ver 1.0, 200429 Elling W. Jacobsen, KTH

12 EL2520 Lecture notes 12: Model Predictive Control

Model Predictive Control (MPC) has gained an increasing popularity in almost all branches of industry in the last decades and is becoming something of a standard controller on par with PID, albeit significantly more advanced. The basic idea in MPC is to use the available dynamic model to predict the future, and use the control input to optimize the predicted future. The objective is similar to what is used in LQG, namely minimizing a quadratic objective function in the future process outputs and control inputs. However, while LQG was introduced in continuous time and with an infinite time horizon, MPC is based on discrete time computations and considers a finite time horizon. Although the considered horizon is finite, the optimization is repeated at every sample instance, and with a constant horizon this implies that the horizon is constantly moving forward in time. For this reason, MPC is also often called *receding horizon control.* One of the main advantages of MPC is that constraints on the control inputs and outputs can be included in the optimization.

Below we give a brief introduction to MPC. We start by introducing sampling and discrete time models, and then formulate and solve the discrete time finite horizon LQR[1] problem. A key is that the LQR solution is implemented in a receding fashion and we show that we then can include constraints in the problem.

12.1 Sampling and Discrete Time Models

When a controller is implemented on a digital computer, computations can only be done at discrete time instances. See Figure 12.1. The control input is computed at discrete time points and typically kept constant between these time instants (zero-order-hold). The continuous output from the process G is sampled every h seconds, the sampling time, and fed into the computer. The time of sampling the output and computing the control input are usually assumed to coincide.

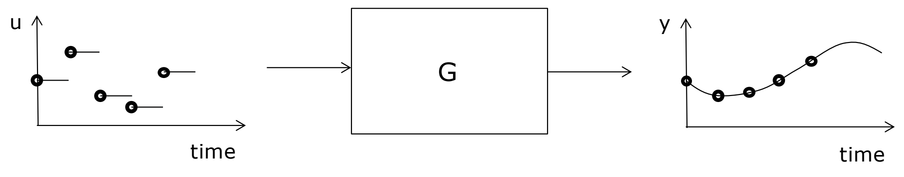

Figure 12.1: Zero-order hold input and output sampling.

To describe the process dynamics at the sampling instances, consider the system evolution

[1]Linear Quadratic Regulator, corresponding to the determinstic part of the LQG problem.

between two sample instances. The solution of the linear differential equations

$$\dot{x} = Ax(t) + Bu(t)$$

is

$$x(t+h) = e^{Ah}x(t) + \int_{s=0}^{h} e^{As}Bu(t+s)ds \tag{1}$$

If u is held constant between samples so that $u(t) = u_t, t \in [t, t+h]$, then

$$\begin{aligned} x(t+h) &= A_D x(t) + B_D u_t \ ; \quad A_D = e^{Ah} \ ; \quad B_D = \int_{s=0}^{h} e^{As}Bds & (2) \\ y(t) &= Cx(t) + Du_t & (3) \end{aligned}$$

This a discrete-time linear dynamic system. For notational convenience we drop the reference to physical time and write

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k & (4) \\ y_k &= Cx_k + Du_k & (5) \end{aligned}$$

where $u_0, u_1, \ldots$ is the input sequence, $y_0, y_1, \ldots$ is the output sequence and $x_0, x_1, \ldots$ is the state evolution. Note that the discrete time system is stable if all eigenvalues of A are inside the unit circle in the complex plane.

12.2 Finite Horizon LQR

The Linear Quadratic Regulator (LQR) over a finite horizon N determines the input sequence $U = u_0, \ldots, u_{N-1}$ that minimizes a square function of the system states and control inputs over the considered horizon

$$\min_U J(U) = \min_U \sum_{k=0}^{N-1} (x_k^T Q_1 x_k + u_k^T Q_2 u_k) + x_N^T Q_f x_N \tag{6}$$

Here Q_1, Q_2, Q_f are positive definite weight matrices. Note that the final state cost, with weight Q_f, mainly is used to ensure stability.

To solve (6), introduce the notation $X = (x_0, x_1, \ldots, x_N), U = (u_0, u_1, \ldots, u_{N-1})$. Then, from model (5) we get

$$\begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix} = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ B & 0 & \cdots & 0 \\ AB & B & 0 & \cdots \\ \vdots & \vdots & & \\ A^{N-1}B & A^{N-2}B & \cdots & B \end{bmatrix} \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{N-1} \end{bmatrix} + \begin{bmatrix} I \\ A \\ A^2 \\ \vdots \\ A^N \end{bmatrix} x_0$$

which can be expressed as

$$X = GU + Hx_0$$

where $G \in \mathbb{R}^{Nn \times Nm}, \ H \in \mathbb{R}^{Nn \times n}$.

The objective function $J(U)$ can be written on matrix form

$$J(U) = X^T \mathrm{d}iag(Q_1 \quad Q_1 \quad \cdots \quad Q_f)X + U^T \mathrm{d}iag(Q_2 \quad Q_2 \quad \cdots \quad Q_2)U = X^T\bar{Q}_1X + U^T\bar{Q}_2U$$

Now, inserting the model $X = GU + Hx_0$ we get

$$J(U) = (U^TG^T + x_0^TH^T)\bar{Q}_1(GU + Hx_0) + U^T\bar{Q}_2U = U^T\underbrace{(G^T\bar{Q}_1G + \bar{Q}_2)}_{P_{LQ}}U + 2\underbrace{x_0^TH^T\bar{Q}_1G}_{q_{LQ}}U + \underbrace{x_0^TH^T\bar{Q}_1Hx_0}_{r_{LQ}}$$

Thus, we can write the objective function as

$$J(U) = U^TP_{LQ}U + 2q_{LQ}U + r_{LQ}$$

Taking the derivative with respect to the input sequence U we get

$$\frac{dJ(U)}{dU} = 2P_{LQ}U + 2q_{LQ}$$

and setting this to zero[2] we get the optimal control

$$U^* = -P_{LQ}^{-1}q_{LQ}$$

12.3 Constrained Receding Horizon Control

The solution to the discrete time LQR problem above is an open-loop control with no feedback. In order to incorporate feedback, to deal with model uncertainty and disturbances, one can implement it in a receding horizon fashion in which the optimization is repeated at every sample and the system state is updated at every sample using an observer, e.g., a Kalman filter. By keeping a fixed length of the horizon in every sample, the horizon will then move forward in time for each sample, hence the name receding horizon control. The use of an observer to update the state estimate at every sample ensures that there is feedback from the output measurements. The algorithm is then, at each sample,

1. Solve LQR over horizon N
2. Implement u_0
3. Let system evolve one sample and update state estimate using an observer, and repeat from 1.

Model Predictive Control (MPC) is based on this receding horizon control principle, but with the addition of constraints in the LQR problem. The constrained LQR problem is

$$\begin{array}{ll} \text{minimize} & J(U) = \sum_{k=0}^{N-1}(x_k^TQ_1x_k + u_k^TQ_2u_k) + x_N^TQ_fx_N \\ \text{subject to} & u_{\min} \leq u_k \quad \leq u_{\max}, \quad k = 0, \ldots, N-1 \\ & y_{\min} \leq Cx_k \leq y_{\max}, \quad k = 1, \ldots, N \\ & x_{k+1} = Ax_k + Bu_k \end{array} \tag{7}$$

[2]The second order derivative is positive

The addition of constraints is important since the control inputs are essentially always constrained in real applications and MPC can deal with this in an optimal fashion. The ability to set constraints on the outputs is also important in many applications, e.g., set a maximum temperature in a chemical reactor or set the minimum speed of an airplane.

The constrained LQR problem above can be written on the form of a Quadratic Programming (QP) problem

$$\begin{array}{ll} \text{minimize} & u^T P u + 2q^T u + r \\ \text{subject to} & Au \leq b \end{array} \tag{8}$$

To write the original problem (7) on the QP form (8), note first that we as above can write the objective function on the form

$$J(U) = U^T P_{LQ} U + 2q_{LQ} U + r_{LQ}$$

The objective function $J(U)$ is convex provided $Q_2 > 0$ (positive definite). To write the constraints on the form in (8), consider first the constraint $u_{min} \leq u_k \leq u_{max}$ which can be written

$$\begin{bmatrix} 1 \\ -1 \end{bmatrix} u_k \leq \begin{bmatrix} u_{max} \\ -u_{min} \end{bmatrix}$$

For the constraints on the output y we introduce $Y = \begin{pmatrix} y_0 & y_1 & \cdots & y_N \end{pmatrix}^T$ and the constraint

$$y_{\min}\ \mathbf{1} \leq Y \leq y_{\max}\ \mathbf{1}$$

where $\mathbf{1}$ is a vector with all elements 1. Let

$$\bar{C} = \begin{bmatrix} C & 0 & \dots & 0 \\ 0 & C & 0 & 0 \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & C \end{bmatrix}$$

Then $Y = \bar{C}X$ and

$$Y \geq y_{min}\ \mathbf{1} \ \Leftrightarrow\ \bar{C}(GU + Hx_0) \geq y_{min}\ \mathbf{1} \ \Leftrightarrow\ \underbrace{\bar{C}G}_{A_{\underline{y}}} U \geq \underbrace{y_{min}\ \mathbf{1} - \bar{C}Hx_0}_{b_{\underline{y}}}$$

Similarly,

$$\underbrace{\bar{C}G}_{A_{\bar{y}}} U \leq \underbrace{y_{max}\ \mathbf{1} - \bar{C}Hx_0}_{b_{\bar{y}}}$$

The overall QP problem is then

$$\min_U U^T P_{LQ} U + 2q_{LQ} U + r_{LQ}$$

subject to

$$\begin{bmatrix} A_{\bar{y}} \\ -A_{\underline{y}} \\ I \\ -I \end{bmatrix} U \leq \begin{bmatrix} b_{\bar{y}} \\ -b_{\underline{y}} \\ u_{max}\mathbf{1} \\ -u_{min}\mathbf{1} \end{bmatrix}$$

In summary, at each sample the MPC controller receives a state estimate for x_0 from an observer, e.g., a Kalman filter, and then solves the above QP problem. The first input

u_0 in the determined input sequence is then implemented on the system and the system evolves on sample time before obtaining a new state estimate, solving the QP problem and so on.

For an example, MPC control of the DC servo, see the lecture slides for Lecture 12.

Some remarks on the MPC

- The main tuning parameters are the weights Q_1, Q_2, Q_f and the length of the prediction horizon N. Often different horizons are used for the prediction and the control input.
- In some cases, the QP problem may become infeasible, that is, there exists no solution that meets all constraints. To avoid this it is common to soften the constraints on the outputs by introducing so called *slack* variables. In essence, one then allows these constraints to be violated but penalize the violation in the objective function. Infeasibility may still happen, and one therefore often have a backup solution, e.g., in terms of a standard LQ controller.

The main purpose of this lecture has been to introduce the concept of MPC. To learn more in-depth about the theoretical basis of MPC and its application we refer to the course EL2700 Model Predictive Control given in period 1.