Instituto Superior Técnico

Bachelor's Degree in Aerospace Engineering

Distributed Predictive Control and Estimation

Professor Pedro Batista

Laboratory Work - Report 1

Group 7

95807 - João Peixoto - joao.p.martins.peixoto@tecnico.ulisboa.pt
95855 - Tomás Nunes - tomas.trindade.nunes@tecnico.ulisboa.pt
96375 - Filipe Valquaresma - filipevalquaresma@tecnico.ulisboa.pt
96420 - Leonardo Eitner - leonardo.eitner@tecnico.ulisboa.pt

2021/2022
P4

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.

1 P1 - Basics on Constrained Optimization

1.1 Computing the minimum of a Rosenbrock function

The Rosenbrock function given is

$$f(\mathbf{x}) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2 \tag{1}$$

Using MATLAB's function `fminunc()` it is possible to compute the unconstrained minimum of a function, given that you provide an initial estimate, $\mathbf{x_0}$. In order to compute the constrained minimum, function `fmincon()` was used, which also required a constriction as an input. As stated in the lab handout,

$$\mathbf{x_0} = \begin{bmatrix} -1 \\ 1 \end{bmatrix} \qquad\qquad x_1 \leq 0,5 \tag{2}$$

1.1.1 Result analysis

In order to make the visualisation of the minima easier, a 3D plot of the function in equation 1 was made, where it is possible to notice that there are various local minima (figure 1).

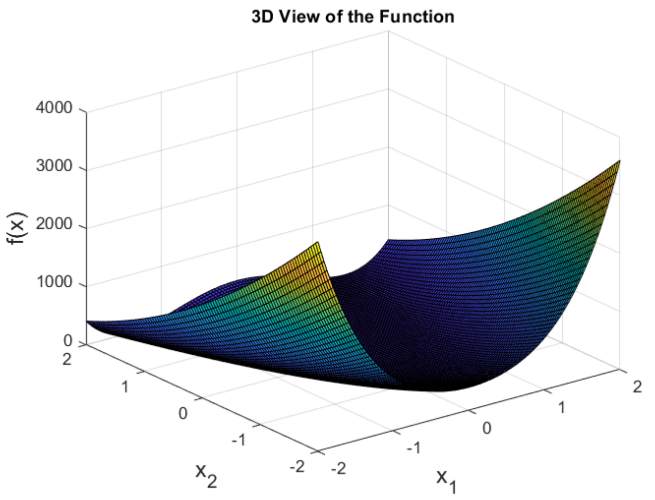

Figure 1: 3D View of the Function

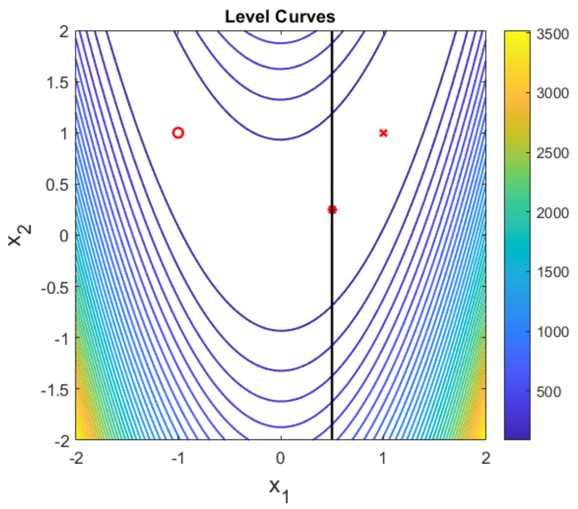

Figure 2: Level Curves

In figure 2, the level curves of the same function were plotted, along with the unconstrained minimum (red cross) and the constrained minimum (red asterisk), using the initial condition $\mathbf{x_0}$ (red circumference). The black line represents the constriction made, which limited x_1 to its left side.

From analysing figure 2 one can understand that `fmincon()` provides the closest minimum which obeys the restriction made.

1.2 Finding an estimate of the boundary of a minimum attraction basin

The function given is

$$f(\mathbf{x}) = x_1^4 - 10x_1^2 + x_2^4 - 10x_2^2 \tag{3}$$

When looking at figure 3 it becomes clear that this function has 4 absolute minima, and that the one to which the function `fminunc()` will converge will depend on the initial input $\mathbf{x_0}$.

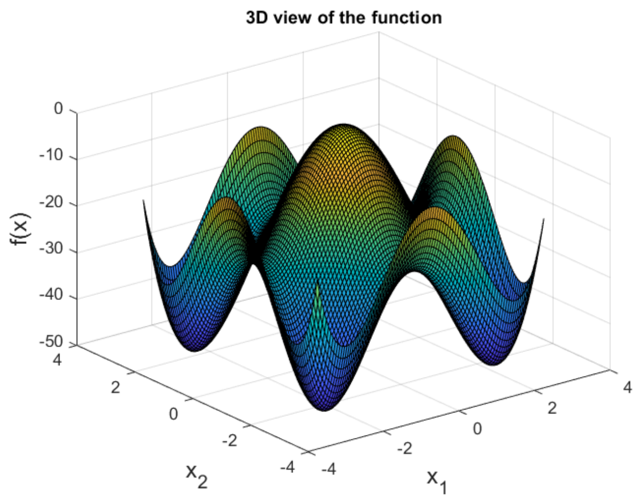

Figure 3: 3D View of the Function

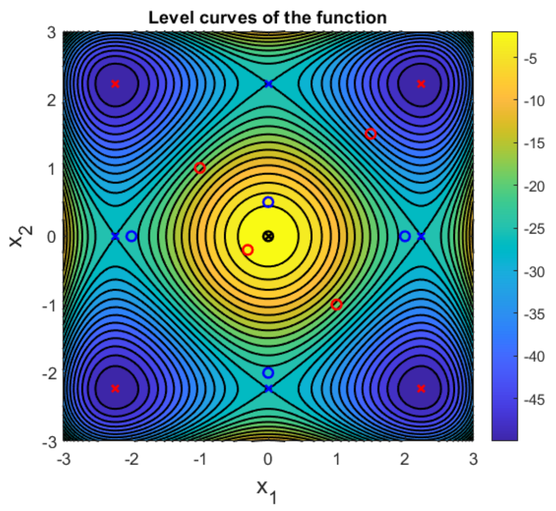

Figure 4: Level Curves

In figure 4 the level curves were plotted and the variation of the output of the function `fminunc()` with the initial value $\mathbf{x_0}$ was tested. With this test some limitations were found, as the function didn't always converge to a minimum. The function's output is represented with a cross and the initial value $\mathbf{x_0}$ is represented with a circle. Some of the values that converged to an actual minimum, which comprise the large majority of cases, are shown in red. Some limitations, which are likely associated with the computational method, were found and represented in different colours: blue for points which converged to saddle points, and black for those which converged to a maximum. These exceptions happen when the initial value chosen is exactly between either two or four minima, in which cases they converge to a saddle point or to a maximum, respectively.

Figure 5 shows the attraction basin for the minimum located approximately at $(x_1, x_2) = (2.236, 2.236)$. The bounded region was determined by "shooting" several starting estimates ($\mathbf{x_0}$) from the known minimum and increasing the angle by small constant increments. In order to speed up the process, a variable step method was chosen, thus reducing the amount of computations required. This results in an irregularly shaped bounded area. This convergence basin is a relatively conservative estimate, meaning that it is assumed that there are no false positives.

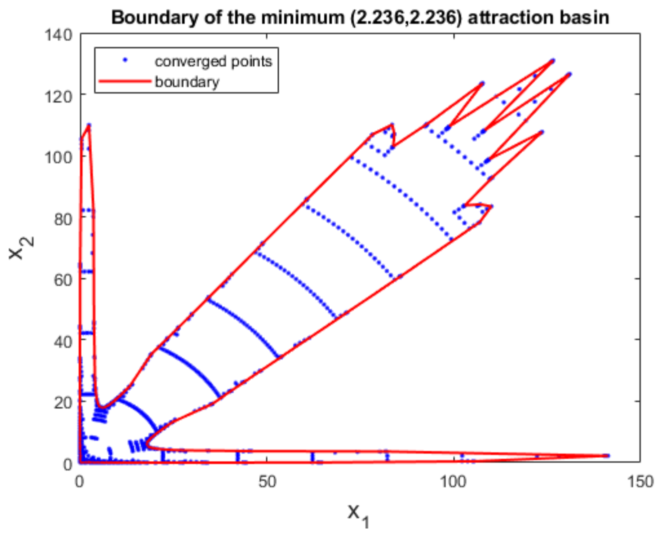

Figure 5: Attraction basin around a known minimum.

2 P2 - Basics on Receding Horizon Control

For this exercise, the following open-loop unstable first order discrete plant was considered:

$$x(t+1) = 1,2 \cdot x(t) + u(t) \tag{4}$$

2.1 Computing the optimal LQ state feedback gain

In order to compute the infinite horizon optimal feedback gain (K_{LQ}), MATLAB function `dlqr()` was used, which solves, in order to the positive definite $n \times n$ matrix S, the discrete-time algebraic Riccati equation (ARE), and then computes its optimal LQ gain, given that the matrices A, B, R and Q are provided.

The weights of the state $x(t)$ and the input $u(t)$ in the cost function are given by the weight matrices Q and R, respectively. As long as these vary equally, the cost function's response won't change and, therefore, Q was kept constant and only the changes in K_{LQ} associated with changes in R were analysed.

Table 1: Variation of K_{LQ} with R, with constant Q.

R	0,1	1	10	100	1000	10000
K_{LQ}	1,1026	0,7935	0,4882	0,3844	0,3685	0,3669

From looking at table 1 one can conclude that the gain K_{LQ} decreases with R, however, its value eventually stabilises at around 0,37. Increasing R translates to a reduction in the actuation or input $u(t)$, which naturally will lead to a slower response from the system. This relationship is implied whenever the effect of a varying R on a system's variable is analysed from here on forth.

2.2 Finite horizon, receding horizon control

Considering now the case of a finite horizon, rather than an infinite one, we can define the cost function J_{RH} as a function where R is the positive scalar weight.

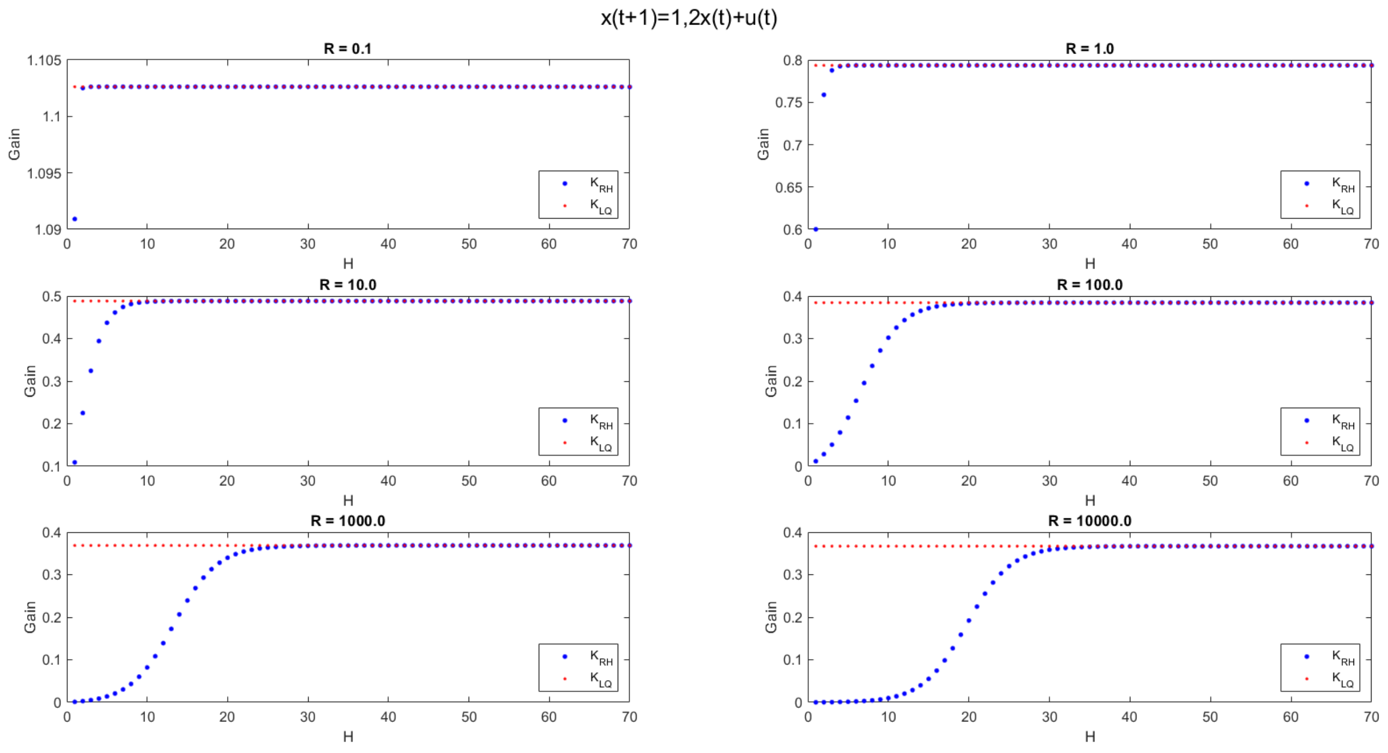

Figure 6: Comparison between Infinite and Finite Horizon optimal gains and respective convergence.

As can be seen in figure 6, for small R (or rather, a small R/Q ratio, as Q is kept constant), the receding horizon gain K_{RH} converges quickly (with a small horizon) to the infinite horizon gain. As R increases, it is necessary to increase the horizon in order to achieve convergence.

2.3 Eigenvalues and their relationship with the stability boundary

The eigenvalues were computed using the same MATLAB functions as the ones used for computing the gain. For a plant to be stable, its eigenvalues must be located within the unit circle ($|\lambda| < 1$).

Table 2: Variation of $|\lambda_{LQ}|$ with R

R	0,1	1	10	100	1000	10000
$\lvert\lambda_{LQ}\rvert$	0,09738	0,40650	0,71180	0,81560	0,83150	0,83310

From table 1, the absolute value of the eigenvalue $|\lambda_{LQ}|$ increases with R, converging to approximately 0,83. This means that the lower the value of R, the faster the system will be, as expected.

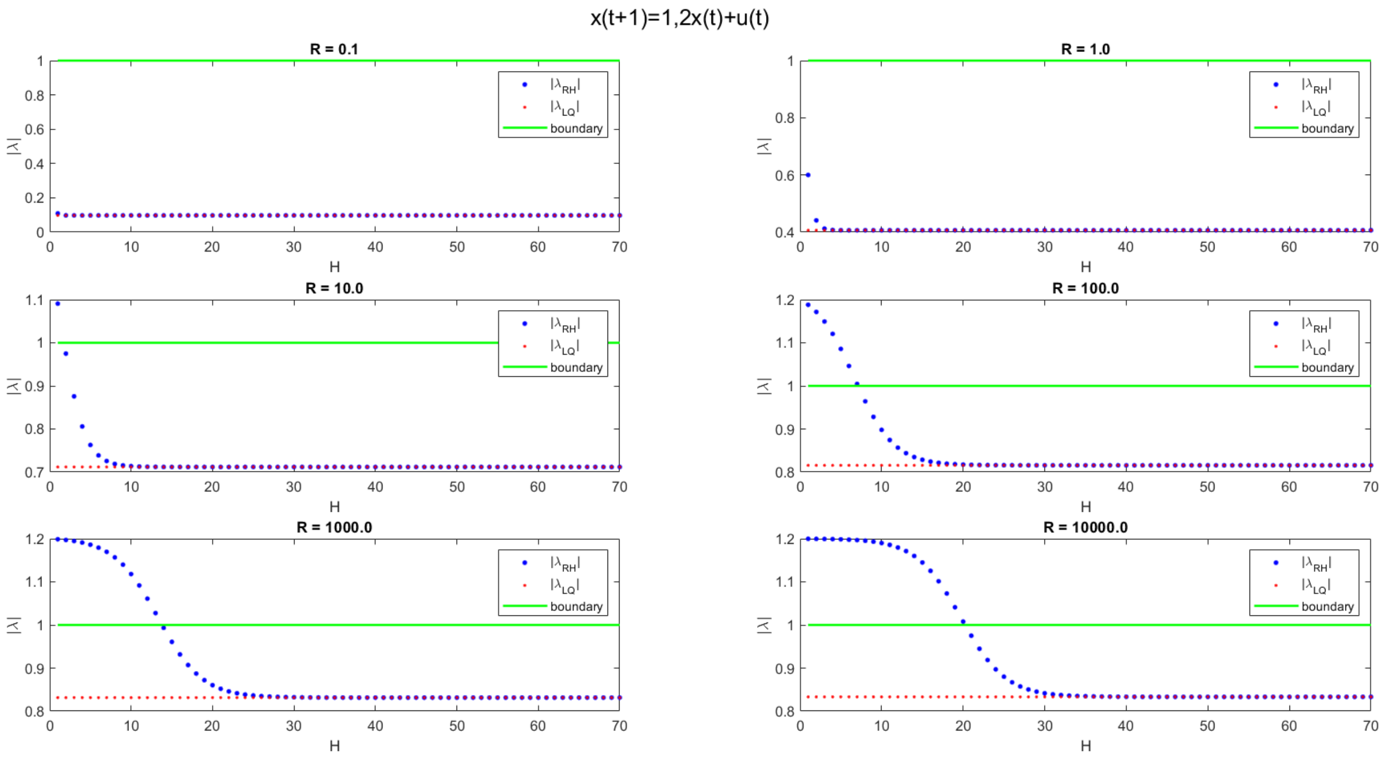

Figure 7: Variation of the module of the eigenvalues $|\lambda_{RH}|$ with R

For small values of R, $|\lambda_{RH}|$ remains smaller than 1, *i.e.* inside the stability region, but starting from $R = 10$, there are values outside of this region. Again, as R increases, so does the required horizon in order to, first, have the eigenvalue located inside the stability region, and second, converge to the LQ eigenvalue.

2.4 Advantages of enlarging the horizon H

As can be seen in the previous section, a large horizon ensures the system will be stable ($|\lambda| < 1$). This is especially important in cases where R is very large, as while for small values of R a small horizon is enough to asymptotically approach the ideal LQ gain and eigenvalue, large values require a bigger horizon to achieve the same result.

2.5 Comparison with a stable plant

In this subsection, a different plant was also considered. While the plant for subsection 2 was unstable, this subsection deals with the study of a stable plant:

$$x(t+1) = 0,8x(t) + u(t) \tag{5}$$

The same plots shown before are presented below for this case.

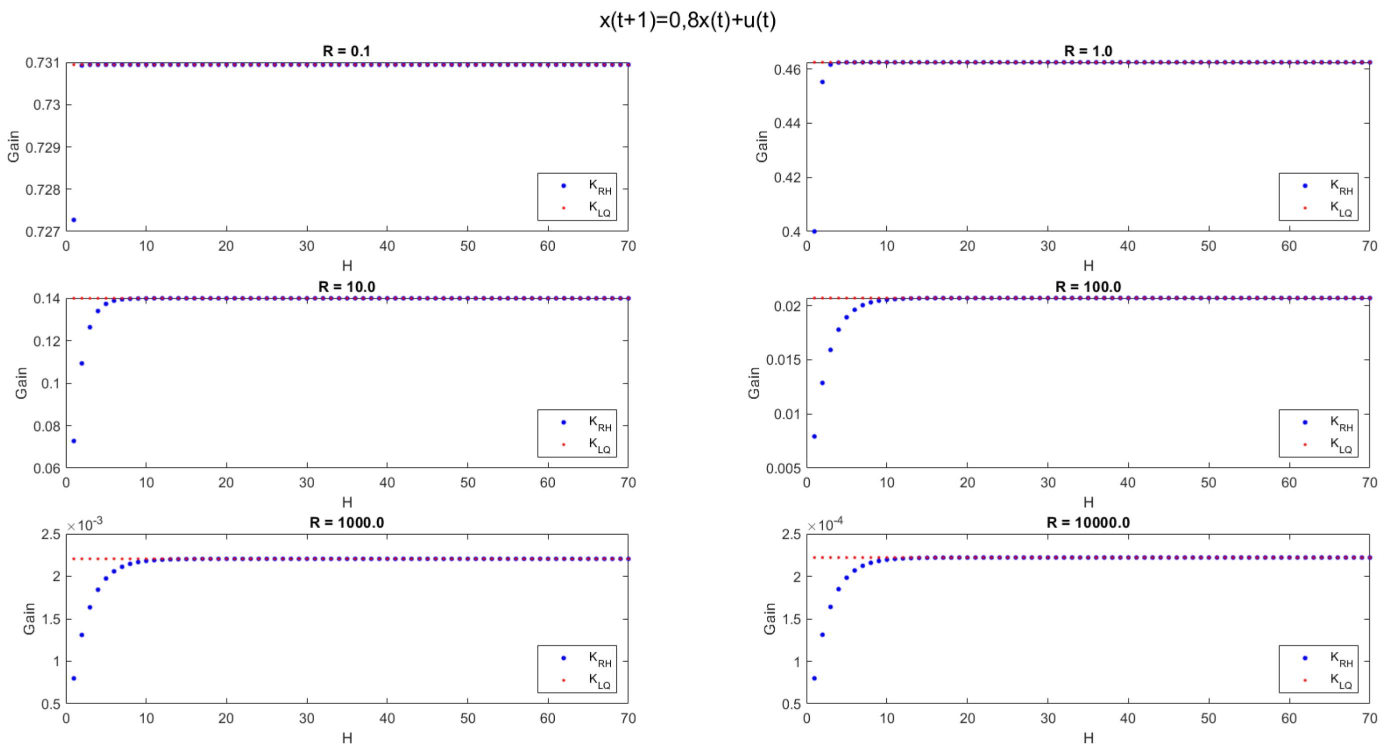

Figure 8: Variation of receding horizon gain K_{RH} with R.

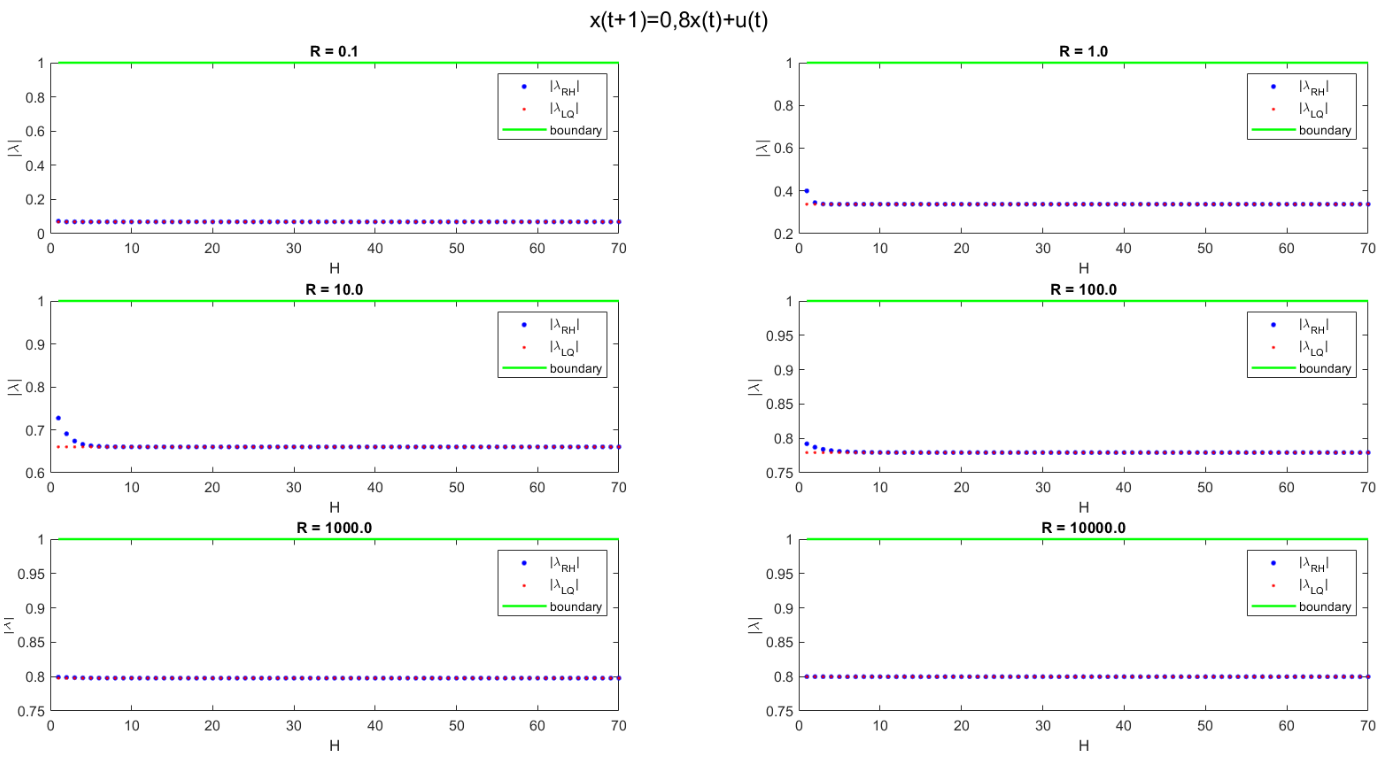

Figure 9: Variation of the module of the eigenvalues $|\lambda_{RH}|$ with R

When compared with the unstable plant, both the receding horizon gain and the eigenvalue converge much more quickly to the infinite horizon gain and eigenvalue, respectively. Furthermore, the absolute value of the infinite horizon eigenvalues are slightly smaller, meaning that the system will be faster than the unstable one. In the case of the receding horizon eigenvalue, unlike with the unstable plant, it is never bigger than 1, remaining in the stability region, as expected. Looking at the results for the stable and unstable plants, it is clear that a large horizon is advantageous in an unstable plant so as to have the eigenvalue inside the stability region and achieve convergence to both the optimal LQ gain and eigenvalue.

3 P3 - Basics on Predictive Control

3.1 Stabilising an unstable plant using MPC

Similarly to what was done previously, we begin by considering

$$x(k+1) = 1,2x(k) + u(k); \quad y(k) = x(k) \tag{6}$$

as the unstable first-order discrete-time plant to stabilise, this time by using model predictive control, MPC, in figure 10, with varying constraints, so as to illustrate their effects on the outcome.

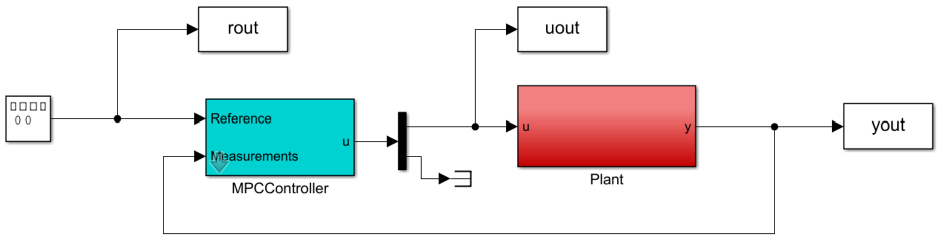

Figure 10: Simulink model of the MPC

Several factors influence the inputs, namely, the weights on the cost function and the size of the control horizon (H). In the plots, the effect of the different parameters is illustrated. Each plot represents the effect of a given parameter by keeping the other parameters at their default value. Table 3 shows the default values:

Table 3: Default values for the MPC parameters.

$\lvert u\rvert_{max}$	$\lvert\Delta u\rvert_{max}$	$\lvert z\rvert_{max}$	Q	R	H_p
100	100	10000	1000	10000	3

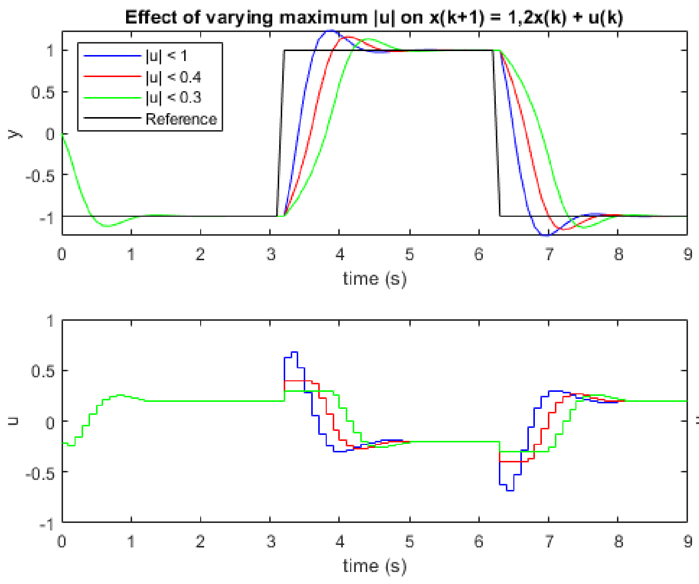

Figure 11: Effect of restrictions on the control variable

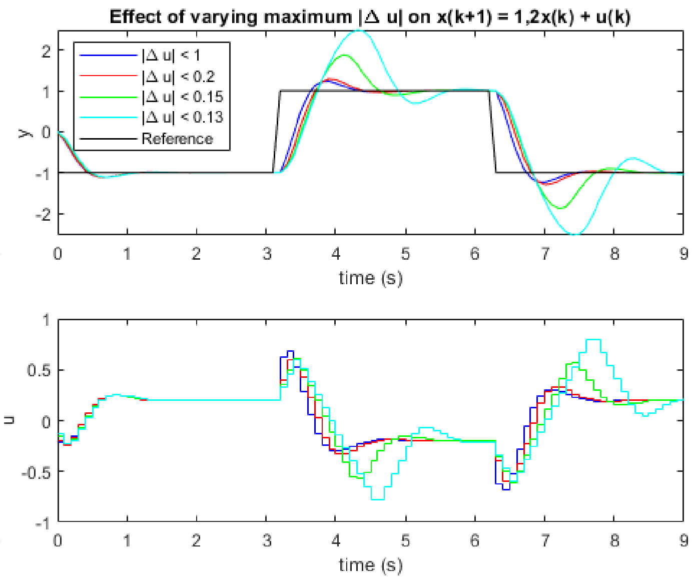

Figure 12: Effect of restrictions in the increments of the control variable

In the above figures, the effect of constraints on the control variables is shown: in figure 11, constraints are applied to the maximum (absolute) value of the control variable u, while in figure 12 the constraints are applied to the maximum increment of the control variable Δu. As the maximum value of the control variable increases, the response gets faster, but the overshoot is increased. For $u_{max} < 0.22$, the system is unstable. In the case of constraints on the increments, as the maximum increment decreases, the response is slower and the overshoot increases. For this parameter, the stable-unstable boundary is approximately 0.1.

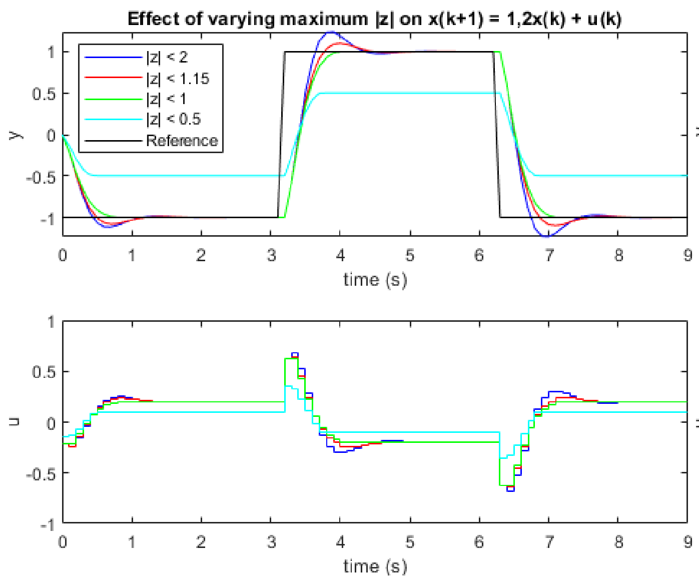

Figure 13: Effect of restrictions on the controlled variable

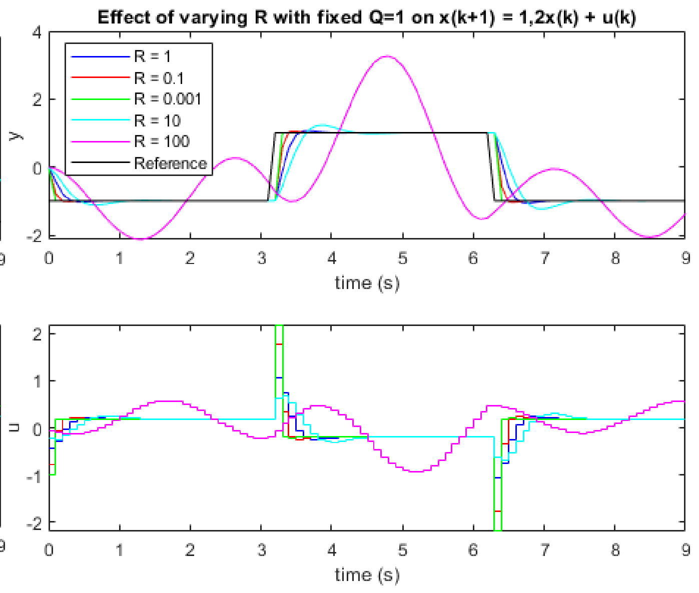

Figure 14: Effect of the weighting matrices from the cost function

Figure 13 plots the response for multiple values of the controlled variable z. As the constraints tighten ($|z|$ decreases), the overshoot decreases, but there are no noticeable differences in the settling time except when the controlled variable is limited to close to the reference, which is the case for the green curve. The cyan curve shows the response with a maximum allowed value for the controlled variable smaller than the reference. There are no noticeable oscillations, but, as expected, the response cannot match the reference.

In figure 14 the effect of the weighting matrices is plotted. In this case, Q was kept constant ($Q = 1$) and R varied. As R increases, so does the overshoot and settling time. For large R, the system is unstable. The largest case studied was $R = 100$, in which the system oscillates greatly and in fact is already unstable, visible if a larger time period is chosen.

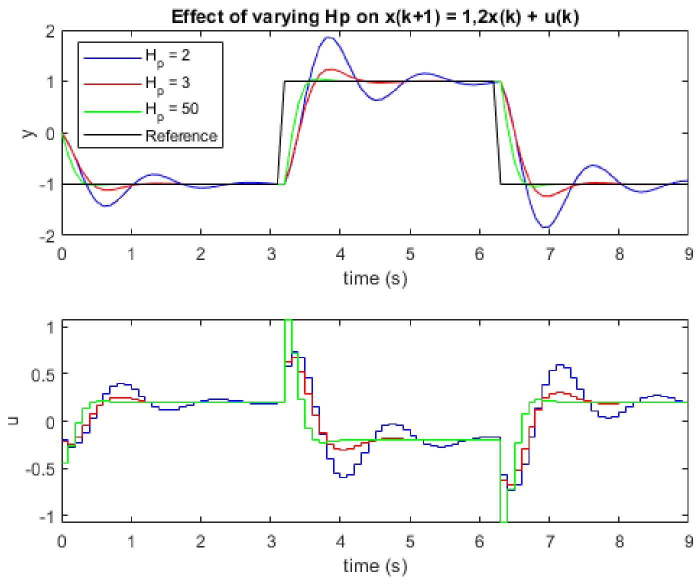

Figure 15: Effect of the horizon

Lastly, figure 15 portrays the response for different values of the horizon H_p. Both the overshoot and the settling time decrease as the horizon increases. A reasonably large horizon like $H_p = 50$ shows almost no oscillations. It is interesting to note that the MPC algorithm has problems with large horizons starting from approximately $H_p = 100$ which result in MATLAB returning errors.

3.2 Improving performance with a larger horizon

Now, moderate constrains were applied to the control amplitude with the same unstable plant as before, but with a higher R. The goal was to obtain a situation in which enlarging the horizon improves performance. Though the term "performance" is itself quite vague, for the purpose of this assignment it will be interpreted as a faster response and with less overshoot. In other words, the output should be as similar as possible in shape to the reference. The set of parameters defined is listed below:

Table 4: MPC parameters with moderate control constraints.

$\lvert u\rvert_{max}$	$\lvert\Delta u\rvert_{max}$	$\lvert z\rvert_{max}$	Q	R
0,45	0,13	1,5	1	100

Figure 16 plots the response of the system for multiple values of the horizon H_p against an arbitrary reference curve.

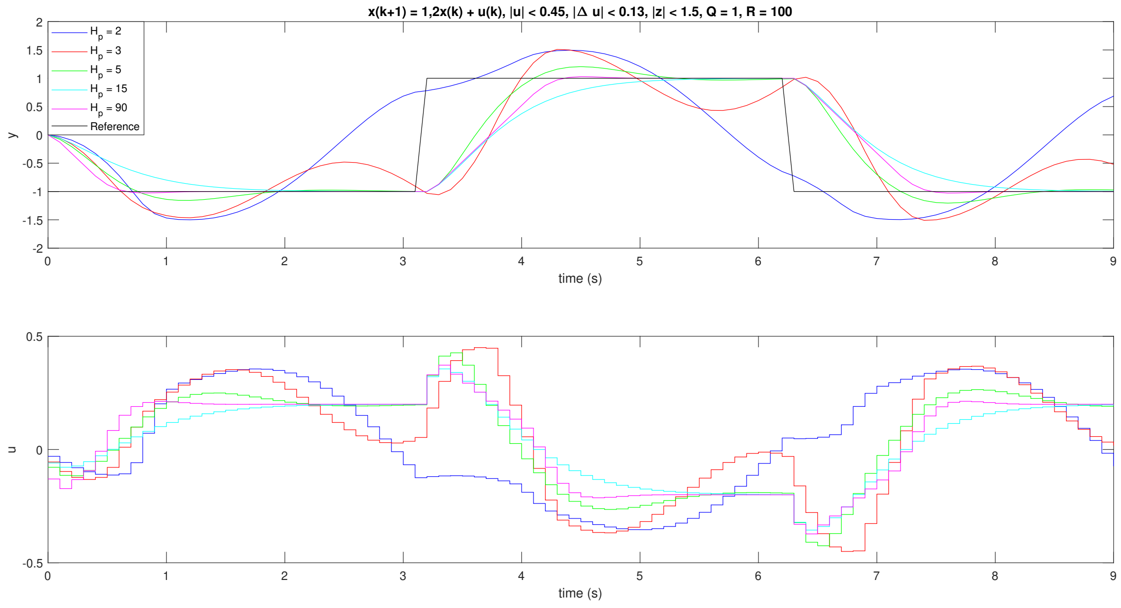

Figure 16: Effect of the horizon in the performance of the plant

For very small horizons (2 and 3), the output roughly follows the reference, but with large oscillations and overshoots. As the horizon increases, so does the accuracy of the output, reducing the oscillations and overshoot and approaching the change in the reference value more quickly. As per the objective stated in the beginning of this subsection, enlarging the horizon improves the performance of this system.