Uma imagem com Tipo de letra, logótipo, Gráficos, captura de ecrã

Descrição gerada automaticamente

Instituto Superior Técnico

Distributed Predictive Control and Estimation

MEEC

Laboratory Report

**Group: 18**

**Authors:**

Alexandre Leal (103098)alexandre.leal@tecnico.ulsiboa.pt

Diogo Sampaio (103068) diogo.sampaio@tecnico.ulisboa.pt

Francisco Tavares (103402)

Marta Valente (103574)

[francisco.carreira.tavares@tecnico.ulisboa.pt](mailto:francisco.carreira.tavares@tecnico.ulisboa.pt)

[marta.valente@tecnico.ulisboa.pt](mailto:marta.valente@tecnico.ulisboa.pt)

2nd Semester – 4th Quarter – 2024/2025

*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.*

## **P1 – Basic on Constrained Optimization**

For the vector , the Rosenbrock function is defined as:

(1)

As illustrated in *Figure 1*, the Rosenbrock function exhibits a narrow and curved valley, with a single global minimum at:

(2)

We will compute a cost function (both with and without constraints) in MATLAB, aiming to identify its minimum and visualise the results using clear and informative graphical outputs.

**Unconstrained Minimum**

Since the function is non-negative, any point at which must correspond to a global minimiser. By setting each squared term in to zero, we obtain:

(3)

This is the only stationary point, which can be verified by evaluating the gradient:

(4)

Subsequently, the quasi-Newton method was applied using MATLAB’s fminunc function, with an initial guess of:

(5)

The optimisation converged to the point:

(6)

Which closely approximates the theoretical minimum, confirming the effectiveness of the method.

**Constrained Minimum**

Afterwards, we impose a constraint on the search space:

(7)

As there are no stationary points within the feasible region defined by this constraint, the constrained minimiser must lie on the boundary . Applying the method of Lagrange multipliers with:

(8)

We solve:

(9)

To confirm this result, we applied MATLAB’s *fmincon* function with an initial guess of ), which returned:

(10)

Once again, the solution closely matches the theoretical value, confirming the validity of the analytical approach.

**Graphical Interpretation**

For a better understanding of the constraint effect, we generated these three figures:

Uma imagem com Saturação de cores, captura de ecrã, Gráficos

Os conteúdos gerados por IA poderão estar incorretos.Uma imagem com texto, Saturação de cores, captura de ecrã, arco-íris

Os conteúdos gerados por IA poderão estar incorretos.

Figure 1: Optimisation Points and Constraint. Figure 2: Heatmap and Contours.

*Figure 1* presents a logarithmic heatmap of the Rosenbrock function overlaid with white contour lines. This plot displays only the function shape without including any optimisation points or annotations. In contrast, *Figure 2* shows the initial point (o), the unconstrained solution (x) and the constrained minimum (\*), along with the shaded infeasible region defined by and its corresponding dashed boundary.

Finally, *Figure 3* shows a 3D surface plot of the Rosenbrock function, highlighting the curved valley that leads to the global minimum.

Uma imagem com diagrama, file, Gráfico, Saturação de cores

Os conteúdos gerados por IA poderão estar incorretos.

Figure 3: 3D Surface Plot.

## **P2 – Basic on Receding Horizon Control**

In this section, we analyse how optimal state feedback gains evolve under infinite- and finite-horizon formulations. For an unstable and stable open-loop plant:

(11)

(12)

The goal is to understand how the control horizon and input penalisation influence system behaviour and stability.

## **P2.1 – LQ State Feedback Gain**

We begin by computing the infinite-horizon Linear Quadratic gain ​, obtained via MATLAB’s *dlqr* function, for the unstable plant. The associated cost function is:

(13)

For the LQ gain obtained was . As expected, smaller values of led to more aggressive control strategies, resulting in larger gains , while higher values of produced more conservative gains. This trade-off illustrates the balance between tracking performance and energy expenditure in optimal control design. These infinite-horizon gains serve as reference values when evaluating the performance of the finite-horizon controller.

## **P2.2 – Optimal RH Gain**

We then evaluated the optimal gains for predictive controllers with a finite horizon , using the associated cost function:

(14)

This formulation allows us to observe how the finite-horizon controller approaches the infinite-horizon solution asincreases. As expected, the gains converge to the gain ​, with convergence occurring more rapidly for smaller values of . When , the gain remains constant for all , since control effort is not penalised.

Uma imagem com texto, file, Gráfico, diagrama

Os conteúdos gerados por IA poderão estar incorretos.

Figure 4: Optimal feedback gains for the unstable system, for various values of R.

*Figure 4* shows how, for an unstable system (), the gains increase with horizon length and approach the dashed reference values . This demonstrates the horizon’s role in improving control aggressiveness and reducing tracking error.

## **P2.3 – Closed-Loop Eigenvalue**

To assess closed-loop stability, we analysed the eigenvalue . A system is considered stable if the magnitude of the eigenvalue satisfies . *Figure 5* and *Figure 7* show the evolution of for the unstable and stable systems, respectively.

Uma imagem com file, Gráfico, texto, diagrama

Os conteúdos gerados por IA poderão estar incorretos.

Figure 5: Absolute eigenvalues for the unstable system, plotted against horizon H.

In the unstable case, *Figure 5*reveals that small horizons and high fail to stabilise the system (). Increasing gradually brings below one, with faster improvement for lower , which confirms the theoretical prediction that longer horizons are critical to stabilise unstable dynamics.

## **P2.4 – Different Horizon values**

Enlarging the prediction horizon in receding horizon control leads to several observable effects when compared to the infinite-horizon LQ gain .

As shown in *Figure 4*, which plots for the unstable system, the RH gain increases with and asymptotically approaches the corresponding gain. This convergence is faster for smaller values of . When , the gain is already equal to , and does not depend on . However, for , the effect of increasing is significant: it improves performance by better approximating the infinite-horizon optimal control. In *Figure 5*, we observe how the eigenvalues decrease with , crossing below 1 (i.e., entering the stable region) only after a sufficiently long horizon is reached — especially when is large. This highlights that in unstable systems, short horizons may fail to stabilize the system, and enlarging is crucial for ensuring stability.

## **P2.5 – Open-loop stable plant**

Finally, we repeated the study for a stable open-loop plant (Eq. (12)). As anticipated, the system remained stable across all and the gains were considerably lower than those required for the unstable case. The eigenvalues were consistently inside the unit circle and showed little sensitivity to horizon variation, confirming the system's intrinsic stability.

Uma imagem com texto, file, captura de ecrã, Gráfico

Os conteúdos gerados por IA poderão estar incorretos.

Figure 6: Optimal receding horizon gains for the stable system, for various values of R, over different values of H.

*Figure 6* shows finite-horizon gains for a stable system with , across the selected values of . For , the gain remains constant since the control effort is not penalised. For , the gain increases with and converges towards the gain (dashed lines). The convergence is faster for smaller , indicating that a short horizon is often sufficient to achieve near-optimal behaviour in stable systems.

Uma imagem com texto, captura de ecrã, file, Gráfico

Os conteúdos gerados por IA poderão estar incorretos.

Figure 7: Absolute eigenvalues for the stable system, plotted against horizon H.

*Figure 7* demonstrates that for the stable case (), all eigenvalues remain well within the stability boundary, regardless of – This indicates that short horizons are sufficient.

In this exercise, we fixed the state weighting matrix to to maintain focus on the effects of varying the input penalization and prediction horizon. We selected a representative range of , capturing behaviors from fully aggressive control () to highly conservative strategies (). These values allow for a clear comparison of control effort versus performance trade-offs. The horizon was varied from 1 to 15, which proved sufficient to observe the convergence of the receding horizon gain to the infinite-horizon optimal gain , as well as the stabilization effects on the closed-loop eigenvalue. This parameter range successfully revealed the contrasting impact of horizon length on stable versus unstable systems, demonstrating that large horizons are critical for stabilizing unstable plants, while shorter horizons suffice in the stable case. These choices ensure the results are both illustrative and representative of practical control scenarios.

## **P3 – Model Identification**

To implement Model Predictive Control (MPC), we require a dynamic model of the plant. While first-principles modelling based on heat transfer laws is possible for the TCLab system, we will use data-driven linear system identification, which is more scalable and general. This approach assumes no prior knowledge of the internal dynamics and relies only on input-output data.

We aim to obtain a discrete-time, linear, time-invariant model valid near a steady-state operating point , where:

(15)

We define deviations from equilibrium:

(16)

and assume the system can be approximated by the incremental model:

(17)

where is a Gaussian disturbance. This model is used for prediction, observer design, and controller synthesis.

In the first experiment (*Figure 8*), the heater input was set to 25% to drive the system to an equilibrium around 40 °C. Small step variations of ±5–10% were then applied, allowing sufficient settling time. This structure captures the incremental response and supports the use of a SISO model, as Temperature 2 remained largely unaffected. The resulting dataset was used to estimate the model using MATLAB’s *ssest* function.

A graph of a graph of a graph

AI-generated content may be incorrect.

Figure 8: Open-loop experiment used for model validation.

For validation, we performed a second open-loop experiment using faster and larger amplitude input changes (see *Figure 9*). Here, the system was excited with larger and faster input changes without full settling. Despite this, Temperature 1 exhibited smooth and consistent dynamics, remaining within the operating range. The identified model was simulated under this new input, and the predicted output matched the measured temperature with low mean squared error, confirming the model’s ability to generalise.

A close-up of a graph

AI-generated content may be incorrect.

Figure 9: Open-loop experiment used for system identification. Heater input (top), temperature sensor readings (bottom).

During both open-loop experiments, the raw temperature traces exhibited significantly more scatter than what was expected. We repeated the tests several times under different environmental conditions (e.g. varying ambient drafts, board positioning, and room temperature) but observed similarly noisy measurements. We believe this noise stems from hardware limitations of the TCLab sensors and heater PWM. Given time constraints, we proceeded with the best dataset available.

We selected a state dimension , which minimised the MSE and provided a good fit on both datasets without overfitting. However, higher-order models would lead to overfitting due the high amount of noise present in our measurements. This model will be used in the following stages for MPC and Kalman filter implementation.

Adicionar se calhar uma tabela ou um paragrafo a mostrar os diferentes valores de n e correspondentes MSE obtidos

## **P4 – MPC and Kalman Filter Design**

In Part 4 we pass from model identification to predictive control and estimation of the single-heater TCLab system. Starting from the incremental, discrete-time state-space model

Adicionar aqui qualquer coisa?

## **P4.2 – Unconstrained MPC: Horizon & Weight Tuning**

After embedding our dense‐form MPC in the TCLab simulation script—calling ***mpc\_solve*** at each 2 s step to compute and apply only the first Δu—we next examined how the prediction horizon affects both performance and computation time. Figure X shows that solver runtime rises from well under 0.05 % of the sampling period at to over 0.6 % at . Meanwhile, Figure X2 demonstrates that closed-loop rise-time improves significantly as increases from 10 to 20, but beyond the incremental benefit becomes negligible: the responses for and are virtually identical (their absolute difference is plotted in green). By choosing , we capture near–infinite-horizon behavior in only about 0.06 % of the 2 s interval, satisfying the rule-of-thumb that MPC computation should remain below 10 % of the sample time.

Figure X,X2,X3

With fixed at 50, we then varied the control weight over (Figure X3). A low (e.g.\ 0.02) produces very aggressive inputs—so aggressive that Δu can exceed 100 % if unconstrained—while a high (e.g.\ 1) yields sluggish regulation with minimal control effort. We settled on because it strikes a balance: rise-time of roughly 210 s with peak Δu near 50 %, all while keeping solver runtimes well under 1 % of the sampling period. These choices of and form a reliable, real-time MPC baseline for the constrained and estimator-augmented extensions that follow.

## **P4.3 – Actuator Saturation via Input Constraints**

To enforce the heater’s physical limits, we include simple lower‐ and upper‐bound constraints on the **incremental** input within our QP. Since the actual power is

Requiring is equivalent to

for every step in the horizon. In practice, these bounds replace the empty input‐constraint arguments in ***quadprog***, so that the optimization can only choose inside at each future time.

Because our nominal weight never pushes the unconstrained controller beyond these limits, we reduce (leaving fixed) to provoke saturation. Figure X4 overlays the unconstrained and bounded MPC responses: in blue you see the unconstrained input exceeding 100 % for several steps; in orange the same controller strictly caps at 100 % (and 0 % on the lower end). The corresponding temperature trajectories (bottom plots) reveal only a small performance loss under the bounds, while the yellow trace highlights the absolute output difference between the two. This simple addition of bounds ensures all commanded heater powers are physically realizable for the remainder of our experiments.

Figure X4

## **P4.4 – Feed-forward Reference Tracking & Bias Analysis**

To trach a +5 °C step, we first compute the unique steady‐state increment that satisfies,

So that and . Defining

we run the exact same , MPC on δ-variables (with bounds shifted by ). As Figure X5 (blue) shows, the output error δ converges to zero and reaches exactly in closed loop.

Next, we simulate a 10 % increase in the model constant ​ (mimicking ambient‐temperature mismatch). The orange trace in Figure 7 now reveals a clear steady state offset in both δ and δ. Although the MPC predictions inside its horizon still drive δ to 0 (see Figure X6), the real plant settles with bias because the disturbance induced by the altered ​ is not included in the prediction model.

In short, the feed-forward change of variables perfectly enforces zero error in the **nominal** model, but any unmodeled constant disturbance (here from ​) produces an irreducible offset—exactly the motivation for introducing a disturbance-augmented Kalman filter in the next section.

Figures X5 AND X6

## **P4.5 – Safety Limit: Hard vs. Soft Output Constraints**

To enforce a maximum temperature of 55 °C, we augment our dense‐form MPC so that the predicted output vector

satisfies for each step . In practice this means adding the inequality with and directly into the call to ***quadprog***.

We then test this “hard” safety constraint by commanding a reference of 60 °C (i.e.\ ) under the same disturbance and noise used elsewhere. For the first few steps the solver finds a feasible sequence of control increments, but once the noisy temperature briefly exceeds 55 °C the quadratic program becomes infeasible, and MATLAB returns ***exitflag = –2*** (“no feasible point found”). In other words, there is simply no sequence of future Δ that can both respect the model equations and drive the plant below 55 °C once it has risen above that bound.

To recover feasibility, we soften the cap by introducing nonnegative slack variables at each predicted step and penalizing in the cost. The constraint becomes

so that any unavoidable limit violation is “paid for” by ​. With a suitably large penalty α, the solver now always finds a solution (exitflag = 1) while keeping ​ as close to zero as possible.

Figure X7 shows the hard‐constrained simulation, where the MPC eventually fails under noise, and Figure X8 shows the soft‐constrained result—here the temperature is clamped at or just above 55 °C, and the bottom‐right subplot plots the slack ​ over time, with spikes exactly when the cap is reached. This soft‐constraint strategy guarantees a real‐time, feasible MPC that respects the safety requirement with minimal and transparent limit relaxations.

FIGURE X7 AND X8

## **P4.6 – Augmented-State Kalman Filter Design & Open-Loop Test**

...

## **P4.7 – Estimator-Based MPC: Closed-Loop Disturbance Compensation**

...

## **P5 – Application to the Real System**

...