1 ARX to Matrix form

Starting from ARX

$$A(q)y(t) = B(q)u(t-n_k) + e(t)$$

q is delay operator. Specifically,

$$\begin{aligned} A(q) &= 1 + a_1 q^{-1} + ... + a_{n_a} q^{-n_a} \\ B(q) &= b_1 + b_2 q^{-1} + ... + b_{n_b} q^{-n_b+1} \end{aligned}$$

Let's assume $n_a = n_b = p$, p is ARX order. And $n_k = 0$

$$\begin{aligned} A(q) &= 1 + a_1 q^{-1} + ... + a_p q^{-p} \\ B(q) &= b_1 + b_2 q^{-1} + ... + b_p q^{-p+1} \end{aligned}$$

Convert ARX form to Matrix form. This transformation makes calculating $y(t)$ easily.

$$y(t) + a_1 y(t-1) + ... + a_p y(t-p) = b_1 u(t) + ... + b_p u(t-p+1)$$

$$y(t) = A y_p(t-p) + B u_p(t-p+1)$$

$$y_p(t-p) = \begin{bmatrix} y(t-p) \\ y(t-p+1) \\ \vdots \\ y(t-1) \end{bmatrix}_{p\times 1}, u_p(t-p+1) = \begin{bmatrix} u(t-p+1) \\ u(t-p+2) \\ \vdots \\ u(t) \end{bmatrix}_{p\times 1}$$

Divide u to u and x. u is control inputs, and x is system inputs.

$$y(t) = A y_p(t-p) + B_1 u_p(t-p+1) + B_2 x_p(t-p+1) \tag{1}$$

Purpose of MPC is predict model's optimum output in prediction horizon. When length of prediction horizon is h, calculating this value is needed to optimization process in MPC.

$$y_h(t+1) = \begin{bmatrix} y(t+1) \\ y(t+2) \\ \vdots \\ y(t+h) \end{bmatrix} \tag{2}$$

To get the elements of (2), equation (1) can be used.

$$\begin{aligned}
y(t) &= Ay_p(t-p) + B_1u_p(t-p+1) + B_2x_p(t-p+1) \\
y(t+1) &= Ay_p(t-p+1) + B_1u_p(t-p+2) + B_2x_p(t-p+2) \\
&\vdots \\
y(t+p) &= Ay_p(t) + B_1u_p(t+1) + B_2x_p(t+1) \\
&\vdots \\
y(t+n) &= Ay_p(t-p+n) + B_1u_p(t-p+1+n) + B_2x_p(t-p+1+n)
\end{aligned}$$

Calculating $y_p(t+1)$ with $y_p(t)$ can simplify equation (1).

$$\begin{aligned}
\begin{bmatrix} y(t+1) \\ y(t+2) \\ \vdots \\ y(t+p) \end{bmatrix} &= \left[\begin{matrix} \begin{bmatrix} 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix} \\ A \end{matrix}\right]_{p\times p} \begin{bmatrix} y(t) \\ y(t+1) \\ \vdots \\ y(t+p-1) \end{bmatrix} + \left[\begin{matrix} \begin{bmatrix} 0 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 0 \end{bmatrix} \\ B_1 \end{matrix}\right]_{p\times p} \begin{bmatrix} u(t+1-d_1) \\ u(t+2-d_1) \\ \vdots \\ u(t+p-d_1) \end{bmatrix} \\
&\quad + \left[\begin{matrix} \begin{bmatrix} 0 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 0 \end{bmatrix} \\ B_2 \end{matrix}\right]_{p\times p} \begin{bmatrix} x(t+1-d_2) \\ x(t+2-d_2) \\ \vdots \\ x(t+p-d_2) \end{bmatrix}
\end{aligned}$$

$$y_p(t+1) = Cy_p(t) + Du_p(t+1-d_1) + Ex_p(t+1-d_2) \tag{3}$$

d_1 and d_2 is delay of control inputs and system inputs. From equation (1) and (3), $y(t+n)$ can be written with $y_p(t-p)$. It means that future output($y(t+n)$) can be composed of past information(before time step t).

For simplicity, let's assume $d_1 = d_2 = 1$.

$$
\begin{aligned}
y(t+n) &= Ay_p(t+n-p) + B_1u_p(t+n-p) + B_2x_p(t+n-p) \\
&= A\{Cy_p(t+n-p-1) + Du_p(t+n-p-1) + Ex_p(t+n-p-1)\} \\
&+ B_1u_p(t+n-p) + B_2x_p(t+n-p) \\
&= ACy_p(t+n-p-1) + ADu_p(t+n-p-1) + AEx_p(t+n-p-1) \\
&+ B_1u_p(t+n-p) + B_2x_p(t+n-p) \\
\\
&= AC\{Cy_p(t+n-p-2) + Du_p(t+n-p-2) + Ex_p(t+n-p-2)\} \\
&+ ADu_p(t+n-p-1) + AEx_p(t+n-p-1) + B_1u_p(t+n-p) + B_2x_p(t+n-p) \\
&= AC^2y_p(t+n-p-2) + ACDu_p(t+n-p-2) + ACEx_p(t+n-p-2) \\
&+ ADu_p(t+n-p-1) + AEx_p(t+n-p-1) + B_1u_p(t+n-p) + B_2x_p(t+n-p) \\
&\vdots \\
&= AC^{n-1}y_p(t-p+1) \\
&+ \sum_{i=1}^{n-1} AC^{n-(i+1)}Du_p(t+n-p+(i-n)) + B_1u_p(t+n-p) \\
&+ \sum_{i=1}^{n-1} AC^{n-(i+1)}Ex_p(t+n-p+(i-n)) + B_2x_p(t+n-p)
\end{aligned}
$$

With this equation, compute the outputs in prediction horizon.($y(t+1)$ $y(t+h)$) In next session, matrix T and U will come out. Deriving that matrix will be omit... :)

1.1 Derive T and U.

Our objective is calculating $y_h(t+1)$. First, let's look at u_p term in $y_h(t+1)$.

$$
\begin{aligned}
y(t+1) &= \sim + B_1u_p(t+1-p) \\
y(t+2) &= \sim + B_1u_p(t+2-p) \\
y(t+3) &= \sim + B_1u_p(t+3-p) \\
&\vdots \\
y(t+h-1) &= \sim + B_1u_p(t+h-1-p) \\
y(t+h) &= \sim + B_1u_p(t+h-p)
\end{aligned}
$$

can be summarized like this

$$y_h(t+1) = \sim + \overline{B_1}u_{p+h}(t+1-p)$$

in the same manner,

$$
\begin{aligned}
y(t+1) &= \sim + ADu_p(t+1-p) \\
y(t+2) &= \sim + ADu_p(t+2-p) \\
y(t+3) &= \sim + ADu_p(t+3-p) \\
&\vdots \\
y(t+h-1) &= \sim + ADu_p(t+h-1-p) \\
y(t+h) &= \sim + ADu_p(t+h-p)
\end{aligned}
$$

can be summarized like this

$$y_h(t+1) = \sim + \overline{AD}u_{p+h}(t+1-p)$$

at the end..

$$
\begin{aligned}
y(t+1) &= \sim + AC^{n-2}Du_p(t+1-p) \\
y(t+2) &= \sim + AC^{n-2}Du_p(t+2-p) \\
y(t+3) &= \sim + AC^{n-2}Du_p(t+3-p) \\
&\vdots \\
y(t+h-1) &= \sim + AC^{n-2}Du_p(t+h-1-p) \\
y(t+h) &= \sim + AC^{n-2}Du_p(t+h-p)
\end{aligned}
$$

can be summarized like this

$$y_h(t+1) = \sim + \overline{AC^{n-2}D}u_{p+h}(t+1-p)$$

So coefficient of u_p term can be written as T.

$$T = \ \overline{B_1} + \overline{AD} + \ldots + \overline{AC^{n-2}D}$$

In the same manner, coeffiecient of x_p term can be written as U.

$$U = \ \overline{B_2} + \overline{AD} + \ldots + \overline{AC^{n-2}E}$$

Coefficient of y_p term is sum of AC^n. Let's write that coefficients as V. Then

$$y_h(t+1) = Tu_{p+h-1}(t-p+1) + Ux_{p+h-1}(t-p+1) + Vy_p(t-p+1)$$

$$= \begin{bmatrix} y(t+1) \\ y(t+2) \\ \vdots \\ y(t+h) \end{bmatrix} = \begin{bmatrix} T_1 \\ T_2 \end{bmatrix}^T \left[\begin{array}{c} u(t-p+1) \\ u(t-p+2) \\ \vdots \\ u(t-1) \\ u(t) \\ \hline u(t+1) \\ \vdots \\ u(t+h-1) \end{array}\right] + \cdots$$

$$= y_h(t+1) = T_1 u_p(t-p+1) + T_2 u_{h-1}(t+1) + Ux_{p+h-1}(t-p+1) + Vy_p(t-p+1)$$

$y_h(t+1)$	= calculated outputs(T_in or Gas)
$u_p(t-p+1)$	= past control inputs
$u_{h-1}(t+1)$	= future control inputs - calculated with optimization
$x_{p+h-1}(t-p+1)$	= past and future system inputs
$y_p(t-p+1)$	= past outputs

except $y_h(t+1)$ and $u_{h-1}(t+1)$, other variables comes from dataset. To check optimization process and outcomes comfortably, matrices should be converted . Let's do little more...

$\vdots$

After some conversion, T_2 was converted to R_1 and R_2, $u_{h-1}(t+1)$ is converted to u_1 and u_2. Just group not so important variable sets to S. So our equation can be shown as

$$y_h(t+1) = R_1 u_1 + R_2 u_2 + S \tag{4}$$

To using QP(Quadratic Programming), convert matrix with final form. It's last conversion, for now... :)

$$y_h(t+1) = Hu + f$$

$$H = \begin{bmatrix} R_1 & R_2 \\ 0 & 0 \end{bmatrix}_{2h\times 2h}, \quad u = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}_{2h\times h}, \quad f = \begin{bmatrix} S \\ 0 \end{bmatrix}_{2h\times h}$$

1.2 When delay is 0

Starting from (3), delay is 0 so

$$y_p(t+1) = Cy_p(t) + Du_p(t+1) + Ex_p(t+1) \tag{5}$$

$$\begin{aligned} y(t+n) = {} & AC^{n-1}y_p(t-p+1) \\ & + \sum_{i=2}^{n} AC^{n-i}Du_p(t+n-p+(i-n)) + B_1u_p(t+n-p+1) \\ & + \sum_{i=2}^{n} AC^{n-i}Ex_p(t+n-p+(i-n)) + B_2x_p(t+n-p+1) \end{aligned}$$

$$\begin{aligned} y_h(t+1) &= Tu_{p+h-1}(t-p+2) + Ux_{p+h-1}(t-p+2) + Vy_p(t-p+1) \\ &= T_1u_{p-1}(t-p+2) + T_2u_h(t+1) + Ux_{p+h-1}(t-p+2) + Vy_p(t-p+1) \\ &= R_1u_1 + R_2u_2 + S \end{aligned}$$

$$\begin{aligned} & u = u_1, u_2 = u_h(t+1) \\ & S = T_1u_{p-1}(t-p+2) + Ux_{p+h-1}(t-p+2) + Vy_p(t-p+1) \end{aligned}$$

2 Optimization

2.1 Introduction

Objective of optimization in this problem is minimize energy usage(gas) while keeping suitable indoor temperautre(T_{in}). So first strategy is place gas in objective function, and place T_{in} in constraints. Other important constraints are integer conditions. Since u_1 is signal and u_2 is boiler set point, u_1 is binary and u_2 is integer which range within 30 to 80.

ARX model which predicts T_{in} has such good prediction accuracy, but in signal off conditions, when boiler set point changes , output of model(T_{in}) changes, which is not the expected result. So when building the T_{in} model, boiler set point variables are replaced by multiplying boiler setpoint and signal. Through this process, model is becoming more robust. Therefore, during oprimization process, boiler set point must be 0 when signal is 0. This condition integrated in to inequality constraints. Occupancy must be considered when calculating T_{in}.

ARX model which predicts $gas\,usage$ shows poor performance. It calculates non-zero values while signal is off(0). So signal is multiplying to ARX model itself.

In this problem, LP will be used first and QP(Quadratic Programming) will be. In LP, $gas\,usage$ model is used as original version, and in QP, multiplying version will be used.