Fitting Model Predictive Control to an Epidemiological Model

Tamás Epres

July 2024

1 Introduction

The past few years have highlighted the fact that controlling and managing a pandemic is not a trivial task. Managing the progression of an epidemic is a complex problem that requires knowledge from multiple scientific disciplines; in this case, we may encounter issues that necessitate engineering problem-solving. In my work, I will primarily illuminate the engineering solutions and tasks necessary for controlling an epidemic from the perspective of control theory; these include the quantization of the control signal and the limitation of the duration of interventions. In solving this problem, I will apply model predictive control to various epidemiological models. Initially, I will introduce the applied strategy, the basics of model predictive control on a simple linear model, and then apply the strategy to a process model describing the epidemiological dynamics of a country. Last but not least, I aim to fit model predictive control to a multi-agent model mimicked by a neural network.

2 Introduction to Model Predictive Control

2.1 Definition of Model Predictive Control

In this section, I will focus primarily on the theoretical considerations of control, emerging problems, and not on the epidemic model. Model predictive control (hereafter referred to by the English abbreviation MPC) is a control strategy and method whose essence is to provide control sequences that minimize a certain error, knowing the dynamics that describe the process. In other words, MPC optimizes an error within a given time window and provides a control signal based on that. Let us start with a simple linear discrete-time process:

$$\mathbf{x_{k+1}} = \mathbf{A}\mathbf{x_k} + \mathbf{B}u_k \tag{1}$$

$$\mathbf{y_k} = \mathbf{C}\mathbf{x_k} \tag{2}$$

Where:

- $\mathbf{y}_k$ is the response vector of the process at time k.
- $\mathbf{x}_k$ is the state vector of the process at time k.
- u_k is the control input at time k.
- $\mathbf{A}$ is the matrix describing the dynamics of the process, the system matrix.
- $\mathbf{B}$ is the vector characteristic of the process that determines the control input.

For simplicity, let's deal with SISO systems and restrict our state variables to be one-dimensional. Thus, y_k and x_k are also one-dimensional, as is u_k. In addition, there is a cost function that defines the error that the algorithm seeks to minimize. In our first example, our goal is to bring the system from an initial state to another state prescribed by us. Thus, the optimization problem takes the following form:

$$\min \sum_{i=0}^{N} (\mathbf{x}_{k+i|k} - \mathbf{x}_{req})^2 \tag{3}$$

$$\mathbf{x}_{k+i+1|k} = \mathbf{A}\mathbf{x}_{k+i|k} + \mathbf{B}u_{k+i|k} \tag{4}$$

$$u_{down} < u_{k+i|k} < u_{up} \tag{5}$$

$$\mathbf{x}_{k+i|k} \in \mathbb{R}^n \tag{6}$$

$$u_{k+i|k} \in U \tag{7}$$

Where:

- N is the time horizon, up to which the system's responses are observed.
- $\mathbf{x}_{k+i|k}$ represents the responses of the process at time k along the time horizon with the variable i.
- $u_{k+i|k}$ is the control signal at time k along the time horizon with the variable i.
- u_{down} is the lower bound of the control signal.
- u_{up} is the upper bound of the control signal.
- $\mathbf{x}_{req}$ is the desired state vector prescribed by us.
- $U \in \{U_0, U_1, \dots, U_9\}$

The objective of the optimization is to minimize equation (3), which is the objective, while equations (4) and (5) represent the constraints to be adhered to. It is evident that both u_k and x_k are the variables to be optimized. Furthermore, while the state vectors of the system can take continuous values, the control signals can only be selected from a set with finite cardinality. This optimization problem is known as a Mixed Integer Linear Problem (MILP).

2.2 Waiting Time Constraints (WTC)

As mentioned earlier, this section does not deal with the epidemiological model but rather addresses the issue of waiting time constraints. The basis of the problem is as follows: a given control action cannot be changed abruptly, and it is not advisable to maintain it for too long due to social and economic reasons. Therefore, we introduce constraints that regulate the duration of control actions, determining how long a control signal can remain active. In other words, we can apply a given control signal to the system at any time; however, if we intervene, the value of the intervention must be maintained for a minimum duration, and after a certain time, this value can no longer be applied until another control action is taken. To solve this, we introduce new variables to be optimized. [1]

$$\mathbf{v_j} = [v_j^0, v_j^1, \ldots, v_j^n]^T$$

Where:

$$v_j^l = \begin{cases} 1, & \text{if the given } u \text{ is active} \\ 0, & \text{otherwise} \end{cases}$$

We also assume that:

$$\sum_{l=0}^{n} v_j^l = 1 \tag{8}$$

Thus, $\mathbf{v_j}$ is a vector where all elements are 0 except for the index corresponding to the value of u at that particular time. For example, if $u = U_1$ and $U \in \{U_0, U_1, \ldots, U_9\}$, then $\mathbf{v_j} = [0, 1, 0, \ldots, 0]$. We also introduce a counter $\mathbf{H}$ that tracks how many v_j values have been active up to a given time point. If a v_j with a different value from the previous one appears in the sequence, the counter resets. Additionally, we introduce two parameters, $\mathbf{W}_{Tmin}$ and $\mathbf{W}_{Tmax}$, which determine how long the given v_j index should be maintained and after what duration it should change its value. Their operation can be described as follows:

$$\mathbf{H}_{j+1} = (\mathbf{H}_j + \mathbf{v}_j) \odot \mathbf{v}_j \tag{9}$$

Where the $\odot$ operator denotes element-wise multiplication, also known as the Hadamard product.

$$\mathbf{v}_j^T \cdot (\mathbf{W}_{Tmin} - \mathbf{H}_{j+1}) \leq M \cdot (\mathbf{v}_j^T \cdot \mathbf{v}_{j+1}) \tag{10}$$

Where M is a large number ensuring that the inequality holds when $\mathbf{v}_j = \mathbf{v}_{j+1}$.

$$\mathbf{H}_{j+1} \leq \mathbf{W}_{Tmax} \tag{11}$$

While equation (9) is responsible for counting, (10) establishes the lower bound, and (11) ensures the upper bound.

2.3 Simulation Results without waiting time constraints

In the following two section, I present some simulation results. The optimization problem was implemented in Python using the Pyomo [2] and NumPy [3] packages, with visualization handled by the Matplotlib [4] package. In this example, I used the Gurobi [5] solver to solve the optimization problem since we are dealing with a convex problem. (A problem is convex if the objective function is convex, the equalities are affine, and the inequalities are convex.) [6]

In this section, I show the simulation result without waiting time constraints. The parameters defining the system dynamics are:

$$\mathbf{A} = 1.1 \qquad \mathbf{B} = 1.0$$

Additionally, the **U** set is defined as follows:

$$\mathbf{U} \in \{-10, -9, -8, \ldots, 10\}$$

Let the initial state $\mathbf{x}_0$ and the reference state $\mathbf{x}_{req}$ be as follows:

$$\mathbf{x}_0 = 0.0 \qquad \mathbf{x}_{req} = 100.$$

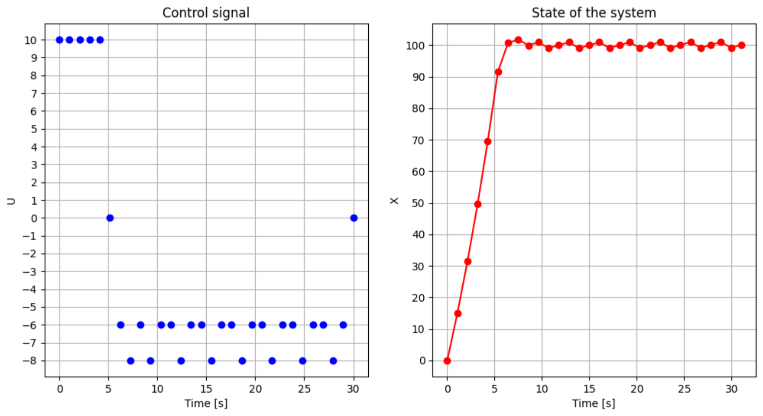


Figure 1: On the left, the control sequence; on the right, the evolution of the system.

As can be seen, the system will oscillate around the response. This is because the control signal does not fit the dynamics and does not take on a value that would direct the system precisely to the reference signal.

2.4 Simulation Results with waiting time constraints

In this section, I show the simulation result with waiting time constraints. The parameters defining the system dynamics are the same with the in the before section.

Furthermore the $\mathbf{W}_{Tmin}$, and $\mathbf{W}_{Tmax}$ are defined as follows:

$$\mathbf{W}_{Tmin} = 2 \qquad \mathbf{W}_{Tmax} = 4$$

The initial state $\mathbf{x}_0$ and the reference state $\mathbf{x}_{req}$ are 0.0 and 100. as well.

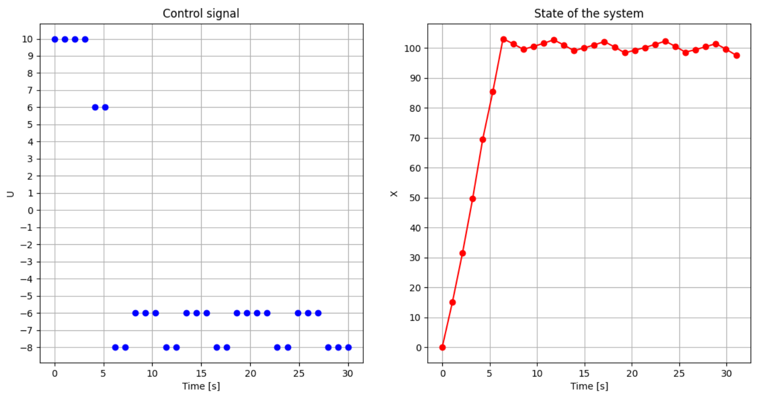


Figure 2: On the left, the control sequence; on the right, the evolution of the system.

We can also observe that the oscillation consists of distinct segments, which is due to the restriction on the duration of interventions. It can be noted that if there were no constraints on the control signal, the system would reach the reference signal without oscillation.

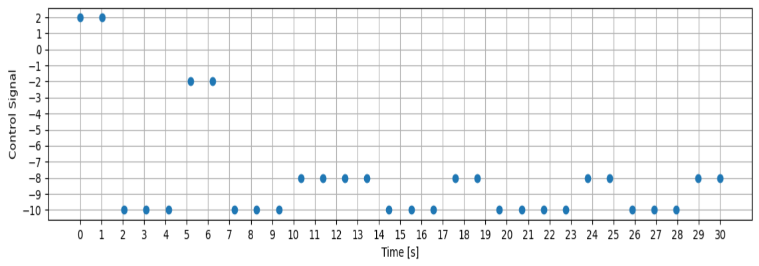
For the sake of completeness, I extended the system to a two-dimensional case. In this case, the parameters were as follows:

$$\mathbf{A} = \begin{pmatrix} 1 & 1.4 \\ 1.4 & 1. \end{pmatrix} \qquad \mathbf{B} = [1., 1.]$$

The $\mathbf{U}$ set, as well as the $\mathbf{W}_{Tmin}$ and $\mathbf{W}_{Tmax}$ parameters, are identical to those in the previous one-dimensional case. The time horizon in this case, as in the one-dimensional case, was 30 s.

The initial state $\mathbf{x}_0$ and the reference state $\mathbf{x}_{req}$ are as follows:

$$\mathbf{x}_0 = [0., 0.] \qquad \mathbf{x}_{req} = [10., 10.]$$



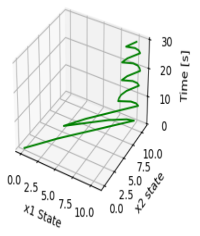


Figure 3: Above: control sequence; below: system evolution.

3 Epidemic Process Model

3.1 Description of the Model Dynamics

In the following section, we will examine the epidemic model for which the strategy discussed in the previous chapter was developed, with slight modifications. The essence of the process model is as follows: the society is divided into different groups depending on the stage of infection they are in. The transition from one group to another is described by differential equations, meaning this model is a continuous-time model. Let us see the individual groups and the description of the dynamics.

[7] The uninfected population (S) consists of those who have not yet contracted the infection but may potentially become infected. The latent (L) population includes those who have encountered the infection but do not yet show any symptoms. Subsequently, the population moves to the pre-symptomatic infectious (P) group, from which individuals either progress to the symptomatic infected (I) group or the asymptomatic infected (A) group. Those who do not show symptoms are likely to recover from the disease, which requires defining the recovered (R) group. Those who show symptoms either recover and move to the R group or are hospitalized (H). Those in the hospital either recover or die (D). The dynamics are described by the following differential equations:

$$S'(t) = -\beta \frac{[P(t) + I(t) + \delta A(t)]S(t)}{N} \tag{12}$$
$$L'(t) = \beta \frac{[P(t) + I(t) + \delta A(t)]S(t)}{N} - \alpha L(t) \tag{13}$$
$$P'(t) = \alpha L(t) - pP(t) \tag{14}$$
$$I'(t) = qpP(t) - \rho_1 I(t) \tag{15}$$
$$A'(t) = (1 - q)pP(t) - \rho_A A(t) \tag{16}$$
$$H'(t) = \rho_1 \nu I(t) - hH(t) \tag{17}$$
$$R'(t) = \rho_1(1 - \nu)I(t) + \rho_A A(t) + (1 - \mu)hH(t) \tag{18}$$
$$D'(t) = \mu hH(t) \tag{19}$$

Where:

- β: The infection rate.
- δ: The infectiousness ratio in the $A(t)$ group.
- N: The total population size.
- α: The rate of transition from the $L(t)$ state.
- p: The rate of transition from the $P(t)$ state.
- q: The probability that an individual in the $P(t)$ state moves to the $I(t)$ state.
- ρ_1: The recovery or transition rate of infected individuals ($I(t)$).
- ρ_A: The recovery or transition rate of asymptomatic individuals ($A(t)$).
- ν: The probability that an infected individual ($I(t)$) requires hospitalization.
- h: The mortality rate of hospitalized patients ($H(t)$).

The dynamics are illustrated in the following graph:

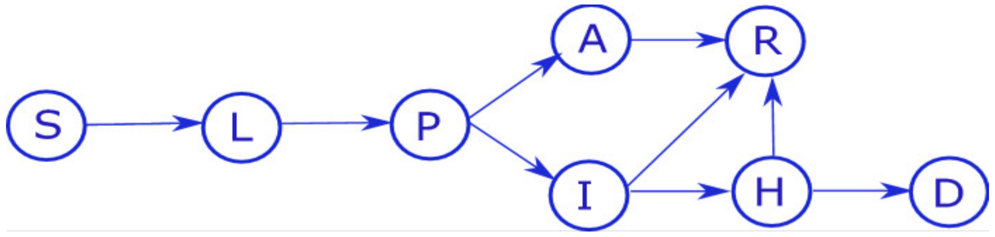


Figure 4: Population dynamics over the course of the epidemic.

Since we need a discrete-time model for optimization, the system of differential equations mentioned above must be solved using a numerical method. In my solution, I used a fourth-order Runge-Kutta method [8]. For numerical stability, I normalized the states by the size of the entire population.

Regarding the model, we can confirm that the model has a conserved property, meaning that if we sum all the equations, we would get exactly one when normalized by the population size. That is, no new individuals are created in the model. Furthermore, it can be established that, if the states are normalized in the system:

$$\lim_{t\to\infty} |S(t) + R(t) + H(t)| = 1$$

This means everyone will end up in either the S, R, or D groups. The model does not address the mechanism of reinfection.

We can now turn to the optimization problem. As shown in the previous chapter, the problem consists of an objective and its constraints.

$$min \sum_{i=0}^{N} u_i^2 + (S_i - S_{term})^2 \tag{20}$$

$$\mathbf{x}_{i+1} = F(\mathbf{x}_i) \tag{21}$$

$$\mathbf{x}_i[5] <= 20000 \tag{22}$$

Where:

- N is the time horizon.
- i is the running variable.
- u_i is the intervention (control input) introduced at time i.
- $\mathbf{x}_i$ is the state vector of the system at time i.
- $\mathbf{x}_k[5]$ is the number of people in the hospital at time i.
- S_{term} is the terminal state of the system, i.e., the state of the S group when the epidemic has definitely subsided.
- S_i is the size of the S group at time i.

Equation (20) represents the objective of the pandemic, which is to minimize the value of interventions to reduce economic damage and bring the system to a state where the epidemic has subsided ($\mathbf{x}_{term}$ terminal state). Equation (21) represents the sampling of the system dynamics, i.e., how we transition from the previous state to the next. Equation (22) imposes a constraint that the number of hospitalized patients should not exceed a certain value to prevent capacity issues in the public health system.

We can also address the concept of the terminal set or terminal state. [1] A terminal state refers to the condition where, even with zero interventions, the number of hospitalized patients strictly decreases monotonically, indicating the development of herd immunity. Clearly, if we reach the terminal set, the epidemic will no longer spread. These are the states we aim to achieve during control, as reflected in the second term of the (20) expression, similar to how we aimed to achieve a reference state in the first example. Determining where these terminal states are is a very challenging problem. It is not trivial to determine whether a state is reachable in a nonlinear model. In my solution, I used a brute-force approach. By starting from random initial states, I observed the system to see whether those states were stable or not. The random initialization was implemented so that the individual groups would only take "reasonable" values and not unrealistic values.

3.2 Simulation Results without waiting time constraints

The solution was implemented in Python using the Pyomo and NumPy libraries, just as in the first chapter. The Baron solver [9] was used to solve the optimization problem. Below are the parameters used for the simulation:

$$\mathbf{x}_{init} = [9{,}800{,}000{-}50, 50, 0., 0., 0., 0., 0., 0.]$$

That is, the total population is 9,800,000, while the number of people in the latent group is 50. Additionally, the time-related parameters are:

$$t_{end} = 180 \qquad dt = 1$$

The total time horizon was set to 180 days, with the constraint that decisions can only be made weekly, and each decision must be maintained for one week. The sampling time is 1 day. In that case the partition of the control signal is dense.

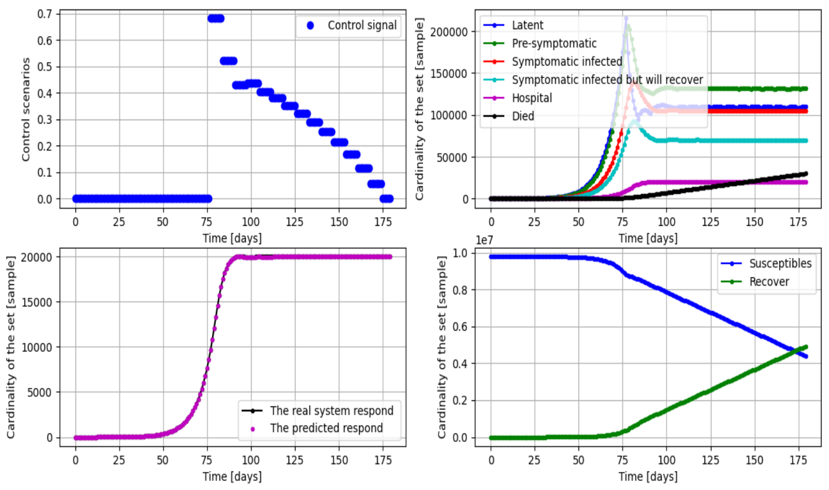


Figure 5: Changes in the system over time. The intervention sequence is shown in the upper left corner. The upper right corner shows the number of latent states and deceased individuals. The lower right corner displays the actual (continuous) and predicted (dashed) number of people in the hospital. The lower left corner shows the number of recovered and still uninfected individuals.

The following observations can be made: if we prescribe the number of people in the hospital, we can only reach the terminal set with a certain slope and speed, meaning time must pass to achieve herd immunity. Therefore, the larger the hospital capacity, the faster we can get through the epidemic. It is also evident that the size of the susceptible group S, and the groups capable of infection (L, P, A, I) is critical. Their relative sizes determine whether a second wave or a new peak in the hospital group H might occur with no further interventions.

3.3 A New Strategy, Improvement of the Control

In this section, I will present another approach to controlling the epidemic. It's clear that if we don't change the behavior of the system but write different constraints and objective functions, we will derive different strategies and different trajectories of the epidemic. We can examine the following statements and questions:

- The purpose of the control is to reach the terminal set rapidly thus, what is the point of minimizing the control input?
- Defining the maximum hospital capacity as a fixed number is unrealistic. We assume that in real life, the number of patients will oscillate around the desired value.
- We should verify that we have reached the terminal set and ensure that a second wave does not emerge in the evolution of the system.
- What are the exact values of the control input?
- What happen if we don't reach the terminal set? The concept of the quasi terminal set.

In my opinion, it is more effective to reach the terminal set quickly rather than to minimize the control inputs. To support this view, let's consider the following situation. Suppose we aim to minimize the expression $min \sum_{k=0}^{N} u_k^2$ the time horizon is 180 days and the hospital capacity is 20000 and the number of the initial latent patients is 80.

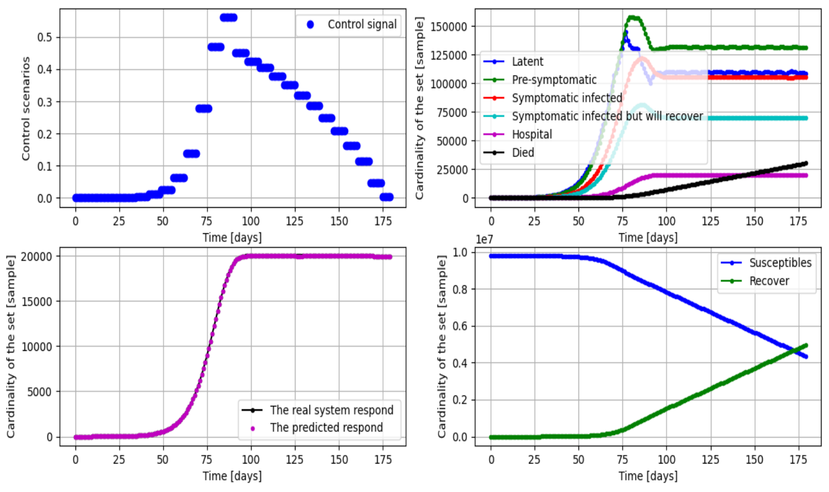


Figure 6

I will also present the results when the main goal is to reach the terminal set. In this case, the terminal set is 40% of the real population, and the objective function is $\min \sum_{i=0}^{N}(S_i - S_{term})^2$:

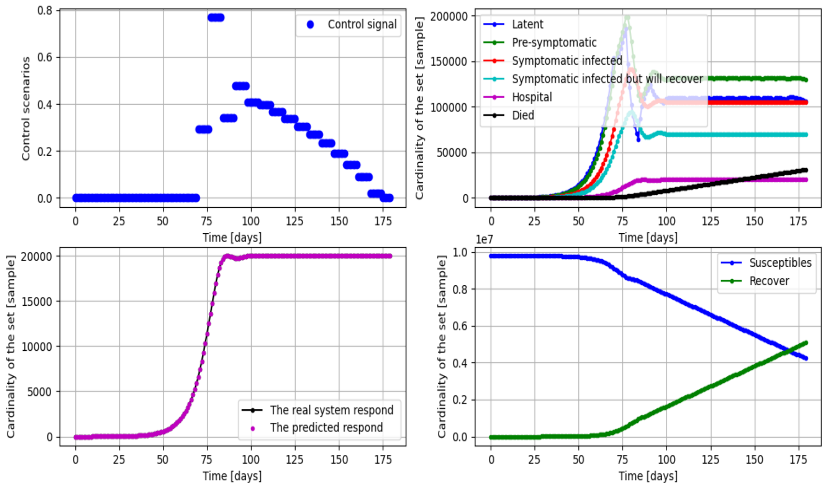


Figure 7

As we can see, there is no significant difference in strategy between the two examples. We observe that, in the second instance, we reach the terminal set less frequently. The reason is that, in the first example, we use the control input from the beginning, which delays the progression of the epidemic. We can assume that we should not interfere until the maximum capacity is reached to achieve herd immunity more quickly. In my view, if we focus solely on reaching the terminal set, we can overcome the epidemic faster, and the economic losses might be less than if we only minimize the control input. It is important to note that we should reach the terminal set by the end of the time horizon.

We can observe the "staircase" strategy in Figures 4, 5, and 6. This is interesting because, in real life, we can only use 10-20 different input values. Why does this happen? The answer lies in the strict constraints on the number of patients. Since the number of groups capable of infection and the and as the number of patients constant, and the susceptibles decreases, fewer input values are required over time. In reality, we cannot insist that the number of patients should never exceed the capacity; the numbers will oscillate due to inaccuracies in the model or because of effects that we do not observe or measure. This is why I reformulated the hospital capacity constraint. We aim to soften this constraint by stating that it should only be satisfied every 15th day. In this case, we set the hospital capacity to 18500 to account for the oscillation. Furthermore, we use an input value of 0 after the 200th day to demonstrate that we have overcome the pandemic. In this case, the initial number of latent individuals is 100 and the $t_{min} = 2$ days.

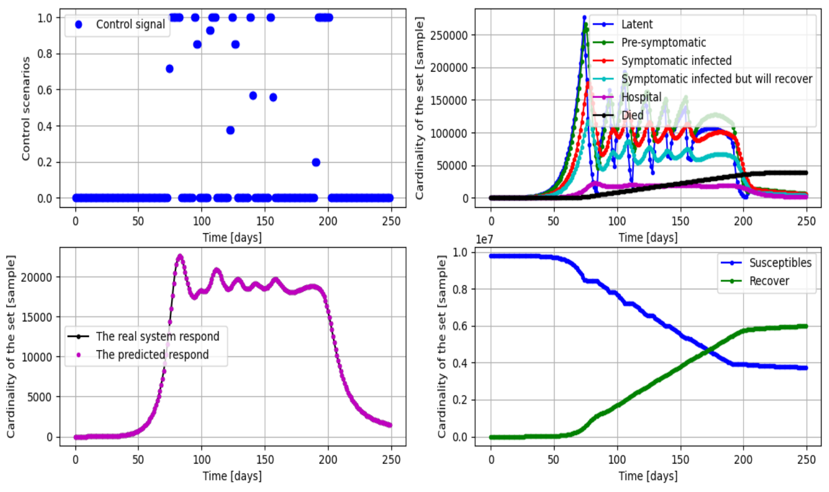


Figure 8

We can clearly see that after reaching the terminal set, which is 39% of the real population, we have overcome the epidemic. Furthermore, we observe that the control inputs have only 10 different values. The domain of the control signal is as follows:

$U \in [0.0, 0.196396, 0.376978, 0.55669, 0.6943, 0.714966, 0.849426, 0.851748, 0.929108, 1.0]$

To prove that we have found a terminal set, I ran a 1750-day simulation with this control input.

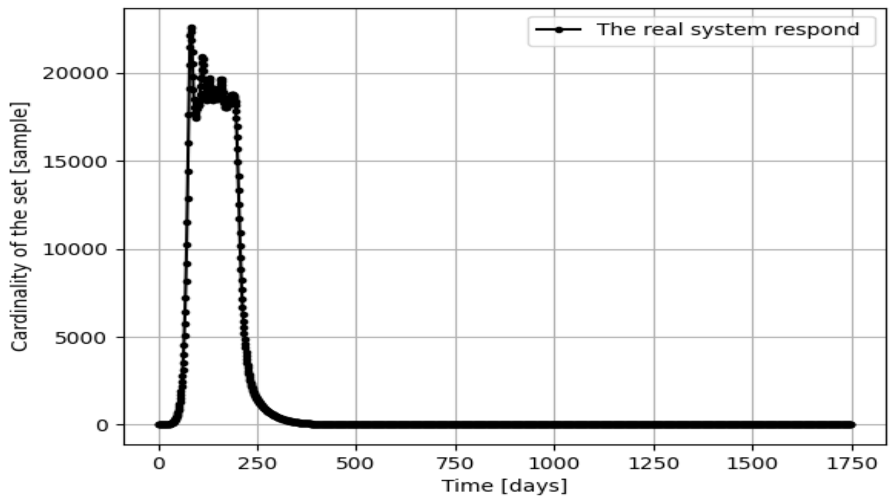


Figure 9: In the x axis show the time in days and in the y axis there is the number of the patients.As we can see, there is no second wave after we release the control input.

At his point, I will introduce the concept of a quasi-terminal set. In control theory, when managing dynamic systems, it's crucial to consider what happens if the system does not reach a terminal set by the end of the control horizon. Specifically, in the context of hospital management, if we fail to reach a terminal set, the number of patients may continue to grow over time, potentially exceeding the hospital's capacity.

A set is considered a quasi-terminal set if, although it is not a terminal set, releasing the control input will not result in the number of patients in the hospital exceeding the hospital's capacity. In other words, while the system does not settle into a desired terminal state, it remains within acceptable bounds regarding patient numbers.

To illustrate this concept, let's consider an example:

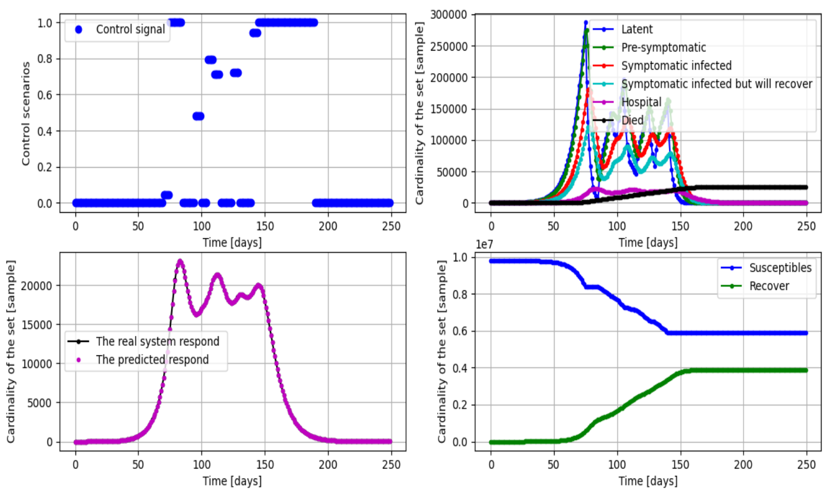


Figure 10.

In this case the goal is to reach a quasi terminal set which in this example is the 60% of the real population. Furthermore $t_{min} = 5$ is days and after the 200th day we use 0 control input. The domain of the control signal is as follows:

$$U \in [0.0, 0.04294, 0.48284, 0.71459, 0.72295, 0.79512, 0.94266, 1.0]$$

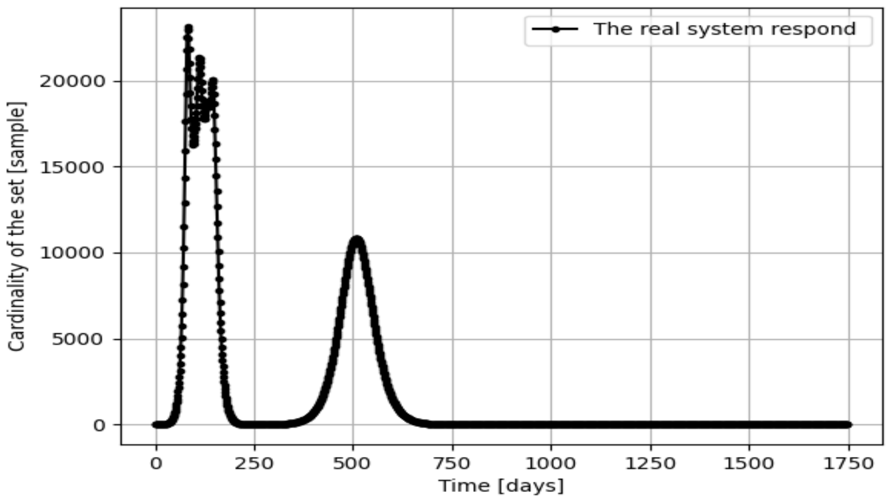


Figure 11: As we can see the number of the patients are always below the hospital capacity.

Based on the simulation results, we can identify all quasi-terminal sets. By pinpointing the exact set where the number of patients reaches the hospital capacity, we can conclude that all sets below this threshold are quasi-terminal sets. This approach ensures that patient numbers are managed effectively, avoiding scenarios where the hospital capacity is exceeded, even when the control input is removed. This is about 66% of the real population.

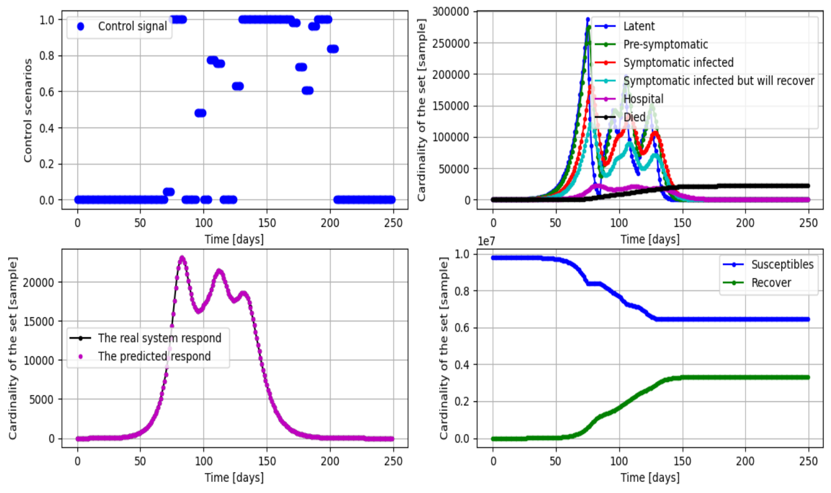


Figure 12

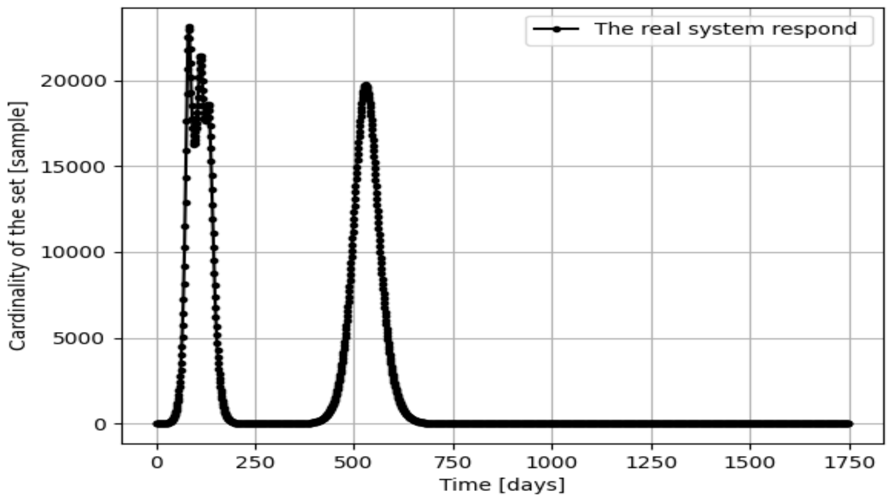


Figure 13

3.4 Simulation with waiting time constraints

With waiting time constraints, the complexity of the epidemic model increases significantly, so in this section, I have restricted the parameters to much smaller values. Let:

$$t_{end} = 10 \qquad t_{min} = 2 \qquad t_{max} = 4 \qquad dt = 1$$

The dynamics and parameters of the epidemic are the same as in the previous section. Let the initial conditions also be the same as in the previous chapter, but the maximum number of patients should be 1.1. The domain of the control signal is as follows:

$$U \in \{0, 0.1, \ldots, 0.8, 1\}$$

For the sake of completeness, I show the results with waiting time constraints and without the waiting time constraints. We assume that in the begin the control signal is 0. In this case we minimalize the $\sum_{k=0}^{N} u_k^2$ expression because in this scale we can't reach the terminal set.

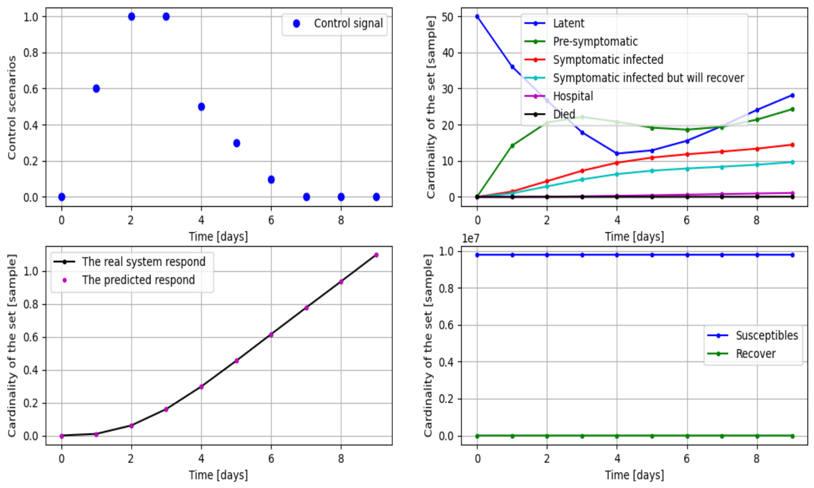


Figure 14: The evolution of the system without waiting time constraints.

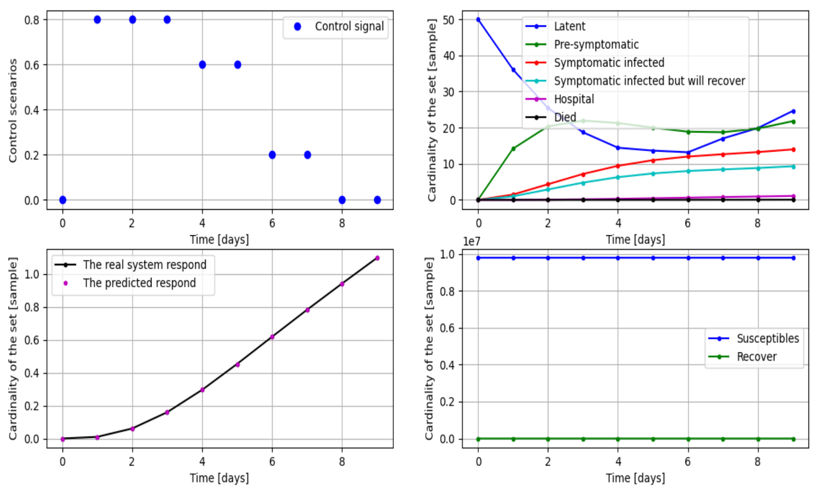


Figure 15: The evolution of the system with waiting time constraints.

As we can see, we found similar strategies, and the evolution of the systems is the same. In this case, in my opinion, the optimization problem behaves like a minimal energy problem. Because we want to reach the 'minimal energy' of the system, we found similar progress.

References

[1] A. F. E. H.-V. A. G. J.E. Serenoa, A. D'Jorgea, *Switched NMPC for epidemiological and social-economic control objectives in SIR-type systems*, 2023.

[2] "Pyomo documention 6.8.0." [Online]. Available: https://pyomo.readthedocs.io/en/stable/

[3] "Numpy documention." [Online]. Available: https://numpy.org/doc/

[4] "Matplotlib 3.9.2 documentation." [Online]. Available: https://matplotlib.org/stable/index.html

[5] "Gurobi solver documention." [Online]. Available: https://www.gurobi.com/documentation/

[6] L. V. Stephen Boyd, *Convex Optimization*, 2004.

[7] G. S. G. R. Tamás Péni, Balázs Csutak, *Nonlinear model predictive control with logic constraints for COVID-19 management*, 2020.

[8] "Runge-kutta method." [Online]. Available: https://en.wikipedia.org/wiki/Runge%E2%80%93Kutta_methods

[9] “Baron solver documention.” [Online]. Available: https://minlp.com/downloads/docs/baron%20manual.pdf