PYU33C01, S. Hutzler, 2022
Assignment 2: Least Square Fits

The aim of this exercise is a numerical description of Irish Covid data.

The figure below shows daily numbers of reported Covid cases for about 600 days since the onset of the pandemic. Four "waves" of infections are clearly visible. How can we probe whether they feature exponential growth or decay?

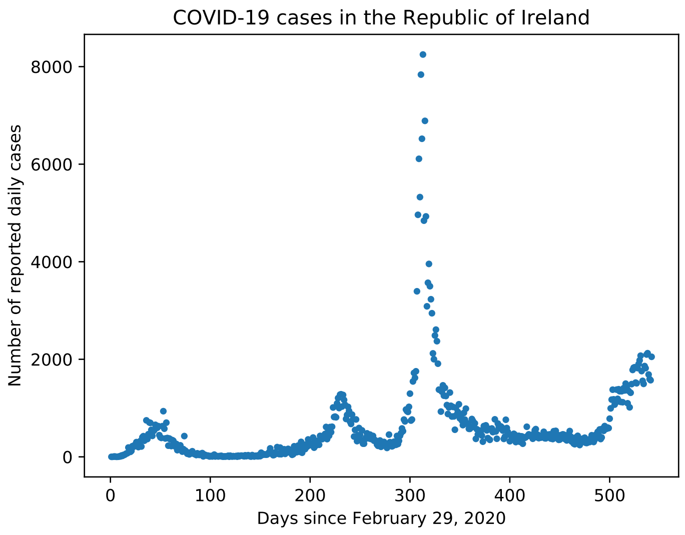

Figure 1: Number of daily reported Covid-19 cases in the Republic since the 29th of February, 2020. (Data downloaded from https://covid19ireland-geohive.hub.arcgis.com/)

The exponentially rising part of a wave should be described by

$$n(t) = n_0 \exp[\lambda(t - t_0)] \tag{1}$$

for the daily number of reported cases, $n(t)$. The constant λ needs to be determined from a least square fit. The value of t_0 marks the onset of a particular wave, with corresponding case number $n(t_0) = n_0$, eg. $t_0 = 0$ for the onset of the first wave. Note that in the fits below we will treat n_0 as a further fit parameter, so as not to give too much relevance to the number of recorded cases at time t_0.(This will account for daily fluctuations, e.g. there were generally less cases reported on Sundays.) A decaying wave has a negative (decay) constant λ.

Beginning and end of a wave are best determined by plotting the natural logarithm of the number of cases vs time, with

$$\log n(t) = \ln n_0 + \lambda(t - t_0). \tag{2}$$

Writing $a = \ln n_0 - \lambda t_0$ and $b = \lambda$ we can identify the functional form of $\ln n(t)$ as that of a straight line, i.e.

$$\ln n(t) = a + bt \tag{3}$$

We are now ready to analyse the Covid data set.

Assignment Tasks

- Download the dataset COVIDData.dat from Blackboard and display it by writing appropriate Python code. This should reproduce the figure shown above.
- By re-plotting the data as $\ln n(t)$ vs t you will observe that it may (roughly) be described by a sequence of straight line segments.
- Write Python code to carry out a linear regression/least square fit of the data, i.e fit each of the straight line segments to Eqn. (3) and record the two fit parameters a and b for each segments. You will need to choose appropriate values for t_0 (ie the start of the rise of a particular wave, or the start of its decline), and also set an appropriate range of times over which you want to fit.
- Produce a plot of $\ln(N)$ vs t which shows both the Covid data and the various straight line fits.
- Having determined the values of a and b for the first three waves you can use these values to determine the constants n_0 and λ of Eqn. (2). You can thus plot the Covid data again on a linear scale, but now add the exponential curves of Eqn. (1) to the plot.
- In your report include a table of the values of the various decay constants for the different waves.
- The fourth wave appears to look different. Why could this be? Can you think of a possible functional form to describe it better than an exponential could do?

- **Comment:** A timeline of the development of Covid in Ireland and the various political decisions that dealt with it is found at `https://en.wikipedia.org/wiki/COVID-19_pandemic_in_the_Republic_of_Ireland#First_Wave:_February%E2%80%93August` In principle such information can be used to determine the time-lags between the setting of country-wide lock-downs and the stop in the rise of Covid numbers. However, there were also partial lockdowns in place, which makes the assessment trickier.

Performing least square fits using Python

- You can use a scipy library to perform a non-linear least square fit of your data. You require the following commands:

  ```
  import scipy.optimize as optimization
  import numpy
  ```

  To load the data into your code you can use

  ```
  days, cases = np.loadtxt("COVIDData.dat", skiprows=1, unpack=True)
  ```

  where

  ```
   skiprows=1
  ```

  skips the first row of the input data and

  ```
   unpack=True
  ```

  returns a pair of arrays (here called: days, cases), one for each column.

  You can choose a selection of the data that you want to fit by defining for example

  ```
  firstWave = np.arange(0, 51)
  ```

  which selects the data for the days 0 to 51, when used in the form

```
 days[firstWave]
```

or

```
 cases[firstWave]
```

The fitting function (containing two or more fit parameters) is defined by

```
def func(x, a,b):
...
     return ...
```

Fitting of the selected data is done via

```
fitparameter=optimization.curve_fit(func, days[firstWave],
cases[firstWave])[0]
```

Placing [0] at the end specifies that only the values of the fitparameters are stored (and not also the covariance matrix of the fit)

The fit parameters returned by the fit are accessed via

```
par1=fitparameter[0]
par2=fitparameter[1]
```

(For more info on fitting simply google "Least-squares fitting in Python" or check out `https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.curve_fit.html` for a very detailed description of the fit package.)

- Submit a report (as pdf) **via Blackboard** which details your findings, and should include relevant figures. In addition submit your Python code as a *.py* file which should also print your name and student number onto the screen when executed.

submission deadline: Friday, October 21, 2022