TÉCNICO LISBOA

Instituto Superior Técnico

Estimação e Controlo Preditivo Distribuído / Distributed Predictive Control and Estimation

MEEC

Laboratory work, 2nd Semester 2022/2023

Students:
Tomás Marques Videira Fonseca | 66325 |
tomas.mvf@gmail.com
Afonso Brito Caiado Alemão | 96135 |
afonso.alemao@tecnico.ulisboa.pt
José Miguel Valério Antunes | 96258 |
jose.m.valerio.antunes@tecnico.ulisboa.pt
Rui Pedro Canário Daniel | 96317 |
ruipcdaniel@tecnico.ulisboa.pt

Group **2**

Teachers:
João Miranda Lemos, Pedro Batista

Ethical commitment to originality: The group of students identified above guarantees that the text of this report and all the software and results delivered were entirely carried out by the elements of the group, with a significant participation of all of them, and that no part of the work or the software and results presented was obtained from other people or sources.

June 2, 2023

0 Introduction

This laboratory work aims to stimulate practice with basic concepts of model predictive control (MPC) and then to obtain the solution of a control problem with MPC using a software package.

1 P1

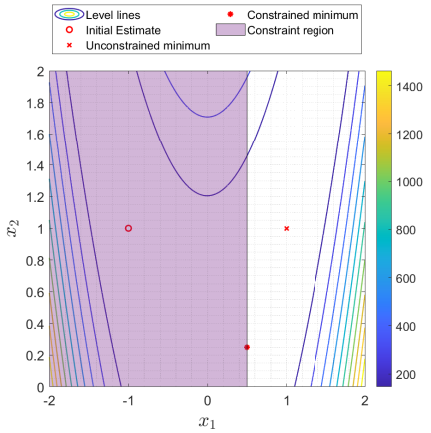
MPC is based on on-line minimization of a cost function to obtain the value of the control variable to apply to the plant, allowing to enforce constraints. This section's objective is to get acquainted with the basics of numerical function minimization using Matlab Optimization Toolbox.

1.1

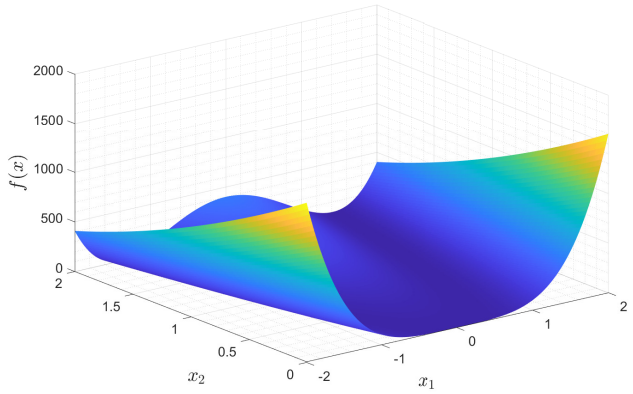
Using the Matlab solver `fminunc`, we compute the unconstrained minimum of the Rosenbrock function given by (1), considering x as in (2). Then, we use `fmincon` for constrained function minimization, by computing the minimum that satisfies the constraint in (3) with $A = [1 \;\; 0]$ and $B = 0.5$. In both cases, we take x_0, given by (2), as the initial estimate of the minimum. In Fig. 1(a), in the same plot, we show the level lines of the function (using Matlab function `contour`), the initial estimate (x_0), the unconstrained minimum obtained (`xopt` $= [1 \;\; 1]^T$), and the constrained minimum (`xoptconstr` $= [0.5 \;\; 0.25]^T$). We also plot a 3-D view of the function in Fig. 1(b) (using Matlab function `surf`).

$$f(x) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2 \qquad (1) \qquad\qquad x = [x_1 \;\; x_2]^T, \quad x_0 = [-1 \;\; 1]^T \qquad (2)$$

$$x_1 \leq 0.5 \iff Ax \leq B \qquad (3)$$



(a): Level lines, initial estimate, and unconstrained and constrained minimums.



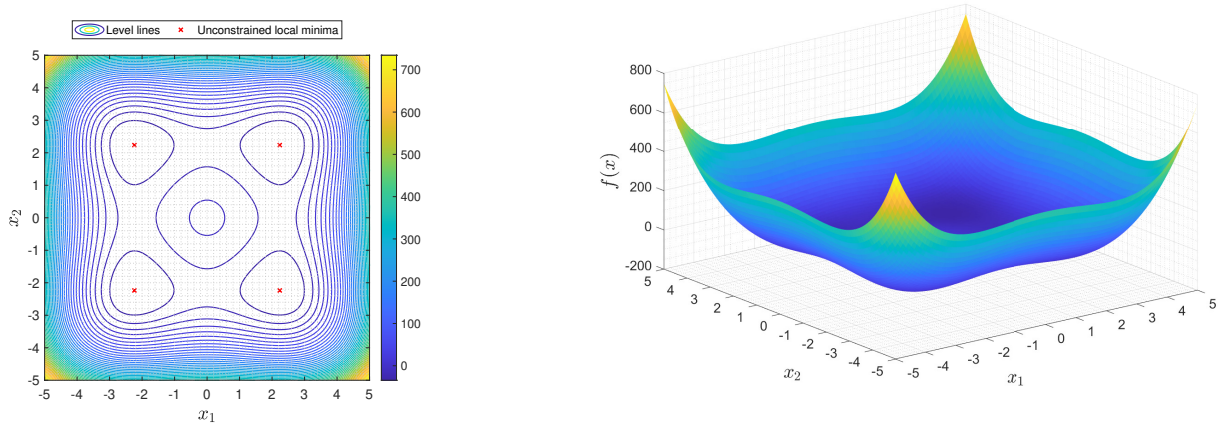
(b): 3-D view of the function.

Figure 1: Analysis of Rosenbrock function.

1.2

Using the same software tools as in the previous question, we compute the several local minima of function (4). Each point in the grid was used as an initial estimate, and then for each one, we got the unconstrained minimum using the `fminunc` function, obtaining a total of 4 different local minima. The grid was defined by a set of 100×100 points ranging from $x_1 \in [-5; 5]$ and $x_2 \in [-5; 5]$ since all of the 4 obtained minimizers are located in this region and (4) has degree four. In Fig. 2(a) we plot the level lines of the function and its several local minimizers. Also, we report a 3-D view of the function in Fig. 2(b). The code for this implementation is in `P1_2_obtained_min.m`.

$$f(x) = x_1^4 - 10x_1^2 + x_2^4 - 10x_2^2 \qquad (4)$$



(a): Level lines and unconstrained minimums.

(b): 3-D view of the function.

Figure 2: Analysis of function (4).

There are 4 local minimizers: x_{opt_1}, x_{opt_2}, x_{opt_3} and x_{opt_4}, given by:

$$a = 2.2361, \quad x_{opt_1} = [-a, -a]^T, \; x_{opt_2} = [-a, a]^T, \quad x_{opt_3} = [a, -a]^T, \; x_{opt_4} = [a, a]^T$$

For all of them, $f(x_{opt_i}) = -50$. Then, we considered one of the local minimizers (x_{opt_1}). The next objective was to find an estimate of the boundary of its attraction basin, i.e., the set of initial points, in which applying the unconstrained minimization algorithm converge to the considered minimum.

Initially, we used a "brute force" approach and then improved the algorithm in order to reduce the number of performed optimizations. The main difference is how we select the initial points.

1.2.1 Initial approach: brute force

The initial approach performs an optimization for each grid point as an initial estimate: $N_1 \times N_2$ points in the range $x_1 \in [x_{1_{min}}; x_{1_{max}}]$ and $x_2 \in [x_{2_{min}}; x_{2_{max}}]$, with $N_1 = N_2 = 200$, $x_{1_{min}} = x_{2_{min}} = -80$, $x_{1_{max}} = x_{2_{max}} = 5$. This way, it tests if each grid point belongs to the convergence region. The initial points that converged were stored. Then, the corresponding boundary was determined using the Matlab function `boundary`. The code for this implementation is in `P1_2_brute_force.m`. The obtained results are reported in Fig. 3(a).

This approach is computationally intensive because it performs an optimization for each grid point in order to test if it belongs to the convergence region: $N_1 \times N_2 = 40000$ optimizations. In this case, there are points outside the largest connected convergence region containing the minimum that were stored in the convergence region, compromising the accuracy of the attraction basis boundary estimation.

1.2.2 Improved approach

In order to reduce the region of selected initial points, the main idea of our improved approach is to perform the optimization only for a few points inside the convergence region.

Our reference point is x_{opt_1}. Then, starting from it, a wide range of possible directions (different angles θ) are explored. For each direction, we search for the boundary by varying the radius using the following method. In the initial stage, the radius increases as a function of $2^n - 1$, multiplied by the initial step, being n how many points we have advanced in that direction until we reach the maximum step value, 4. This value was experimentally determined by a series of tests.

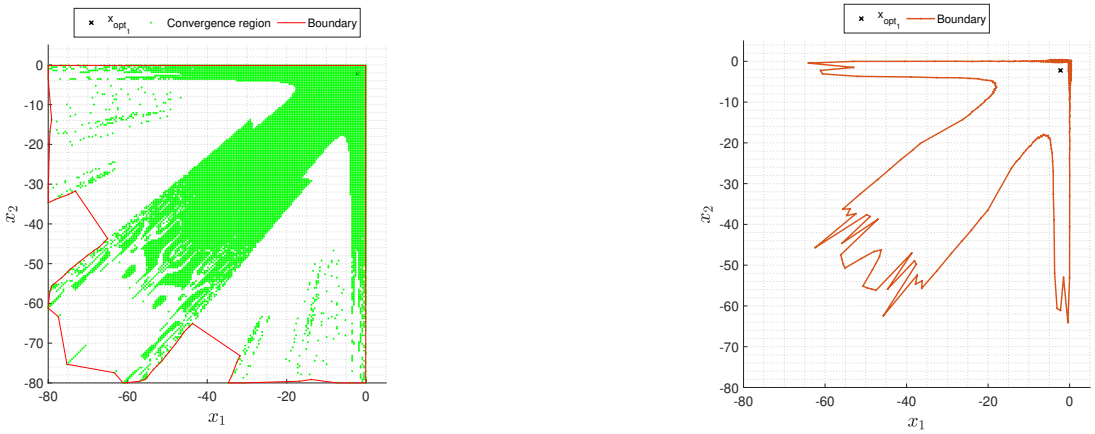
After finding a point outside of the convergence region, the search method changes. From here on, each time a point is found outside (inside), the radius decreases (increases) half of the absolute value of the current step and the step is updated. This will continue until the step is lower than a certain threshold (`error_tolerance_radius` $= 0.2$). With this approach, in order to find the boundary, we only need to store the limit point of the connected convergence region for each direction.

The number of directions comes from the number of times we vary θ, i.e., by setting this value to $360 \times 1.2 = 432$, we will check every $0.8(3)^\circ$ of the 360° circle. The angle and radius steps were chosen

with the objective of achieving a good trade-off between computational load and the accuracy of the attraction basis boundary estimation.

An improvement made was, for each direction, starting with the appropriate radius step to where we expect to find the boundary: boundary of $\theta_i \approx$ boundary of θ_{i-1}. So for each direction the step radius is initialized to `MIN(MAX(final radius of` θ_{i-1}`; step_default); step_max`), with `step_default`$= 0.5$ and `step_max`$= 4$.

The obtained results are reported in Fig. 3(b). In this case, the program performs fewer optimizations (`num_optim` $= 3303 << 40000$), and the accuracy of the attraction basis boundary estimation is higher, i.e., we obtain approximately the connected region that includes the minimum.



(a): Brute force approach.

(b): Improved approach.

Figure 3: Boundary of x_{opt_1}, obtained with two different approaches.

2 P2

In this section, the goal is to compute the state feedback gain yield by the receding horizon (RH) strategy in a problem with linear dynamics, quadratic cost and without constraints, and to compare it with the infinite horizon gain. In sections 2.1-2.4 we study the open-loop unstable 1^{st} order plant given by (5) (**Plant 1**) and the open-loop stable 1^{st} order plant (6) (**Plant 2**). As requested, $Q = 1$ and we select values of R yielding representative results: $R = \{0.01;\ 0.1;\ 1;\ 10;\ 100\}$. The computations for this section were performed in the code in file `P2.m`.

$$x(t+1) = 1.2x(t) + u(t) \qquad (5) \qquad\qquad x(t+1) = 0.8x(t) + u(t) \qquad (6)$$

2.1

The infinite horizon linear quadratic (LQ) optimal control problem consists of minimizing the infinite horizon unconstrained quadratic cost given by (7) subject to (8) in order to compute the optimal LQ state feedback gain, K_{LQ}. For Plant 1, $A = 1.2$ and $B = 1$, and for Plant 2, $A = 0.8$ and $B = 1$. The optimal LQ control is specified by the linear state feedback in (9).

The optimal LQ control exists if (A, B) is stabilizable, $R \succ 0$ and $Q \succeq 0$, and (Q, A) has no unobservable mode on the unit circle, i.e., if (C, A) is observable with $Q = C^T C$. A state realization given by (8) is said to be stabilizable if there is a state feedback given by (9) such that the closed-loop is asymptotically stable. Equivalently, (A, B) is stabilizable if there is a vector of gains K_{LQ} such that the eigenvalues of $A - BK_{LQ}$ are all inside the unit circle [1]. So, in this case, the system is stable if condition (10) verifies. Since the observability matrix $O(C, A) = [C \quad CA \quad \dots \quad CA^{n-1}]^T$ verifies $\text{rank}[O(C, A)] = n = \dim(x) = 1$, (C, A) is observable. As requested, we take $Q = 1 \geq 0$, implying $C = 1$, and the selected values of R guarantee $R > 0$.

$$J_{LQ}(u) = \sum_{t=1}^{\infty} x^T(t)Qx(t) + u^T(t)Ru(t) \quad (7) \qquad x(t+1) = Ax(t) + Bu(t) \quad (8)$$

$$u(t) = -K_{LQ}x(t)\ , \quad K_{LQ} = (B^TSB + R)^{-1}B^TSA \quad (9)$$

$$|\text{eigenvalue}(A - BK_{LQ})| < 1 \quad (10)$$

Using the function `dlqr` we compute the discrete-time K_{LQ}, S (solution of Riccati equation) and the vector of eigenvalues of the closed-loop system dynamics $A - BK_{LQ}$ and report them, for both plants, in Table 1. In all studied cases, condition (10) is verified, so both systems are stable for infinite horizon.

Table 1: Obtained optimal K_{LQ} and the eigenvalue of the closed-loop system dynamics for both plants.

Plant 1					
R	0.01	0.1	1	10	100
K_{LQ}	1.1883	1.1026	0.7935	0.4882	0.3844
eig$(A - BK_{LQ})$	0.0117	0.0974	0.4065	0.7118	0.8156

Plant 2					
R	0.01	0.1	1	10	100
K_{LQ}	0.7921	0.7309	0.4624	0.1399	0.0207
eig$(A - BK_{LQ})$	0.0079	0.0691	0.3376	0.6601	0.7793

2.2

In this question, we consider the finite horizon unconstrained quadratic cost defined over a horizon with H steps, as stated in (11). As the plant input is scalar, R is a positive scalar weight. The output equation is given by $y(t) = Cx(t)$. Using the state model, and defining Y, U, Π as in (12) and W as in [2], the optimal receding horizon control is given by the state feedback in (13) with the optimal feedback receding horizon gain K_{RH}, where e_1 has dimension H.

$$J_{RH}(u;t) = \sum_{i=0}^{H-1} x^T(t+i+1)Qx(t+i+1) + Ru^2(t+i) \quad (11)$$

$$Y = [y(1) \ \ \ldots \ \ y(H)]^T\ , \quad U = [u(0) \ \ \ldots \ \ u(H-1)]^T\ , \quad \Pi = [CA \ \ CA^2 \ \ \ldots \ \ CA^H]^T \quad (12)$$

$$u(t) = -K_{RH}x(t)\ , \quad K_{RH} = e_1M^{-1}W^T\Pi\ , \quad e_1 = [1 \ \ 0 \ \ \ldots \ \ 0]\ , \quad M = W^TW + RI \quad (13)$$

K_{RH} was obtained for different values of the horizon H and R. The range for H was $\{1, 2, \ldots, H_{max}\}$. In order to obtain representative results of the convergence of this gain with the increase of H the value chosen for H_{max} was 30. In Fig. 4, for each R, we report K_{RH} as a function of H, superimposed on a line that corresponds to K_{LQ}, for both plants. For each R, with the increase of H, K_{RH} increases, converging to K_{LQ}. The value of H from which K_{RH} reaches a value in the neighborhood of $\pm 1\%$ of K_{LQ}, $H_{1\%}$, is reported in Table 2.

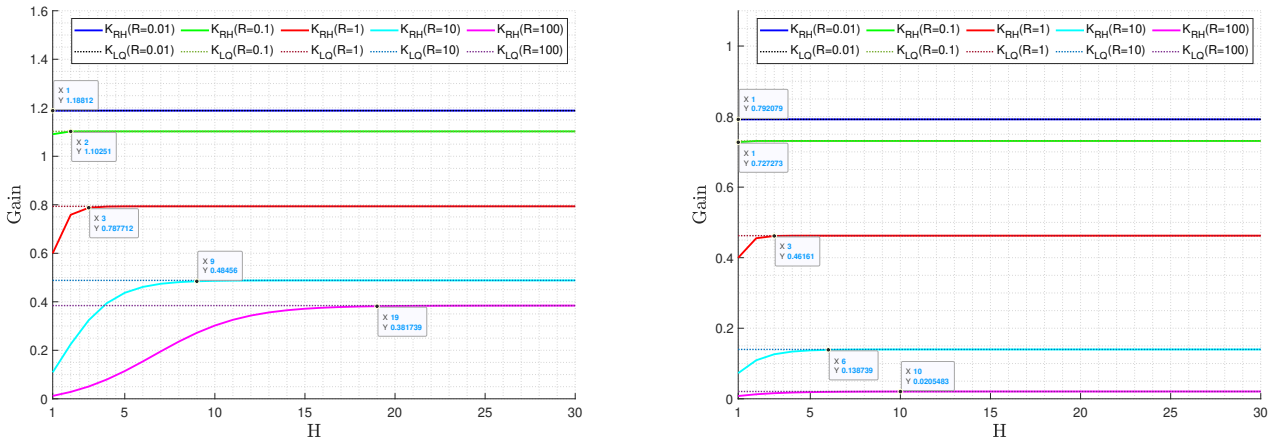
Table 2: The horizon, $H_{1\%}$, from which K_{RH} is in the neighborhood of $\pm 1\%$ of K_{LQ}.

	R	0.01	0.1	1	10	100
$H_{1\%}$	Plant 1	1	1	3	6	10
	Plant 2	1	2	3	9	19

2.3

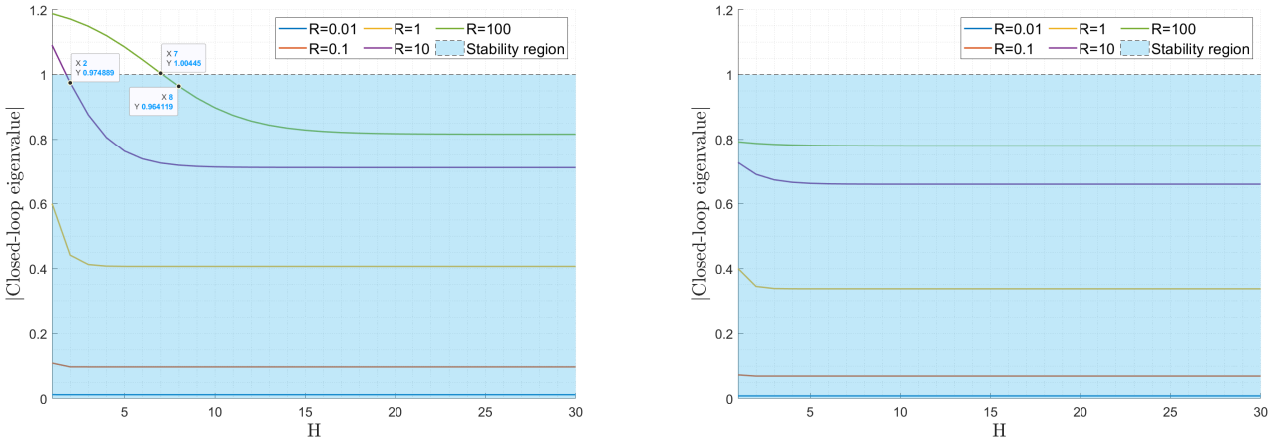
In Fig. 5, for both plants, we report the absolute value of the closed-loop eigenvalue (the eigenvalue of $A - BK_{RH}$) as a function of H, for each R, using $H_{max} = 30$. As described in 2.1, a state realization given by (8) is said to be stabilizable if there is a vector of gains K such that the eigenvalues of $A - BK$ are all inside the unit circle. So in this case, the system is stable if the condition (14) verifies. There is a stability region shown in Fig. 5 where the stability boundary is $|\text{eigenvalue}(A - BK_{RH})| = 1$.

$$|\text{eigenvalue}(A - BK_{RH})| < 1 \quad (14)$$



(a): Plant 1. (b): Plant 2.

Figure 4: K_{RH} as a function of H superimposed on a line that corresponds to K_{LQ}, for each R.



(a): Plant 1. (b): Plant 2.

Figure 5: Absolute value of the closed-loop eigenvalue as a function of H, for each R.

With the increase of H, the absolute value of the eigenvalue of $A - BK_{RH}$ decreases. So with the increase of H, if the eigenvalue tends towards inside the unit circle, the system is stabilizing. The absolute value of the eigenvalue of $A - BK_{RH}$ increases with R, so the increase in R brings the system closer to the unstable region. By (11), R is the weight for minimizing the energy of $u(t)$, which means that as R grows, the system will cause $u(t)$ to approach 0, for any t. Thus restricting the input $u(t)$, the system will be closer to becoming unstable. For $R >> Q$, the system will give more weight to minimizing the energy of $u(t)$ and less to following the reference.

For the same values of R and H, the absolute value of the eigenvalue of $A - BK_{RH}$ is higher for Plant 1. In Plant 1, for $R = \{0.01;\ 0.1;\ 1\}$ the system is stable for any H, but for $R = \{10;\ 100\}$ there is a constraint for the values of the horizon that guarantee stability: for $R = 10$, the system is stable if $H \geq 2$, and for $R = 100$, the system is stable for $H \geq 8$. In Plant 2, for all the values of R studied the system is stable for any H.

2.4

As verified in 2.2, with the increase of H, K_{RH} increases, converging to K_{LQ}. For a lower R value, higher the gains K_{LQ} and K_{RH}, and K_{RH} reaches K_{LQ} for lower values of H (quickly). The convergence in Plant 2 is quicker than in Plant 1. Additionally, for the same values of R, K_{LQ} is higher for Plant 1

and for the same values of R and H, K_{RH} is also higher for Plant 1. The prediction horizon must be large enough in order that K_{RH} is sufficiently close to K_{LQ} that stabilizes the closed-loop.

As demonstrated in 2.3, with the increase of H, the absolute value of the eigenvalue of $A - BK_{RH}$ decreases. The system is stable if condition (14) is verified, i.e., if the eigenvalues of $A - BK_{RH}$ are all inside the unit circle. For the same values of R and H, the absolute value of the eigenvalue of $A - BK_{RH}$ is higher for Plant 1.

R is the weight for minimizing the energy of $u(t)$, which means that as R decreases, the system will give more weight to $y(t)$ following the reference. When R decreases, tending towards 0, the gains K_{LQ} and K_{RH} increase, and the absolute value of the eigenvalue of $A - BK_{RH}$ decreases, thus making convergence from K_{RH} to K_{LQ} faster (in function of H).

In Plant 1, for smaller values of R, as $\{0.01;\ 0.1;\ 1\}$, the system is stable for any H. However, for higher values of R, as $\{10;\ 100\}$, the system is stable only for H above a certain threshold. On the other hand, in Plant 2, for the values of R studied the system is stable for any H.

Consequently, when enlarging H, the initial control move approximates the one obtained with an infinite horizon, which is stabilizing: H must be large enough to ensure the stability of the closed-loop.

Increasing H is advantageous up to values that guarantee the stability of the system and until K_{RH} converges to K_{LQ} ($K_{RH} \approx K_{LQ}$). From this point on, increasing H will maintain the system stable and will not bring an increase in K_{RH} since it has already converged to K_{LQ} and will remain approximately constant, so it will only be increasing computational load unnecessarily.

2.5

In sections 2.1-2.4 we already studied the open-loop stable 1^{st} order plant in (6). As described in section 2.4, for the values of R studied, Plant 2 is always stable because its eigenvalue of $A - BK_{RH}$ is inside the unit circle, and also the convergence of K_{RH} to K_{LQ} is quicker for Plant 2 than for Plant 1.

In Plant 1 for higher values of R as $\{10, 100\}$, the system is stable only for H above a certain threshold. So, in this plant is more advantageous to use enlarged values of H to guarantee the stability of the system, whereas for Plant 2 whatever the value of H the system is always stable.

Additionally, increasing R decreases both gains K_{LQ} and K_{RH}, and the convergence of K_{RH} to K_{LQ} as a function of H becomes slower, being Plant 1 convergence of K_{RH} to K_{LQ} slower than for Plant 2. Aiming to achieve $K_{RH} \approx K_{LQ}$, it is necessary to increase H, especially for the slowest system to converge, which is Plant 1. Furthermore, for the same values of R and H, K_{RH} is higher for Plant 1, converging to K_{LQ}, which is also higher for Plant 1. So increasing H to stabilize the system for Plant 1 also allows obtaining higher gain K_{RH}.

For Plant 2 increasing H may not be as advantageous as for Plant 1, because Plant 2 already guarantees the stability of the system. Also, its K_{RH} is quicker to converge to K_{LQ}, and from H that guarantees convergence, increasing H unnecessarily increases computational load. Therefore we can conclude that for Plant 1 is it more advantageous to use enlarged values of H.

3 P3

The objective of this section is to configure and use the MPC software package MPCtools 1.0, a software package that embeds MPC algorithms, by varying and analyzing the controller parameters.

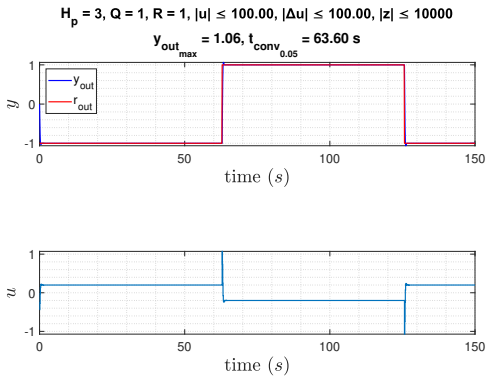
3.1

Considering the 1^{st} order open-loop unstable plant in (15), we simulate its control with MPC, illustrating the effect of constraints on the input, on the input rate of change and on the state, as well as the effects of the choice of the cost weights and the horizon. With the choice $H_w = 1$ and $H_u = H_p$, the cost function is quadratic, given by (16), and the time increments $\Delta u = u(k) - u(k-1)$ are penalized, instead of $u(k)$ which allows tracking non-zero references without error.

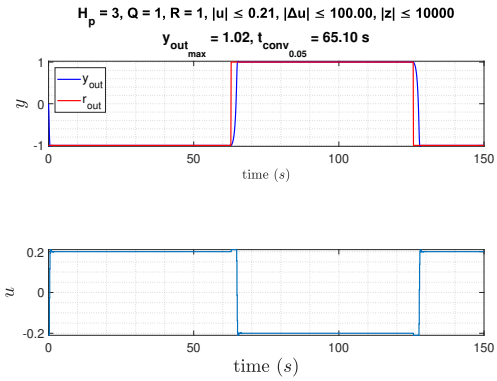
The considered default values were $|u| \leq 100$, $|\Delta u| \leq 100$, $|z| \leq 10000$, $R = 1$, $Q = 1$ and $H_p = 3$. The results obtained by varying the controller parameters and constraints (using the default in the remaining) are reported in Fig. 6 and Fig. 7, which also includes the maximum value obtained for y_{out}, and the time instant the system converges to $r_{out} = 1$ at a margin of 5% of the final value, $t_{conv_{0.05}}$.

$$x(k+1) = 1.2x(k) + u(k)\ , \quad y(k) = x(k) \tag{15}$$

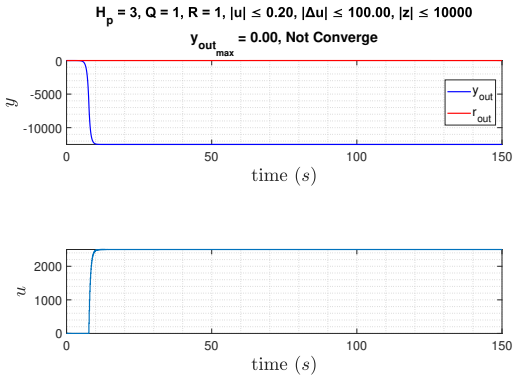
$$J(k) = \sum_{i=1}^{H_p} \|\hat{z}(k+i|k) - r(k+1|k)\|_Q^2 + \sum_{i=0}^{H_p-1} \|\Delta\hat{u}(k+i|k)\|_R^2\ , \text{ with } \|v\|_M^2 = v^T M v \tag{16}$$



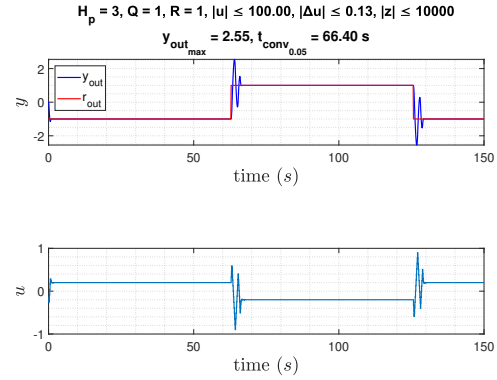
(a): Default.



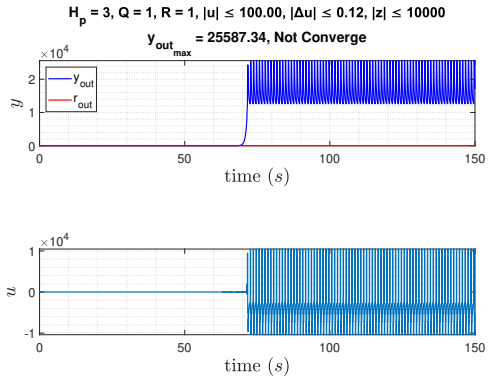
(b): $|u| \leq 0.21$.



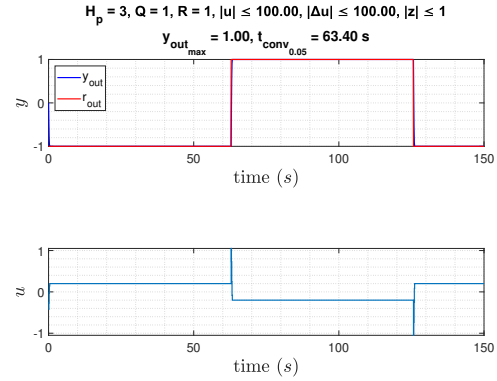
(c): $|u| \leq 0.2$.



(d): $|\Delta u| \leq 0.13$.



(e): $|\Delta u| \leq 0.12$.



(f): $|z| \leq 1$.

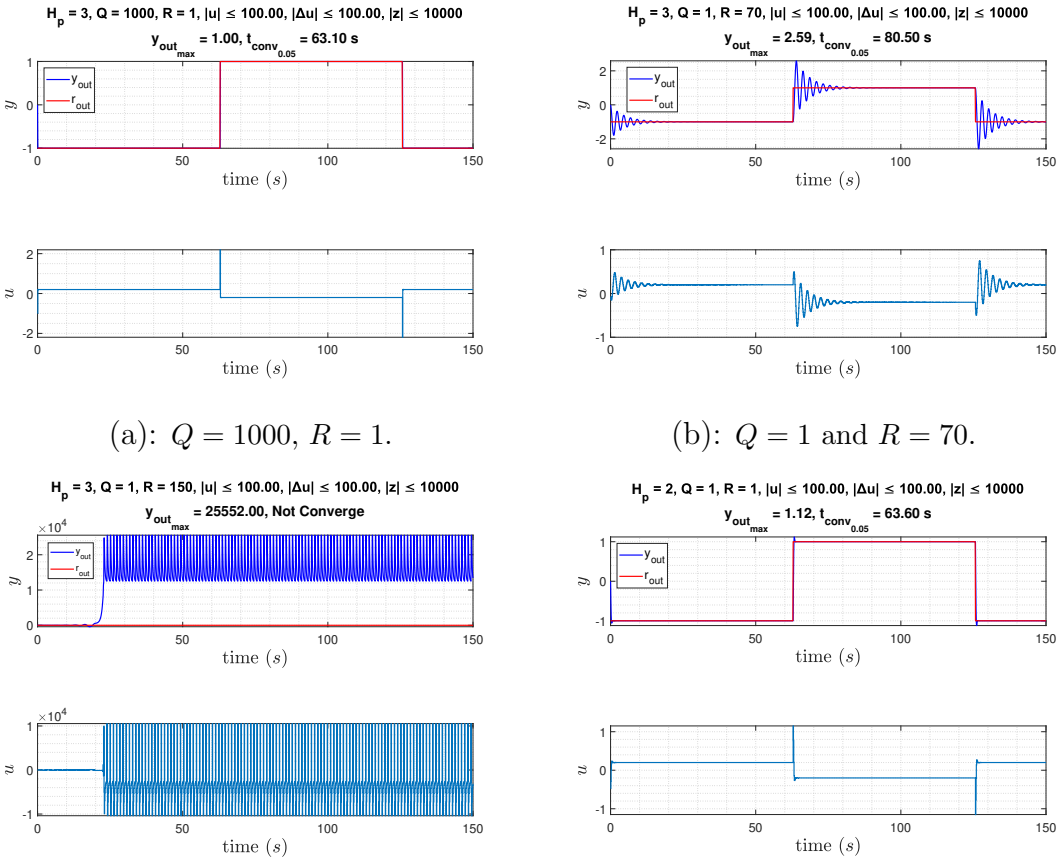
Figure 6: Control with MPC varying constraints.

With a constraint on the input, $|u(k)| \leq u_{lim}$, the stricter the constraint, in cases that y_{out} converges to r_{out}, this convergence is usually slower, as in Fig. 6(b). If u_{lim} is sufficiently small, the system may not converge to the reference (e.g., $u_{lim} = 0.2$ in Fig. 6(c)), because restricting the energy of the input, as we verify in 2.4 can lead the system to not converge to the reference. In this case, it is necessary to

increase H_p in order to guarantee convergence to the reference and to make it faster.

With a constraint on the input rate of change, $|\Delta u(k)| \leq \Delta u_{lim}$, the stricter the constraint, in cases that y_{out} converges to r_{out}, this convergence is usually slower, with higher overshoot and with most considerable oscillations (decrease of performance), as in Fig. 6(d). If Δu_{lim} is sufficiently small, the system may not converge to the reference (e.g., $\Delta u_{lim} = 0.12$ in Fig. 6(e)) because restricting the energy of Δu has a similar effect of increasing R due to (16). In this case, it is also necessary to increase H_p in order to guarantee the convergence to the reference and to increase performance.

By (15), a constraint on the state $x(k)$ is a constraint on the measured output $y(k)$ which in this case is equal to the controlled output $z(k)$, so $|z(k)| \leq z_{lim}$. The reference signal $|r(k)| \leq 1$ and with $z_{lim} = \infty$ we have $|z_{max}| \approx 1.06$, so we consider $1 \leq z_{lim} \leq 1.06$. In this range, as z_{lim} decreases y_{out} has lower overshoot (e.g., $z_{lim} = 1$ in Fig. 6(f)).



(a): $Q = 1000$, $R = 1$.

(b): $Q = 1$ and $R = 70$.

(c): $Q = 1$ and $R = 150$.

(d): $H_p = 2$.

Figure 7: Control with MPC varying parameters.

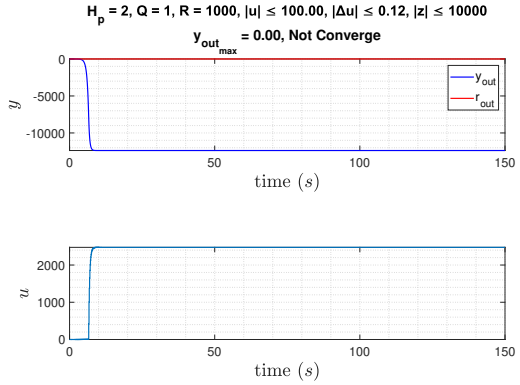
By (16), Q is the weight for minimizing the error between y_{out} and r_{out}, and R is the weight for minimizing the energy of Δu. In Fig. 7(a), $Q >> R$, so the system will give more weight to y_{out} following the reference which makes the convergence of y_{out} to r_{out} faster, with lower overshoot and smaller oscillations, as Q/R increases.

If $R >> Q$, the system gives more weight to minimizing the energy of Δu. A decrease in Q/R can make the system not converge to the reference, as in Fig. 7(c), this being a problem that can be solved by increasing H_p. In case the system converges to the reference, as Q/R decreases, the convergence of y_{out} to r_{out} is slower, with higher overshoot and with most considerable oscillations, as in Fig. 7(b).

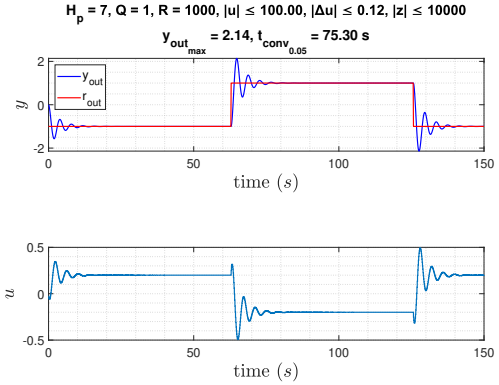
For $H_p \geq 2$, the convergence of y_{out} to r_{out} is fast, with low overshoot and small oscillations. However, the increase of H_p does not represent a considerable improvement in performance, which remains similar, and computational load increases (e.g. $H_p = 2$ in Fig. 7(d), and $H_p = 3$ in Fig. 6(a)).

3.2

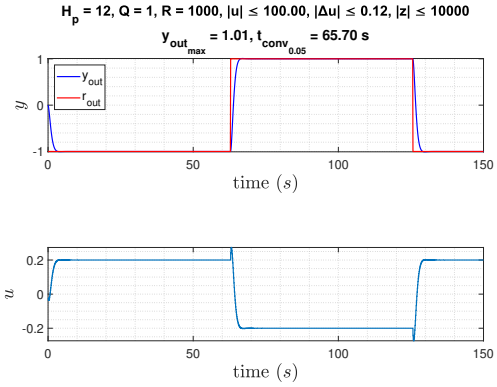
Considering a situation in which the weight R on the control variable is large in comparison to Q, $\big(Q = 1$ and $R = R/Q = 1000\big)$ and with a constraint on the input rate of change $|\Delta u(k)| \leq 0.12$, we study a situation that shows that enlarging the prediction horizon H_p results in an increase in performance. In Fig. 8, we report the results obtained for control with MPC by varying H_p, also using $|u| \leq 100$ and $|z| \leq 10000$. This situation was chosen in order to show that a longer horizon enables the controller to account for longer-term dynamics and transient behaviors in the system, and may allow the system to converge to the reference and can improve its performance. It also allows more flexibility to satisfy the constraints within the given actuation limits.



(a): $H_p = 2$.



(b): $H_p = 7$.



(c): $H_p = 12$.

Figure 8: Control with MPC for $|u| \leq 100$, $|\Delta u| \leq 0.12$, $|z| \leq 10000$, $R = 1000$ and $Q = 1$, varying H_p.

For $H_p < 7$ (e.g., $H_p = 2$), the output y_{out} can't follow the reference r_{out} (does not converge to it). For $H_p = 7$, y_{out} converges to r_{out} in a slow and oscillatory way. For $7 \leq H_p \leq 12$ (e.g., $H_p = 12$), by increasing H_p, the convergence of y_{out} to r_{out} becomes faster, with lower overshoot and smaller oscillations (increase on performance). However, for $H_p > 12$ the increase of H_p does not represent a considerable improvement in performance, which remains similar, and the computational load increases.

4 Conclusion

In this work, we explored the basic concepts of constrained optimization, RH control, and predictive control with the purpose of configuring and using a software package to analyze variations in the controller parameters and their effects.

References

[1] Distributed Predictive Control and Estimation (ECPD) Slides - Slides of Theoretical Classes 2022/2023, 2nd Semester (MEEC), by João Miranda Lemos;

[2] ECPD 2022/2023, Laboratory Work, by João Miranda Lemos.