Derivations

Xuan Li

1 from MPC to QP

In this section I will use a one state, one control input with quadratic loss function to illustrate the equivalence of MPC formulation with Quadratic Programming problem. In addition, this could also apply to general finite optimal control problem with a linear system dynamic.

1.1 Notations

For the one state variable, one control variable linear MPC (infinite control) problem, a general expression is

$$\begin{aligned}
\min_{\mathbf{U}} \quad & \sum_{k=0}^{N-1} q \cdot u^2(k) \\
\text{s.t} \quad & x(k+1) = Ax(k) + Bu(k) + w(k) \\
& \underline{x} \leq x(k) \leq \overline{x}, \quad \forall k = 0, 1, ..., N \\
& \underline{u} \leq u(k) \leq \overline{u}, \quad \forall k = 1, ..., N-1 \\
& x(0) = x_0
\end{aligned}$$

where x is the state, u is control input, w is disturbance, N is the optimization horizon, and $\mathbf{U} = [u(0), ..., u(N-1)]$, $\mathbf{W} = [w(0), ..., w(N-1)]$. On the other hand, with a different set of notation, a standard Quadratic Programming problem is expressed as

$$\begin{aligned}
\min_{\mathbf{X}} \quad & \mathbf{X}^T\mathbf{Q}\mathbf{X} + \mathbf{f}^T\mathbf{X} \\
\text{s.t.} \quad & \mathbf{A}\mathbf{X} \leq \mathbf{B}
\end{aligned}$$

where $\mathbf{X} \in \mathbb{R}^N, \mathbf{Q} \in \mathbb{R}^N \times \mathbb{R}^N \mathbf{f} \in \mathbb{R}^N \mathbf{A} \in \mathbb{R}^M \times \mathbb{R}^N, \mathbf{B} \in \mathbb{R}^M$. In the next part we will derive how to transfer the MPC problem into QP. The high-level ideas is to regard the $\mathbf{X}$ in QP as $\mathbf{U}$ in MPC, and reformulate the state and input constraints in MPC according to the system dynamic.

1.2 Conversion

The derivation is based on induction. First we want to get rid of x in terms of u. Since

$$\begin{aligned}
x(1) &= Ax(0) + Bu(0) + w(0) \\
x(2) &= Ax(1) + Bu(1) + w(1) \\
&= A[Ax(0) + Bu(0) + w(0)] + Bu(1) + w(1) \\
&= A^2x(0) + ABu(0) + Bu(1) + Aw(0) + w(1) \\
x(3) &= Ax(2) + Bu(2) + w(2) \\
&= A[A^2x(0) + ABu(0) + Bu(1) + Aw(0) + w(1)] + Bu(2) + w(2) \\
&= A^3x(0) + \{A^2Bu(0) + ABu(1) + Bu(2)\} + \{A^2w(0) + Aw(1) + w(2)\} \\
&...
\end{aligned}$$

from which we can infer

$$\mathbf{X} = \tilde{\mathbf{A}}\mathbf{U} + \tilde{\mathbf{B}}\mathbf{W} + \tilde{\mathbf{C}}x_0$$

where

$$\tilde{\mathbf{A}} = \begin{bmatrix} B & 0 & \cdots & \cdots & 0 \\ AB & B & \cdots & \cdots & 0 \\ A^2B & AB & B & \cdots & 0 \\ \vdots & \ddots & & & 0 \\ \vdots & & \ddots & & \vdots \\ A^{N-1}B & A^{N-2}B & \cdots & \cdots & B \end{bmatrix}$$

$$\tilde{\mathbf{B}} = \begin{bmatrix} 1 & 0 & \cdots & \cdots & 0 \\ A & 1 & \cdots & \cdots & 0 \\ A^2 & A & 1 & \cdots & 0 \\ \vdots & \ddots & & & 0 \\ \vdots & & \ddots & & \vdots \\ A^{N-1} & A^{N-2} & \cdots & \cdots & 1 \end{bmatrix}$$

and

$$\tilde{\mathbf{C}} = [A, A^2, ..., A^N]^T$$

Thus, it is easy to see that $x(k)$ could be expressed in terms of control input $\mathbf{U}$, $x_0 = x(0)$ and $w(k)$ which are predefined. In this manner, the original problem could be reformulated only using $\mathbf{U}$. To be specific, the constraints on original $x(k)$ can be expressed as

$$\begin{aligned} &\underline{x} \cdot \mathbf{1}^N \leq \tilde{\mathbf{A}}\mathbf{U} + \tilde{\mathbf{B}}\mathbf{W} + \tilde{\mathbf{C}}x_0 \leq \overline{x} \cdot \mathbf{1}^N \\ \Longrightarrow\ &\underline{x} \cdot \mathbf{1}^N - \tilde{\mathbf{B}}\mathbf{W} - \tilde{\mathbf{C}}x_0 \leq \tilde{\mathbf{A}}\mathbf{U} \leq \overline{x} \cdot \mathbf{1}^N - \tilde{\mathbf{B}}\mathbf{W} - \tilde{\mathbf{C}}x_0 \end{aligned}$$

where $\mathbf{1}^N = [1, 1, ..., 1]^T, \mathbf{1} \in \mathbb{R}^N$. On the other hand, the constraints on $u(k)$ is

$$\underline{u} \cdot \mathbf{1}^N \leq \mathbf{U} \leq \overline{u} \cdot \mathbf{1}^N$$

Lastly, define

$$\mathbf{Q} = q \cdot \mathbf{I}^N$$

where $\mathbf{I}^N \in \mathbb{R}^N \times \mathbb{R}^N$ is the diagonal matrix. Finally, if we set

$$\mathbf{A} = \begin{bmatrix} \tilde{\mathbf{A}} \\ -\tilde{\mathbf{A}} \\ \mathbf{1}^N \\ -\mathbf{1}^N \end{bmatrix}$$

$$\mathbf{B} = \begin{bmatrix} \overline{x} \cdot \mathbf{1}^N - \tilde{\mathbf{B}}\mathbf{W} - \tilde{\mathbf{C}}x_0 \\ -\left(\underline{x} \cdot \mathbf{1}^N - \tilde{\mathbf{B}}\mathbf{W} - \tilde{\mathbf{C}}x_0\right) \\ \overline{u} \cdot \mathbf{1}^N \\ -\underline{u} \cdot \mathbf{1}^N \end{bmatrix}$$

and the final QP expression is

$$\begin{aligned} \min_{\mathbf{U}} \quad & \mathbf{U}^T\mathbf{Q}\mathbf{U} \\ \text{s.t.} \quad & \mathbf{A}\mathbf{U} \leq \mathbf{B} \end{aligned}$$

2 from MPC to SQP

2.1 General Idea

In the previous section we have derive how to convert MPC to QP, getting ride of the initial state variables $x(k)$ using the iterative system dynamic. However, if we take one step back, a more straight forward equivalent expression would be

$$\begin{aligned} \min_{\mathbf{X}} \quad & \mathbf{X}^T\mathbf{Q}\mathbf{X} + \mathbf{f}^T\mathbf{X} \\ \text{s.t.} \quad & \mathbf{A}\mathbf{X} \leq \mathbf{B} \\ & \mathbf{C}\mathbf{X} = \mathbf{D} \end{aligned}$$

so that we can explicit include both control input $u(k)$ and $u(k)$. Indeed, the above formulation belongs to one canonical Sequential Quadratic Programming (SQP), which could be solved efficiently by standard solvers. Based on the steps from the previous section, we will derive how to MPC to SQP below.

2.2 Conversion

Start from MPC induction that

$$\begin{aligned} & x(1) = Ax(0) + Bu(0) + w(0) \\ \Longrightarrow\ & x(1) - Bu(0) = Ax(0) + w(0) \\ & x(2) = Ax(1) + Bu(1) + w(1) \\ \Longrightarrow\ & x(2) - Bu(1) = Ax(1) + w(1) \\ & x(3) = Ax(2) + Bu(2) + w(2) \\ \Longrightarrow\ & x(3) - Bu(2) = Ax(2) + w(2) \\ & \cdots\cdots \\ & x(N) = Ax(N-1) + Bu(N-1) + w(N-1) \\ \Longrightarrow\ & x(N) - Bu(N-1) = Ax(N-1) + w(N-1) \end{aligned}$$

Thus, if we define the vector $\tilde{X}$ as

$$\tilde{\mathbf{X}} = [x(1), x(2), ..., x(N), u(0), u(1), ..., u(N-1)]$$

matrix $\tilde{\mathbf{C}} \in \mathbb{R}^N \times \mathbb{R}^{2N}$, vector $\tilde{\mathbf{D}} \in \mathbb{R}^{2N}$ as

$$\tilde{\mathbf{C}} = \begin{bmatrix} 1 & \cdots & \cdots & \cdots & 0 & -B & \cdots & \cdots & \cdots & 0 \\ -A & 1 & \cdots & \cdots & 0 & 0 & -B & \cdots & \cdots & 0 \\ 0 & -A & 1 & \cdots & 0 & 0 & 0 & -B & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & \vdots & 0 & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & \cdots & -A & 1 & 0 & \cdots & \cdots & \cdots & -B \end{bmatrix}$$

$$\tilde{\mathbf{D}} = \begin{bmatrix} Ax(0) + w(0) \\ w(1) \\ w(2) \\ \vdots \\ w(N-1) \end{bmatrix}$$

then we can infer the equivalence that

$$\tilde{\mathbf{C}}\tilde{\mathbf{X}} = \tilde{\mathbf{D}}$$

On the other hand, define matrix $\tilde{\mathbf{A}} \in \mathbb{R}^{4N} \times \mathbb{R}^{2N}$, vector $\tilde{\mathbf{B}} \in \mathbb{R}^{2N}$, that

$$\tilde{\mathbf{A}} = \begin{bmatrix} \mathbf{I}^{2N} \\ -\mathbf{I}^{2N} \end{bmatrix}$$

$$\tilde{\mathbf{B}} = \begin{bmatrix} \overline{x} \cdot \mathbf{1}^N \\ \overline{u} \cdot \mathbf{1}^N \\ -\underline{x} \cdot \mathbf{1}^N \\ -\underline{u} \cdot \mathbf{1}^N \end{bmatrix}$$

then the inequality holds

$$\tilde{\mathbf{A}}\tilde{\mathbf{X}} \leq \tilde{\mathbf{B}}$$

Finally, define $\mathbf{Q} \in \mathbb{R}^{2N} \times \mathbb{R}^{2N}$ as

$$\mathbf{Q} = \begin{bmatrix} \mathbf{0}^N & \mathbf{0}^N \\ \mathbf{0}^N & q \cdot \mathbf{I}^N \end{bmatrix}$$

Given this, we have reformulated the above MPC problem in SQP format that

$$\begin{aligned} \min_{\tilde{\mathbf{X}}} \quad & \tilde{\mathbf{X}}^T \mathbf{Q} \tilde{\mathbf{X}} \\ \text{s.t.} \quad & \tilde{\mathbf{A}}\tilde{\mathbf{X}} \leq \tilde{\mathbf{B}} \\ & \tilde{\mathbf{C}}\tilde{\mathbf{X}} = \tilde{\mathbf{D}} \end{aligned}$$

3 from nonlinear (bilinear) MPC to Nonlinear Programming

3.1 General Idea

Now assume that bilinear term, the product of state and control variable, exists in the dynamic equation, which yields the nonlinear MPC formulation as:

$$\begin{aligned}
\min_{\mathbf{U}} \quad & \sum_{k=0}^{N-1} q \cdot u^2(k) \\
\text{s.t} \quad & x(k+1) = Ax(k) + Bu(k) + Cx(k) \cdot u(k) + w(k) \\
& \underline{x} \leq x(k) \leq \overline{x}, \quad \forall k = 0, 1, ..., N \\
& \underline{u} \leq x(u) \leq \overline{u}, \quad \forall k = 1, ..., N-1 \\
& x(0) = x_0
\end{aligned}$$

To make the problem solvable using nonlinear programming (NP) solver 'fmincon' in Matlab, it needs to be converted into the equivalent expression that

$$\begin{aligned}
\min_{\mathbf{X}} \quad & f(\mathbf{X}) \\
\text{s.t.} \quad & \underline{\mathbf{X}} \leq \mathbf{X} \leq \overline{\mathbf{X}} \\
& A_{eq}(\mathbf{X}) = \mathbf{0}
\end{aligned}$$

where $f(\cdot),\ A_{eq}(\cdot)$ are objective function and equality constraints in nonlinear format.

3.2 Conversion

The steps are similar as converting linear MPC to SQP format. Starting from MPC induction that

$$\begin{aligned}
& x(1) = Ax(0) + Bu(0) + Cx(0)u(0) + w(0) \\
\Longrightarrow\ & \{x(1)\} - B\{u(0)\} - \{Ax(0) + w(0)\} - \{Cx(0)u(0)\} = 0 \\
& x(2) = Ax(1) + Bu(1) + Cx(1)u(1) + w(1) \\
\Longrightarrow\ & \{x(2) - Ax(1)\} - B\{u(1)\} - \{w(1)\} - \{Cx(1)u(1)\} = 0 \\
& \qquad\qquad ..... \\
& x(N) = Ax(N-1) + Bu(N-1) + +Cx(N-1)u(N-1) + w(N-1) \\
\Longrightarrow\ & \{x(N) - Ax(N-1)\} - B\{u(N-1)\} - \{w(N-1)\} - \{Cx(N-1)u(N-1)\} = 0
\end{aligned}$$

Thus, define the vector $\tilde{\mathbf{X}}$ as

$$\tilde{\mathbf{X}} = [x(1), x(2), ..., x(N), u(0), u(1), ..., u(N-1)]$$

matrix $\tilde{\mathbf{C}} \in \mathbb{R}^N \times \mathbb{R}^{2N}$, vector $\tilde{\mathbf{D}} \in \mathbb{R}^{2N}$ as

$$\tilde{\mathbf{C}} = \begin{bmatrix}
1 & \cdots & \cdots & \cdots & 0 & -B & \cdots & \cdots & \cdots & 0 \\
-A & 1 & \cdots & \cdots & 0 & 0 & -B & \cdots & \cdots & 0 \\
0 & -A & 1 & \cdots & 0 & 0 & 0 & -B & \cdots & 0 \\
0 & \ddots & \ddots & \ddots & \vdots & 0 & \ddots & \ddots & \ddots & \vdots \\
0 & \cdots & \cdots & -A & 1 & 0 & \cdots & \cdots & \cdots & -B
\end{bmatrix}$$

$$\tilde{\mathbf{D}} = - \begin{bmatrix} Ax(0) + w(0) \\ w(1) \\ w(2) \\ \vdots \\ w(N-1) \end{bmatrix}$$

then the nonlinear constraints $A_{eq}(\cdot)$ can be expressed as

$$A_{eq}(\tilde{\mathbf{X}}) = \tilde{\mathbf{C}} \cdot \tilde{\mathbf{X}} + \tilde{\mathbf{D}} \cdot \tilde{\mathbf{X}} - C \cdot \tilde{\mathbf{X}}[1:N] \circ \tilde{\mathbf{X}}[N+1:2N]$$

where $\circ$ represents the element-wise product of two vector, i.e.,

$$[x_1, y_1] \circ [x_2, y_2] = [x_1 x_2,\ y_1 y_2]$$

In addition, define the corresponding lower and upper bounds as

$$\underline{\tilde{\mathbf{X}}} = \begin{bmatrix} \underline{x} \cdot \mathbf{1}^N \\ \underline{u} \cdot \mathbf{1}^N \end{bmatrix}$$

and

$$\overline{\tilde{\mathbf{X}}} = \begin{bmatrix} \overline{x} \cdot \mathbf{1}^N \\ \overline{u} \cdot \mathbf{1}^N \end{bmatrix}$$

Lastly, define

$$\mathbf{Q} = \begin{bmatrix} \mathbf{0}^N & \mathbf{0}^N \\ \mathbf{0}^N & q \cdot \mathbf{I}^N \end{bmatrix}$$

then the objective function as

$$f(\tilde{\mathbf{X}}) = \tilde{\mathbf{X}}^T \mathbf{Q} \tilde{\mathbf{X}}$$

Thus, we get the equivalent formulation as

$$\begin{aligned} \min_{\tilde{\mathbf{X}}} \quad & f(\tilde{\mathbf{X}}) \\ \text{s.t.} \quad & \underline{\tilde{\mathbf{X}}} \leq \tilde{\mathbf{X}} \leq \overline{\tilde{\mathbf{X}}} \\ & A_{eq}(\tilde{\mathbf{X}}) = \mathbf{0} \end{aligned}$$