Model Predictive Control_Part 2

In our previous discussion in Part 1, we explored the concept of Model Predictive Control (MPC) and highlighted its significance. Now, in this partition, we will delve into the formulation of the algorithms introduced earlier. If you're a general reader, there's no need to panic, as basic algebra is all you need to grasp the information presented.

From previous post, we already know that on one hand, Dynamic Matrix Control (DMC), Model Algorithmic Control (MAC), and Generalized Predictive Control (GPC) often involve frequency domain considerations. DMC typically relies on analyzing and utilizing the system's step response, while MAC depends on the impulse response. GPC, in contrast, incorporates aspects from both the step and impulse responses.

On the other hand, Predictive Functional Control (PFC) operates in the time domain and formulates control strategies based on the state-space representation of the system. In this approach, understanding the impulse and step responses, as well as grasping the concept of state space representation, becomes essential. One might ask, what is an impulse- or a step- response or what is a state-space representation. Let’s get some help from the technology and ask ChatGPT to give us a generic answer for each question above.

*What is an impulse response?

“Impulse response refers to the output of a system when it is subjected to an impulse input, often represented by the Dirac delta function. An impulse input is an instantaneous and infinitesimally short signal with a very high amplitude.

In the context of systems analysis, especially in fields like control theory and signal processing, the impulse response provides crucial information about the behavior and characteristics of a system. By applying an impulse input to a system and observing its output over time, the impulse response graphically illustrates how the system responds to sudden, momentary changes.

Mathematically, the impulse response is often denoted by "$h(t)$" in continuous time or "$h[n]$" in discrete time, where "t" represents time and "n" denotes discrete time instances. The convolution of the system's impulse response with an input signal yields the system's output for that input.

Understanding the impulse response is fundamental in analyzing and designing systems, as it allows engineers and researchers to predict system behavior to various inputs and assess stability, performance, and other characteristics.”

Based on this description, it is easy to suggest the following relation between impulse input $u[n]$, and output $y[n]$ of a system in discrete time domain:

$$y[n] = \sum_{i=1}^{\infty} h_i u[n-i] \tag{1}$$

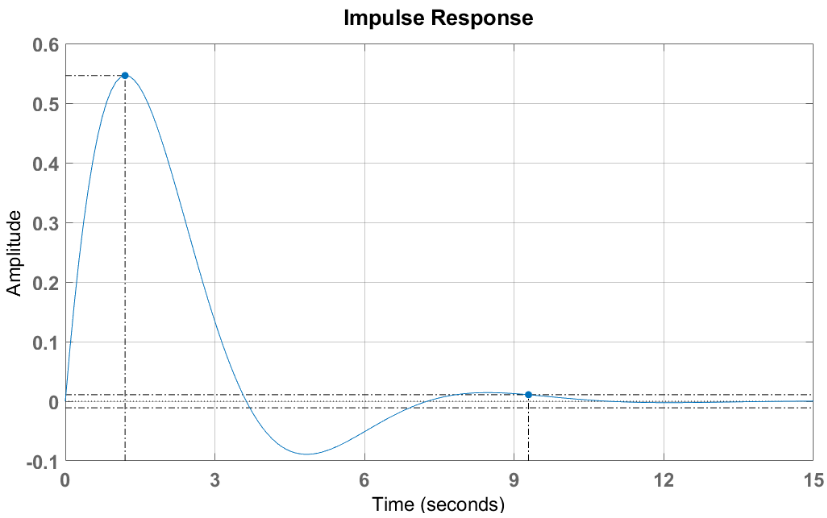

Figure 1: an example of impulse-response plot of a system

***What is a step response?**

"A step response is a common analysis tool in the fields of control theory and system dynamics. It refers to the output of a system when it is subjected to a step input, which is a sudden and permanent change in the input signal. The step input is often represented as a unit step function, which abruptly transitions from one constant value to another.

In practical terms, a step response illustrates how a system reacts to a change in its input over time. It is particularly useful in evaluating the dynamic behavior, stability, and performance of a system.

The key features of a step response typically include:

Rise Time: The time it takes for the system's output to go from a specified lower value to a specified higher value for the first time.

Settling Time: The time it takes for the system's output to reach and remain within a specified range around the final steady-state value.

Overshoot: The extent by which the system's output exceeds the final steady-state value before settling.

Steady-state Value: The final output value that the system approaches as time goes to infinity.

Analyzing the step response helps engineers and researchers understand how a system behaves in response to sudden changes, and it provides valuable insights into the system's stability, transient behavior, and overall performance. The step response is often graphically represented, showing the system's output as a function of time following the application of a step input."

Denoting the step response by "$g(t)$" in continuous time or "$g[n]$" in discrete time, where "t" represents time and "n" denotes discrete time instances, $u[n]$ representing the step input. The convolution of the system's step response with an input signal yields the system's output for that input and can be formulated as follows.

$$y[n] = \sum_{i=1}^{n} g_i[u[n-i] - u[n-i-1]] = \sum_{i=1}^{n} g_i \Delta u[n-i] \quad (2)$$

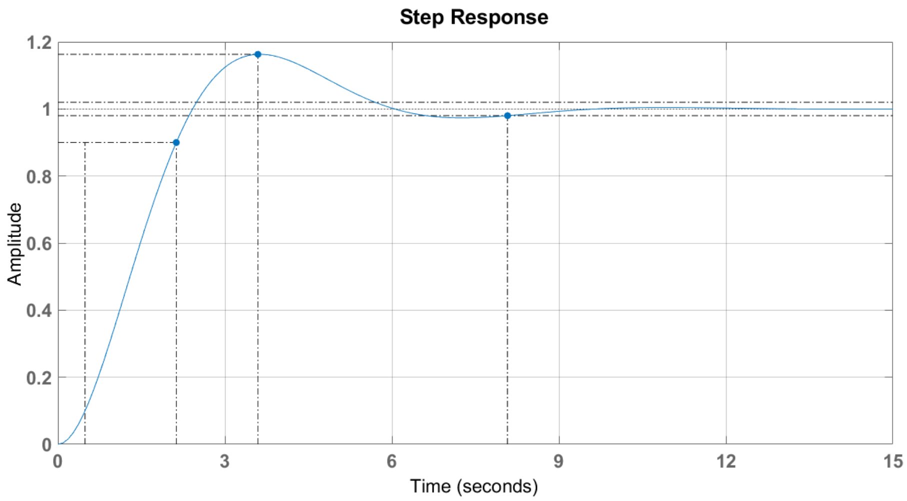

Figure 2: an example of step-response plot of a system

*** What is a state-space representation?**

"A state-space representation is a mathematical model used to describe the dynamic behavior of a system in control theory and engineering. It is a compact and comprehensive way to represent a system's dynamics, making it particularly useful for analysis, simulation, and control design. The representation involves two main components: state variables and state equations.

State Variables (x): These are a set of variables that describe the internal state of the system. They capture the essential information needed to predict the future behavior of the system. The number of state variables corresponds to the order of the system.

State Equations: These are a set of first-order differential or different equations that define how the state variables evolve over time. State equations are typically expressed in matrix form, representing the system's dynamics and how the state variables change with respect to time."

By considering $x(t)$, $u(t)$ and $y(t)$ to be states, inputs and outputs vectors, respectively. And A, B, C and D to be system parameters' matrices, the general form of a state-space representation of a time invariant continuous system can be formulated as

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) + Du(t) \end{aligned} \quad (3)$$

It is to discretize it as (4).

$$\begin{aligned} x[n] &= Ax[n] + Bu[n] \\ y[n] &= Cx[n] + Du[n] \end{aligned} \quad (4)$$

Having acquired the necessary knowledge in control theory, let's delve into formulating MPC schemes. In the list the first scheme is Dynamic Matrix Control (DMC). To accomplish this task, it is crucial to formulate the predicted output. By leveraging the connection between input and output, as delineated in equation (2), a straightforward suggestion emerges:

$$\hat{y}[n+k] = \sum_{i=1}^{\infty} g_i \Delta u[n+k-i] + \hat{n}[n+k] \tag{5}$$

breaking the sigma in (5) into future and past inputs, yields to

$$\hat{y}[n+k] = \underbrace{\sum_{i=1}^{k} g_i \Delta u[n+k-i]}_{future\ inputs} + \underbrace{\sum_{i=k+1}^{\infty} g_i \Delta u[n+k-i] - \sum_{i=1}^{\infty} g_i \Delta u[n-i]}_{past\ inputs + prediction\ error} \tag{6}$$

since the system is stable, one can modify (6) as

$$\begin{aligned} \hat{y}[n+k] &= \sum_{i=1}^{k} g_i \Delta u[n+k-i] + \sum_{i=1}^{N} (g_{k+i} - g_i) \Delta u[n-i] \\ &= \sum_{i=1}^{k} g_i \Delta u[n+k-i] + f[n+k] \end{aligned} \tag{7}$$

now, given the prediction horizon (p) and control horizon (m), one can articulate as follows.

$$\begin{aligned} \hat{y}[n+1|n] &= g_1 \Delta u[n] + f[n+1] \\ \hat{y}[n+2|n] &= g_2 \Delta u[n] + g_1 \Delta u[n+1] + f[n+2] \\ &\vdots \\ \hat{y}[n+p|n] &= \sum_{i=p-m+1}^{p} g_i \Delta u[n+p-i] + f[n+p] \end{aligned} \tag{8}$$

using matrix representation for (8), yields to

$$\widehat{\boldsymbol{Y}} = \boldsymbol{Gu} + \ \mathbf{Z}\boldsymbol{U}_{-} = \boldsymbol{Gu} + \ \boldsymbol{f} \tag{9}$$

where

$$\widehat{\boldsymbol{Y}} = \begin{bmatrix} \hat{y}[n+1|n] \\ \hat{y}[n+2|n] \\ \vdots \\ \hat{y}[n+p|n] \end{bmatrix}, \boldsymbol{u} = \begin{bmatrix} \Delta u[n] \\ \Delta u[n+1] \\ \vdots \\ \Delta u[n+p] \end{bmatrix}, \boldsymbol{U}_{-} = \begin{bmatrix} \Delta u[n-1] \\ \Delta u[n-2] \\ \vdots \\ \Delta u[n-N] \end{bmatrix} \tag{10}$$

$$\boldsymbol{G} = \begin{bmatrix} g_1 & 0 & \cdots & 0 \\ g_2 & g_1 & & 0 \\ \vdots & & \ddots & \vdots \\ g_m & g_{m-1} & & g_1 \\ \vdots & \vdots & \cdots & \vdots \\ g_p & g_{p-1} & & g_{p-m+1} \end{bmatrix}$$

$$\mathbf{Z} = \begin{bmatrix} g_2 - g_1 & \cdots & g_{N+1} - g_N \\ \vdots & \ddots & \vdots \\ g_{p+1} - g_1 & \cdots & g_{N+p} - g_1 \end{bmatrix} \tag{11}$$

as discussed, this scheme employs a predictive control strategy, wherein the control inputs are dynamically adjusted over time to minimize the predicted error in the output. Consequently, the following cost function is proposed for optimization, and the solutions derived from optimizing this function yield the desired control inputs.

$$j = \left(\widehat{\boldsymbol{Y}} - \boldsymbol{Y}\right)^T \boldsymbol{Q}\left(\widehat{\boldsymbol{Y}} - \boldsymbol{Y}\right) + \boldsymbol{u}^T \boldsymbol{R}\boldsymbol{u} \tag{12}$$

By substituting equation (9) into equation (12) and subsequently taking the derivative with respect to u, one can obtain

$$\frac{\boldsymbol{d}j}{\boldsymbol{du}} = \frac{\boldsymbol{d}}{\boldsymbol{du}}((\boldsymbol{Gu} + \boldsymbol{f} - \boldsymbol{Y})^T \boldsymbol{Q}(\boldsymbol{Gu} + \boldsymbol{f} - \boldsymbol{Y}) + \boldsymbol{u}^T \boldsymbol{Ru}) \tag{13}$$

locate an optimal solution by setting the derivative of j with respect to $\boldsymbol{u}$ equal to zero ($\frac{\boldsymbol{dj}}{\boldsymbol{du}}$).

$$\boldsymbol{G}^T \boldsymbol{Q}(\boldsymbol{Gu} + \boldsymbol{f} - \boldsymbol{Y}) + \boldsymbol{Ru} = \boldsymbol{0} \tag{14}$$

It is straightforward to calculate $\boldsymbol{u}$ as

$$\boldsymbol{u} = (\boldsymbol{G}^T \boldsymbol{QG} + \boldsymbol{R})^{-1} \boldsymbol{G}^T \boldsymbol{Q}(\boldsymbol{f} - \boldsymbol{Y}) \tag{15}$$

following the Receding Horizon lemma, it is imperative to apply solely the first the first element of the control input, $\boldsymbol{u}$, and subsequently recompute the optimal control values for the next iteration. Now, the DMC scheme is formulated. Let's delve into the second scheme in the list, Model Algorithmic Control (MAC). As previously explained, MAC closely resembles DMC with a minor adjustment; rather than utilizing the step-response of a system, MAC employs its impulse-response. Consequently, (8) can be reformulated with respect to (1), incorporating the impulse-response to express the predicted outputs in MAC as follows

$$\begin{gathered}
\hat{y}[n+1|n] = h_1 u[n] + h_2 u[n-1] + h_3 u[n-2] + \cdots + h_N u[n-N+1] \\
\hat{y}[n+2|n] = h_2 \Delta u[n] + h_1 u[n+1] + h_3 u[n-1] + h_4 u[n-2] + \cdots + h_N u[n-N+2] \\
\vdots \\
\hat{y}[n+p|n] = \sum_{i=p-m+1}^{p} h_i u[n+p-i] + \sum_{i=p-m+1}^{N} h_i u[n+p-i]
\end{gathered} \tag{16}$$

using matrix representation

$$\widehat{\boldsymbol{Y}} = \boldsymbol{Hu} + \boldsymbol{\mathcal{H}U_-} = \boldsymbol{Hu} + \boldsymbol{\mathcal{F}} \tag{17}$$

$$\widehat{\boldsymbol{Y}} = \begin{bmatrix} \hat{y}[n+1|n] \\ \hat{y}[n+2|n] \\ \vdots \\ \hat{y}[n+p|n] \end{bmatrix}, \boldsymbol{u} = \begin{bmatrix} u[n] \\ u[n+1] \\ \vdots \\ u[n+m-1] \end{bmatrix}, \boldsymbol{U_-} = \begin{bmatrix} u[n-N+1] \\ u[n-N+2] \\ \vdots \\ u[n-1] \end{bmatrix} \tag{18}$$

$$\begin{gathered}
\boldsymbol{H} = \begin{bmatrix} h_1 & 0 & \cdots & 0 \\ h_2 & h_1 & & 0 \\ \vdots & & \ddots & \vdots \\ h_m & h_{m-1} & \cdots & h_1 \end{bmatrix} \\
\boldsymbol{\mathcal{H}} = \begin{bmatrix} h_N & \dots & h_i & \cdots & h_2 \\ 0 & \dots & h_j & & h_3 \\ & \vdots & & \ddots & \vdots \\ 0 & \dots & h_N & \cdots & h_{m+1} \end{bmatrix}
\end{gathered} \tag{19}$$

suggesting the identical cost function as presented in (12) and replicating the process leads to calculation of control inputs as

$$\boldsymbol{u} = (\boldsymbol{H}^T \boldsymbol{QH} + \boldsymbol{R})^{-1} \boldsymbol{H}^T \boldsymbol{Q}(\boldsymbol{\mathcal{F}} - \boldsymbol{Y}) \tag{20}$$

same as DMC, following the Receding Horizon lemma, it is crucial to apply only the first sample of the control input, $\boldsymbol{u}$, and subsequently recompute the optimal control values. This is how MAC scheme is formulated. The third scheme listed is Predictive Function Control (PFC). Referring back to the state-space realization in equation (4), recognizing that the systems are proper leads to $D = 0$ in the state-space representation. Consequently, we can propose the following model for predicting the output in the state-space realization.

$$\begin{aligned}\hat{\boldsymbol{x}}[n+1] &= \boldsymbol{A}_{\wp}\hat{\boldsymbol{x}}[n] + \boldsymbol{B}_{\wp}\boldsymbol{u}[n] \\ \hat{\boldsymbol{y}}[n] &= \boldsymbol{C}_{\wp}[n]\end{aligned} \tag{21}$$

Adding an integrator to the system to reduce the steady state error, results in

$$\begin{aligned}\hat{\boldsymbol{y}}[n+1] - \hat{\boldsymbol{y}}[n] &= \boldsymbol{C}_{\wp}\big[\hat{\boldsymbol{x}}[n+1] - \hat{\boldsymbol{x}}[n]\big] \\ &= \boldsymbol{C}_{\wp}\Big[\boldsymbol{A}_{\wp}\hat{\boldsymbol{x}}[n] + \boldsymbol{B}_{\wp}\boldsymbol{u}[n] - \boldsymbol{A}_{\wp}\hat{\boldsymbol{x}}[n-1] - \boldsymbol{B}_{\wp}\boldsymbol{u}[n-1]\Big] \\ &= \boldsymbol{C}_{\wp}\boldsymbol{A}_{\wp}\big[\hat{\boldsymbol{x}}[n] - \hat{\boldsymbol{x}}[n-1]\big] + \boldsymbol{C}_{\wp}\boldsymbol{B}_{\wp}\big[\boldsymbol{u}[n] - \boldsymbol{u}[n-1]\big] \\ &= \boldsymbol{C}_{\wp}\boldsymbol{A}_{\wp}\Delta\hat{\boldsymbol{x}}[n] + \boldsymbol{C}_{\wp}\boldsymbol{B}_{\wp}\Delta\boldsymbol{u}[n]\end{aligned} \tag{22}$$

selecting $\boldsymbol{x}[n] = [\Delta\hat{\boldsymbol{x}}[n] \quad \hat{\boldsymbol{y}}[n]]^T$ as the updated state vector, the revised state-space formulation is expressed as

$$\begin{aligned}\overbrace{\begin{bmatrix}\Delta\hat{\boldsymbol{x}}[n+1] \\ \hat{\boldsymbol{y}}[n+1]\end{bmatrix}}^{\boldsymbol{x}[\boldsymbol{n}+\boldsymbol{1}]} &= \overbrace{\begin{bmatrix}\boldsymbol{A}_{\wp} & \boldsymbol{0} \\ \boldsymbol{C}_{\wp}\boldsymbol{A}_{\wp} & \boldsymbol{I}\end{bmatrix}}^{A}\boldsymbol{x}[n] + \overbrace{\begin{bmatrix}\boldsymbol{B}_{\wp} \\ \boldsymbol{C}_{\wp}\boldsymbol{B}_{\wp}\end{bmatrix}}^{B}\Delta\boldsymbol{u}[n] \\ \boldsymbol{y}[n] &= \underbrace{[\boldsymbol{0} \quad \boldsymbol{I}]}_{\boldsymbol{C}}\,\boldsymbol{x}[n]\end{aligned} \tag{23}$$

with the provided prediction horizon (p) and control horizon (m), expressing the predictions can be easily accomplished as follows

$$\begin{aligned}\boldsymbol{y}[n+1] &= \boldsymbol{C}\boldsymbol{x}[n+1] = \boldsymbol{C}\boldsymbol{A}\boldsymbol{x}[n] + \boldsymbol{C}\boldsymbol{B}\Delta\boldsymbol{u}[n] \\ \boldsymbol{y}[n+2] &= \boldsymbol{C}\boldsymbol{A}^2\boldsymbol{x}[n] + \boldsymbol{C}\boldsymbol{A}\boldsymbol{B}\Delta\boldsymbol{u}[n] + \boldsymbol{C}\boldsymbol{B}\Delta\boldsymbol{u}[n+1] \\ \boldsymbol{y}[n+3] &= \boldsymbol{C}\boldsymbol{A}^3\boldsymbol{x}[n] + \boldsymbol{C}\boldsymbol{A}^2\boldsymbol{B}\Delta\boldsymbol{u}[n] + \boldsymbol{C}\boldsymbol{A}\boldsymbol{B}\Delta\boldsymbol{u}[n+1] + \boldsymbol{C}\boldsymbol{B}\Delta\boldsymbol{u}[n+2] \\ &\vdots \\ \boldsymbol{y}[n+p] &= \boldsymbol{C}\boldsymbol{A}^p\boldsymbol{x}[n] + \boldsymbol{C}\boldsymbol{A}^{p-1}\boldsymbol{B}\Delta\boldsymbol{u}[n] + \boldsymbol{C}\boldsymbol{A}^{p-2}\boldsymbol{B}\Delta\boldsymbol{u}[n+1] + \cdots + \boldsymbol{C}\boldsymbol{A}^{p-m}\boldsymbol{B}\Delta\boldsymbol{u}[n+2]\end{aligned} \tag{24}$$

leveraging matrices to present the predictions, one can have

$$\boldsymbol{Y} = \mathbb{A}\boldsymbol{x}[\boldsymbol{n}] + \mathbb{B}\mathbb{U} \tag{25}$$

where

$$\boldsymbol{Y} = \begin{bmatrix}y[n+1] \\ y[n+2] \\ \vdots \\ y[n+p]\end{bmatrix}, \mathbb{A} = \begin{bmatrix}\boldsymbol{CB} & \boldsymbol{0} & \boldsymbol{0} & & \boldsymbol{0} \\ \boldsymbol{CAB} & \boldsymbol{CB} & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ \boldsymbol{CA}^2\boldsymbol{B} & \boldsymbol{CAB} & \boldsymbol{CB} & & \boldsymbol{0} \\ & \vdots & & \ddots & \vdots \\ \boldsymbol{CA}^{p-1}\boldsymbol{B} & \boldsymbol{CA}^{p-2}\boldsymbol{B} & \boldsymbol{CA}^{p-3}\boldsymbol{B} & \cdots & \boldsymbol{CA}^{p-m}\boldsymbol{B}\end{bmatrix} \tag{26}$$

and to determine the values for control inputs, it is crucial to introduce a cost function and minimize it over the prediction horizon. A potential candidate can be identified in equation (27).

$$J = (\boldsymbol{Y} - \boldsymbol{Y}_s)^T\boldsymbol{Q}(\boldsymbol{Y} - \boldsymbol{Y}_s) + \mathbb{U}^T\boldsymbol{R}\mathbb{U} \tag{27}$$

Obtaining an optimal solution involves setting the derivative of the cost function with respect to the control inputs ($\frac{dJ}{d\mathbb{U}}$) equal to zero. This leads to the calculation of $\mathbb{U}$ as:

$$\mathbb{U} = \left(\mathbb{B}^{\boldsymbol{T}}\boldsymbol{Q}\mathbb{B} + \boldsymbol{R}\right)^{-1}\mathbb{B}^T\boldsymbol{Q}(\boldsymbol{Y}_s - \mathbb{A}\boldsymbol{x}[\boldsymbol{n}]) \tag{28}$$

similarly, following the Receding Horizon lemma, it is imperative to apply solely the first element of control input sequence, $\mathbb{U}$, and subsequently recompute the optimal control values for the next time step. The final method in the list is GPC, a model-based predictive control approach that initiates with system identification. GPC, depending on the specific modeling and optimization techniques employed during system identification, integrates features of predictive control akin to DMC, MAC, or PFC, rendering it a versatile and comprehensive control strategy. For instance, if the identification process yields a step-response, GPC is applied following the DMC approach. In the presence of an available impulse response, GPC is implemented using the MAC method. Likewise, a parallel relationship exists for state-space realization between PFC and GPC.