REPORT

ASSIGNMENT 3

COVID-19 MODELLING

Name : KAUSTUV RAY

Department : ELECTRICAL ENGINEERING

Discipline / Stream : Artificial Intelligence

SR No : 04-03-06-10-51-21-1-19308

IMPLEMENTATION SUMMARY

The following are the functions used in the program-

1. projection(params)

Parameters-
params: list containing β, S(0), E(0), I(0), R(0), CIR(0)

return type: list
It returns a list containing β, S(0), E(0), I(0), R(0), CIR(0).

summary: This function makes sure that the parameters are within specified constraints. If parameters do not satisfy constraints, then the parameter is set to its limit.

2. get_training_data():

Parameters-
None

return type: pandas.core.frame.DataFrame
It returns a DataFrame containing t, ΔV, $\overline{c}$, T.

summary: This function returns a DataFrame containing t, ΔV, $\overline{c}$, T (where t is the day number starting from 16 March 2021. ΔV is the number of vaccinations per day. $\overline{c}$ is running seven-day average of Δconfirmed(t). $T(t)$ is the average number of tests done during the past 7 days. The DataFrame contains data from 16 March 2021 to 26 April 2021.

3. running_average(a)

Parameters-
a: list of numbers.

return type: numpy.ndarray

summary: This function returns a numpy array containing the seven-day running average of the list ***a***. When there are less than seven available days, the average of the available days is taken.

4. SEIRV_model(initial_values)

Parameters-
initial_values: list containing β, S(0), E(0), I(0), R(0), CIR(0)

return type: list

summary: We create four lists ***S, E, I, R*** that contain S(t), E(t), I(t), R(t) values. For each value of ***t***, we calculate $\Delta S(t)$, $\Delta E(t)$, $\Delta I(t)$, $\Delta R(t)$ using the equations given in problem description 1.(a), and add with the previous values before storing in ***S, E, I, R***.

In the function, some values of ***S*** were negative. So, a condition was added so that the negative value goes to zero, and the remaining values are scaled so that their sum is 70 million.
The function returns a list that contains S(t), E(t), I(t), R(t) values from t=0 to t=41. (t=0 refers to 16 March 2021, and t = 41 refers to 26 April 2021)

5. loss_function(params)

Parameters-
params: list containing β, S(0), E(0), I(0), R(0), CIR(0)

return type: float

summary: The cases-to-infections ratio is calculated for t = 0 to t = 41 using the expression CIR(t) = CIR(0)*T(t_0)/T(t). where $T(t)$ is the average number of tests done during the past 7 days and t0 is 16 March 2021.
Then we calculate the running seven-day average of $\alpha e(t)$ (where $e(t) = E(t)/\mathrm{CIR}(t)$)
Then we use the loss function as given in the problem description, and return the mean squared error.

6. gradient(params)

Parameters-
params: list containing β, S(0), E(0), I(0), R(0), CIR(0)

return type: numpy.ndarray

summary: For estimating the gradient, β is perturbed on either side by ±0.01, CIR(0) is perturbed on either side by ±0.1, and all other parameters by ±1. It returns a numpy array containing the gradients for each parameter.

7. gradient_descent(params)

Parameters-
params: list containing β, S(0), E(0), I(0), R(0), CIR(0)

return type: numpy.ndarray

summary: The ***get_training_data()*** function is called to extract the values of ΔV, $\bar{c}$, T. Then the loss is calculated using ***loss_function***. The parameters are updated till the loss < 0.01.
Then the function returns a numpy array containing the optimal values of β, S(0), E(0), I(0), R(0), CIR(0).

8. new_reported_cases()

Parameters- None
return type: list

summary: This function returns a list of Δconfirmed values from 16 March 2021 onwards.

9. plot_cases(result, beta)

Parameters-
result: list
result contains the predictions for ***S,E,I,R*** till 31 December 2021.
beta: float

summary: Find the average CIR value for the training period. Get ***E*** from ***result*** and reported cases by using ***new_reported_cases***(). Divide ***E*** by average CIR to get the number of cases. Then plot a graph that shows the number of new cases predicted on each day till 31 December 2021. It also shows the number of new reported cases till 20 September 2021.

10. plot_S_fraction(result, beta)

Parameters-
result: list
result contains the predictions for ***S,E,I,R*** till 31 December 2021.
beta: float

summary: Get ***S*** from ***result***. Then plot a graph that shows the evolution of the fraction of the susceptible population.

11. open_loop_control(params)

Parameters-
params: list containing β, S(0), E(0), I(0), R(0), CIR(0)

return type: list

summary: This is similar to ***SEIRV_model().*** But here ΔV(t) is taken as 200000 for t >= 42. For immunity wanning we set $\Delta W(t) = \Delta R(t - 180) + \varepsilon \Delta V(t - 180)$, when t is larger than 11 September 2021.
In the function, some values of ***S*** and ***R*** were negative. So, a condition was added so that the negative value goes to zero, and the remaining values are scaled so that their sum is 70 million.
The function returns a list that contains S(t), E(t), I(t), R(t) values till 31 December.

12. closed_loop_control(params)

Parameters-
params: list containing β, S(0), E(0), I(0), R(0), CIR(0)

return type: list

summary: This is similar to ***open_loop_control().*** But here the β values are updated as mentioned in the problem description. The updating starts from 27 March.

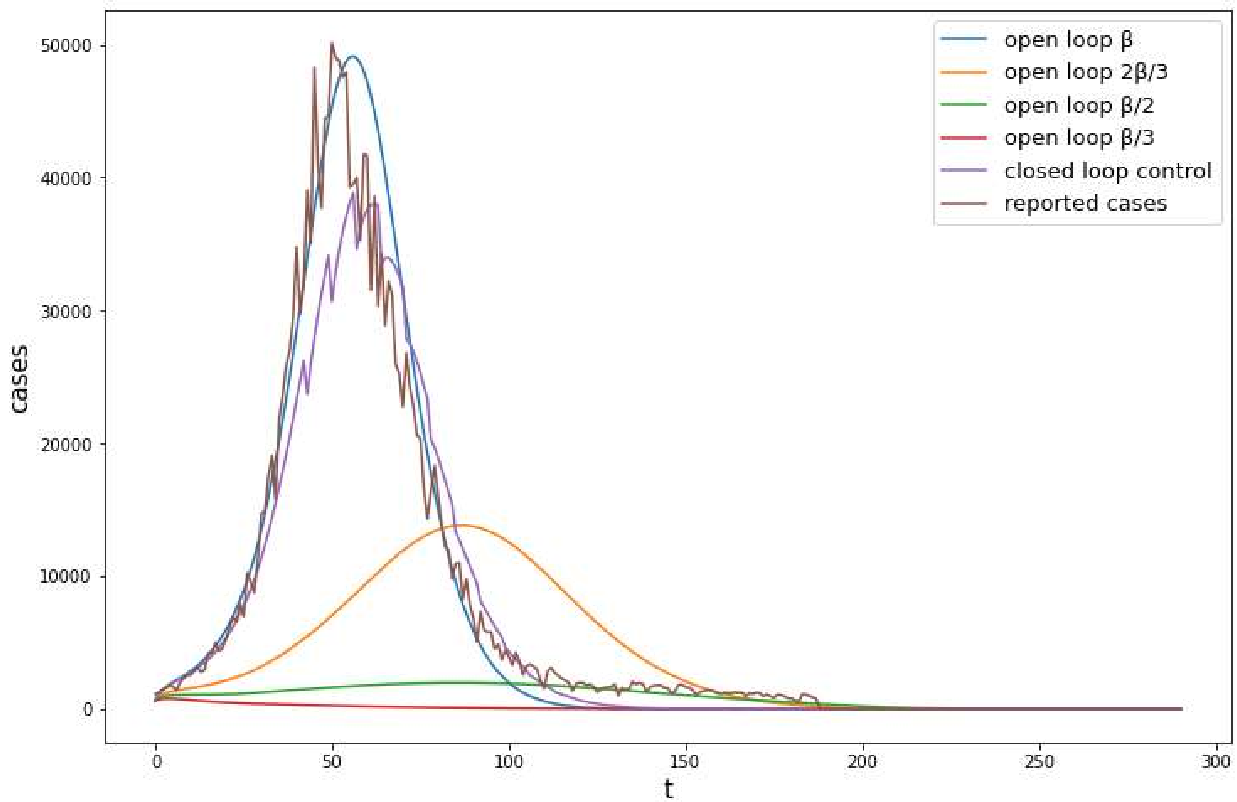
OUTPUT

GRADIENT DESCENT: iterations: 34   loss = 0.007057469495405773
best_params [0.470504737336695, 47214999.99999966, 111999.99999546476, 272999.99998825626,
22399999.999999817, 29.549793811281727]

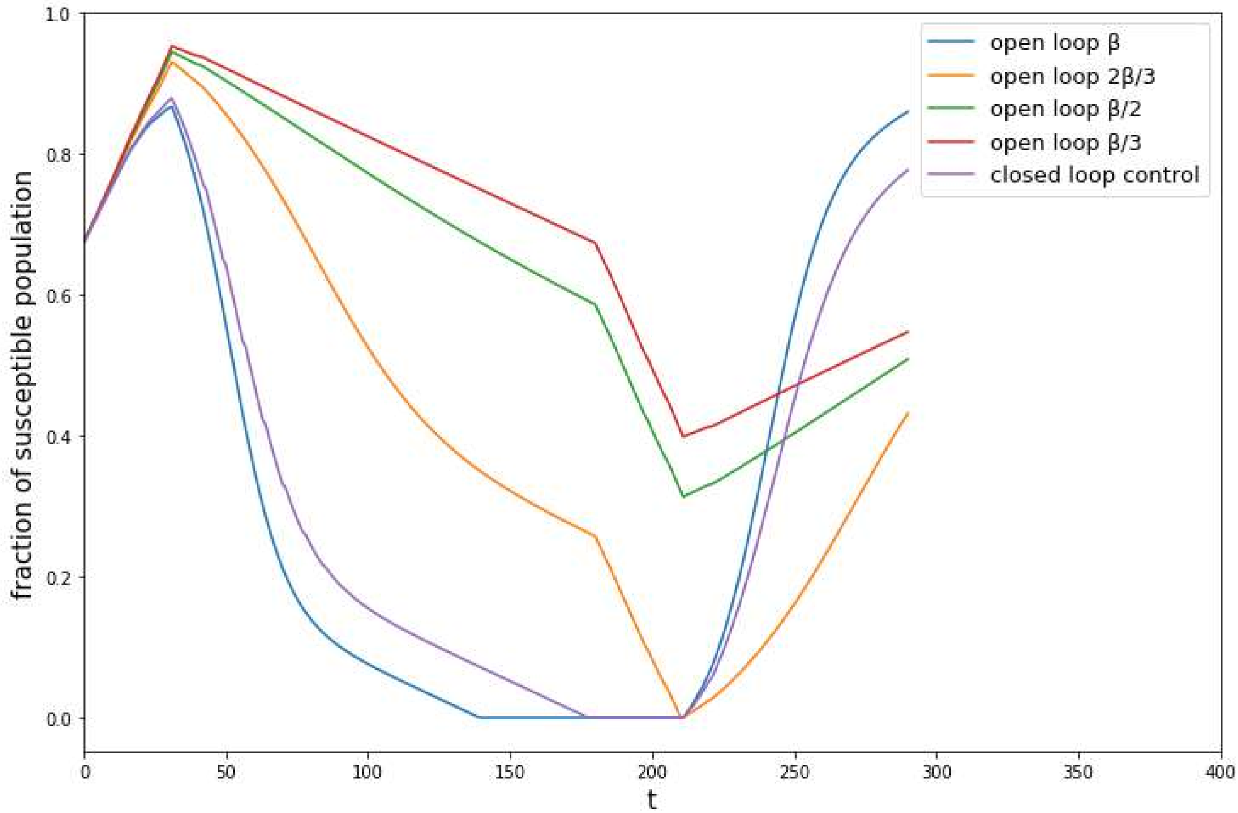
PLOTS

All the plots below are from t = 0 (16 March 2021) to t = 290 (31 December 2021)

number of new cases predicted and reported for open loop and closed loop control



Evolution of the fraction of the susceptible population for open loop and closed loop control



CONCLUSION

I learnt about different methods used to model COVID-19. I understood how the number of cases changes with different contact rates and initial values. I used pandas to create and manipulate Dataframes. I was able to get a loss of 0.007057469495405773 while fitting the model. I used the model to predict the number of new cases of infection till 31 December 2021.