C21 Model Predictive Control

4 Lectures Michaelmas Term 2018 Mark Cannon
1 Example class mark.cannon@eng.ox.ac.uk

Contents

1 Introduction **2**
1.1 Objectives . . . 3
1.2 Books . . . 3
1.3 Predictive control strategy . . . 4
1.3.1 Prediction . . . 4
1.3.2 Optimization . . . 5
1.3.3 Receding horizon implementation . . . 5
1.3.4 Historical development . . . 6
1.4 Prediction model . . . 7
1.4.1 Linear plant model . . . 8
1.4.2 Nonlinear plant model . . . 9
1.4.3 Discrete and continuous-time prediction models . . . 9
1.5 Constraint handling . . . 10
1.5.1 De-tuned optimal control . . . 13
1.5.2 Anti-windup strategies . . . 13
1.5.3 Model predictive control . . . 14

2 Prediction and optimization 预测和优化 **16**
2.1 Prediction equations . . . 16
2.1.1 LTV prediction models . . . 18
2.2 Unconstrained optimization . . . 18
2.2.1 Horizon length and performance . . . 20

2.3 Infinite horizon cost . . . 21
2.3.1 The relationship between unconstrained MPC and LQ-optimal control . . . 25
2.4 Incorporating constraints . . . 26
2.5 Quadratic Programming . . . 27
2.5.1 Active set algorithms . . . 28
2.5.2 Interior point QP algorithms . . . 29

3 Closed-loop properties of MPC 闭环性质 **34**
3.1 Lyapunov stability analysis . . . 35
3.2 Terminal constraints for recursive feasibility . . . 41
3.3 Constraint checking horizon . . . 44
3.4 Closed-loop performance . . . 49

4 Disturbances and robustness 微扰强健性，sensibility **54**
4.1 MPC with integral action . . . 56
4.2 Robustness to constant disturbances . . . 58
4.2.1 Robust constraint satisfaction . . . 58
4.2.2 Pre-stabilized predictions . . . 59
4.2.3 Tubes and time-varying disturbances . . . 61

1 Introduction HMM??

Model Predictive Control (MPC) is an optimal control strategy based on numerical optimization. Future control inputs and future plant responses are predicted using a system model and optimized at regular intervals with respect to a performance index. From its origins as a computational technique for improving control performance in applications within the process and petrochemical industries, predictive control has become arguably the most widespread advanced control methodology currently in use in industry. MPC has a sound theoretical basis and its stability, optimality, and robustness properties are well understood.

Despite being very simple to design and implement, MPC algorithms can control large scale systems with many control variables, and, most importantly, MPC provides a systematic method of dealing with constraints on inputs and states. Such constraints are present in all control engineering applications and represent limitations on actuators and plant states arising from physical, economic, or safety constraints. In MPC these constraints are accounted for explicitly by solving a constrained optimization problem in real-time to determine the optimal predicted inputs. Nonlinear plant dynamics can be similarly incorporated in the prediction model.

This course covers the basic principles of model predictive control, considering its theoretical properties and implementation issues. The main emphasis of the course is on the design of cost and constraints and analysis of closed-loop properties. Connections with unconstrained optimal control, Lyapunov stability theory and optimization problems are investigated. Extensions relating to set point tracking, robustness to disturbances and nonlinear dynamics are considered.

Please send corrections or suggestions to `mark.cannon@eng.ox.ac.uk`

Web pages: `weblearn.ox.ac.uk`
`markcannon.github.io/teaching`

1.1 Objectives 滚动域控制（receding horizon control） = MPC

At the end of the course you should be able to do the following:

- Understand the advantages of receding horizon control, its limitations and areas of application.
- Know how to formulate receding horizon control as:
  (a). fixed-term feedback controller (for unconstrained linear systems);
  (b). a quadratic program (for linearly constrained linear systems);
  (c). a nonlinear program (general case).
- Understand and know how to determine the stability properties of a predictive controller in terms of:
  (a). recursive feasibility guarantees;
  (b). monotonic cost decrease through optimization.
- Know how to design terminal constraints through a constraint checking horizon.
- Know how to incorporate integral action.
- Know how to ensure robustness to bounded disturbances.

1.2 Books

1. Kouvaritakis, B. and Cannon, M. *Model Predictive Control: Classical, Robust and Stochastic*. Springer, 2015.

   Recommended reading: Chapters 1, 2 and 3.

2. Rawlings, J.B. and Mayne, D.Q. *Model Predictive Control: Theory and Design*. Nob Hill Publishing, 2009.

3. Maciejowski, J.M.. *Predictive control with constraints*. Prentice Hall, 2002.

   Recommended reading: Chapters 1-3, 6 and 8.

1.3 Predictive control strategy [1]

A model predictive control law contains the basic components of prediction, optimization and receding horizon implementation. A summary of each of these ingredients is given below.

1.3.1 Prediction

The future response of the controlled plant is predicted using a dynamic model. This course is concerned mainly with the case of discrete-time linear systems with state-space representation 状态–空间表示

$$x_{k+1} = Ax_k + Bu_k, \tag{1.1}$$

where x_k and u_k are the model state and input vectors at the kth sampling instant. Given a predicted input sequence, the corresponding sequence of state predictions is generated by simulating the model forward over the prediction horizon, of say N sampling intervals. For notational convenience, these predicted sequences are often stacked into vectors $\mathbf{u}, \mathbf{x}$ defined by

$$\mathbf{u}_k = \begin{bmatrix} u_{0|k} \\ u_{1|k} \\ \vdots \\ u_{N-1|k} \end{bmatrix}, \qquad \mathbf{x}_k = \begin{bmatrix} x_{1|k} \\ x_{2|k} \\ \vdots \\ x_{N|k} \end{bmatrix} \tag{1.2}$$

Here $u_{i|k}$ and $x_{i|k}$ denote input and state vectors at time $k+i$ that are predicted at time k (i.e. predictions of their values i steps ahead), and $x_{i|k}$ is governed by the prediction model:

$$x_{i+1|k} = Ax_{i|k} + Bu_{i|k}, \quad i = 0, 1, \ldots \tag{1.3}$$

with initial condition (at the start of the prediction horizon) defined by

$$x_{0|k} = x_k.$$

[1] Reading: *Kouvaritakis & Cannon* Chap.1 and §2.1 or *Maciejowski* Chap.1.

1.3.2 Optimization

The predictive control feedback law is computed by minimizing a predicted performance cost, which is defined in terms of the predicted sequences $\mathbf{u}, \mathbf{x}$. This course is mainly concerned with the case of quadratic cost, for which the predicted cost has the general form:

$$J(x_k, \mathbf{u}_k) = \sum_{i=0}^{N} \Big(\|x_{i|k}\|_Q^2 + \|u_{i|k}\|_R^2 \Big) \tag{1.4}$$

where $\|x\|_Q^2 = x^\top Q x$ and Q, R are positive definite matrices (Q may be positive semi-definite). The optimal control sequence for the problem of minimizing the predicted cost is denoted $\mathbf{u}_k^*(x_k)$, and the optimal value of the cost is $J^*(x_k) = J(x_k, \mathbf{u}_k^*)$, which is often written as J_k^* for simplicity:

$$\mathbf{u}_k^*(x_k) = \arg\min_{\mathbf{u}_k} J(x_k, \mathbf{u}_k)$$
$$J_k^* = J^*(x_k) = J(x_k, \mathbf{u}_k^*) = \min_{\mathbf{u}_k} J(x_k, \mathbf{u}_k).$$

If the system is subject to input and state constraints, then these could be included in the optimization as constraints on $\mathbf{u}_k$.

1.3.3 Receding horizon implementation

Only the first element of the optimal predicted input sequence $\mathbf{u}_k^*$ is input to the plant:

$$u_k = u_{0|k}^*. \tag{1.5}$$

The process of computing $\mathbf{u}_k^*$ by minimizing the predicted cost and implementing the first element of $\mathbf{u}_k^*$ is then repeated at each sampling instant $k = 0, 1, \ldots$. For this reason the optimization defining $\mathbf{u}_k^*$ is known as an **online** optimization.

The prediction horizon remains the same length despite the repetition of the optimization at future time instants (Fig. 1), and the approach is therefore known as a **receding horizon** strategy.

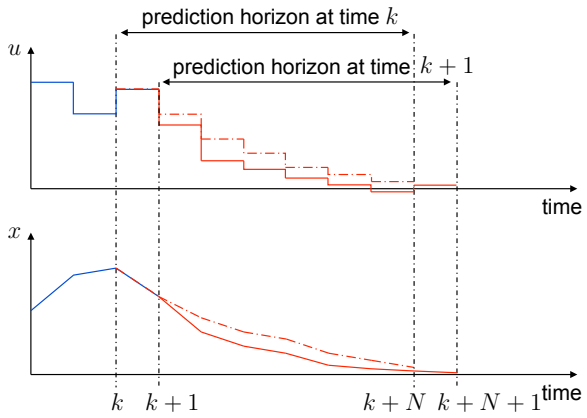


Figure 1: The receding horizon strategy

Since the state predictions $\mathbf{x}_k$ and hence the optimal input sequence $\mathbf{u}_k^*$ depend on the current state measurement x_k, this procedure introduces feedback into the MPC law, thus providing a degree of robustness to modelling errors and uncertainty. A second feature of the receding horizon approach is that, by continually shifting the horizon over which future inputs are optimized, it attempts to compensate for the fact that this horizon is finite. This can be seen in Fig. 2, which illustrates that the predicted response may differ significantly from the closed-loop response. Provided the cost and constraints are designed correctly, a receding horizon strategy can ensure that the performance of the closed-loop system is at least as good as that of the optimal prediction.

1.3.4 Historical development

The MPC strategy has been discovered and re-invented several times. Receding horizon approaches were used in the 1960s and 70s to define computational methods for optimal control problems that have no closed-form solution. Predictive control reappeared in the entirely different context of industrial process control in the 1980s as a means of exploiting continual improvements in computational resources to improve performance. More recently the approach has

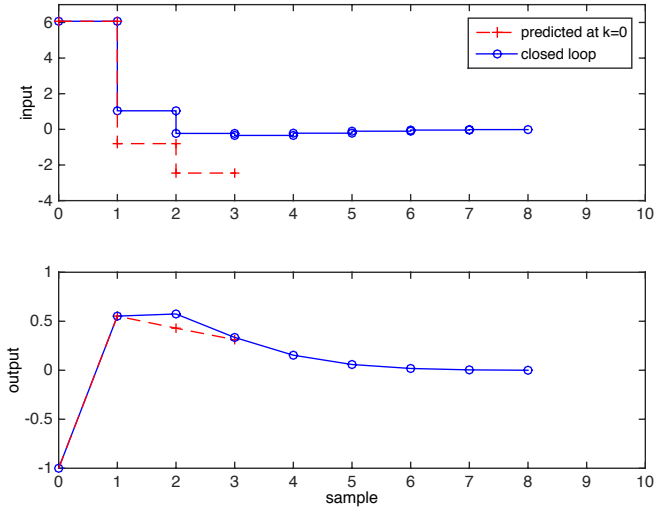


Figure 2: Closed-loop responses and responses predicted at $k = 0$ for an unconstrained 2nd order system with horizon $N = 3$.

been used as a general technique for deriving stabilizing controllers for constrained systems. At the same time, the availability of faster computers and improvements in computational efficiency of predictive controllers (including nonlinear and robust MPC schemes) have extended its range of applications to include fast sampling systems.

1.4 Prediction model

A very wide class of plant model can be incorporated in a predictive control strategy. This includes linear, nonlinear, discrete and continuous-time models. Prediction models may be deterministic, stochastic, or fuzzy.

1.4.1 Linear plant model

plant model

For linear systems, the dependence of predictions $\mathbf{x}_k$ on $\mathbf{u}_k$ is linear. A quadratic predicted cost such as (1.4) is therefore a quadratic function of the input sequence $\mathbf{u}_k$. Thus J can be expressed as a function of $\mathbf{u}$ in the form

$$J(x_k, \mathbf{u}_k) = \mathbf{u}_k^\top H \mathbf{u}_k + 2f^\top \mathbf{u}_k + g \tag{1.6}$$

where H is a constant positive definite (or possibly positive semidefinite) matrix, and $f = f(x_k)$ and $g = g(x_k)$ are respectively a vector and a scalar which depend on x_k. Linear input and state constraints likewise imply linear constraints on $\mathbf{u}_k$ which can be expressed

$$A_c \mathbf{u}_k \leq b_c \tag{1.7}$$

where A_c is a constant matrix and the vector $b_c = b_c(x_k)$ is in general a function of x_k. The online MPC optimization therefore comprises the minimization over $\mathbf{u}$ of a quadratic objective subject to linear constraints:

$$\begin{aligned} \underset{\mathbf{u}}{\text{minimize}} \quad & \mathbf{u}^\top H \mathbf{u} + 2f^\top \mathbf{u} && \text{(1.8a)} \\ \text{subject to} \quad & A_c \mathbf{u} \leq b_c && \text{(1.8b)} \end{aligned}$$

This class of optimization problem is known as a quadratic programming (QP) problem, and given that H is a positive definite matrix and the constraints are linear, it is easy to show that (1.8a,b) is a convex problem (i.e. both the objective (1.8a) and constraints (1.8b) are convex functions of the optimization variable $\mathbf{u}$). It can be solved efficiently and reliably using specialized algorithms. For example Matlab's QP solver (quadprog) running on a 2.3 GHz quadcore processor with 16 GB memory typically needs around 2 ms to solve problems involving 10 variables and 100 constraints. Commercial solvers that are optimized for speed can provide typical solution times for this problem of the order of 1 ms.

1.4.2 Nonlinear plant model

If a nonlinear prediction model is employed, then due to the nonlinear dependence of the state predictions $\mathbf{x}_k$ on $\mathbf{u}_k$, the MPC optimization problem is significantly harder than for the linear model case. This is because the cost in equation (1.4), which can be written as $J(\mathbf{u}_k, x_k)$, and the constraints, $g(\mathbf{u}_k, x_k) \leq 0$, are in general nonconvex functions of $\mathbf{u}_k$, so that the optimization problem:

$$\begin{aligned} \underset{\mathbf{u}}{\text{minimize}} \quad & J(x_k, \mathbf{u}) && \text{(1.9a)} \\ \text{subject to} \quad & g(x_k, \mathbf{u}) \leq 0 && \text{(1.9b)} \end{aligned}$$

becomes a nonconvex nonlinear programming (NLP) problem.

As a result there will in general be no guarantee that a solver will converge to a global minimum of (1.9), and the times required to find even a local solution are typically orders of magnitude greater than for QP problems of similar size. For example, to solve the MPC optimization derived from an inverted pendulum control problem with 10 variables in $\mathbf{u}$, solution times of 10 sec are typically needed on a 2GHz PC. Unlike QP solvers, the computational loads of solvers for nonlinear programming problems are strongly problem-dependent.

1.4.3 Discrete and continuous-time prediction models

In a typical MPC implementation, the online optimization is solved periodically at times $t = kT$, $k = 0, 1, \ldots$, and for each k, the control law $u = u^*_{0|k}$ is implemented until the solution of the optimization at $t = (k+1)T$ becomes available. Clearly therefore, T has to be at least as large as the computation time required to perform the online optimization, and ideally T should be very much larger than this if the computation delay is not accounted for explicitly in predictions. However this constraint does not apply to the choice of the sampling interval for a discrete-time prediction model, which could be dictated by other considerations (e.g. the bandwidth of the plant or disturbance signals). In fact it is possible to use a model sampling interval, T_{samp}, which is smaller

than T provided that:

$$T_{samp} = T/m, \quad \text{for integer } m.$$

This condition makes it possible for the previously computed optimal predicted input sequence to be used at current time if necessary (see Fig. 3), which in turn allows for a guarantee of stability and convergence of the MPC strategy, as will be discussed in Section 3. For simplicity a discrete-time prediction model with $T_{samp} = T$ will be assumed in these notes.

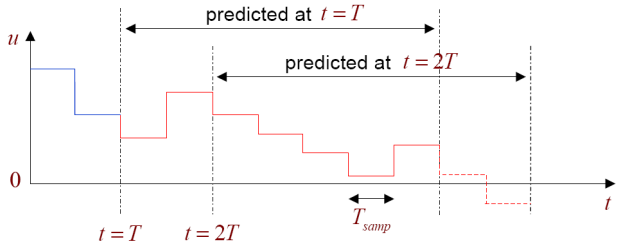


Figure 3: Input predictions at successive optimization times for the case that $T = 2T_{samp}$.

Note that it is also possible to use continuous-time prediction models in MPC provided that, when performing the optimization to determine the current optimal trajectory, it is always possible to implement the previously computed optimal trajectory. The use of continuous-time prediction models is desirable when the plant dynamics have no closed-form discrete-time representation, which is often the case for nonlinear systems with phenomenological (i.e. physics-based) models. Specialized algorithms incorporating numerical integration of continuous-time prediction models are available for solving the online optimization.

1.5 Constraint handling 约束条件

In addition to the obvious *equality constraints* that the state and input should satisfy the model dynamics, *inequality constraints* on input and state variables

are encountered in every control problem. While the equality constraints are usually handled implicitly (i.e. the plant model is used to write predicted state trajectories as functions of initial conditions and input trajectories), the inequality constraints are imposed as explicit constraints within the online optimization problem. This course is concerned with linear inequality constraints.

Input constraints. These can have the form of **absolute** constraints, i.e.

$$\underline{u} \le u_k \le \overline{u} \tag{1.10}$$

or **rate** constraints:

$$\underline{\triangle u} \le u_k - u_{k-1} \le \overline{\triangle u}. \tag{1.11}$$

Input constraints commonly arise as a result of actuator limits, e.g. torque saturation in d.c. motors and flow saturation in valves.

State constraints. Linear state constraints have the general form

$$\underline{g_c} \le G_c x_k \le \overline{g_c} \tag{1.12}$$

where G_c is a constant matrix and $\underline{g_c}, \overline{g_c}$ are constant vectors. State constraints may be active during transients, e.g. aircraft stall speed, or in steady-state operation as a result of e.g. economic constraints on process operation.

Hard/soft constraints. Constraints are classified as either **hard** or **soft**. Hard constraints must always be satisfied, and if this is not possible the problem is **infeasible**. On the other hand, soft constraints may be violated if necessary to avoid infeasibility. This course will only consider hard constraints.

We next give brief descriptions of several common methods of handling input constraints. It is possible to enforce satisfaction of input constraints simply by saturating the unconstrained controller u^{free}, e.g. by defining

$$u_k = \begin{cases} \min\{u^{free}, \overline{u}\} & \text{if } u^{free} \ge 0 \\ \max\{u^{free}, \underline{u}\} & \text{if } u^{free} < 0 \end{cases}$$

However, if constraints are ignored in the design of u^{free}, then the saturated control law is likely to give poor closed-loop responses and may lead to instability. For example, Figure 4 shows the characteristically slow and oscillatory

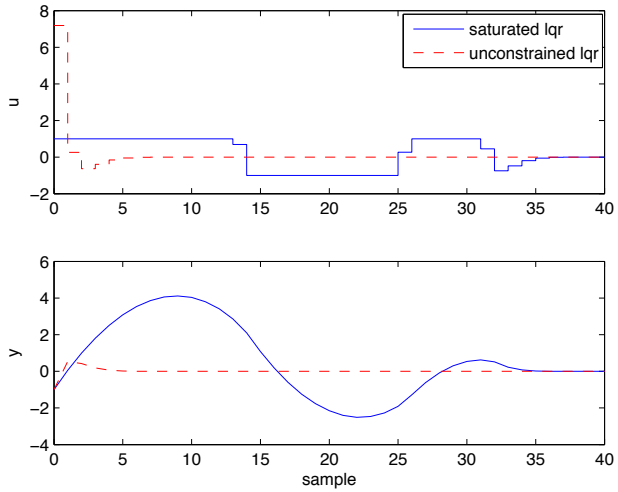


Figure 4: Step response of saturated linear controller (LQ-optimal feedback).

step responses of linear system to a saturated linear feedback law, which was designed to be LQ-optimal for the unconstrained system. In this example the system has model and constraints:

$$x_{k+1} = \begin{bmatrix} 1.1 & 2 \\ 0 & 0.95 \end{bmatrix} x_k + \begin{bmatrix} 0 \\ 0.0787 \end{bmatrix} u_k, \quad y_k = \begin{bmatrix} -1 & 1 \end{bmatrix} x_k$$
$$-1 \leq u_k \leq 1$$

The performance index

$$\sum_{k=0}^{\infty} [y_k^2 + R u_k^2], \quad R = 0.01.$$

therefore gives the unconstrained LQ-optimal controller as:

$$u_k = K x_k, \quad K = \begin{bmatrix} -4.3608 & -18.7401 \end{bmatrix}$$

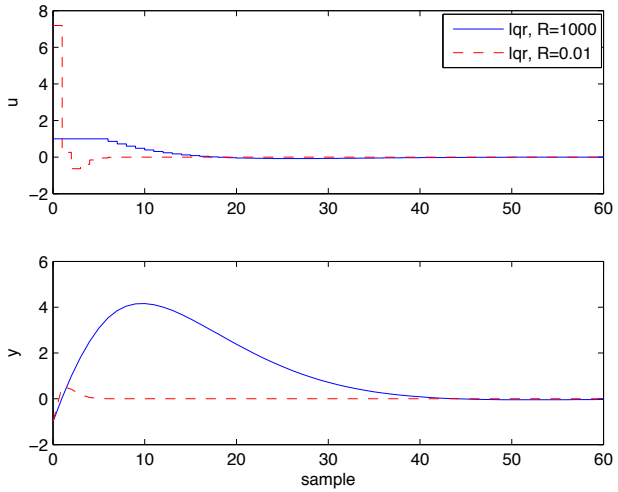


Figure 5: Step response of detuned LQ-optimal feedback law.

1.5.1 De-tuned optimal control

It may be possible to account for input constraints by increasing the input weighting R in the LQ performance cost until the associated optimal feedback law satisfies constraints in the desired operating region. This approach is clearly suboptimal in terms of the original performance index and can lead to very slow closed-loop responses. For example Fig. 5 shows the step response of the system of Fig. 4 when R is chosen to be large enough ($R = 1000$) that constraints are satisfied by the unit step response.

1.5.2 Anti-windup strategies

Anti-windup methods aim to prevent the instability that can occur in controllers which incorporate integral action when input constraints are active. An example is the anti-windup PI controller shown in Figure 6. To avoid the degradation of performance that results when the integrated error term be-

comes large while the input is saturated, this uses a modified control law:

$$u = \mathrm{sat}(Ke + v),$$
$$T_i \dot{v} + v = u.$$

[Here the saturation block ensures that $u(t)$ remains at all times within limits, and it is clear from the transfer function from e to u in Fig. 6 that

$$u(t) = K\left(e(t) + \frac{1}{T_i}\int^t e\, dt'\right)$$

whenever $\underline{u} < u(t) < \overline{u}$. On the other hand, if u lies on either its upper or lower limit, then $v(t)$ converges (exponentially) to $\underline{u}$ or $\overline{u}$, and $u(t)$ therefore becomes unsaturated quickly after a subsequent change in the sign of $e(t)$. Although relatively easy to implement, this approach does not have an obvious extension to systems with more than one input and output, and provides no guarantees of stability or performance.]

1.5.3 Model predictive control

Applying predictive control to the example used to generate Figures 4 and 5 leads to the responses shown in Figure 7. Although the MPC optimization employs a finite number of degrees of freedom in predicted input sequences, in this case the resulting control law is optimal for the infinite horizon problem (this will be shown in Section 3). Of course the price of optimality is the online solution of a QP, which constitutes an enormous increase in computation relative to simple anti-windup or saturation approaches.

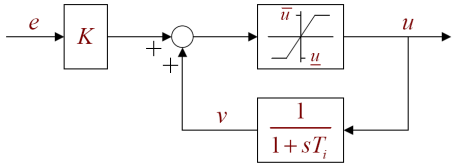


Figure 6: Block diagram of anti-windup proportional + integral controller.

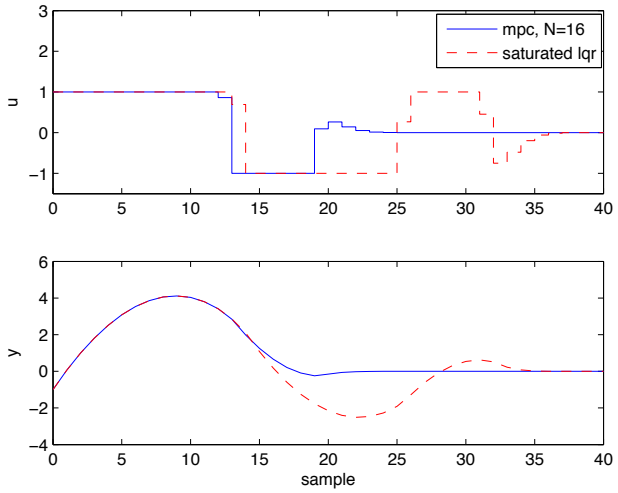
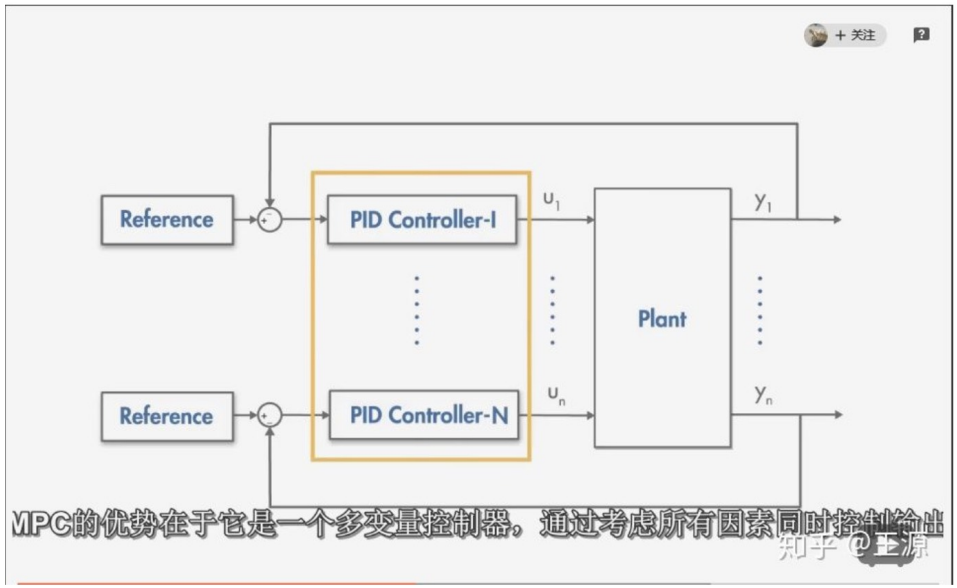


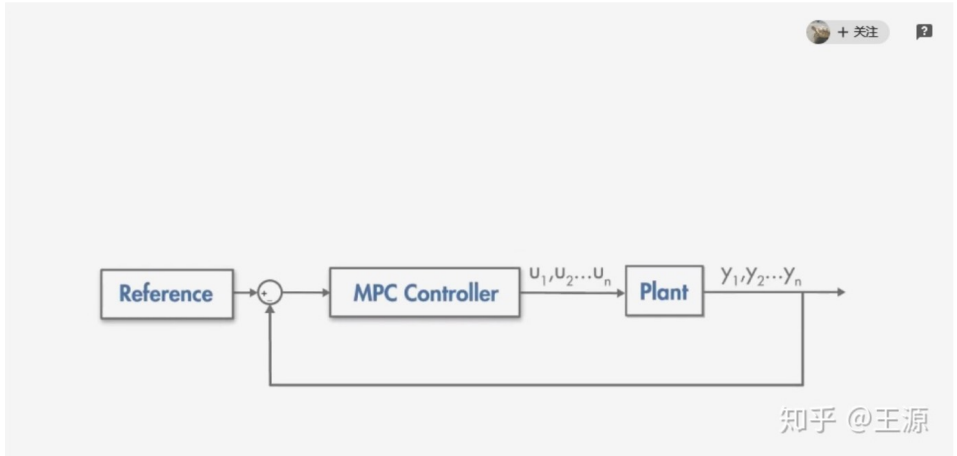
Figure 7: Step response of constrained MPC law.

Summary of Section 1

- MPC is a feedback law based on prediction, optimization, and receding horizon implementation. The optimization is performed over open-loop predictions, which are based on a plant model.
- Constraints on system inputs and states can be hard or soft. Constraints are handled sub-optimally using de-tuned optimal control or anti-windup strategies whereas MPC enables constraints to be handled optimally.

模型预测控制善于处理多输入多输出系统，对于如下图所示的多输入多输出系统，虽然也可以采用多个PID控制器进行控制，但是往往由于各个控制回路之间存在耦合关系，其PID参数的整定变得非常困难。与之相对的模型预测控制的优势就是非常方便的可以考虑多输入多输出系统，用一个MPC控制器就可以解决.

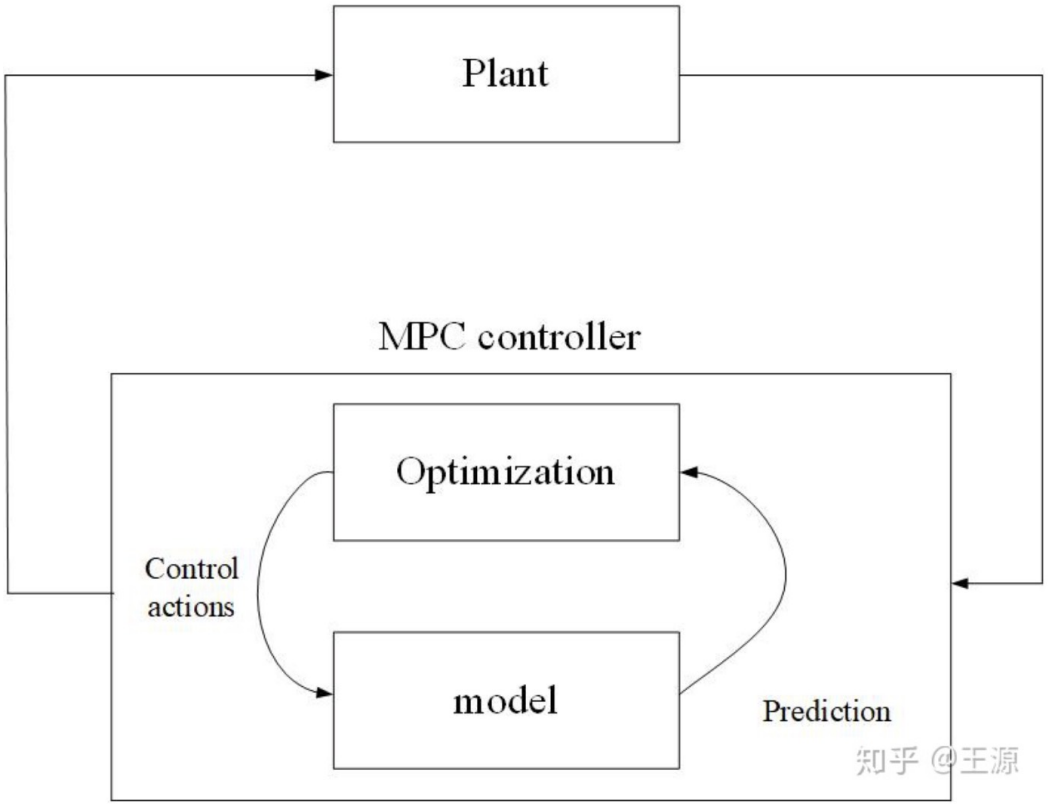




模型预测控制可以处理约束:
现实世界中往往存在各种各样的约束条件，例如前面举得开车的例子中，对车速有约束（因为有交通法规限速），对加速度也有约束（因为发动机的负载是有限的）。由于模型预测控制是通过构建优化问题来求解控制器的动作的，所以可以非常自然的将这些约束建立在优化问题中以此来保证这些约束的满足。

模型预测控制是有向前考虑未来时间步的有限时域优化

模型预测控制的实现是通过建立优化问题的模型，求解该优化问题来得到控制器的输出的。如下图所示：



Introduction about MPC: https://zhuanlan.zhihu.com/p/99409532

2 Prediction and optimization 预测和优化

2.1 Prediction equations [2]

This section derives an expression for the predicted state trajectory $\mathbf{x}_k$ in terms of the predicted input trajectory $\mathbf{u}_k$. This expression allows the general quadratic cost:

$$J(x_k, \mathbf{u}_k) = \sum_{i=0}^{N-1} \left(\|x_{i|k}\|_Q^2 + \|u_{i|k}\|_R^2 \right) + \|x_{N|k}\|_{\overline{Q}}^2 \tag{2.1}$$

to be written as a quadratic function of $\mathbf{u}_k$ in the form (1.6). Note that $u_{N|k}$ is omitted from (2.1) since no other terms in the cost, J, depend on $u_{N|k}$. Furthermore it is possible (and in fact desirable) to use a different weighting matrix $\overline{Q}$ in the term involving the terminal predicted state $x_{N|k}$. By choosing an appropriate value for $\overline{Q}$, this allows the cost over an infinite prediction to be accounted for in J (under certain assumptions on $u_{i|k}$, $i = N, N+1, \ldots$), as we discuss in Section 2.3.

The predicted state sequence generated by the linear state-space model (1.3) with input sequence $\mathbf{u}_k$ can be written

$$\begin{aligned} x_{0|k} &= x_k \\ x_{1|k} &= Ax_k + Bu_{0|k} \\ x_{2|k} &= A^2 x_k + ABu_{0|k} + Bu(k+1|k) \\ &\vdots \end{aligned}$$

In compact notation:

$$x_{i|k} = A^i x_k + \mathcal{C}_i \mathbf{u}_k, \quad i = 0, \ldots, N$$

$$\text{or} \quad \mathbf{x}_k = \mathcal{M} x_k + \mathcal{C} \mathbf{u}_k, \quad \text{where } \mathcal{M} = \begin{bmatrix} A \\ A^2 \\ \vdots \\ A^N \end{bmatrix}. \tag{2.2}$$

[2] Reading: *Kouvaritakis & Cannon* §2.3, *Maciejowski* §2.6.

Here $\mathcal{C}$ is the (convolution) matrix with rows $\mathcal{C}_i$ defined by

$$\mathcal{C} = \begin{bmatrix} B & 0 & \cdots & 0 \\ AB & B & \cdots & 0 \\ \vdots & \vdots & \ddots & \\ A^{N-1}B & A^{N-2}B & \cdots & B \end{bmatrix}, \qquad \begin{aligned} &\mathcal{C}_0 = 0, \\ &\mathcal{C}_i = i\text{th block row of } \mathcal{C}. \end{aligned} \tag{2.3}$$

Substituting for $x_{i|k}$ in (2.1) and collecting terms (try this yourself) gives:

$$J = \mathbf{u}_k^\top H \mathbf{u}_k + 2x_k^\top F^\top \mathbf{u}_k + x_k^\top G x_k \tag{2.4}$$

where

$$H = \mathcal{C}^\top \tilde{Q} \mathcal{C} + \tilde{R}, \qquad F = \mathcal{C}^\top \tilde{Q} \mathcal{M} \tag{2.5a}$$

$$G = \mathcal{M}^\top \tilde{Q} \mathcal{M} + Q \tag{2.5b}$$

with $\tilde{Q} = \begin{bmatrix} Q & 0 & \cdots & 0 \\ 0 & \ddots & & \vdots \\ \vdots & & Q & 0 \\ 0 & \cdots & 0 & \overline{Q} \end{bmatrix}$ and $\tilde{R} = \begin{bmatrix} R & 0 & \cdots & 0 \\ 0 & \ddots & & \vdots \\ \vdots & & R & 0 \\ 0 & \cdots & 0 & R \end{bmatrix}$.

Note that matrices H, F, and G can be computed offline.

Example 2.1. For the 2nd order system considered in Section 1, with

$$A = \begin{bmatrix} 1.1 & 2 \\ 0 & 0.95 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 0.0787 \end{bmatrix}, \quad C = \begin{bmatrix} -1 & 1 \end{bmatrix}$$

the convolution matrix $\mathcal{C}$ for $N = 4$ is given by

$$\mathcal{C} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0.0787 & 0 & 0 & 0 \\ 0.1574 & 0 & 0 & 0 \\ 0.0748 & 0.0787 & 0 & 0 \\ 0.3227 & 0.1574 & 0 & 0 \\ 0.0710 & 0.0748 & 0.0787 & 0 \\ 0.4970 & 0.3227 & 0.1574 & 0 \\ 0.0675 & 0.0710 & 0.0748 & 0.0787 \end{bmatrix}$$

so that the cost with $Q = C^\top C$, $R = 0.01$, and $\overline{Q} = Q$ is given by (2.4) with

$$H = \begin{bmatrix} 0.2708 & 0.1224 & 0.0157 & -0.0338 \\ 0.1224 & 0.0863 & 0.0143 & -0.0198 \\ 0.0157 & 0.0143 & 0.0230 & -0.0065 \\ -0.0338 & -0.0198 & -0.0065 & 0.0162 \end{bmatrix}, \quad F = \begin{bmatrix} 0.9772 & 4.9253 \\ 0.3832 & 2.1739 \\ 0.0162 & 0.2189 \\ -0.1152 & -0.6175 \end{bmatrix}$$

$$G = \begin{bmatrix} 7.5892 & 22.7765 \\ 22.7765 & 103.6836 \end{bmatrix}.$$

◊

2.1.1 LTV prediction models

The above formulation applies to systems with linear time-varying models:

$$x_{k+1} = A_k x_k + B_k u_k.$$

In this case the state predictions can be written

$$x_{i|k} = \prod_{j=i-1}^{0} A_{k+j} x_k + \mathcal{C}_i(k)\mathbf{u}_k \quad i = 0, \ldots, N$$

where

$$\mathcal{C}_0(k) = 0$$

$$\mathcal{C}_i(k) = \begin{bmatrix} \prod_{j=i-1}^{1} A_{k+j}B_k & \prod_{j=i-1}^{2} A_{k+j}B_k & \cdots & B_{k+i-1} & 0 & \cdots & 0 \end{bmatrix}$$

and the cost (2.1) is therefore given by (2.4) with H_k, F_k, and G_k as defined in (2.5a,b), with $\mathcal{C}_i$ replaced by $\mathcal{C}_i(k)$. However, unlike the LTI case, the cost matrices H, F, G are now time-varying and therefore must be computed online.

2.2 Unconstrained optimization [3]

In the absence of constraints, the optimization $\mathbf{u}_k^* = \arg\min_{\mathbf{u}_k} J(x_k, \mathbf{u}_k)$ has a closed-form solution which can be derived by considering the gradient of J

[3] Reading: *Kouvaritakis & Cannon* §2.2 and §2.3, *Maciejowski* §3.1.

with respect to $\mathbf{u}$:

$$\nabla_{\mathbf{u}} J = 2H\mathbf{u} + 2Fx.$$

Clearly $\nabla_{\mathbf{u}} J = 0$ must be satisfied at a minimum point of J, and since H is positive definite (or positive semidefinite), any $\mathbf{u}$ such that $\nabla_{\mathbf{u}} J = 0$ is necessarily a minimum point. Therefore the optimal $\mathbf{u}^*$ is unique only if H is nonsingular and is then given by[4]

$$\mathbf{u}_k^* = -H^{-1}Fx_k. \tag{2.6}$$

If H is singular (i.e. positive semidefinite rather than positive definite), then the optimal $\mathbf{u}^*$ is non-unique, and a particular solution of $\nabla_{\mathbf{u}} J = 0$ has to be defined as $\mathbf{u}_k = -H^\dagger F x_k$ where $H^\dagger$ is a left inverse of H (i.e. $H^\dagger H = I$).

Implementing the first element of the optimal prediction $\mathbf{u}_k^*$ at each sampling instant k defines a receding horizon control law. Since H and F are constant, this is a linear time-invariant feedback controller $u_k = Lx_k$, where the gain matrix L is the first row of $-H^{-1}F$ for the single-input case (or the first n_u rows for the case that u has dimension n_u), i.e.

$$u_k = u_{0|k}^* = Lx_k, \quad L = -\begin{bmatrix} I_{n_u} & 0 & \cdots & 0 \end{bmatrix} H^{-1}F. \tag{2.7}$$

Example 2.2. For the system considered in example 2.1, the optimal predicted input sequence for the cost with horizon $N = 4$ and weights $Q = C^\top C$, $R = 0.01$, $\overline{Q} = Q$ is obtained by substituting H and F into (2.6):

$$\mathbf{u}_k^* = -H^{-1}Fx_k = \begin{bmatrix} -4.3563 & -18.6889 \\ 1.6383 & 1.2379 \\ 1.4141 & 2.9767 \\ 0.5935 & 1.8326 \end{bmatrix} x_k.$$

The corresponding receding horizon control law is therefore a fixed linear feedback controller:

$$u_k = -\begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} H^{-1}Fx_k = Lx_k \quad L = \begin{bmatrix} -4.3563 & -18.6889 \end{bmatrix}.$$

[4] Sufficient conditions for H to be non-singular are for example that $R > 0$ or that $Q, \overline{Q} > 0$ and the pair (A, B) is controllable.

Figure 8 shows the predicted response at $k = 0$ and the closed-loop response for initial condition $x_0 = (0.5, -0.5)$. ◊

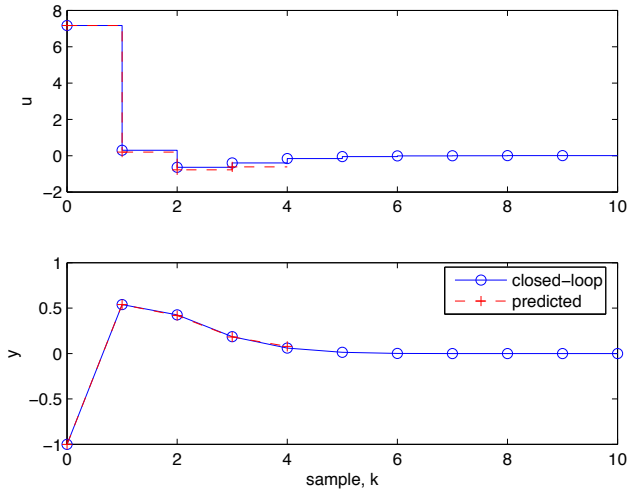


Figure 8: Predicted and closed-loop responses for example 2.2 with $N = 4$.

2.2.1 Horizon length and perform

$\mathbf{u}_k^* = -H^{-1}Fx_k.$

Close–loop

The linear feedback form of (2.6) is to be expected since $\mathbf{u}^*$ is the solution of a LQ-optimal control problem. However, unlike the infinite horizon LQ-optimal control problem (see B4 Lecture Notes), for which there is no difference between the optimal predicted input sequence and its receding horizon implementation in the absence of disturbances and model errors, there can be significant discrepancy between the predictions of (2.6) and closed-loop responses with the receding horizon controller (2.7). This discrepancy tends to increase as the horizon is reduced, as the example below shows.

差异

Example 2.3. The table below gives the linear feedback gain L and the corresponding closed-loop poles (i.e. the eigenvalues of $A + BL$) as the horizon N is reduced for the system of example 2.2. Predicted responses at $k = 0$ and closed-loop responses for initial condition $x_0 = (0.5, -0.5)$ are shown in Figures 9 and 10.

	$N = 4$	$N = 3$	$N = 2$	$N = 1$
L	$[-4.36 \ \ -18.69]$	$[-3.80 \ \ -16.98]$	$[1.22 \ \ -3.95]$	$[5.35 \ \ 5.10]$
$\lambda(A+BL)$	$0.29 \pm 0.17j$	$0.36 \pm 0.22j$	$1.36, 0.38$	$2.15, 0.30$

◊

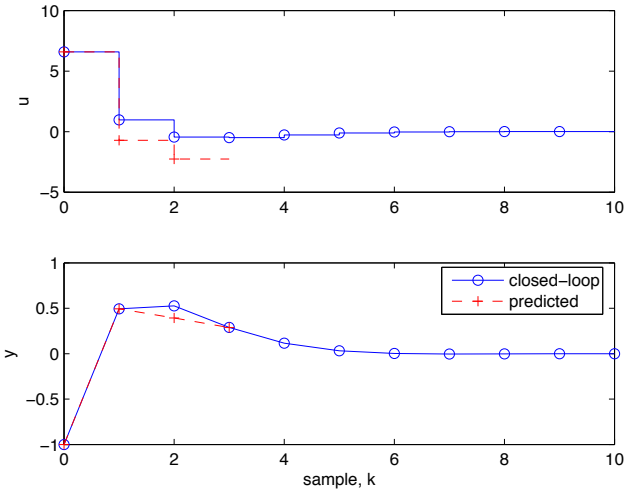


Figure 9: Predicted and closed-loop responses for $N = 3$.

2.3 Infinite horizon cost [5]

Because of the difference between predicted and closed-loop responses, there is no guarantee that a receding horizon controller based on a finite-horizon cost will achieve the optimal predicted performance in closed-loop operation. In fact the closed-loop system may even be unstable. This can be seen in the previous example, where the closed-loop poles lie outside the unit circle if $N < 3$. The cause of instability here becomes clearer if the predictions at each time k are

[5] Reading: *Kouvaritakis & Cannon* §2.2, §2.3 and §2.6, *Maciejowski* §6.2.

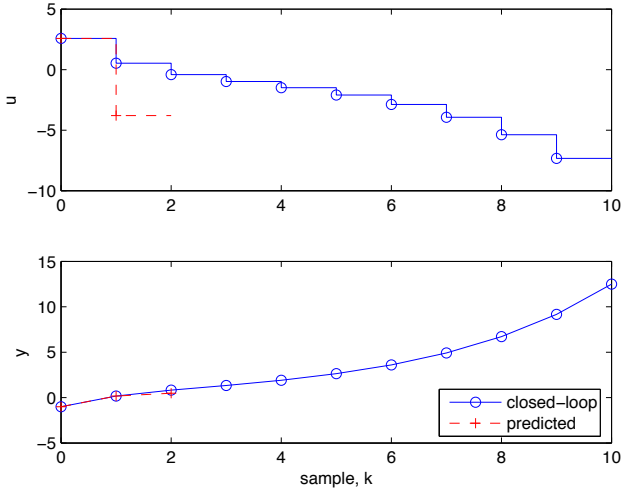


Figure 10: Predicted and closed-loop responses for $N = 2$.

compared with the closed-loop response (Fig. 11). The use of a short-sighted objective leads to optimal predictions that continually underestimate the need to force the outputs that lie further into the future to zero. This behaviour is common to (but not limited to) non-minimum phase systems, for which the predicted tracking errors must in general increase initially in order to allow for a decrease later in the prediction horizon.[6]

This problem is avoided entirely if performance is evaluated over an infinite prediction horizon, i.e. if the cost is defined as

$$J(x_k, \mathbf{u}_k) = \sum_{i=0}^{\infty} \Big(\|x_{i|k}\|_Q^2 + \|u_{i|k}\|_R^2 \Big). \tag{2.8}$$

To make sure that the problem of minimizing this cost is tractable, it is then necessary to define the predicted input sequence over the infinite prediction horizon in such a way that the number of free variables in the MPC optimization

[6] For example the response of aircraft altitude to changes in elevator angle is non-minimum phase since moving the elevator so as to bring the plane's nose up and hence gain height results in an initial loss of altitude.

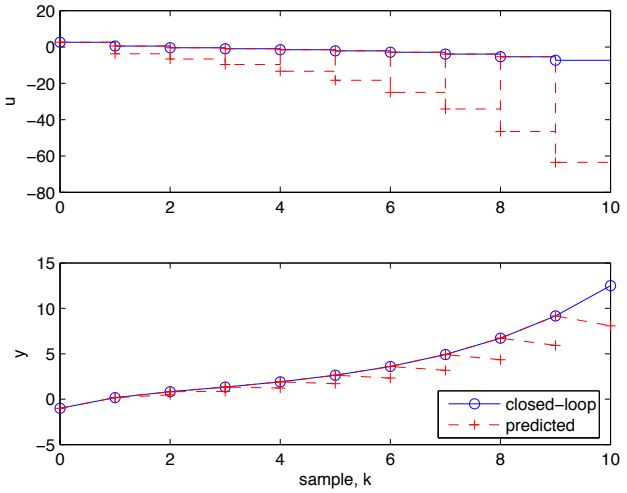


Figure 11: Closed-loop responses and predicted responses at $k = 0, 1, \ldots, 8$ for $N = 2$.

remains finite. A convenient means of achieving this (which in Section 3 will be shown to be optimal under certain conditions) is through the **dual mode** predictions:

$$u_{i|k} = \begin{cases} \text{optimization variables} & i = 0, 1, \ldots, N-1 \quad \text{(mode 1)} \\ Kx_{i|k} & i = N, N+1, \ldots \quad \text{(mode 2)} \end{cases} \tag{2.9}$$

Here mode 1 refers to an initial horizon of N samples over which the predicted inputs are variables in the MPC optimization. On the other hand, inputs are defined by a stabilizing feedback law ($u = Kx$) over the remaining infinite horizon of mode 2 (Fig. 12).

For the dual mode input predictions (2.9), an infinite horizon cost need only be evaluated explicitly over mode 1 since $J(x_k, \mathbf{u}_k)$ can be re-written in the form of (2.1). This is done by choosing the **terminal weighting matrix** $\overline{Q}$ so that the term $x_{N|k}^\top \overline{Q} x_{N|k}$ is equal to the cost over the mode 2 prediction

$$J(x_k, \mathbf{u}_k) = \sum_{i=0}^{N-1} \Big(\|x_{i|k}\|_Q^2 + \|u_{i|k}\|_R^2 \Big) + \|x_{N|k}\|_{\overline{Q}}^2 \tag{2.1}$$

$$u_{i|k} = \begin{cases} \text{optimization variables} & i = 0, 1, \ldots, N-1 \quad \text{(mode 1)} \\ Kx_{i|k} & i = N, N+1, \ldots \quad \text{(mode 2)} \end{cases}$$

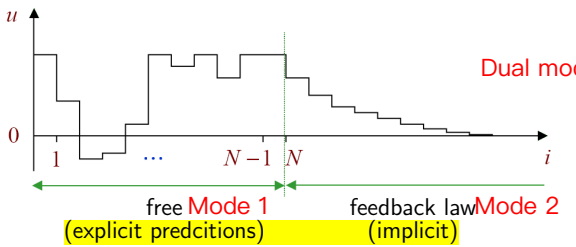


Figure 12: Dual mode input predictions.

horizon, which is achieved by specifying $\overline{Q}$ as the solution of the Lyapunov equation:

$$\overline{Q} - (A+BK)^\top \overline{Q}(A+BK) = Q + K^\top RK, \tag{2.10}$$

as shown below.

Theorem 2.1 (Terminal weighting matrix)**.** *Along closed-loop trajectories of the model (1.1) under the feedback law $u_k = Kx_k$, the infinite horizon quadratic cost is given by:*

$$\sum_{i=0}^{\infty} \left(\|x_i\|_Q^2 + \|u_i\|_R^2 \right) = x_0^\top \overline{Q} x_0 \tag{2.11}$$

where $\overline{Q}$ satisfies (2.10).

To prove this result, first pre- and post-multiply (2.10) by $x_i^\top$ and x_i:

$$\overline{Q} - (A+BK)^\top \overline{Q}(A+BK) = Q + K^\top RK, \quad \|x_i\|_{\overline{Q}}^2 - \|(A+BK)x_i\|_{\overline{Q}}^2 = \|x_i\|_Q^2 + \|Kx_i\|_R^2.$$

Defining $V(x) = x^\top \overline{Q} x$, and using $u_i = Kx_i$, $x_{i+1} = (A+BK)x_i$, this is equivalent to

$$V(x_i) - V(x_{i+1}) = \|x_i\|_Q^2 + \|u_i\|_R^2,$$

Summing this equation over $i = 0, 1, \ldots$ therefore gives

$$V(x_0) - \lim_{k\to\infty} V(x_k) = \sum_{i=0}^{\infty} \left(\|x_i\|_Q^2 + \|u_i\|_R^2 \right),$$

but $V(x_k) = \|(A+BK)^k x_0\|_{\overline{Q}}^2 \to 0$ as $k \to \infty$ due to the assumption that $(A+BK)$ is stable. Therefore

$$V(x_0) = \sum_{i=0}^{\infty} \left(\|x_i\|_Q^2 + \|u_i\|_R^2 \right),$$

which implies (2.11) since $V(x_0) = x_0^\top \overline{Q} x_0$.

Note that:

1. The Lyapunov equation (2.10) has a (unique) solution for $\overline{Q}$ if and only if the eigenvalues of $A+BK$ lie inside the unit circle, since this condition is necessary and sufficient for convergence of the infinite sum in (2.11).
2. It is easy to show that $\overline{Q}$ is positive definite if either $Q + K^\top RK > 0$ or $Q = C^\top C$ where $(A+BK, C)$ is observable.
3. From (2.11) it is clear that (2.1) is equal to the infinite horizon cost (2.8).

2.3.1 The relationship between unconstrained MPC and LQ-optimal control

The obvious choice for the mode 2 feedback gain K is the LQ-optimal gain for the cost (2.8), since this gives optimal predicted performance over mode 2. Due to the optimality of predictions over both modes 1 and 2, the optimal predicted trajectory $\mathbf{u}^*$ in (2.6) is then necessarily identical to the infinite horizon optimal input sequence:

$$\mathbf{u}_k^* = \begin{bmatrix} K \\ K(A+BK) \\ \vdots \\ K(A+BK)^{N-1} \end{bmatrix} x_k, \qquad \mathbf{u}_k^* = -H^{-1}Fx_k.$$

implying that the receding horizon control law (2.7) is in fact equal to the LQ-optimal feedback law $u = Kx$. This result should not be surprising since the model and cost are the same for MPC and LQ-optimal control, here MPC simply provides an alternative method of determining the optimal control law.

$$u_k = u_{0|k}^* = Lx_k, \quad L = -\begin{bmatrix} I_{n_u} & 0 & \cdots & 0 \end{bmatrix} H^{-1} F. \tag{2.7}$$

MPC: 预测最优输入

Example 2.4. For the model and cost weights Q, R considered in example 2.2, the LQ-optimal gain is $K = \begin{bmatrix}-4.36 & -18.74\end{bmatrix}$. The solution of the Lyapunov equation (2.10) for $\overline{Q}$ and the corresponding H, F, G in (2.5a,b) for $N = 4$ are given by

$$\overline{Q} = \begin{bmatrix}3.9153 & 4.8269\\ 4.8269 & 13.8564\end{bmatrix}, \qquad G = \begin{bmatrix}13.8384 & 66.6933\\ 66.6933 & 413.1200\end{bmatrix}$$

$$H = \begin{bmatrix}1.4402 & 0.9840 & 0.5870 & 0.2624\\ 0.9840 & 0.7218 & 0.4363 & 0.2000\\ 0.5870 & 0.4363 & 0.3043 & 0.1413\\ 0.2624 & 0.2000 & 0.1413 & 0.0958\end{bmatrix}, \quad F = \begin{bmatrix}3.6741 & 23.9448\\ 2.3664 & 16.1810\\ 1.3259 & 9.4964\\ 0.5562 & 4.1784\end{bmatrix}.$$

From (2.7), the receding horizon implementation of the optimal predicted input sequence therefore defines the control law

$$u_k = -\begin{bmatrix}1 & 0 & 0 & 0\end{bmatrix} H^{-1} F x_k = \begin{bmatrix}-4.3608 & -18.7401\end{bmatrix} x_k$$

which is identical to the LQ-optimal controller for the same cost. ◊

2.4 Incorporating constraints [7] 内置约束

The purpose of MPC is clearly not to emulate the unconstrained LQ-optimal controller, which after all is simply a linear feedback law that can be computed offline using knowledge of the plant model. The real advantage of MPC lies in its ability to determine nonlinear feedback laws which are optimal for constrained systems through numerical calculations that are performed online (i.e. in between samples). Having determined the quadratic cost as a function of input predictions in sections 2.1 and 2.3, this section re-writes linear input and state constraints:

MPC的用武之地：二次规划……

$$\underline{u} \le u_k \le \overline{u} \tag{2.12a}$$
$$\underline{x} \le x_k \le \overline{x} \tag{2.12b}$$

in the form $A_c \mathbf{u} \le b_c$, which is suitable for including in the problem of optimizing J subject to constraints.

[7] Reading: *Kouvaritakis & Cannon* §2.4, *Maciejowski* §3.2.

$$x_{i|k} = A^i x_k + \mathcal{C}_i \mathbf{u}_k, \quad i = 0, \dots, N$$
$$\text{or} \quad \mathbf{x}_k = \mathcal{M} x_k + \mathcal{C} \mathbf{u}_k, \quad \text{where } \mathcal{M} = \begin{bmatrix}A\\ A^2\\ \vdots\\ A^N\end{bmatrix}. \tag{2.2}$$

The input constraints (2.12a) are equivalent to $u_k \le \overline{u}$ and $-u_k \le -\underline{u}$. These constraints applied to mode 1 predictions $u_{i|k}$, $i = 0, \dots, N-1$ can therefore be expressed in terms of $\mathbf{u}_k$ as

$$\begin{bmatrix}I\\ -I\end{bmatrix} \mathbf{u}_k \le \begin{bmatrix}\mathbf{1}\overline{u}\\ -\mathbf{1}\underline{u}\end{bmatrix} \tag{2.13}$$

where $\mathbf{1}$ is a vector of ones for the single input case (or $\mathbf{1} = \begin{bmatrix}I_{n_u} & \cdots & I_{n_u}\end{bmatrix}$ for the case that u has dimension n_u). Similarly, using (2.2), the state constraints (2.12b) applied to mode 1 predictions $x_{i|k}$, $i = 1, \dots, N$ are equivalent to:

$$\begin{bmatrix}\mathcal{C}_i\\ -\mathcal{C}_i\end{bmatrix} \mathbf{u}_k \le \begin{bmatrix}\overline{x}\\ -\underline{x}\end{bmatrix} + \begin{bmatrix}-A^i\\ A^i\end{bmatrix} x_k, \quad i = 1, \dots, N. \tag{2.14}$$

The combination of (2.12a,b) applied to mode 1 predictions can therefore be expressed as constraints on $\mathbf{u}_k$ of the form

$$A_c \mathbf{u}_k \le b_0 + B_x x_k \tag{2.15}$$

where A_c, b_0, B_x are constant matrices that can be determined offline.

2.5 Quadratic Programming [8] 二次规划

Combining the objective function and constraints derived above, the optimization of the infinite horizon cost (2.8) subject to constraints (2.15) requires the solution of a QP problem:

$$\begin{aligned}\underset{\mathbf{u}}{\text{minimize}} \quad & \mathbf{u}^\top H \mathbf{u} + 2 x_k^\top F^\top \mathbf{u}\\ \text{subject to} \quad & A_c \mathbf{u} \le b_0 + B_x x_k\end{aligned} \tag{2.16}$$

Since H is positive (semi-)definite (as discussed in section 2.2), and since the constraints are linear, this is a convex optimization problem which therefore has a unique solution. This section outlines the two types of algorithm (active set and interior point algorithms) commonly used to solve QP problems, but to help explain these methods a general result of constrained optimization theory is given first.[9]

[8] Reading: *Kouvaritakis & Cannon* §2.8, *Maciejowski* §3.3

[9] A full treatment of the theory of constrained optimization is beyond the scope of this course. For a fuller discussion see e.g. the lecture notes on C6B Optimization.

二次规划的解法： active set and interior point algorithms

Theorem 2.2 (Optimization with equality constraints). *If* $\mathbf{u}^*$ *satisfies*

$$\begin{aligned}\mathbf{u}^* = \arg\min_{\mathbf{u}} \quad & f(\mathbf{u}) \\ \text{subject to} \quad & c_i(\mathbf{u}) = 0, \quad i = 1, \ldots, m\end{aligned}$$

(where f *and* c_i *are smooth functions), then scalars* λ_i^*, $i = 1, \ldots, m$ *exist satisfying*

$$\nabla_{\mathbf{u}} f(\mathbf{u}^*) + \sum_{i=1}^{m} \lambda_i \nabla_{\mathbf{u}} c_i(\mathbf{u}^*) = 0 \tag{2.17a}$$

$$c_i(\mathbf{u}^*) = 0, \quad i = 1, \ldots, m. \tag{2.17b}$$

Condition (2.17a) is an extension of the gradient condition, $\nabla_{\mathbf{u}} f(\mathbf{u}^*) = 0$, which must be satisfied at a minimum point of $f(\mathbf{u})$ if $\mathbf{u}$ is unconstrained. The addition of the second term simply ensures that $f(\mathbf{u})$ cannot be reduced by perturbing $\mathbf{u}^*$ by an incremental distance in any direction for which the constraint $c_i(\mathbf{u}) = 0$ remains satisfied. This is because (2.17a) forces the gradient $\nabla_{\mathbf{u}} f(\mathbf{u})$ to be normal in $\mathbf{u}$-space to one or more of the constraint surfaces $c_i(\mathbf{u}) = 0$ at $\mathbf{u} = \mathbf{u}^*$. The scalars λ_i are known as **Lagrange multipliers**.

Note that:

1. Condition (2.17a) also applies to the case of inequality constraints, $c_i(\mathbf{u}) \leq 0$, but with extra conditions on the sign of the multipliers, namely $\lambda_i^* \geq 0$ and $\lambda_i^* = 0$ if $c_i(\mathbf{u}^*) < 0$.
2. Problems with inequality constraints are generally harder to solve than similar equality constrained problems since only a subset of the constraints may be active (i.e. satisfied with equality) at the solution.

2.5.1 Active set algorithms

These compute the solution of the QP optimization (2.16) by solving a sequence of problems involving only equality constraints. Let $a_i^\top$ denote the ith row of A_c and b_i the ith element of $b_0 + B_x x_k$, for $i = 1, \ldots, m$. Then an individual constraint $a_i^\top \mathbf{u} \leq b_i$ is **active** at the solution if $a_i^\top \mathbf{u}^* = b_i$, and **inactive** if $a_i^\top \mathbf{u}^* < b_i$. Clearly the inactive constraints can be removed from the problem without affecting the solution (Fig. 13), and it follows that $\mathbf{u}^*$ is also the solution of the equality constrained problem:

$$\begin{aligned}\underset{\mathbf{u}}{\text{minimize}} \quad & \mathbf{u}^\top H \mathbf{u} + 2 x_k^\top F^\top \mathbf{u} \\ \text{subject to} \quad & a_i^\top \mathbf{u} = b_i, \quad i \in \mathcal{A}^*\end{aligned}$$

where $\mathcal{A}^* = \{i : \ a_i^\top \mathbf{u}^* = b_i\}$ is the set of active constraints at the solution of (2.16). The solution of (2.16) can therefore be found by iteratively: (i) selecting a possible combination of active constraints, (ii) solving the corresponding equality constrained problem (by solving the linear equations (2.17a,b)), and (iii) testing the optimality of the solution or the closest point to the solution at which all of the inequality constraints are satisfied. Note that the optimality of a trial solution can be determined from the associated Lagrange multipliers (using the conditions given in point 1 above). Also the successive active sets are chosen so as to make the objective function decrease at each iteration, and well-designed active set solvers therefore manage to avoid testing large numbers of possible combinations of active constraints.

2.5.2 Interior point QP algorithms

This approach solves an unconstrained problem:

$$\underset{\mathbf{u}}{\text{minimize}} \quad \mu\{\mathbf{u}^\top H \mathbf{u} + 2 f^\top \mathbf{u}\} + \phi(\mathbf{u}) \tag{2.18}$$

at each iteration, where μ is a scalar and $\phi(\mathbf{u})$ is a **barrier function** which is finite whenever $\mathbf{u}$ satisfies constraints but which tends to infinity as $\mathbf{u}$ approaches a constraint boundary. For constraints $a_i^\top \mathbf{u} \leq b_i$, the barrier function is typically defined as $\phi(\mathbf{u}) = \sum_{i=1}^{m} -\log(b_i - a_i^\top \mathbf{u})$, and the solution of (2.18) for any given μ therefore necessarily satisfies constraints. It can also be shown that the solution of (2.18) tends to $\mathbf{u}^*$ as $\mu \to \infty$ (see for example Fig. 14, for which $\mathbf{u}^* = 1$). Interior point methods therefore increase μ over successive iterations until constraints are met to within a given tolerance.

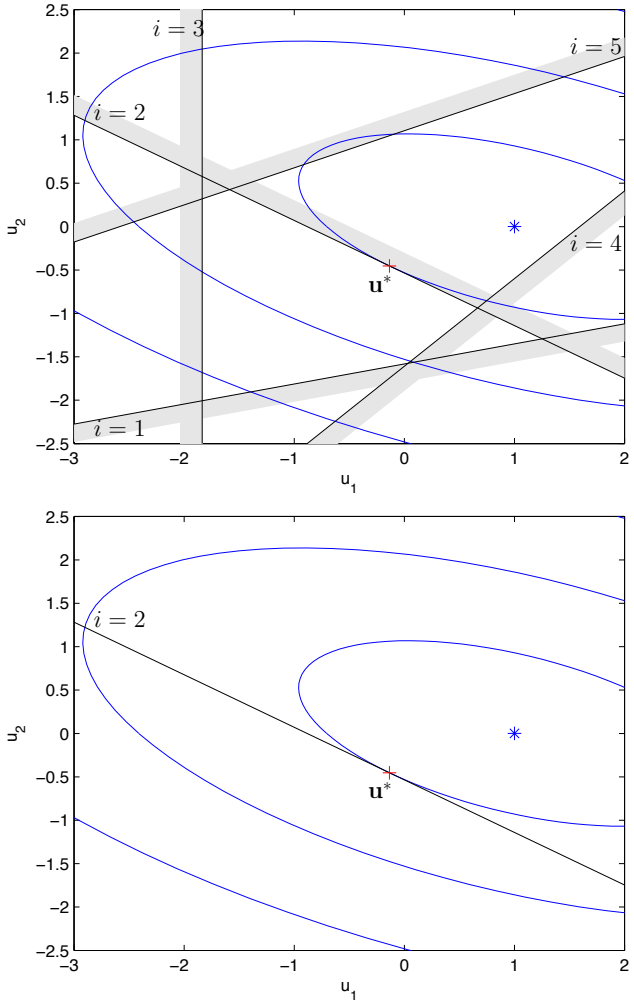


Figure 13: **Upper:** QP problem with 5 inequality constraints. Only constraint 2 is active at the solution. **Lower:** Minimizing the same function subject to constraint 2 alone yields the same solution. The ellipses are contours of the objective function and the unconstrained minimum is marked '$*$'.

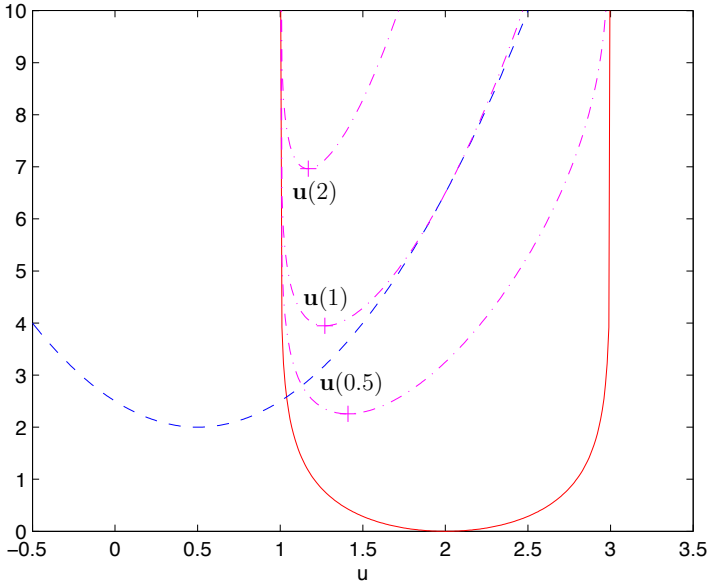


Figure 14: Interior point iteration for a scalar example. Solid line: barrier function $\phi(\mathbf{u})$ for constraints $1 \leq \mathbf{u} \leq 3$. Dashed line: quadratic objective function $J(\mathbf{u})$. Dashed-dotted lines: $\mu J(\mathbf{u}) + \phi(\mathbf{u})$ plotted for $\mu = 0.5, 1, 2$. Minimum points $\mathbf{u}(\mu)$ for $\mu = 0.5, 1, 2$ are marked '+'.

The interior point approach has a lower computational load for large-scale problems involving hundreds of optimization variables. Unlike active set solvers however, it is not possible to initialize the iteration at an estimate close to the actual solution because the corresponding value for μ is unknown. This can be a big disadvantage in predictive control, where a good estimate of the current optimal control sequence can usually be determined from the solution of the optimization problem computed at the previous sampling instant.

Summary of Section 2

- A linear discrete-time plant model enables the future predicted states to be determined as linear functions of the vector of future inputs $\mathbf{u}_k$. Therefore a quadratic performance index J is a quadratic function of $\mathbf{u}$, and linear constraints on inputs and states are equivalent to linear constraints on $\mathbf{u}$.
- To improve closed-loop performance of MPC, the horizon over which the cost J is evaluated should be extended to infinity. Assuming dual mode predictions (where the predicted future inputs $u_{i|k}$ are defined by a linear state feedback controller for all $i \geq N$), this is done by including a terminal weighting matrix $\overline{Q}$ in J, where $\overline{Q}$ is computed by solving a Lyapunov matrix equation. The resulting MPC is identical to LQ-optimal control if constraints are inactive.
- Determining the sequence $\mathbf{u}$ that minimizes J subject to linear constraints on inputs and states is a Quadratic Programming problem, which can be solved using either active set or interior point QP solvers.

3 Closed-loop properties of model predictive control[10]

MPC的闭环性质　保证稳定性和最优性能

This section is concerned with the performance of predictive control laws in closed-loop operation. Specifically: how to choose the cost and constraints so as to ensure closed-loop stability and optimal performance when there is no modelling error or unknown disturbance.

Section 2.8 explained how to extend the horizon of the MPC cost to infinity by including a terminal weight in the cost function. The use of an infinite cost horizon ensures that the predicted and actual input and state trajectories are identical when there are no constraints, since in both cases the optimal input is given by a fixed linear state feedback controller. But when constraints are included in the optimization problem, the predicted and actual responses may differ even if an infinite cost horizon is used, as the following example demonstrates.

Example 3.1. Input constraints:

$$-1 \le u_k \le 1$$

are imposed on the system that was considered in examples 2.1-2.4. To accommodate these constraints, a predictive control law is constructed by solving at each sample $k = 0, 1, 2, \ldots$ the optimization problem:

$$\begin{aligned} &\underset{\mathbf{u}_k}{\text{minimize}}\ J(x_k, \mathbf{u}_k) = \sum_{i=0}^{N-1} \left(\|x_{i|k}\|_Q^2 + \|u_{i|k}\|_R^2 \right) + \|x_{N|k}\|_{\overline{Q}}^2 \\ &\text{subject to}\ -1 \le u_{i|k} \le 1, \quad \text{for } i = 0, 1, \ldots, N-1 \end{aligned}$$

and implementing $u_k = u_{0|k}^*$.

Here $Q = C^\top C$, $R = 0.01$, $N = 2$, and $\overline{Q}$ is the solution of (2.10) for the case that K is the unconstrained LQ-optimal gain ($K = \begin{bmatrix} -4.36 & -18.74 \end{bmatrix}$). Using (2.5) and (2.13), the optimization can be written in the form of the QP

[10] Reading: *Kouvaritakis & Cannon* §2.4, §2.5, §2.6

problem (2.16) with

$$H = \begin{bmatrix} 0.3043 & 0.1413 \\ 0.1413 & 0.0958 \end{bmatrix} \quad F = \begin{bmatrix} 1.0958 & 5.5443 \\ 0.4596 & 2.5417 \end{bmatrix}$$

$$A_c = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ -1 & 0 \\ 0 & -1 \end{bmatrix} \quad b_0 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \quad B_x = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

The optimization can be performed using e.g. Matlab's `quadprog` function:

```
uPredicted = quadprog(H,F*x,Ac,b0+Bx*x);
```

where `x` is the current plant state. The current input to the plant is then given by `u = uPredicted(1)`. For initial condition $x_0 = (0.5, -0.5)$, the closed-loop performance of this controller is good (Fig. 15), but for $x_0 = (0.8, -0.8)$ we get the unstable response of Figure 16. ◊

3.1 Lyapunov stability analysis[11]

The instability of example 3.1 is potentially catastrophic, even though the MPC optimization remains feasible at all times. Here the closed-loop system is nonlinear because constrained MPC is a nonlinear control law, so closed-loop stability cannot be checked by considering closed-loop system poles. However it can be seen that the MPC algorithm in the example is potentially destabilizing by looking at the time-variation of the optimal predicted cost $J_k^* = J(x_k, \mathbf{u}_k^*)$ (Fig. 17). Even for the well-behaved response of Fig. 15, J_k^* is initially increasing. Although the *predicted* state trajectories are necessarily stable (since the cost over an infinite horizon is finite), the initial increase in J_k^* implies:

- the stored energy in the system is initially increasing, since J_k^* is in this case a positive definite function of x_k.
- the closed-loop input trajectory does not follow an optimal predicted trajectory, since otherwise the predicted cost could not increase over time.

[11] Reading: *Kouvaritakis & Cannon* §2.6

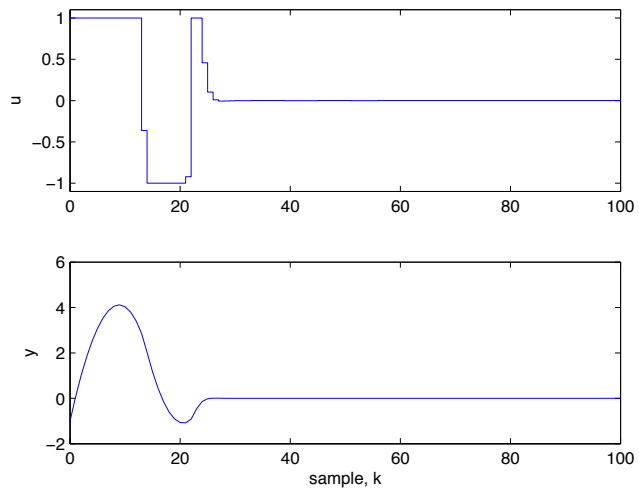


Figure 15: Response for initial condition $x_0 = (0.5, -0.5)$.

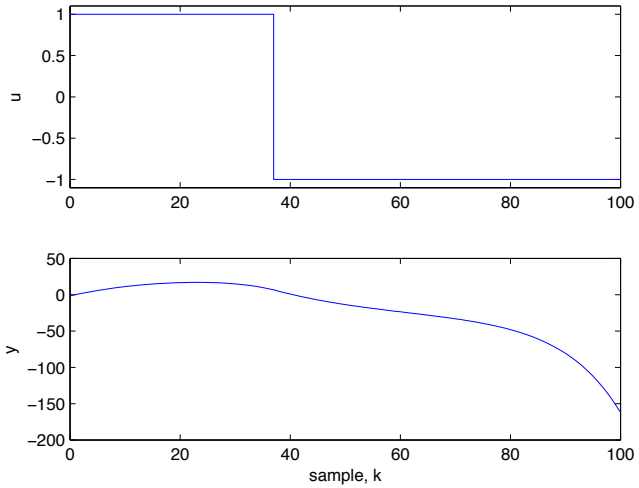


Figure 16: Response for initial condition $x_0 = (0.8, -0.8)$.

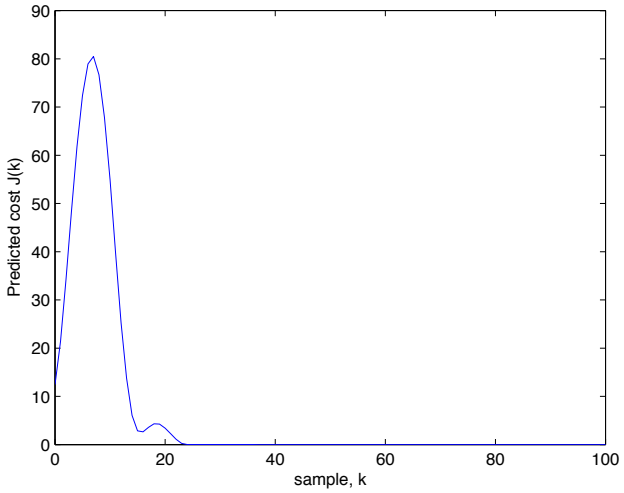


Figure 17: Optimal predicted cost J_k^* for example 3.1 with $x_0 = (0.5, -0.5)$.

These observations suggest that instability could be avoided if J_k^* were decreasing over time, or equivalently that stability can be analyzed by considering J_k^* as a Lyapunov function. Before giving details, some extensions of Lyapunov stability analysis (see C21 Nonlinear Systems lecture notes) to discrete-time systems are first summarized below.

Equilibrium point. x_0 is an equilibrium point of the system $x_{k+1} = f(x_k)$ if and only if $f(x_0) = x_0$. As usual $x = 0$ is assumed to be the equilibrium of interest, and $f(0) = 0$.

Stable equilibrium. $x = 0$ is a stable equilibrium if, for all $k > 0$, the state x_k remains within an arbitrarily small region of state space containing $x = 0$ whenever the initial condition x_0 lies sufficiently close to $x = 0$, i.e. for all $R > 0$ there exists $r > 0$ such that

$$\|x_0\| < r \quad \Longrightarrow \quad \|x_k\| < R, \quad \forall k > 0.$$

Theorem 3.1 (Stability). *For the discrete time system $x_{k+1} = f(x_k)$, if there exists a continuously differentiable scalar function $V(x)$ such that:*

(a). $V(x)$ is positive definite

(b). $V\big(f(x)\big) - V(x) \leq 0$

whenever $\|x\|$ is sufficiently small, then $x = 0$ is a stable equilibrium point.

Theorem 3.2 (Convergent series). *For any sequence $\{a_0, a_1, \ldots\}$:*

if $\sum_{k=0}^{n} a_k$ tends to a finite limit as $n \to \infty$, then $a_k \to 0$ as $k \to \infty$.

Theorem 3.3 (Asymptotic convergence). *For the discrete time system $x_{k+1} = f(x_k)$, if there exists a continuously differentiable scalar function $V(x)$ such that:*

(a). $V(x)$ is positive definite

(b). $V\big(f(x)\big) - V(x) \leq -l(x) \leq 0$

then $l(x_k) \to 0$ as $k \to \infty$.

Note that:

1. Theorem 3.2 is a direct consequence of the more general condition for series convergence known as Cauchy's convergence test.
2. The proof of the convergence result in Theorem 3.3 follows directly from Theorem 3.2, since (b) implies that

$$l(x_k) \leq V(x_k) - V(x_{k+1})$$

Summing both sides of this inequality over $k = 0, 1, \ldots$ therefore gives

$$\sum_{k=0}^{\infty} l(x_k) \leq V(x_0) - \lim_{k\to\infty} V(x_k).$$

The RHS of this inequality is necessarily finite because $V(x_k) \geq 0$ and $V(x_{k+1}) \leq V(x_k)$ imply that $V(x_k)$ tends to a finite limit as $k \to \infty$. Therefore $l(x) \to 0$ from Theorem 3.2.

Assume for simplicity that the MPC optimization problem is always feasible (this assumption is removed in the next section), i.e. for all k there exist predicted input and state trajectories that satisfy the constraints. Then the optimal value of the predicted cost (2.1) is a function of x_k:

$$J(x_k, \mathbf{u}_k^*) = J_k^* = V(x_k),$$

and although this function is not known explicitly, it is straightforward to derive conditions under which (a) and (b) of Theorem 3.3 are satisfied.

Positive definiteness of J_k^*. If either of the following conditions holds:

(i). Q is positive definite

(ii). $(A, Q^{1/2})$ is an observable pair, where $Q^{1/2}$ is any matrix with $Q^{1/2\top} Q^{1/2} = Q$,

then J_k^* is a positive definite function of x_k.

Here condition (i) simply ensures that the first term (and hence the entire sum) in (2.1) is positive definite in x_k. On the other hand if (ii) is satisfied, then $J_k^* = 0$ implies $\|Q^{1/2} x_{i|k}\|_2^2 = 0$, $i = 0, 1, \ldots$, and hence $x_k = 0$ since $(A, Q^{1/2})$ is observable. Since $J_k^* \geq 0$ for all x_k, it follows that J_k^* is positive definite in x_k.

Convergence of J_k^*. If both of the following conditions hold:

(iii). the terminal weight in (2.1) is chosen so that J_k^* is equivalent to the infinite horizon cost (2.8),

(iv). the optimal predicted input sequence computed at time k is feasible (i.e. satisfies constraints) for the optimization problem at time $k+1$,

then the optimal predicted cost is non-increasing and satisfies

$$J_{k+1}^* - J_k^* \leq -\Big(\|x_k\|_Q^2 + \|u_k\|_R^2\Big) \tag{3.1}$$

along closed-loop trajectories.

Given condition (iv), the inequality (3.1) derives from the fact that the optimal cost at current time must be at least as small as the cost evaluated for the tail of the optimal predicted input sequence computed at the previous sample. To show this, denote the input sequence at time $k+1$ corresponding to the optimal prediction at time k as $\tilde{\mathbf{u}}_{k+1}$:

$$\tilde{\mathbf{u}}_{k+1} = \{u^*_{1|k}, u^*_{2|k}, \ldots, Kx^*_{N|k}\}. \tag{3.2}$$

For convenience we will refer to this sequence as the **tail** (see Fig. 18). Note that Nth element of $\tilde{\mathbf{u}}_{k+1}$ is equal to $u^*_{N|k}$ and is therefore defined by the state feedback law $u = Kx$, which is assumed to be employed over the mode 2 horizon in the definition of the terminal weight $\overline{Q}$. Since $J(x_k, \mathbf{u}_k)$ is an infinite horizon cost, the value $\tilde{J}_{k+1} = J(x_{k+1}, \tilde{\mathbf{u}}_{k+1})$ associated with $\tilde{\mathbf{u}}_{k+1}$ is simply J^*_k minus the term relating to time k, i.e.

$$\begin{aligned}
\tilde{J}_{k+1} = J(x_{k+1}, \tilde{\mathbf{u}}_{k+1}) &= \sum_{i=1}^{N} \big(\|x_{i|k}\|^2_Q + \|u^*_{i|k}\|^2_R\big) + \|x(k+N+1|k)\|^2_{\overline{Q}} \\
&= \sum_{i=1}^{\infty} \big(\|x_{i|k}\|^2_Q + \|u^*_{i|k}\|^2_R\big) \\
&= \sum_{i=0}^{\infty} \big(\|x_{i|k}\|^2_Q + \|u^*_{i|k}\|^2_R\big) - \big(\|x_k\|^2_Q + \|u_k\|^2_R\big) \\
&= J^*_k - \big(\|x_k\|^2_Q + \|u_k\|^2_R\big).
\end{aligned}$$

But $\tilde{\mathbf{u}}_{k+1}$ is suboptimal at time $k+1$, so the optimal value satisfies

$$\begin{aligned}
J^*_{k+1} &\le \tilde{J}_{k+1} \\
&= J^*_k - \big(\|x_k\|^2_Q + \|u_k\|^2_R\big),
\end{aligned}$$

which implies (3.1).

By combining conditions (i) or (ii) with (iii) and (iv), and using Theorems 3.1-3.3, we get the following result.

> If $J(x_k, \mathbf{u}_k)$ is an infinite horizon cost and $(A, Q^{1/2})$ is observable, then $x = 0$ is a stable equilibrium for the closed-loop system and x_k converges asymptotically to zero, provided that the tail $\tilde{\mathbf{u}}_k$ is feasible for all $k > 0$.

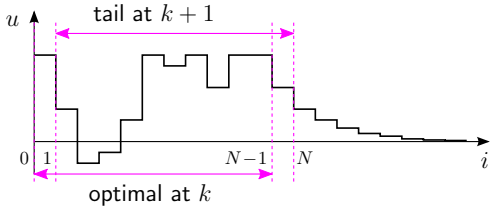


Figure 18: The optimal prediction $\mathbf{u}^*_k$ at time k and the tail $\tilde{\mathbf{u}}_{k+1}$ at time $k+1$.

3.2 Terminal constraints for recursive feasibility[12]

The guarantees of closed-loop stability and convergence derived in the previous section rely on the assumption that the predictions generated by the tail, $\tilde{\mathbf{u}}_k$, satisfy constraints at each time $k = 1, 2, \ldots$. From the definition (3.2) it is clear that $\tilde{\mathbf{u}}_{k+1}$ satisfies constraints over the first $N-1$ sampling intervals of the prediction horizon, since the optimal predictions at time k, namely $\{u^*_{1|k}, \ldots, u^*_{N-1|k}\}$, necessarily satisfy constraints. However for $\tilde{\mathbf{u}}_{k+1}$ to be feasible at $k+1$, we also need the Nth element of $\tilde{\mathbf{u}}_{k+1}$ (i.e. $u^*_{N|k} = Kx^*_{N|k}$) to satisfy constraints, and this requires extra constraints to be introduced in the MPC optimization at k.

Extra constraints that are introduced to ensure feasibility of the tail are known as **terminal constraints** since they apply to mode 2 predictions (which are governed by time-invariant feedback), and are therefore equivalent to constraints on the terminal state prediction $x_{N|k}$. For convenience we denote the region in which $x_{N|k}$ must lie in order to satisfy given terminal constraints as Ω (Fig. 19).

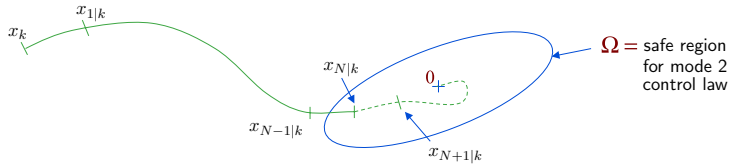


Figure 19: The terminal constraint set Ω.

[12] Reading: *Kouvaritakis & Cannon* §2.4, §2.5

The terminal constraints must be constructed so as to ensure feasibility of the MPC optimization recursively, i.e. so that the tail predictions necessarily satisfy constraints, including the terminal constraints themselves. Consider for example the case of input and state constraints: $\underline{u} \leq u \leq \overline{u}$ and $\underline{x} \leq x \leq \overline{x}$. To ensure that the tail satisfies constraints over the first N steps of the prediction horizon at time $k+1$, we need to include a terminal constraint

$$x_{N|k} \in \Omega \quad \Longrightarrow \quad \begin{cases} \underline{u} \leq Kx_{N|k} \leq \overline{u} \\ \underline{x} \leq x_{N|k} \leq \overline{x} \end{cases} \tag{3.3}$$

in the MPC optimization. But the predictions generated by $\tilde{\mathbf{u}}_{k+1}$ must also satisfy the terminal constraint $x_{N+1|k+1} \in \Omega$ in the MPC optimization at time $k+1$, and this is equivalent to an additional constraint on the predictions at time k:

$$x_{N+1|k} \in \Omega \tag{3.4}$$

The conditions of (3.3) and (3.4) lead to the following results.

Recursive feasibility. Necessary and sufficient conditions for the predictions generated by the tail $\tilde{\mathbf{u}}_{k+1}$ to be feasible at time $k+1$ whenever the MPC optimization at time k is feasible are:

(i). system constraints are instantaneously satisfied at all points in Ω, i.e. (3.3) is satisfied

(ii). Ω is invariant in mode 2, which is equivalent to requiring that

$$(A+BK)x_{N|k} \in \Omega \quad \text{for all} \quad x_{N|k} \in \Omega$$

If Ω satisfies (i) and (ii), then the MPC optimization:

$$\begin{aligned} \underset{\mathbf{u}_k}{\text{minimize}} \quad & J(x_k, \mathbf{u}_k) = \sum_{i=0}^{N-1} \left(\|x_{i|k}\|_Q^2 + \|u_{i|k}\|_R^2 \right) + \|x_{N|k}\|_Q^2 \\ \text{subject to} \quad & \underline{u} \leq u_{i|k} \leq \overline{u}, \quad i = 0, 1, \ldots, N-1 \\ & \underline{x} \leq x_{i|k} \leq \overline{x}, \quad i = 1, \ldots, N-1 \\ & x_{N|k} \in \Omega \end{aligned} \tag{3.5}$$

is guaranteed to be feasible at all times $k > 0$ provided it is feasible at $k = 0$.

This means that the **region of attraction** (denoted $\mathcal{S}_\Omega$) for the MPC law is given by the set of all initial conditions from which it is possible to drive the state predictions into Ω over the mode 1 horizon, i.e.

$$\mathcal{S}_\Omega = \left\{ \begin{array}{ll} x_0 \ : \ \exists \mathbf{u}_0 \ \text{ satisfying } & x_{N|0} \in \Omega \\ & \underline{u} \leq u_{i|0} \leq \overline{u} \ \ \forall i \in \{0, \ldots, N-1\} \\ & \underline{x} \leq x_{i|0} \leq \overline{x} \ \ \forall i \in \{1, \ldots, N\} \end{array} \right\} \tag{3.6}$$

3.3 Constraint checking horizon[13]

The set of feasible initial plant states $\mathcal{S}_\Omega$ can be enlarged by increasing N (the horizon within which the predicted state must reach Ω), or by enlarging Ω (the target set). For any given N it is desirable to make $\mathcal{S}_\Omega$ as large as possible in order to maximize the allowable operating region of the MPC law, and this implies that Ω should be made as large as possible. This section is concerned with the design of the terminal constraints defining Ω, and it is shown below that the largest possible Ω is obtained if the terminal constraints are defined simply by forcing the mode 2 predictions to satisfy the system input and state constraints over a sufficiently long horizon.

The method of imposing terminal constraints by imposing system constraints on mode 2 predictions is based on two facts:

- The conditions derived in the previous section for ensuring recursive feasibility require system constraints to be satisfied over the entire mode 2 horizon, i.e.
  $\underline{u} \le K(A+BK)^i x_{N|k} \le \overline{u}$ and $\underline{x} \le (A+BK)^i x_{N|k} \le \overline{x}$
  for all $i \ge 0$.
  Hence the largest possible Ω is defined by
  $$\Omega = \{x:\ \underline{u} \le K(A+BK)^i x \le \overline{u},\ \ \underline{x} \le (A+BK)^i x \le \overline{x},\ \ i=0,1,\ldots\}.$$
- Constraint satisfaction over the *infinite* mode 2 horizon can be determined by checking constraints over a long enough *finite* horizon.

To justify the claim in the second point above, consider the case of input constraints alone: $\underline{u} \le u \le \overline{u}$, and let Π_M be the set of initial conditions for which input constraints are satisfied over a horizon of length M under the mode 2 feedback law $u = Kx$, i.e.

$$\Pi_M = \{x:\ \underline{u} \le K(A+BK)^i x \le \overline{u},\ \ \ i=0,1,\ldots,M\}. \tag{3.7}$$

Then the second fact above can be equivalently stated as

$$\Pi_\infty = \Pi_{N_c} \tag{3.8}$$

[13] Reading: *Kouvaritakis & Cannon* §2.4

for some finite horizon N_c, and this is a consequence of the assumption that $(A+BK)$ is a stable matrix ($|\lambda\{A+BK\}| < 1$) so that $(A+BK)^i \to 0$ as $i \to \infty$. As a result, for any given x, the perpendicular distance d_i of the hyperplane defined by $K(A+BK)^i x = \overline{u}$ from $x = 0$ tends to infinity as $i \to \infty$:

$$d_i = \frac{\overline{u}}{\|K(A+BK)^i\|_2} \to \infty \ \text{ as } i \to \infty.$$

But Π_∞ is finite (assuming that $(A+BK, K)$ is an observable pair) since constraints must be violated at some future time whenever x_0 is large enough, and therefore the strip of state space defined by

$$\{x:\ \underline{u} \le K(A+BK)^i x \le \overline{u}\}$$

necessarily contains Π_∞ for all $i > N_c$ for some finite N_c. Therefore all points $x \in \Pi_{N_c}$ lie within Π_∞, and this implies (3.8).

Example 3.2. For the constrained system considered in Example 3.1, suppose that the mode 2 feedback law is defined as the LQ-optimal controller for cost matrices $Q = C^\top C$ and $R = 1$: $u_k = Kx_k$, $K = \begin{bmatrix}-1.19 & -7.88\end{bmatrix}$. Then Π_M, $M = 0, 1, 2, \ldots$ are given by:

$$\begin{aligned}
\Pi_0 &= \{x:\ -1 \le \begin{bmatrix}-1.19 & -7.88\end{bmatrix} x \le 1\} \\
\Pi_1 &= \Pi_0 \cap \{x:\ -1 \le \begin{bmatrix}-0.5702 & -4.9754\end{bmatrix} x \le 1\} \\
\Pi_2 &= \Pi_1 \cap \{x:\ -1 \le \begin{bmatrix}-0.1621 & -2.7826\end{bmatrix} x \le 1\} \\
\Pi_3 &= \Pi_2 \cap \{x:\ -1 \le \begin{bmatrix}0.0818 & -1.2427\end{bmatrix} x \le 1\} \\
&\ \ \vdots
\end{aligned}$$

and in this case $\Pi_M = \Pi_4$ for all $M > 4$ (Fig. 20), which therefore implies $N_c = 4$. ◊

The minimum constraint checking horizon N_c that is required for a given plant model and mode 2 feedback law can be computed offline using the following property.

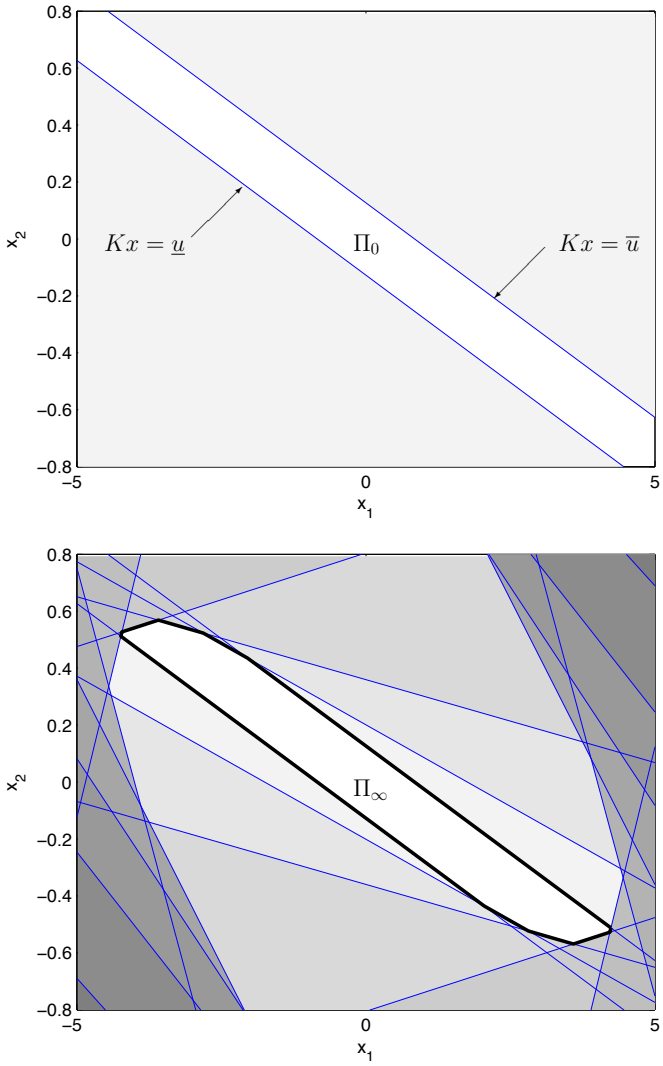


Figure 20: Upper plot: Π_0. Lower plot: $\Pi_4 = \Pi_5 = \Pi_6 = \cdots = \Pi_\infty$.

Theorem 3.4 (Constraint checking horizon). $\Pi_{N_c} = \Pi_\infty$ *if and only if:* $x \in \Pi_{N_c+1}$ *whenever* $x \in \Pi_{N_c}$.

This powerful result can be proved by induction:

(a). First note that, if $x \in \Pi_{N_c+1}$ whenever $x \in \Pi_{N_c}$, then $\Pi_{N_c} \subseteq \Pi_{N_c+1}$. But we also have $\Pi_{N_c+1} = \Pi_{N_c} \cap \{x : \underline{u} \leq K(A+BK)^{N_c+1}x \leq \overline{u}\}$, implying that $\Pi_{N_c+1} \subseteq \Pi_{N_c}$ by definition. Thus we conclude that if $x \in \Pi_{N_c+1}$ whenever $x \in \Pi_{N_c}$, then $\Pi_{N_c} = \Pi_{N_c+1}$.

(b). If $x \in \Pi_{N_c} = \Pi_{N_c+1}$, then $\underline{u} \leq Kx \leq \overline{u}$ and $(A+BK)x \in \Pi_{N_c} = \Pi_{N_c+1}$, and hence $x \in \Pi_{N_c+2}$.

(c). By repeatedly applying the argument in (b) we conclude that $x \in \Pi_{N_c} = \Pi_{N_c+1}$ implies $x \in \Pi_{N_c+i}$, for all $i = 2, 3, \ldots$, and therefore $x \in \Pi_\infty$.

Because of Theorem 3.4, it is only necessary to check whether the constraints over the first M steps of mode 2 ensure that the constraints are satisfied at the $M+1$st step in order to determine N_c. For the case of input constraints, this is equivalent to checking whether the following statement is true:

$$\underline{u} \leq K(A+BK)^i x \leq \overline{u}, \quad i = 0, 1, \ldots, M \quad \Longrightarrow \quad \underline{u} \leq K(A+BK)^{M+1} x \leq \overline{u}$$

If the input u has dimension n_u, this is done by computing the maximum over x of each element of $K(A+BK)^{M+1}x$ subject to constraints:

$$u_{\max,j} = \max_x K_j(A+BK)^{M+1}x \ \text{ s.t. } \ \underline{u} \leq K(A+BK)^i x \leq \overline{u}, \quad i = 0, 1, \ldots, M$$

$$u_{\min,j} = \min_x K_j(A+BK)^{M+1}x \ \text{ s.t. } \ \underline{u} \leq K(A+BK)^i x \leq \overline{u}, \quad i = 0, 1, \ldots, M$$

for $j = 1, 2, \ldots, n_u$, (requiring the solution of $2n_u$ linear programming problems). Constraints at the $M+1$st step are satisfied if $u_{\max,j} \leq \overline{u}_j$ and $u_{\min,j} \geq \underline{u}_j\}$, for all j. Thus the minimum constraint checking horizon N_c can be computed by solving a finite number of linear programs. The algorithm is summarized for the case of $n_u = 1$ in Figure 21.

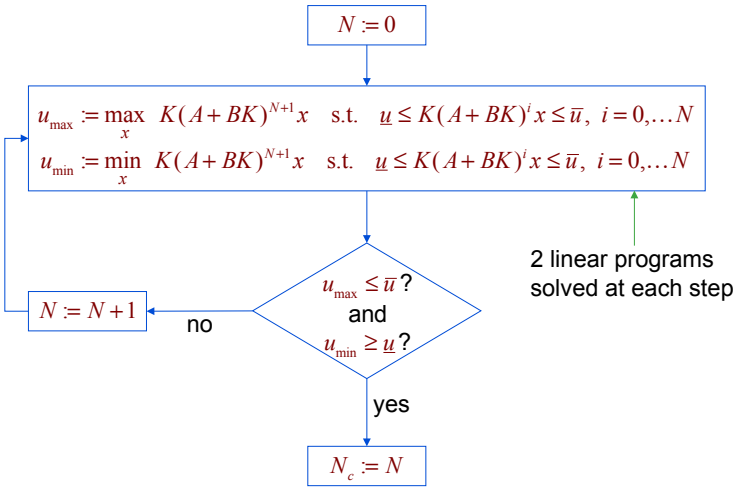


Figure 21: Algorithm for computing the mode 2 constraint checking horizon N_c for the case of input constraints $\underline{u} \le u \le \overline{u}$, with 1-dimensional input u.

Example 3.3. Applying the algorithm of Figure 21 to the system and mode 2 feedback law considered in Example 3.2, we get:

$$u_{\max} = \max_x \Big\{ \begin{bmatrix} -0.1621 & -2.7826 \end{bmatrix} x \;\; \text{s.t.} \;\; \begin{bmatrix} -1 \\ -1 \end{bmatrix} \le \begin{bmatrix} -1.1877 & -7.8773 \\ -0.5702 & -4.9754 \end{bmatrix} x \le \begin{bmatrix} 1 \\ 1 \end{bmatrix} \Big\}$$
$$= 1.9801$$
$$\implies N_c > 1$$
$$u_{\max} = \max_x \Big\{ \begin{bmatrix} 0.0818 & -1.2427 \end{bmatrix} x \;\; \text{s.t.} \;\; \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix} \le \begin{bmatrix} -1.1877 & -7.8773 \\ -0.5702 & -4.9754 \\ -0.1621 & -2.7826 \end{bmatrix} x \le \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \Big\}$$
$$= 1.2570$$
$$\implies N_c > 2$$

$$u_{\max} = \max_x \Big\{ \begin{bmatrix} 0.2061 & -0.2466 \end{bmatrix} x \;\; \text{s.t.} \;\; \begin{bmatrix} -1 \\ -1 \\ -1 \\ -1 \end{bmatrix} \le \begin{bmatrix} -1.1877 & -7.8773 \\ -0.5702 & -4.9754 \\ -0.1621 & -2.7826 \\ 0.0818 & -1.2427 \end{bmatrix} x \le \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \Big\}$$
$$= 1.0152$$
$$\implies N_c > 3$$
$$u_{\max} = \max_x \Big\{ \begin{bmatrix} 0.2498 & 0.3308 \end{bmatrix} x \;\; \text{s.t.} \;\; \begin{bmatrix} -1 \\ -1 \\ -1 \\ -1 \\ -1 \end{bmatrix} \le \begin{bmatrix} -1.1877 & -7.8773 \\ -0.5702 & -4.9754 \\ -0.1621 & -2.7826 \\ 0.0818 & -1.2427 \\ 0.2061 & -0.2466 \end{bmatrix} x \le \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \Big\}$$
$$= 0.8893$$
$$u_{\min} = -u_{\max} = -0.8893$$
$$\implies N_c = 4$$

Hence $N_c = 4$, which is in agreement with the plots in Fig 20. ◊

3.4 Closed-loop performance[14]

With the terminal constraint $x_{N|k} \in \Omega = \Pi_{N_c}$ incorporated in (3.5), the optimization solved at each sample to determine the MPC law $u_k = u^*_{0|k}$ can be expressed:

$$\begin{aligned} \underset{\mathbf{u}}{\text{minimize}} \;\; & J(x_k, \mathbf{u}_k) = \sum_{i=0}^{N-1} \big(\|x_{i|k}\|_Q^2 + \|u_{i|k}\|_R^2 \big) + \|x_{N|k}\|_{\overline{Q}}^2 \\ \text{subject to} \;\; & \underline{u} \le u_{i|k} \le \overline{u}, \quad i = 0, 1, \dots, N-1 \\ & \underline{x} \le x_{i|k} \le \overline{x}, \quad i = 1, \dots, N-1 \\ & \underline{u} \le K(A+BK)^i x_{N|k} \le \overline{u}, \quad i = 0, 1, \dots, N_c \\ & \underline{x} \le (A+BK)^i x_{N|k} \le \overline{x}, \quad i = 0, 1, \dots, N_c \end{aligned} \tag{3.9}$$

[14] Reading: *Kouvaritakis & Cannon* §2.5, §2.6

Note that:

- This is a quadratic programming (QP) problem, and the extra computational burden due to the inclusion of terminal constraints is small since these constraints are linear (the main factor determining the computational load of QP is the number of free variables).
- The size of the set of feasible initial conditions $\mathcal{S}_\Omega$ increases as N is increased.
- The optimal predicted cost J_k^* is reduced as N is increased.

The performance of the closed-loop system:

$$J_{\rm cl} = \sum_{k=0}^{\infty} \Big(\|x_k\|_Q^2 + \|u_k\|_R^2 \Big) \tag{3.10}$$

is likely to improve as N is increased due to the reduction in predicted cost. However, for given x_0, there exists some finite value N_∞ (which depends on x_0) such that no improvement in closed-loop performance can be obtained for $N > N_\infty$. This is because the terminal constraints must be inactive for sufficiently large N, and there cannot be any reduction in cost if N is increased further. Therefore the optimal performance for an infinite number of degrees of freedom is obtained if $N = N_\infty$. This ideal optimal performance is known as **constrained LQ-optimal** performance.

Example 3.4. For the system and cost matrices considered in Example 3.2, the table below gives the variation with N of the closed-loop performance of the MPC law defined by (3.9) (with $N_c = 4$). In each case the initial state is $x_0 = (-7.0, 0.5)$. The closed-loop responses in Figure 22 demonstrates that $N_\infty = 10$ for this initial condition.

N	5	6	7	10	> 10
J_0^*	295.2	287.7	286.8	286.6	286.6
$J_{\rm cl}$	286.7	286.7	286.6	286.6	286.6

$\Diamond$

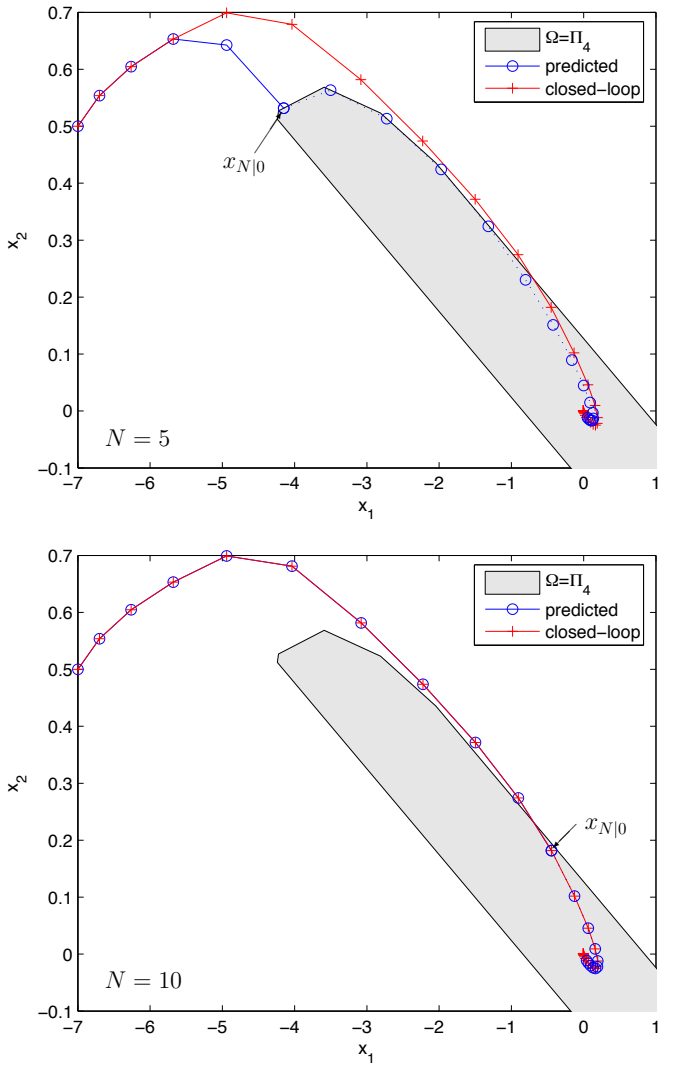


Figure 22: Predicted and closed-loop state trajectories in Example 3.3. Upper plot: $N = 5$. Lower plot: $N = 10$.

Summary of Section 3

- If $\overline{Q}$ and N_c are determined (offline) by solving the Lyapunov equation (2.10), and by computing the required N_c so that $\Pi_\infty = \Pi_{N_c}$, then the objective J_k is equal to the cost evaluated over the infinite prediction horizon, and the constraints in (3.9) ensure that predictions satisfy the system constraints at all future times: $k, k+1, k+2, \ldots$.
- If a solution to (3.9) exists initially (at time $k = 0$), then the MPC optimization will be feasible at all times $k > 0$, and the closed-loop system is therefore stabilized from all initial conditions x_0 in $\mathcal{S}_\Omega$.
- Constrained LQ-optimal closed-loop performance is obtained with a sufficiently long mode 1 horizon N.

4 Disturbances and robustness [15]

This section considers methods of providing robustness to model uncertainty in the form of unknown disturbances acting on the plant. Integral action and robust constraint handling are described for the case of constant disturbances with known upper and lower bounds. Extensions to time-varying disturbances are also briefly discussed.

The discussion in this section focuses on linear plant models with unknown additive disturbances of the form

$$x_{k+1} = Ax_k + Bu_k + Dw_k, \quad y_k = Cx_k. \tag{4.1}$$

Here D is a known matrix and w_k is an unknown disturbance[16]. For simplicity we assume the state x_k can be measured and we ignore sensor noise.

Model uncertainty resulting from this kind of disturbance typically arises in control applications as a result of: (i) incomplete knowledge of the plant model, (ii) random processes in the system dynamics, or (iii) the use of observer state estimates in prediction models (although in this case the uncertainty lies in the estimated state). A particular instance of (i) (which is extremely common in setpoint tracking problems) is due to uncertainty in the plant d.c. gain.

To understand how disturbances arise in the setpoint tracking problem, consider the problem of driving the output y_k to a constant setpoint y^0. The required steady state input is u^0, where

$$x^0 = Ax^0 + Bu^0, \quad y^0 = Cx^0 \implies y^0 = C(I-A)^{-1}Bu^0.$$

Hence u^0 is unique if the matrix $C(I-A)^{-1}B$ of steady state gains is invertible (assuming for simplicity that the number of plant inputs and outputs are equal), and

$$u^0 = [C(I-A)^{-1}B]^{-1}y^0 \qquad x^0 = (I-A)^{-1}Bu^0.$$

[15] Reading: *Kouvaritakis & Cannon* §3.1–3.3

[16] The disturbance w_k in (4.1) is a **process** disturbance rather than an **output** disturbance; an example of the latter is the sensor noise term v in a measurement $y_k = Cx_k + v_k$.

Given the steady state values x^0 and u^0, a change of variables:

$$x_k^\delta = x_k - x^0, \quad u_k^\delta = u_k - u^0, \quad x_{k+1}^\delta = Ax_k^\delta + Bu_k^\delta$$

converts the setpoint tracking problem into a regulation problem:

$$\begin{aligned} \underset{\mathbf{u}^\delta}{\text{minimize}} \quad & J(x_k, \mathbf{u}_k) = \sum_{i=0}^{N-1}\left(\|x_{i|k}^\delta\|_Q^2 + \|u_{i|k}^\delta\|_R^2\right) + \|x_{N|k}^\delta\|_{\overline{Q}}^2 \\ \text{subject to} \quad & \underline{u} \le u_{i|k}^\delta + u^0 \le \overline{u}, \quad i = 0, 1, \ldots, N+N_c \\ & \underline{x} \le x_{i|k}^\delta + x^0 \le \overline{x}, \quad i = 1, \ldots, N+N_c \end{aligned}$$

This optimization problem can be used to define an MPC law by using the methods of Sections 2.3 and 3.3 to design the terminal weighting matrix $\overline{Q}$ and terminal constraints via a constraint checking horizon N_c. The optimization can be cast in terms of the predicted sequence $\mathbf{u}_k^\delta = \{u_{0|k}^\delta, \ldots, u_{N-1|k}^\delta\}$ as a QP problem. Having computed the solution, $\mathbf{u}_k^{\delta *} = \{u_{0|k}^{\delta *}, \ldots, u_{N-1|k}^{\delta *}\}$, at time k, the control law is implemented via

$$u_k = u_{0|k}^{\delta *} + u^0.$$

However, if the required steady state input u^0 is not known exactly, and an estimate $\hat{u}^0$ is used instead in the definition of the control law,

$$u_k = u_{0|k}^{\delta *} + \hat{u}^0,$$

then the system governing x^δ will contain a constant disturbance term, since

$$u_k^\delta = u_{0|k}^{\delta *} + (\hat{u}^0 - u^0) \implies x_{k+1}^\delta = Ax_k^\delta + Bu_{0|k}^{\delta *} + B(\hat{u}^0 - u^0).$$

As a result there will be a non-zero steady state error because $x^\delta = 0$ will not be an equilibrium of the closed-loop system, i.e.

$$\text{if} \quad x_k^\delta = 0 \quad \text{then} \quad u_{0|k}^{\delta *} = 0, \quad \text{so} \quad x_{k+1}^\delta = B(\hat{u}^0 - u^0) \neq 0.$$

In fact it is possible to determine the steady state error since the MPC law converges to the unconstrained LQ feedback law (assuming that constraints are not active at steady state), i.e. $u_{0|k}^{\delta *} = Kx_{0|k}^\delta$, and the closed-loop system

therefore becomes:

$$x_{k+1}^{\delta} = (A+BK)x_k^{\delta} + B(\hat{u}^0 - u^0)$$
$$\implies \begin{cases} x_k^{\delta} \to (I-A-BK)^{-1}B(\hat{u}^0-u^0) \\ y_k \to y^0 + C(I-A-BK)^{-1}B(\hat{u}^0-u^0) \end{cases}$$

as $k \to \infty$, giving a steady state error in y of $C(I-A-BK)^{-1}B(\hat{u}^0-u^0)$.

4.1 MPC with integral action

A common approach to removing steady state error due to a constant disturbance is to introduce integral action into the controller. This can be done simply by including an integrator in the MPC law. Assuming for example that the target setpoint for y is $y^0 = 0$, so that the tracking error is $y - y^0 = y$, we can introduce integral action with a controller of the form

$$\boxed{\begin{aligned} u_k &= u_{0|k}^* + K_I v_k \\ v_{k+1} &= v_k + y_k \end{aligned}}$$

where K_I is an integral gain and v is the integrator state. Note that, if this modification is not accounted for in the cost and constraints (how to do this is described below and in section 4.2), then closed-loop performance and stability will no longer be guaranteed. However, *if the closed-loop system is stable*, then the steady state error must be zero since

$$u_k \to u^{ss} \implies v_k \to v^{ss} \implies y_k \to 0$$

as $k \to \infty$, for some finite u^{ss}, v^{ss}.

If there are no system constraints, then integral action can be introduced without compromising closed-loop stability just by modifying the cost to include a penalty on the predicted integrator state:

$$J(x_k, \mathbf{u}_k) = \sum_{i=0}^{\infty} \left(\|z_{i|k}\|_{Q_z}^2 + \|u_{i|k}\|_R^2 \right). \tag{4.2}$$

Here z is the state of an augmented system, which contains both the plant and integrator state:

$$z_{0|k} = \begin{bmatrix} x_k \\ v_k \end{bmatrix} \quad z_{i+1|k} = \begin{bmatrix} A & 0 \\ C & I \end{bmatrix} z_{i|k} + \begin{bmatrix} B \\ 0 \end{bmatrix} u_{i|k} \tag{4.3}$$

and

$$Q_z = \begin{bmatrix} Q & 0 \\ 0 & Q_I \end{bmatrix}$$

for some weighting matrices Q, Q_I. The unconstrained LQ-optimal feedback for the cost (4.2) has the form

$$u_{0|k}^* = \begin{bmatrix} K & K_I \end{bmatrix} z_{0|k} = Kx_k + K_I v_k \tag{4.4}$$

where the relative size of the integral gain K_I is determined by Q_I.

The (unconstrained optimal, linear) control law (4.4) applied to the disturbed system (4.1) leads to a closed-loop system which converges to zero steady state error regardless of the size of w because:

- For $w = 0$: the optimal value of the cost (4.2) satisfies
$$J_{k+1}^* - J_k^* = -\|z_k\|_{Q_z}^2 - \|u_k\|_R^2$$
and the Lyapunov stability analysis of section 3.1 therefore implies that $z = 0$ is asymptotically stable provided the pair
$$\left(\begin{bmatrix} A & 0 \\ C & I \end{bmatrix}, \begin{bmatrix} Q^{1/2} & 0 \\ 0 & Q_I^{1/2} \end{bmatrix} \right)$$
is observable.
- For $w \neq 0$: the optimal cost is not guaranteed to decrease along closed-loop trajectories (the predicted and closed-loop behaviour will differ since $w = 0$ is assumed in the predictions), however the stability guarantee for $w = 0$ implies that the poles of the (linear) closed-loop system under (4.4):
$$z_{k+1} = \begin{bmatrix} A+BK & BK_I \\ C & I \end{bmatrix} z_k + \begin{bmatrix} D \\ 0 \end{bmatrix} d$$
lie inside the unit circle. Therefore $z_k \to z^{ss}$ for some finite z^{ss}, implying $y_k \to y^0$.

4.2 Robustness to constant disturbances

If $w = 0$ is assumed in the prediction model (4.1), then incorporating constraints on the control input and system state into a constrained minimization of the cost (4.2) leads to an MPC optimization problem of the form

$$\begin{aligned} \underset{\mathbf{u}}{\text{minimize}} \quad & J(x_k, \mathbf{u}_k) = \sum_{i=0}^{N-1} \Big(\|z_{i|k}\|_{Q_z}^2 + \|u_{i|k}\|_R^2 \Big) + \|z_{N|k}\|_{\overline{Q}_z}^2 \\ \text{subject to} \quad & \underline{u} \le u_{i|k} \le \overline{u}, \quad i = 0, 1, \ldots, N + N_c \\ & \underline{x} \le x_{i|k} \le \overline{x}, \quad i = 1, \ldots, N + N_c \end{aligned} \tag{4.5}$$

Here the predicted values of z are governed by the augmented system (4.3), and $\overline{Q}_z, N_c$ are determined for the mode 2 feedback law $u = Kx + K_I v$. Since the disturbance w is not accounted for in predictions, there is clearly no guarantee of recursive feasibility and therefore no closed-loop stability guarantee. However *if the closed-loop system is stable*, the steady state error must be zero since z_k must converge to a finite steady state value.

4.2.1 Robust constraint satisfaction

To obtain a guarantee that the MPC optimization problem has a solution at all times $k > 0$ provided it is feasible at $k = 0$, we need to ensure that the predictions satisfy constraints for all possible disturbances. For the case of constant disturbance vectors with elements that lie between known upper and lower bounds:

$$w_k = w, \qquad \underline{w} \le w \le \overline{w} \tag{4.6}$$

the augmented state predictions can be expressed

$$z_{i|k} = \hat{z}_{i|k} + e_{i|k}$$

$$\hat{z}_{i|k} = \begin{bmatrix} A & 0 \\ C & I \end{bmatrix}^i z_k + \mathcal{C}_i \mathbf{u}_k, \qquad e_{i|k} = \sum_{j=0}^{i-1} \begin{bmatrix} A & 0 \\ C & I \end{bmatrix}^j \begin{bmatrix} D \\ 0 \end{bmatrix} w$$

where $\hat{z}_{i|k}$ denote the predictions for $w = 0$ and $e_{i|k}$ accounts for the effects of non-zero w. Hence for robust constraint satisfaction we require

$$\begin{aligned} \underline{u} \le u_{i|k} \le \overline{u} \\ \underline{x} \le \begin{bmatrix} I & 0 \end{bmatrix} \hat{z}_{i|k} + \begin{bmatrix} I & 0 \end{bmatrix} e_{i|k} \le \overline{x} \end{aligned} \tag{4.7}$$

for $i = 0, 1, \ldots$ and all w in the interval $[\underline{w}, \overline{w}]$. Hence the constraints in (4.7) are satisfied for all w if and only if they are satisfied when the uncertain terms are replaced by their worst case values, namely

$$\underline{h}_i = \min_{\underline{w} \le d \le \overline{w}} \begin{bmatrix} I & 0 \end{bmatrix} e_{i|k}, \qquad \overline{h}_i = \max_{\underline{w} \le d \le \overline{w}} \begin{bmatrix} I & 0 \end{bmatrix} e_{i|k}$$

(where the maximizations and minimizations apply elementwise, one for each element of $[I \;\; 0] e_{i|k}$). Since $e_{i|k}$ depends linearly on w, the vectors of bounds $\underline{h}_i$, $\overline{h}_i$ can be computed by solving a set of linear programs, e.g.

$$\underline{h}_i = \min_{\underline{w} \le d \le \overline{w}} \begin{bmatrix} I & 0 \end{bmatrix} \sum_{j=0}^{i-1} \begin{bmatrix} A & 0 \\ C & I \end{bmatrix}^j \begin{bmatrix} D \\ 0 \end{bmatrix} w$$

$$\overline{h}_i = \max_{\underline{w} \le d \le \overline{w}} \begin{bmatrix} I & 0 \end{bmatrix} \sum_{j=0}^{i-1} \begin{bmatrix} A & 0 \\ C & I \end{bmatrix}^j \begin{bmatrix} D \\ 0 \end{bmatrix} w$$

for $i = 1, 2, \ldots$. The robust constraints (4.7) are then equivalent to tightened constraints on the nominal predictions of the state:

$$\begin{aligned} \underline{u} \le u_{i|k} \le \overline{u} \\ \begin{bmatrix} I & 0 \end{bmatrix} \underline{x} - \underline{h}_i \le \begin{bmatrix} I & 0 \end{bmatrix} \hat{z}_{i|k} \le \overline{x} - \overline{h}_i. \end{aligned}$$

It is convenient to think of uncertain component of the predicted state sequence, $\{e_{0|k}, e_{1|k}, e_{2|k}, \ldots\}$, as a lying within a *tube* defined by a collection of sets, one for each time-step of the prediction horizon. Note that this tube does not depend on the initial state z_k, and the bounds $\underline{h}_i$, $\overline{h}_i$ can therefore be computed offline.

4.2.2 Pre-stabilized predictions

If the open-loop system is unstable, then the tube containing the predictions of e diverges, and hence the bounds $\underline{h}_i$, $\overline{h}_i$ grow in magnitude as i increases. This

can severely restrict the feasible initial condition set $\mathcal{S}_\Omega$ of a robust predictive control law in which constraints must be satisfied for all possible disturbances. The problem can be overcome by expressing predictions as perturbations on a stabilizing linear feedback law, which we take as the mode 2 feedback law:

$$u_{i|k} = \begin{bmatrix} K & K_I \end{bmatrix} z_{i|k} + c_{i|k},$$
$$c_{i|k} = \begin{cases} \text{optimization variables}, & i = 0, 1, \ldots, N-1 \\ 0, & i \geq N \end{cases}$$

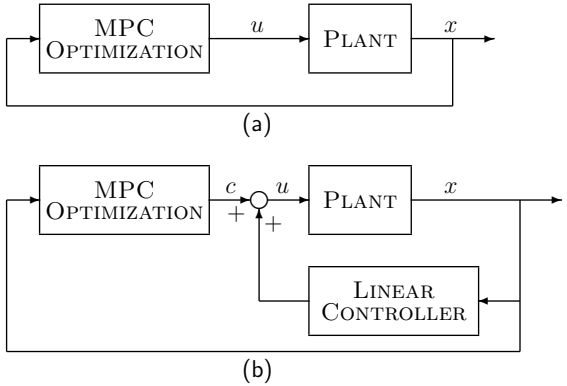


Figure 23: The feedback structure of nominal MPC (a) and the feedback structure of MPC with pre-stabilized predictions (b)

With this modification, the state predictions are given by

$$\begin{aligned} z_{i|k} &= \hat{z}_{i|k} + e_{i|k} \\ \hat{z}_{i|k} &= \Psi^i z_k + \mathcal{C}_i' \mathbf{c}_k \\ e_{i|k} &= \sum_{j=0}^{i-1} \Psi^j \Delta w \end{aligned} \tag{4.8}$$

where $\Psi = \begin{bmatrix} A+BK & BK_I \\ C & I \end{bmatrix}$ is a stable matrix and where $\Delta = \begin{bmatrix} D \\ 0 \end{bmatrix}$.

Therefore the tube containing e converges, and, for any given w, $e_{i|k}$ converges to a constant value as $i \to \infty$. The cost and constraints can be reformulated

in terms of the degrees of freedom in $\mathbf{c}_k$ using the approach described in section 2.

Example 4.1. Consider the constrained first order system with a constant unknown disturbance w:

$$\begin{aligned} &x_{k+1} = ax_k + u_k + w, \quad |w| \leq 1 \qquad a = 2 \\ &|x_k| \leq 2 \end{aligned}$$

(note that the open-loop system is unstable since $a > 1$). The effect of disturbances on state predictions is given by

$$e_{i|k} = \sum_{j=0}^{i-1} a^j w = (2^i - 1)w$$

and for robust constraint satisfaction, we therefore require the upper and lower bounds of Fig 24 to remain within constraints. Clearly this cannot be achieved for any mode 1 horizon $N > 1$.

However the pre-stabilized predictions: $u_{i|k} = -1.9x_{i|k} + c_{i|k}$ result in predictions

$$\begin{aligned} x_{i+1|k} &= 0.1x_{i|k} + c_{i|k} + w \\ e_{i|k} &= \sum_{j=0}^{i-1} 0.1^j w = (1 - 0.1^j)w/0.9 \end{aligned}$$

For robust constraint satisfaction we now require that the upper and lower bounds in Fig 25 remain within constraints, which can be satisfied for some $\mathbf{c}_0$ for all $i > 0$. ◊

4.2.3 Tubes and time-varying disturbances [17]

Robust constraint handling strategies for systems with time-varying disturbances within known bounds:

$$x_{k+1} = Ax_k + Bu_k + Dw_k, \quad \underline{w} \leq w_k \leq \overline{w} \tag{4.9}$$

[17] Reading: *Kouvaritakis & Cannon* §3.2–3.3

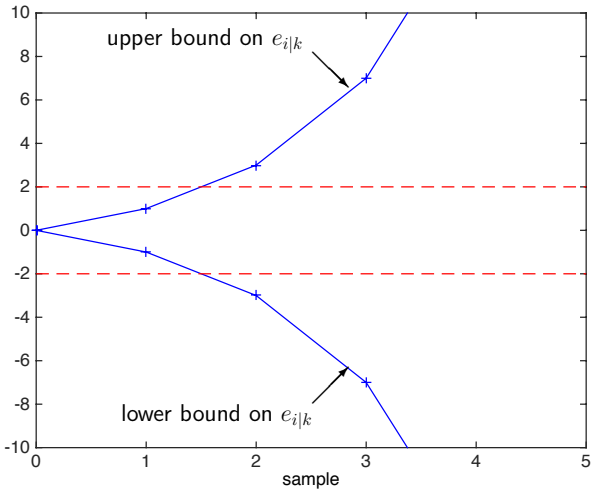


Figure 24: The tube containing the uncertainty e in the state predictions in Example 4.1.

can be derived using a similar approach to the one described above for constant disturbances. In this case, when integral action is incorporated and pre-stabilized predictions are used (as discussed in Sections 4.1 and 4.2.2), the uncertainty in predictions is governed by

$$e_{i+1|k} = \Psi e_{i|k} + \Delta w_k$$
$$\Psi = \begin{bmatrix} A+BK & BK_I \\ C & I \end{bmatrix}, \qquad \Delta = \begin{bmatrix} D \\ 0 \end{bmatrix},$$

with $e_{0|k} = 0$. Since $e_{i|k}$ does not depend on the plant state, the tube containing e can again be determined offline and the constraints of (4.7) are again equivalent to tightened constraints on the uncertainty-free predictions

$$\underline{u} - \underline{g}_i \le \begin{bmatrix} K & K_I \end{bmatrix} \hat{z}_{i|k} + c_{i|k} \le \overline{u} - \overline{g}_i$$
$$\underline{x} - \underline{h}_i \le \begin{bmatrix} I & 0 \end{bmatrix} \hat{z}_{i|k} \le \overline{x} - \overline{h}_i$$

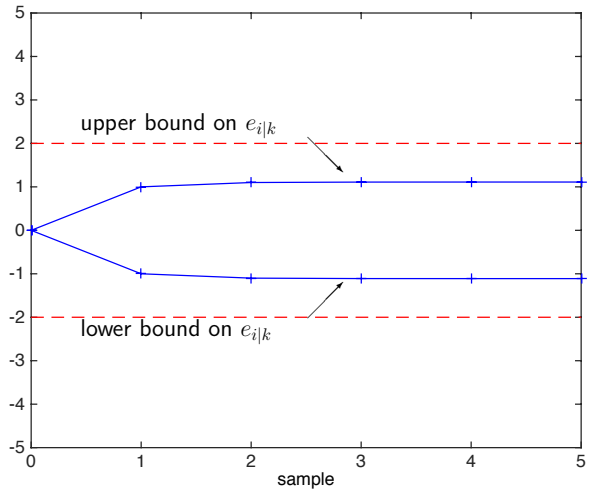


Figure 25: The stabilized tube containing the predicted uncertainty e for the case of pre-stabilized predictions in Example 4.1.

where $\underline{g}_i$, $\overline{g}_i$, $\underline{h}_i$, $\overline{h}_i$ are defined for $i = 1, 2, \ldots$ by a sequence of linear programs:

$$\underline{h}_i = \underline{h}_{i-1} + \min_{\underline{w} \le d \le \overline{w}} \begin{bmatrix} I & 0 \end{bmatrix} \Psi^{i-1} \Delta w, \qquad \overline{h}_i = \overline{h}_{i-1} + \max_{\underline{w} \le d \le \overline{w}} \begin{bmatrix} I & 0 \end{bmatrix} \Psi^{i-1} \Delta w$$
$$\underline{g}_i = \underline{g}_{i-1} + \min_{\underline{w} \le d \le \overline{w}} \begin{bmatrix} K & K_I \end{bmatrix} \Psi^{i-1} \Delta w, \qquad \overline{g}_i = \overline{g}_{i-1} + \max_{\underline{w} \le d \le \overline{w}} \begin{bmatrix} K & K_I \end{bmatrix} \Psi^{i-1} \Delta w$$

with $\underline{h}_0 = \overline{h}_0 = 0$ and $\underline{g}_0 = \overline{g}_0 = 0$.

The online optimization defining the MPC law therefore takes the form:

$$\begin{aligned} \underset{\mathbf{c}_k}{\text{minimize}} \quad & J(x_k, \mathbf{u}_k) \\ \text{subject to} \quad & \underline{u} - \underline{g}_i \le \begin{bmatrix} K & K_I \end{bmatrix} \hat{z}_{i|k} + c_{i|k} \le \overline{u} - \overline{g}_i \\ & \underline{x} - \underline{h}_i \le \begin{bmatrix} I & 0 \end{bmatrix} \hat{z}_{i|k} \le \overline{x} - \overline{h}_i \\ & \text{for } i = 0, 1, \ldots, N + N_c \end{aligned} \tag{4.10}$$

where $\hat{z}_{i|k}$ is given in terms of z_k and $\mathbf{c}_k$ by (4.8), and the control law is

$$u_k = \begin{bmatrix} K & K_I \end{bmatrix} z_k + c^*_{0|k}.$$

Two possible choices for the objective minimized in (4.10) are

- $J = J_{\max}$: an upper bound on the performance cost over all possible disturbance sequences;
- $J = \hat{J}$: the nominal cost (i.e. the cost that is obtained by assuming $w_{i|k} = 0$ at all prediction times $i = 0, 1, \ldots$).

In many cases the nominal cost can result in less conservative control than the upper-bound cost. However this depends on the likelihood of large disturbances occurring, and the cost must therefore be chosen by taking into account information on the probability distribution of stochastic disturbances.

Both approaches provide stability guarantees. In the case of the nominal cost, this derives from the property that, if $[K\ K_I]$ is chosen as the LQ-optimal gain for the unconstrained nominal system (as described in Section 4.1), then the optimal value of $\mathbf{c}_k$ is zero whenever constraints are inactive. Therefore the cost $\hat{J}$ takes the form:

$$\hat{J} = \mathbf{c}_k^\top H_c\, \mathbf{c}_k + x_k^\top P_x\, x_k, \quad \text{for } H_c > 0$$

and it can furthermore be shown that H_c is block-diagonal. Consequently, minimizing $\hat{J}_k$ is equivalent to minimizing $\mathbf{c}_k^\top H_c\, \mathbf{c}_k$ in (4.10), and the recursive feasibility of the constraints in (4.10) combined with the block diagonal structure of H_c together imply that

$$\mathbf{c}_{k+1}^\top H_c\, \mathbf{c}_{k+1} \le \mathbf{c}_k^\top H_c\, \mathbf{c}_k - c_{0|k}^\top P_c\, c_{0|k}$$

where P_c is a positive-definite matrix. Therefore, even though the nominal cost $\hat{J}_k$ may not decrease monotonically, the convergence analysis of Section 3.1 implies

$$\sum_{k=0}^{\infty} c_{0|k}^\top P_c\, c_{0|k} \le \infty \quad \text{and} \quad c_{0|k} \to 0 \text{ as } k \to \infty.$$

Finally, the robust stability of $x = 0$ follows from the fact that if $[K\ K_I]$ is LQ-optimal, then the gain γ of the closed loop system from the $c_{0|k}$ sequence to the state sequence z_k is necessarily finite, i.e.

$$\sum_{k=0}^{\infty} \|z_k^\top\|^2 \le \gamma \sum_{k=0}^{\infty} c_{0|k}^\top P_c\, c_{0|k} \le \infty.$$

(Note that this is a sensible notion of stability for the case of persistent time-varying additive uncertainty, for which it is not possible to enforce asymptotic convergence of the state z_k to zero).

The stability guarantee for the upper-bound cost $J_{\max}$ follows from the fact that the value of $J_{\max}$ that is obtained by minimizing the worst case cost in (4.10) at each sampling instant is necessarily monotonically non-increasing. Stability can therefore be demonstrated by using Lyapunov-like analysis applied to this optimal cost. Note however that the MPC optimization (4.10) becomes a min-max optimization in this case, which can require much greater computational effort than a QP problem.

Summary of Section 4

- Integral action can be incorporated in a predictive control law by including the integrated output error (evaluated for $w = 0$) in the predicted performance cost. Then:
  - the closed-loop system is asymptotically stable if $w = 0$
  - there is no steady state error if the response is stable and $w \neq 0$.
- Robust constraint satisfaction (i.e. for all possible disturbance values) can be ensured by imposing a finite number of linear constraints on predictions. The MPC optimization is then guaranteed to be feasible at all times $k > 0$ if a solution exists at $k = 0$.

李亞普諾夫方程

维基百科，自由的百科全书

李亞普諾夫方程（Lyapunov equation）是控制理論中的名詞，**離散李亞普諾夫方程**的型式如下：

$$AXA^H - X + Q = 0$$

其中Q為埃尔米特矩阵，A^H為A的共轭转置。**而連續李亞普諾夫方程**則是

$$AX + XA^H + Q = 0$$

李亞普諾夫方程應用在控制理論中的許多分支中，例如稳定性分析及最优控制。李亞普諾夫方程是得名自俄羅斯數學家亞歷山大·李亞普諾夫。

目录

在穩定性中的應用

求解的計算層面

解析解

離散時間

連續時間

相關條目

參考資料

外部連結

在穩定性中的應用

在以下的定理中，$A, P, Q \in \mathbb{R}^{n \times n}$，且$P$和$Q$是對應矩陣。而$P > 0$的意思是指$P$矩陣為正定矩陣。

定理（連續時間版本）：給定任意$Q > 0$，存在唯一$P > 0$滿足$A^T P + PA + Q = 0$的充份必要條件是線性系統$\dot{x} = Ax$是全域漸近穩定。二次函數$V(z) = z^T Pz$是李亞普諾夫函數，可以驗證系統的穩定性。

定理（離散時間版本）：給定任意$Q > 0$，存在唯一$P > 0$滿足$A^T PA - P + Q = 0$的充份必要條件是線性系統$x(k+1) = Ax(k)$是全域漸近穩定。$z^T Pz$為其李亞普諾夫函數。

求解的計算層面

有特殊的軟體可以求解李亞普諾夫方程。若是離散型式，常會用Kitagawa的Schur法[1]，若是連續型式，則會用Bartels和Stewart的計算法[2]。

解析解

定義$\mathbf{vec}(A)$（向量化）運算子是將矩陣A的所有列堆起来所形成的列向量，而$A \otimes B$是A和B的克罗内克积。兩種李亞普諾夫方程都可以用矩陣方程的解來表示。而且，若矩陣A穩定，解也可以用積分（連續時間）或是無限項和（離散時間）來表示。

離散時間

利用$\mathbf{vec}(ABC) = (C^T \otimes A)\,\mathbf{vec}(B)$的結果，可以得到

$$(I_{n^2} - \bar{A} \otimes A)\,\mathbf{vec}(X) = \mathbf{vec}(Q)$$

其中I_{n^2}為可相乘的單位矩陣[3]。可以積分或或是求解線性方程，即可以得到$\mathbf{vec}(X)$。再將各列重新整理，即可得到X。

而且，若A穩定，解X也可以表示為

$$X = \sum_{k=0}^{\infty} A^k Q (A^H)^k 。$$

連續時間

再利用克罗内克积和$\mathbf{vec}(A)$運算子，可以得到矩陣方程

$$(I_n \otimes A + \bar{A} \otimes I_n)\,\mathbf{vec}\,X = -\,\mathbf{vec}\,Q,$$

其中$\bar{A}$是將A各元素取共軛得到的矩陣。

類似離散時間的情形，若A穩定，解X也可以表示為

$$X = \int_0^{\infty} e^{A\tau} Q e^{A^H \tau} d\tau.$$

相關條目

- 西爾維斯特方程
- 代數Riccati方程
- 卡尔曼滤波

參考資料

1. Kitagawa, G. An Algorithm for Solving the Matrix Equation X = F X F' + S. International Journal of Control. 1977, **25** (5): 745–753. doi:10.1080/00207177708922266.
2. Bartels, R. H.; Stewart, G. W. Algorithm 432: Solution of the matrix equation AX + XB = C. Comm. ACM. 1972, **15** (9): 820–826. doi:10.1145/361573.361582.