Report of Programming Exercise

Model Predictive Control
Lecturer: Prof. Melanie Zeilinger, Dr. Andrea Carron
Group Members: Qi Ma, Yue Li

May 18, 2021

1 Define values of A^c , B^c and d^c

Rearrange the system dynamics into matrix form:

$$\begin{bmatrix} \dot{T}^c_{vc}(t) \\ \dot{T}^c_{F1}(t) \\ \dot{T}^c_{F2}(t) \end{bmatrix} = \begin{bmatrix} -\frac{1}{m_{vc}}(\alpha_{F1,vc}+\alpha_{F2,VC}+\alpha_{env,vc}) & \frac{1}{m_{vc}}\alpha_{F1,vc} & \frac{1}{m_{vc}}\alpha_{F2,vc} \\ \frac{1}{m_{F1}}\alpha_{vc,F1} & -\frac{1}{m_{F1}}(\alpha_{vc,F1}+\alpha_{F2,F1}) & \frac{1}{m_{F1}}\alpha_{F2,F1} \\ \frac{1}{m_{F2}}\alpha_{vc,F2} & \frac{1}{m_{F2}}\alpha_{F1,F2} & -\frac{1}{m_{F2}}(\alpha_{vc,F2}+\alpha_{F1,F2}) \end{bmatrix} \begin{bmatrix} T^c_{vc}(t) \\ T^c_{F1}(t) \\ T^c_{F2}(t) \end{bmatrix} + \begin{bmatrix} \frac{1}{m_{vc}}\beta_{11} & \frac{1}{m_{vc}}\beta_{12} & \frac{1}{m_{vc}}\beta_{13} \\ \frac{1}{m_{F1}}\beta_{21} & \frac{1}{m_{F1}}\beta_{22} & \frac{1}{m_{F1}}\beta_{23} \\ \frac{1}{m_{F2}}\beta_{31} & \frac{1}{m_{F2}}\beta_{32} & \frac{1}{m_{F2}}\beta_{33} \end{bmatrix} \begin{bmatrix} p^c_{vc}(t) \\ p^c_{F1}(t) \\ p^c_{F2}(t) \end{bmatrix} + \begin{bmatrix} \frac{1}{m_{vc}} & 0 & 0 \\ 0 & \frac{1}{m_{F1}} & 0 \\ 0 & 0 & \frac{1}{m_{F2}} \end{bmatrix} \begin{bmatrix} T_{env}\alpha_{env,vc}+d_{vc} \\ d_{F1} \\ d_{F2} \end{bmatrix}$$

Thus, we have:

$$A^c = \begin{bmatrix} -\frac{1}{m_{vc}}(\alpha_{F1,vc}+\alpha_{F2,VC}+\alpha_{env,vc}) & \frac{1}{m_{vc}}\alpha_{F1,vc} & \frac{1}{m_{vc}}\alpha_{F2,vc} \\ \frac{1}{m_{F1}}\alpha_{vc,F1} & -\frac{1}{m_{F1}}(\alpha_{vc,F1}+\alpha_{F2,F1}) & \frac{1}{m_{F1}}\alpha_{F2,F1} \\ \frac{1}{m_{F2}}\alpha_{vc,F2} & \frac{1}{m_{F2}}\alpha_{F1,F2} & -\frac{1}{m_{F2}}(\alpha_{vc,F2}+\alpha_{F1,F2}) \end{bmatrix}$$

$$B^c = \begin{bmatrix} \frac{1}{m_{vc}}\beta_{11} & \frac{1}{m_{vc}}\beta_{12} & \frac{1}{m_{vc}}\beta_{13} \\ \frac{1}{m_{F1}}\beta_{21} & \frac{1}{m_{F1}}\beta_{22} & \frac{1}{m_{F1}}\beta_{23} \\ \frac{1}{m_{F2}}\beta_{31} & \frac{1}{m_{F2}}\beta_{32} & \frac{1}{m_{F2}}\beta_{33} \end{bmatrix}$$

$$d^c = \begin{bmatrix} T_{env}\alpha_{env,vc}+d_{vc} \\ d_{F1} \\ d_{F2} \end{bmatrix}$$

By calculation:

$$A^c = \begin{bmatrix} -8.6310e-05 & 5.3571e-06 & 9.5238e-06 \\ 5.7692e-05 & -2.3333e-04 & 1.7564e-04 \\ 3.0769e-04 & 5.2692e-04 & -8.3462e-04 \end{bmatrix}$$

$$B^c = \begin{bmatrix} 1.9048e-07 & -1.1905e-08 & -1.1905e-08 \\ 2.5641e-07 & 2.0513e-06 & 1.2821e-07 \\ 7.6923e-07 & 1.1538e-06 & 6.9231e-06 \end{bmatrix}, d^c = \begin{bmatrix} 3900 \\ 0 \\ 0 \end{bmatrix}$$

2 Discretize the system

Use Euler's method to discretize the continuous system:

$$\dot{T}^c_{vc}(t) \approx \frac{T^c_{vc}(t+T_s) - T^c_{vc}(t)}{T_s}$$

The system dynamics then translate to:

$$\begin{bmatrix} T_{vc}(k+1) \\ T_{F1}(k+1) \\ T_{F2}(k+1) \end{bmatrix} =$$

$$\begin{bmatrix} -\frac{T_s}{m_{vc}}(\alpha_{F1,vc} + \alpha_{F2,VC} + \alpha_{env,vc}) + 1 & \frac{T_s}{m_{vc}}\alpha_{F1,vc} & \frac{1}{m_{vc}}\alpha_{F2,vc} \\ \frac{T_s}{m_{F1}}\alpha_{vc,F1} & -\frac{T_s}{m_{F1}}(\alpha_{vc,F1} + \alpha_{F2,F1}) & \frac{T_s}{m_{F1}}\alpha_{F2,F1} \\ \frac{T_s}{m_{F2}}\alpha_{vc,F2} & \frac{T_s}{m_{F2}}\alpha_{F1,F2} & -\frac{T_s}{m_{F2}}(\alpha_{vc,F2} + \alpha_{F1,F2}) \end{bmatrix} \begin{bmatrix} T_{vc}(k) \\ T_{F1}(k) \\ T_{F2}(k) \end{bmatrix}$$

$$+ T_s \begin{bmatrix} \frac{1}{m_{vc}}\beta_{11} & \frac{1}{m_{vc}}\beta_{12} & \frac{1}{m_{vc}}\beta_{13} \\ \frac{1}{m_{F1}}\beta_{21} & \frac{1}{m_{F1}}\beta_{22} & \frac{1}{m_{F1}}\beta_{23} \\ \frac{1}{m_{F2}}\beta_{31} & \frac{1}{m_{F2}}\beta_{32} & \frac{1}{m_{F2}}\beta_{33} \end{bmatrix} \begin{bmatrix} p_{vc}(k) \\ p_{F1}(k) \\ p_{F2}(k) \end{bmatrix}$$

$$+ T_s \begin{bmatrix} \frac{1}{m_{vc}} & 0 & 0 \\ 0 & \frac{1}{m_{F1}} & 0 \\ 0 & 0 & \frac{1}{m_{F2}} \end{bmatrix} \begin{bmatrix} T_{env}\alpha_{env,vc} + d_{vc} \\ d_{F1} \\ d_{F2} \end{bmatrix}$$

Thus, we have:

$$A = \begin{bmatrix} -\frac{T_s}{m_{vc}}(\alpha_{F1,vc} + \alpha_{F2,VC} + \alpha_{env,vc}) + 1 & \frac{T_s}{m_{vc}}\alpha_{F1,vc} & \frac{T_s}{m_{vc}}\alpha_{F2,vc} \\ \frac{T_s}{m_{F1}}\alpha_{vc,F1} & -\frac{T_s}{m_{F1}}(\alpha_{vc,F1} + \alpha_{F2,F1}) & \frac{T_s}{m_{F1}}\alpha_{F2,F1} \\ \frac{T_s}{m_{F2}}\alpha_{vc,F2} & \frac{T_s}{m_{F2}}\alpha_{F1,F2} & -\frac{T_s}{m_{F2}}(\alpha_{vc,F2} + \alpha_{F1,F2}) \end{bmatrix}$$

$$B = T_s \begin{bmatrix} \frac{1}{m_{vc}}\beta_{11} & \frac{1}{m_{vc}}\beta_{12} & \frac{1}{m_{vc}}\beta_{13} \\ \frac{1}{m_{F1}}\beta_{21} & \frac{1}{m_{F1}}\beta_{22} & \frac{1}{m_{F1}}\beta_{23} \\ \frac{1}{m_{F2}}\beta_{31} & \frac{1}{m_{F2}}\beta_{32} & \frac{1}{m_{F2}}\beta_{33} \end{bmatrix}$$

$$B_d = T_s \begin{bmatrix} \frac{1}{m_{vc}} & 0 & 0 \\ 0 & \frac{1}{m_{F1}} & 0 \\ 0 & 0 & \frac{1}{m_{F2}} \end{bmatrix}$$

3 Steady-state and delta-formulation

According to the description, the desired temperature:

$$T_{sp} = \begin{bmatrix} 25 \\ -42 \\ -18.5 \end{bmatrix}$$

The matrix form of the system dynamics is given by:

$$T(k+1) = AT(k) + Bp(k) + B_d d$$

Within steady-state:

$$\Delta T(k) = T(k) - T_{sp} = 0$$

Substitute above equation into system dynamics:

$$(I - A)T_{sp} = Bp_{sp} + B_d d$$

We have:

$$p_{sp} = B^{-1}[(I - A)T_{sp} - B_d d]$$

by calculation: $p_{sp} = \begin{bmatrix} 8237.3 \\ -4911.1 \\ -241.5 \end{bmatrix}$ and the delta-formulation:

$$\Delta T(k+1) = A\Delta T(k) + B\Delta p(k)]$$

4 State constraints

According to the task description, we have:

$$T_{min} = \begin{bmatrix} 22 \\ -45 \\ -19 \end{bmatrix}, T_{max} = \begin{bmatrix} 28 \\ -39 \\ -17 \end{bmatrix}, p_{min} = \begin{bmatrix} 0 \\ -7500 \\ -1500 \end{bmatrix}, p_{max} = \begin{bmatrix} 15000 \\ 0 \\ 0 \end{bmatrix}$$

$$\Delta T_{min} = T_{min} - T_{sp}, \quad \Delta T_{max} = T_{max} - T_{sp}$$
$$\Delta p_{min} = p_{min} - p_{sp}, \quad \Delta p_{max} = p_{max} - p_{sp}$$

Thus:

$$\Delta T_{min} = \begin{bmatrix} -3 \\ -3 \\ -0.5 \end{bmatrix}$$
$$\Delta T_{max} = \begin{bmatrix} 3 \\ 3 \\ 1.5 \end{bmatrix}$$
$$\Delta p_{min} = \begin{bmatrix} -8.2373e+03 \\ -2.5889e+03 \\ -1.2585e+03 \end{bmatrix}$$
$$\Delta p_{max} = \begin{bmatrix} 6.7627e+03 \\ 4.9111e+03 \\ 2.4146e+02 \end{bmatrix}$$

5 Simulate the system from initial condition

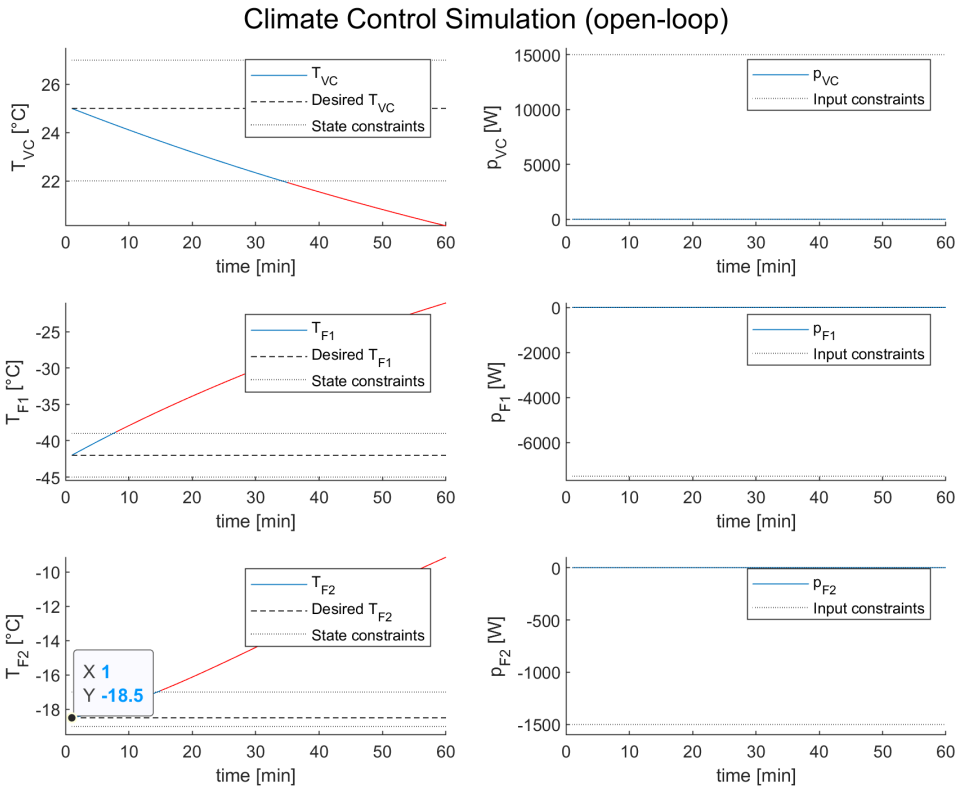

Figure 1: simulate_building(T0_1)

Without controller input, the three temperatures will gradually change to the same and violate state constraints.

6 Unconstrained Optimal Control with LQR controller

According to the task description, we have three subtasks namely:

1) LQR controller design.
2) LQR heurstic tunning.
3) Visualization and best Q chossing.

For subtask 1)Infinite Horizon LQR controller take the consideration of whole Horizon until the state converges. In short, it minimizes the infinite object function:

$$J_\infty(x(0) = \min_{u(.)} \sum_{i=0}^{\infty} (x_i^T Q x_i + u_i^T R u_i)$$

subject to:

$$x_{i+1} = Ax_i + Bu_i, i = 0, 1, 2, ...,$$
$$x_0 = x(0)$$

As with the Dynamic Programming approach, the optimal input of the form:

$$u^*(k) = -(B^T P_\infty B + R)^{-1} B^T P_\infty A x(k) := F_\infty x(k)$$

and the infinite-horizon cost-to-go is:

$$J_\infty(x(k) = x(k)^T P_\infty x(k)$$

In order to solve this, generalize from RDE, it satisfy the Algebraic Riccati equation (ARE):

$$P_\infty = A^T P_\infty A + Q - A^T P_\infty B(B^T P_\infty B + R)^{-1} B^T P_\infty A$$

By solving the ARE function we could getP_∞ and then we can get the control strategyF_∞

However, once we have the strategy we still need appropriate parameter so it meet our need, including:

2.1) minimizes the deviation from the steady-state after 15 min of closed-loop control

2.2) the total energy used for heating and cooling during 60 min of closed-loop simulation must not exceed 16 kWh.

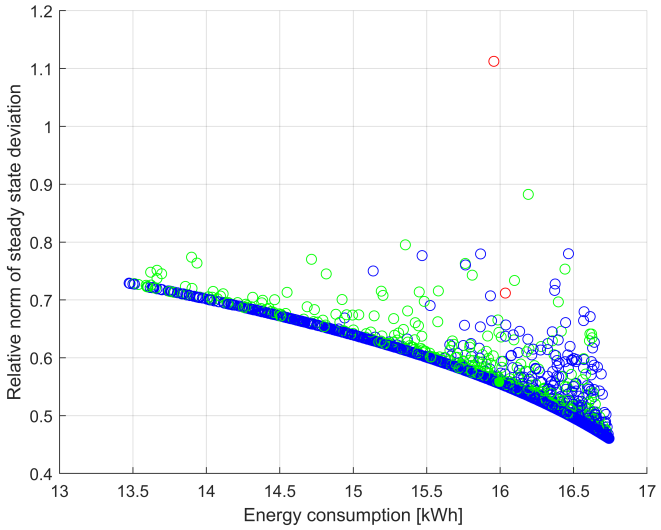

Figure 2: heuristic tuning(2500 iterations) consider input violation

Therefore, initialize the $R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ $Q = \begin{bmatrix} randi([1, 10e6]) & 0 & 0 \\ 0 & randi([1, 10e6]) & 0 \\ 0 & 0 & randi([1, 10e6]) \end{bmatrix}$ for each iterations and store all the deviation and total energy cost. For the parameters which will lead to constrain violation will be colored. In order to get a relatively good parameters We run the whole loop in 2500 and 5000 times.

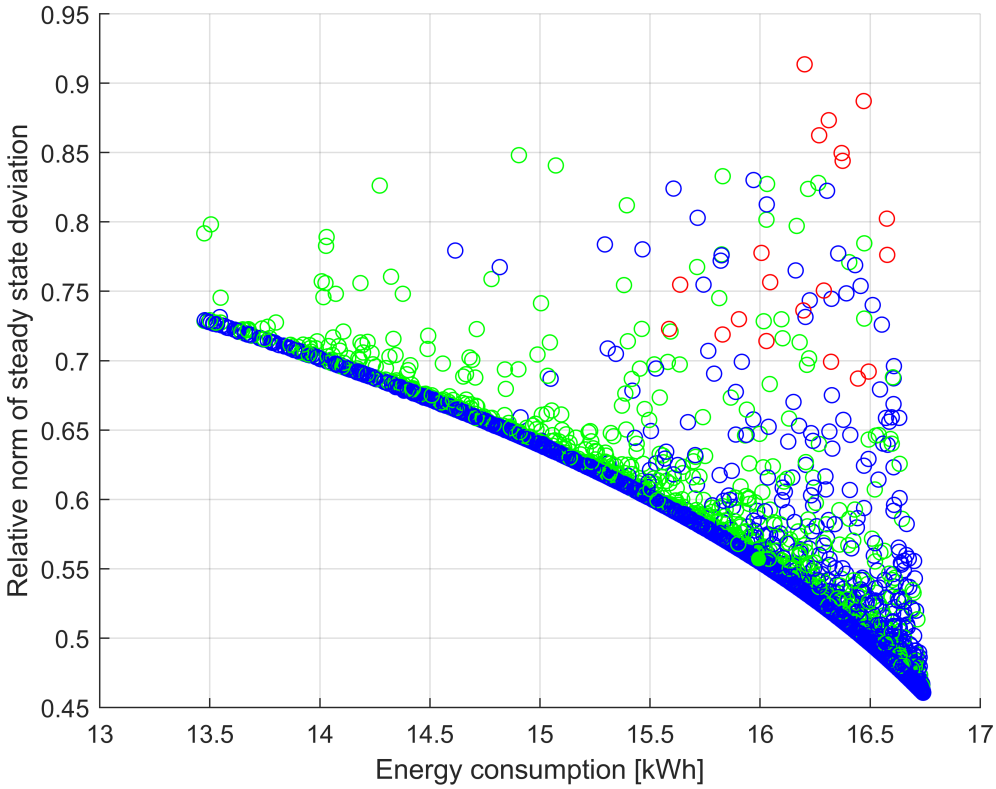

Figure 3: heuristic tuning(5000 iterations)

The controller result in state constrain violation colored with red while input constrain violation colored with blue. Green circle represent controller parameters which meet our need.
Figure 2 and Figure 3 shows two low-bound curve which is reasonable because for each energy consumption in one hour it exist a lowest deviation from steady-state.
By choosing the lowest deviation and total energy not exceed 16 kWh, the filled green circle is the resulting $Q_1 = \begin{bmatrix} 4089136 & 0 & 0 \\ 0 & 2136443 & 0 \\ 0 & 0 & 7215423 \end{bmatrix}$

After checking the moodle we find that in tuning Q the input constrain is not necessary so the result without input constrain is: $Q_2 = \begin{bmatrix} 4096366 & 0 & 0 \\ 0 & 6099988 & 0 \\ 0 & 0 & 7914287 \end{bmatrix}$
The difference between this two Q will lead to different performance in task 6 and task 7. Even the lowest deviation is almost the same in two cases but Q_2 will result input constrain violation at first to get the set point as soon as possible. In order to get a more reasonable result we choose Q_1 as the best Q for LQR controller.

7 Qualitative influence of Q and R

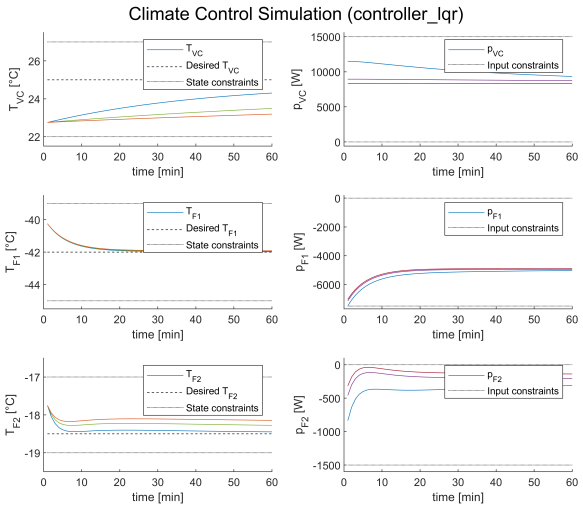

Figure 4: different diagonal element in R

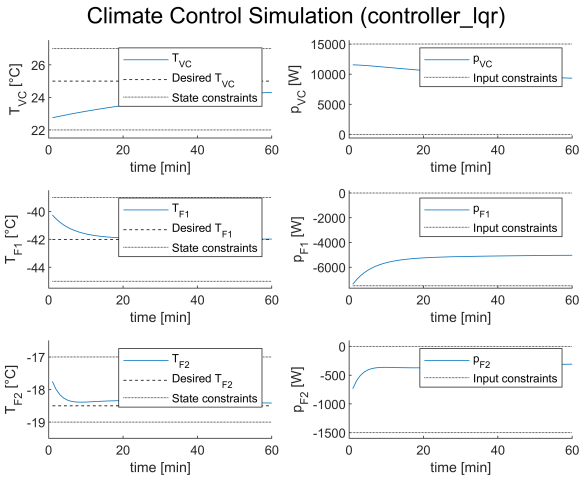

Figure 5: Closed-loop simulation plot of T^1_{init}

Since stage cost: $I(x_i, u_i) = x_i^T Q x_i + u_i^T R u_i$ When Q have one diagonal element increased, then it means this correspondent state has more weights in state costs so the system will try stabilize this state or make this state to origin as soon as possible.
Similar to Q, when R have a diagonal element increased it means the control cost to control this state become more expansive, so the LQR controller will decrease the "control force" of controlling the correspondent state. In order to analyze the influence of changes of the diagonal elements in R we draw the system with different R, namely $R_1 = diag([1, 1, 1])$ which is blue, $R_2 = diag([10, 1, 1])$ which is green, $R_1 = diag([100, 1, 1])$ which is red. Obviously the larger diagonal elements slow down the control speed.

8 Result of another starting states

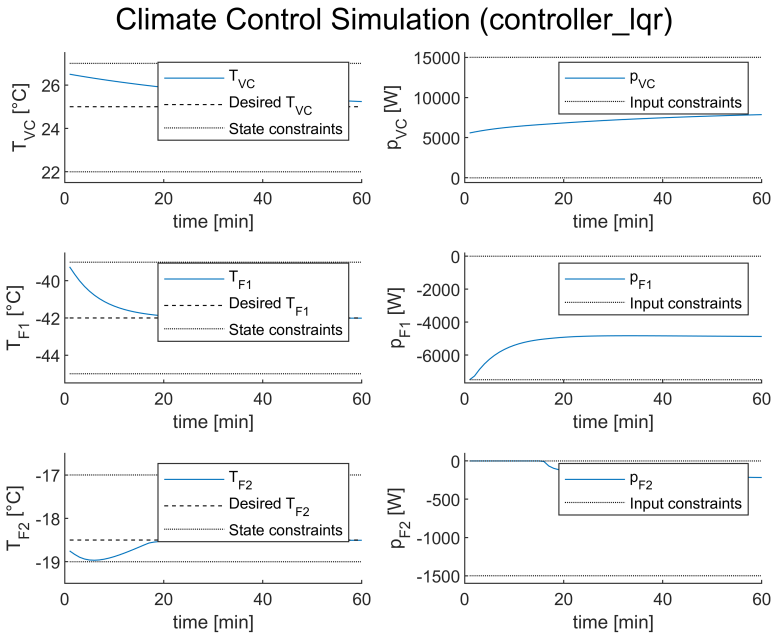

Figure 6: Closed-loop simulation plot of T^2_{init}

When we use the Q tuned by $T^{(}_{init}1)$ use in $T^{(}_{init}2)$ it occurs input constrain violation. This is reasonable because compared to former case the $deltaT(3)$ is negative, so the states of T_F2 will much easier to have state violation.

After tuning with $T^{(}_{init}2)$ we get Q like: $Q_1 = \begin{bmatrix} 4096366 & 0 & 0 \\ 0 & 6099988 & 0 \\ 0 & 0 & 7914287 \end{bmatrix}$

$Q_2 = \begin{bmatrix} 4096366 & 0 & 0 \\ 0 & 6099988 & 0 \\ 0 & 0 & 7914287 \end{bmatrix}$

9 Compute the invariant set of LQR controller

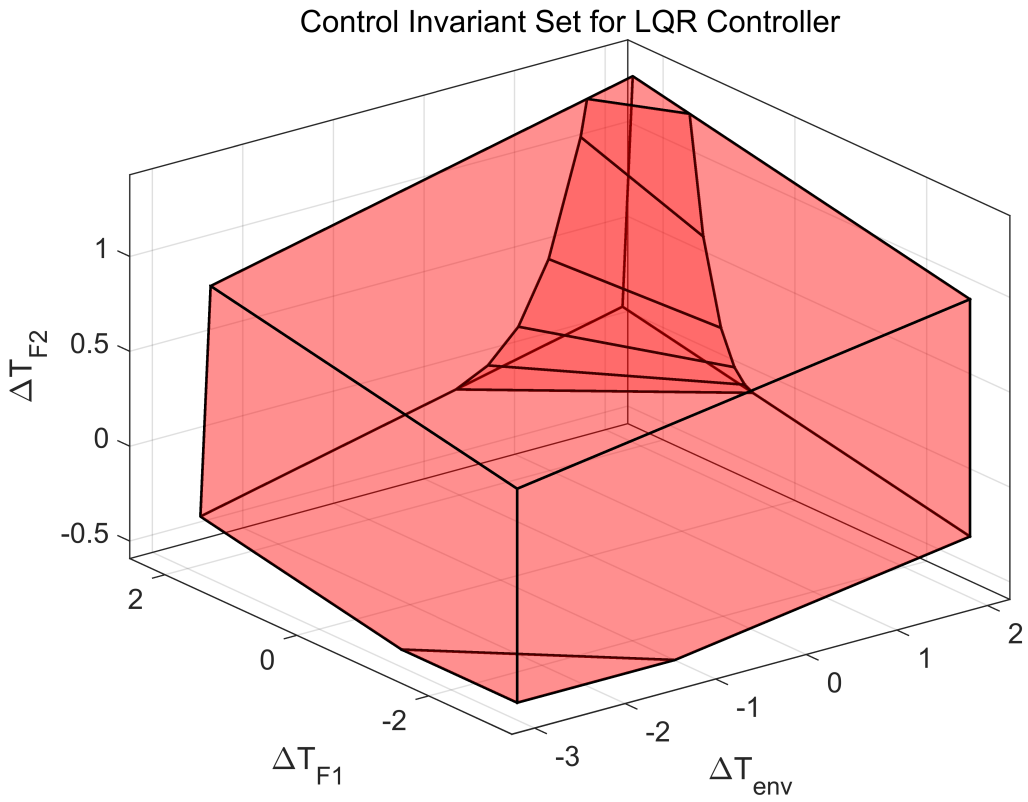

Figure 7: invariance set of LQR together with constraints

As shown in the figure below, the control invariant set of LQR controller is a subset of initial conditions. Computed by MPT toolbox, the invariance set of this specific LQR controller is defined by 15 inequality equations.

10 Compute the infinite horizon cost under the LQR control law

With the recursive approach, we obtain the infinite horizon cost function:

$$J(x(0)) = x(0)^T P_\infty x(0)$$

where P_∞ is the solution of the Algebraic Riccati equation:

$$P_\infty = A^T P_\infty A + Q - A^T P_\infty B(B^T P_\infty B + R)^{-1} B^T P_\infty A$$

11 From LQR to model predictive controller

Pass the Q and R to MPC controller and set terminal cost equal to the LQR infinite horizon cost, we can obtain the close-loop simulations of MPC.

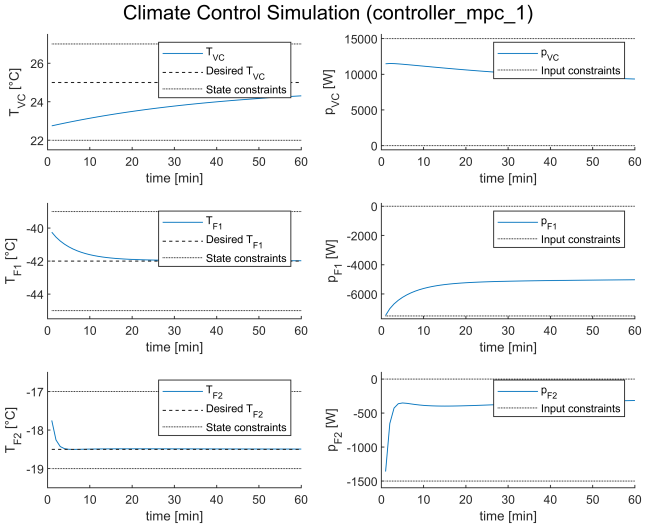

Figure 8: close-loop simulation from T0_1 by mpc1 in scen1

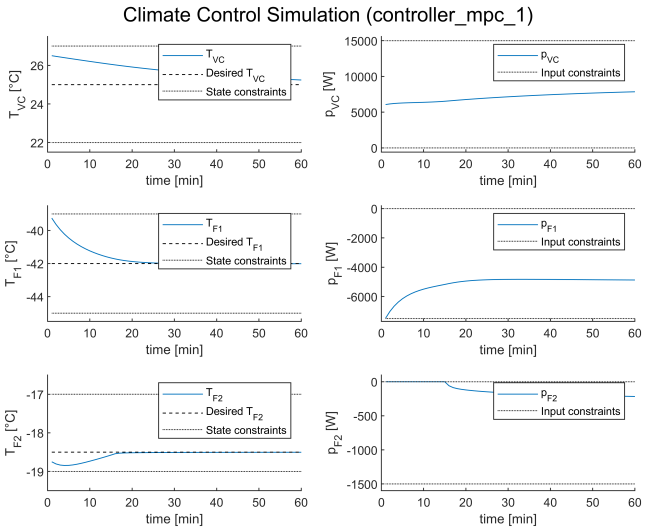

Figure 9: close-loop simulation from T0_2 by mpc1 in scen1

We compare the difference with results in task 7 and 8 by plotting the two simulations together.

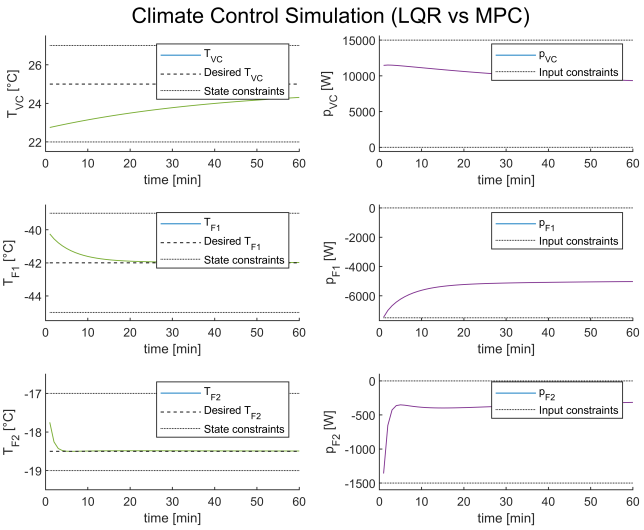

Figure 10: compare results starting from T0_1 by mpc1 and LQR

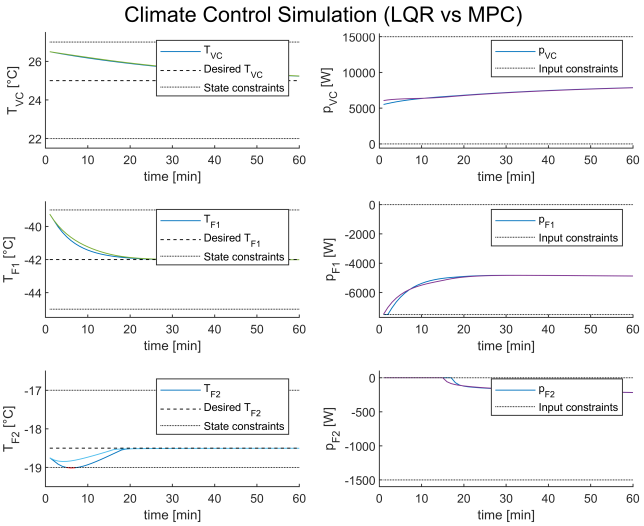

Figure 11: compare results starting from T0_2 by mpc1 and LQR

It can be seen that in the case of T0_1, the figures are fully overlapped, showing that the two controllers are equivalent. Since the MPC controller uses the same Q, R and the terminal cost, it is equivalent to the infinite horizon LQR controller. For T0_2, when using the original LQR controller, T_F2 will exceed the state constraints at beginning, but with MPC, it will not; because MPC will re-plan 30 steps for the current state at each time-step, which makes it more robust.

12 MPC with theoretical closed-loop guarantees

Setting the terminal constraint at zero is one of the methods to provide feasibility and stability guarantees for MPC. Using general notations, the cost function is given by:

$$J^{*}\left(x\left(k\right)\right)=\min_{U} l_{f}\left(x_{N}\right)+\sum_{i=0}^{N-1} l\left(x_{i},u_{i}\right)$$

Assuming the feasibility of initial condition $x(0)$, we can prove that the cost function $J^{*}\left(x\left(k\right)\right)$ is a Lyapunov function, thus the origin is an asymptotically stable equilibrium point for the resulting closed-loop system.

13 Implement the MPC with zero terminal constraint

With zero terminal constraint, the comptroller drives the system to the set point with off-set free. It is worth mentioning that under the initial conditions of $T0\_2$, the temperature T_{F2} reaches the state limit, but does not exceed.

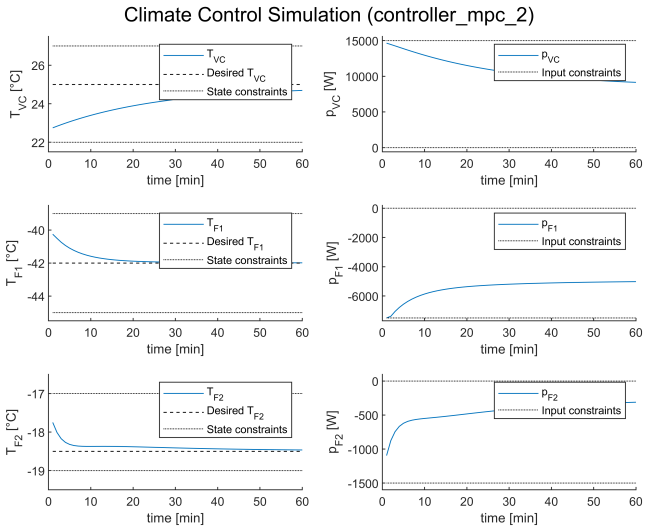

Figure 12: results by mpc2 starting from T0_1

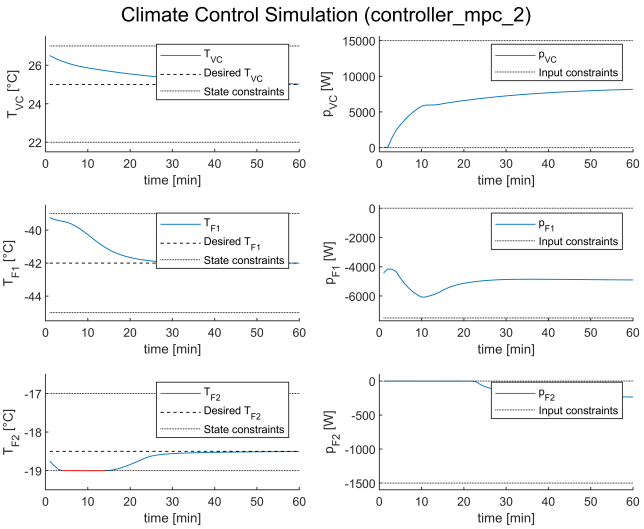

Figure 13: results by mpc2 starting from T0_2

14 MPC with different terminal constraint

Here we use the invariance set of the LQR controller as the terminal constraint, the results are shown by following figures.

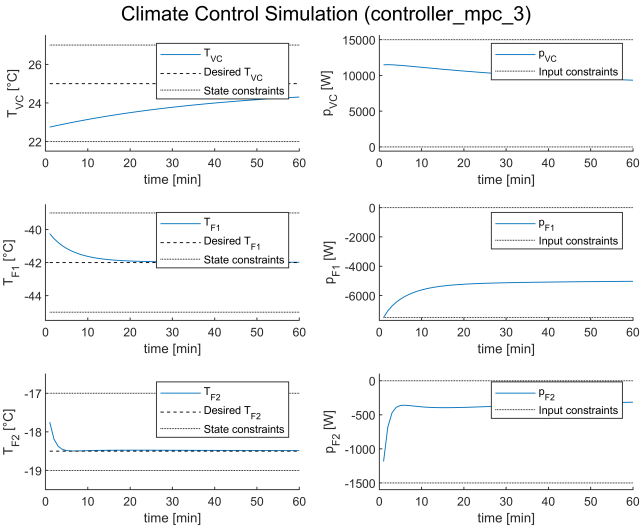

Figure 14: results by mpc3 starting from T0_1

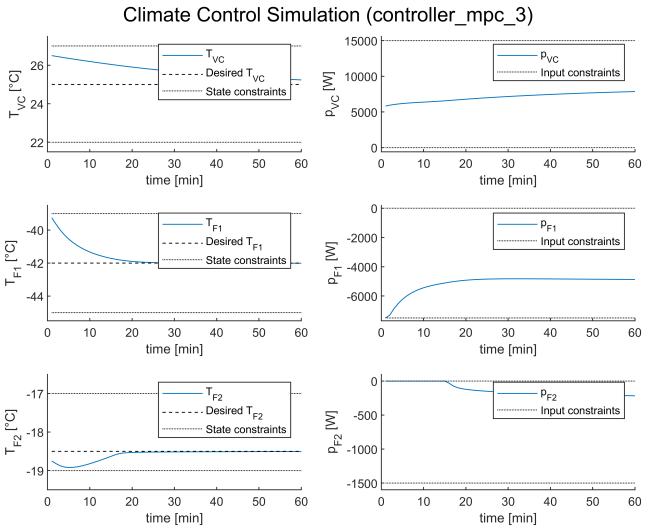

Figure 15: results by mpc3 starting from T0_2

15 Compare the closed-loop trajectories and optimization cost

Here, by plotting the results of different controllers together, it can be seen that the trajectories of mpc1 and mpc3 overlap, and the corresponding cost is also the same; the trajectory of mpc2 is different; by calculating the optimization cost, it shows that the input cost of mpc2, which directly reflects the power consumption, is the highest, implying the controller is less environmentally friendly than mpc1 and mpc3.

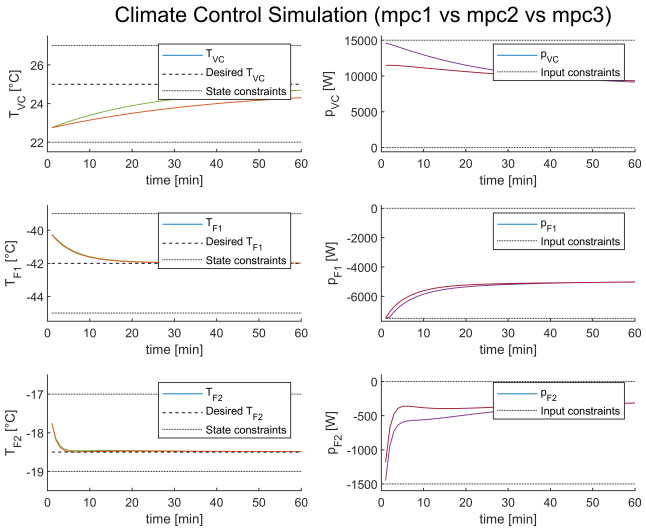

Figure 16: compare the close-loop trajectories starting from T0_1

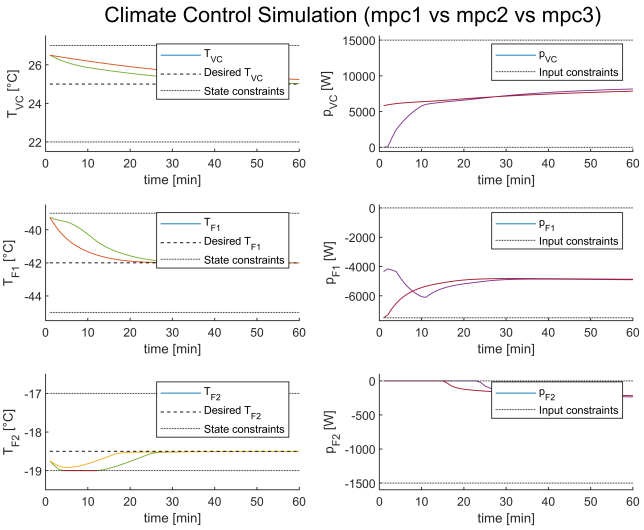

Figure 17: compare the close-loop trajectories starting from T0_2

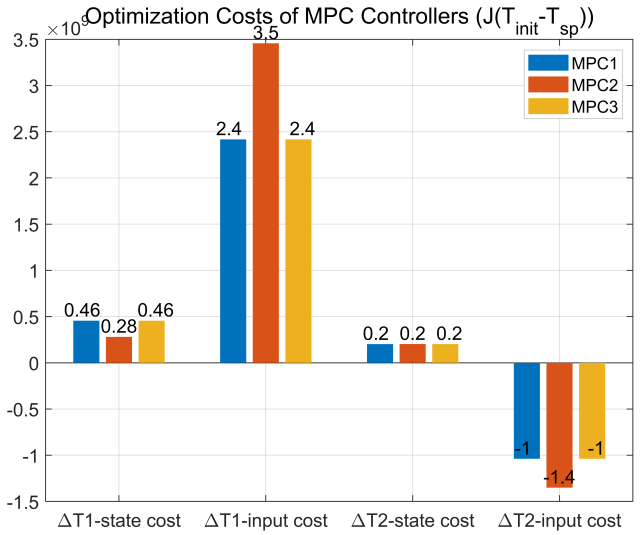

Figure 18: compare the optimization costs of three controllers

16 Setting a different terminal set as constraint

Here we scale the computed invariance set of the LQR controller to $\mathcal{X}_f$ and therefore maintains the invariance. $\mathcal{X}_f$ is actually a subset of the computed maximum invariance set. So $\mathcal{X}_f \cap X_{LQR}$ is also an invariant set under the LQR controller, and the results show that it yields an asymptotically stable controller.

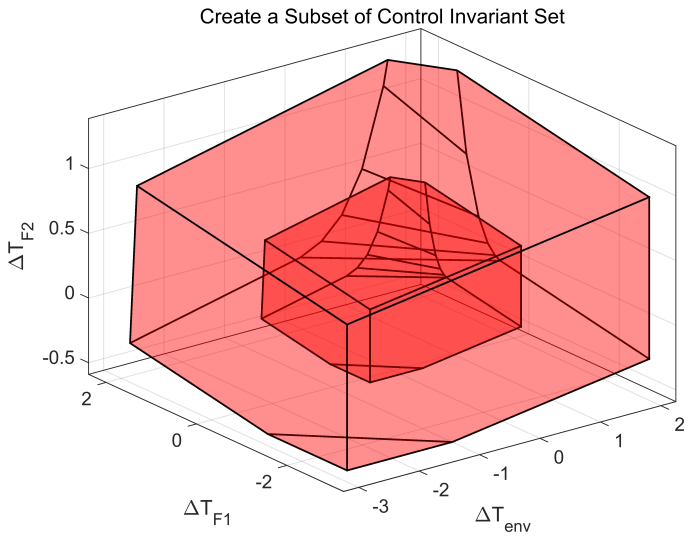

Figure 19: scale the original invariant set to $\mathcal{X}_f$

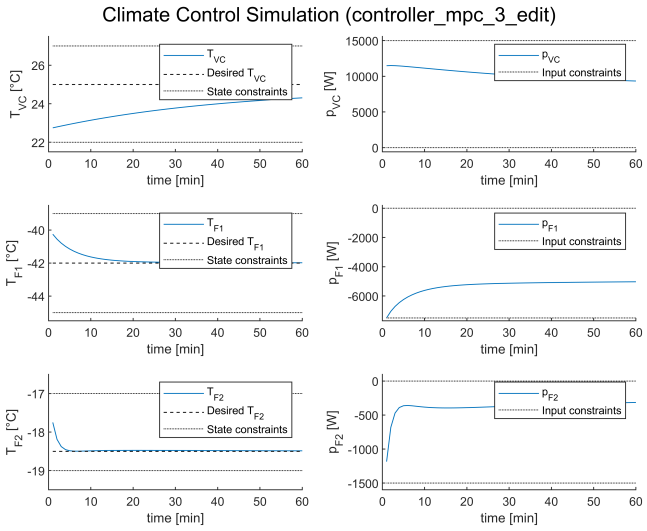

Figure 20: using the new mpc starting from T0_1

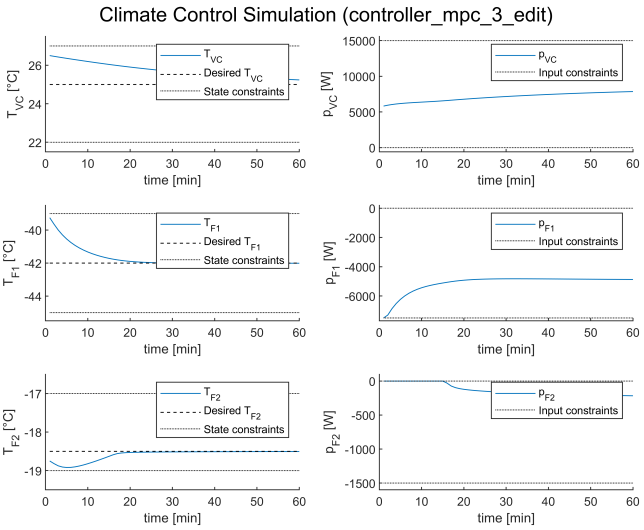

Figure 21: using the new mpc starting from T0_2

17 Under Scenario2

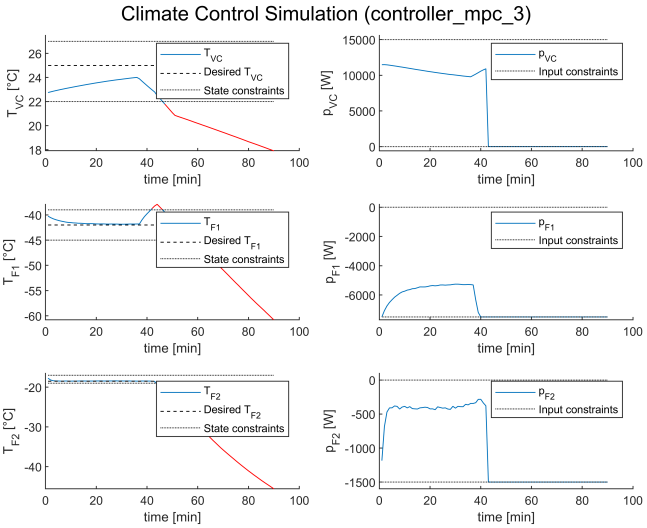

Figure 22: using the controller mpc3 under scen2

It can be seen from the figure, at around 40 minutes, the outside imposed a lot of disturbance on the system. At this time, the system states are no longer feasible for the controller mpc3, and the controller crashes.

18 Extend the MPC controller with soft constraints

Utilizing soft constraints, the cost function is given by:

$$\min_{u} \sum_{i=0}^{N-1} x_i^T Q x_i + u_i^T R u_i + l_\varepsilon\left(\varepsilon_i\right) + X_N^T P X_N + l_\varepsilon\left(\varepsilon_N\right)$$

Slack variables are introduced to minimize the duration of the violation and minimize the size of the violation.
The results show that the system temperature exceeded the limit for 10 minutes, but MPC successfully provided feasibility.

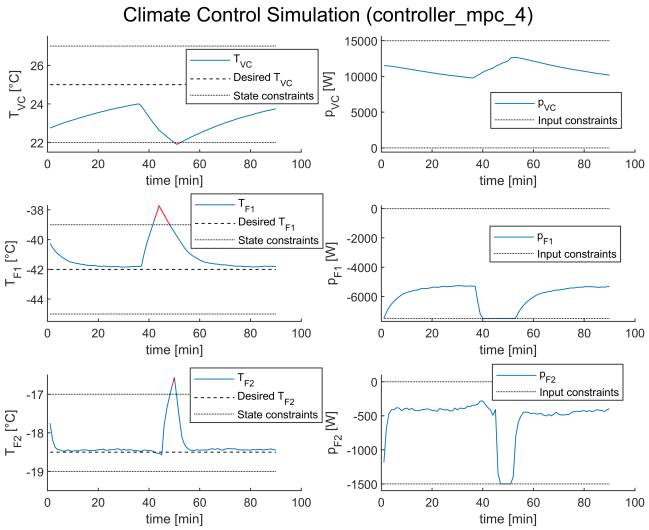

Figure 23: implementing mpc4 with soft constraints

19 Compare trajectories under scenario1

By plotting the trajectories together, it shows that the figures fully overlaps, thus the behavior of using mpc3 and mpc4 is unchanged.

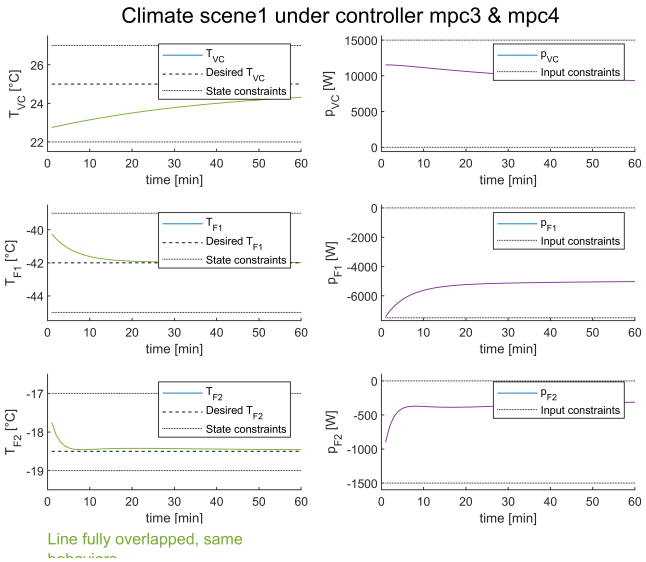

Figure 24: same behaviors using mpc3 and mpc4 under scen1

20 Equip the soft-constraint MPC with knowledge of the future disturbance

Our goal is to define a matrix that represents the expected temperature disturbance for the length of the close-loop simulation. We implement a moving-average method. Each time the controller make predictions for the horizon, it uses the calculated disturbance of last three times, obtained by the difference between the estimated values and the actual temperatures, as the input disturbance this time. In this way, the controller can quickly react to sharp disturbance in the input.
The result shows that using mpc5, the system temperatures won't exceed the state constraints, which yields a great improvement compared to the controller mpc4.

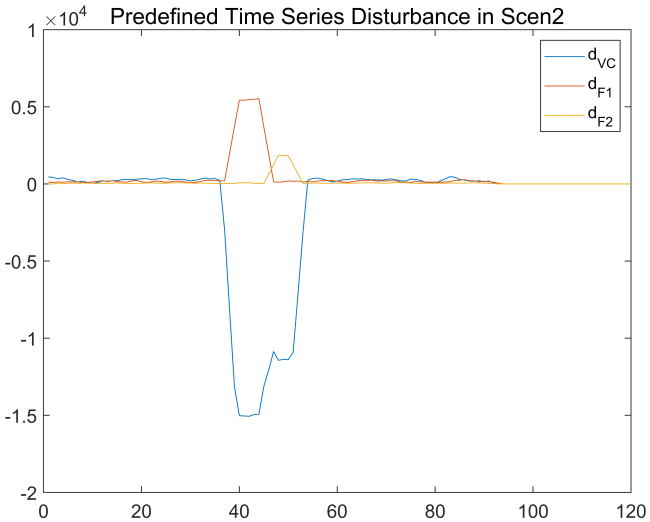

Figure 25: predefined disturbance for mpc5

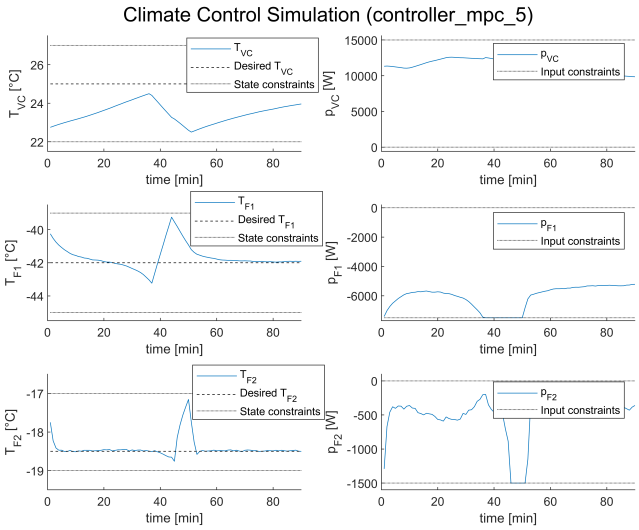

Figure 26: equip the soft-constraint MPC with knowledge of disturbance

21 Offset-free MPC

Provide the matrices A_{aug}, B_{aug}, C_{aug} and D_{aug} for the augmented state.
The augmented system dynamics are given by:

$$\begin{bmatrix} T(k+1) \\ d(k+1) \end{bmatrix} = \begin{bmatrix} A & B_d \\ 0 & I \end{bmatrix} \begin{bmatrix} T(k) \\ d(k) \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} p(k)$$

$$y_k = \begin{bmatrix} C & C_d \end{bmatrix} \begin{bmatrix} x(k) \\ d(k) \end{bmatrix} + C_d d(k)$$

We have:

$$A_{aug} = \begin{bmatrix} A & B_d \\ O & I \end{bmatrix}, B_{aug} = \begin{bmatrix} B \\ O \end{bmatrix}, C_{ang} = \begin{bmatrix} I & 0 \end{bmatrix}, D_{aug} = 0$$

22 Design an observer gain L in the linear observer

L is provided by pole-placing method. The eigenvalues we choose are $[0, 0, 0.3, 0.3, 0.3, 0]$, the obtained observer gain is:

$$L = \begin{bmatrix} -1.695 & -0.0003214 & -0.0005714 \\ -0.003462 & -1.686 & -0.01054 \\ -0.01846 & -0.03162 & -1.6499 \\ -49000 & 0 & 0 \\ 0 & -4550 & 0 \\ 0 & 0 & -1517 \end{bmatrix}$$

The expression for computing the steady-state based on the estimated disturbances.

$$\begin{bmatrix} A - I & B \\ C & 0 \end{bmatrix} \begin{bmatrix} \widehat{x}_\infty \\ u_\infty \end{bmatrix} = \begin{bmatrix} -B_d \widehat{d}_\infty \\ y_\infty - C_d \widehat{d}_\infty \end{bmatrix}$$

Here, $\widehat{d}_\infty$ and y_∞ are known, so we can calculate $\widehat{x}_\infty$ and u_∞.
The error dynamics are given by:

$$\begin{bmatrix} T(k+1) - \widehat{T}(k+1) \\ d(k+1) - \widehat{d}(k+1) \end{bmatrix} = \left(\begin{bmatrix} A & B_d \\ 0 & I \end{bmatrix} + \begin{bmatrix} L_x \\ L_d \end{bmatrix} \begin{bmatrix} I & 0 \end{bmatrix} \right) \begin{bmatrix} x(k) - \widehat{x}(k) \\ d(k) - \widehat{d}(k) \end{bmatrix}$$

The resulting eigenvalues are $[0, 0, 0.3, 0.3, 0.3, 0]$.

23 Tracks the computed steady-state

We utilize the augmented system dynamics and store the state and disturbance estimates using persistent variables. The three temperature residuals are all within 0.1℃.

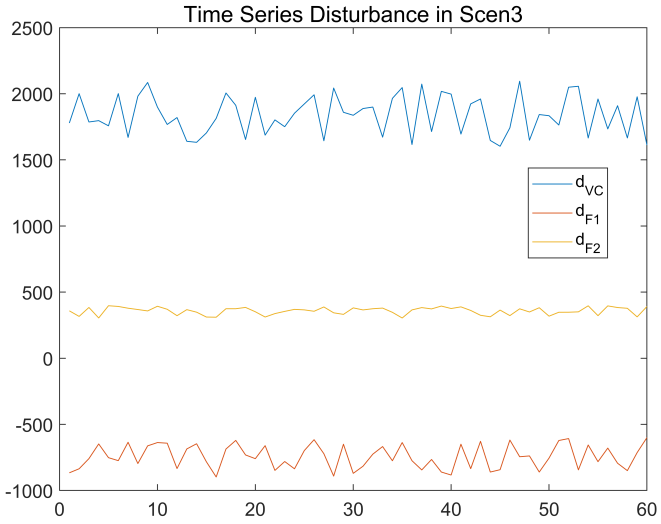

Figure 27: given disturbance in scen3

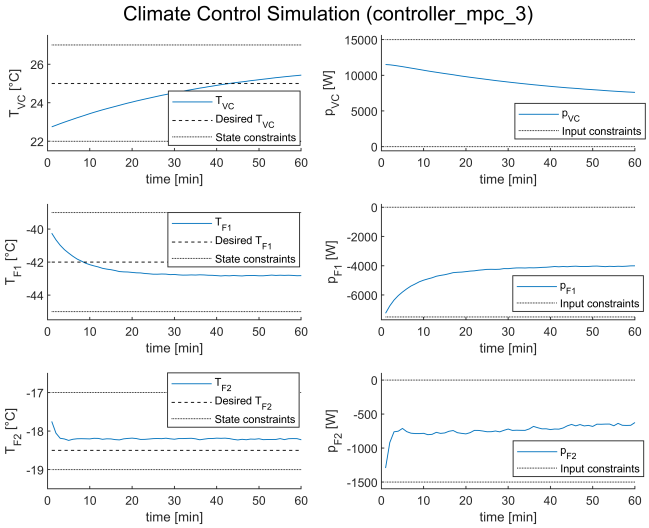

Figure 28: scen3 with mpc3 results in offset

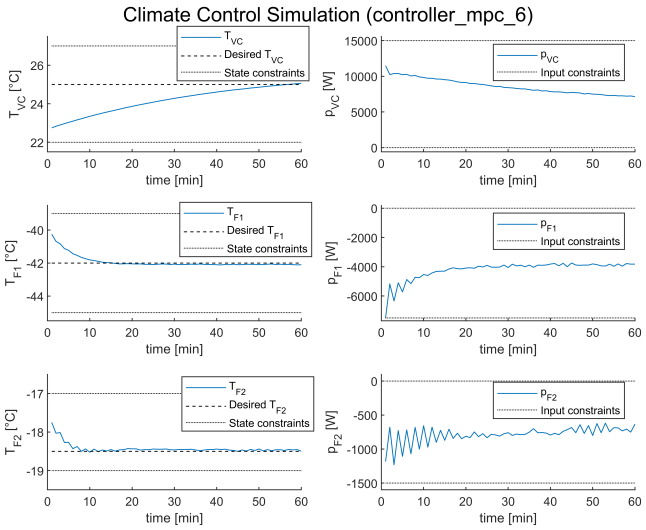

Figure 29: scen3 with mpc6 results in offset-free tracking

24 Implement a MPC using FORCES PRO

We call the yalmip2forces function to generate the code. According to the information provided by FORCES PRO, this is a bit slower than the direct input format of FORCES, but the time performance is still much better than the original controller.

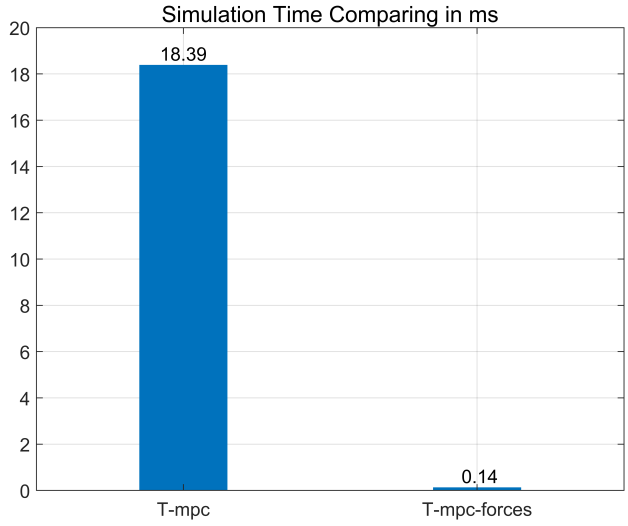

Figure 30: compare the simulation times of mpc1 and mpc1_FORCES