Mohit Kumar
SR No. 04-01-03-10-51-21-1-19825
MTech Artificial Intelligence

Assignment 5: COVID19 modelling

python packages used:

- numpy
- pandas
- datetime
- math
- matplotlib

Data Preprocessing I performed the following data preprocessing on the given dataset:

1. Only keep the columns corresponding to "Date", "Confirmed", "Tested" and "First Dose", and drop the rest of the columns.
2. Transform the data of "Confirmed", "Tested" and "First Dose" columns from cumulative to per day values.
3. Perform 7 day average on these three columns for days starting from March 16, 2021 to the end. In order to be able to perform 7-day averaging, I use the data from March 9, 2021.
4. Extrapolate the "Tested" column till December 31, 2021 by using the last day data of "Tested" till December 31, 2021.
5. Extrapolate the "First Dose" data by assuming 2 lac vaccinations per day from April 27, 2021 to December 31, 2021.

Implementation details I have implemented the following functions:

generate_timeseries This function takes as input the model parameters and generates the time series of the SEIRV model for a given number of days, ndays. It also calculates the 7 day average on the generated time-series, and the estimated number of cases $e(t) = E(t)/CIR(t)$.

future_timeseries This function takes as input the model parameters and generates future predictions with varying β values according to open loop or closed loop. It also computes average new cases each day.

calculate_loss This function takes as input the model parameters and generates a time series using the generate_timeseries function. It then performs a 7 day average on the estimated no. of cases e(t) and compares it with the actual number of cases for the given time duration. It then calculates the loss between the generated time series and the ground truth using the following loss function.

$$l(P) = \frac{1}{42} \sum_{t=\text{ March 16}}^{\text{April 26}} (\log(\bar{c}(t)) - \log(\alpha \bar{e}(t)))^2$$

calculate_gradient This function takes as input the model parameters and computes the loss for the given set of parameters internally using the calculate_loss function. It then calculates the gradient of the loss with respect to these parameters using the perturbation method, by perturbing the value of Beta by $0.01, \mathrm{CIR}_0$ by 0.1 and E0, I0, R0 by 1. It then returns the computed gradient to the main function as a numpy array.

gradient_descent This function takes as input the initial model parameters, and performs gradient descent on the parameters until the loss is less than 0.01. It then returns the optimal parameters. **As S0 + E0 + I0 + R0 = N. I only optimize the values of E0, I0 and R0 and calculate S0 = N - E0 + I0 + R0. Also after updating the values of the parameters, I project them inside their feasible region using project_params function.**

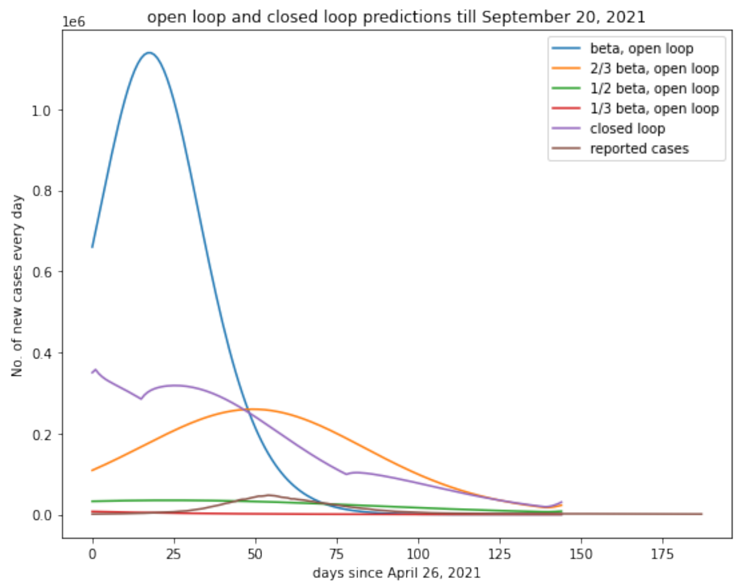
plot_new_cases_per_day This function takes as input the model parameters and plots the estimated future COVID-19 cases using the future_timeseries function in five different scenarios, 4 different values of β using open loop control and one closed loop control which internally adjusts the β values based on average cases of the previous week. It then plots these five scenarios along with the actual number of reported cases.

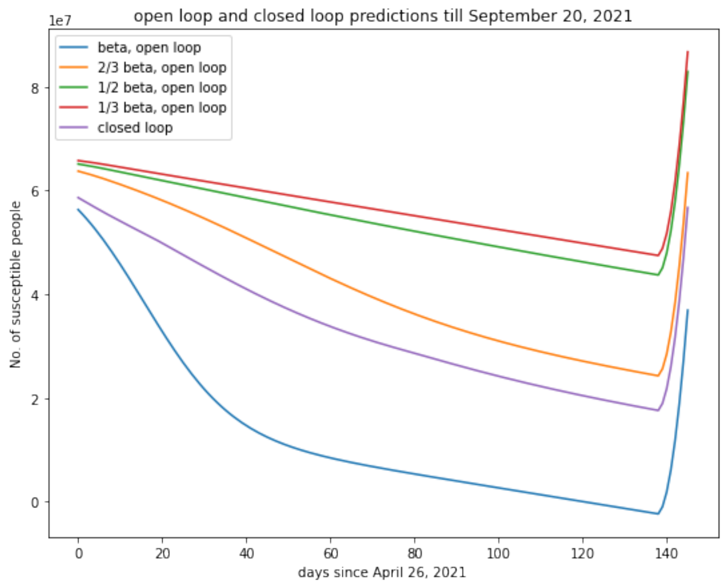
plot_susceptible This function takes as input the model parameters and plots the evolution of the susceptible population for both open loop and closed loop controls.

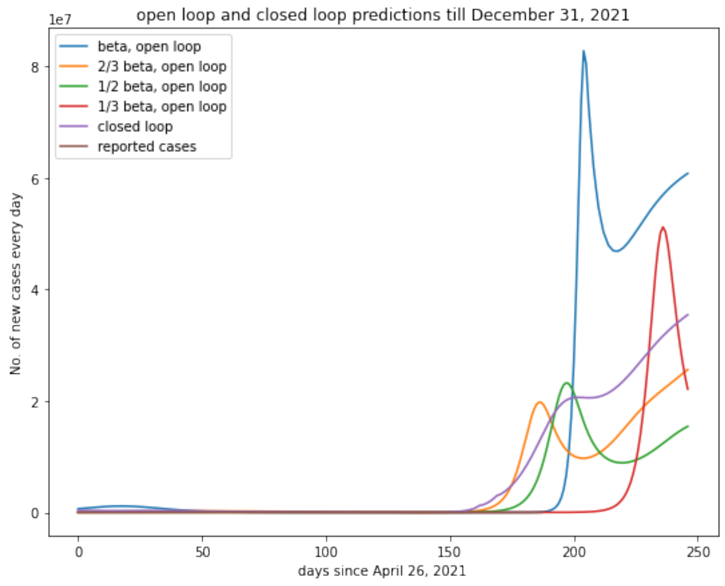
Results
model parameters are as follows:

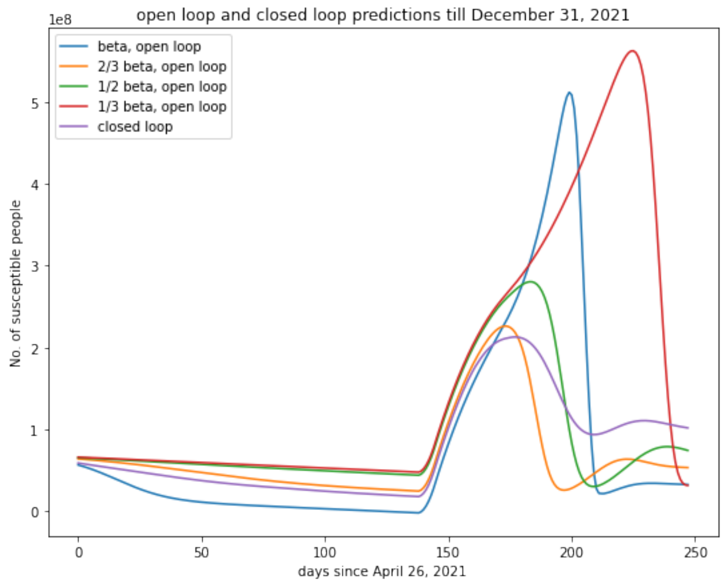
parameter	initial value	optimal value
β	0.43557	0.4633046404112653
S0	68515002	59215002
E0	199999	199999
I0	84999	84999
R0	1200000	10500000
CIR0	29	28.90

The corresponding loss for the above parameters is 0.008239823642672386.
I have used these parameters to generate future predictions till September 20, 2021 using different β values for both open and closed loop control and compared the predictions with the reported cases. The first plot compares the future predictions, average new cases each day of both open loop and closed loop controls using different β values with the number of reported cases till September 20, 2021. The second plot shows the evolution of the susceptible population till September 20, 2021.


open loop and closed loop predictions till September 20, 2021
1e6
No. of new cases every day
days since April 26, 2021
beta, open loop
2/3 beta, open loop
1/2 beta, open loop
1/3 beta, open loop
closed loop
reported cases
0.0
0.2
0.4
0.6
0.8
1.0
0
25
50
75
100
125
150
175


open loop and closed loop predictions till September 20, 2021
1e7
No. of susceptible people
days since April 26, 2021
beta, open loop
2/3 beta, open loop
1/2 beta, open loop
1/3 beta, open loop
closed loop
0
2
4
6
8
0
20
40
60
80
100
120
140





Observations

- As expected, the daily new cases are highest for β, followed by 2/3 β, 1/2 β and 1/3 β. Due to high contact rate, more people will get infected, but due to the rapid rise in the exposed, infected and recovered population, the curve also bends down very quickly as susceptible population also gets reduced quickly for larger β values.
- The plot for closed loop is noticeably rugged with sharp edges, because of the dynamically changing β values.

- We can observe that the reported cases is smaller than most of the predictions.
- For the cases of 1/2 β and 1/3 β, we can observe that the daily cases plummets down and steadily goes to zero, implying that the contact rates are too small for the disease to spread to others quickly and eventually the pandemic dies down because of the very low transmissibility.
- The susceptible population reduces down more quickly for higher values of β as compared to lower values of β. This is because if β value is higher, the virus quickly infects most of the population and hence the susceptible population which is not yet exposed reduces more quickly.