Model Predictive Control — Lecture 2
Basic Formulation of MPC

Ling KV (ekvling@ntu.edu.sg)

January 2025

School of Electrical and Electronic Engineering

MPC Example, [JMM2001, Example 1.3]

Plant: $G(z) = \frac{2}{z-0.7}$

What is the optimal input $\hat{u}(k|k)$ to drive the future plant output $y(k+2)$ to set-point $w = 3$, assuming that $\hat{u}(k+1|k) = \hat{u}(k|k)$?

Solution:
From the model, we have the difference equation:

$$y(k) = 0.7y(k-1) + 2u(k-1)$$

which we can use to predict the future outputs as follows:

$$\begin{aligned}
\hat{y}(k+2|k) &= 0.7\hat{y}(k+1|k) + 2\hat{u}(k+1|k) \\
&= 0.7(0.7y(k|k) + 2\hat{u}(k|k)) + 2\hat{u}(k+1|k) \\
&= 0.49y(k|k) + 1.4\hat{u}(k|k) + 2\hat{u}(k+1|k) \\
&= w
\end{aligned}$$

Hence

$$\hat{u}(k|k) = \frac{1}{1.4+2}(w - 0.49y(k|k))$$

where we have assumed that $\hat{u}(k+1|k) = \hat{u}(k|k)$.

MPC Example, continue...

What is the optimal input if we required that

$$\hat{y}(k+1|k) = \hat{y}(k+2|k) = w = 3?$$

Solution:
Recall that

$$\begin{bmatrix} \hat{y}(k+1|k) \\ \hat{y}(k+2|k) \end{bmatrix} = \begin{bmatrix} 0.7 \\ 0.49 \end{bmatrix} y(k|k) + \begin{bmatrix} 2 \\ 3.4 \end{bmatrix} \hat{u}(k|k)$$

In this case, there are more equations to be satisfied than there are variables. We therefore have to be satisfied with an approximate solution. Most commonly, this approximate solution is the least squares solution. In Matlab, the LS solution is obtained using the 'backslash' operator, i.e. if Ax=b, x=A\b. Thus,

$$\hat{u}(k|k) = \begin{bmatrix} 2 \\ 3.4 \end{bmatrix} \backslash \left(\begin{bmatrix} w \\ w \end{bmatrix} - \begin{bmatrix} 0.7 \\ 0.49 \end{bmatrix} y(k|k) \right)$$

MPC Example, continue...

We can already see at this point how MPC can easily be given the capability of respecting constraints. If there are constraints on the inputs and/or outputs, then the simple ‘linear least squares’ solution has to be replaced by a ‘constrained least squares’ solution. With constraints, it is no longer possible to obtain a closed-form solution and some form of iterative optimisation algorithm must be employed. If the constraints are in the form of linear inequalities, then we have a ‘quadratic program (QP)’ problem.

MPC Example, continue...

Now suppose that we allow a more complicated future input trajectory. We could allow the input to change over the next N_u steps (N_u is known as the control horizon)
What is the optimal control if $N_u = 2$?

Solution:
In this case, we have

$$\begin{bmatrix} \hat{y}(k+1|k) \\ \hat{y}(k+2|k) \end{bmatrix} = \begin{bmatrix} 0.7 \\ 0.49 \end{bmatrix} y(k|k) + \begin{bmatrix} 2 & 0 \\ 1.4 & 2 \end{bmatrix} \begin{bmatrix} \hat{u}(k|k) \\ \hat{u}(k+1|k) \end{bmatrix}$$

and obtained

$$\begin{bmatrix} \hat{u}(k|k) \\ \hat{u}(k+1|k) \end{bmatrix} = \begin{bmatrix} 2 & 0 \\ 1.4 & 2 \end{bmatrix} \backslash \left(\begin{bmatrix} w \\ w \end{bmatrix} - \begin{bmatrix} 0.7 \\ 0.49 \end{bmatrix} y(k|k) \right)$$

A Simple MATLAB Program

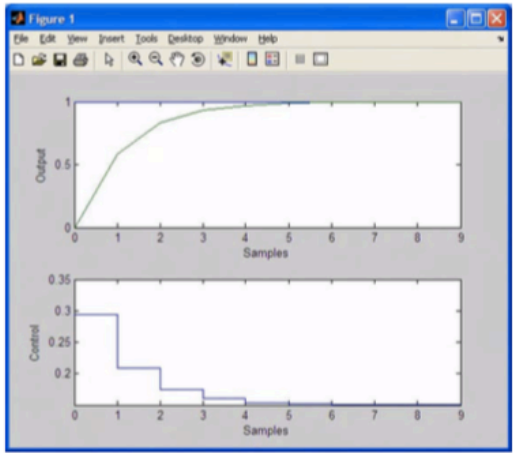
```
% A simple matlab program to closed-loop control

clear all
% define the plant and set-point
az = [1 -0.7]; bz = [2];
w = 1; %set-point

% storage for simulation
yplant=0; uplant=0; ydata=[]; udata=[]; wdata=[]; tdata=[];

for k=1:10
    %measure the current plant output
    yk = yplant;
    % calculate the MPC control
    uk = (w - 0.49*yk)/(1.4+2);
    % store the data for plotting later
    tdata = [tdata; k-1]; wdata = [wdata; w];
    ydata = [ydata; yk];  udata = [udata; uk];
    % apply the control to the plant
    uplant = uk;
    % simulate the plant
    yp_new = -az(2)*yplant + bz*uplant;
    yplant = yp_new;
end

subplot(211),plot(tdata,wdata,tdata,ydata),ylabel('Output'),xlabel('Samples')
subplot(212),stairs(tdata,udata),ylabel('Control'),xlabel('Samples')
```



A Basic Formulation of MPC

From the simple `MATLAB` program, we see that the sequence of action at time step `k` is the following:

1. Obtain measurement $y(k)$
2. Compute the required plant input $u(k)$
3. Apply $u(k)$ to the plant

In determining the MPC control, one needs

- a process model to predict the future plant outputs
- an optimization criteria (cost function)

Prediction Using a State Space Model, 1/4

Consider the discrete-time state space model

$$x(k+1) = A_p x(k) + B_p u(k), \qquad y(k) = C_p x(k) \tag{1}$$

where x is a n-dimensional state vector, u is a l-dimensional input vector and y is a m-dimensional measured output vector.
We can predicted the process output by iterating the model,

$$\hat{y}(k+1|k) = C_p \hat{x}(k+1|k) = C_p A_p x(k) + C_p B_p \hat{u}(k|k)$$

Similarly,

$$\begin{aligned} \hat{y}(k+2|k) &= C_p \hat{x}(k+2|k) \\ &= C_p A_p^2 x(k) + C_p A_p B_p \hat{u}(k|k) + C_p B_p \hat{u}(k+1|k) \end{aligned}$$

Prediction Using a State Space Model, 2/4

and in general,

$$\begin{aligned} \hat{y}(k+i|k) &= C_p \hat{x}(k+i|k) \\ &= C_p A_p^i x(k) + \sum_{j=1}^{i} C_p A_p^{j-1} B_p \hat{u}(k+i-j|k) \end{aligned} \tag{2}$$

State space model has the advantage that it can be used for multi-variable processes in a straightforward manner.

Prediction Using a State Space Model, 3/4

If we collect the predicted outputs into a vector, we obtained an expression for the vector of predicted outputs

$$\begin{bmatrix} \hat{y}(k+1) \\ \hat{y}(k+2) \\ \vdots \\ \hat{y}(k+N) \end{bmatrix} = \begin{bmatrix} C_pA_p \\ C_pA_p^2 \\ \vdots \\ C_pA_p^N \end{bmatrix} x(k) + \begin{bmatrix} C_pB_p & 0 & \dots & 0 \\ C_pA_pB_p & C_pB_p & & \vdots \\ \vdots & & \ddots & \vdots \\ C_pA_p^{N-1}B_p & C_pA_p^{N-2}B_p & \dots & C_pB_p \end{bmatrix} \begin{bmatrix} \hat{u}(k) \\ \hat{u}(k+1) \\ \vdots \\ \hat{u}(k+N-1) \end{bmatrix}$$

Now, note that

$$\begin{bmatrix} \hat{u}(k) \\ \hat{u}(k+1) \\ \vdots \\ \hat{u}(k+N-1) \end{bmatrix} = \begin{bmatrix} u(k-1) \\ u(k-1) \\ \vdots \\ u(k-1) \end{bmatrix} + \begin{bmatrix} I & 0 & \dots & 0 \\ I & I & & \vdots \\ \vdots & & \ddots & \vdots \\ I & I & \dots & I \end{bmatrix} \begin{bmatrix} \Delta\hat{u}(k) \\ \Delta\hat{u}(k+1) \\ \vdots \\ \Delta\hat{u}(k+N-1) \end{bmatrix}$$

Prediction Using a State Space Model, 4/4

Combining, we have

$$\begin{bmatrix} \hat{y}(k+1) \\ \hat{y}(k+2) \\ \vdots \\ \hat{y}(k+N) \end{bmatrix} = \underbrace{\begin{bmatrix} C_pA_p \\ C_pA_p^2 \\ \vdots \\ C_pA_p^N \end{bmatrix} x(k) + \begin{bmatrix} C_pB_p \\ C_pA_pB_p + C_pB_p \\ \vdots \\ \sum_{i=0}^{N-1} C_pA_p^iB_p \end{bmatrix} u(k-1)}_{\text{past}}$$

$$+ \underbrace{\begin{bmatrix} C_pB_p & \cdots & 0 \\ C_pA_pB_p + C_pB_p & \cdots & 0 \\ \vdots & \ddots & \vdots \\ \sum_{i=0}^{N-1} C_pA_p^iB_p & \cdots & C_pB_p \end{bmatrix} \begin{bmatrix} \Delta\hat{u}(k) \\ \Delta\hat{u}(k+1) \\ \vdots \\ \Delta\hat{u}(k+N-1) \end{bmatrix}}_{\text{future}}$$

or, in vector-matrix notation

$$\hat{\mathbf{Y}} = \underbrace{\Phi x(k) + \Upsilon u(k-1)}_{\text{past}} + \underbrace{G\hat{\mathbf{U}}}_{\text{future}} \tag{3}$$

Cost Function

The general aim is that the future output ($\hat{y}$) within the considered horizons should follow a pre-determined reference signal ($\hat{w}$) and, at the same time, the control effort ($\Delta\hat{u}$) necessary for doing so should be penalized. Such objective can be expressed mathematically as

$$J = \sum_{i=N_1}^{N_2}[\hat{y}(k+i|k) - w(k+i|k)]^2 + \lambda\sum_{i=1}^{N_u}[\Delta\hat{u}(k+i-1|k)]^2 \quad (4)$$

The parameters N_1 and N_2 are the minimum and maximum prediction horizons, and N_u is the control horizon. λ is the control weighting which expresses the relative importance between the tracking errors ($\hat{y} - \hat{w}$) and control effort ($\Delta\hat{u}$).
The cost function can be written in vector form as

$$J = (\hat{\mathbf{Y}} - \hat{\mathbf{W}})^T(\hat{\mathbf{Y}} - \hat{\mathbf{W}}) + \lambda\hat{\mathbf{U}}^T\hat{\mathbf{U}}$$

Mini-Tutorial 1, Quadratic Forms

Expressions like $x^T Qx$ and $u^T Ru$, where x, u are vectors and Q, R are symmetric matrices, are called quadratic forms and are often written as $\|x\|_Q^2$ and $\|u\|_R^2$. They are just compact representations of certain quadratic functions in several variables.
For example,

$$10x_1^2 + 8x_1x_2 + 3x_2^2 = \begin{bmatrix} x_1 & x_2 \end{bmatrix} \begin{bmatrix} 10 & 4 \\ 4 & 3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = x^T Qx$$

If $V = x^T Qx$, then the gradient of V is given by

$$\nabla V = \begin{bmatrix} \frac{\partial V}{\partial x_1}, & \frac{\partial V}{\partial x_2}, & \ldots, & \frac{\partial V}{\partial x_n} \end{bmatrix}^T = 2Qx$$

Mini-Tutorial 2, The Least Squares Problem

$$\min_{\theta} J = \|W - H\theta\|_Q^2 \quad \text{gives} \quad \theta_{\text{opt}}^* = (H^T Q H)^{-1} H^T Q W$$

Proof:

$$\begin{aligned} J &= \|W - H\theta\|_Q^2 = (W - H\theta)^T Q (W - H\theta) \\ &= W^T Q W + \theta^T (H^T Q H)\theta - 2\theta^T H^T Q W \end{aligned}$$

The gradient of J is

$$\nabla J = 2(H^T Q H)\theta - 2H^T Q W$$

Hence, setting $\nabla J = 0$ gives

$$\theta_{\text{opt}}^* = (H^T Q H)^{-1} H^T Q W$$

Obtaining the MPC Control Law, 1/2

The control law is derived by finding the control effort ($\Delta\hat{u}$) which minimises the objective function. To do this, the values of the predicted outputs $\hat{y}(k+i|k)$ are expressed as a function of past values of inputs and outputs and future control signals by making use of the process model.
Recall that the vector of predicted outputs is
$\hat{\mathbf{Y}} = \Phi x(k) + \Upsilon u(k-1) + G\hat{\mathbf{U}}$ and for simplicity, let's denote it as

$$\hat{\mathbf{Y}} = \hat{\mathbf{F}} + G\hat{\mathbf{U}}, \text{ where } \hat{\mathbf{F}} = \Phi x(k) + \Upsilon u(k-1) \text{ is independent of } \hat{\mathbf{U}}$$

Thus

$$\begin{aligned} J &= (\hat{\mathbf{Y}} - \hat{\mathbf{W}})^T(\hat{\mathbf{Y}} - \hat{\mathbf{W}}) + \lambda\hat{\mathbf{U}}^T\hat{\mathbf{U}} \\ &= (\hat{\mathbf{F}} + G\hat{\mathbf{U}} - \hat{\mathbf{W}})^T(\hat{\mathbf{F}} + G\hat{\mathbf{U}} - \hat{\mathbf{W}}) + \lambda\hat{\mathbf{U}}^T\hat{\mathbf{U}} \\ &= \hat{\mathbf{U}}^T(G^TG + \lambda I)\hat{\mathbf{U}} + 2\hat{\mathbf{U}}^TG^T(\hat{\mathbf{F}} - \hat{\mathbf{W}}) + (\hat{\mathbf{F}} - \hat{\mathbf{W}})^T(\hat{\mathbf{F}} - \hat{\mathbf{W}}) \end{aligned}$$

Obtaining the MPC Control Law, 2/2

The gradient of J wrt to $\hat{\mathbf{U}}$ is

$$\nabla J = 2(G^T G + \lambda I)\hat{\mathbf{U}} + 2G^T(\hat{\mathbf{F}} - \hat{\mathbf{W}})$$

and the optimal is obtained when the gradient is zero, giving

$$\hat{\mathbf{U}} = (G^T G + \lambda I)^{-1} G^T(\hat{\mathbf{W}} - \hat{\mathbf{F}}) = (G^T G + \lambda I)^{-1} G^T(\hat{\mathbf{W}} - \Phi x(k) - \Upsilon u(k-1))$$

If one does not have knowledge of future set-points, then a commonly made assumption/simplication is to assume a constant set-point from time k and onwards. Thus,

$$\hat{\mathbf{W}} = \begin{bmatrix} w(k) & w(k) & \dots & w(k) \end{bmatrix}^T$$

If the control horizon, $N_u < N$, this implies that

$$\Delta\hat{u}(k+i|k) = 0, i \geq N_u$$

and the appropriate columns of the G matrix can be deleted to simplify computations. In particular, for single-input model, $G^T G$ is a scalar when $N_u = 1$.

Receding Horizon Strategy

MPC adopts a receding horizon control strategy, i.e., only the first element of the control moves is applied to the plant, and then the optimisation process repeats at the next sampling instant with a new measurement. So, only $\Delta\hat{u}(k|k)$ is needed, leading to

$$\begin{aligned}\Delta\hat{u}(k|k) &= \begin{bmatrix} I & 0 & \dots & 0 \end{bmatrix} \hat{\mathbf{U}} \\ &= \begin{bmatrix} I & 0 & \dots & 0 \end{bmatrix} (G^T G + \lambda I)^{-1} G^T (\hat{\mathbf{W}} - \Phi x(k) - \Upsilon u(k-1))\end{aligned}$$

The above equation can be written as[1]

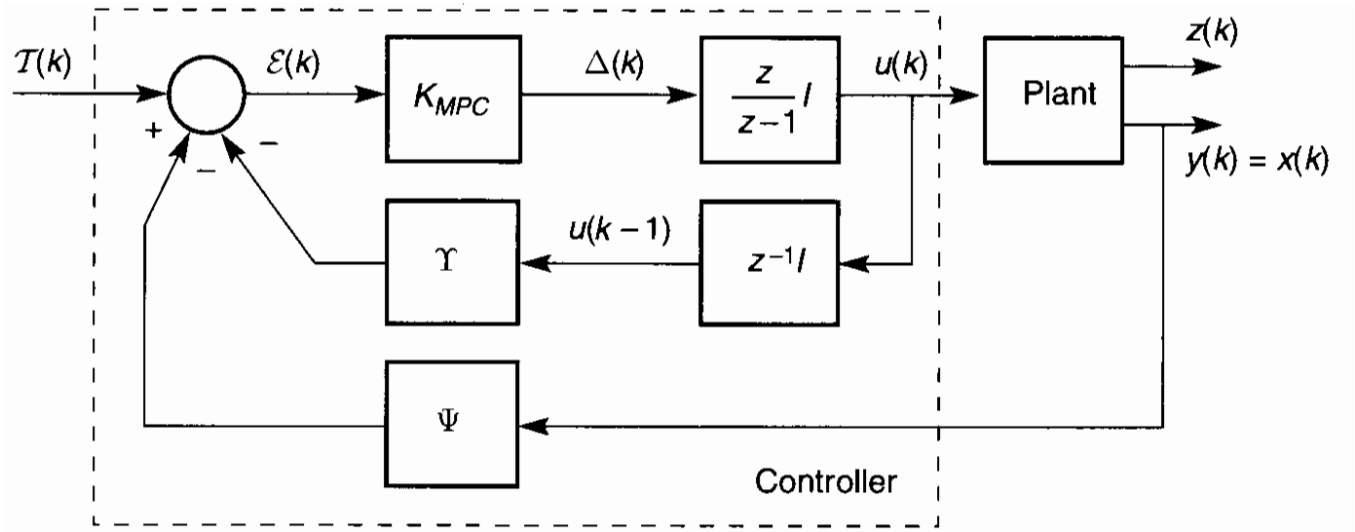
$$\Delta\hat{u}(k|k) = K_1 w(k) - K_2 x(k) - K_3 u(k-1) \tag{5}$$

Since the gains K_1, K_2 and K_3 are constant and do not depend on k, this shows that MPC, when constraints are not considered, is a time-invariant linear feedback controller.

[1] assuming constant future set-point

Structure of Unconstrained MPC

From Eq. 5, the predictive controller, for the unconstrained problem and with full state measurement, can be drawn as shown below:

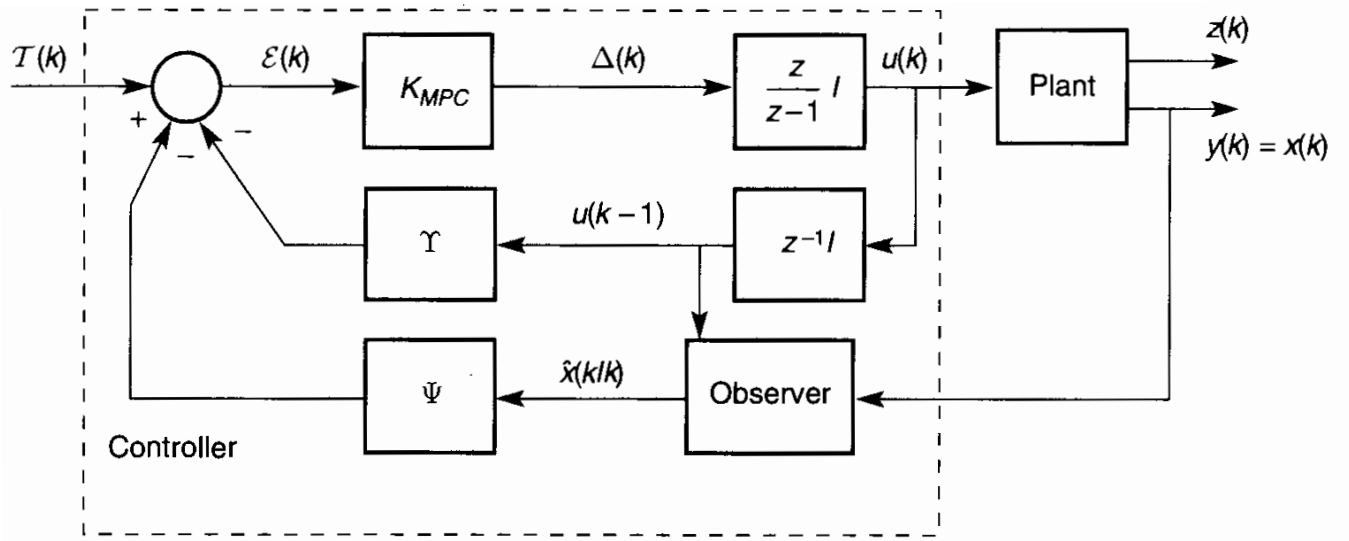


where

$$K_{\text{MPC}} = \begin{bmatrix} I & 0 & \dots & 0 \end{bmatrix} (G^T G + \lambda I)^{-1} G^T$$

Structure of Unconstrained MPC, with state observer

When we do not have measurement of the whole state vector, an observer[2] can be used to estimate the state vector from the input and output signals. With the state observer, we arrived at the controller structure shown below:



[2] See Maciejowski (2001), p.58 for a mini-tutorial on state observers.

Many State Space Representations, 1/3

Instead of the state space model (1), we could also derive the MPC control law by using a state space model with an incremental input:

$$\begin{bmatrix} x(k+1) \\ u(k) \end{bmatrix} = \begin{bmatrix} A_p & B_p \\ 0 & I \end{bmatrix} \begin{bmatrix} x(k) \\ u(k-1) \end{bmatrix} + \begin{bmatrix} B_p \\ I \end{bmatrix} \Delta u(k)$$
$$y(k) = \begin{bmatrix} C_p & 0 \end{bmatrix} \begin{bmatrix} x(k) \\ u(k-1) \end{bmatrix} \tag{6}$$

Here is another way:

$$\begin{bmatrix} \Delta x(k+1) \\ y(k) \end{bmatrix} = \begin{bmatrix} A_p & 0 \\ C_p & I \end{bmatrix} \begin{bmatrix} \Delta x(k) \\ y(k-1) \end{bmatrix} + \begin{bmatrix} B_p \\ 0 \end{bmatrix} \Delta u(k)$$
$$y(k) = \begin{bmatrix} C_p & I \end{bmatrix} \begin{bmatrix} \Delta x(k) \\ y(k-1) \end{bmatrix} \tag{7}$$

Many State Space Representations, 2/3

Yet another possibility is:

$$\begin{bmatrix} \Delta x(k+1) \\ y(k+1) \end{bmatrix} = \begin{bmatrix} A_p & 0 \\ C_p A_p & I \end{bmatrix} \begin{bmatrix} \Delta x(k) \\ y(k) \end{bmatrix} + \begin{bmatrix} B_p \\ C_p B_p \end{bmatrix} \Delta u(k)$$

$$y(k) = \begin{bmatrix} 0 & I \end{bmatrix} \begin{bmatrix} \Delta x(k) \\ y(k) \end{bmatrix} \tag{8}$$

These state space models have the form

$$\xi(k+1) = A\xi(k) + B\Delta u(k), \qquad y(k) = C\xi(k) \tag{9}$$

where ξ is the augmented state vector and the matrices A, B and C are defined in the context, for example,

$$A \leftarrow \begin{bmatrix} A_p & 0 \\ C_p A_p & I \end{bmatrix}, \quad B \leftarrow \begin{bmatrix} B_p \\ C_p B_p \end{bmatrix}, \quad C \leftarrow \begin{bmatrix} 0 & I \end{bmatrix},$$

if the augmented state space model (8) has been selected.

Many State Space Representations, 3/3

It can be shown that the predictions based on the augmented state space model (9) is

$$\begin{bmatrix} \hat{y}(k+1) \\ \hat{y}(k+2) \\ \vdots \\ \hat{y}(k+N) \end{bmatrix} = \underbrace{\begin{bmatrix} CA \\ CA^2 \\ \vdots \\ CA^N \end{bmatrix} \xi(k)}_{\text{past}} + \underbrace{\begin{bmatrix} CB & 0 & \dots & 0 \\ CAB & CB & 0 & \vdots \\ \vdots & & \ddots & \vdots \\ CA^{N-1}B & \dots & \dots & CB \end{bmatrix} \begin{bmatrix} \Delta\hat{u}(k|k) \\ \Delta\hat{u}(k+1|k) \\ \vdots \\ \Delta\hat{u}(k+N-1|k) \end{bmatrix}}_{\text{future}}$$

or, in vector-matrix notation

$$\hat{\mathbf{Y}} = \underbrace{\Phi\xi(k)}_{\text{past}} + \underbrace{G\hat{\mathbf{U}}}_{\text{future}} \tag{10}$$

and the unconstrained optimal control is

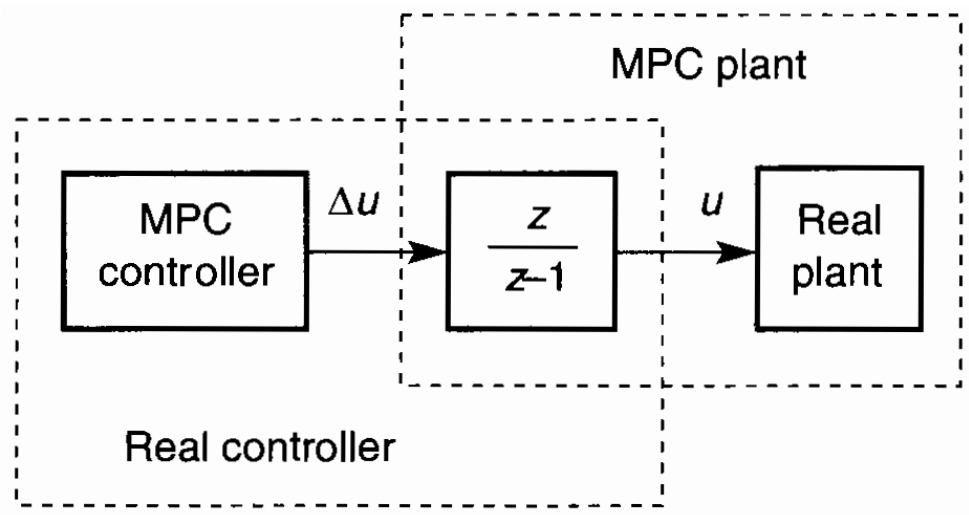
$$\hat{\mathbf{U}} = (G^T G + \lambda I)^{-1} G^T (\hat{\mathbf{W}} - \Phi\xi(k)) \tag{11}$$

and the receding-horizon control law is

$$\Delta\hat{u}(k|k) = K_1 w(k) + K_2 \xi(k) \tag{12}$$

Controller and Plant Boundary

The plant model is usually express in terms of input u, but the MPC cost function penalises changes in the input Δu. It is therefore convenient to regard the MPC controller as poducing the signal Δu and the plant as having Δu as its input.



Ex 3.3: Swimming Pool Example

The water temperature in a heated swimming pool, θ, is related to the heater input power, q, and the ambient air temperature, θ_a, according to the equation

$$T\frac{d\theta}{dt} = kq + \theta_a - \theta$$

where $T = 1$ hour and $k = 0.2^{\circ}\mathrm{C}/kW$. Predictive control is to be applied to keep the water at a desired temperature, and a sampling interval $T_s = 0.25$ hour is to be used. The control update interval is to be the same as T_s.

1. Use `MATLAB` to show that the corresponding discrete-time model is

$$\theta(k+1) = 0.7788\theta(k) + 0.0442q(k) + 0.2212\theta_a(k)$$

2. Verify that if the horizons $N_1 = 1$, $N_2 = 10$ and $N_u = 3$ are used, then

$$\Delta q(k) = 22.604w(k) - 17.604\Delta\theta(k) - 22.604\theta(k)$$

Exercise 1

1. Derive the prediction equation (Eq. 10) using the state space model given by Eq. 6. Show that it is identical to Eq. 3 obtained in slide 11.
2. Obtain a general formula for the the prediction equation (Eq. 10), incorporating the parameters N_1, N_2 and N_u.
3. If $A \in \mathcal{R}^{n\times n}$, $B \in \mathcal{R}^{n\times m}$, and $C \in \mathcal{R}^{p\times n}$, what are the dimensions of the matrices Φ and G in terms of N_1, N_2 and N_u?
4. **MPC with terminal state weighting**
   Derive the MPC control law which minimises the cost function:

$$J = \sum_{i=1}^{N}[(\hat{w}(k+i|k) - \hat{y}(k+i|k)]^2 + \lambda \sum_{i=1}^{N}[\Delta\hat{u}(k+i-1|k)]^2 + \hat{x}(k+N|k)^T Q_N \hat{x}(k+N|k)$$

Lab Exercises, 1/2

1. **Predictions based on state space models**
   Given state space model with incremental input (Eq. 9), write a `MATLAB` function to compute the output predictions (Eq. 10). Use the following function prototype: `[Phi,G] = predictions(A,B,C,N1,N2,Nu)`
2. **Unconstrained receding-horizon MPC Control Law**
   Write a `MATLAB` function to compute the MPC control law (Eq. 12). Use the following function prototype: `[K1,K2] = MPC1(Phi,G,lambda)`
3. **Simulation of a MPC control system, I**
   Use the simple `MATLAB` program (slide 6) as a guide, together with the two functions above, simulate a MPC control system. Study the effects of the MPC tuning parameters on the closed loop performance. Which state space model(s) gives zero steady state offset even when there is unknown constant load disturbance?

Lab Exercises, 2/2

4. **Simulation of a MPC control system, II**
   Alternatively, a (unconstrained) MPC control system can be simulated using `SIMULINK`. Build a `SIMULINK` model as shown below.
5. Do the same for the other two state space models: Eq. 7 and Eq. 8. Which state space model(s) gives zero steady state offset even when there is unknown constant load disturbance?