PROJECT REPORT

FM1015 Modelling of Dynamic Systems

University of South-Eastern Norway

Department of Electrical Engineering, Information Technology and Cybernetics

Faculty of Technology, Natural Science and Maritime Sciences, Campus Porsgrunn

Submitted by:

Ozgur Yalcin	-	EET, 238769
Vahid Arzhang	-	EPE, 238806
Yousef Hedayati	-	EPE, 238809
Dhanush Wagle	-	EPE, 238810

Submitted to:

Prof Bernt Lie

November 13th, 2020

Problem 1: Getting familiar with infection models

Dynamic SIR Model

We use the following model for the number of susceptible, infection, and recovered individuals[1]:

$$\frac{dS}{dt} = \dot{S}_i - \dot{S}_e - \frac{k_i}{N} IS \tag{1}$$

$$\frac{dI}{dt} = \dot{I}_i - \dot{I}_e - \frac{k_i}{N} IS - k_r I \tag{2}$$

$$\frac{dR}{dt} = \dot{R}_i - \dot{R}_e + k_r I \tag{3}$$

For problem 1, immigration and emigration are neglected, and the total number of people is taken as constant. i.e. $\dot{S}_i = \dot{I}_i = \dot{R}_i \equiv 0$ and $\dot{S}_e = \dot{I}_e = \dot{R}_e \equiv 0$ and $N = S + I + R$.

We can simplify the model and, also write it in per capita form [1]:

$$\frac{d\check{S}}{dt} = -k_i \check{I}\check{S} \tag{4}$$

$$\frac{d\check{I}}{dt} = (k_i \check{S} - k_r)\check{I} \tag{5}$$

$$\frac{d\check{R}}{dt} = k_r \check{I} \tag{6}$$

1. SIR model is implemented in python as:

```
N_sim =int((14 - 3)/0.1) + 1

for k in range(0, N_sim-1):

    # ODE's for SIR Model
        dS_dt = -(ki/N)*I[k]*S[k]
        dI_dt = ((ki/N)*S[k] - (kr))*I[k]
        dR_dt = kr*I[k]

        # State updates using the Forward Euler's method
        S[k+1] = S[k] + dS_dt *Ts
        I[k+1] = I[k] + dI_dt *Ts
        R[k+1] = R[k] + dR_dt *Ts
```

2. The constants values in the equations were taken from Section 3.8 in [1] as: $N = 763$, $k_i = 1.9075$, $k_r = 0.3$, and initial condition as: $I_0 = 25$, $S_0 = N - I_0 = 738$, $R_0 = 0$. And the model is simulated in python.

3. The results of the simulation are shown in figure 1:

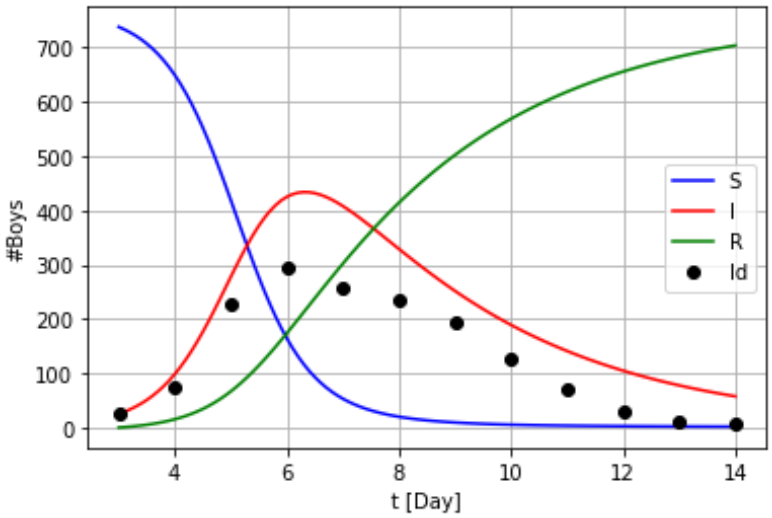


Figure 1: Boarding school measles infection from simulation

We have from Fig 5 in [1]:

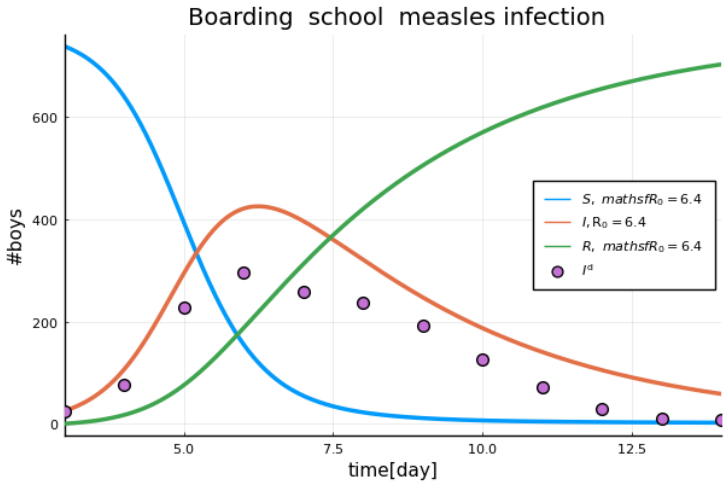


Figure 2: Boarding school measles infection figure from [1]

Comparing figure 1 and 2, the exact same plot for SIR model implemented in Julia and Python is seen where the susceptible reaches zeros in about 10th day and day 14th are the time for total recovery and comparing with real values of infected in both figures, the infection curve is quite similar with a slightly higher infected cases number in the simulation than the real values.

Problem 2: Covid-19 model for Italy based on the extended SEIR dynamic model

1. In this case by a realistic and closer view, we should add some cumulative, the symbol C to refer to confirmed infected, the symbol A for asymptomatic infected and the symbol CC to reach a more accurate diagram, so the following equations are the base equations to create this model.

$$\mathcal{E}_{\mathrm{i}}: S \xrightarrow{\mathrm{I+A}^{k_{\mathrm{i}}}} \mathrm{E}, \qquad r_{\mathrm{i}} = k_{\mathrm{i}}\left(\check{I} + \check{A}\right)\check{S}$$

$$\mathcal{E}_{\mathrm{e}}: \mathrm{E} \rightarrow^{k_{\mathrm{e}}} \mathrm{I}, \qquad r_{\mathrm{e}} = k_{\mathrm{e}}\check{E}$$

$$\mathcal{E}_{\mathrm{c}}: \mathrm{I} \rightarrow^{k_{\mathrm{c}}} \mathrm{C}, \qquad r_{\mathrm{c}} = k_{\mathrm{c}}\check{I}$$

$$\mathcal{E}_{\mathrm{a}}: \mathrm{I} \rightarrow^{k_{\mathrm{a}}} \mathrm{A}, \qquad r_{\mathrm{a}} = k_{\mathrm{a}}\check{I}$$

$$\mathcal{E}_{\mathrm{cr}}: \mathrm{C} \rightarrow^{k_{\mathrm{r}}} \mathrm{R}, \qquad r_{\mathrm{cr}} = k_{\mathrm{r}}\check{C}$$

$$\mathcal{E}_{\mathrm{ar}}: \mathrm{A} \rightarrow^{k_{\mathrm{r}}} \mathrm{R}, \qquad r_{\mathrm{ar}} = k_{\mathrm{r}}\check{A}$$

$$\begin{pmatrix} -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 \\ 0 & 0 & 0 & 0 & -1 & 1 \end{pmatrix} \begin{pmatrix} S \\ E \\ I \\ C \\ A \\ R \end{pmatrix} \leftarrow \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

$$\begin{pmatrix} r_{S,g} \\ r_{E,g} \\ r_{I,g} \\ r_{C,g} \\ r_{A,g} \\ r_{R,g} \end{pmatrix} = \upsilon^{T} \begin{pmatrix} r_{\mathrm{i}} \\ r_{\mathrm{e}} \\ r_{\mathrm{c}} \\ r_{\mathrm{a}} \\ r_{\mathrm{cr}} \\ r_{\mathrm{ar}} \end{pmatrix} = \begin{pmatrix} -1 & 0 & 0 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & -1 & 0 & 0 \\ 0 & 0 & 1 & 1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix} \begin{pmatrix} r_{\mathrm{i}} \\ r_{\mathrm{e}} \\ r_{\mathrm{c}} \\ r_{\mathrm{a}} \\ r_{\mathrm{cr}} \\ r_{\mathrm{ar}} \end{pmatrix} = \begin{pmatrix} -r_{\mathrm{i}} \\ r_{\mathrm{i}} - r_{\mathrm{e}} \\ r_{\mathrm{e}} - r_{\mathrm{c}} - r_{\mathrm{a}} \\ r_{\mathrm{c}} + r_{\mathrm{a}} - r_{\mathrm{cr}} \\ -r_{\mathrm{ar}} \\ r_{\mathrm{cr}} + r_{\mathrm{ar}} \end{pmatrix}$$

After the above computations we have:

$$\frac{dS}{dt} = -r_{\mathrm{i}} = \frac{-k_{\mathrm{i}}}{N} * (I + A) * S \tag{7}$$

$$\frac{dE}{dt} = r_{\mathrm{i}} - r_{\mathrm{e}} = \frac{k_{\mathrm{i}}}{N} * (I + A) * S - k_{\mathrm{e}} * E \tag{8}$$

$$\frac{dI}{dt} = r_{\mathrm{e}} - r_{\mathrm{c}} - r_{\mathrm{a}} = k_{\mathrm{e}} * E - k_{\mathrm{c}} * I - k_{\mathrm{a}} * I \tag{9}$$

$$\frac{dC}{dt} = r_{\mathrm{c}} + r_{\mathrm{a}} - r_{\mathrm{cr}} = k_{\mathrm{c}} * I - k_{\mathrm{r}} * C \tag{10}$$

$$\frac{dA}{dt} = -r_{\mathrm{ar}} = k_{\mathrm{a}} * I - k_{\mathrm{r}} * A \tag{11}$$

$$\frac{dR}{dt} = r_{\mathrm{cr}} + r_{\mathrm{ar}} = k_{\mathrm{r}} * C + k_{\mathrm{r}} * A \tag{12}$$

For the cumulative confirmed we could neglect the rate of recovered people and get the derivative of CC as:

$$\frac{dCC}{dt} = k_{\mathrm{c}} * I \tag{13}$$

2. The model is simulated in python as shown in Appendix A and after simulating and solving the obtained formulas by the forward Euler method the result of the model is created as shown in the appendix. From the comparison between our model by the time evolution of cumulative infected and from fig 22 in [1], we can conclude that there is a decent similarity between these two curves.

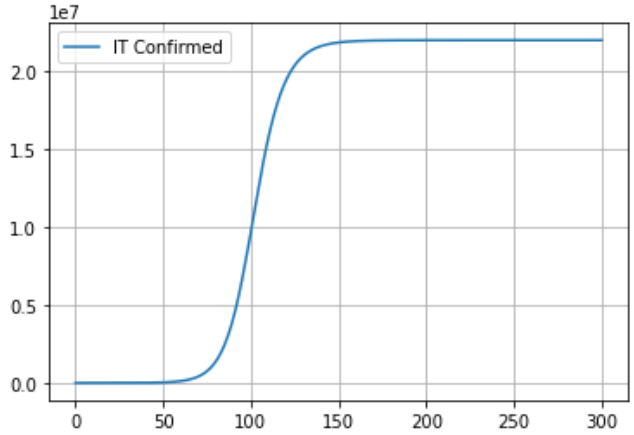


Figure 3: Cumulative number of confirmed infected in Italy

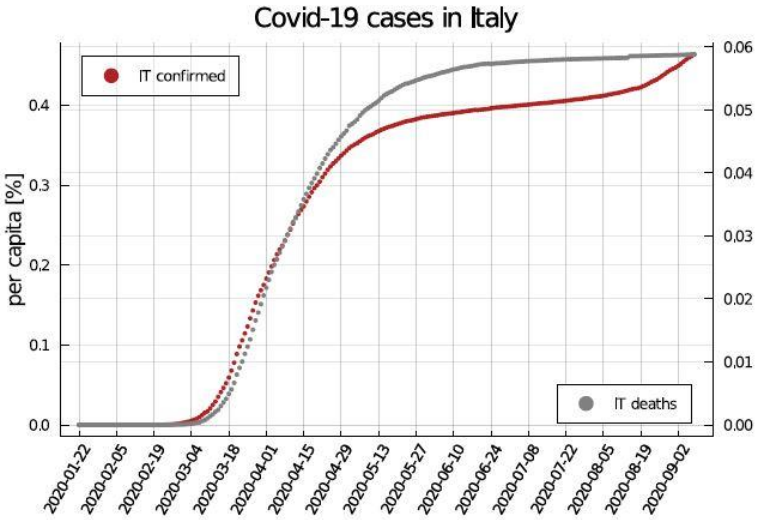


Figure 4: Cumulative number of confirmed infected in Italy [1]

3. The model can be simplified by removing E compartment in the SEICAR model that becomes SICAR model which means that some fraction of susceptibles are now moved to infected directly rather than exposed, and for new model R_0 changes since there is no k_e which will change the k_i to 0.225, and the results of the simulation is shown in the graph below which indicates the same curve for both models.

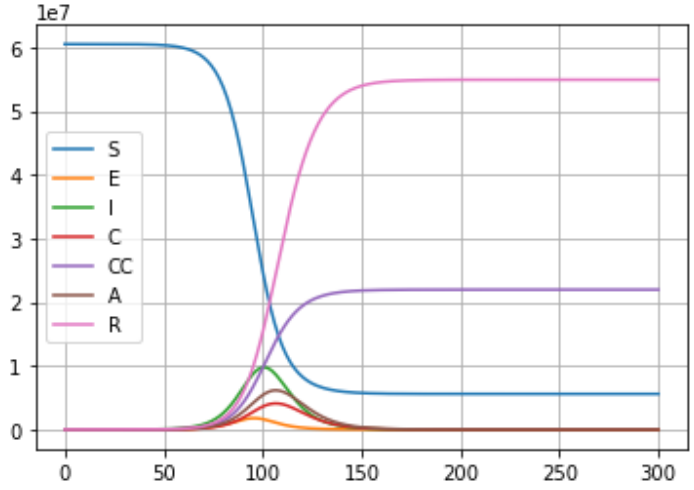


Figure 5: SEICAR Model

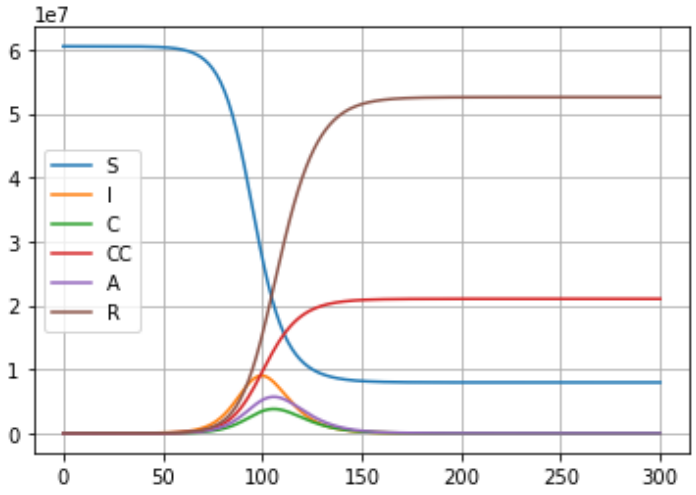


Figure 6: SICAR Model

Problem 3. Expanding Covid-19 model for Italy with mitigation model

1. Considering $u = \frac{100+change\ in\ mobility}{100}$ as the mitigation policy, we take the values of change in mobility from fig 25 in [1] for social distancing only and quantify u for Italy as:

$$u(t) = \begin{Bmatrix} 1, & prior\ to\ February\ 1 \\ 0.9, & [February\ 1, \quad February\ 27] \\ 0.85, & [February\ 27, \quad March\ 4] \\ 0.2, & [March\ 4, \quad Marcha\ 27] \\ 0.25, & [March\ 27, \quad April\ 6] \\ 0.5, & [April\ 6, \quad May\ 4] \\ 0.85, & [May\ 4, \quad June\ 18] \\ 0.9, & [After\ June\ 18] \end{Bmatrix}$$

2. The modified model with mitigation description was formulated using equation 32 and 33[1] as:

$$\frac{dx}{dt} = \frac{1}{T_{\mathrm{i}}}(u - x) \tag{14}$$

$$k_{\mathrm{i}} = k_{\mathrm{i}}^{o} x \tag{15}$$

Where, the initial value of x, $\mathrm{x}[0] = 1$, and the mitigation policy 'u' varies with time as given above in part 1,

$$T_{\mathrm{i}} = 10.5d$$
$$k_{\mathrm{i}}^{o} = 0.25168d^{-1}$$

The model was implemented in python as shown in the Appendix A:

Where the values of the constants were taken from Table 4 in [1];

$N = 60500000$, $k_{\mathrm{i}}^{o} = 0.25168$, $k_{\mathrm{e}} = 1$, $k_{\mathrm{ca}} = 0.1667$, $\eta = 0.4$, $k_{\mathrm{c}} = 0.05714$, $k_{\mathrm{a}} = 0.0857$, $k_{\mathrm{r}} = 0.1429$, $T_{\mathrm{i}} = 10.5$, $c_1 = 63.7$, $c_2 = 0.135$

and initial conditions from Table 5 in [1];

$S_0 = 60500000$, $E_0 = 0.2779$, $I_0 = 150.499$, $C_0 = 1$, $A_0 = 0.202$, $R_0 = 0$.

Finally the states are updated using the Forward Euler's Method as: $y_{n+1} = (1 + \Delta t)y_n$ for all the S,E,I,C,CC,A,R variables.

3. Cumulative number of confirmed infected in Italy was plotted in python program as:

From figure 22 in [1], it is seen that the curve somewhat flattens at 0.4 which is 24.2 million confirmed cases, and in the figure we got from the simulation, the curve flattens at 21.8 million, so we can conclude that there is a decent similarity between these two curves.

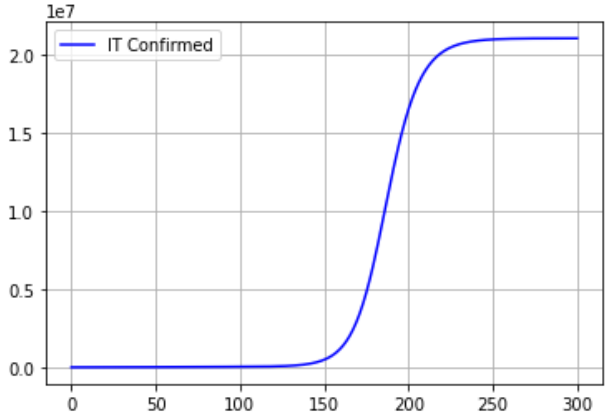


Figure 7: Cumulative number of confirmed infected in Italy

Problem 4: Combined model for Italy and Spain with population interaction

1. By assuming well mixed compartments, emigration can be written as:

$$\check{S}_{\mathrm{e}} = \check{S} \tag{16}$$

The general equation for susceptible,

$$\frac{dS}{dt} = \frac{-k_{\mathrm{i}} * \left(\check{I} + \check{A}\right) * \check{S}}{N} \tag{17}$$

If emigration and immigration apply to the equation, it can be written as:

$$\frac{dS}{dt} = \check{S}_{\mathrm{i}} - \check{S}_{\mathrm{e}} - \frac{k_{\mathrm{i}} * \left(\check{I} + \check{A}\right) * \check{S}}{N} \tag{18}$$

Because $\check{S}_e$ is equal to $\check{S}$ and total number of populations assumed constant in Italy, the equation might be rearranging as:

$$\frac{dS\_I}{dt} = \frac{N_{\mathrm{i}} * S\_I}{N\_I} - \frac{N_{\mathrm{e}} * S\_I}{N\_I} - \frac{k_{\mathrm{i}} * \left(\check{I} + \check{A}\right) * \widetilde{S\_I}}{N\_I} \tag{19}$$

where N_{i} denotes the number of immigrations, and N_{e} denotes the number of emigrations.

The following questions ask for combining Spain and Italy models together, so new letters are presented as:

N_I: Population in Italy

S_I: Number of susceptible in Italy

2. In the process of developing models for Spain, same logic is used as Italy's equations. However, operating conditions and parameters are updated as given in Table 4 and 5 in the project task[1]. Mitigation policy values are chosen the given data for Spain social distancing data. *U* value is calculated for day interval as follows, and the python code is given in Appendix A.

$$U = \begin{cases} 1, & 0 < t \leq 40 \\ 0.2, & 40 < t \leq 50 \\ 0.15, & 50 < t \leq 70 \\ 0.25, & 70 < t \leq 90 \\ 0.35, & 90 < t \leq 120 \\ 0.75, & 120 < t \leq 160 \\ 0.9, & t > 160 \end{cases}$$

3. The following python codes represent the combination of immigration and emigration models as an input for Italy and an output for Spain. The important point is that emigration out of Spain is written as immigration for Italy. The same logic is used for immigration. Immigration into Italy is assumed that equals to the number of emigrations out of Spain. Thus, equations are written as:
dS_I_dt = Ni*S_S[k]/N_S - Ne*S_I[k]/N_I - (ki_I/N_I)*(I_I[k]+A_I[k])*S_I[k]

where

Ni: The number of immigrations

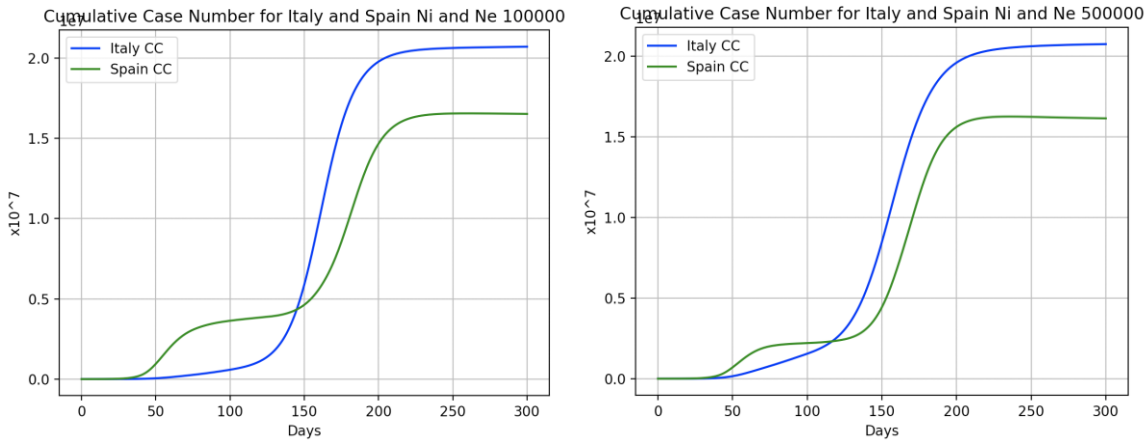
Ne: The number of emigrations

N_I: Population in Italy

N_S: Population in Spain

Significantly, this is applied to the rest of codes. Thus, codes are written as shown in Appendix A;

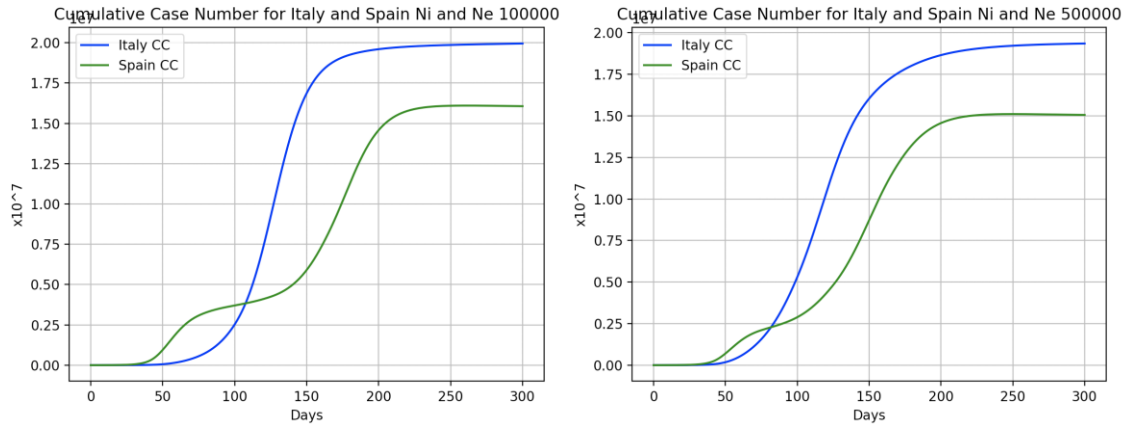
4. With combined model for Italy and Spain, some variations were made as follows:



(A) (B)

Figure 8: Variation in immigration/emigration rate

Graph A represents when immigration and emigration between two the countries are equal and the number is 100000. However, Graph B demonstrates the situation when immigration and emigration reach 5 times higher than example A. Looking into details, higher immigration and emigration values led to more case numbers in Italy before 150 days in graph B. In contrast, Spain experience having fewer case numbers before 150 days in graph B. This can be explained by either specified u values and the number of populations.



(C) (D)

Figure 9: Variation in mitigation policy

Graph C and D represent when U value in Italy is different from its initial state. Additionally, immigration and emigration numbers were kept constant to make a better comparison between graphs C, D, and A, B. It is assumed that mitigation policy was not strict compared to the initial state in Italy. Thus, high U values were chosen only for Italy. It can be clearly seen that higher immigration and emigration numbers can cause high case numbers when the mitigation policy relaxed. But more importantly, although mitigation policy was constant in Spain, there is an increase in the number of cases before 150 days due to immigration and emigration factor.

Problem 5. Quenching the covid-19 epidemics

1. Herd immunity:

First step is setting mitigation policy to u ≡ 1 and $R_0 \approx 4$. In order to simulate the system, we need to calculte k_i^0. From Table 4 in [1] we have the equation for the basic reproduction number as follows:

$$R_0 = \frac{(c_2 + k_{ca})(c_2 + k_e)(c_2 + k_r)}{k_{ca} k_e (c_2 + k_r + k_a)} \left(1 + \frac{k_a}{k_r}\right) \tag{20}$$

By assuming R_0 = 4, the value for c_2 becomes equal to 0.235. Thus, we are able to calculate τ_0 through the equation from [2] which is given below:

$$\tau_0 = \frac{(c_2 + k_{ca})(c_2 + k_e)(c_2 + k_r)}{k_e S_0 (c_2 + k_r + k_a)} \tag{21}$$

For calculating k_i^0, we have the equation:

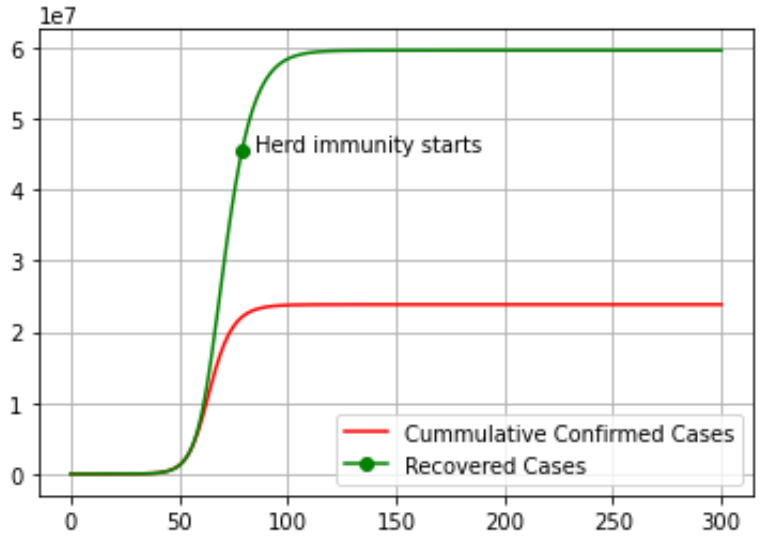
$$k_i^0 = N.\tau_0 \tag{22}$$

Where, N is the number of populations in Italy and the value for τ_0 calculated by Eq.2. The new value for k_i^0 is 0.39224 d^{-1}.

The next step is defining the herd immunity to obtain the fraction of the population has been infected. In order to achieve this goal, we are supposed to find the proportion of a population that needs to be immune before herd immunity kicks in. The formula for calculating the herd-immunity threshold is $\left(1 - \frac{1}{R_0}\right) \times 100$, meaning that the more people who become infected by each individual who has the virus, the higher the proportion of the population that needs to be immune to reach herd immunity [3].

By selecting R0 = 4, the threshold would be 75%, and by considering Italy, the population needs to be immune would be 45375000.

According to the graph for Recovered individuals is shown in figure 10 with the number calculated above, the herd immunity will start after 79 days. Therefore, the number of confirmed infected individuals would be 22061264 which is 36.46% of the population.



(Population needs to be immune before herd immunity kicks in: 45429664)

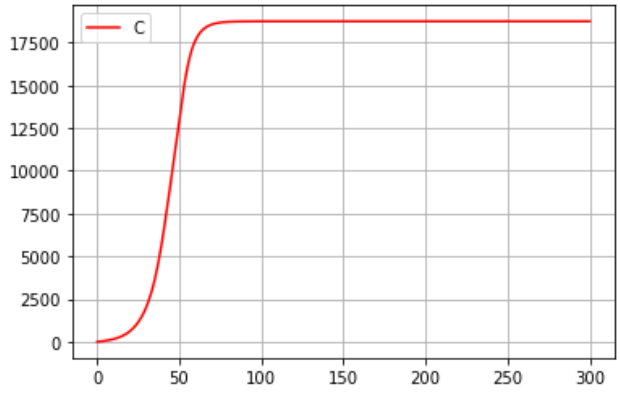
(Population has been infected: 22061264)

Figure 10: Cumulative number of confirmed infected and Recovered individuals in Italy in the case of herd immunity

Trying to reach herd immunity is not a defensible plan and it would lead to a catastrophic loss of human lives, because you cannot control infection spread to high-risk people. Deaths are only one part of the equation, individuals who become ill with the disease can experience long-term health effects like cardiovascular and respiratory problems. Besides, there are a lot of financial consequences so that relying on herd immunity alone would overwhelm hospitals.
Another point is the reinfection matter. If the people who are infected become susceptible again in a year, then basically you will never reach herd immunity.

2. Vaccination:

First, we simulate the system with mitigation policy and vaccination which is implemented on 50th day from starting point of modeling. Figure 11 illustrates the effect of both mitigation and vaccination on the number of confirmed individuals.

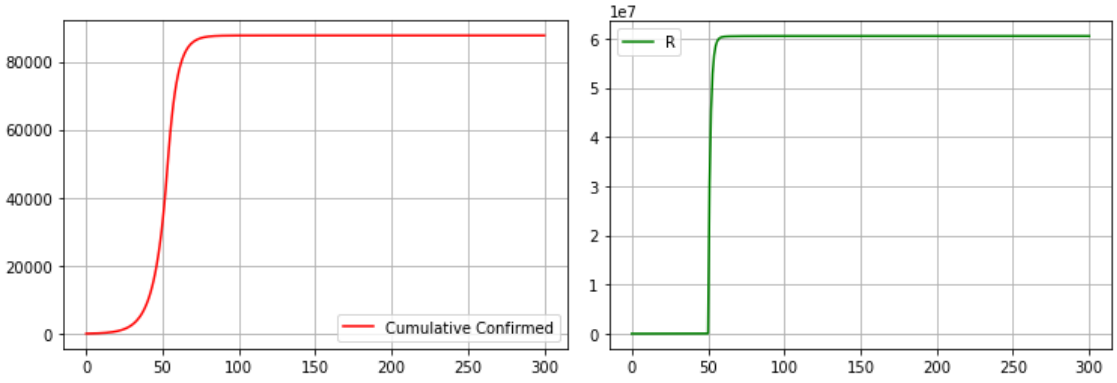


(Number of Cumulative Confirmed Individuals: 18742)

Figure 11: Cumulative confirmed individuals with mitigation policy and vaccination

As it is shown in figure 11, the number of cumulative confirmed individuals has decreased significantly due to the effective approaches. Before the vaccination, the curve is rising and by implementing it, confirmed cases start to flatten after 50 days.

Second, in the case of simulation in presence of vaccination and without mitigation policy, the cumulative confirmed and recovered individuals are shown in figure 12.



(Number of Cumulative Confirmed Individuals: 87671)

Figure 12: Cumulative confirmed individuals and Recovered number without mitigation policy

In this case, the number of cumulative confirmed cases increased. The number of recovered individuals equals the total population of Italy which means that everyone has been vaccinated.

Appendix A: Python Code for combined model of Italy and Spain considering migration and mitigation policy

```
# import packages
import matplotlib.pyplot as plt
import numpy as np
# Model parameters for Italy
N_I = 60500000
kr_I = 1/7
ki0_I = 0.25168
ke_I = 1                # α in original paper
c1_I = 63.7             # χ2 in original paper
c2_I = 0.135            # χ3 in original paper
kca_I = 1/7             # ν in original paper
n_I = 0.4               # f in original paper
kc_I = kca_I * n_I      # ν1 in original paper
ka_I = kca_I*(1-n_I)    # ν2 in original paper
Ti = 10.5

# Model parameters for Spain
N_S = 46700000
kr_S = 1/7
ki0_S = 0.33204
ke_S = 1                 # α in original paper
c1_S = 433.3             # χ2 in original paper
c2_S = 0.194             # χ3 in original paper
kca_S = 1/7              # ν in original paper
```

```
n_S = 0.4                  # f in original paper
kc_S = kca_S * n_S         # ν1 in original paper
ka_S = kca_S * (1-n_S)     # ν2 in original paper
N = N_I + N_S
Ni = 100000                # immigration rate for both Italy and Spain
Ne = 100000                # emmigration rate for both Italy and Spain

# simulation time settings
Ts = 1 # Time step [s]
t_start = 0
t_stop = 300
N_sim =int((t_stop - t_start)/Ts) + 1    # Number of Time_steps

# Preallocation of arrays for plotting :
S_I = np.zeros(N_sim)
E_I = np.zeros(N_sim)
I_I = np.zeros(N_sim)
C_I = np.zeros(N_sim)
CC_I = np.zeros(N_sim)
A_I = np.zeros(N_sim)
R_I = np.zeros(N_sim)
X_I = np.zeros(N_sim)
t = np.linspace(t_start, t_stop, N_sim)
S_S = np.zeros(N_sim)
E_S = np.zeros(N_sim)
I_S = np.zeros(N_sim)
C_S = np.zeros(N_sim)
CC_S = np.zeros(N_sim)
A_S = np.zeros(N_sim)
R_S = np.zeros(N_sim)
X_S = np.zeros(N_sim)

# Initialization :
S_I[0] = N_I
E_I[0] = (c1_I + kca_I) / ke_I
I_I[0]= (c1_I * c2_I) / kc_I
C_I[0] = 1
CC_I[0] = 1
A_I[0] = (ka_I*I_I[0])/(kr_I+c1_I)
R_I[0] = 0
X_I[0] = 1

S_S[0] = N_S
E_S[0] = (c1_S + kca_S) / ke_S
I_S[0]= (c1_S * c2_S) / kc_S
C_S[0] = 1
CC_S[0] = 1
A_S[0] = (ka_S*I_S[0])/(kr_S+c1_S)
R_S[0] = 0
X_S[0] = 1

# Simulation loop :
for k in range(0, N_sim-1):

    # Values of U
    if 0<= k <=27:
        U = 0.9
```

```
    elif 27<= k <=33:
        U = 0.85
    elif 33<= k <=57:
        U = 0.2
    elif 57<= k <=67:
        U = 0.25
    elif 67<= k <=95:
        U = 0.5
    elif 95<= k <=140:
        U = 0.85
    else:
        U = 0.9
    # Mitigation Model
    dX_I_dt = (1/Ti)*(U-X_I[k])
    ki_I = ki0_I*X_I[k]

    dS_I_dt = Ni*S_S[k]/N_S - Ne*S_I[k]/N_I-(ki_I/N_I)*(I_I[k]+A_I[k])*S_I[k]
    dE_I_dt = Ni*E_S[k]/N_S - Ne*E_I[k]/N_I +
((ki_I/N_I)*(I_I[k]+A_I[k])*S_I[k]) - ke_I*E_I[k]
    dI_I_dt = Ni*I_S[k]/N_S - Ne*I_I[k]/N_I + ke_I*E_I[k]-kc_I*I_I[k]-
ka_I*I_I[k]
    dC_I_dt = Ni*C_S[k]/N_S - Ne*C_I[k]/N_I + kc_I*I_I[k]-kr_I*C_I[k]
    dCC_I_dt = Ni*CC_S[k]/N_S - Ne*CC_I[k]/N_I + kc_I*I_I[k]
    dA_I_dt = Ni*A_S[k]/N_S - Ne*A_I[k]/N_I + ka_I*I_I[k]-kr_I*A_I[k]
    dR_I_dt = Ni*R_S[k]/N_S - Ne*R_I[k]/N_I + kr_I*C_I[k]+kr_I*A_I[k]

    # State updates using the Euler method :
    S_I[k+1] = S_I[k] + dS_I_dt *Ts
    E_I[k+1] = E_I[k] + dE_I_dt *Ts
    I_I[k+1] = I_I[k] + dI_I_dt *Ts
    C_I[k+1] = C_I[k] + dC_I_dt *Ts
    A_I[k+1] = A_I[k] + dA_I_dt *Ts
    R_I[k+1] = R_I[k] + dR_I_dt *Ts
    CC_I[k+1] = CC_I[k] + dCC_I_dt *Ts
    X_I[k+1] = X_I[k] + dX_I_dt *Ts

    # Values of U
    if 0<= k <=40:
        U = 1
    if 40<= k <=50:
        U = 0.2
    elif 40<= k <=70:
        U = 0.15
    elif 70<= k <=90:
        U = 0.25
    elif 90<= k <=120:
        U = 0.35
    elif 120<= k <=160:
        U = 0.7
    else:
        U = 0.9
    # Mitigation Model
    dX_S_dt = (1/Ti)*(U-X_S[k])
    ki_S = ki0_S*X_S[k]

    dS_S_dt = Ni*S_I[k]/N_I - Ne*S_S[k]/N_S -
(ki_S/N_S)*(I_S[k]+A_S[k])*S_S[k]
```

```
    dE_S_dt = Ni*E_I[k]/N_I - Ne*E_S[k]/N_S +
((ki_S/N_S)*(I_S[k]+A_S[k])*S_S[k]) - ke_S*E_S[k]
    dI_S_dt = Ni*I_I[k]/N_I - Ne*I_S[k]/N_S + ke_S*E_S[k]-kc_S*I_S[k]-
ka_S*I_S[k]
    dC_S_dt = Ni*C_I[k]/N_I - Ne*C_S[k]/N_S + kc_S*I_S[k]-kr_S*C_S[k]
    dCC_S_dt = Ni*CC_I[k]/N_I - Ne*CC_S[k]/N_S + kc_S*I_S[k]
    dA_S_dt = Ni*A_I[k]/N_I - Ne*A_S[k]/N_S + ka_S*I_S[k]-kr_S*A_S[k]
    dR_S_dt = Ni*R_I[k]/N_I - Ne*R_S[k]/N_S + kr_S*C_S[k]+kr_S*A_S[k]

    # State updates using the Euler method :
    S_S[k+1] = S_S[k] + dS_S_dt *Ts
    E_S[k+1] = E_S[k] + dE_S_dt *Ts
    I_S[k+1] = I_S[k] + dI_S_dt *Ts
    C_S[k+1] = C_S[k] + dC_S_dt *Ts
    A_S[k+1] = A_S[k] + dA_S_dt *Ts
    R_S[k+1] = R_S[k] + dR_S_dt *Ts
    CC_S[k+1] = CC_S[k] + dCC_S_dt *Ts
    X_S[k+1] = X_S[k] + dX_S_dt *Ts

# Plotting :
plt.close('all')
plt.figure(1)
plt.plot(t, CC_I, 'b-')
plt.plot(t, CC_S, 'g-')
plt.legend(labels=('Italy CC', 'Spain CC'))
plt.grid()
plt.show()
```

References

[1] 'Project', FM1015 Modelling of Dynamic Systems, Bernt Lie, September 18, 2020

[2] Liu, Z., Magal, P., Seydi, O., & Webb, G. (2020). A Model to Predict COVID-19 Epidemics with Applications to South Korea, Italy, and Spain

[3] Kin On Kwok et al (2020). Herd Immunity – estimating the level required to halt the COVID-19 epidemics in affected countries. Journal of Infection 80 (2020) e32-e33