`LogEpi`: Logistic curves applied to Covid-19

Beatriz Cuyabano and Gabriel Rovere

April 6, 2020

This vignette offers a brief explanation on how the logistic curve is used to model epidemiological events, and provides a set of examples of analysis using the R package `LogEpi`[1].

A logistic curve can be used to fit a counting process that starts at zero and ends at a determined number. Such curve has been widely used to describe the growth of a population, the growth of cases in a disease outbreak, or the growth of deaths in a such outbreak. The general logistic function is described as:

$$y \;=\; f(x \mid a, b, c) \;=\; \frac{a}{1 + \exp\left[(b - x)/c\right]}. \tag{1}$$

The value of y represents the cumulative counts of the event to be described, x represents the days passed after the first occurrence of this event, and a, b, and c are the parameters of the event. To simplify the description, $x = 1$ at the day of the first occurrence.

- a is the total number of occurrences at the stabilization of the event,

- b is the day in which we will observe the maximum new occurrences of the event,

- c is the speed of the process.

The parameters are obtained by minimizing the mean square error (MSE) of the model:

$$\text{MSE} \;=\; \frac{1}{n} \sum_{i=1}^{n} \left(y_i - \frac{a}{1 + \exp\left[(b - x_i)/c\right]} \right)^2, \tag{2}$$

$$\hat{a}, \hat{b}, \hat{c} \;=\; argmin\ \text{MSE}, \tag{3}$$

constraining their values to $a \geq max(y_i)$, $b \geq 1$, and $c > 0$.

Note that this curve does not take into account any demographic or social information. The adjustment is based on the observed numbers provided. If the outbreak is still in its initial stage, projections will be necessary to model the entire outbreak. This will be detailed with examples

---

[1] `LogEpi` is in its initial versions. Please help us improve our work in this package by reporting any bugs, errors, or suggestions to bia.cdc@gmail.com.

throughout this document. The latest Covid-19 data used for the examples was downloaded directly into R using the `LogEpi` package, and is provided by Johns Hopkins University (`https://systems.jhu.edu/research/public-health/ncov/`). The datasets are available at the repository `https://github.com/CSSEGISandData/COVID-19`. This data contains the historical numbers of confirmed cases, deaths, and recovery, both globally and detailed for all the states of the United States. The examples are based on the data downloaded April 6, 2020.

We will start by loading the package and defining the folder and filenames template to download the Covid-19 data.

```
# load LogEpi package
library(LogEpi)


# define folder and filenames template for the Covid-19 data
tmp.folder <- "Covid19_folder"
tmp.filename <- "Covid19_JHdata"
```

Function `load_JH_db()` dowloads the datasets into the folder defined in object `tmp.folder`, using the object `tmp.filename` as the filenames template.

```
# download the data from the JHU repository
# https://github.com/CSSEGISandData/COVID-19
loc <- load_JH_db(folder=tmp.folder,filename=tmp.filename)
```

As of today (April 6, 2020), the recovery data is not available specifically for the US states, only for the country as a whole, therefore the warning message below is expected.

```
# Warning messages:
# 1: In download.file(info.url,paste0(folder,"/",filename,"_",data.type, : cannot open
#    URL 'https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_
#    19_data/csse_covid_19_time_series/time_series_covid19_recovered_US.csv': HTTP sta
#    tus was '404 Not Found'
# 2: In download.file(info.url,paste0(folder,"/",filename,"_",data.type, : cannot open
#    URL 'https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_
#    19_data/csse_covid_19_time_series/time_series_covid19_recovered_US.csv': HTTP sta
#    tus was '404 Not Found'
```

The output of `load_JH_db()` is a list with two objects: `countries` and `US_states`. These are the lists of places for which the data is available.

```
head(loc$countries)

## [1] "Afghanistan"        "Albania"             "Algeria"
## [4] "Andorra"            "Angola"              "Antigua and Barbuda"


head(loc$US_states)

## [1] "Alabama"        "Alaska"        "American Samoa" "Arizona"
## [5] "Arkansas"       "California"
```

To extract the Covid-19 data from a location of interest, function `extract_covid19_data()` will be used. If the interest is in the data from a country, set argument `data="global"`. If the interest is in the data from a US state, set argument `data="US"`. The output will be a list with three objects: `confirmed`, `deaths`, and `recovered`. For data on states of the US, `recovered` will be NULL. Each object has a first column with the name of a province/state/administration within the location, and the subsequent columns are the cumulative reported numbers on the dates indicated as the column's name. If no information is available for sublocations, the first column will be empty and only one row of data will be output. We offer three examples to illustrate.

China

```
dataCovid <- extract_covid19_data("China","global",folder=tmp.folder,filename=tmp.filename)

head(dataCovid$confirmed[,1:8],7)

##   Province/State 1/22/20 1/23/20 1/24/20 1/25/20 1/26/20 1/27/20 1/28/20
## 1          Anhui       1       9      15      39      60      70     106
## 2        Beijing      14      22      36      41      68      80      91
## 3      Chongqing       6       9      27      57      75     110     132
## 4         Fujian       1       5      10      18      35      59      80
## 5          Gansu       0       2       2       4       7      14      19
## 6      Guangdong      26      32      53      78     111     151     207
## 7        Guangxi       2       5      23      23      36      46      51

head(dataCovid$deaths[,1:8],7)

##   Province/State 1/22/20 1/23/20 1/24/20 1/25/20 1/26/20 1/27/20 1/28/20
## 1          Anhui       0       0       0       0       0       0       0
## 2        Beijing       0       0       0       0       0       1       1
## 3      Chongqing       0       0       0       0       0       0       0
## 4         Fujian       0       0       0       0       0       0       0
## 5          Gansu       0       0       0       0       0       0       0
## 6      Guangdong       0       0       0       0       0       0       0
## 7        Guangxi       0       0       0       0       0       0       0

head(dataCovid$recovered[,1:8],7)

##   Province/State 1/22/20 1/23/20 1/24/20 1/25/20 1/26/20 1/27/20 1/28/20
## 1          Anhui       0       0       0       0       0       0       0
## 2        Beijing       0       0       1       2       2       2       4
## 3      Chongqing       0       0       0       0       0       0       0
## 4         Fujian       0       0       0       0       0       0       0
## 5          Gansu       0       0       0       0       0       0       0
## 6      Guangdong       0       2       2       2       2       4       4
## 7        Guangxi       0       0       0       0       0       0       2
```

Italy

```
dataCovid <- extract_covid19_data("Italy","global",folder=tmp.folder,filename=tmp.filename)
dataCovid$confirmed[,1:8]
```

```
##   Province/State 1/22/20 1/23/20 1/24/20 1/25/20 1/26/20 1/27/20 1/28/20
## 1                      0       0       0       0       0       0       0
```

```
dataCovid$deaths[,1:8]
```

```
##   Province/State 1/22/20 1/23/20 1/24/20 1/25/20 1/26/20 1/27/20 1/28/20
## 1                      0       0       0       0       0       0       0
```

```
dataCovid$recovered[,1:8]
```

```
##   Province/State 1/22/20 1/23/20 1/24/20 1/25/20 1/26/20 1/27/20 1/28/20
## 1                      0       0       0       0       0       0       0
```

Michigan (US state)

```
dataCovid <- extract_covid19_data("Michigan","US",folder=tmp.folder,filename=tmp.filename)
head(dataCovid$confirmed[,1:8])
```

```
##     Admin2 1/22/20 1/23/20 1/24/20 1/25/20 1/26/20 1/27/20 1/28/20
## 1  Alcona       0       0       0       0       0       0       0
## 2   Alger       0       0       0       0       0       0       0
## 3 Allegan       0       0       0       0       0       0       0
## 4  Alpena       0       0       0       0       0       0       0
## 5  Antrim       0       0       0       0       0       0       0
## 6  Arenac       0       0       0       0       0       0       0
```

```
head(dataCovid$deaths[,1:8])
```

```
##     Admin2 1/22/20 1/23/20 1/24/20 1/25/20 1/26/20 1/27/20 1/28/20
## 1  Alcona       0       0       0       0       0       0       0
## 2   Alger       0       0       0       0       0       0       0
## 3 Allegan       0       0       0       0       0       0       0
## 4  Alpena       0       0       0       0       0       0       0
## 5  Antrim       0       0       0       0       0       0       0
## 6  Arenac       0       0       0       0       0       0       0
```

```
head(dataCovid$recovered[,1:8])
```

```
## NULL
```

Now, we want to build epidemic tables, with the data displayed in columns. This will be done using function `mkEpiTable()`. By default, the output is a table with cumulative the cumulative numbers of confirmed cases, deaths and recovered, as well as the number of active cases on the dates. Setting `daily=TRUE` will change the output to a table with the numbers reported in each day. The number of active cases is reported equally for both `daily=TRUE` and `daily=FALSE`, as this number is neither

cumulative nor new occurences. For locations with detailed data by province/state/administration, data can be narrowed to a specific sublocation, or a set of sublocations. We offer a few examples on two different locations to illustrate.

China

```
dataCovid <- extract_covid19_data("China","global",folder=tmp.folder,filename=tmp.filename)
tbCovid <- mkEpiTable(dataCovid)
head(tbCovid)

##          date confirmed deaths recovered active
## 1 2020-01-22       548     17        28    503
## 2 2020-01-23       643     18        30    595
## 3 2020-01-24       920     26        36    858
## 4 2020-01-25      1406     42        39   1325
## 5 2020-01-26      2075     56        49   1970
## 6 2020-01-27      2877     82        58   2737

tbCovid <- mkEpiTable(dataCovid,daily=TRUE)
head(tbCovid)

##          date confirmed deaths recovered active
## 1 2020-01-22       548     17        28    503
## 2 2020-01-23        95      1         2    595
## 3 2020-01-24       277      8         6    858
## 4 2020-01-25       486     16         3   1325
## 5 2020-01-26       669     14        10   1970
## 6 2020-01-27       802     26         9   2737

tbCovid <- mkEpiTable(dataCovid,specific="Hubei")
head(tbCovid)

##          date confirmed deaths recovered active
## 1 2020-01-22       444     17        28    399
## 2 2020-01-23       444     17        28    399
## 3 2020-01-24       549     24        31    494
## 4 2020-01-25       761     40        32    689
## 5 2020-01-26      1058     52        42    964
## 6 2020-01-27      1423     76        45   1302

tbCovid <- mkEpiTable(dataCovid,specific=c("Hubei","Guangdong"))
head(tbCovid)

##          date confirmed deaths recovered active
## 1 2020-01-22       470     17        28    425
## 2 2020-01-23       476     17        30    429
## 3 2020-01-24       602     24        33    545
## 4 2020-01-25       839     40        34    765
## 5 2020-01-26      1169     52        44   1073
## 6 2020-01-27      1574     76        49   1449
```

Michigan (US state)

```
dataCovid <- extract_covid19_data("Michigan","US",folder=tmp.folder,filename=tmp.filename)
tbCovid <- mkEpiTable(dataCovid)
head(tbCovid)

##          date confirmed deaths recovered active
## 1 2020-01-22         0      0        NA    NA
## 2 2020-01-23         0      0        NA    NA
## 3 2020-01-24         0      0        NA    NA
## 4 2020-01-25         0      0        NA    NA
## 5 2020-01-26         0      0        NA    NA
## 6 2020-01-27         0      0        NA    NA

head(tbCovid[tbCovid$confirmed > 0,])

##           date confirmed deaths recovered active
## 50 2020-03-11         2      0        NA    NA
## 51 2020-03-12         2      0        NA    NA
## 52 2020-03-13        16      0        NA    NA
## 53 2020-03-14        25      0        NA    NA
## 54 2020-03-15        32      0        NA    NA
## 55 2020-03-16        54      0        NA    NA

tbCovid <- mkEpiTable(dataCovid,specific="Ingham")
head(tbCovid)

##          date confirmed deaths recovered active
## 1 2020-01-22         0      0        NA    NA
## 2 2020-01-23         0      0        NA    NA
## 3 2020-01-24         0      0        NA    NA
## 4 2020-01-25         0      0        NA    NA
## 5 2020-01-26         0      0        NA    NA
## 6 2020-01-27         0      0        NA    NA

head(tbCovid[tbCovid$confirmed > 0,])

##           date confirmed deaths recovered active
## 52 2020-03-13         1      0        NA    NA
## 53 2020-03-14         1      0        NA    NA
## 54 2020-03-15         1      0        NA    NA
## 55 2020-03-16         1      0        NA    NA
## 56 2020-03-17         3      0        NA    NA
## 57 2020-03-18         5      0        NA    NA
```
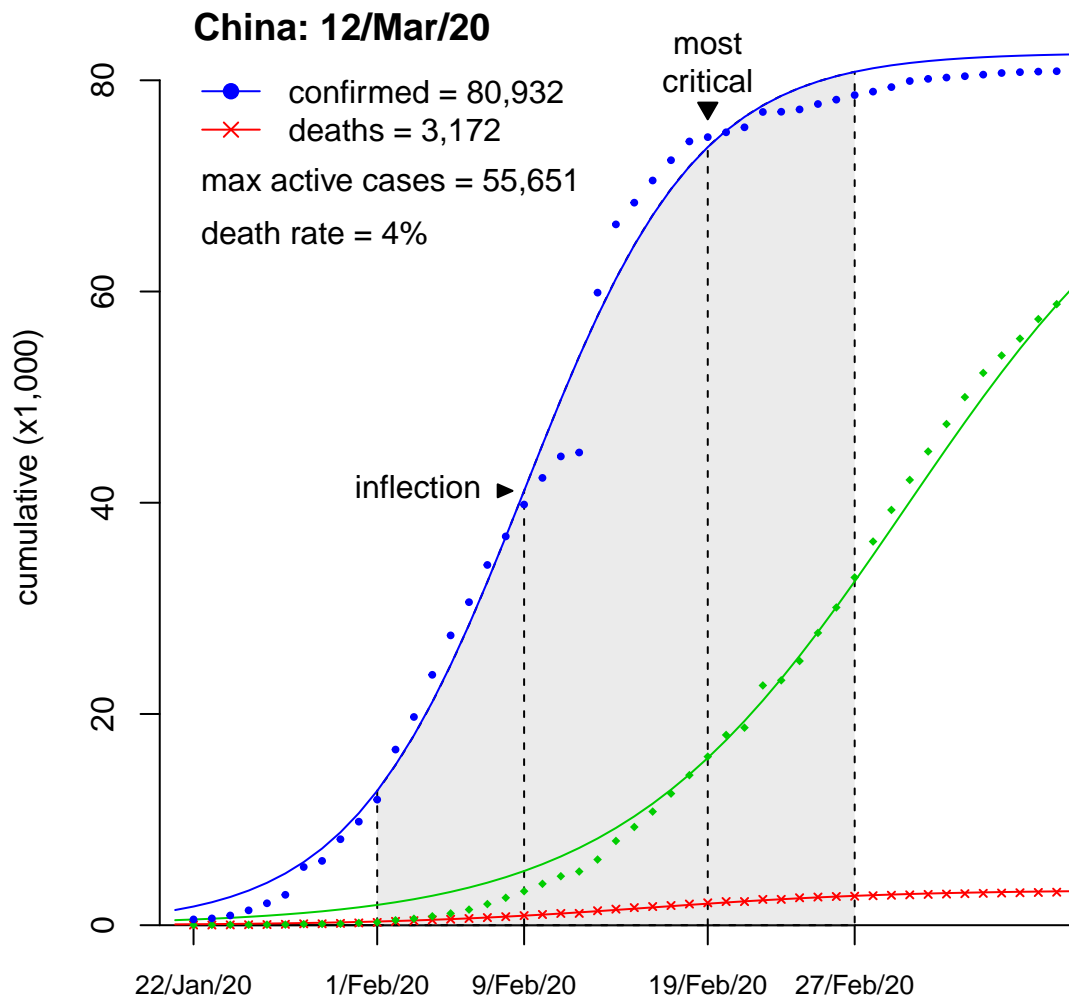
The next step is to use these tables generated by `mkEpiTable()` to model the logistic curves of the epidemic in the location of interest. We will now use the function `mkEpiCurves()`.

China

```
dataCovid <- extract_covid19_data("China","global",folder=tmp.folder,filename=tmp.filename)
tbCovid <- mkEpiTable(dataCovid)
fitCovid <- mkEpiCurves(tbCovid,plot.title="China")
```



```
# print the parameters of the curves
fitCovid$par

##                a        b        c
## confirmed  82602 19.03933 4.719325
## deaths      3333 25.85590 6.917347
## recovered  79269 39.81607 7.792875

# print the dates in which the first observation of the epidemic events occurred
fitCovid$day1

##    confirmed      deaths    recovered
## "2020-01-22" "2020-01-22" "2020-01-22"

# print the key dates of the of this epidemic
fitCovid$key_dates
```

```
## start_critical    inflection  most_critical   end_critical
##   "2020-02-01"   "2020-02-09"   "2020-02-19"   "2020-02-27"


# rates with which the epidemic event develops
head(fitCovid$rates)


##          date confirmed    deaths recovered
## 1 2020-01-22  0.000000 0.000000  0.000000
## 2 2020-01-23  1.173358 1.058824  1.071429
## 3 2020-01-24  1.430793 1.444444  1.200000
## 4 2020-01-25  1.528261 1.615385  1.083333
## 5 2020-01-26  1.475818 1.333333  1.256410
## 6 2020-01-27  1.386506 1.464286  1.183673
```
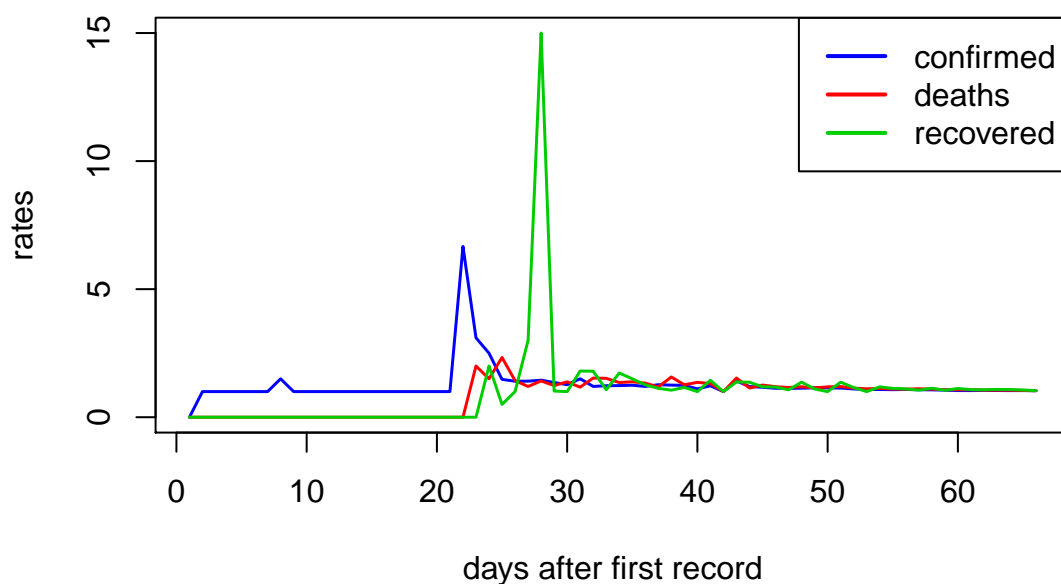
The rates in `fitCovid$rates` are the percentage with which the events develop. For example, if in a day the total number reported of the epidemic event is 100, and 120 on the next day, the rate observed in this next day is 120/100 = 1.2, an increase of 20%.

```
# plot the rates
xrange <- c(1,nrow(fitCovid$rates))
yrange <- range(fitCovid$rates[,-1])
plot(NA,xlim=xrange,ylim=yrange,xlab="days after first record",ylab="rates")
lines(1:nrow(fitCovid$rates),fitCovid$rates$confirmed,lwd=1.5,col=4)
lines(1:nrow(fitCovid$rates),fitCovid$rates$deaths,lwd=1.5,col=2)
lines(1:nrow(fitCovid$rates),fitCovid$rates$recovered,lwd=1.5,col=3)
abline(v=which(fitCovid$rates$date == today()))
legend("topright",lwd=2,col=c(4,2,3),legend=names(fitCovid$rates)[-1])
```

Italy

```
dataCovid <- extract_covid19_data("Italy","global",folder=tmp.folder,filename=tmp.filename)
tbCovid <- mkEpiTable(dataCovid)
fitCovid <- mkEpiCurves(tbCovid,plot.title="Italy")
```
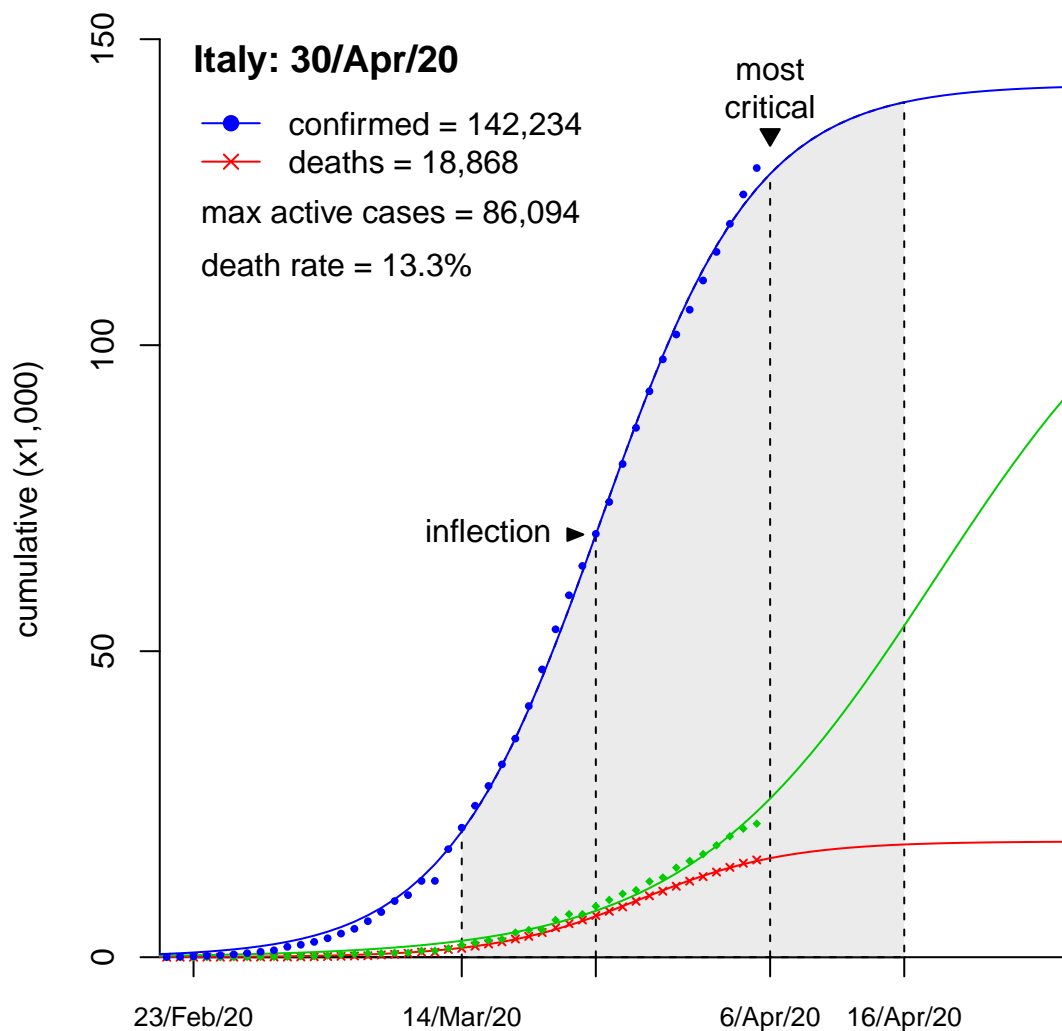


```
# print the parameters of the curves
fitCovid$par

##                   a        b        c
## confirmed 142543.43 54.36203 5.812610
## deaths     18915.67 36.29729 5.462303
## recovered 123627.76 57.30615 9.270946

# print the dates in which the first observation of the epidemic events occurred
fitCovid$day1

##    confirmed       deaths    recovered
## "2020-01-31" "2020-02-21" "2020-02-22"

# print the key dates of the of this epidemic
fitCovid$key_dates
```

```
## start_critical    inflection  most_critical    end_critical
##   "2020-03-14"   "2020-03-24"   "2020-04-06"   "2020-04-16"

# rates with which the epidemic event develops
head(fitCovid$rates)

##          date confirmed deaths recovered
## 1 2020-01-31         0      0         0
## 2 2020-02-01         1      0         0
## 3 2020-02-02         1      0         0
## 4 2020-02-03         1      0         0
## 5 2020-02-04         1      0         0
## 6 2020-02-05         1      0         0
```

```r
# plot the rates
xrange <- c(1,nrow(fitCovid$rates))
yrange <- range(fitCovid$rates[,-1])
plot(NA,xlim=xrange,ylim=yrange,xlab="days after first record",ylab="rates")
lines(1:nrow(fitCovid$rates),fitCovid$rates$confirmed,lwd=1.5,col=4)
lines(1:nrow(fitCovid$rates),fitCovid$rates$deaths,lwd=1.5,col=2)
lines(1:nrow(fitCovid$rates),fitCovid$rates$recovered,lwd=1.5,col=3)
abline(v=which(fitCovid$rates$date == today()))
legend("topright",lwd=2,col=c(4,2,3),legend=names(fitCovid$rates)[-1])
```

Michigan (US state)

```
dataCovid <- extract_covid19_data("Michigan","US",folder=tmp.folder,filename=tmp.filename)
tbCovid <- mkEpiTable(dataCovid)
fitCovid <- mkEpiCurves(tbCovid,plot.title="Michigan")
```
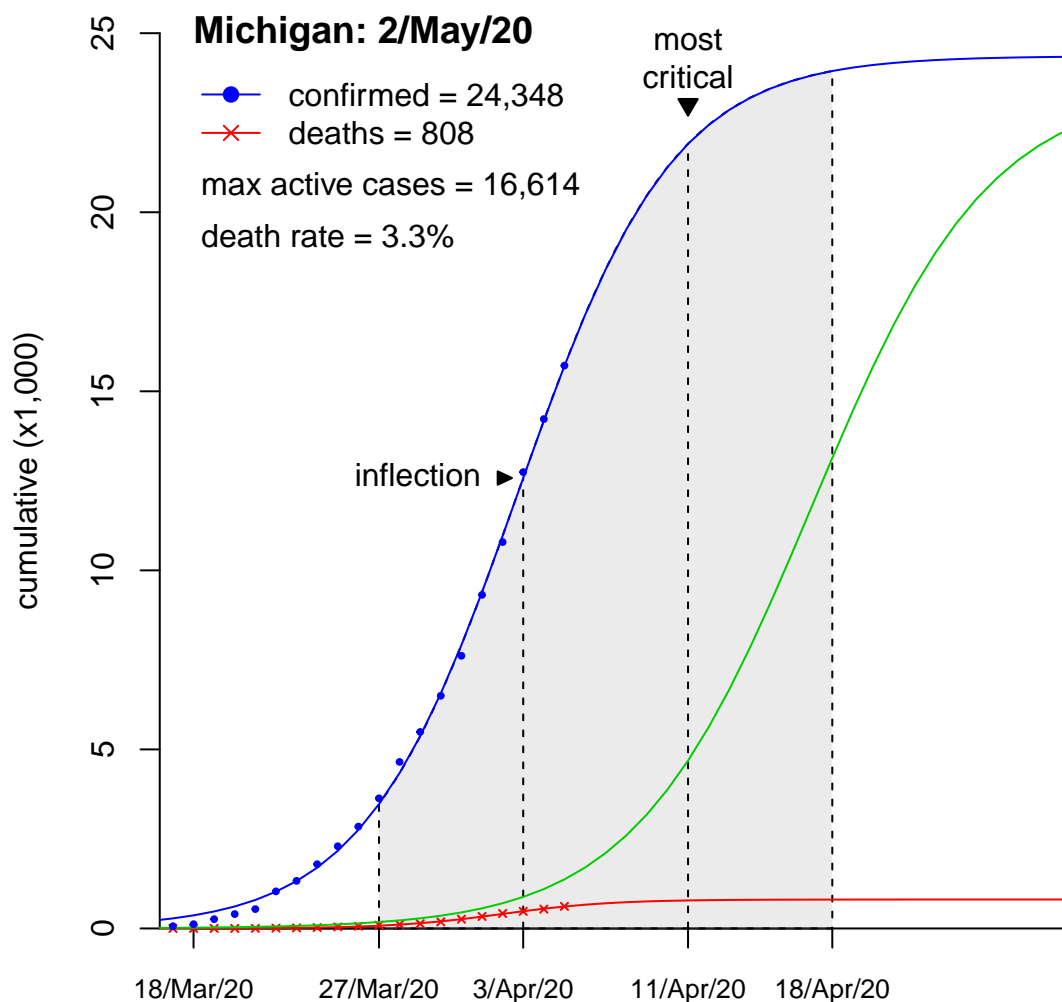


```
# print the parameters of the curves
fitCovid$par

##                    a        b        c
## confirmed 24361.5974 23.75432 3.763596
## deaths      808.0161 15.98536 2.644672
## recovered 23553.5813 24.00000 4.308768

# print the dates in which the first observation of the epidemic events occurred
fitCovid$day1

##    confirmed      deaths    recovered
## "2020-03-11" "2020-03-18" "2020-03-25"

# print the key dates of the of this epidemic
fitCovid$key_dates
```

```
## start_critical     inflection  most_critical    end_critical
##    "2020-03-27"   "2020-04-03"   "2020-04-11"   "2020-04-18"


# rates with which the epidemic event develops
head(fitCovid$rates)


##          date confirmed deaths recovered
## 1 2020-03-11    0.0000      0         0
## 2 2020-03-12    1.0000      0         0
## 3 2020-03-13    8.0000      0         0
## 4 2020-03-14    1.5625      0         0
## 5 2020-03-15    1.2800      0         0
## 6 2020-03-16    1.6875      0         0
```

```
# plot the rates
xrange <- c(1,nrow(fitCovid$rates))
yrange <- range(fitCovid$rates[,-1])
plot(NA,xlim=xrange,ylim=yrange,xlab="days after first record",ylab="rates")
lines(1:nrow(fitCovid$rates),fitCovid$rates$confirmed,lwd=1.5,col=4)
lines(1:nrow(fitCovid$rates),fitCovid$rates$deaths,lwd=1.5,col=2)
lines(1:nrow(fitCovid$rates),fitCovid$rates$recovered,lwd=1.5,col=3)
abline(v=which(fitCovid$rates$date == today()))
legend("topright",lwd=2,col=c(4,2,3),legend=names(fitCovid$rates)[-1])
```
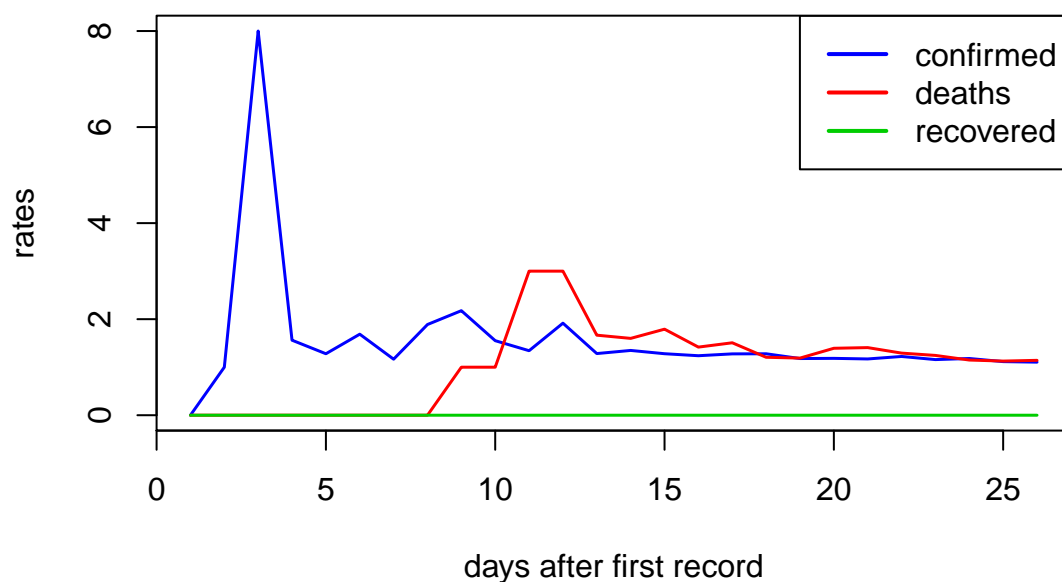
United States

```
dataCovid <- extract_covid19_data("US","global",folder=tmp.folder,filename=tmp.filename)
tbCovid <- mkEpiTable(dataCovid)
fitCovid <- mkEpiCurves(tbCovid,plot.title="USA")
```
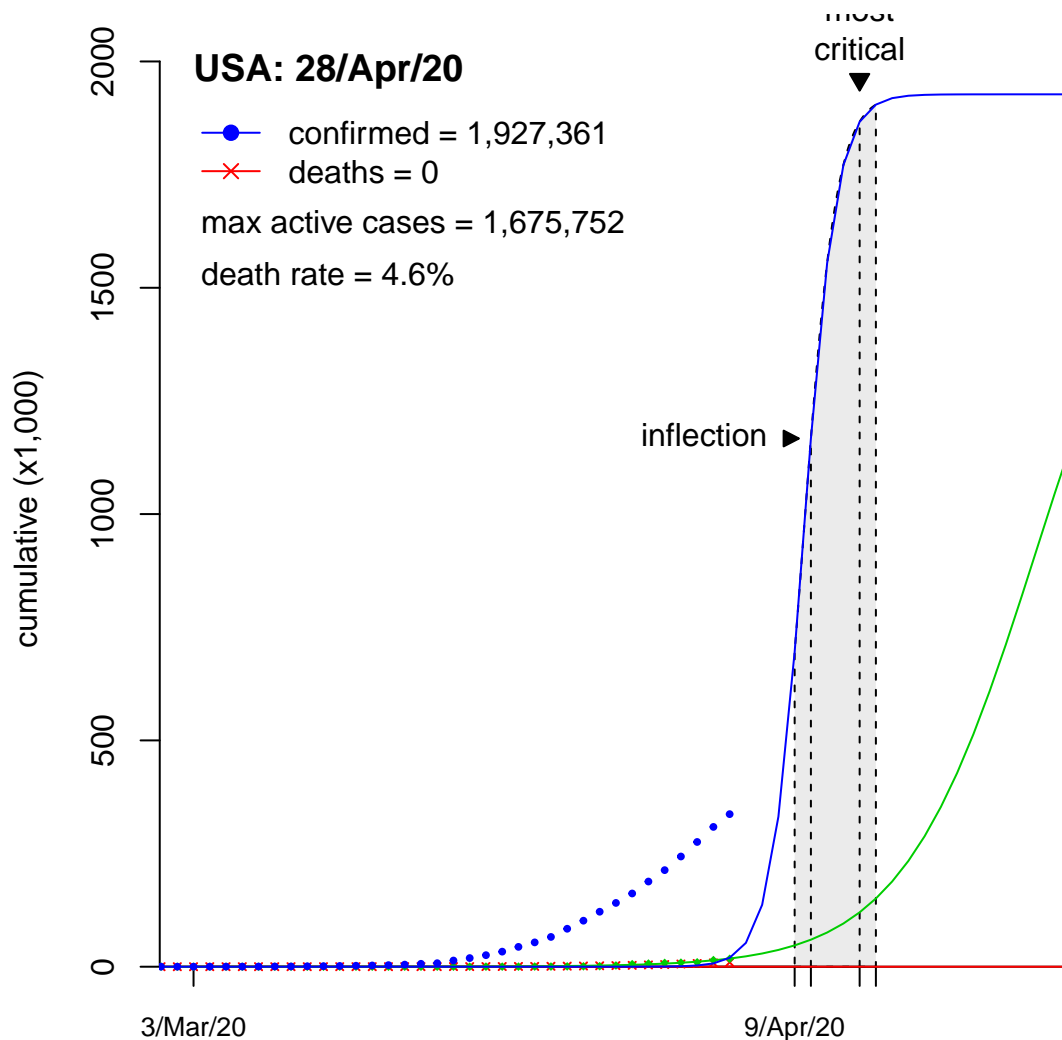


```
# print the parameters of the curves
fitCovid$par
```

```
##                    a         b        c
## confirmed 1927360.74  79.57167 1.000000
## deaths      89075.85 164.48998 1.758284
## recovered 1838284.89  75.87678 4.088403
```

The figure generated indicates a poor convergence of the model on the current data. This is an indicator that the current state of the epidemic in the US is still not providing sufficient data to model the epidemic. We will work with projections to better understand the situation in the US. In this case, we will generate scenarios for this projection.

```
# project 7 days ahead of developing the epidemic in the current rates
fitCovid <- mkEpiCurves(tbCovid,plot.title="USA",project=7)
```
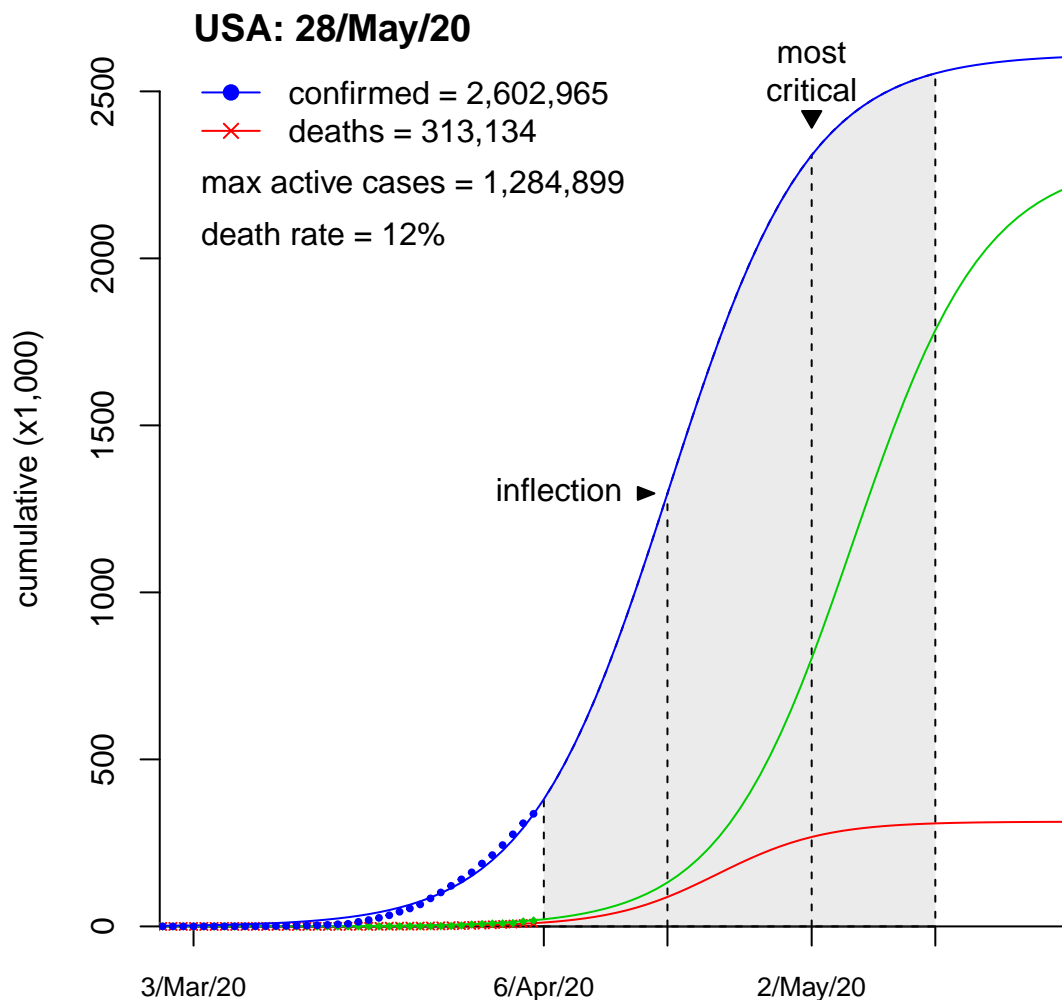


```
# print the parameters of the curves
fitCovid$par

##                    a        b        c
## confirmed 2611806.5 88.10044 6.842560
## deaths     313561.8 54.86306 5.176168
## recovered 2298244.7 88.00000 6.421118

# print the key dates of the of this epidemic
fitCovid$key_dates

## start_critical     inflection  most_critical    end_critical
##   "2020-04-06"   "2020-04-18"   "2020-05-02"    "2020-05-14"

# print the rates of the projection
unlist(fitCovid$rates_proj)

## confirmed    deaths
##  1.123794  1.200066
```