Incentives and Optimal Vaccination in an SIR Model

Giuseppe Cognata, Amine Lamouchi, Thomas Noel

l'X
ÉCOLE
POLYTECHNIQUE
UNIVERSITÉ PARIS-SACLAY

Mathematical Modelling Project

April-June 2021

Acknowledgements

We sincerely thank Professor Matrolia for his insightful guidance throughout this project.

1 Introduction

The spread of Covid-19 over the last year has shown the importance of effective sanitary regulation when dealing with very contagious diseases. The most common and effective way of limiting the infection is vaccination. Effective vaccination policies have lead of the near eradication of dangerous diseases like Poliomyelitis and Smallpox[1]. Currently, countries all around the world are racing against the clock to vaccinate their population from Covid-19. However, the rise of anti-vax sentiments could hinder vaccination efforts in many countries, potentially leaving many vulnerable to the dangers of the virus. As such, it is vital that governments implement an efficient vaccination policy that deals with these risks.

To this aim, we propose to model people's choice to get vaccinated or not with an optimal control problem. We will not however model individual decisions, opting instead to view this problem from the eyes of the entire population which has control over the vaccination rate. In order to do this, we will need to model the spread of a disease, and the impact that vaccination has on it. The SIR model, a well known compartmental model in epidemiology, offers an effective yet simple way to model the spread of infectious diseases.
This will allow us to numerically solve the optimal control problem, and derive the corresponding vaccination rate, which we then use to study the evolution of the disease. To further extend our study, we will implement an incentive scheme that deals with high vaccination aversion.

2 The SIR model

We begin by presenting the basic SIR model, and then extend it to better suit our topic of investigation. We will also conduct numerical simulations of the model with different parameters to showcase possible evolutions of the system.

2.1 Building the model

The fundamental model is known as the SIR model without vital dynamics[2]. We assume that the total population is a constant $N > 0$. This model divides the population in three distinct compartments:

- Compartment S: represents the number of healthy people susceptible to the disease. When they contract it, they go to the compartment I.
- Compartment I: represents the number of infected individuals. In time, they recover from the disease and move to the compartment R.
- Compartment R: represents the number of recovered people. When you arrive in this compartment, you can't go back to neither the susceptible nor the infected compartments (recovery is final).

Then at time $t \geq 0$, the number of susceptible individuals is $S(t)$, the number of infected is $I(t)$ and the number of recovered is $R(t)$. We also have that, $\forall\, t \geq 0,\ N = S(t) + I(t) + R(t)$.

To study the evolution of the disease, the SIR model consists of a three-dimensional system of differential equations, each describing the dynamics of one of the compartments.
Let $\beta \in [0, 1]$ be the effective contact rate in proportion to the total population. Therefore, as the number possible interaction between susceptible people and infected is SI, then the number of

those which actually result in infection is βSI. We also let $\gamma \in [0,1]$ be the rate at which infected individuals recover from the disease.
From this, the basic SIR model is expressed as:

$$\begin{cases} \frac{dS}{dt} = -\beta S(t)I(t) \\ \frac{dI}{dt} = \beta S(t)I(t) - \gamma I(t) \\ \frac{dR}{dt} = \gamma I(t) \end{cases} \tag{1}$$

We can observe by adding up the equations in (1) that $\frac{dS}{dt} + \frac{dI}{dt} + \frac{dR}{dt} = 0$, confirming that our model maintains a constant population.

This model, however, is not very accurate for representing a disease which stays in a population for long periods of time. To be able to use this model on a long-term disease, we need to include vital dynamics, which are illustrated by a new parameter $\alpha \in [0,1]$. This constant expresses the death and birth rates of the population. We assume them to be equal per our assumption that the population remains constant. Then, the dynamics of the systems are now expressed by:

$$\begin{cases} \frac{dS}{dt} = \alpha N - \alpha S(t) - \beta S(t)I(t) \\ \frac{dI}{dt} = \beta S(t)I(t) - \alpha I(t) - \gamma I(t) \\ \frac{dR}{dt} = \gamma I(t) - \alpha R(t) \end{cases} \tag{2}$$

It is again easy to check that the sum the derivatives in (2) is zero.

Finally, we extend the model one step further by including a vaccination rate $\mu \in [0,1]$. This addition then changes the description of the three compartments:

- Compartment S: represents the number of healthy people susceptible to the disease. When they contract it, they go to the compartment I. Although, if they get vaccinated before contracting it, they can move directly to the compartment R.
- Compartment I: represents the number of infected individuals. In time, they recover from the disease and move to the compartment R. They can't be vaccinated.
- Compartment R: represents both the number of vaccinated individuals and recovered people. When you arrive in this compartment, you can't go back to neither the susceptible nor the infected compartments (recovery is final).

Hence, our system of equations becomes:

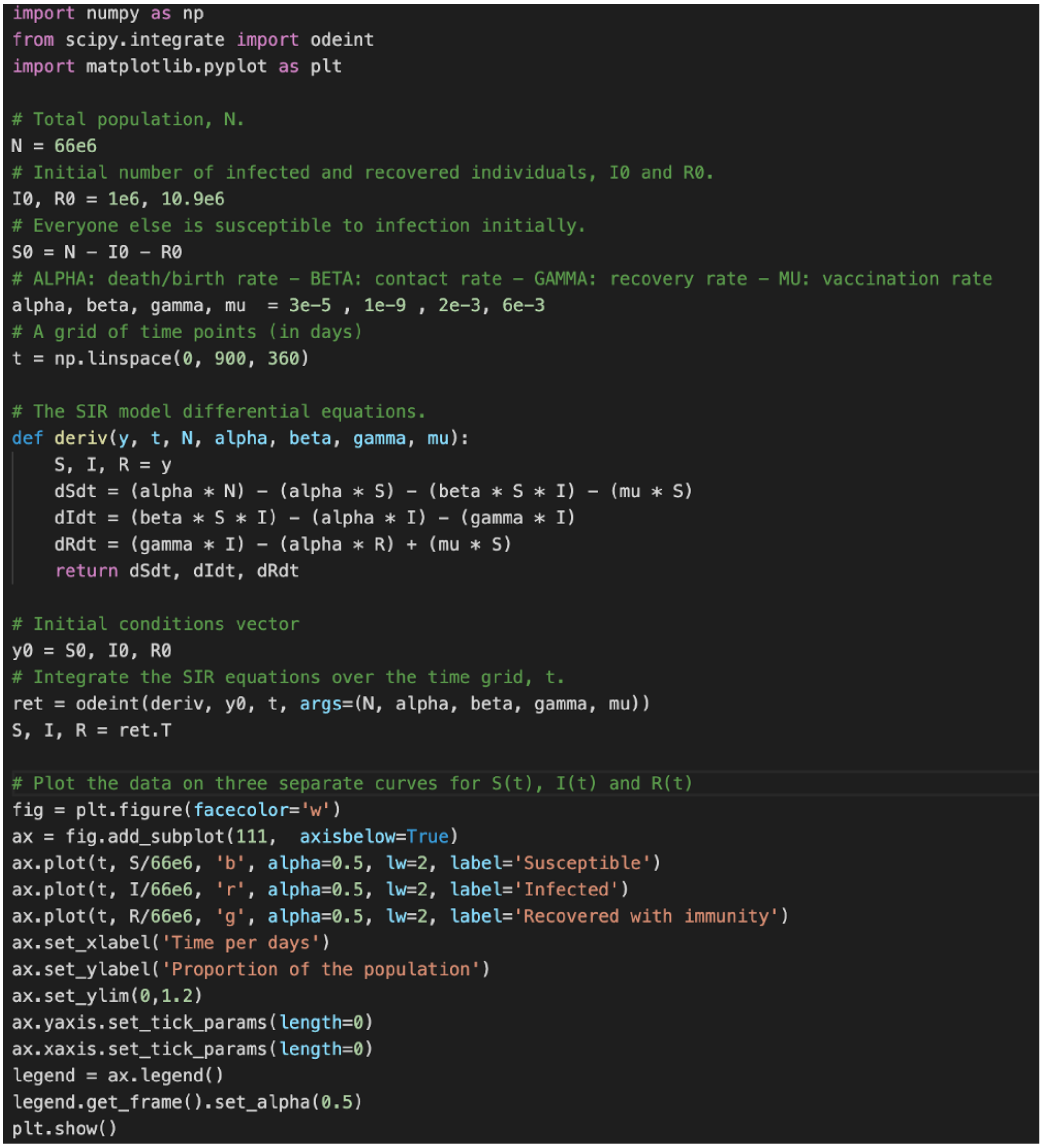
$$\begin{cases} \frac{dS}{dt} = \alpha N - \alpha S(t) - \beta S(t)I(t) - \mu S(t) \\ \frac{dI}{dt} = \beta S(t)I(t) - \alpha I(t) - \gamma I(t) \\ \frac{dR}{dt} = \gamma I(t) + \mu S(t) - \alpha R(t) \end{cases} \tag{3}$$

It's readily checked that the sum of the terms in the equations in (3) is zero.

2.2 Numerical simulations

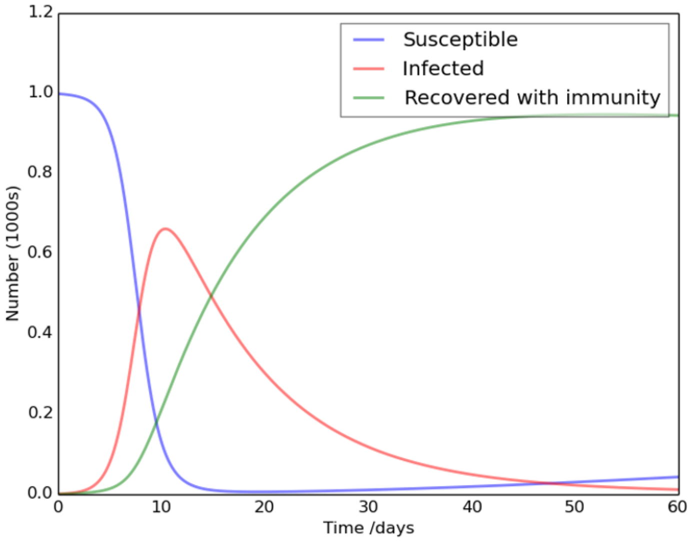
In the following part, we present our numerical simulation for the SIR model with vital dynamics. The code[3] we implemented begins by specifying the initial conditions of the model as well as its four parameters. It then builds the system of differential equations in the function *deriv* and integrates it over a fixed period of time. For this latter step, we used the function *odeint* from python's

mathematical library *scipy*. The last part of our code is for plotting the numerical results for each compartment of the model.
The figure below shows the code used for conducting the simulations.



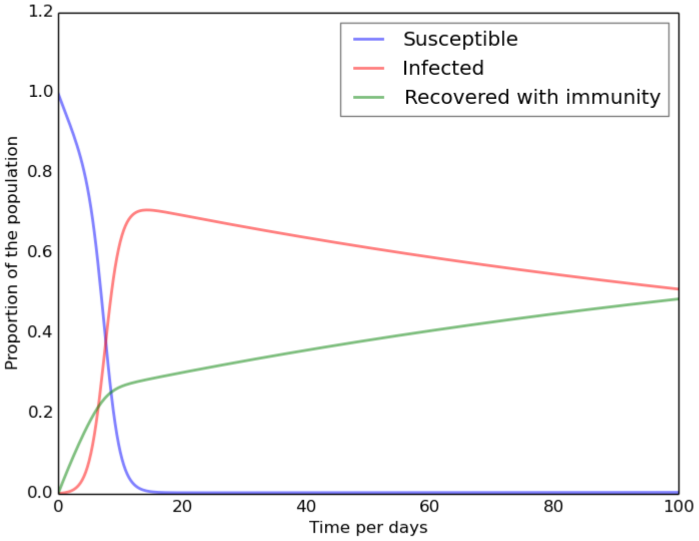
```
import numpy as np
from scipy.integrate import odeint
import matplotlib.pyplot as plt

# Total population, N.
N = 66e6
# Initial number of infected and recovered individuals, I0 and R0.
I0, R0 = 1e6, 10.9e6
# Everyone else is susceptible to infection initially.
S0 = N - I0 - R0
# ALPHA: death/birth rate - BETA: contact rate - GAMMA: recovery rate - MU: vaccination rate
alpha, beta, gamma, mu  = 3e-5 , 1e-9 , 2e-3, 6e-3
# A grid of time points (in days)
t = np.linspace(0, 900, 360)

# The SIR model differential equations.
def deriv(y, t, N, alpha, beta, gamma, mu):
    S, I, R = y
    dSdt = (alpha * N) - (alpha * S) - (beta * S * I) - (mu * S)
    dIdt = (beta * S * I) - (alpha * I) - (gamma * I)
    dRdt = (gamma * I) - (alpha * R) + (mu * S)
    return dSdt, dIdt, dRdt

# Initial conditions vector
y0 = S0, I0, R0
# Integrate the SIR equations over the time grid, t.
ret = odeint(deriv, y0, t, args=(N, alpha, beta, gamma, mu))
S, I, R = ret.T

# Plot the data on three separate curves for S(t), I(t) and R(t)
fig = plt.figure(facecolor='w')
ax = fig.add_subplot(111,  axisbelow=True)
ax.plot(t, S/66e6, 'b', alpha=0.5, lw=2, label='Susceptible')
ax.plot(t, I/66e6, 'r', alpha=0.5, lw=2, label='Infected')
ax.plot(t, R/66e6, 'g', alpha=0.5, lw=2, label='Recovered with immunity')
ax.set_xlabel('Time per days')
ax.set_ylabel('Proportion of the population')
ax.set_ylim(0,1.2)
ax.yaxis.set_tick_params(length=0)
ax.xaxis.set_tick_params(length=0)
legend = ax.legend()
legend.get_frame().set_alpha(0.5)
plt.show()
```

By repeatedly running the code while changing its parameters, we observed different behaviours and equilibria. The figures below show two possible states for the system depending on the chosen parameters. For these two plots, we take a total population of $N = 1000$ and initial conditions $S(0) = 999$, $I(0) = 1$ and $R(0) = 0$



Code run with $\alpha = 0.002$, $\beta = 0.001$, $\gamma = 0.1$ and $\mu = 0.001$

The above graph, shows that the disease is eradicated after 2 months from the outbreak while in the following one, we can observe that the infection remains present at equilibrium.



Code run with $\alpha = 0.002$, $\beta = 0.001$, $\gamma = 0.005$ and $\mu = 0.04$

3 Finding the optimal vaccination rate

3.1 The population's problem

A population presented with the option to get vaccinated or not will control the vaccination rate directly. Hence, we can replace the number μ in (3) with the function $\mu : \mathbb{R}^+ \to [0, 0.9]$. We take the maximum value of μ to be smaller than one as we assume that it's impossible to vaccinate an entire population at once. The system now becomes:

$$\begin{cases} \frac{dS}{dt} = \alpha N - \alpha S(t) - \beta S(t)I(t) - \mu(t)S(t) \\ \frac{dI}{dt} = \beta S(t)I(t) - \alpha I(t) - \gamma I(t) \\ \frac{dR}{dt} = \gamma I(t) + \mu(t)S(t) - \alpha R(t) \end{cases} \tag{4}$$

We assume that getting infected causes an individual to incur a fixed cost $c > 0$, perhaps due to treatment costs, or simply as a loss in utility. Furthermore, we assume that the population incurs a loss $L : [0, 0.9] \to \mathbb{R}$ to getting vaccinated that is increasing and convex in μ. For the sake of simplicity, we choose:

$$L(\mu(t)) = k\frac{\mu^2(t)}{2}$$

where $k > 0$ represents a natural adversity of the population to vaccination, perhaps due to distrust of the companies providing it. We assume that L is increasing because, as more people get vaccinated, the number of those experiencing side effects is likely to increase. Moreover, it's reasonable to assume that the marginal rate of distrust is increasing in μ, hence the convexity assumption.

Then, the population's problem is to find, over a fixed period of time $T > 0$, a vaccination rate that balances the cost incurred by getting infected and the loss from increased vaccinations. We therefore define the following objective function for the population:

$$J(\mu) := \int_0^T cI(t) + L(\mu(t))\, dt$$

then the population's problem becomes:

$$\min_{\mu\in[0,0.9]} J(\mu) = \min_{\mu\in[0,0.9]} \left(\int_0^T cI(t) + k\frac{\mu^2(t)}{2}\, dt \right) \tag{5}$$

subject to:

$$\begin{cases} \frac{dS}{dt} = \alpha N - \alpha S(t) - \beta S(t)I(t) - \mu(t)S(t) \\ \frac{dI}{dt} = \beta S(t)I(t) - \alpha I(t) - \gamma I(t) \\ \frac{dR}{dt} = \gamma I(t) + \mu(t)S(t) - \alpha R(t) \end{cases}$$

We will assume that this system admits an optimum μ^* such that $J(\mu^*) = \min_{\mu\in[0,0.9]} J(\mu)$. Hence, we may apply an important result in optimal control theory: Pontryagin's maximum/minimum principle, whose statement is in the appendix. Applying this principle, the Hamiltonian of our problem is given by:

$$H = cI(t) + k\frac{\mu^2(t)}{2} + \lambda_1(t)\frac{dS}{dt} + \lambda_2(t)\frac{dI}{dt} + \lambda_3(t)\frac{dR}{dt}$$

We have omitted the arguments of the Hamiltonian for presentation purposes. Here, λ_1, λ_2 and λ_3 are the components fo the vector λ as defined in the statement of Pontryagin's maximum principle.

Let $(S^*(t), I^*(t), R^*(t))$ be the solution to the system in (5) with a vaccination rate μ^*. We now compute:

$$\frac{\partial H}{\partial \mu} = k\mu(t) - \lambda_1(t)S(t) + \lambda_3(t)S(t)$$

So the optimality condition reads:

$$\frac{\partial H}{\partial \mu}(t, \mu^*(t), S^*(t)) = k\mu^*(t) - \lambda_1(t)S^*(t) + \lambda_3(t)S^*(t) = 0$$

and hence:

$$\mu^*(t) = \frac{S^*(t)}{k}(\lambda_1(t) - \lambda_3(t)) \tag{6}$$

We also have that:

$$\begin{aligned}
\frac{\partial H}{\partial S} &= (-\alpha - \beta I(t) - \mu(t))\lambda_1(t) + \beta I(t)\lambda_2(t) + \mu(t)\lambda_3(t) \\
\frac{\partial H}{\partial I} &= c - \beta S(t)\lambda_1(t) + (\beta S(t) - \alpha - \gamma)\lambda_2(t) + \gamma\lambda_3(t) \\
\frac{\partial H}{\partial R} &= -\alpha\lambda_3(t)
\end{aligned}$$

The adjoint equations then are:

$$\begin{cases}
\lambda_1'(t) = (\alpha + \beta I^*(t) + \mu^*(t))\lambda_1(t) - \beta I^*(t)\lambda_2(t) - \mu^*(t)\lambda_3(t) \\
\lambda_2'(t) = \beta S^*(t)\lambda_1(t) - (\beta S^*(t) - \alpha - \gamma)\lambda_2(t) - \gamma\lambda_3(t) - c \\
\lambda_3'(t) = \alpha\lambda_3(t)
\end{cases}$$

We can then substitute the value for μ^* form (6) into the system and re-arrange it to obtain:

$$\begin{cases}
\lambda_1'(t) = \beta I^*(t)(\lambda_1(t) - \lambda_2(t)) + \alpha\lambda_1(t) + \frac{S^*(t)}{k}(\lambda_1(t) - \lambda_3(t))(\lambda_1(t) - \lambda_3(t)) \\
\lambda_2'(t) = \beta S^*(t)(\lambda_1(t) - \lambda_2(t)) + (\alpha + \gamma)\lambda_2(t) - \gamma\lambda_3(t) - c \\
\lambda_3'(t) = \alpha\lambda_3(t)
\end{cases}$$

Finally, we recall the transversality conditions:

$$\lambda_1(T) = \lambda_2(T) = \lambda_3(T) = 0$$

We can explicitly obtain that $\lambda_3(t) = \lambda_3(0)e^{\alpha t}$, and, by the transversality conditions above, $\lambda_3(T) = \lambda_3(0)e^{\alpha T} = 0$. Hence, we deduce that $\lambda_3(0) = 0$, so that $\lambda_3(t) = 0$ for all $t \in [0, T]$. Therefore we obtain that:

$$\mu^*(t) = \frac{S^*(t)}{k}\lambda_1(t) \tag{7}$$

Furthermore, the Hamiltonian is reduced to:

$$H = cI(t) + k\frac{\mu^2(t)}{2} + \lambda_1(t)\frac{dS}{dt} + \lambda_2(t)\frac{dI}{dt}$$

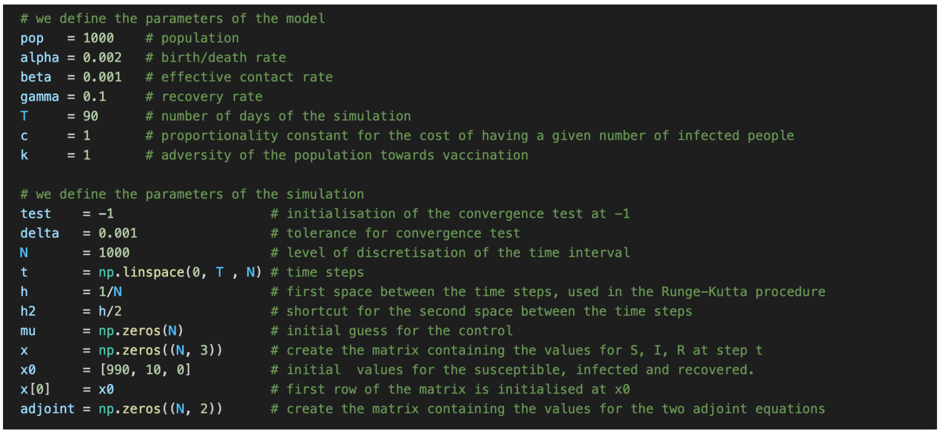
and the adjoint equations become:

$$\begin{cases}
\lambda_1'(t) = \beta I^*(t)(\lambda_1(t) - \lambda_2(t)) + \alpha\lambda_1(t) + \frac{S^*(t)}{k}\lambda_1^2(t) \\
\lambda_2'(t) = \beta S^*(t)(\lambda_1(t) - \lambda_2(t)) + (\alpha + \gamma)\lambda_2(t) - c
\end{cases} \tag{8}$$

Notice that, even after this simplification, the system is not only non-linear, but contains the state variables, whose explicit forms are unknown. Hence, we cannot compute an explicit solution for this system, and thus neither for μ^*.

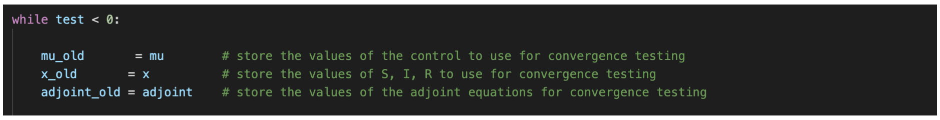
3.2 Numerical solution

We now numerically solve the optimal control problem in (5) using the Forward-Backward Sweep method[4]. The first step is to make an initial guess of the control, which we take as a null function. We note that this guess is permitted since we don't divide by mu anywhere in the model.
We begin the simulation by stating the parameters of the model and defining the variables needed for the underlying Runge-Kutta fourth order procedure.



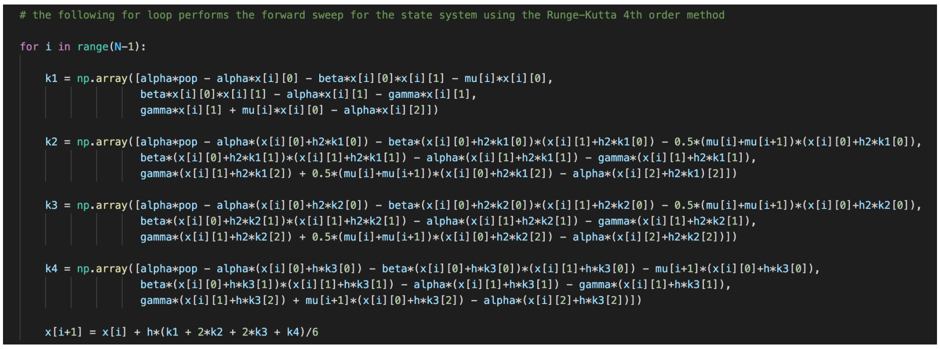
```
# we define the parameters of the model
pop   = 1000    # population
alpha = 0.002   # birth/death rate
beta  = 0.001   # effective contact rate
gamma = 0.1     # recovery rate
T     = 90      # number of days of the simulation
c     = 1       # proportionality constant for the cost of having a given number of infected people
k     = 1       # adversity of the population towards vaccination

# we define the parameters of the simulation
test    = -1                    # initialisation of the convergence test at -1
delta   = 0.001                 # tolerance for convergence test
N       = 1000                  # level of discretisation of the time interval
t       = np.linspace(0, T , N) # time steps
h       = 1/N                   # first space between the time steps, used in the Runge-Kutta procedure
h2      = h/2                   # shortcut for the second space between the time steps
mu      = np.zeros(N)           # initial guess for the control
x       = np.zeros((N, 3))      # create the matrix containing the values for S, I, R at step t
x0      = [990, 10, 0]          # initial  values for the susceptible, infected and recovered.
x[0]    = x0                    # first row of the matrix is initialised at x0
adjoint = np.zeros((N, 2))      # create the matrix containing the values for the two adjoint equations
```

We then define a while loop that will perform a convergence test at each of its iterations to ensure we have found a solution.



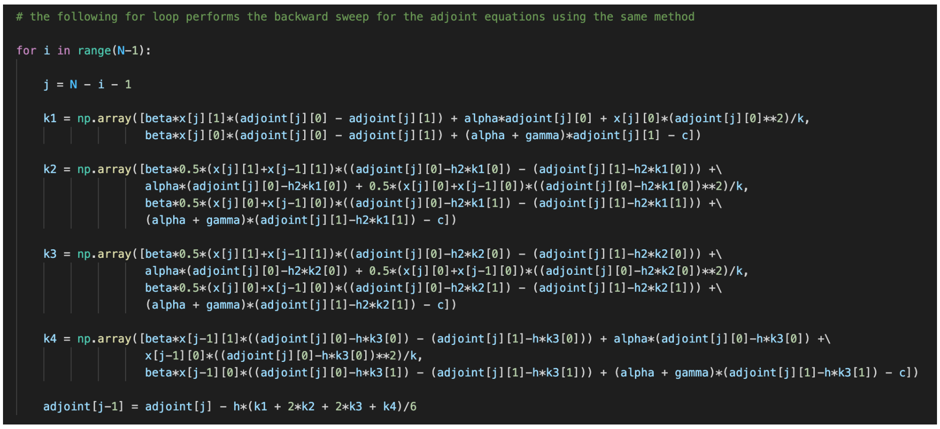
```
while test < 0:

    mu_old      = mu        # store the values of the control to use for convergence testing
    x_old      = x          # store the values of S, I, R to use for convergence testing
    adjoint_old = adjoint   # store the values of the adjoint equations for convergence testing
```

Then, we compute the state in (4) forward in time using the RK4 procedure and storing the values at each time t in the x matrix.



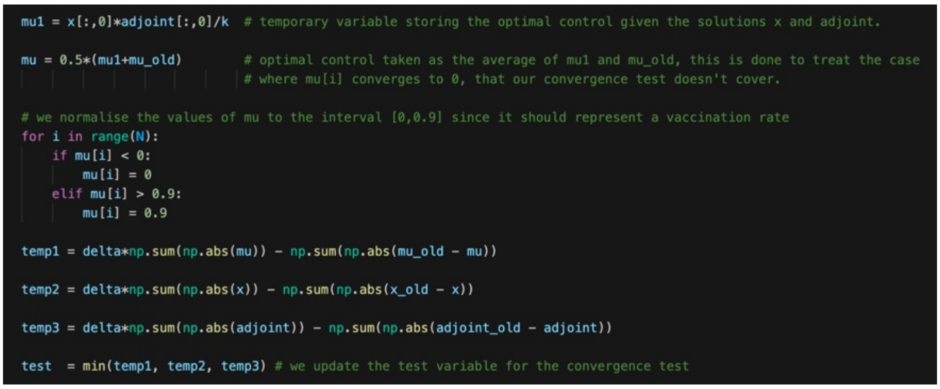
```
# the following for loop performs the forward sweep for the state system using the Runge-Kutta 4th order method

for i in range(N-1):

    k1 = np.array([alpha*pop - alpha*x[i][0] - beta*x[i][0]*x[i][1] - mu[i]*x[i][0],
                   beta*x[i][0]*x[i][1] - alpha*x[i][1] - gamma*x[i][1],
                   gamma*x[i][1] + mu[i]*x[i][0] - alpha*x[i][2]])

    k2 = np.array([alpha*pop - alpha*(x[i][0]+h2*k1[0]) - beta*(x[i][0]+h2*k1[0])*(x[i][1]+h2*k1[0]) - 0.5*(mu[i]+mu[i+1])*(x[i][0]+h2*k1[0]),
                   beta*(x[i][0]+h2*k1[1])*(x[i][1]+h2*k1[1]) - alpha*(x[i][1]+h2*k1[1]) - gamma*(x[i][1]+h2*k1[1]),
                   gamma*(x[i][1]+h2*k1[2]) + 0.5*(mu[i]+mu[i+1])*(x[i][0]+h2*k1[2]) - alpha*(x[i][2]+h2*k1)[2]])

    k3 = np.array([alpha*pop - alpha*(x[i][0]+h2*k2[0]) - beta*(x[i][0]+h2*k2[0])*(x[i][1]+h2*k2[0]) - 0.5*(mu[i]+mu[i+1])*(x[i][0]+h2*k2[0]),
                   beta*(x[i][0]+h2*k2[1])*(x[i][1]+h2*k2[1]) - alpha*(x[i][1]+h2*k2[1]) - gamma*(x[i][1]+h2*k2[1]),
                   gamma*(x[i][1]+h2*k2[2]) + 0.5*(mu[i]+mu[i+1])*(x[i][0]+h2*k2[2]) - alpha*(x[i][2]+h2*k2[2])])

    k4 = np.array([alpha*pop - alpha*(x[i][0]+h*k3[0]) - beta*(x[i][0]+h*k3[0])*(x[i][1]+h*k3[0]) - mu[i+1]*(x[i][0]+h*k3[0]),
                   beta*(x[i][0]+h*k3[1])*(x[i][1]+h*k3[1]) - alpha*(x[i][1]+h*k3[1]) - gamma*(x[i][1]+h*k3[1]),
                   gamma*(x[i][1]+h*k3[2]) + mu[i+1]*(x[i][0]+h*k3[2]) - alpha*(x[i][2]+h*k3[2])])

    x[i+1] = x[i] + h*(k1 + 2*k2 + 2*k3 + k4)/6
```

Now, we solve the adjoint system in (8) backwards in time again using the RK4 procedure.



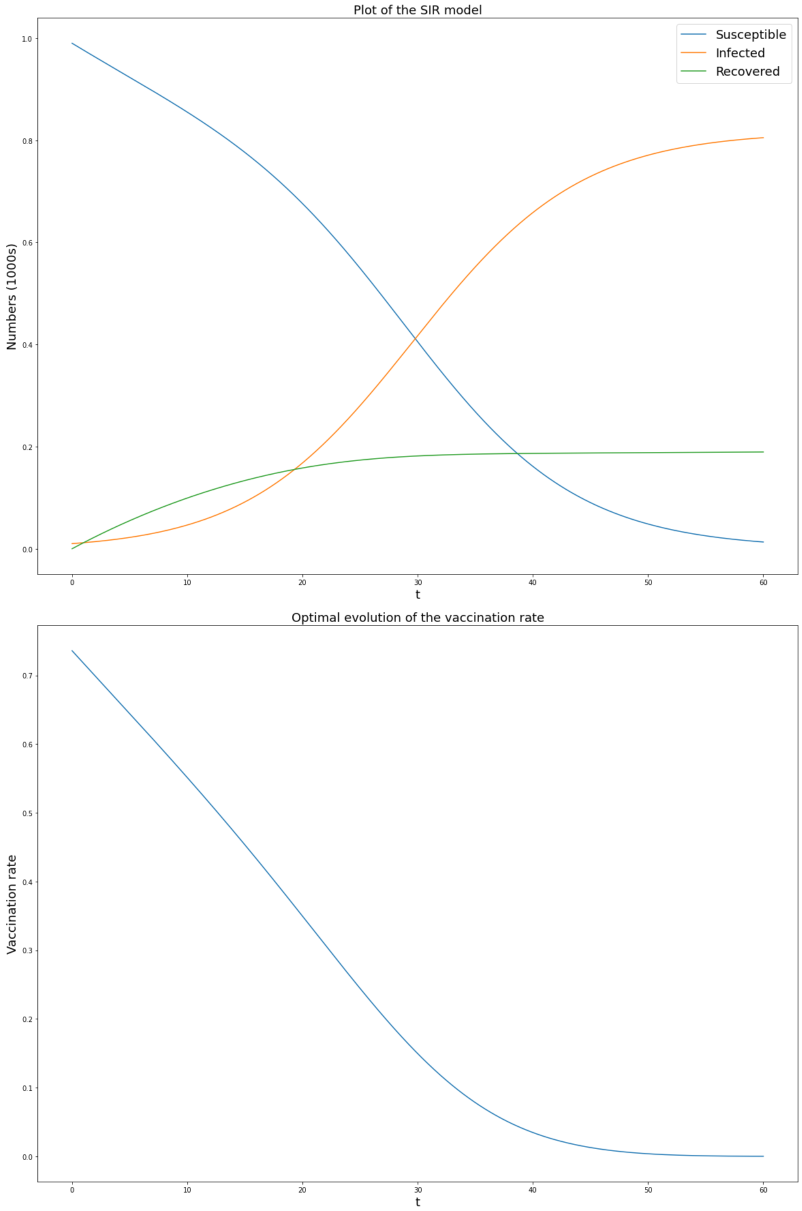
```
# the following for loop performs the backward sweep for the adjoint equations using the same method

for i in range(N-1):

    j = N - i - 1

    k1 = np.array([beta*x[j][1]*(adjoint[j][0] - adjoint[j][1]) + alpha*adjoint[j][0] + x[j][0]*(adjoint[j][0]**2)/k,
                   beta*x[j][0]*(adjoint[j][0] - adjoint[j][1]) + (alpha + gamma)*adjoint[j][1] - c])

    k2 = np.array([beta*0.5*(x[j][1]+x[j-1][1])*((adjoint[j][0]-h2*k1[0]) - (adjoint[j][1]-h2*k1[0])) +\
                   alpha*(adjoint[j][0]-h2*k1[0]) + 0.5*(x[j][0]+x[j-1][0])*((adjoint[j][0]-h2*k1[0])**2)/k,
                   beta*0.5*(x[j][0]+x[j-1][0])*((adjoint[j][0]-h2*k1[1]) - (adjoint[j][1]-h2*k1[1])) +\
                   (alpha + gamma)*(adjoint[j][1]-h2*k1[1]) - c])

    k3 = np.array([beta*0.5*(x[j][1]+x[j-1][1])*((adjoint[j][0]-h2*k2[0]) - (adjoint[j][1]-h2*k2[0])) +\
                   alpha*(adjoint[j][0]-h2*k2[0]) + 0.5*(x[j][0]+x[j-1][0])*((adjoint[j][0]-h2*k2[0])**2)/k,
                   beta*0.5*(x[j][0]+x[j-1][0])*((adjoint[j][0]-h2*k2[1]) - (adjoint[j][1]-h2*k2[1])) +\
                   (alpha + gamma)*(adjoint[j][1]-h2*k2[1]) - c])

    k4 = np.array([beta*x[j-1][1]*((adjoint[j][0]-h*k3[0]) - (adjoint[j][1]-h*k3[0])) + alpha*(adjoint[j][0]-h*k3[0]) +\
                   x[j-1][0]*((adjoint[j][0]-h*k3[0])**2)/k,
                   beta*x[j-1][0]*((adjoint[j][0]-h*k3[1]) - (adjoint[j][1]-h*k3[1])) + (alpha + gamma)*(adjoint[j][1]-h*k3[1]) - c])

    adjoint[j-1] = adjoint[j] - h*(k1 + 2*k2 + 2*k3 + k4)/6
```

Finally, we compute μ as the average of the value in the previous loop and the formula for μ^*. We also normalize its values to ensure they remain in $[0, 0.9]$. With the values of the x, adjoint and mu matricies, we perform the convergence test to ensure that the solution is close to the one in the previous iteration.



```
mu1 = x[:,0]*adjoint[:,0]/k  # temporary variable storing the optimal control given the solutions x and adjoint.

mu = 0.5*(mu1+mu_old)        # optimal control taken as the average of mu1 and mu_old, this is done to treat the case
                             # where mu[i] converges to 0, that our convergence test doesn't cover.

# we normalise the values of mu to the interval [0,0.9] since it should represent a vaccination rate
for i in range(N):
    if mu[i] < 0:
        mu[i] = 0
    elif mu[i] > 0.9:
        mu[i] = 0.9

temp1 = delta*np.sum(np.abs(mu)) - np.sum(np.abs(mu_old - mu))

temp2 = delta*np.sum(np.abs(x)) - np.sum(np.abs(x_old - x))

temp3 = delta*np.sum(np.abs(adjoint)) - np.sum(np.abs(adjoint_old - adjoint))

test  = min(temp1, temp2, temp3) # we update the test variable for the convergence test
```

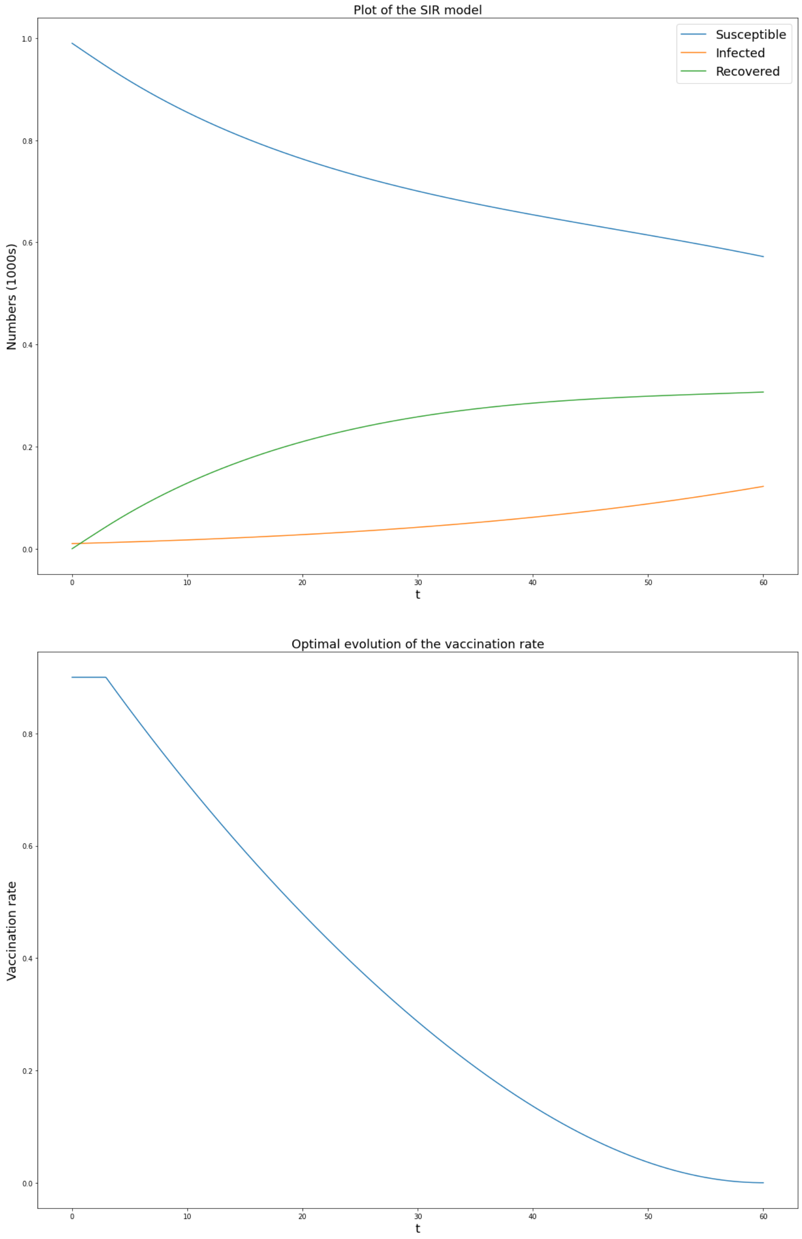
Using the code described above, we simulated two types of viral diseases: an aggressive virus then a weaker one. For each case we chose a certain level of aversion towards vaccination while keeping a constant cost of infection ($c = 1$). To control for the strength of the disease, we used its β and γ parameters (respectively the effective contact rate and the recovery rate). While for the aversion towards vaccination we chose different values for the k parameter. For both plots, we took $N = 1000$, $S(0) = 990$, $I(0) = 10$ and $R(0) = 0$.
For each set of parameters we graphed (5) and (7) over a time period of two months (so for $T = 60$). The two figures below show the case of the aggressive disease.



Code run with $\alpha = 0.002$, $\beta = 0.01$, $\gamma = 0.01$ and $k = 1000$

Verifying that the adjoint variables respect the transversality conditions, we are certain the μ we computed is the optimal control.
We can also notice that, when the disease is aggressive, the infected will rapidly increase. However, thanks to the effects of vaccination, the susceptible will go towards zero fairly quickly. Furthermore, we notice that there are not many recovered individuals towards the end of the two months. Finally, we observe that vaccination rapidly decreases as there are less susceptible individuals.

Now, we run the code in the case of the weak disease and lower aversion to vaccination.



Code run with $\alpha = 0.002$, $\beta = 0.004$, $\gamma = 0.4$ and $k = 100$

Again, we confirmed that the transversality conditions are satisfied.
In the second case, we notice that, while the infected are increasing, they remain at a fairly low level. Furthermore, we observe that the susceptible are rapidly decreasing, while the recovered settle at around 30% of the population.
Since k is relatively low compared to the first case, we now observe a much higher initial vaccination rate, and a less steep decline towards the end of the two-month period.

4 Incentivising vaccination

As we have seen in the previous section, a controlled, optimal vaccination rate can make a very large difference in the spread of a disease. However, if the population is too adverse to vaccination (namely if k is very large), then the disease can still persevere. This suggests that governments could implement an incentive scheme to increase the number of vaccinated individuals, and ensure that the disease is dealt with more efficiently.

4.1 The population's problem under incentives

Suppose that a government decides to incentivise vaccination by introducing a tax on the population. We assume that this tax is linear in the number of susceptible individuals. We assume that the tax takes the form of a function $C : [0, N] \to \mathbb{R}$ defined by:

$$C(S(t)) = \tau S(t)$$

where $\tau > 0$. Then, the population's objective function becomes:

$$J(\mu) = \int_0^T cI(t) + L(\mu(t)) + C(\mu(t))\, dt$$

and the problem in (5) now becomes:

$$\min_{\mu\in[0,0.9]} J(\mu) = \min_{\mu\in[0,0.9]} \left(\int_0^T cI(t) + \tau S(t) + k\frac{\mu^2(t)}{2}\, dt \right) \tag{9}$$

subject to the system in (4).We once again assume that this system satisfies the sufficient conditions for the existence of an optimum. Hence, we may apply once again Pontryagin's maximum principle.

The Hamiltonian of the system is:

$$H = cI(t) + \tau S(t) + k\frac{\mu^2(t)}{2} + \lambda_1(t)\frac{dS}{dt} + \lambda_2(t)\frac{dI}{dt} + \lambda_3(t)\frac{dR}{dt}$$

Then, we have that $\frac{\partial H}{\partial \mu}$ is the same as in the previous problem, and μ^* is defined as in (7). We also have that $\frac{\partial H}{\partial I}$ and $\frac{\partial H}{\partial R}$ are the same as before. However, we now have:

$$\frac{\partial H}{\partial S} = (-\alpha - \beta I(t) - \mu(t))\lambda_1(t) + \beta I(t)\lambda_2(t) + \mu(t)\lambda_3(t) + \tau$$

so that the new adjoint system is:

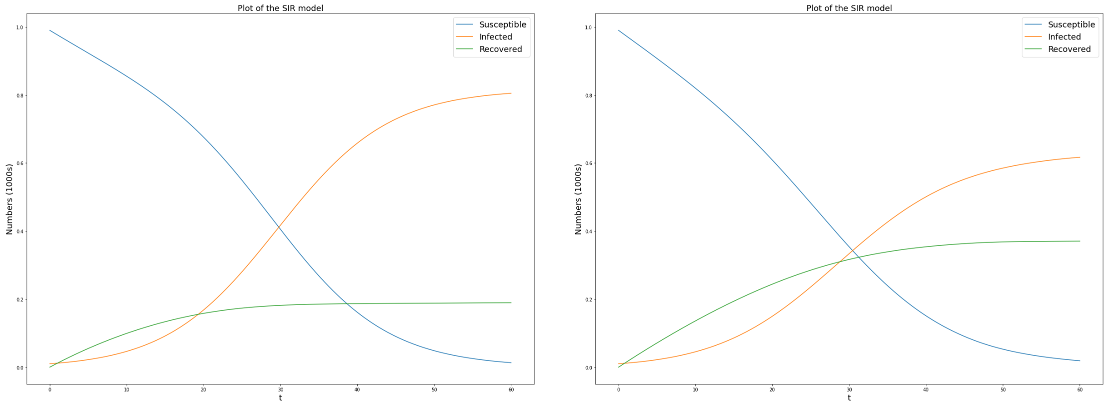
$$\begin{cases} \lambda_1'(t) = (\alpha + \beta I^*(t) + \mu^*(t))\lambda_1(t) - \beta I^*(t)\lambda_2(t) - \mu^*(t)\lambda_3(t) - \tau \\ \lambda_2'(t) = \beta S^*(t)\lambda_1(t) - (\beta S^*(t) - \alpha - \gamma)\lambda_2(t) - \gamma\lambda_3(t) - c \\ \lambda_3'(t) = \alpha\lambda_3(t) \end{cases}$$

Thanks to the transversality conditions $\lambda_1(T) = \lambda_2(T) = \lambda_3(T) = 0$, we once again obtain $\lambda_3(t) = 0$ for all $t \in [0, T]$. Hence, replacing the expression for μ^*, the adjoint system becomes:

$$\begin{cases} \lambda_1'(t) = \beta I^*(t)(\lambda_1(t) - \lambda_2(t)) + \alpha\lambda_1(t) + \frac{S^*(t)}{k}\lambda_1^2(t) - \tau \\ \lambda_2'(t) = \beta S^*(t)(\lambda_1(t) - \lambda_2(t)) + (\alpha + \gamma)\lambda_2(t) - c \end{cases} \tag{10}$$

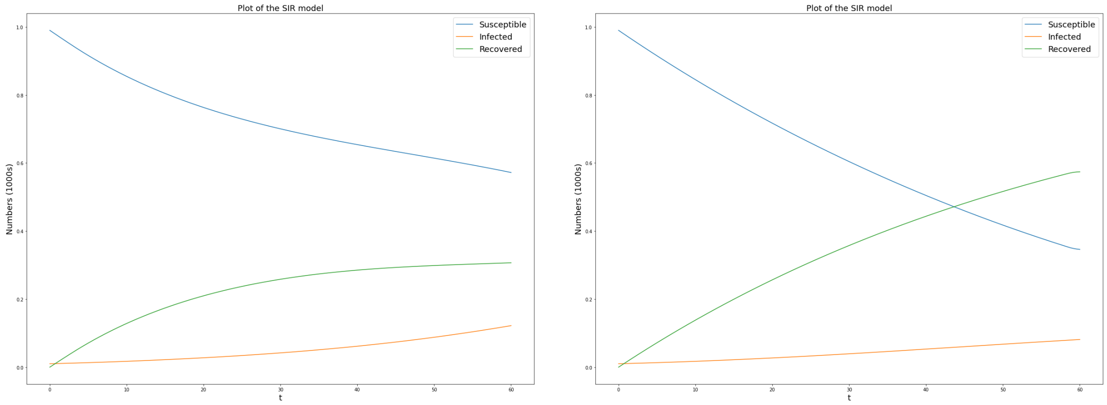
4.2 Final numerical solution

Similarly to section (3.2), we will use the Forward-Backward sweep method to solve the systems in (4) and (10). We will just modify the second *for* loop in our code to take into account the changes in the adjoint system. Taking the same set of parameters as in section (3.2), we will compare the evolution of the state with and without taxation. Note that we have verified that the transversality conditions are met for all the cases studied in this section. Firstly, we will plot the case of an aggressive disease with high aversion to vaccination.



Code run with $\alpha = 0.002$, $\beta = 0.004$, $\gamma = 0.4$, $k = 100$, $\tau = 0$ (left) and $\tau = 100$ (right)

As we can see, introducing a high tax to the population provides a strong incentive to get vaccinated, even though k is very high. This in turn leads to a better outcome, with the total number of infected at the end of the vaccination campaign being lower. Furthermore, we can see that the number of recovered grows more rapidly, and reaches a higher terminal value.
We now simulate the case of a weak disease, but with a lower aversion to vaccination.



Code run with $\alpha = 0.002$, $\beta = 0.004$, $\gamma = 0.4$, $k = 100$, $\tau = 0$ (left) and $\tau = 10$ (right)

We can observe that, even with a relatively low taxation rate, the vaccination drastically improves. This causes the susceptible population to fall at a much faster rate, and conversely the recovered increase faster. We can even observe that, at the end of the two months, there are more recovered individuals than there are susceptible, which was not the case for $\tau = 0$. Finally, we notice that, at the end of the vaccination campaign, there are slightly less infected individuals when a tax is imposed.

5 Conclusion

We have modelled the spread of a disease with the appropriate version of the SIR model. We then simulated it with python for different sets of parameters and discussed the resulting plots. Afterwards, we represented the population's problem of choosing the best vaccination rate through an optimal control problem. Applying Pontryagin's maximum principle, we derived the necessary conditions for the solution, which allowed us to numerically solve for it. To do so, we used the Forward-Backward sweep method relying on a Runge-Kutta fourth order procedure. Using these results, we simulated the spread of the disease when vaccination is optimal. Finally, we repeated this process to solve the optimal control problem with incentivised vaccination and compared the results.

In our study, we found that an optimal vaccination rate can limit the spread of a disease, even without incentives. However, we observed that, if the disease is too aggressive or there is too much aversion to vaccination , the number of recovered individuals does not rise as much as desired, with the majority being infected. This is why we implemented an incentive scheme for vaccination in the form of a linear tax in the susceptible individuals. This implementation greatly improved from the original optimum, with infected decreasing and recovered increasing. This was particularly noticeable in the case of a weak disease, where a two-month vaccination campaign resulted in the majority of the population being immunized.

It's important to mention that our models have limitations. Firstly, the SIR model assumes a constant population, and further that the disease causes no deaths. Additionally, for some diseases, getting infected once does not guarantee immunity, which our model does not take into account. Secondly, the optimal vaccination rates we computed do not account for production, distribution and administration costs which could lead to shortages. In fact, this is a central issue that many countries are currently facing when vaccinating for Covid-19.
Nevertheless, our results are still relevant, as we consider short time periods, which reduces the issue of assuming a constant population. Indeed, in January 2021, only 40% of French citizens wanted to get vaccinated for Covid-19, delaying the French vaccination campaign compared to it's European neighbours[5]. To deal with this issue, countries like Serbia have implemented monetary incentives for vaccination[6].

6 Appendix

The statement of the principle (we will state the minimum principle here) is as follows:

Fix $n \in \mathbb{N}$. Let F be an integrable function taking three arguments: a parameter $t \in \mathbb{R}$, a vector $x : \mathbb{R} \to \mathbb{R}^n$ called the state and a function $u : \mathbb{R} \to \mathcal{U}$ called the control. The function is defined as $(t, x(t), u(t)) \mapsto F(t, x(t), u(t))$. Here, $\mathcal{U}$ is the set of admissible controls for the problem. Then, define the objective function:

$$J(u) = \int_{t_0}^{t1} F(t, x(t), u(t))\, dt$$

with $t_0, t_1 \in \mathbb{R}$. Suppose that we wish to solve:

$$\min_{u \in \mathcal{U}} J(u) \quad \text{subject to} \quad \begin{cases} x'(t) = g(t, x(t), u(t)) \\ x(t_0) = x_0 \end{cases} \tag{11}$$

for some function g. Assume that there exists an optimal control u^* and an optimal state x^* which is the solution of the ODE in (11) with $u = u^*$, then, we define the Hamiltonian of the problem as:

$$H(t, x(t), u(t), \lambda(t)) = F(t, x(t), u(t)) + \lambda(t) g(t, x(t), u(t))$$

where $\lambda : \mathbb{R} \to \mathbb{R}^n$ is a differentiable function. Then the following conditions must hold:

- The optimality condition: $\dfrac{\partial H}{\partial u}(t, x^*(t), u^*(t)) = 0$
- The adjoint equation: $\lambda'(t) = -\dfrac{\partial H}{\partial x}(t, x^*(t), u^*(t))$
- The transversality condition: $\lambda(t_1) = 0$

Note that these are necessary conditions for an existing optimum, they do not prove an optimum's existence.

7 References and Bibliography

1. https://www.beckershospitalreview.com/quality/6-diseases-the-u-s-eliminated-thanks-to-vaccines.html
2. https://www.maa.org/press/periodicals/loci/joma/the-sir-model-for-spread-of-disease-the-differential-equation-model
3. https://scipython.com/book/chapter-8-scipy/additional-examples/the-sir-epidemic-model/
4. Lenhart, S. and Workman, J.T. "Numerical Solutions for Optimal Control"
5. https://www.thelocal.fr/20210226/6-reasons-frances-covid-vaccine-rollout-has-been-so-slow/
6. https://www.pri.org/stories/2021-05-11/cash-beer-doughnuts-can-incentives-work-get-people-vaccinated

Other Sources:

Zaman Gul et al. "Stability analysis and optimal vaccination of an SIR epidemic model". 30 April 2008.
Boscian, Ugo and Piccoli, Benetto. "An Introduction to Optimal Control".