5. Model predictive control

In this chapter, we will add control and state constraints to our linear-quadratic regulator problem. Specifically, we are interested in finding the optimal control policy for the problem

$$\begin{array}{lll} \text{minimize} & \sum_{t=0}^{\infty} x_t^T Q_1 x_t + u_t^T Q_2 u_t & \\ \text{subject to} & x_{t+1} = Ax_t + Bu_t & t = 0, 1, \dots \\ & x_t \in X_t & t = 0, 1, \dots \\ & u_t \in U_t & t = 0, 1, \dots \end{array}$$

Unfortunately, even for linear systems and linear (polyhedral) state and control constraints, the optimal control policy is difficult to find. The cost-to-go function is typically not quadratic and is therefore hard to compute and represent (compare Example 3.7). Rather than relying on dynamic programming and aiming for optimal stationary policies, we will compute open loop controls optimized over a finite horizon, apply the first control action, and then replan the next sampling instant. This is the essence of model predictive control, MPC. We will show that these control policies can be designed to have guaranteed stability and near-optimal performance. To develop intuition, we will start by considering the receding horizon LQR problem without constraint, and show how it can be set-up to ensure closed-loop stability.

5.1 Linear-quadratic receding-horizon control

The basic idea behind *receding-horizon control* (RHC) is to plan an optimal control sequence over a fixed horizon N, but then implement only the first control action and repeat the planning at the next sample instant; see Figure 5.1. In contrast to open-loop policies, where the complete control sequence depends only on the initial state, the receding horizon control is a feedback policy since the control action at each time depends on the state at that time.

We will first study linear-quadratic receding-horizon control, where the planning problem

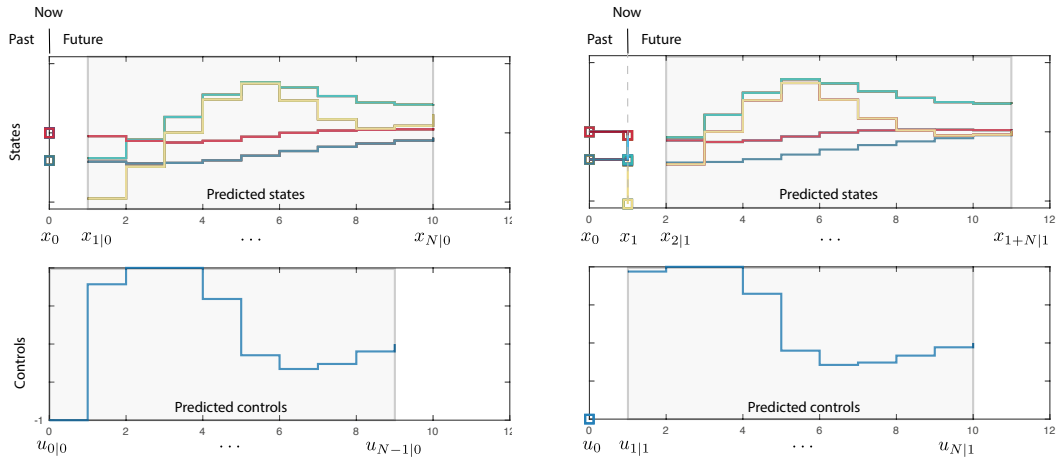

Figure 5.1: The receding-horizon principle. At time $t = 0$, we measure the true state x_0 and plan an optimal control $\{u_{0|0}, \ldots, u_{N-1|0}\}$ with respect to predicted states $\{x_{1|0}, \ldots, x_{N|0}\}$ over a finite horizon N. We implement only the first control, *i.e.* let $u_0 = u_{0|0}$. At time $t = 1$, we measure x_1 (which may differ from $x_{1|0}$ due to model inaccuracies of disturbances) and re-plan over the prediction horizon. We implement the first move, wait until the next sampling instant and repeat.

solved at each sampling instant is a finite-horizon LQR problem:

$$\begin{array}{lll} \text{minimize} & \sum_{k=0}^{N-1} x_{t+k|t}^T Q_1 x_{t+k|t} + u_{t+k|t}^T Q_2 u_{t+k|t} + x_{t+N|t}^T Q_f x_{t+N|t} & \\ \text{subject to} & x_{t+k+1|t} = A x_{t+k|t} + B u_{t+k|t} & k = 0, \ldots, N-1 \\ & x_{t|t} = x_t & \end{array} \tag{5.1}$$

Here, we have introduced the notation $x_{t+k|t}$ as the decision variable which represents the controller's prediction of the process state at time $t+k$, given knowledge of the true state at time t. Similarly, $u_{t+k|t}$ is the decision variable which represents the predicted optimal control at time $t+k$. At every time t, the controller measures x_t and uses it as initial value for its predictions. The optimal control actions over the planning horizon N are then computed and the controller applies

$$u_t = u_{t|t}^\star \tag{5.2}$$

i.e. the predicted optimal control at time t with respect to the optimization problem (5.1).

We have seen how the finite-horizon LQR problem (5.1) can be solved using dynamic programming. The optimal control sequence is on the form

$$u_{t+k|t}^\star = -L_k x_{t+k|t}^\star \qquad\qquad k = 0, \ldots, N-1$$

where

$$L_k = (Q_2 + B^T P_{k+1} B)^{-1} B^T P_{k+1} A$$

and the matrices P_k are computed via the Riccati recursion

$$P_{k-1} = Q_1 + A^T P_k A - A^T P_k B (Q_2 + B^T P_k B)^{-1} B^T P_k A \tag{5.3}$$

from boundary value $P_N = Q_f$. The receding-horizon control policy is to apply the first action in the optimized control sequence $\{u_{t|t}^\star, u_{t+1|t}^\star, \ldots, u_{t+N-1|t}^\star\}$, *i.e.* to use

$$u_t = u_{t|t}^\star = -L_0 x_{t|t}^\star = -L_0 x_t$$

Thus, although we formally replan the control actions over a finite horizon at every time instant, we always apply the same linear feedback based on the actual process state. Unfortunately, as the next example shows, this policy does not necessarily yield a stable closed loop.

■ **Example 5.1** Consider a discrete-time linear system with

$$A = \begin{bmatrix} 5/4 & -1/4 \\ 1/4 & 1 \end{bmatrix}, \quad B = \begin{bmatrix} 1/4 \\ 1/4 \end{bmatrix}$$

and the receding-horizon problem given by $Q_1 = Q_f = I$, $Q_2 = 1$ and $N = 1$. The one-step optimal receding-horizon control is

$$u_t = -L_0 x_t = -\begin{bmatrix} 1/3 & 1/6 \end{bmatrix} x_t$$

which yields an unstable closed-loop system

$$x_{t+1} = (A - BL_0)x_t = \begin{bmatrix} 7/6 & -7/24 \\ 1/6 & 23/24 \end{bmatrix} x_t$$

Admittedly, a one-step prediction horizon is very short and it is natural to ask if a larger horizon would yield a stable closed-loop. Although longer horizon *often* helps, it is by no means guaranteed. For this system, receding-horizon control will yield a stable closed loop for $N = 3$, result in closed-loop instability for N in the range $4-7$, and then ensure stability for larger horizons. ■

Even though the relationship between prediction horizon and closed-loop stability is difficult to characterize, we should expect to be able to guarantee closed-loop stability by the appropriate terminal cost. The intuition behind this expectation is that the infinite-horizon LQR control, which we have shown results in an asymptotically stable closed-loop, is computed by minimizing the cost of a single stage plus the cost-to-go

$$u^\star(x) = \arg\inf_w \left\{ x^T Q_1 x + w^T Q_2 w + v(Ax + Bw) \right\}$$

The infinite-horizon cost-to-go is quadratic $v(x) = x^T P x$, where P satisfies the ARE (4.4), so it fits perfectly into our problem formulation; see Figure 5.1. The next result confirms our intuition.

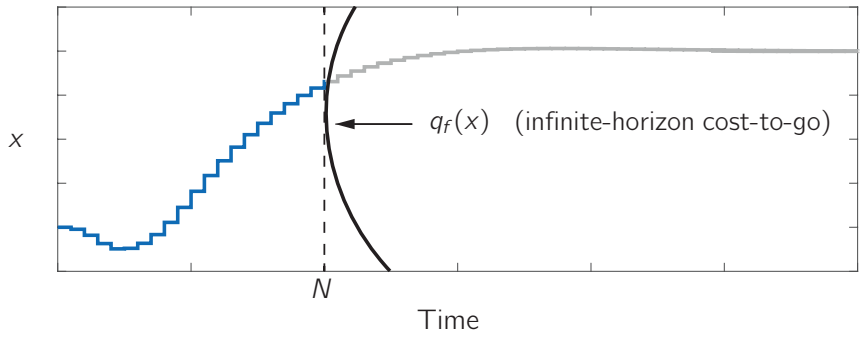

Figure 5.2: By using the infinite-horizon LQR cost-to-go as terminal cost for the receding-horizon LQR, the two strategies effectively become identical.

Theorem 5.1.1 Consider the linear system $x_{t+1} = Ax_t + Bu_t$ with (A,B) reachable. Let $Q_2 \succ 0$ and $(A, Q_1^{1/2})$ be observable. Then, if $Q_f = P$ where

$$P = Q_1 + A^T PA - (B^T PA)^T (Q_2 + B^T PB)^{-1} B^T PA$$

the receding-horizon control (5.1), (5.2) results in an asymptotically stable closed-loop system

for all $N \geq 1$. Moreover, the control is a linear state feedback $u_t = -Lx_t$ where L satisfies

$$L = (Q_2 + B^T P B)^{-1} B^T P A$$

Proof. Letting $Q_f = P$ in (5.3) yields $P_t = P$ for all times. Hence, the optimal control equals the infinite-horizon optimal LQ solution, which is guaranteed to yield an asymptotically closed-loop under the given conditions. ■

■ **Example 5.2** We can now return to the previous example, and set Q_f equal to the stationary Riccati solution for the given linear system and cost matrices Q_1 and Q_2. We then find

$$Q_f = P = \begin{bmatrix} 22.4351 & -26.6541 \\ -26.6541 & 48.6485 \end{bmatrix}, \qquad L_0 = \begin{bmatrix} 0.0266 & 2.7297 \end{bmatrix}$$

and it is straight-forward to verify that the closed-loop system is indeed stable. ■

Recall that the Lyapunov function in the stability proof for the infinite-horizon LQR is the infinite-horizon cost-to-go $v(x) = x^T P x$ where P satisfies the algebraic Riccati equation (4.4). By letting $Q_f = P$ in the receding-horizon formulation, the finite-horizon LQ problem coincides with the infinite-horizon problem. We can therefore use the infinite-horizon cost-to-go as Lyapunov function also for the receding-horizon control. Moreover, the infinite-horizon cost-to-go is identical to the optimal value of the predicted RHC cost

$$J^\star(x) = \begin{array}{ll} \min\limits_{\substack{u_{t|t},\ldots,u_{t+N-1|t} \\ x_{t|t},\ldots,x_{t+N|t}}} & \sum_{k=0}^{N-1} q(x_{t+k|t}, u_{t+k|t}) + q_f(x_{t+N|t}) \\ \text{subject to} & x_{t+k+1|t} = Ax_{t+k|t} + Bu_{t+k|t} \\ & x_{t|t} = x \end{array} \tag{5.4}$$

where $q(x,u) = x^T Q_1 x + u^T Q_2 u$ and $q_f(x) = x^T Q_f x$. As the next result shows, $Q_f = P$ is not the only choice for which the predicted RHC cost (5.4) works as Lyapunov function.

Theorem 5.1.2 Consider the system $x_{t+1} = Ax_t + Bu_t$ with (A,B) reachable. If $Q_1 \succeq 0$ with $(Q_1^{1/2}, A)$ observable, $Q_2 \succ 0$ and $Q_f = P$ where P satisfies the Lyapunov equation

$$(A - B\tilde{L})^T P (A - B\tilde{L}) - P = -(Q_1 + \tilde{L}^T Q_2 \tilde{L})$$

for some $\tilde{L}$ such that $A - B\tilde{L}$ is Schur, then the receding horizon control (5.1), (5.2) results in an asymptotically stable closed loop for all $N \geq 1$.

Proof. Let $\{x^\star_{t+k|t}\}$ and $\{u^\star_{t+k|t}\}$ be the optimal solution to (5.4) with $x = x_t$. Then

$$J^\star(x_t) = \sum_{k=0}^{N-1} q(x^\star_{t+k|t}, u^\star_{t+k|t}) + q_f(x^\star_{t+N|t})$$

Assume that we apply the optimal predicted control $u_t = u^\star_{t|t}$. What would be the predicted cost of x_{t+1}, under the control sequence $\{u^\star_{t+1|t}, \ldots, u^\star_{t+N-1|t}, \tilde{u}_{t+N|t}\}$? Since $x_{t+1} = Ax_t + Bu^\star_{t|t} = x^\star_{t+1|t}$, this control will lead to the predicted state sequence $\{x^\star_{t+1|t}, x^\star_{t+2|t}, \ldots, x^\star_{t+N|t}, \tilde{x}_{t+N+1|t}\}$ where $\tilde{x}_{t+N+1|t} = Ax^\star_{t+N|t} + B\tilde{u}_{t+N|t}$. The predicted cost of x_{t+1} under the proposed control sequence is therefore

$$\begin{aligned} J(x_{t+1}) &= \sum_{k=1}^{N-1} q(x^\star_{t+k|t}, u^\star_{t+k|t}) + q(x^\star_{t+N|t}, \tilde{u}_{t+N|t}) + q_f(\tilde{x}^\star_{t+N+1|t}) = \\ &= J^\star(x_t) - q(x^\star_{t|t}, u^\star_{t|t}) - q_f(x^\star_{t+N|t}) + q(x^\star_{t+N|t}, \tilde{u}_{t+N|t}) + q_f(\tilde{x}^\star_{t+N+1|t}) \end{aligned}$$

Let $\tilde{u}_{t+N|t} = -\tilde{L}x^\star_{t+N|t}$, so that $\tilde{x}_{t+N+1|t} = (A - B\tilde{L})x^\star_{t+N|t}$. If we pick Q_f as suggested in the theorem, the last three terms vanish in the above expression. Since $J^\star(x_{t+1}) \leq J(x_{t+1})$, we then have

$$J^\star(x_{t+1}) \leq J^\star(x_t) - q(x^\star_{t|t}, u^\star_{t|t}) = J^\star(x_t) - \left(x_t^T Q_1 x_t + u_t^T Q_2 u_t\right)$$

where we have used $x^\star_{t|t} = x_t$, $u^\star_{t|t} = u_t$. By summing both sides of the inequality

$$\sum_{t=0}^{\infty} \left(x_t^T Q_1 x_t + u_t^T Q_2 u_t\right) \leq J^\star(x_0) - \lim_{k\to\infty} J^\star(x_k) \leq J^\star(x_0)$$

where the second inequality follows since $J^\star$ is non-negative for all arguments. Since $J^\star$ is bounded, Cauchy's convergence criterion implies that

$$\lim_{t\to\infty} \left(x_t^T Q_1 x_t + u_t^T Q_2 u_t\right) = 0.$$

As both the terms are non-negative, they must both tend to zero. Now, $Q_2 \succ 0$ implies that $\lim_{t\to\infty} u_t = 0$ and by the observability assumption $x_t^T Q_1 x_t \to 0$ implies that $\lim_{t\to\infty} x_t = 0$. ■

Receding-horizon linear quadratic control is somewhat artificial, since the control is always a linear state feedback whose gains can be computed off-line. However, this will no longer be the case when we introduce state and control constraints. As we will see, the insight that we have gained by analyzing the linear case will be useful in bringing understanding to the (considerably more complex) control of constrained systems. The observation that we can get an unstable closed-loop if we choose too short a horizon and an inappropriate terminal penalty will also be true in the presence of constraints. Choosing the final penalty to reflect the infinite-horizon cost-to-go at the end of the horizon (as was done in Theorem 5.1.1) will remain a good idea, and closed-loop stability can often be established using the predicted infinite-horizon control cost as Lyapunov function.

5.2 Model-predictive control: a first attempt

We now return to the infinite-horizon LQ problem under constraints

$$\begin{array}{lll} \text{minimize} & \sum_{t=0}^{\infty} x_t^T Q_1 x_t + u_t^T Q_2 u_t & \\ \text{subject to} & x_{t+1} = Ax_t + Bu_t & t = 0, 1, \dots \\ & x_t \in X, \quad u_t \in U & t = 0, 1, \dots \end{array} \tag{5.5}$$

with the standing assumptions that (A, B) is reachable, $Q_2 \succ 0$, $Q_1 \succeq 0$ with $(A, Q^{1/2})$ observable, and that the constraint sets X and U are convex. Formally, we can view this problem as a convex optimization problem with an infinite number of variables $\{x_t, u_t\}_{t=0}^{\infty}$. We can split this problem into two: an initial finite-horizon problem followed by an infinite-horizon *tail problem*:

$$\begin{array}{lll} \text{minimize} & \sum_{t=0}^{N-1} x_t^T Q_1 x_t + u_t^T Q_2 u_t + \sum_{t=N}^{\infty} x_t^T Q_1 x_t + u_t^T Q_2 u_t & \\ \text{subject to} & x_{t+1} = Ax_t + Bu_t & t = 0, 1, \dots \\ & x_t \in X, \quad u_t \in U & t = 0, 1, \dots \end{array}$$

We can then replace the tail problem by the infinite-horizon cost-to-go function from x_N at time N

$$\begin{array}{lll} \text{minimize} & \sum_{t=0}^{N-1} x_t^T Q_1 x_t + u_t^T Q_2 u_t + v(x_N) & \\ \text{subject to} & x_{t+1} = Ax_t + Bu_t & t = 0, 1, \dots, N-1 \\ & x_t \in X, \quad u_t \in U & t = 0, 1, \dots, N-1 \end{array}$$

So far, this reformulation is only formal, since we do not have any efficient way for computing and representing $v(x_N)$. However, we do know that v is quadratic in the absence of constraints. We

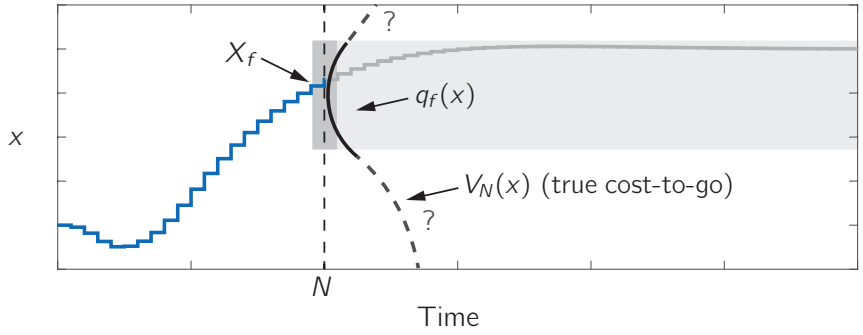

Figure 5.3: The terminal state ensures that the state at the end of the prediction horizon is in a region of the state space where we have a good (typically quadratic) estimate of the remaining cost-to-go for the associated infinite-horizon problem.

will therefore approximate v by a quadratic estimate q_f of the cost-to-go function and introduce a constraint which enforces x_N to be such that we can expect the system to remain in linear operation from time N and on; see Figure 5.3.

Thus, in the model predictive control approach, we perform finite-horizon planning in a receding-horizon fashion, *i.e.* measure the plant state x_t, solve the optimization problem

$$\begin{array}{lll} \text{minimize} & \sum_{k=0}^{N-1} x_{t+k|t}^T Q_1 x_{t+k|t} + u_{t+k|t}^T Q_2 u_{t+k|t} + x_{t+N|t}^T Q_f x_{t+N|t} & \\ \text{subject to} & x_{t+k+1|t} = Ax_{t+k|t} + Bu_{t+k|t} & k = 0,\ldots,N-1 \\ & x_{t+k|t} \in X, \quad u_{t+k|t} \in U & k = 0,\ldots,N-1 \\ & x_{t+N|t} \in X_f & \\ & x_{t|t} = x_t & \end{array} \tag{5.6}$$

in variables $\{x_{t|t},\ldots,x_{t+N|t}\}$ and $\{u_{t|t},\ldots,u_{t+N-1|t}\}$, and then apply the control

$$u_t = u_{t|t}^\star \tag{5.7}$$

(the first action control in the planned sequence). At the next sampling instant, we measure the plant state again and repeat the same procedure.

If X, U and X_f are polyhedral (described by linear inequalities), the problem is a convex quadratic program, which can be efficiently solved. We can either use an algebraic modelling language, such as YALMIP, or manually translate the problem into a standard form for QPs, as described in Section 3.2. Figure 5.2 describes the code required to formulate an MPC controller with simple magnitude bounds on the control and output.

Altitude control of a Cessna citation aircraft

To get some initial experience with MPC, we consider altitude control of an aircraft. The particular model we will use is taken from [**Mac:02**], and describes the linearized dynamics of a Cessna citation aircraft flying with a speed of 128.2 m/s at an altitidude of 5000 m.

$$\dot{x}(t) = \begin{bmatrix} -1.2822 & 0 & 0.98 & 0 \\ 0 & 0 & 1 & 0 \\ -5.4293 & 0 & -1.8366 & 0 \\ -128.2 & 128.2 & 0 & 0 \end{bmatrix} x(t) + \begin{bmatrix} -0.3 \\ 0 \\ -17 \\ 0 \end{bmatrix} u(t)$$

$$y(t) = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

```
% Introduce decision variables
for t=1:N,
    x{t}=sdpvar(n,1); u{t}=sdpvar(m,1);
end;

% Define objective and contstraints
obj=0; cons=[];
for t=1:N-1,
    obj=obj+x{t}'*Q1*x{t}+u{t}'*Q2*u{t};
    cons=[cons, x{t+1}==A*x{t}+B*u{t}]; % dynamics
    cons=[cons, ymin<= C*x{t} <= ymax]; % state constraints
    cons=[cons, umin<= u{t} <=umax];    % control constraints
end;
obj=obj+x{N}'*Qf*x{N};                  % terminal cost
cons=[cons, MN*x{N} <= mN];             % terminal constraint

% Construct controller object
controller=optimizer(cons, obj, [], x{1}, u{1});
```

Figure 5.4: YALMIP code to define MPC controller with simple bounds on u and y and a terminal set on the form $X_f = \{x \mid M_N x \leq m_N\}$ The controller is called by using $u = \texttt{controller}(x)$.

Here, the control input is the elevator angle, and the states represent the angle of attack, pitch angle, pitch rate and altitude, respectively. These terms are described in more detail in Figure 5.5. The control input is limited to $\pm 15^\circ (\pm 0.262 \text{rad/s})$ and has a slew-rate (rate-of-change) limit of $\pm 30^\circ/\text{s}$. For passenger comfort, the pitch angle is limited to $\pm 20^\circ (0.349 \text{rad})$. Our aim is to design a controller which can perform swift changes in altitude while maintaining passenger comfort.

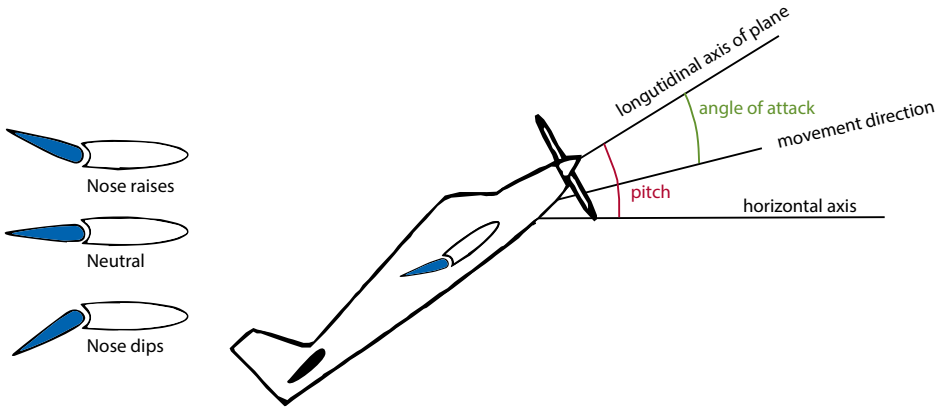

Figure 5.5: The altitude is the vertical distance between the center of mass of the plane and the ground; the angle between the horizontal axis and the longitudinal axis of the plane is called the pitch, while the angle between the moment direction and the longitudinal axis is called the angle of attack. The control problem is to manipulate the elevator surfaces on the wings in order to control the altitude, pitch rate and angle of attack (positive angles are downward, leading the nose to dip).

We will base our design on a discrete-time model sampled with $h = 0.25$ s and penalty matrices $Q_1 = I$ and $Q_2 = 10$. For simplicity, we begin with disregarding the slew-rate constraint on the actuator, and consider a small altitude change of 10m by letting $x_0 = (0,0,0,10)$. The linear LQR-optimal controller behaves well in linear simulations, but oscillates widely when the actuator magnitude constraints are included in the simulations; see Figure 5.6. In addition, the comfort constraint on pitch rate is violated. For our initial MPC design, we use a prediction horizon of $N = 10$ (2.5 seconds) and $Q_f = 0$. In contrast to the linear design, the MPC controller produces a

well-behaved response, satisfying both control and state constraints; see Figure 5.6(right).

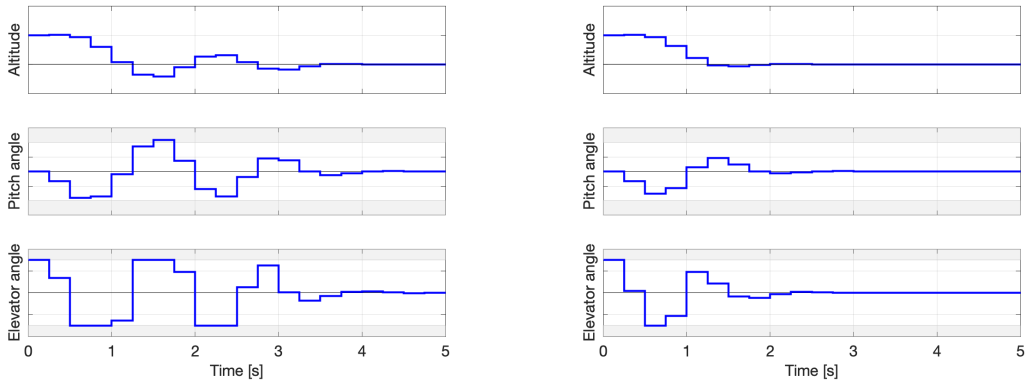

Figure 5.6: The optimal LQ controller, which disregards actuator and state constraints, behaves well in linear operation, but is unable to handle a descent of 10 meters in altitude without causing large oscillations (left). The MPC controller, on the other hand, acounts for the constraints in the design and is able to execute a well-damped descent while respecting control and state limits.

It is also straight-forward to include the slew-rate constraint

$$|u_k - u_{k-1}| \leq \delta_{\max} \qquad k = 0, 1, \ldots, N-1 \tag{5.8}$$

in the MPC design. The simplest approach is to simply add the linear constraints (5.8) to the QP problem (5.6). An alternative approach is to use a prediction model where the input increments are used as inputs. Specifically, we can replace the prediction model $x_{k+1} = Ax_k + Bu_k$ by

$$\begin{bmatrix} x_{k+1} \\ u_k \end{bmatrix} = \begin{bmatrix} A & B \\ 0 & I \end{bmatrix} \begin{bmatrix} x_k \\ u_{k-1} \end{bmatrix} + \begin{bmatrix} B \\ I \end{bmatrix} \Delta u_k$$

and consider Δu_k as an input. In this way, input constraints in the original model appear as state constraints in this model, while slew-rate constraints in the original model are transformed into input constraints. When we include the slew-rate constraints in the simulations, the (linear) LQR controller results in an unstable closed loop. The MPC design, on the other hand, deals well with the actuator limitations, see Figure 5.7(left).

The MPC controller that we have simulated does not use a terminal penalty, but relies on the relatively long prediction horizon to have near-optimal performance. It should therefore come as no surprise that when we reduce the prediction horizon to $N = 4$ the closed-loop is no longer asymptotically stable, but the aircraft goes into an undamed oscillation; see Figure 5.7(right). By letting $Q_f = P$ where P satisfies the algebraic Riccati equation of the associated LQR problem, and letting X_f be the invariant set of the LQR state feedback, stablity is re-established and the controller performs similarly to when $N = 10$ (not shown). However, when we try to execute an altitude change of 30 meters, instead, the MPC planning problem no longer admits any solution and the MPC strategy is unable to suggest any control action.

5.3 Ensuring stability and recursive feasibility

Our initial experiments with MPC on the aircraft model has shown that there are still some things that can go wrong when an MPC controller operates in closed loop:

1. the planning problem may become infeasible, in which case the optimization subroutine cannot generate any valid solution to (5.6), and

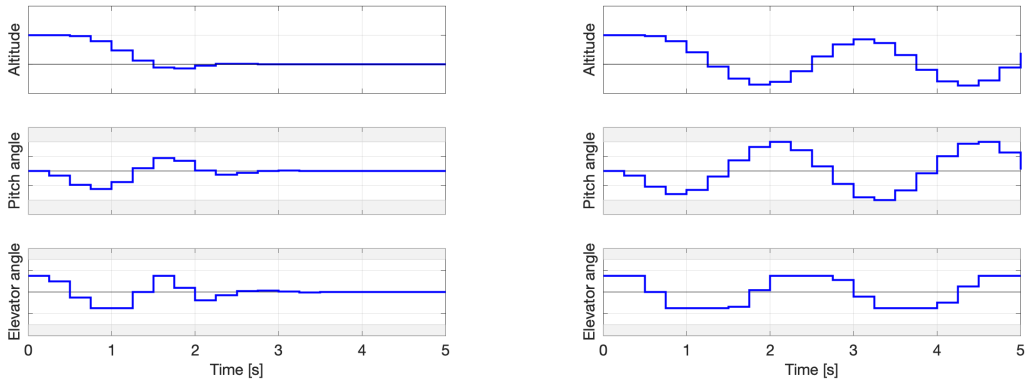

Figure 5.7: The MPC controller also handles slew-rate constraints on the elevators (left). However, when we reduce the prediction horizon to $N = 4$, the controller is no longer able to stabilize the altitude (right). In this example, this problem can be remedied by adding the appropriate terminal constraint and terminal penalty (not shown).

2. the MPC control policy might render the closed-loop system unstable.

In this section, we will give theoretical conditions that ensure feasibility and closed-loop stability for model predictive control. We begin by studying feasibility.

5.3.1 Recursive feasibility

It should be clear that the planning problem is, in general, only feasible for a limited set of initial values. As a simple example, consider the scalar integrator

$$x_{k+1} = x_k + u_k,$$

with $x_0 > 0$. Assume that we would like to ensure that $|x_N| \leq \varepsilon$ using a control which satisfies $|u_k| \leq u_{\max}$. The optimal control is then to use $u_k = -u_{\max}$, which would reach the target set in

$$\frac{1}{u_{\max}}(x_0 - \varepsilon)$$

iterations. Thus, larger initial values and smaller control authority requires larger planning horizons. Similarly, a smaller target set (smaller value of ε) also requires a longer horizon. If the open loop system is asymptotically stable, the dynamics will drive the state faster to zero and one can show that the horizon only grows logarithmically in x_0 (not linearly, as for the integrator). If the open-loop dynamics is unstable, on the other hand, there will be certain states which cannot be driven to zero with a bounded control, no matter the horizon length.

For linear systems with more than one state, it is harder to characterize how the set of feasible states depends on the horizon length. It makes sense to use at least $N \geq n$, since we will in general need n steps to drive the system state to a target, even in the absence of constraints. When states and controls are constrained, the required horizon length to reach a target set typically increases, but no explicit formula exists. In essence, one has to resort to numerical calculations.

In model predictive control, however, feasibility is even more complex. As the next example shows, starting an MPC controller from a feasible state does not guarantee that the planning problems will remain feasible for all future times.

■ **Example 5.3 — Loss of feasibility.** Consider the double integrator

$$x_{t+1} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} x_t + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u_t$$

subject to the constraints $|u_t| \leq 0.5$ and $\|x_t\|_\infty \leq 5$ for all t. In addition, we let

$$Q_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad Q_2 = 1, \quad N = 3, \quad X_f = \mathbb{R}^2.$$

Starting from $x_0 = \begin{bmatrix} -3; 3 \end{bmatrix}$, the MPC planning problem is feasible and the computed control is $u_0 = -0.5$. Applying this control yields $x_1 = [0; 2.5]$ for which no feasible solution to the MPC planning exists. Figure 5.8 (left) illustrates the set of states for which the MPC planning problem has a feasible solution: while x_0 is inside, x_1 is not, and feasibility is lost. Comparing this set with the control invariant set (Figure 5.8, right) reveals that x_1 is could, in fact, not have been driven to zero by any controller. ∎

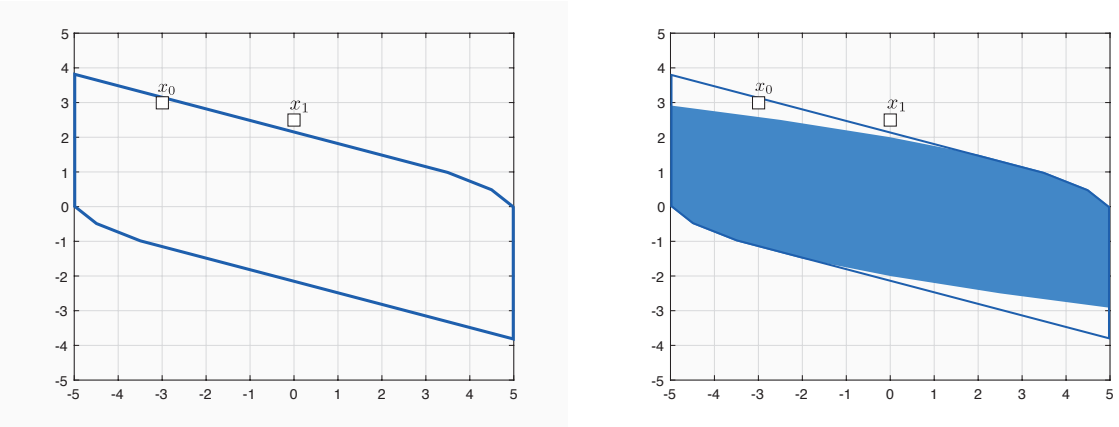

Figure 5.8: Left figure shows the set of initial values for which the MPC planning problem in Example 5.3 admits a feasible solution. Even for initial states in this set, the MPC planning problem can become infeasible in the future. In particular, as the right figure shows, x_1 lies outside the control invariant set for the system, so no admissible controller can handle this initial value.

The property that the MPC planning problem remains feasible for all future times is referred to as *recursive feasibility*, defined next.

Definition 5.3.1 — Recursive feasibility. The MPC problem (5.6) is *recursively feasible* from x_0 if the existence of a feasible solution implies that the problem (5.6) will remain feasible from every state along the closed-loop trajectory.

If we initialize our system from a recursive feasible state, then there will be a feasible solution to (5.6) for every future state generated under the MPC control law (5.7). The next result states that recursive feasibility is ensured if we use terminal sets which are control invariant.

Proposition 5.3.1 If the terminal set X_f is control invariant, then the MPC problem (5.6) is recursively feasible from all system states x_0 that admit a feasible solution.

Proof. By assumption, x_0 allows us to solve (5.6) and find optimal controls $\{u^\star_{t|t}, \ldots, u^\star_{t+N-1|t}\}$, with an associated predicted state evolution $\{x^\star_{t|t}, \ldots, x^\star_{t+N|t}\}$. Since $x_1 = Ax_0 + Bu^\star_{t|t} = x^\star_{t+1|t}$, we can apply the control sequence $\{u^\star_{t+1|t}, \ldots, u^\star_{t+N-1|t}, \tilde{u}_{t+N|t}\}$ from x_1 to obtain a state evolution $\{x^\star_{t+1|t}, \ldots, x^\star_{t+N|t}, \tilde{x}_{t+N+1|t}\}$. As $x^\star_{t+N|t} \in X_f$ and X_f is control invariant, there exists a $\tilde{u}_{t+N|t}$ which brings $\tilde{x}_{t+N+1|t} = Ax^\star_{t+N|t} + B\tilde{u}_{t+N|t}$ into X_f. Using such a $\tilde{u}_{t+N|t}$ in the proposed control sequence (and the corresponding $\tilde{x}_{t+N+1|t}$ as the final element of the predicted state sequence) yields a feasible solution to (5.6). Repeating the same argument for $t = 2, 3, \ldots$ establishes recursive feasibility.

We can now return to the double-integrator example and add a terminal set in the planning problem to ensure recursive feasibility.

■ **Example 5.4 — Recursive feasibility of double integrator.** We can suspect that since N is so small, the set of initially feasible (and, hence, recursively feasible) states will be limited since the controller has to drive the initial states into X_f in 3 steps. As shown in blue in Figure 5.9(left), using $X_f = \{0\}$ does indeed give a small set of recursively feasible states. Letting X_f be the invariant set of the infinite-horizon LQR controller, on the other hand, leads to the larger gray set of recursively feasible states. In addition, as shown in the right figure, increasing the horizon enlarges the set of recursive feasible states. ■

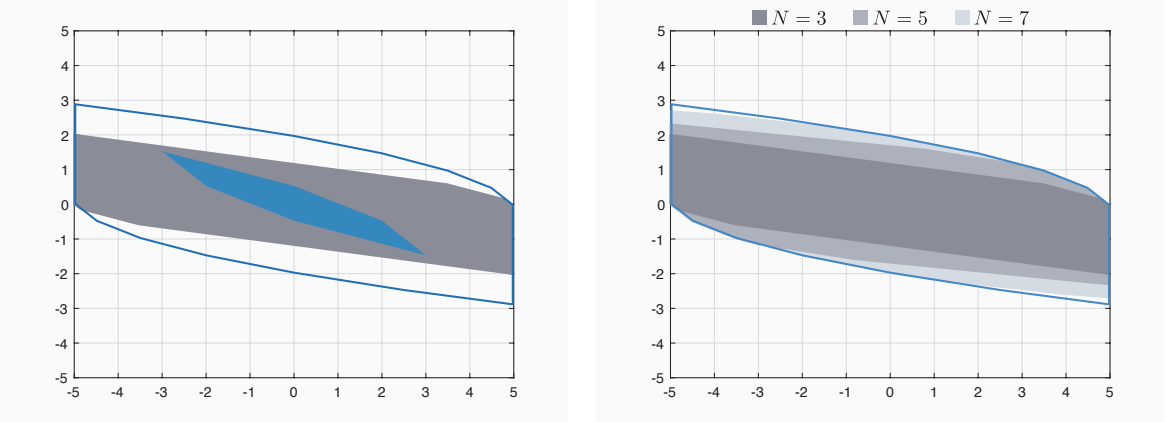

Figure 5.9: Left figure shows how a smaller terminal set gives a smaller set of recursively feasible states. The right figure shows how a longer prediction horizon N allows to enlarge the set of recursively feasible states.

5.3.2 Asymptotic stability of the model predictive control closed-loop

Once recursive feasibility can be ensured, we can give a closed-loop stability guarantee using a similar approach to the one we used for unconstrained receding-horizon control.

Theorem 5.3.2 — MPC stability. Let (A,B) be reachable and consider the linear system $x_{t+1} = Ax_t + Bu_t$ subject to constraints $x_t \in X$ and $u_t \in U$ for all $t \geq 0$ under the model predictive control (5.6),(5.7). If the following conditions hold:

(a) $Q_1 \succeq 0$ with $(Q_1^{1/2}, A)$ observable, $Q_2 \succ 0$ and $q_f(x) = x^T Q_f x$ with $Q_f \succ 0$.
(b) X, U and X_f are closed and contain 0 in their interior.
(c) X_f is control invariant under the given dynamics and constraints
(d) $\min_{\tilde{u}\in U, Ax+B\tilde{u}\in X_f} q_f(Ax+B\tilde{u}) - q_f(x) + x^T Q_1 x + \tilde{u}^T Q_2 \tilde{u} \leq 0$ for all $x \in X_f$.

then $\lim_{t\to\infty} x_t = 0$ from all initial values x_0 for which (5.6) admits a feasible solution.

Proof. By Proposition 5.3.1, Assumption (c) implies recursive feasibility. To prove stability, we will show that the predicted cost

$$
\begin{aligned}
J^\star(x) = \min_{\substack{u_{t|t},\dots,u_{t+N-1|t} \\ x_{t|t},x_{t+1|t},\dots,x_{t+N|t}}} \quad & \sum_{k=0}^{N-1} q(x_{t+k|t}, u_{t+k|t}) + q_f(x_{t+N|t}) \\
\text{subject to} \quad & x_{t+k+1|t} = Ax_{t+k|t} + Bu_{t+k|t} && k = 0,\dots,N-1 \\
& x_{t+k|t} \in X, u_{t+k|t} \in U && k = 0,\dots,N-1 \\
& x_{t+N|t} \in X_f \\
& x_{t|t} = x
\end{aligned}
\tag{5.9}
$$

is a Lyapunov function. Let x_t admit a feasible solution to (5.6) at time t and

$$U_t^\star(x_t) = \{u_{t|t}^\star, \dots, u_{t+N-1|t}^\star\}$$

be the optimal predicted control sequence associated with $J^\star(x_t)$. By the same arguments used in Proposition 5.3.1, there is a $\tilde{u}_{t+N|t}$ such that

$$\widetilde{U}_{t+1} = \{u_{t+1|t}^\star, u_{t+2|t}^\star, \dots, u_{t+N-1|t}^\star, \tilde{u}_{t+N|t}\}$$

is feasible when evaluating the predicted cost for $x_{t+1} = Ax_t + Bu_{t|t}^\star$. This cost is

$$\begin{aligned}\widetilde{J}(x_{t+1}) &= \sum_{k=1}^{N-1} q(x_{t+k|t}^\star, u_{t+k|t}^\star) + q(x_{t+N|t}^\star, \tilde{u}_{t+N|t}) + q_f(Ax_{t+N|t}^\star + B\tilde{u}_{t+N|t}) = \\ &= J^\star(x_t) - q(x_{t|t}^\star, u_{t|t}^\star) - q_f(x_{t+N|t}^\star) + q(x_{t+N|t}^\star, \tilde{u}_{t+N|t}) + q_f(Ax_{t+N|t}^\star + B\tilde{u}_{t+N|t})\end{aligned}$$

Since this control sequence is suboptimal, $J^\star(x_{t+1}) \leq \widetilde{J}(x_{t+1})$. Moreover, as $\check{x}_N^\star \in X_f$, Assumptions (c) and (d) imply that we can find $\tilde{u}_N$ such that the last three terms vanish and

$$J^\star(x_{t+1}) \leq J^\star(x_t) - q(x_t, u_t)$$

Under the given assumptions, $q(x_t, u_t)$ is a positive semidefinite function of x_t, so asymptotic stability follows by the same argument as in Theorem 5.1.2 if we can show that $J^\star(x)$ is continuous and positive definite. This part of the proof is more technical, and we refer the interested reader to [14] for the details. ■

Conditions (c) and (d) of the stability theorem are quite complex to deal with. The control invariant set may be costly to compute and complex to represent, and (d) is difficult to verify in general. It is therefore common to simply use the forward invariant set for the closed-loop dynamics under the infinite-horizon LQR control as X_f and use the infinite-horizon cost-to-go for the LQR controller to define q_f. This leads to the following corollary.

Proposition 5.3.3 Consider the discrete-time linear system $x_{t+1} = Ax_t + Bu_t$ with (A, B) reachable and constraints $x_t \in X$ and $u_t \in U$ for all $t \geq 0$. Let u_t be defined by the model predictive control law (5.6), (5.7). If conditions (a) and (b) of Theorem 5.3.2 hold and $Q_f = P$ where P satisfies

$$P = Q_1 + A^T PA - (B^T PA)^T (Q_2 + B^T PB)^{-1} (B^T PA)$$

and $X_f \subseteq X \cap \{x \mid -Lx \in U\}$ is positively invariant under $x_{t+1} = (A - BL)x_t$ where

$$L = (Q_2 + B^T PB)^{-1} B^T PA,$$

then $\{x_t\}$ converges asymptotically to zero from all x_0 for which (5.6) admits a feasible solution.

5.3.3 The interplay between horizon length and optimality

We motivated MPC by dividing an infinite-horizon optimal control problem into an N-step constrained LQR problem and an infinite-horizon tail problem. We have already seen how this approach may limit the set of recursive feasible states, especially when N is short and the terminal set is small. In addition, when we glue the two solutions together, we should expect a certain level of suboptimality since we force x_N into X_f. At the same time, if $(x, u) = (0, 0)$ lies in the interior of the constraint set, the finite-horizon problem will drive $x_N \to 0$ as $N \to \infty$. Hence, if the horizon is long enough, we should expect that x_N will be driven to X_f, even if this is not explicitly enforced, meaning that no optimality is lost by requiring that $x_n \in X_f$. One can prove that this intuition is indeed true. We state the following result without proof and refer to [15] for the full details.

Proposition 5.3.4 Let the origin lie in the interior of both X and U, and assume that Q_f has been chosen as in Proposition 5.3.3. Then there exists a finite horizon N_∞, which depends on x_t, with the property that whenever $N \geq N_\infty$: the sequence $\{u^\star_{t|t}, \ldots, u^\star_{t+N-1|t}\}$ which achieves the minimum cost for (5.6) the is equal to the first N terms of the infinite sequence $\{u^\star_t, u^\star_{t+1}, \ldots\}$ which minimizes the infinite-horizon constrained LQR cost (5.5).

In practice, we may need to use a value of N which is smaller than N_∞. As the next example shows, this does not necessarily mean that we loose optimality.

■ **Example 5.5** Consider the system

$$x_{t+1} = \begin{bmatrix} 0.5 & -1.25 \\ 1.0 & 0.0 \end{bmatrix} x_t + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u_t$$

under constraints $\|x_t\|_\infty \leq 1$, and $|u_t| \leq 1$. We let the stage-cost be defined by $Q_1 = I$ and $Q_2 = \rho I$.

We apply Corollary 5.3.3 and use the infinite-horizon LQR cost-to-go function as terminal penalty, and an invariant set for the closed-loop dynamics of the corresponding LQR-optimal controller as terminal set. Figure 5.10 demonstrates how the predicted trajectories for different horizon lengths may differ. When the horizon is short, one has to accept a increase in the LQR cost to be able to drive the system into the terminal state at the end of the prediction horizon. As shown in the same figure (right), the two control laws nevertheless generate the same closed-loop trajectories (since only the first predicted action is used, and the complete prediction is recomputed from the resulting plant state measured at the next sampling instant.)

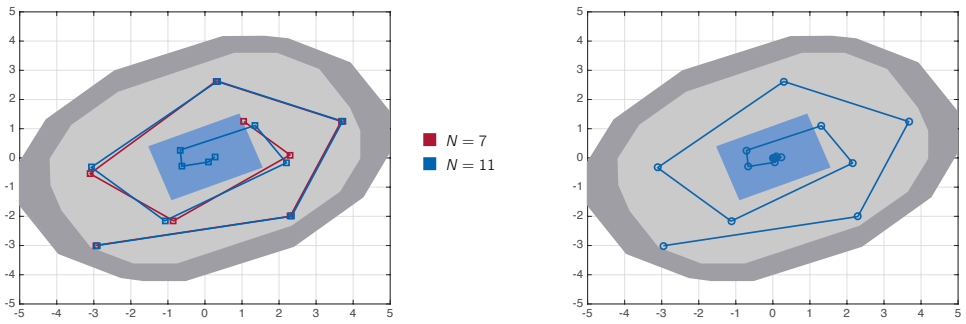

Figure 5.10: Predicted closed-loop trajectories for two different horizon lengths (left), and corresponding closed-loop trajectories (right). Note how the controller always enforces the state at the end of the prediction horizon to lie in the terminal set.

■

5.4 Practical MPC considerations

Our basic MPC formulation has features for ensuring recursive feasibility, stability and (for sufficiently long horizons) near-optimal performance. Still, many practical MPC implementations make use of a few additional ideas. We will describe some of the most important in this section.

5.4.1 Softening of constraints

We have shown that careful use of terminal sets and terminal costs guarantees both recursive feasibility and closed-loop stability. Still, there are reasons to be critical to our analysis. For example, we have assumed that the model used for predictions in our controller is identical to the actual process dynamics, and that there are no unknown disturbances acting on the system. So even

if our idealized analysis tells us that we should have recursive feasibility, it may still be that the optimization problem (5.6) becomes infeasible when run against the real process.

We can guarantee that infeasibility never happens by *softening the constraints*. The basic idea is to introduce non-negative (vectors of) *slack variables*, $s_k \geq 0$ and replace any inequality constraints

$$Hx_{t+k|t} \leq h$$

describing state constraints in the MPC problem by

$$Hx_{t+k|t} - s_k \leq h.$$

These inequalities can always be satisfied, possibly using large slack variables $s = (s_0, \ldots, s_{N+N_c})$. To make sure that constraints are only violated when necessary, one also adds a positive definite slack cost $\sigma(s)$ to the objective function of the MPC planning problem *i.e.* consider the objective

$$\sum_{k=0}^{N-1} x_{t+k|t}^T Q_1 x_{t+k|t} + u_{t+k|t}^T Q_2 u_{t+k|t} + q_f(x_{t+N|t}) + \sigma(s)$$

The slack penalty is typically quadratic, linear or a combination of both, *e.g.*

$$\sigma(s) = \kappa_q \|s\|_2^2 + \kappa_l \|s\|_1$$

where the coefficients κ_q and κ_l should be large enough to discourage constraint violations unless strictly necessary. The next example demonstrates the effect of constraint softening.

■ **Example 5.6 — Softening constraints.** Consider the linear system

$$\begin{aligned} x_{t+1} &= \begin{bmatrix} 0.8 & -0.6 & 0.2 \\ 1 & 0 & 0 \\ 0 & 0.8 & 0.6 \end{bmatrix} x_t + \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} u_t \\ y_t &= \begin{bmatrix} -1 & 1 & 1 \end{bmatrix} x_t \end{aligned}$$

subject to the constraints $|u_t| \leq 1$ and $|y_t| \leq 1$. For simplicity, we use an MPC controller without terminal state and terminal cost, and let $Q_1 = C^T C$, $Q_2 = 1$ and $N = 20$.

From the initial value $x_0 = (0.8, 0.8, 0.8)$, it is impossible to respect the upper limit on the output, and without softened constraints, the MPC problem would have been infeasible. Figure 5.11 demonstrates the effect of constraint softening with linear and quadratic penalties, respectively. The simulations show how larger penalties on the slacks give better constraint satisfaction. We can also notice how the quadratic penalty is better at avoiding large deviations, while the linear penalty is better at ensuring that the constraints are violated for a short period of time, if possible. ■

It is possible to give a theoretical justification for the observations in the example. One can prove that an infinitesimal violation ε of the original constraints leads to a decrease in the cost which is proportional to the violation. Thus, if one lets $\sigma(s)$ be quadratic, it will always be better to violate the constraint slightly (deceasing the cost proportionally to the violation, and accepting a slack penalty cost with the slower quadratic growth). To avoid these effects, one should make sure that $\sigma(s)$ grows linearly around the origin.

Although there is some theory determining tuning the slack penalty weights, in many practical applications one adopts a quadratic slack penalty and tune the weight to be large enough in simulations. Unfortunately, letting the coefficients be too large can lead to numerical ill-conditioning of the associated optimization problem, so some care has to be taken.

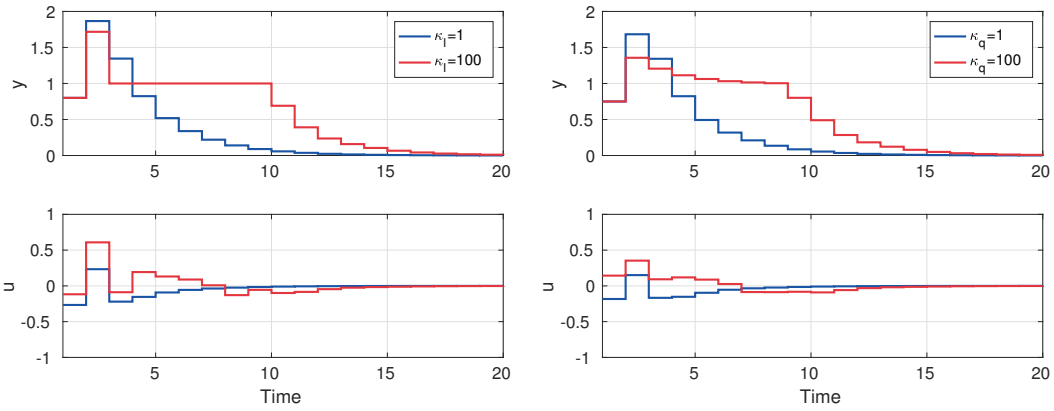

Figure 5.11: Larger penalties on slacks reduces constraint violation. Quadratic penalties are better at reducing the amplitude of violations, while linear penalties are better at limiting the duration of constraint violations.

5.4.2 The dual mode paradigm, horizons and constraints

The MPC design proposed in Corollary 5.3.3, where the infinite-horizon LQR cost-to-go is used as terminal cost and the invariant set for the associated optimal linear state feedback is used as terminal set, is sometimes referred to as *dual mode* MPC. The name comes from viewing the first N steps, where the predicted control sequence is based on open-loop optimization accounting for the constraints, as a first mode. The time from $N+1$ and on is seen as a second mode, where the infinite-horizon optimal LQR controller is used.

Many practical MPC implementations make use of yet another, intermediate, mode. Thus, the first mode allows for full freedom in optimizing the N first control actions accounting for the constraints. During the next N_c steps, the predictions are based on the assumption that the control is a linear feedback of the state, and constraints on the state and control are accounted for in the planning problem. Since the control policy during these N_c steps is fixed, the predicted states can only be influenced through $\check{x}_N$ (which is determined by the control actions during the first N steps). Beyond step $N+N_c$, one assumes that the stationary optimal controller does not violate any constraints and captures the cost-to-go by a quadratic terminal cost.

One reason for using a separate control horizon N and constraint horizon N_c is to decrease the computational effort of solving the associated quadratic program. Since the control moves are only free in the first mode, letting N be small reduces the number of variables in the optimization problem. We can compensate the short-sightedness of the predictions by adding constraint checking for another N_c steps. In these scenarios, it is still natural to require that x_{t+N+N_c} belongs to an invariant set of the linear optimal policy.

Another reason for using a separate constraint horizon is to avoid explicit calculation of the invariant terminal set. Recall that to ensure recursive feasibility, the terminal set should be control invariant. If we assume that a fixed linear controller is used after the N first steps, however, we should use the associated invariant set to define X_N. We know from Theorem 2.3.2 that if the closed-loop dynamics are asymptotically stable, and its equilibrium point is strictly feasible with respect to the constraints, then the invariant set is generated by a finite number of inequalities:

$$X_f = \left\{x \,|\, Hx \leq h \wedge H(A-BL)x \leq h \wedge \ldots \wedge H(A-BL)^{N_c}x \leq h\right\}.$$

Since we use a linear feedback $u_{t+k|t} = -Lx_{t+k|t}$ for $k = N,\ldots,N+N_c-1$, the predictions in this mode satisfy $x_{t+k+1|t} = (A-BL)x_{t+k|t}$, *i.e.* $x_{t+N+k|t} = (A-BL)^k x_{t+N|t}$. By enforcing the constraint

that $x_{t+N+k|t} \in X = \{x \mid Hx \leq h\}$ for $k = 1, \ldots, N_c$, we effectively require that

$$\begin{aligned} Hx_{t+N|t} &\leq h \\ H(A-BL)x_{t+N|t} &\leq h \\ &\vdots \\ H(A-BL)^{N_c}x_{t+N|t} &\leq h \end{aligned}$$

i.e. that $x_{t+N|t}$ belongs to an invariant constraint of the closed-loop dynamics under the given constraints. Off-line computations are still needed to determine N_c. When we consider reference tracking problems later, the equilibrium point will shift and the terminal set will change. In these situations, despite that N_c also changes when the terminal set changes, it is common to use explicit constraint checking as a heuristic for avoiding on-line recomputation of the terminal set.

At this point, it may be useful to reflect on how the different horizon lengths should be chosen. Intuitively, the total prediction horizon N should be long enough to allow the slowest stable dynamics to settle, or at least for the state vector reach the maximal control invariant set. Since the sampling time depends on the fastest dynamics (to avoid aliasing effects), MPC can be challenging for physical systes with a large difference in the fastest and slowest time constants. The constraint checking horizon N_c should be in the same order as the determinedness index of the invariant set of the Mode 2 linear control law. Since N is determined by the closed-loop settling time for the slowest dynamics and the determinedness index can change when we change the desired equilibrium, setting the horizon lengths becomes an iterative process in practice.

5.4.3 Improved numerical conditioning by scaling and pre-stabilized predictions

We have already touched on numerical conditioning of the optimization problem used to compute the MPC control. Better numerical conditioning tends to lead to both faster and more reliable computations. Two simple techniques for improving the numerical conditioning is to use variable scaling and pre-stabilized predictions.

Variable scaling helps to make sure that key quantities in our prediction model are of the same orders of magnitude. One common technique is to scale states and controls in a way that so that the control signals and sensor values take on values in the interval $[-1, 1]$. In a practical terms, this translates to scaling sensor readings from $\pm y_{\max}$ to ± 1, and scale the computed control actions from ± 1 to the true physical limits $\pm u_{\max}$. Appropriate sample-time selection is also essential to make sure that the system matrix A is neither too close to the identity matrix, nor too large.

When the open-loop dynamics are unstable, the matrices A^k grow rapidly with k and the prediction equations tend to become ill-conditioned if long horizons are used. This problem can be alleviated by use of *pre-stabilized predictions*. We then re-parameterize the controls as

$$u_{t+k|t} = -Lx_{t+k|t} + v_{t+k|t}$$

where $\check{v}_{kt}$ are free variables for $k = 0, \ldots, N-1$ and equal to zero for $k \geq N$. This leads to prediction equations on the form

$$x_{t+k|t} = (A-BL)^k x_{t|t} + \sum_{i=0}^{k-1} (A-BL)^i B v_{t+k-1-i|t}$$

which tend to be much better conditioned, provided that $(A-BL)$ is Schur. To keep down the number of free tuning parameters, it is common to use the infinite-horizon optimal LQR control law to prestabilize the predictions, but any L which gives a well-conditioned optimization problem could in principle be used.

5.5 Reference tracking, disturbance rejection and integral action

We will now demonstrate how to incorporate integral action into an MPC controller. We first study the reference tracking problem without disturbances, to develop a basic understanding of the challenges and limitations. We will then consider the more general problem with constant disturbances at the input and output. Finally, we will discuss a slightly simpler but also more restrictive techniques based on working with output and control increments.

5.5.1 Reference tracking

Let us consider the problem of making the system output follow a constant reference r. Error-free tracking in stationarity requires that there are constant vectors $\bar{u} \in U$ and $\bar{x} \in X$ such that

$$\begin{aligned} \bar{x} &= A\bar{x} + B\bar{u} \\ r &= C\bar{x} \end{aligned}$$

We can write these conditions in matrix form as

$$\begin{aligned} \begin{bmatrix} A-I & B \\ C & 0 \end{bmatrix} \begin{bmatrix} \bar{x} \\ \bar{u} \end{bmatrix} &= \begin{bmatrix} 0 \\ r \end{bmatrix} \\ H_x\bar{x} &\leq h_x \\ H_u\bar{u} &\leq h_u \end{aligned} \tag{5.10}$$

If the reference is fixed for all times, then one could verify if r is feasible by solving

$$\begin{aligned} &\text{minimize} && \bar{x}^T Q_1 \bar{x} + \bar{u}^T Q_2 \bar{u} \\ &\text{subject to} && (5.10) \end{aligned}$$

However, we are interested in considering r as a reference signal which is allowed to shift during the operation of the control system. We will therefore include the target calculation in the MPC planning problem and deal with potential infeasibility of reference request by softening the condition $C\bar{x} = r$ using a slack vector. To obtain error-free tracking we then penalize the difference between the state and its target, $\Delta x_{t+k|t} = x_{t+k|t} + \bar{x}$ and the difference between the control and its target $\Delta u_{t+k|t} = u_{t+k|t} - \bar{u}$. Putting everything together yields the MPC planning problem

$$\begin{aligned} &\text{minimize} && \textstyle\sum_{k=0}^{N-1} \Delta x_{t+k|t}^T Q_1 \Delta x_{t+k|t} + \Delta u_{t+k|t}^T Q_2 \Delta u_{t+k|t} + \Delta x_{t+N|t}^T Q_f \Delta x_{t+N|t} + \varphi(s) \\ &\text{subject to} && \Delta x_{t+k+1|t} = A\Delta x_{t+k|t} + B\Delta u_{t+k|t} && k = 0,\dots,N-1 \\ & && H_x(\Delta x_{t+k|t} + \bar{x}) \leq h_x && k = 0,\dots,N-1 \\ & && H_u(\Delta u_{t+k|t} + \bar{u}) \leq h_u && k = 0,\dots,N-1 \\ & && \Delta x_{t+N|t} + \bar{x} \in X_f \\ & && \begin{bmatrix} A-I & B \\ C & 0 \end{bmatrix} \begin{bmatrix} \bar{x} \\ \bar{u} \end{bmatrix} = \begin{bmatrix} 0 \\ r+s \end{bmatrix} \\ & && \Delta x_{t|t} + \bar{x} = x_t \end{aligned} \tag{5.11}$$

in variables $\{\Delta x_{t+k|t}, \Delta u_{t+k|t}\}, \bar{x}, \bar{u}$ and s. The applied control is

$$u_t = \Delta u_{t|t}^\star + \bar{u}. \tag{5.12}$$

In this formulation X_f is a control invariant set for the dynamics $x_{t+1} = Ax_t + Bu_t$ under the constraints $H_x x_t \leq h_x$ and $H_u u_t \leq h_u$. As we have seen earlier, the computations would be simplified considerably if we could replace this constraint with $\Delta x_{t+N|t} \in \bar{X}_f$ where $\bar{X}_f$ is invariant with respect to the constraints and a specific control law, *e.g.* the infinite-horizon optimal control $\Delta u_{t+k|t} = -L\Delta x_{t+k|t}$. However, since the reference affects the equilibrium point $\bar{x}$, the maximal

invariant set $\bar{X}_f$ will depend on the reference and will have to be re-computed whenever r changes. The simplest solution for avoiding such on-line recomputations is to let $X_f = \{\bar{x}\}$ (i.e. enforcing $\Delta x_{t+N|t} = 0$) or to replace the terminal set constraint by explicit constraint checking over a fixed horizon N_c, as discussed above. The next example illustrates this approach.

■ **Example 5.7** The following model represents the influence of the elevator surface deflection δ on the pitch rate q of an aircraft, see Figure 5.12.

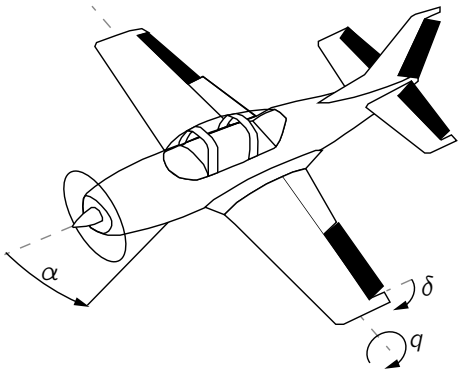

Figure 5.12: The elevator surface deflection δ affects the angle of attack α and the pitch rate q.

$$\begin{bmatrix}\alpha_{k+1}\\ q_{k+1}\end{bmatrix} = \begin{bmatrix}0.9719 & 0.0155\\ 0.2097 & 0.9705\end{bmatrix}\begin{bmatrix}\alpha_k\\ q_k\end{bmatrix} + \begin{bmatrix}0.0071\\ 0.3263\end{bmatrix}\delta_k$$

The control problem is to determine the appropriate elevator surface deflection δ_k for tracking a given reference angle α_r. Both states and the control are subject to upper and lower bounds

$$X = \{(\alpha, q) : -15 \le \alpha \le 30 \wedge -100 \le q \le 100\} \qquad U = \{\delta \ : -25 \le \delta \le 25\}$$

We consider the LQ-cost criterion given by

$$Q_1 = \begin{bmatrix}10 & 0\\ 0 & 1\end{bmatrix}. \quad Q_2 = 1$$

The stationary optimal control is

$$u_t = -L\begin{bmatrix}\alpha_k\\ q_k\end{bmatrix} + l_r\alpha_r = -\begin{bmatrix}1.9603 & 0.8385\end{bmatrix}\begin{bmatrix}\alpha_k\\ q_k\end{bmatrix} + 3.1973\alpha_r$$

with corresponding closed-loop eigenvalues $\lambda_1 = 0.94$ and $\lambda_2 = 0.71$. The closed-loop dynamics converges to within 10% of initial error in $\log(0.1)/\log(0.94) \approx 38$ steps, so we will use a prediction horizon of $N = 40$.

As shown in Figure 5.13, constraint softening allows the MPC controller to act on both feasible and infeasible reference values (*e.g.*, $\alpha_{\min} = -15$). The phase plane plot shown in Figure 5.14(left) gives an alternative perspective of how the MPC controller steers the system state trajectory (blue crosses) to track the given reference (green circles).

We have also illustrated terminal sets corresponding to the closed-loop under the LQ-optimal feedback and the given constraints in Figure 5.14(right). In this example, the reference states inside the constraint set (shown in grey) have a determinedness index of 2 and would thus only require $N_c = 2$. However, the equilibrium corresponding to $\alpha = -15$ lies on the boundary and is (much) more complex to represent with a determinedness index of 38. ■

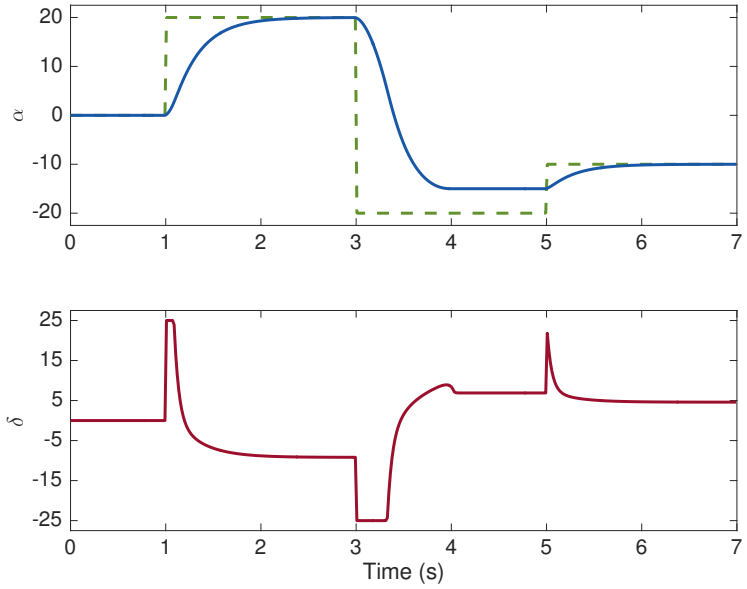

Figure 5.13: Upper plot shows how the controller is able to track feasible references without error and at the same time deal with unfeasible reference requests.

5.5.2 Preview of known future signals

If the reference changes are known in advance, then we can include this information in the MPC planning problem to optimize the state transition even further. Thus, rather than using a constant reference r in (5.11), we supply the known reference trajectory $\{r_k\}$ over the prediction horizon. In the MPC planning problem, we replace the constant target state, controls and slack vectors $(x^{\text{eq}}, u^{\text{eq}}, s)$ by target sequences $(\{x_k^{\text{eq}}\}, \{u_k^{\text{eq}}\}, \{\bar{s}_k\})$ which are allowed to vary over the prediction horizon, but constrained to satisfy (5.10) for all $k = 0, \ldots, N$. The next example illustrate the potential benefits.

■ **Example 5.8** We return to the aircraft problem studied in Example 5.7 and assume that the future reference is known over the prediction horizon. As shown in Figure 5.15, the preview allows the controller to prepare the transition earlier and ensure a faster convergence to the new set-point. ■

5.5.3 Tracking and disturbance rejection using disturbance observers

We will now consider the slightly more complex problem of reference tracking in the presence of disturbances. To this end, we consider the system

$$\begin{aligned} x_{t+1} &= Ax_t + Bu_t + B_d d_t \\ y_t &= Cx_t + C_d d_t \\ e_t &= r - Ey_t \end{aligned} \tag{5.13}$$

Here $d_t \in \mathbb{R}^{n_d}$ is the disturbance vector, $B_d \in \mathbb{R}^{n \times n_d}$ and $C_d \in \mathbb{R}^{p \times n_d}$ are matrices which determine if the disturbance acts on the state evolution or on the measured output. We make a distinction between the output $y_t \in \mathbb{R}^p$ available to the observer and the signals $Ey_t \in \mathbb{R}^{n_r}$ (the *performance outputs*) which we want to track the reference vector $r \in \mathbb{R}^{n_r}$ without stationary error.

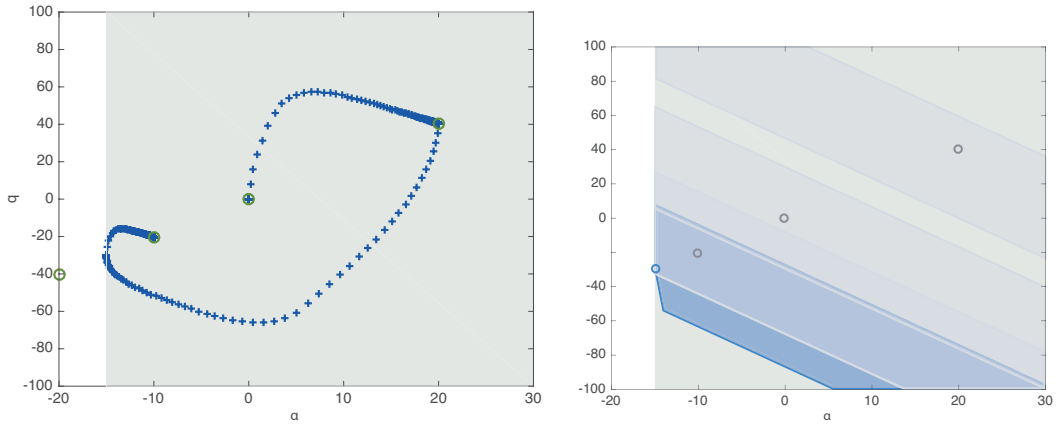

Figure 5.14: The closed-loop trajectories in the phase-plane shows how the system deals with the feasible and infeasible reference requests (left). The right figure shows the invariant sets for the LQ-optimal controller under the given constraints and for the different set-points. The three set-points inside the constraint sets result in invariant sets with simple representations (grey) while the reference on the constraint boundary gives a more complex invariant set (blue).

Compensating for constant disturbances

Assume that the disturbance is constant, $d_t = d$. We can then make a model which describes the evolution of both the system state and the disturbance:

$$\begin{aligned} \begin{bmatrix} x_{t+1} \\ d_{t+1} \end{bmatrix} &= \begin{bmatrix} A & B_d \\ 0 & I \end{bmatrix} \begin{bmatrix} x_t \\ d_t \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} u_t \\ y_t &= \begin{bmatrix} C & C_d \end{bmatrix} \begin{bmatrix} x_t \\ d_t \end{bmatrix} \end{aligned} \tag{5.14}$$

The state vector of this *augmented system* includes both the system state x_t and the disturbance vector d_t. It is natural to ask under what conditions we can estimate the initial states x_0 and d_0 of (5.14) from the output and input sequences $\{y_t\}$ and $\{u_t\}$. The next result answers this question.

Proposition 5.5.1 Assume that the nominal system (A, C) is observable. Then the augmented system is observable if and only if the matrix

$$\begin{bmatrix} A - I & B_d \\ C & C_d \end{bmatrix}$$

has full rank.

Proof. By the PBH test, the augmented system is observable if and only if there is no λ and $v_1 \neq 0$, $v_2 \neq 0$ such that

$$\begin{bmatrix} A & B_d \\ 0 & I \\ C & C_d \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \lambda \begin{bmatrix} v_1 \\ v_2 \\ 0 \end{bmatrix}$$

Since the nominal system is observable, v_2 cannot be zero. This, in turn, implies that we must have $\lambda = 1$ and the PBH test reduces to establishing that the system

$$\begin{bmatrix} A - I & B_d \\ C & C_d \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = 0$$

does not admit any non-zero solutions (v_1, v_2). This is equivalent to the full rank condition posed in the theorem, and the proof is complete. ■

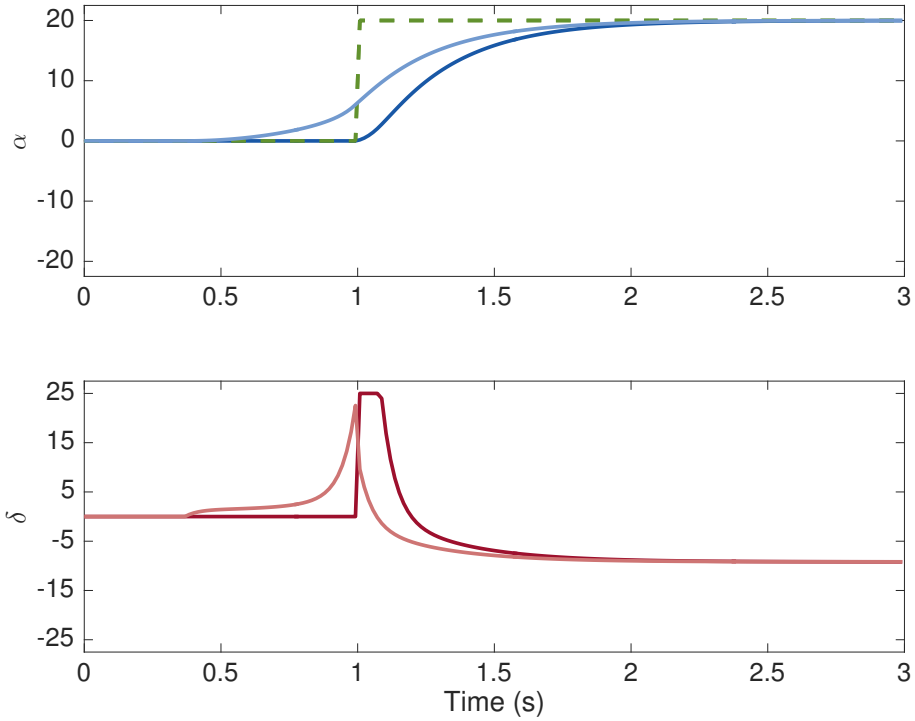

Figure 5.15: By including preview (lighter colors), the controller acts on the reference change ahead of time, ensuring a faster and smoother transition.

If the augmented system is observable, we can attempt to estimate both the plant state x and the disturbance vector d using an observer

$$\begin{aligned} \begin{bmatrix} \hat{x}_{t+1} \\ \hat{d}_{t+1} \end{bmatrix} &= \begin{bmatrix} A & B_d \\ 0 & I \end{bmatrix} \begin{bmatrix} \hat{x}_t \\ \hat{d}_t \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} u_t + \begin{bmatrix} K_x \\ K_d \end{bmatrix} (y_t - \hat{y}_t) \\ \hat{y}_t &= \begin{bmatrix} C & C_d \end{bmatrix} \begin{bmatrix} \hat{x}_t \\ \hat{d}_t \end{bmatrix} \end{aligned} \tag{5.15}$$

For the discussion here, it does not matter how the observer gains are designed (using a Kalman filter approach, pole placement, etc.) as long as its error dynamics are asymptotically stable.

To eliminate the effect of the disturbance in stationarity, we must apply a constant control $\bar{u}$ which drives the system to a steady-state $\bar{x}$ for which $\lim_{t\to\infty} e_t = 0$. Assume that the estimator converges to the stationary disturbance value $\bar{d}$. Then $\bar{x}$ and $\bar{u}$ must satisfy

$$\begin{aligned} \bar{x} &= A\bar{x} + B\bar{u} + B_d\bar{d} \\ r &= EC\bar{x} + EC_d\bar{d} \end{aligned}$$

We re-write these conditions as

$$\begin{bmatrix} A-I & B \\ EC & 0 \end{bmatrix} \begin{bmatrix} \bar{x} \\ \bar{u} \end{bmatrix} = \begin{bmatrix} -B_d\bar{d} \\ r - EC_d\bar{d} \end{bmatrix}$$

and note that we can find $(\bar{x}, \bar{u})$ for every right-hand side if the matrix

$$\begin{bmatrix} A-I & B \\ EC & 0 \end{bmatrix}$$

has full rank. This requires that $m \geq n_r$, so to ensure offset-free tracking we must in general have more control signals than performance outputs that we want to track.

We then propose to use an MPC controller whose objective function penalizes deviations from $\bar{x}$ and $\bar{u}$. The values of the steady-state target state and control are computed from the estimated disturbance under the assumption that the observer has converged, $\hat{d}_t = \bar{d}$:

$$\begin{aligned}
\text{minimize} \quad & \textstyle\sum_{k=0}^{N-1} \Delta x_{t+k|t}^T Q_1 \Delta x_{t+k|t} + \Delta u_{t+k|t}^T Q_2 \Delta u_{t+k|t} + \Delta x_{t+N|t}^T Q_f \Delta x_{t+N|t} + \varphi(s) \\
\text{subject to} \quad & \Delta x_{t+k+1|t} = A\Delta x_{t+k|t} + B\Delta u_{t+k|t} && k = 0,\dots,N-1 \\
& H_x(\Delta x_{t+k|t} + \bar{x}) \le h_x && k = 0,\dots,N-1 \\
& H_u(\Delta u_{t+k|t} + \bar{u}) \le h_u && k = 0,\dots,N-1 \\
& \Delta x_{t+N|t} + \bar{x} \in X_f \\
& \begin{bmatrix} A-I & B \\ EC & 0 \end{bmatrix} \begin{bmatrix} \bar{x} \\ \bar{u} \end{bmatrix} = \begin{bmatrix} -B_d\bar{d} \\ r - EC_d\bar{d} + s \end{bmatrix} \\
& \bar{d} = \hat{d}_t \\
& \Delta x_{t|t} + \bar{x} = x_t
\end{aligned} \tag{5.16}$$

in variables $\{\Delta x_{t+k}, \Delta u_{t+k}\}, \bar{x}, \bar{u}, \bar{d}$ and s. The applied control is

$$u_t = \Delta u_{t|t} + \bar{u} \tag{5.17}$$

This controller has the same issues with the terminal set as the reference tracking MPC, since d and r are unknown at design time. Hence, we cannot easily compute an invariant set off-line, but propose to deal with the terminal set using explicit constraint checking over an additional N_c steps. The next example demonstrates the ideas.

■ **Example 5.9** We return to the aircraft problem from earlier examples, but now with the restriction that we can only measure α. The system is subject to an input disturbance ($B_d = B$, $C_d = 0$). We thus use an observer with the structure

$$\begin{bmatrix} \hat{\alpha}_{k+1} \\ \hat{q}_{k+1} \\ \hat{d}_{k+1} \end{bmatrix} = \begin{bmatrix} 0.9719 & 0.0155 & 0.0071 \\ 0.2097 & 0.9705 & 0.3263 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \hat{\alpha}_k \\ \hat{q}_k \\ \hat{d}_k \end{bmatrix} + \begin{bmatrix} 0.0071 \\ 0.3263 \\ 0 \end{bmatrix} \delta_k + K(\alpha_k - \hat{\alpha}_k).$$

For simplicity, we design the observer gains using pole placement, and set

$$K = \begin{bmatrix} 0.5424 & 6.0523 & 1.4239 \end{bmatrix}^T$$

to place the eigenvalues of $A - KC$ in $\begin{bmatrix} 0.8 & 0.85 & 0.9 \end{bmatrix}$.

We have $n_d = p = m = 1$, and can verify that the rank conditions hold. As shown in Figure 5.16, the controller is able to both track the reference and suppress the disturbance. ■

There are some additional subtleties in ensuring offset-free MPC when the process has multiple inputs and outputs. However, when the number of disturbances equals the number of measured outputs, the following theorem shows that offset-free tracking can be ensured under mild conditions.

Theorem 5.5.2 Consider the discrete-time linear system (5.13) with (A, B) reachable under the constraints $x_t \in X = \{x \mid H_x x \le h_x\}$ and $u_t \in U = \{u \mid H_u u_t \le h_u\}$. Let the control signal be defined by the model predictive controller (5.16) with $s = 0$, $Q_1 \succeq 0$ with $(A, Q_1^{1/2})$ observable, $Q_2 \succ 0$, Q_f and X_f chosen as in Proposition 5.3.3, and $(\hat{x}_t, \hat{d}_t)$ estimated by an asymptotically stable observer on the form (5.15). Let $n_d = p$ and let the matrices

$$\begin{bmatrix} A-I & B_d \\ C & C_d \end{bmatrix} \tag{5.18}$$

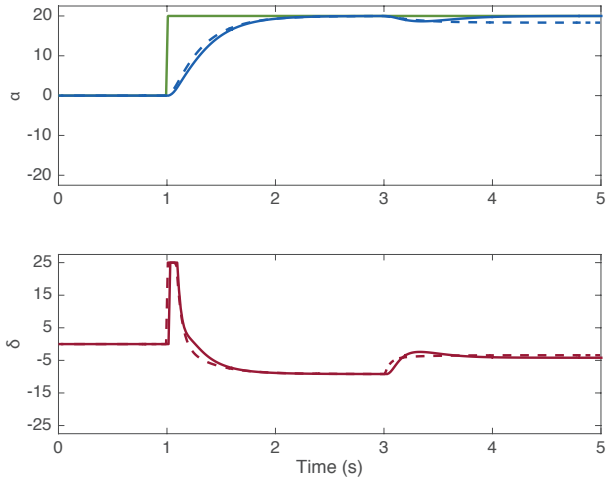

Figure 5.16: MPC controller for the aircraft example. Simulations with (full) and without (dashed) disturbance compensation. Both the reference change at $t = 1$, and the constant input disturbance at $t = 3$ are dealt with without any remaining errors.

and

$$\begin{bmatrix} A-I & B \\ EC & 0 \end{bmatrix} \tag{5.19}$$

have full rank. Assume that the MPC problem is feasible for all times and unconstrained for all $t \geq T$ for some fixed time T. If the closed-loop system reaches a steady-state, then $\lim_{t\to\infty} e_t = 0$.

Proof. If the closed-loop system reaches a steady state $(x^{\text{eq}}, u^{\text{eq}})$, then so does the observer, since its dynamics are asymptotically stable. Moreover, asynmptotic stability of the observer implies that

$$I-(A-KC) = \begin{bmatrix} I-A+K_xC & -B_d+K_xC_d \\ K_dC & K_dC_d \end{bmatrix}$$

has full rank, which necessities that $K_d \in \mathbb{R}^{p\times p}$ has full rank. Let $(\hat{x}^{\text{eq}}, \hat{d}^{\text{eq}})$ be the stationary estimates of the observer. Then

$$K_d(Cx^{\text{eq}} - C\hat{x}^{\text{eq}} - C_d\hat{d}^{\text{eq}}) = 0$$

Since K_d is square and of full rank, this means that $y^{\text{eq}} = C\hat{x}^{\text{eq}} + C_d\hat{d}^{\text{eq}}$ and thus

$$ECx^{\text{eq}} = EC\hat{x}^{\text{eq}} + EC_d\hat{d}^{\text{eq}} = r + EC(\hat{x}^{\text{eq}} - \bar{x}^{\text{eq}})$$

To show that $\hat{x}^{\text{eq}} = \bar{x}$, we combine the estimator steady-state conditions with the target calculation

$$\begin{aligned} \hat{x}^{\text{eq}} - \bar{x} &= A\hat{x}^{\text{eq}} + Bu^{\text{eq}} + B_d\hat{d}^{\text{eq}} - (A\bar{x} + B\bar{u} + B_d\bar{d}) = \\ &= A(\hat{x}^{\text{eq}} - \bar{x}) + B(u^{\text{eq}} - \bar{u}) \end{aligned}$$

Since we have assumed that the MPC controller operates without violating any constraints when $t \geq T$, our choice of terminal penalty implies that

$$u^{\text{eq}} - \bar{u} = -L(\hat{x}^{\text{eq}} - \bar{x})$$

It must therefore hold that

$$(I - A + BL)(\hat{x}^{\text{eq}} - \bar{x}) = 0$$

Under the given assumptions, the infinite-horizon LQR closed-loop is asymptotically stable, so the matrix $I - A + BL$ is invertible and the only solution is $\hat{x}^{\text{eq}} = \bar{x}$. Hence, $Ey^{\text{eq}} = r$ and the proof is complete. □

Note that the theorem above does not assume that the model used in the controller and the actual system dynamics match. Hence, when we design the model used in the disturbance observer and the MPC predictions, the disturbances do not need to correspond to physical signals acting on the system. Rather, they can be seen as a means for introducing integral action and ensuring offset-free tracking. If there is only a few real disturbance signals, one may need to introduce artificial disturbances to meet the requirement that $n_d = p$. These artificial disturbances should then be introduced so that the corresponding B_d and C_d make the matrix (5.18) have full rank.

We conclude this section by an example which illustrates some of subtleties of offset-free MPC.

■ **Example 5.10** Consider the following second-order system

$$\begin{aligned} x_{t+1} &= \begin{bmatrix} 0.95 & -0.2 \\ 0.2 & 0.95 \end{bmatrix} x_t + \begin{bmatrix} 0 \\ -0.2125 \end{bmatrix} (u_t + d_t) \\ y_t &= \begin{bmatrix} 1 & 0 \end{bmatrix} x_t \end{aligned}$$

To account for the disturbance, we form the augmented model

$$\begin{aligned} \begin{bmatrix} x_{t+1} \\ d_{t+1} \end{bmatrix} &= \begin{bmatrix} A & B \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_t \\ d_t \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} u_t \\ y_t &= Cx_t \end{aligned} \tag{5.20}$$

design a disturbance observer and use the offset-free MPC formulation in (5.16). We simulate the closed loop under a unit reference change at $t = 0$ and a unit disturbance at $t = 100$. As shown in Figure 5.17, the controller tracks the reference without offset, also in the presence of a disturbance.

In practice, it is often difficult to make a model that matches the actual system behavior. We therefore simulate the effect of using the augmented model (5.20) for control design when the actual system behaves as

$$x_{t+1} = Ax_t + Bu_t + B_p d_t,$$

with $B_p \neq B$. Specifically, we use

$$B_p = \begin{bmatrix} 0.2125 \\ 0 \end{bmatrix}$$

Figure 5.17 shows that the controller can ensure offset-free tracking despite the modeling mismatch. This is in line with our discussion above: Theorem 5.5.2 does not require that the models match, only that an unconstrained equilibrium is attained in the closed-loop.

Finally, we add an extra sensor to the system and let the estimator have access to measurements of both x_1 and x_2. Somewhat surprisingly, the simulations in Figure 5.17 show that the controller no longer manages to ensure offset-free tracking. To understand this behaviour, consider the system

$$\begin{aligned} x_{t+1} &= Ax_t + Bu_t + w \\ y_t &= Cx_t \end{aligned}$$

with $w \in \mathbb{R}^{n \times n_d}$. We design an observer under the assumption that the disturbance is on the form $w = B_d d$ for some constant vector $d \in \mathbb{R}^{n_d}$, *i.e.* use the estimator

$$\begin{aligned}\hat{x}_{t+1} &= A\hat{x}_t + Bu_t + B_d\hat{d}_t + K_x(y_t - \hat{y}_t)\\ \hat{d}_{t+1} &= \hat{d}_t + K_d(y_t - \hat{y}_t)\\ \hat{y}_t &= C\hat{x}_t\end{aligned}$$

Assume that the estimator has converged to a value $\hat{d}^{\text{eq}}$ with $B_d\hat{d}^{\text{eq}} \neq w$. Define the state estimation error as $\varepsilon_t = x_t - \hat{x}_t$. This error evolves as

$$\varepsilon_{t+1} = (A - K_x C)\varepsilon_t + w - B_d\hat{d}^{\text{eq}}$$

with steady-state

$$\varepsilon^{\text{eq}} = (I - (A - K_x C))^{-1}(w - B_d\hat{d}^{\text{eq}})$$

Hence, the estimated plant states will not tend to the true state values, but exhibit a bias. If

$$K_d(y^{\text{eq}} - \hat{y}^{\text{eq}}) = K_d C\varepsilon^{\text{eq}} = 0 \tag{5.21}$$

then the estimated disturbances will remain at their erroneous estimate $\hat{d}^{\text{eq}}$. Since $K_d \in \mathbb{R}^{n_d \times p}$, this condition can be satisfied for a non-zero ε^{eq} if $p > n_d$. It is the biases in the state and disturbance estimates that cause the stationary tracking errors. In Theorem 5.5.2, we avoid this subtlety by

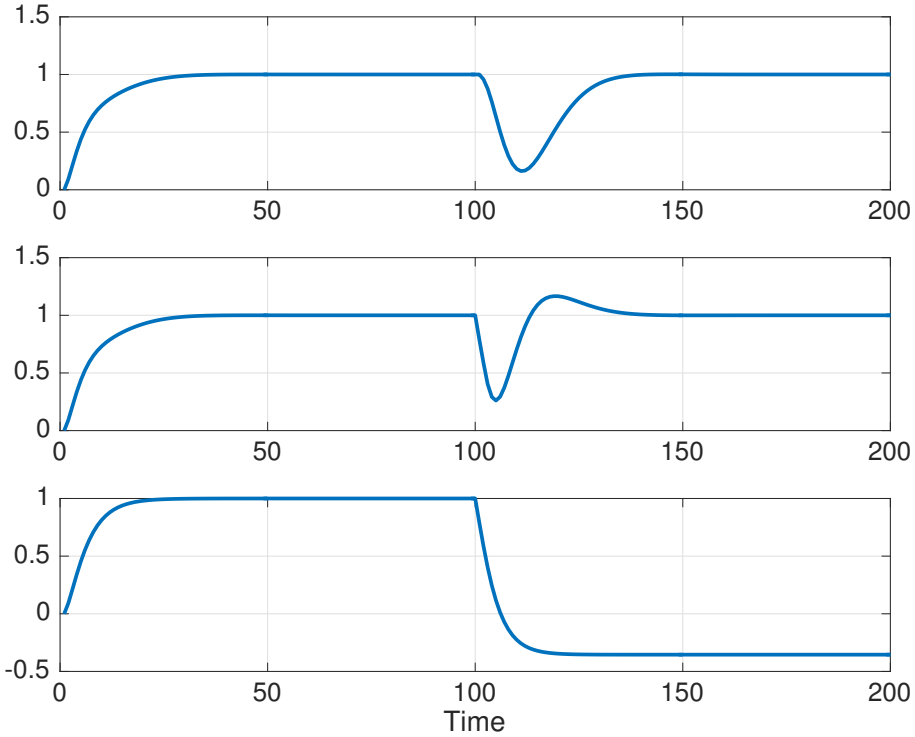

Figure 5.17: Offset-free tracking with perfect match between model and system (top). Even when there is a mismatch between the actual disturbance and its model, the controller manages to track the reference without stationary offset (middle). However, when adding one extra sensor, the performance degrades considerably and the proposed controller can not achieve offset-free tracking.

forcing $n_d = p$, in which case (5.21) can only be satisfied for $\varepsilon^{\text{eq}} = 0$. To demonstrate this effect, we somwhat arbitrarily let

$$B_d = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \qquad C_d = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}.$$

Although this disturbance model also has a mis-match with the actual system dynamics, it has $n_d = p$ and satisfies the rank conditions in Theorem 5.5.2. As the simulations in Figure 5.18 show, the controller can once again track the reference signal without any remaining errors. ∎

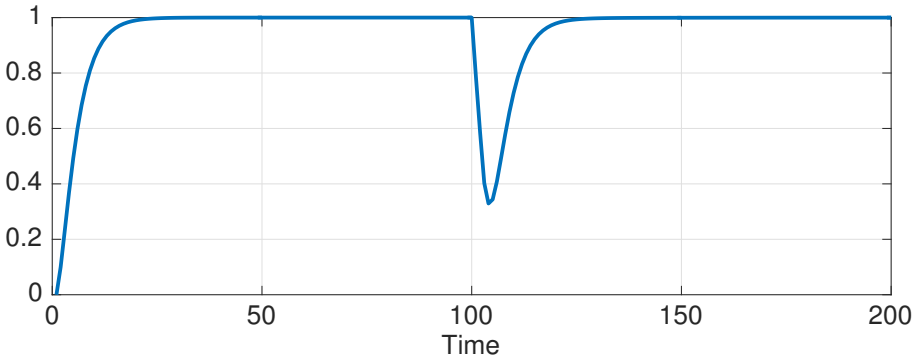

Figure 5.18: The controller using two disturbance states and two measurements achieves offset-free performance, despite being subject to a disturbance that is not perfectly represented by the model.