Data Analytics Assignment – 3

COVID-19 Modelling

Name: G. Aravind **SR No: 18132**

Problem Statement

We use the SEIRV model with data of first dose vaccinations and immunity waning to model and predict COVID-19 cases. We fine tune the parameters (Beta, S_0, E_0, I_0, R_0 and CIR_0) to fit the model to the number of COVID cases between 16th March 2021 to 26th April 2021, and we use those fitted parameters to make future predictions and do various analysis with respect to the contact rate Beta.

Methodology

Formulation of the problem

We implement the following equations as part of the SEIRV model:

$$\Delta S(t) = -\beta(t)S(t)\frac{I(t)}{N} - \varepsilon\Delta V(t) + \Delta W(t)$$
$$\Delta E(t) = \beta(t)S(t)\frac{I(t)}{N} - \alpha E(t)$$
$$\Delta I(t) = \alpha E(t) - \gamma I(t),$$
$$\Delta R(t) = \gamma I(t) + \varepsilon\Delta V(t) - \Delta W(t).$$

And we fix the following parameters $Alpha^{-1}$ = 5.8 days (mean incubation period), $Gamma^{-1}$ = 5 days (mean recovery period) and Epsilon = 0.66 (vaccine efficacy) and N = 70 million (total population).

We also use the following constraints on the initial conditions:

a. R_0 should be between 15.6% and 36% of the population
b. CIR_0 should be between 12.0 and 30.0

And we use the following model for immunity waning:

a. Delta W(t) = R(0) / 30 for t between 16 March 12021 and 15 April 2021
b. Delta W(t) = Delta R(t-180) + Epsilon * Delta V(t - 180) when t is larger than 11 September 2021

We define CIR(t) = CIR(0) * $T(t_0)$ / T(t)

Pre-processing

I have done the following pre-processing on the given dataset in order to make the data usable to solve this problem:

a. After reading the CSV data file as a Pandas DataFrame, I keep only the columns pertaining to "Date", "Confirmed", "Tested" and "First Dose Administered", and I drop the remaining columns as they are not needed to solve this problem.
b. I convert the "Confirmed", "Tested" and "First Dose Administered" columns from cumulative to per-day values
c. I perform 7-day averaging on these 3 columns for days starting from 16th March 2021 to the end. To be able to perform 7-day averaging, I use the data from 9th March 2021 for this averaging purpose.
d. I extrapolate the "Tested" column till 31st December, by using the last day data of "Tested" till 31st December.
e. I extrapolate the "First Dose Administered" data by assuming 2 lac vaccinations per day from 27th April 2021 to 31st December 2021.

Functions

I write suitable functions to perform some repetitive tasks, which are defined and explained below:

a. **generateTimeSeries**: This function takes in the model parameters and generates the time series of the SEIRV model for a given number of days (*ndays*). It also performs 7-day averaging on the generated time-series, and also computes the estimated number of cases e(t) = E(t)/CIR(t), and returns the same.

b. **computeLoss**: This function takes as input the model parameters and generates a time series using the *generateTimeSeries* function. It then performs 7-day averaging on the estimated cases e(t) and compares it with the actual number of cases (ground-truth) for the given period. It also computes the loss between the generated time series and the ground truth using the following loss function:
$$l(P) = \frac{1}{42} \sum_{t=\text{March 16}}^{\text{April 26}} (\log(\bar{c}(t)) - \log(\alpha\bar{e}(t)))^2$$

c. **computeGradient**: This function takes as input the model parameters and computes the loss for this set of parameters internally using the *computeLoss* function. It then estimates the gradient of these parameters using the perturbation method, by perturbing the value of Beta by 0.01, CIR_0 by 0.1 and S_0, E_0, I_0, R_0 by 1 and checking the value of the function. It then returns the estimated gradient to the main function in the form of a numpy array.

d. **optimizeParams**: This function takes as input the initial model parameters, and performs gradient descent on the parameters repeatedly till the loss is down to less than 0.01. It has various options for adjusting the stepsize, if required, to speed up the convergence of the gradient descent. It then returns the optimal parameters when the condition on the loss function is reached.

e. **plotSEIRWithoutImmunityWaning** and **plotSEIRWithImmunityWaning**: These functions take in a given set of parameters and generate and plot the predicted time series of S, E, I and R values, with and without assuming immunity waning.

f. **futureTimeSeries**: This function takes in as input the model parameters and is used to generate future predictions with varying Beta values. Compared to the earlier *generateTimeSeries* function, this function has some additional features like performing Closed Loop Control, computing average new cases per week, computing average new cases each day.

g. **plotNewCasesEachDayOpenAndClosedLoop**: This function takes as input the model parameters and plots the future estimated COVID-19 cases using the *futureTimeSeries* function with 5 different settings – 4 different values of Beta using Open Loop Control settings and one Closed Loop Control Setting which internally adjusts the Beta Values based on previous week's average cases. It then plots these 5 scenarios along with the ground truth of the actual number of reported cases.

h. **plotSusceptibleOpenAndClosedLoop**: This function takes model parameters as input, and similar to above function, plots the evolution of the susceptible population for both Open Loop and Closed Loop Control settings.

i. From the script, the above functions are called appropriately to generate the model parameters and plot the different graphs.

Results

The results after optimizing the model parameters are given below. The model gives a loss of less than 0.01 for the below parameters:

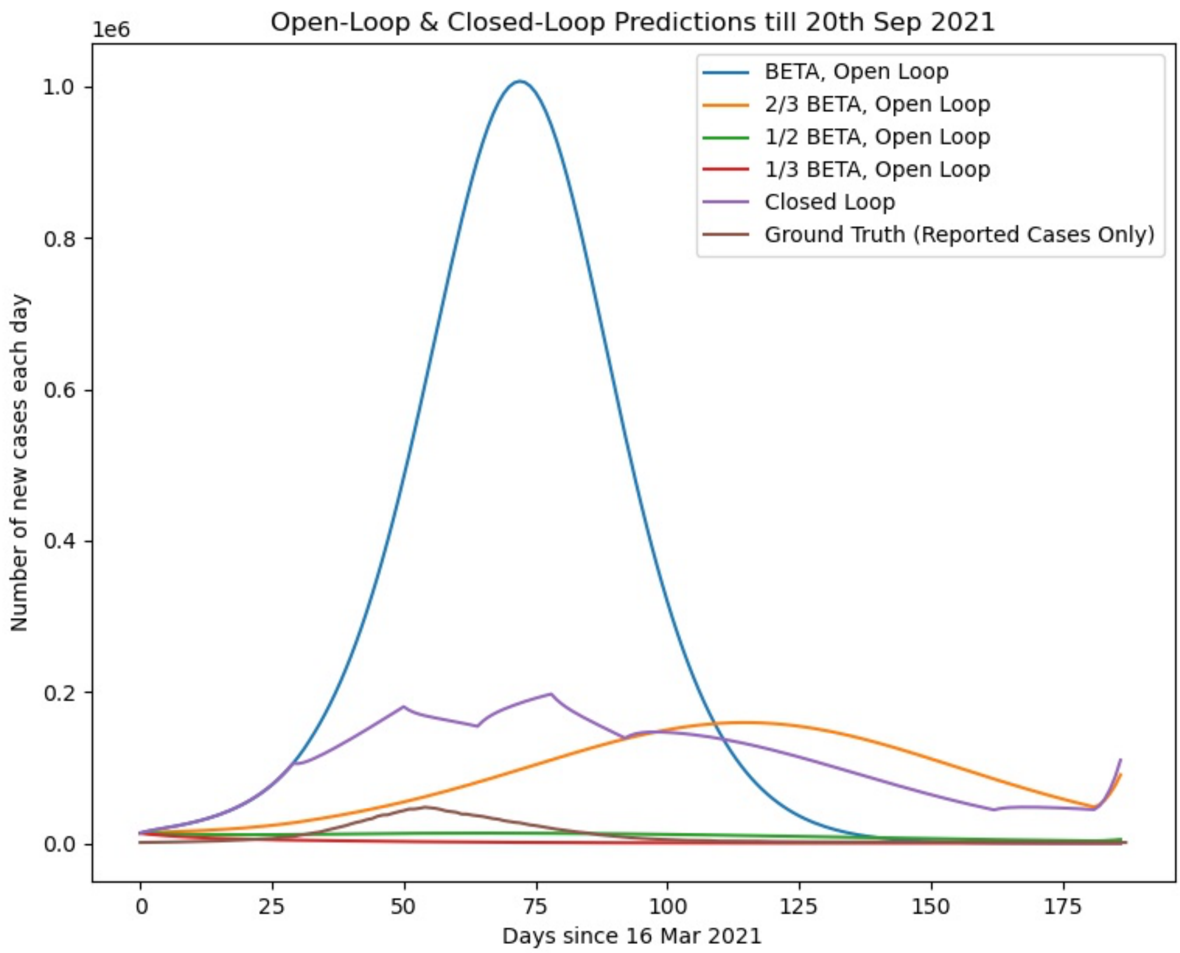
- **a.** Beta **= 0.449722551**
- **b.** S_0 = **48999999.9 (49 million)**
- **c.** E_0 = **76999.9180 (77 thousand)**
- **d.** I_0 = **76999.9182 (77 thousand)**
- e. R_0 = **20852999.9 (20.8 million)**
- **f.** CIR_0 = **12.8716990**

The corresponding loss for the above parameters is **0.0029778394997330934** which is less than the threshold of *0.01.*

I have also used these parameters to generate future predictions till 31^{st} December using different Beta values (both open and closed loop control). As asked in the question, I have also plotted the results till 20^{th} September, and also compared the predictions with the ground truth (reported cases) as shown below.

Plots and Observations

The first plot (attached below) compares the future predictions (average new cases each day) of both open-loop and closed-loop methods (using different Beta values) with the number of reported cases (ground truth) till 20^{th} September.



Observations:

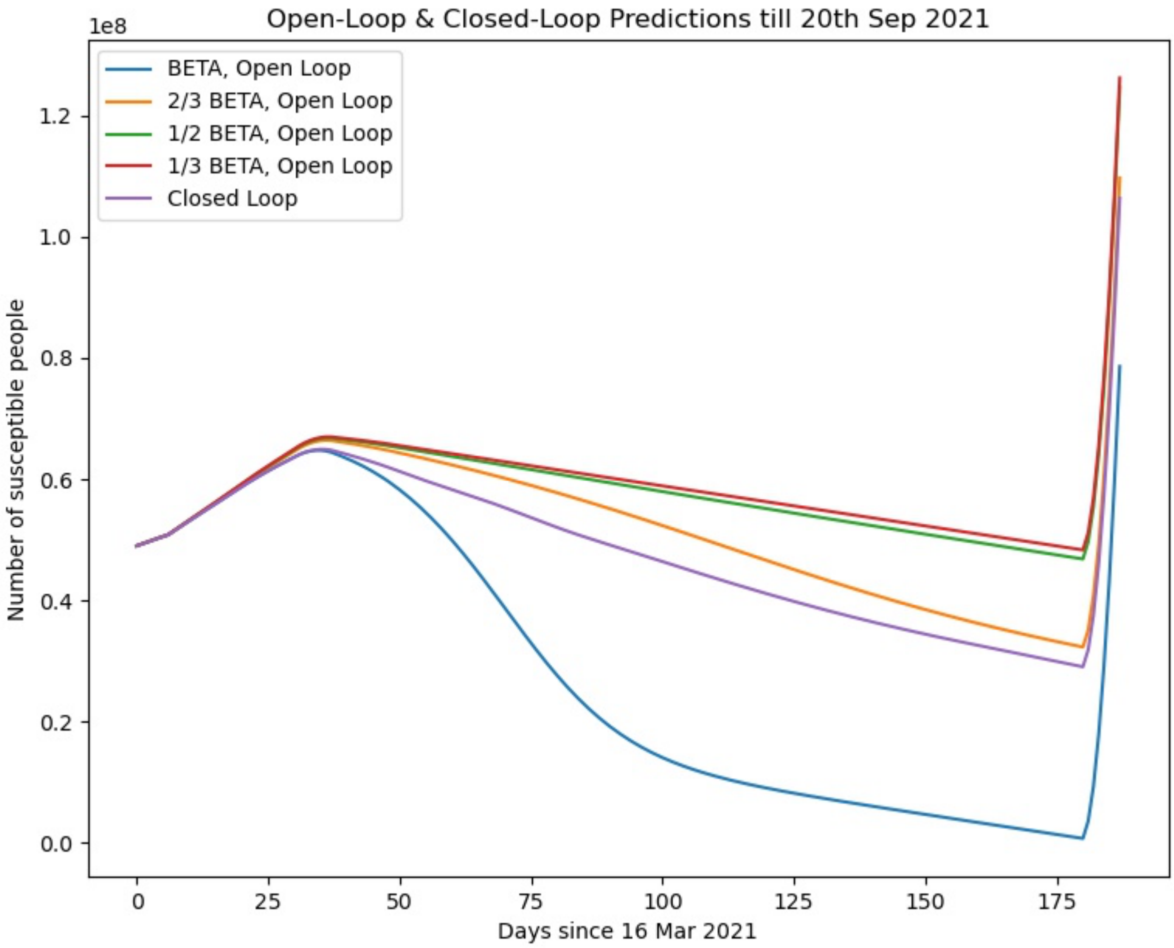
a. As expected, the new daily cases are highest for BETA, followed by 2/3 BETA, 1/2 BETA and 1/3 BETA. The graph for Closed-Loop control is also shown above in Purple color. Higher BETA values means that due to high contact rate, more people will be infected, but due to the rapid growth of the exposed, infected and recovered population, the curve also bends down very quickly because susceptible population also reduced quickly for larger BETA values.

b. The Closed-Loop is noticeably rugged with sharp edges, and that is because of the dynamically adjusting BETA values.

c. The ground truth is plotted in Brown color. We can observe that the ground truth is smaller than most predictions, and that is because the reported cases are always much lesser than the actual cases, especially in the peak of a pandemic.

d. For the cases of 1/2 BETA and 1/3 BETA, we can observe that the daily cases plummets down and steadily goes to zero, which means that the contact rates are too small for the disease to spread to other people quickly, and eventually the pandemic dies down because of the very low transmissibility.

The second plot (as given below) shows the evolution of the susceptible population during till 20^{th} September 2021.



Observation:

As discussed above, the susceptible population reduces down more quickly for higher values of BETA (like the blue graph) as compared to lower values of BETA (like the orange, red and green graphs). This is because if BETA value is higher, the virus quickly infects most of the population and hence the susceptible population (which is not yet exposed) reduces down quickly.