**Introduction**

The Novel COVID-19 Coronavirus is still spreading quickly in several countries and it does not seem like it is going to stop anytime soon as the peak has not yet been reached in many countries.

Since the beginning of its expansion, a large number of scientists across the world have been analyzing this Coronavirus from different perspectives and with different technologies with the hope of coming up with a cure in order to stop its expansion and limit its impact on citizens.

**Top R resources on Coronavirus**

In the meantime, epidemiologists, statisticians and data scientists are working towards a better understanding of the spread of the virus in order to help governments and health agencies in taking the most optimal decisions. This led to the publication of a great deal of online resources about the virus, which I collected and organized in an article covering the [top R resources on Coronavirus](https://www.statsandr.com/blog/top-r-resources-on-covid-19-coronavirus/). This article is a collection of the best resources I’ve had the chance to discover, with a brief summary for each of them. It includes Shiny apps, dashboards, R packages, blog posts and datasets.

Publishing this collection led many readers to submit their piece of work, which made the article even more complete and more insightful for anyone interested in analyzing the virus from a quantitative perspective. Thanks to everyone who contributed and who helped me in collecting and summarizing these R resources about COVID-19!

Given my field of expertise, I am not able to help in this fight against the virus from a medical point of view. However, I still wanted to contribute as much as I could. From understanding better the disease to bringing scientists and doctors together to build something bigger and more impactful, I truly hope that this collection will, to a small extent, help to fight the pandemic.

**Coronavirus dashboard for your own country**

Besides receiving analyses, blog posts, R code and Shiny apps from people across the world, I realized that many people were trying to create a dashboard tracking the spread of the Coronavirus for their own country. So in addition to the collection of top R resources, I also published an article detailing the steps to follow to create a dashboard specific to a country.

The code has been made available on GitHub and is open source so everyone can copy it and adapt it to their own country. The dashboard was intentionally kept simple so anyone with a minimum knowledge in R could easily replicate it, and advanced users could enhance it according to their needs.

**Motivations, limitations and structure of the article**

By seeing and organizing many [R resources about COVID-19](https://www.statsandr.com/blog/top-r-resources-on-covid-19-coronavirus/), I am fortunate enough to have read a lot of excellent analyses on the disease outbreak, the impact of different health measures, forecasts of the number of cases, projections about the length of the pandemic, hospitals capacity, etc.

Furthermore, I must admit that some countries such as China, South Korea, Italy, Spain, UK and Germany received a lot of attention as shown by the number of analyses done on these countries. However, to my knowledge and at the date of publication of this article, I am not aware of any analysis of the spread of the Coronavirus specifically for Belgium. The present article aims at filling that gap.

I usually write articles only about things I consider myself familiar with, mainly statistics and its applications in R. At the time of writing this article, I was however curious where Belgium stands regarding the spread of this virus, I wanted to play with this kind of data in R (which is new to me) and see what comes out.

In order to satisfy my curiosity while not being an expert, in this article I am going to replicate analyses done by more knowledgeable people and apply them to my country, that is, Belgium. From all the analyses I have read so far, I decided to replicate the analyses done by Tim Churches and Prof. Dr. Holger K. von Jouanne-Diedrich. This article is based on a mix of their articles. They both present a very informative analysis on how to model the outbreak of the Coronavirus and show how contagious it is. Their articles also allowed me to gain an understanding of the topic and in particular an understanding of the most common epidemiological model

Other more complex analyses are possible and even preferable, but I leave this to experts in this field. Note also that the following analyses take into account only the data until the date of publication of this article, so the results should not be viewed, by default, as current findings.

In the remaining of the article, we first introduce the model which will be used to analyze the Coronavirus outbreak in Belgium. We also briefly discuss and show how to compute an important epidemiological measure, the reproduction number. We then use our model to analyze the outbreak of the disease in the case where there would be no public health intervention. We conclude the article by summarizing more advanced tools and techniques that could be used to further model COVID-19 in Belgium.

**Analysis of Coronavirus in Belgium**

**A classic epidemiological model: the *SIR* model**

Before diving into the real-life application, we first introduce the model that will be used.

There are many epidemiological models but we will use one of the simplest, the ***SIR* model**. Tim Churches’ explanation of this model and how to fit it using R is so nice, I will reproduce it here with a few minor changes.

The basic idea behind the *SIR* model (**S**usceptible – **I**nfectious – **R**ecovered) of communicable disease outbreaks is that there are three groups (also called compartments) of people:

* those who are healthy but susceptible to the disease: *S*
* the infectious (and thus, infected) people: *I*
* people who have recovered: *R*

SIR model. Source: Wikipedia

To model the dynamics of the outbreak we need three differential equations to describe the rates of change in each group, parameterised by:

* \(\beta\) which controls the transition between *S* and *I*
* \(\gamma\) which controls the transition between *I* and *R*

Formally, this gives:

\[\frac{dS}{dt} = – \frac{\beta IS}{N}\]

\[\frac{dI}{dt} = \frac{\beta IS}{N} – \gamma I\]

\[\frac{dR}{dt} = \gamma I\]

Before fitting the *SIR* model to the data, the first step is to express these differential equations as an R function, with respect to time *t*.

SIR <- function(time, state, parameters) {

par <- as.list(c(state, parameters))

with(par, {

dS <- -beta \* I \* S / N

dI <- beta \* I \* S / N - gamma \* I

dR <- gamma \* I

list(c(dS, dI, dR))

})

}

**Fitting a *SIR* model to the Belgium data**

To fit the model to the data we need two things:

1. a solver for these differential equations
2. an optimiser to find the optimal values for our two unknown parameters, \(\beta\) and \(\gamma\)

The function ode() (for ordinary differential equations) from the {deSolve} R package makes solving the system of equations easy, and to find the optimal values for the parameters we wish to estimate, we can just use the optim() function built into base R.

Specifically, what we need to do is minimise the sum of the squared differences between \(I(t)\), which is the number of people in the infectious compartment \(I\) at time \(t\), and the corresponding number of cases as predicted by our model \(\hat{I}(t)\). This quantity is known as the residual sum of squares (*RSS*):

\[RSS(\beta, \gamma) = \sum\_t \big(I(t) – \hat{I}(t) \big)^2\]

In order to fit a model to the incidence data for Belgium, we need a value *N* for the initial uninfected population. The population of Belgium in November 2019 was 11,515,793 people, according to [Wikipedia](https://en.wikipedia.org/wiki/Belgium). We will thus use *N = 11515793* as the initial uninfected population.

Next, we need to create a vector with the daily cumulative incidence for Belgium, from February 4 (when our daily incidence data starts), through to March 30 (last available date at the time of publication of this article). We will then compare the predicted incidence from the *SIR* model fitted to these data with the actual incidence since February 4. We also need to initialise the values for *N*, *S*, *I* and *R*. Note that the daily cumulative incidence for Belgium is extracted from the {coronavirus} R package developed by Rami Krispin.

# devtools::install\_github("RamiKrispin/coronavirus")

library(coronavirus)

data(coronavirus)

`%>%` <- magrittr::`%>%`

# extract the cumulative incidence

df <- coronavirus %>%

dplyr::filter(Country.Region == "Belgium") %>%

dplyr::group\_by(date, type) %>%

dplyr::summarise(total = sum(cases, na.rm = TRUE)) %>%

tidyr::pivot\_wider(

names\_from = type,

values\_from = total

) %>%

dplyr::arrange(date) %>%

dplyr::ungroup() %>%

dplyr::mutate(active = confirmed - death - recovered) %>%

dplyr::mutate(

confirmed\_cum = cumsum(confirmed),

death\_cum = cumsum(death),

recovered\_cum = cumsum(recovered),

active\_cum = cumsum(active)

)

# put the daily cumulative incidence numbers for Belgium from

# Feb 4 to March 30 into a vector called Infected

library(lubridate)

Infected <- subset(df, date >= ymd("2020-02-04"))$confirmed\_cum

# Create an incrementing Day vector the same length as our

# cases vector

Day <- 1:(length(Infected))

# now specify initial values for N, S, I and R

N <- 11515793

init <- c(

S = N - Infected[1],

I = Infected[1],

R = 0

)

Then we need to define a function to calculate the *RSS*, given a set of values for \(\beta\) and \(\gamma\).

# define a function to calculate the residual sum of squares

# (RSS), passing in parameters beta and gamma that are to be

# optimised for the best fit to the incidence data

RSS <- function(parameters) {

names(parameters) <- c("beta", "gamma")

out <- ode(y = init, times = Day, func = SIR, parms = parameters)

fit <- out[, 3]

sum((Infected - fit)^2)

}

Finally, we can fit the *SIR* model to our data by finding the values for \(\beta\) and \(\gamma\) that minimise the residual sum of squares between the observed cumulative incidence (observed in Belgium) and the predicted cumulative incidence (predicted by our model). We also need to check that our model has converged, as indicated by the message shown below:

# now find the values of beta and gamma that give the

# smallest RSS, which represents the best fit to the data.

# Start with values of 0.5 for each, and constrain them to

# the interval 0 to 1.0

# install.packages("deSolve")

library(deSolve)

Opt <- optim(c(0.5, 0.5),

RSS,

method = "L-BFGS-B",

lower = c(0, 0),

upper = c(1, 1)

)

# check for convergence

Opt$message

## [1] "CONVERGENCE: REL\_REDUCTION\_OF\_F <= FACTR\*EPSMCH"

Convergence is confirmed. Now we can examine the fitted values for \(\beta\) and \(\gamma\):

Opt\_par <- setNames(Opt$par, c("beta", "gamma"))

Opt\_par

## beta gamma

## 0.5986056 0.4271395

Remember that \(\beta\) controls the transition between *S* and *I* (i.e., susceptible and infectious) and \(\gamma\) controls the transition between *I* and *R* (i.e., infectious and recovered). However, those values do not mean a lot but we use them to get the fitted numbers of people in each compartment of our *SIR* model for the dates up to March 30 that were used to fit the model, and compare those fitted values with the observed (real) data.

sir\_start\_date <- "2020-02-04"

# time in days for predictions

t <- 1:as.integer(today() - ymd(sir\_start\_date))

# get the fitted values from our SIR model

fitted\_cumulative\_incidence <- data.frame(ode(

y = init, times = t,

func = SIR, parms = Opt\_par

))

# add a Date column and the observed incidence data

library(dplyr)

fitted\_cumulative\_incidence <- fitted\_cumulative\_incidence %>%

mutate(

Date = ymd(sir\_start\_date) + days(t - 1),

Country = "Belgium",

cumulative\_incident\_cases = Infected

)

# plot the data

library(ggplot2)

fitted\_cumulative\_incidence %>%

ggplot(aes(x = Date)) +

geom\_line(aes(y = I), colour = "red") +

geom\_point(aes(y = cumulative\_incident\_cases), colour = "blue") +

labs(

y = "Cumulative incidence",

title = "COVID-19 fitted vs observed cumulative incidence, Belgium",

subtitle = "(Red = fitted incidence from SIR model, blue = observed incidence)"

) +

theme\_minimal()

From the above graph we see that the number of observed confirmed cases follows, unfortunately, the number of confirmed cases expected by our model. The fact that both trends are overlapping indicates that the pandemic is clearly in an exponential phase in Belgium. More data would be needed to see whether this trend is confirmed in the long term.

The following graph is similar than the previous one, except that the *y*-axis is measured on a log scale. This kind of plot is called a semi-log plot or more precisely a log-linear plot because only the *y*-axis is transformed with a logarithm scale. Transforming the scale in log has the advantage that it is more easily readable in terms of difference between the observed and expected number of confirmed cases and it also shows how the number of observed confirmed cases differs from an exponential trend.

fitted\_cumulative\_incidence %>%

ggplot(aes(x = Date)) +

geom\_line(aes(y = I), colour = "red") +

geom\_point(aes(y = cumulative\_incident\_cases), colour = "blue") +

labs(

y = "Cumulative incidence (log scale)",

title = "COVID-19 fitted vs observed cumulative incidence, Belgium",

subtitle = "(Red = fitted incidence from SIR model, blue = observed incidence)"

) +

theme\_minimal() +

scale\_y\_log10()

The plot indicates that, at the beginning of the pandemic and until March 12, the number of confirmed cases stayed below what would be expected in an exponential phase. In particular, the number of confirmed cases stayed constant at 1 case from February 4 to February 29. From March 13 and until March 30, the number of confirmed cases kept increasing at a rate close to an exponential rate.

We also notice a small jump between March 12 and March 13, which may potentially indicate an error in the data collection, or a change in the testing/screening methods.

**Reproduction number \(R\_0\)**

Our *SIR* model looks like a good fit to the observed cumulative incidence data in Belgium, so we can now use our fitted model to calculate the basic reproduction number \(R\_0\), also referred as basic reproduction ratio.

The reproduction number gives the average number of susceptible people who are infected by each infectious person. In other words, the reproduction number refers to the number of healthy people that get infected per number of sick people. When \(R\_0 > 1\) the disease starts spreading in a population, but not if \(R\_0 < 1\). Usually, the larger the value of \(R\_0\), the harder it is to control the epidemic.

Formally, we have:

\[R\_0 = \frac{\beta}{\gamma}\]

We can compute it in R:

Opt\_par

## beta gamma

## 0.5986056 0.4271395

R0 <- as.numeric(Opt\_par[1] / Opt\_par[2])

R0

## [1] 1.401429

An \(R\_0\) of 1.4 is slightly below the values calculated by others for COVID-19 and the \(R\_0\) for SARS and MERS, which are similar diseases also caused by coronavirus. This is mainly due to the fact that the number of confirmed cases stayed constant and equal to 1 at the beginning of the pandemic.

A \(R\_0\) of 1.4 means that, on average in Belgium, 1.4 persons are infected for each infected person.

**Using our model to analyze the outbreak if there was no intervention**

It is instructive to use our model fitted to the first 56 days of available data on confirmed cases in Belgium, to see what would happen if the outbreak were left to run its course, without public health intervention.

# time in days for predictions

t <- 1:120

# get the fitted values from our SIR model

fitted\_cumulative\_incidence <- data.frame(ode(

y = init, times = t,

func = SIR, parms = Opt\_par

))

# add a Date column and join the observed incidence data

fitted\_cumulative\_incidence <- fitted\_cumulative\_incidence %>%

mutate(

Date = ymd(sir\_start\_date) + days(t - 1),

Country = "Belgium",

cumulative\_incident\_cases = I

)

# plot the data

fitted\_cumulative\_incidence %>%

ggplot(aes(x = Date)) +

geom\_line(aes(y = I), colour = "red") +

geom\_line(aes(y = S), colour = "black") +

geom\_line(aes(y = R), colour = "green") +

geom\_point(aes(y = c(Infected, rep(NA, length(t) - length(Infected)))),

colour = "blue"

) +

scale\_y\_continuous(labels = scales::comma) +

labs(y = "Persons", title = "COVID-19 fitted vs observed cumulative incidence, Belgium") +

scale\_colour\_manual(name = "", values = c(

red = "red", black = "black",

green = "green", blue = "blue"

), labels = c(

"Susceptible",

"Recovered", "Observed incidence", "Infectious"

)) +

theme\_minimal()

The same graph in log scale for the *y*-axis and with a legend for better readability:

# plot the data

fitted\_cumulative\_incidence %>% ggplot(aes(x = Date)) + geom\_line(aes(y = I, colour = "red")) +

geom\_line(aes(y = S, colour = "black")) + geom\_line(aes(y = R, colour = "green")) +

geom\_point(aes(y = c(Infected, rep(NA, length(t) - length(Infected))), colour = "blue")) +

scale\_y\_log10(labels = scales::comma) + labs(y = "Persons (log scale)", title = "COVID-19 fitted vs observed cumulative incidence, Belgium") +

scale\_colour\_manual(name = "", values = c(red = "red", black = "black", green = "green",

blue = "blue"), labels = c("Susceptible", "Observed incidence", "Recovered",

"Infectious")) + theme\_minimal()

**More summary statistics**

Other interesting statistics can be computed from the fit of our model. For example:

* the date of the peak of the pandemic
* the number of severe cases
* the number of people in need of intensive care
* the number of deaths

fit <- fitted\_cumulative\_incidence

# peak of pandemic

fit[fit$I == max(fit$I), c("Date", "I")]

## Date I

## 87 2020-04-30 525315.7

# severe cases

max\_infected <- max(fit$I)

max\_infected / 5

## [1] 105063.1

# cases with need for intensive care

max\_infected \* 0.06

## [1] 31518.94

# deaths with supposed 0.7% fatality rate

max\_infected \* 0.007

## [1] 3677.21

Given these predictions, with the exact same settings and no intervention at all to limit the spread of the pandemic, the peak in Belgium is expected to be reached by the end of April. About 525,000 people would be infected by then, which translates to about 105,000 severe cases, about 32,000 persons in need of intensive care and up to 3,700 deaths

At this point, we understand why such strict containment measures and regulations are taken in Belgium!

Note that those predictions should be taken with a lot of caution. On the one hand, as mentioned above, they are based on rather unrealistic assumptions (for example, no public health interventions, fixed reproduction number \(R\_0\), etc.). More advanced projections are possible with the {projections} package, among others .On the other hand, we still have to be careful and strictly follow public health interventions because previous pandemics such as H1N1 and Spanish flu have shown that incredibly high numbers are not impossible!

The purpose of this article was to give an illustration of how such analyses are done in R with a simple epidemiological model. Those are the numbers our simple model produces and we hope they are wrong.

**Additional considerations**

As previously mentioned, the *SIR* model and the analyses done above are rather simplistic and may not give a true representation of the reality. In the following sections, we highlight five improvements that could be done to enhance theses analyses and lead to a better overview of the spread of the Coronavirus in Belgium.

**Ascertainment rates**

In the previous analyses and graphs, it is assumed that the number of confirmed cases represent all the cases that are infectious. This is far from reality as only a proportion of all cases are screened, detected and counted in the official figures. This proportion is known as the ascertainment rate.

The ascertainment rate is likely to vary during the course of an outbreak, in particular if testing and screening efforts are increased, or if detections methods are changed. Such changing ascertainment rates can be easily incorporated into the model by using a weighting function for the incidence cases.

Tim Churches demonstrates that a fixed ascertainment rates of 20% makes little difference to the modelled outbreak with no intervention, except that it all happens a bit more quickly.

**More sophisticated models**

More sophisticated models could also be used to better reflect real-life transmission processes. For instance, another classical model in disease outbreak is the *SEIR* model. This extended model is similar to the *SIR* model, where **S** stands for **S**usceptible and **R** stands for **R**ecovered, but the infected people are divided into two compartments:

1. **E** for the **E**xposed/infected but asymptomatic
2. **I** for the **I**nfected and symptomatic

These models belong to the continuous-time dynamic models that assume fixed transition rates. There are other stochastic models that allow for varying transition rates depending on attributes of individuals, social networking, etc.

**Modelling the epidemic trajectory using log-linear models**

As noted above, the initial exponential phase of an outbreak, when shown in a log-linear plot (the *y*-axis on a log scale and the *x*-axis without transformation), appears (somewhat) linear. This suggests that we can model epidemic growth, and decay, using a simple log-linear model of the form:

\[log(y)=rt+b\]

where *y* is the incidence, *r* is the growth rate, *t* is the number of days since a specific point in time (typically the start of the outbreak), and *b* is the intercept. In this context, two log-linear models, one to the growth phase before the peak, and one to the decay phase after the peak are fitted to the epidemic (incidence cases) curve.

The doubling and halving time estimates which you very often hear in the news can be estimated from these log-linear models. Furthermore, these log-linear models can also be used on the epidemic trajectory to estimate the reproduction number \(R\_0\) in the growth and decay phases of the epidemic.

**Estimating changes in the effective reproduction number \(R\_e\)**

In our model, we set a reproduction number \(R\_0\) and kept it constant. It would nonetheless be useful to estimate the current effective reproduction number \(R\_e\) on a day-by-day basis so as to track the effectiveness of public health interventions, and possibly predict when an incidence curve will start to decrease.

The {EpiEstim} package in R can be used to estimate \(R\_e\) and allow to take into consideration human travel from other geographical regions in addition to local transmission (Cori et al. 2013; Thompson et al. 2019).

**More sophisticated projections**

In addition to naïve predictions based on a simple *SIR* model, more advanced and complex projections are also possible, notably, with the {projections} package. This packages uses data on daily incidence, the serial interval and the reproduction number to simulate plausible epidemic trajectories and project future incidence.