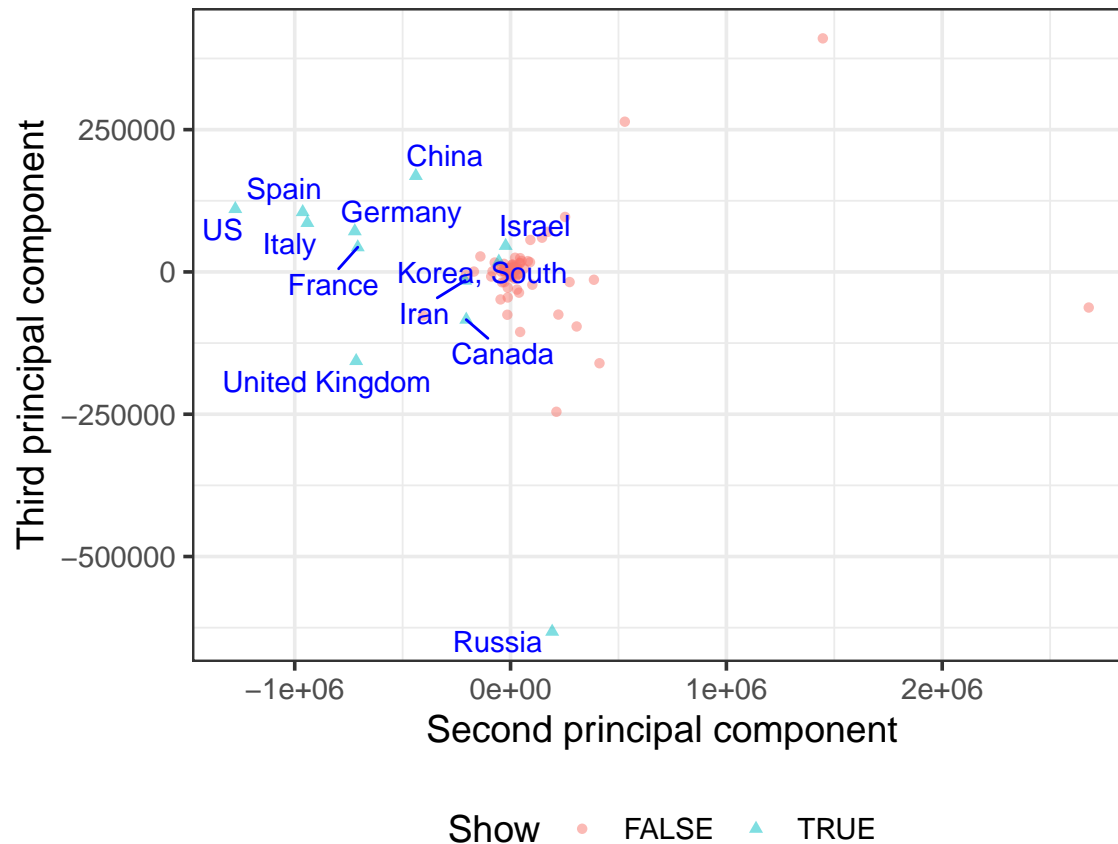
```

Last update on 2020-07-22 14:34:19



```
P = ggplot(pcadf, aes(x=PC2, y=PC3)) +
  geom_point(aes(colour=Show, shape=Show), alpha=0.5) +
  geom_text_repel(data=pcadf[pcadf$Show,], aes(label=Country), colour="blue", size=4) +
  theme_bw(base_size=14) +
  ylab("Third principal component") +
  xlab("Second principal component") +
  ggtitle(paste("Last update on", Sys.time())) +
  theme(legend.position="bottom")
plot(P)
```


Last update on 2020-07-22 14:34:19



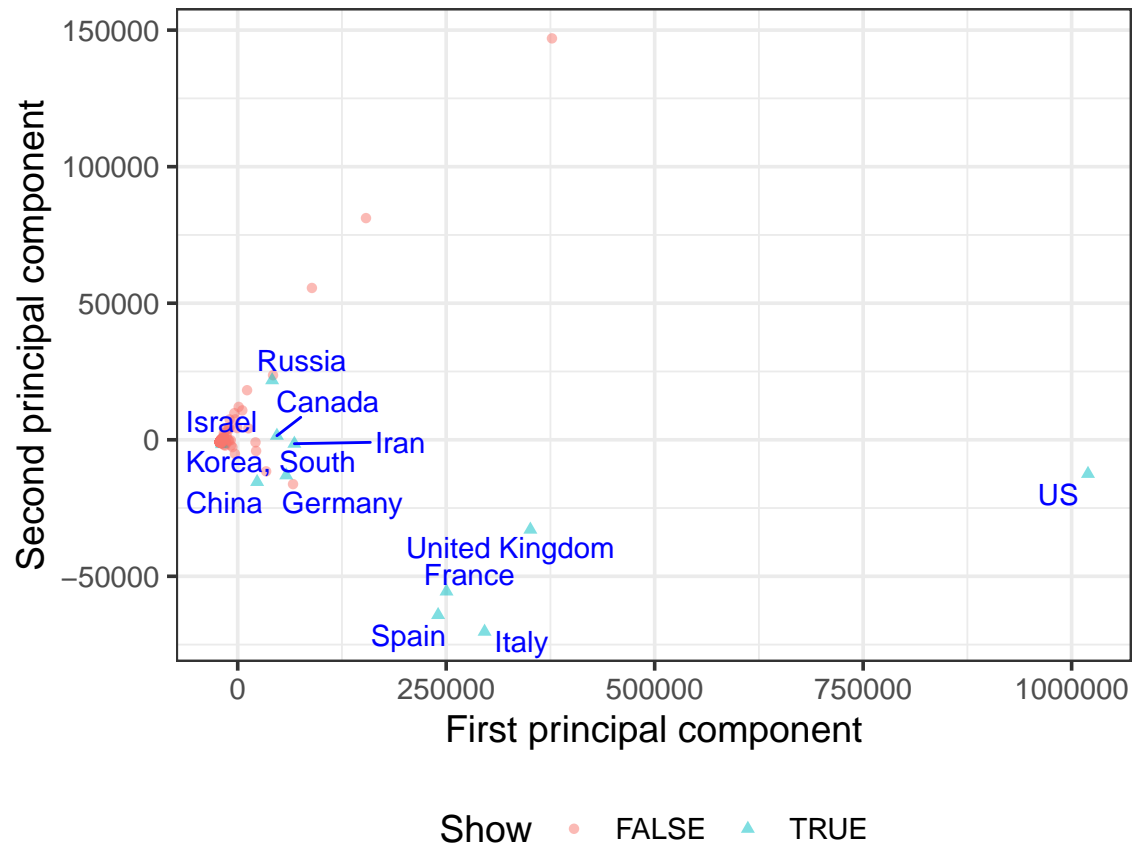
PCA of countries by morbidity

```
deathmat = read.csv("deaths.tsv", header=T,
                    sep="\t", check.names=F, row.names=1)

pcamat = deathmat[,1:(ncol(deathmat) - 1)]
PCA = prcomp(t(pcamat))
pcadf = as.data.frame(PCA$x)
pcadf$Country = colnames(deathmat)[1:nrow(pcadf)]
pcadf$Show = ifelse(pcadf$Country %in% countries, TRUE, FALSE)

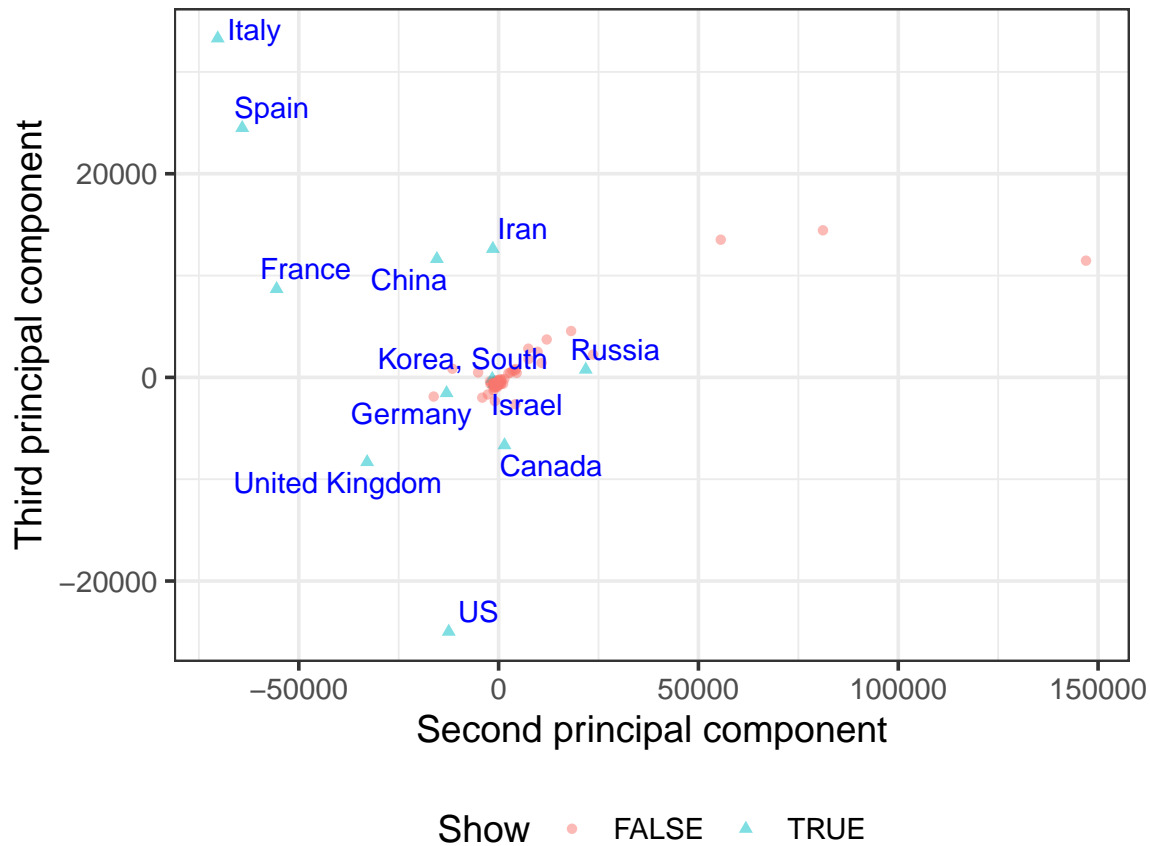
P = ggplot(pcadf, aes(x=PC1, y=PC2)) +
  geom_point(aes(colour=Show, shape=Show), alpha=0.5) +
  geom_text_repel(data=pcadf[pcadf$Show,], aes(label=Country), colour="blue", size=4) +
  theme_bw(base_size=14) +
  xlab("First principal component") +
  ylab("Second principal component") +
  ggtitle(paste("Last update on", Sys.time())) +
  theme(legend.position="bottom")
plot(P)
```

Last update on 2020-07-22 14:34:20



```
P = ggplot(pcadf, aes(x=PC2, y=PC3)) +
  geom_point(aes(colour=Show, shape=Show), alpha=0.5) +
  geom_text_repel(data=pcadf[pcadf$Show,], aes(label=Country), colour="blue", size=4) +
  theme_bw(base_size=14) +
  ylab("Third principal component") +
  xlab("Second principal component") +
  ggtitle(paste("Last update on", Sys.time())) +
  theme(legend.position="bottom")
plot(P)
```

Last update on 2020-07-22 14:34:20



Death rate

The two countries which succeeded in stopping the spread, China and South Korea, provide the most accurate measurement of true death rate. In the death rate plot below, the range of the death rate of these two countries is shown by pink.

```
deathmat = read.csv("deaths.tsv", header=T,
                    sep="\t", check.names=F, row.names=1)
casemat = read.csv("confirmed.tsv", header=T, sep="\t", check.names=F, row.names=1)
deathrate = deathmat[,1:(ncol(deathmat) - 1)] / (0.1 + casemat[,setdiff(colnames(deathmat), "date")])

deathratedf = deathrate
deathratedf$date = casemat$date
moltendf = melt(deathratedf[,c("countries", "date")])
```

Using date as id variables

```
moltendf$date = as.Date(moltendf$date)

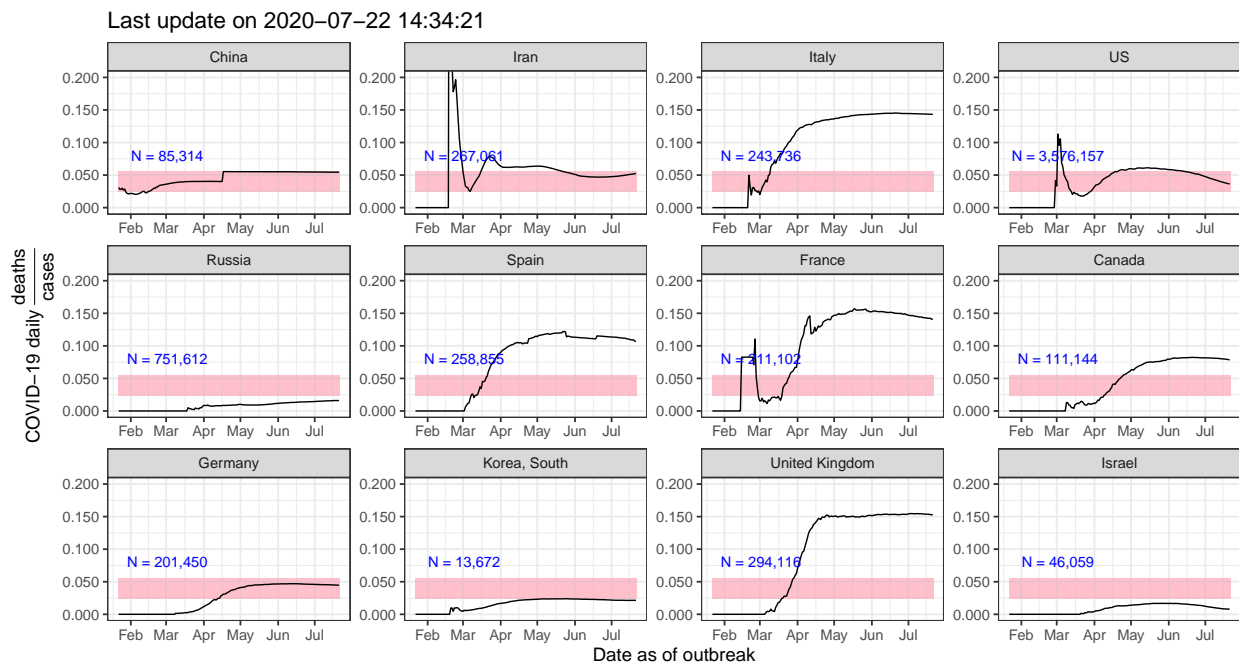
moltendf$Label = NA
deathratedf$date = as.Date(deathratedf$date)
```

```
textmat = casemat[which.max(casemat[, "China"]), ]
textdf = melt(as.data.frame(textmat[, c("countries", "date")]))
```

```
## Using date as id variables
```

```
textdf$date = min(as.Date(deathratedf$date)) + 40
textdf$Label = paste("N =", comma(textdf$value))
textdf$value = 0.08

P = ggplot(moltendf, aes(x=date, y=value, group=variable)) +
  geom_rect(xmin=min(deathratedf$date),
            xmax=max(deathratedf$date+1),
            ymin=max(deathratedf$China),
            ymax=max(deathratedf[, "Korea, South"]),
            alpha=0.05,
            fill="pink") +
  geom_line() +
  geom_text(data=textdf, aes(x=date, y=value, label=Label), colour='blue') +
  theme_bw(base_size=14) +
  facet_wrap(~variable, scales="free") +
  scale_y_continuous(
    name=expression("COVID-19 daily "*frac("deaths", "cases")),
    labels=comma) +
  xlab("Date as of outbreak") +
  ggtitle(paste("Last update on", Sys.time())) +
  coord_cartesian(ylim=c(0, 0.2))
plot(P)
```

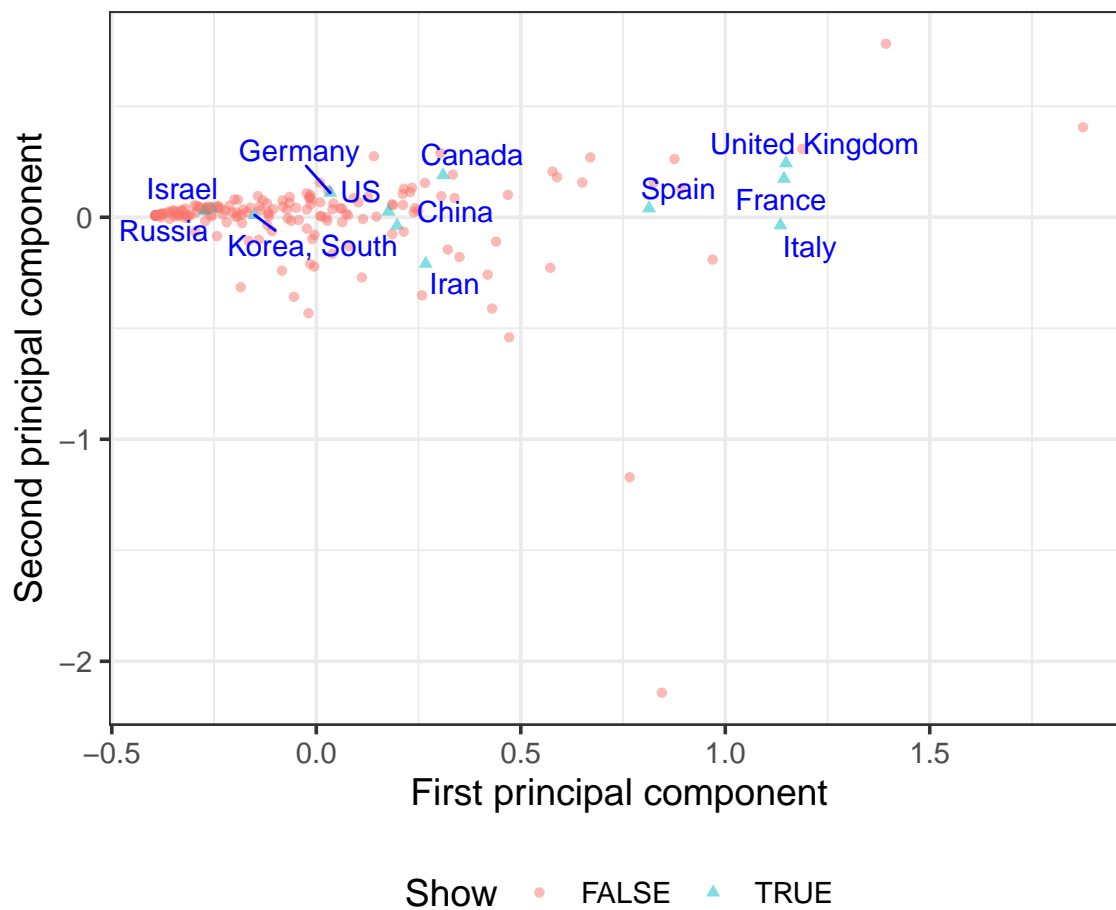


```

pcamat = deathrate
PCA = prcomp(t(pcamat))
pcadf = as.data.frame(PCA$x)
pcadf$Country = colnames(deathmat)[1:nrow(pcadf)]
pcadf$Show = ifelse(pcadf$Country %in% countries, TRUE, FALSE)

P = ggplot(pcadf, aes(x=PC1, y=PC2)) +
  geom_point(aes(colour=Show, shape=Show), alpha=0.5) +
  geom_text_repel(data=pcadf[pcadf$Show,], aes(label=Country), colour="blue", size=4) +
  theme_bw(base_size=14) +
  xlab("First principal component") +
  ylab("Second principal component") +
  theme(legend.position="bottom")
plot(P)

```

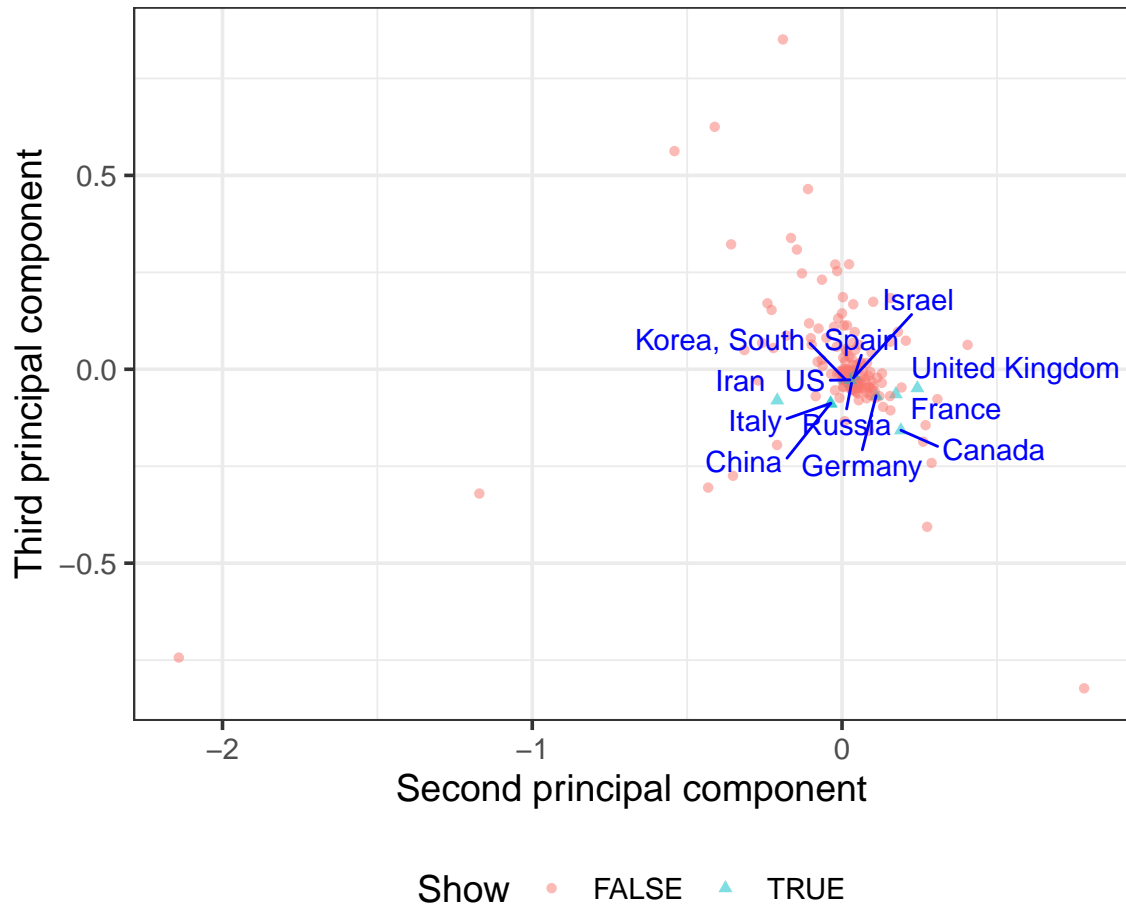


```

P = ggplot(pcadf, aes(x=PC2, y=PC3)) +
  geom_point(aes(colour=Show, shape=Show), alpha=0.5) +
  geom_text_repel(data=pcadf[pcadf$Show,], aes(label=Country), colour="blue", size=4) +
  theme_bw(base_size=14) +
  ylab("Third principal component") +
  xlab("Second principal component") +
  theme(legend.position="bottom")

```

```
plot(P)
```



Canada

```
casepath = "COVID-19/csse_covid_19_data/csse_covid_19_time_series/time_series_covid19_confirmed_global.csv"
casedf = read.csv(casepath, header=T, check.names=F)
casedf = casedf[casedf[,2]=="Canada", c(1, 5:ncol(casedf))]
provinces = casedf[apply(casedf[,2:ncol(casedf)], 1, sum) > 0, 1]
casedf = casedf[casedf[,1] %in% provinces,]
```

```
moltendf = melt(casedf)
```

```
## Using Province/State as id variables
```

```
moltendf$date = as.Date(as.character(moltendf$variable), tryFormats="%m/%d/%y")
```

```
moltendf$variable = moltendf[, "Province/State"]
doubling_df = get_expected(moltendf, rate=2, init=100)
```

```
doubling_df[, "Province/State"] = doubling_df$variable
onepercent_df = get_expected(moltendf, rate=1.01, init=100)
onepercent_df[, "Province/State"] = onepercent_df$variable
tenpercent_df = get_expected(moltendf, rate=1.1, init=100)
tenpercent_df[, "Province/State"] = tenpercent_df$variable
```

```
textdf = melt(casedf[, c(1, ncol(casedf))])
```

```
## Using Province/State as id variables
```

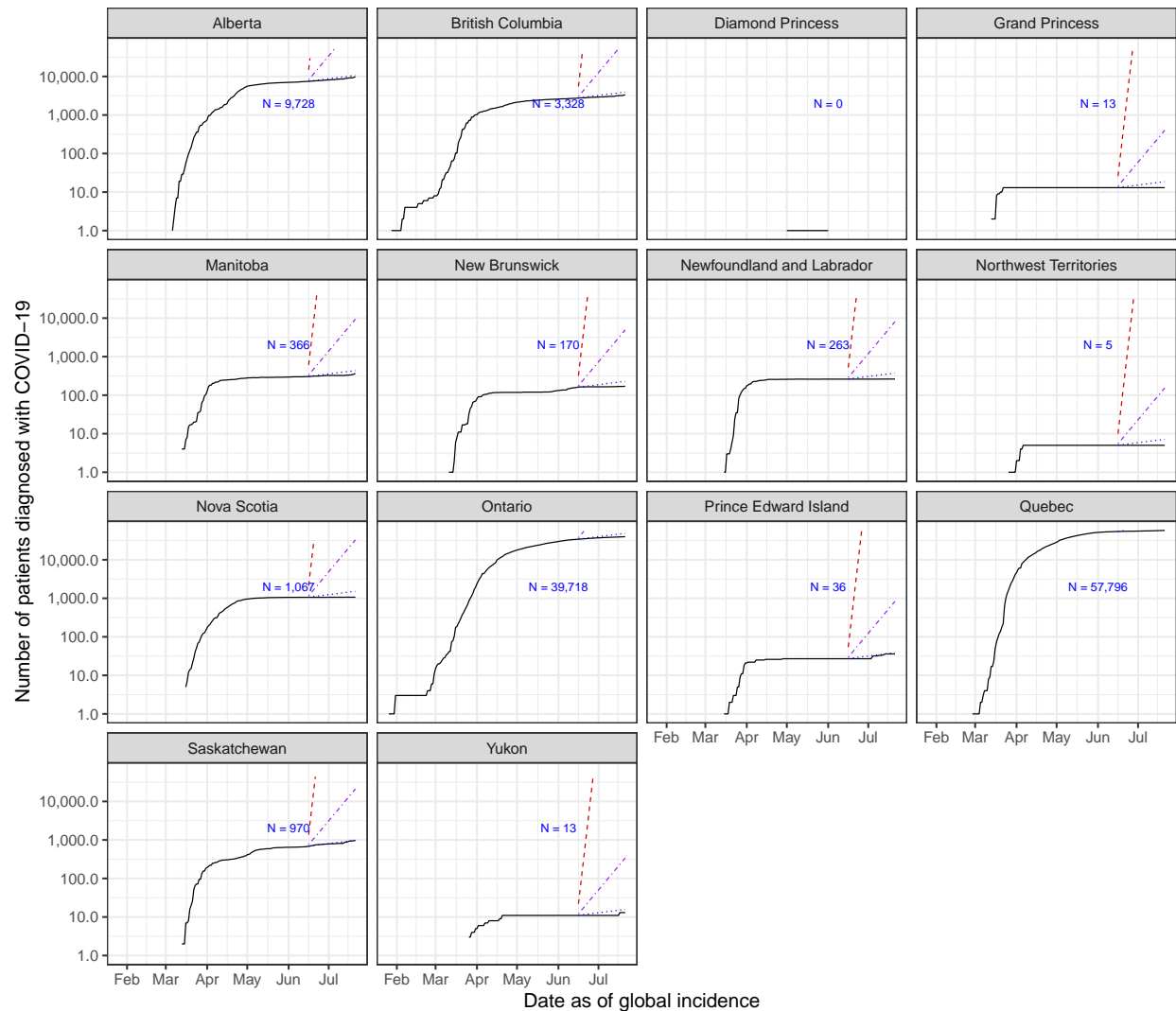
```
textdf$date = as.Date(as.character(textdf$variable), tryFormats="%m/%d/%y")
textdf$date = min(as.Date(textdf$date)) - 50
textdf$Label = paste("N =", comma(round(textdf$value)))
textdf$value = 2000
```

```
P = ggplot(moltendf[moltendf$value > 0,], aes(x=date, y=value, group=`Province/State`)) +
  geom_line() +
  geom_line(data=doubling_df[doubling_df$value > 0,], colour="red3", linetype=2) +
  geom_line(data=onepercent_df[onepercent_df$value > 0,], colour="blue", linetype=3) +
  geom_line(data=tenpercent_df[tenpercent_df$value > 0,], colour="purple", linetype=4) +
  geom_text(data=textdf, aes(label=Label), colour="blue") +
  theme_bw(base_size=18) +
  facet_wrap(~`Province/State`) +
  scale_y_log10("Number of patients diagnosed with COVID-19", labels=comma, limits=c(1, max(moltendf$value)))
  xlab("Date as of global incidence")
plot(P)
```

```
## Warning: Removed 381 row(s) containing missing values (geom_path).
```

```
## Warning: Removed 30 row(s) containing missing values (geom_path).
```

```
## Warning: Removed 87 row(s) containing missing values (geom_path).
```



Provincial new cases in Canada

```

casepath = "COVID-19/csse_covid_19_data/csse_covid_19_time_series/time_series_covid19_confirmed_global.csv"
casedf = read.csv(casepath, header=T, check.names=F)
casedf = casedf[casedf[,2]=="Canada", c(1, 5:ncol(casedf))]
provinces = casedf[apply(casedf[,2:ncol(casedf)], 1, sum) > 0, 1]
casemat = casedf[casedf[,1] %in% provinces,]

list.data = lapply(unique(casemat[,1]), function(province){
  all_cases = as.numeric(casemat[casemat[,1]==province, 2:ncol(casemat)])
  new_cases = diff(all_cases)
  addf = data.frame(date=colnames(casemat)[3:ncol(casemat)],
                    variable=province,
                    value=new_cases)
})
# Some countries (e.g. Spain on 5/25/20) have corrected their stats
# Enforce 0 on the plot

```

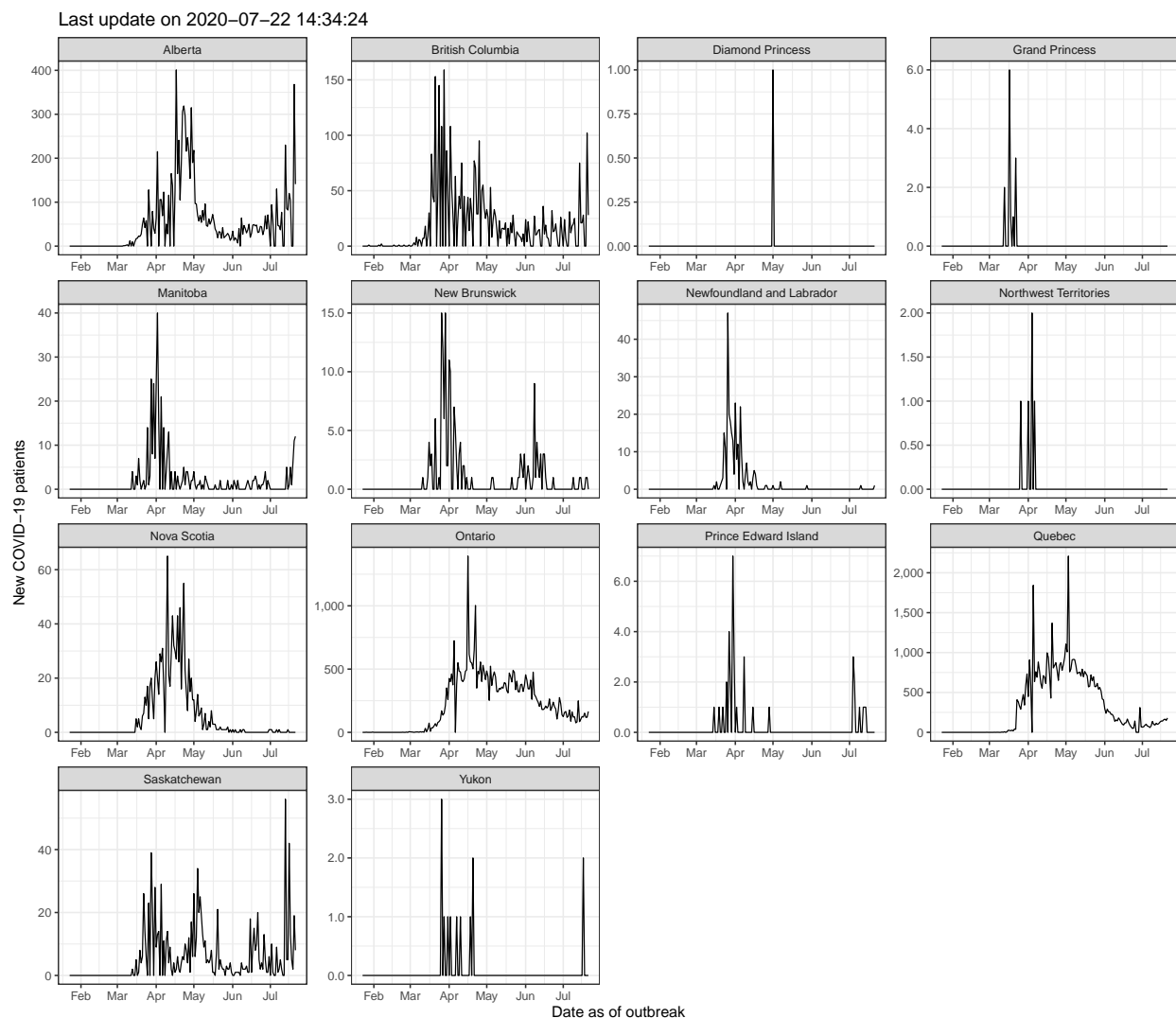


```

moltendf = do.call("rbind", list.data)
moltendf$value[moltendf$value < 0] = 0
moltendf$date = as.Date(moltendf$date, "%m/%d/%y")

P = ggplot(moltendf, aes(x=date, y=value, group=variable)) +
  geom_line() +
  theme_bw(base_size=14) +
  facet_wrap(~variable, scales="free") +
  scale_y_continuous(
    name="New COVID-19 patients", labels=comma) +
  ggtitle(paste("Last update on", Sys.time())) +
  xlab("Date as of outbreak")
plot(P)

```



Canada's deaths

```
casepath = "COVID-19/csse_covid_19_data/csse_covid_19_time_series/time_series_covid19_deaths_global.csv"
casedf = read.csv(casepath, header=T, check.names=F)
casedf = casedf[casedf[,2]=="Canada", c(1, 5:ncol(casedf))]
provinces = casedf[apply(casedf[,2:ncol(casedf)], 1, sum) > 0, 1]
casedf = casedf[casedf[,1] %in% provinces,]
```

```
moltendf = melt(casedf)
```

```
## Using Province/State as id variables
```

```
moltendf$date = as.Date(as.character(moltendf$variable), tryFormats="%m/%d/%y")
```

```
moltendf$variable = moltendf[, "Province/State"]
doubling_df = get_expected(moltendf, rate=2, init=100)
doubling_df[, "Province/State"] = doubling_df$variable
onepercent_df = get_expected(moltendf, rate=1.01, init=100)
onepercent_df[, "Province/State"] = onepercent_df$variable
tenpercent_df = get_expected(moltendf, rate=1.1, init=100)
tenpercent_df[, "Province/State"] = tenpercent_df$variable
```

```
textdf = melt(casedf[, c(1, ncol(casedf))])
```

```
## Using Province/State as id variables
```

```
textdf$date = as.Date(as.character(textdf$variable), tryFormats="%m/%d/%y")
textdf$date = min(as.Date(textdf$date)) - 5
textdf$Label = paste("N =", comma(round(textdf$value)))
textdf$value = max(moltendf$value)
```

```
P = ggplot(moltendf[moltendf$value > 0,], aes(x=date, y=value, group=`Province/State`)) +
  geom_line() +
  geom_line(data=doubling_df[doubling_df$value > 0,], colour="red3", linetype=2) +
  geom_line(data=onepercent_df[onepercent_df$value > 0,], colour="blue", linetype=3) +
  geom_line(data=tenpercent_df[tenpercent_df$value > 0,], colour="purple", linetype=4) +
  geom_text(data=textdf, aes(label=Label), colour="blue") +
  theme_bw(base_size=18) +
  facet_wrap(~`Province/State`) +
  scale_y_log10("Number of deaths due to COVID-19", labels=comma, limits=c(1, max(moltendf$value))) +
  xlab("Date as of first Canadian death")
plot(P)
```

```
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
```

```
## Warning: Removed 293 row(s) containing missing values (geom_path).
```

```
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
```

Warning: Removed 29 row(s) containing missing values (geom_path).

Warning: Removed 64 row(s) containing missing values (geom_path).

