pyMPC Documentation

Marco Forgione

May 23, 2019

Chapter 1

Mathematical formulation

The MPC problem to be solved is:

$$
\begin{aligned}
&\arg\min_{\mathbf{x},\mathbf{u}} \overbrace{\left(x_N - x_{ref}\right)^\top Q_{x_N}\left(x_N - x_{ref}\right)}^{=J_{x_N}} + \overbrace{\sum_{k=0}^{N_p-1} \left(x_k - x_{ref}\right)^\top Q_x\left(x_k - x_{ref}\right)}^{J_x} + \\
&+ \overbrace{\sum_{k=0}^{N_p-1} \left(u_k - u_{ref}\right)^\top Q_u\left(u_k - u_{ref}\right)}^{J_u} + \overbrace{\sum_{k=0}^{N_p-1} \Delta u_k^\top Q_{\Delta u}\Delta u_k}^{J_{\Delta u}} && \text{(1.1a)} \\
&\text{subject to} \\
&x_{k+1} = Ax_k + Bu_k && \text{(1.1b)} \\
&u_{min} \leq u_k \leq u_{max} && \text{(1.1c)} \\
&x_{min} \leq x_k \leq x_{max} && \text{(1.1d)} \\
&\Delta u_{min} \leq \Delta u_k \leq \Delta u_{min} && \text{(1.1e)} \\
&x_0 = \bar{x} && \text{(1.1f)} \\
&u_{-1} = \bar{u} && \text{(1.1g)}
\end{aligned}
$$

where $\Delta u_k = u_k - u_{k-1}$.

The optimization variables are

$$
\begin{aligned}
&\mathbf{x} = \begin{bmatrix} x_0 & x_1 & \dots & x_{N_p} \end{bmatrix}, && \text{(1.2)} \\
&\mathbf{u} = \begin{bmatrix} u_0 & u_1 & \dots & u_{N_p-1} \end{bmatrix}, && \text{(1.3)} \\
& && \text{(1.4)}
\end{aligned}
$$

In a typical implementation, the MPC input is applied in *receding horizon.* At each time step i, the problem (1.1) is solved with $x_0 = x[i]$, $u_{-1} = u[i-1]$ and an optimal input sequence $u_0, \dots, u_{N_p}$ is obtained. The first element of this sequence u_0 is the control input that is actually applied at time instant i. At time instant $i+1$, a new state $x[i+1]$ is measured (or estimated), and the process is iterated.

Thus, formally, the MPC control law is a (static) function of the current state and the previous input:

$$
u_{MPC} = K(x[i], u[i-1]). \tag{1.5}
$$

Note that this function also depends on the references x_{ref} and u_{ref} and on the system matrices A and B.

1.1 Quadratic Programming Formulation

The QP solver expets a problem with form:

$$\min \frac{1}{2} x^\top P x + q^\top x \tag{1.6a}$$

subject to

$$l \leq Ax \leq u \tag{1.6b}$$

The challenge here is to rewrite the MPC optimization problem (1.1) in form (1.6) to use the standard QP solver.

1.1.1 Cost function

By direct inspection, the non-constant terms of the cost function in Q_x are:

$$\begin{aligned} J_{Q_u} = \frac{1}{2} \begin{bmatrix} u_0 & u_1 & \dots & u_{N_p-1} \end{bmatrix}^\top \mathrm{blkdiag}(Q_x, Q_x, \dots, Q_x) \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{N_p-1} \end{bmatrix}^\top + \\ + \begin{bmatrix} -x_{ref}^\top Q_x & -x_{ref}^\top Q_x & \dots & -x_{ref}^\top Q_x \end{bmatrix}^\top \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{N_p-1} \end{bmatrix}^\top \end{aligned} \tag{1.7}$$

and similarly for the term in $Q_{x_{N_p}}$ and the terms in Q_u. For the terms in $Q\Delta u$ we have instead

$$\begin{aligned} J_{\Delta u} = \frac{1}{2} \begin{bmatrix} u_0 & u_1 & \dots & u_{N_p-1} \end{bmatrix}^\top \begin{bmatrix} 2Q_{\Delta u} & -Q_{\Delta u} & 0 & \dots & \dots & 0 \\ -Q_{\Delta u} & 2Q_{\Delta u} & -Q_{\Delta u} & 0 & \dots & 0 \\ 0 & -Q_{\Delta u} & \ddots & \ddots & \ddots & 0 \\ 0 & 0 & \ddots & \ddots & \ddots & 0 \\ 0 & 0 & \ddots & \ddots & \ddots & 0 \\ 0 & 0 & 0 & -Q_{\Delta u} & 2Q_{\Delta u} & -Q_{\Delta u} \\ 0 & 0 & 0 & 0 & -Q_{\Delta u} & Q_{\Delta u} \end{bmatrix} \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{N_p-1} \end{bmatrix}^\top + \\ + \begin{bmatrix} -\bar{u}^\top Q_{\Delta u} & 0 & \dots & 0 \end{bmatrix} \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{N_p-1} \end{bmatrix}^\top \end{aligned} \tag{1.8}$$

1.1.2 Constraints

Linear dynamics

Let us consider the linear equality constraints (1.1c) representing the system dynamcs. These can be written in matrix form as

$$\begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_{N_p-1} \\ x_{N_p} \end{bmatrix} = \overbrace{\begin{bmatrix} 0 & 0 & \dots & 0 \\ A_d & 0 & \dots & 0 \\ 0 & A_d & \dots & 0 \\ \vdots & 0 & \ddots & 0 \\ 0 & 0 & \dots & A_d \end{bmatrix}}^{=\mathcal{A}} \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_{N_p-1} \\ x_{N_p} \end{bmatrix} + \overbrace{\begin{bmatrix} 0 & 0 & \dots & 0 \\ B_d & 0 & \dots & 0 \\ 0 & B_d & \dots & 0 \\ \vdots & 0 & \ddots & 0 \\ 0 & 0 & \dots & B_d \end{bmatrix}}^{=\mathcal{B}} \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{N_p-2} \\ u_{N_p-1} \end{bmatrix} + \overbrace{\begin{bmatrix} \bar{x} \\ 0 \\ \vdots \\ \vdots \\ 0 \end{bmatrix}}^{=\mathcal{C}} \tag{1.9}$$

we get a set of linear equality constraints representing the system dynamics (1.1c). These constraints can be written as

$$\begin{bmatrix} \mathcal{A} - I & \mathcal{B} \end{bmatrix} \begin{bmatrix} x \\ u \end{bmatrix} = \mathcal{C}. \tag{1.10}$$

Variable bounds

Bounds on x and u are readily implemented as

$$\begin{bmatrix} x_{min} \\ u_{min} \end{bmatrix} \leq \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} x \\ u \end{bmatrix} \leq \begin{bmatrix} x_{max} \\ u_{max} \end{bmatrix}. \tag{1.11}$$

However, bounds on x may make the problem unfeasible! A common solution is to transform the constraints in x into soft constraints by means of *slack variables* ϵ.

$$\begin{bmatrix} x_{min} \\ u_{min} \end{bmatrix} \leq \begin{bmatrix} I & 0 & I \\ 0 & I & 0 \end{bmatrix} \begin{bmatrix} x \\ u \\ \epsilon \end{bmatrix} \leq \begin{bmatrix} x_{max} \\ u_{max} \end{bmatrix} \tag{1.12}$$

.