Model Predictive Control

- In the lectures so far, we have looked at techniques to find **reduced-order models** for physical systems and have derived conditions for system **stability**

- In the final lectures we will look at **control**

- There are a vast array of control methods available, but I want to give a very brief overview of one general method known **as Model Predictive Control (MPC)**.

- This is a versatile technique and is applicable to nonlinear systems. Since these often arise in fluid structure interactions, it may be of some interest to you!

3. MPC problem set-up

- Will consider only discrete time control systems

$$x_{k+1} = f(x_k, u_k)$$
$$x_0 \in \mathbb{R}^n$$

- Interpret / assume:

(i) $x_k \in \mathbb{R}^n$ is the state at time t_k

(ii) $u_k \in \mathbb{R}$ is a control input at time t_k

(iii) $f : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^n$ is continuous

(iv) $f(0,0) = 0$ is an equilibrium point of the control system

Goal: find a control strategy

$$u_k = \kappa(x_k), \qquad \kappa : \mathbb{R}^n \to \mathbb{R}$$

to ensure asymptotic stability of closed-loop dynamics

$$x_{k+1} = f(x_k, \kappa(x_k))$$

3. MPC ingredients

- The key idea of MPC is to use the model to **predict** the controlled response of the system over a **horizon** of N future timesteps

- An optimization problem is solved to pick a sequence of optimal future control inputs over the future horizon

- **Only the first** optimal input is applied, and the system evolves over one timestep.

- The optimization process is then repeated, treating the new state of the system as the initial condition.

3. MPC ingredients

- The key idea of MPC is to use the model to **predict** the controlled response of the system over a **horizon** of N future timesteps

- An optimization problem is solved to pick a sequence of optimal future control inputs over the future horizon

- **Only the first** optimal input is applied, and the system evolves over one timestep.

- The optimization process is then repeated, treating the new state of the system as the initial condition.

3.1 Definitions for MPC

Definition 7. *The following definitions will be used to build up an MPC scheme.*

i) **Stage Cost** $L : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}_+$ *is a positive, continuously differentiable, function which quantifies the desirability of the state* $x \in \mathbb{R}^n$ *and the penalty of applying the control action* u.

A typical choice is the quadratic cost

$$L(x, u) = x^\top Q x + \rho |u|^2$$

where $Q \succ 0$ *is an* $n \times n$ *matrices and* $\rho > 0$

ii) **Horizon Length**: $N \in \mathbb{N}$. *The number of steps used to determine the current control action.*

iii) Let $\boldsymbol{u}_{[0:N-1]} = (u_0, u_1, \ldots, u_{N-1}) \in \mathbb{R}^N$ *be a potential sequence of control inputs which can be applied over the prediction horizon.*

iv) Let $\phi(k; x, \boldsymbol{u})$ *be the state of the system after* $k \leq N$ *steps, given an initial condition of* x *and the control sequence* $\boldsymbol{u}_{[0:N-1]} = (u_0, u_1, \ldots, u_{N-1})$. *For example,*

$$\phi(1; x, u_0) = f(x, u_0),$$

and

$$\phi(2; x, (u_0, u_1)) = f(f(x, u_0), u_1),$$

and so on.

v) **Constraints:** *Let* $0 \in U \subseteq \mathbb{R}$ *and* $0 \in X \subseteq \mathbb{R}^n$ *be compact constraint sets for the control input and state.*

vi) **Final State Constraint**: *Let* $0 \in X_f$ *be a compact set which we will require the state at the end of the prediction horizon to belong to.*

vii) **Final Stage Cost:** $V_f : X_f \to \mathbb{R}_+$ *is positive, continuously differentiable, and quantifies the desiribility of the state at the end of the prediction horizon.*

3.1 MPC Optimization Problem

- Given the system is currently at state $x \in \mathbb{R}^n$ the idea of MPC is to solve

$$
\begin{aligned}
\min_{\boldsymbol{u}} \quad & \sum_{k=0}^{N-1} L(\hat{x}_k, u_k) + V_f(\hat{x}_N) \\
\text{subject to} \quad & \hat{x}_{j+1} = f(\hat{x}_j, u_j), \quad j = 0, \dots, N-1, \\
& \hat{x}_0 = x, \\
& \hat{x}_k \in X, \qquad k = 1, \dots, N, \\
& \boldsymbol{u} = (u_0, \dots, u_{N-1}) \in U^N \\
& \hat{x}_N \in X_f.
\end{aligned}
$$

3.1 More MPC Notation

Definition 8. *Some simplifying notation:*

1. *The Value Function* $V_N : \mathbb{R}^n \times \mathbb{R}^N \to \mathbb{R}$ *is given by*

$$V_N(x, \boldsymbol{u}) := \sum_{k=0}^{N-1} L(\hat{x}_k, u_k) + V_f(\hat{x}_N)$$

where it is implicit in the above equation that the states $(\hat{x}_k)_{k=0}^{N}$ *depend on* $(x, \boldsymbol{u})$ *via* $x_{k+1} = f(x_k, u_k)$.

2. *For any* $x \in \mathbb{R}^n$, *let*

$$\mathcal{U}_N(x) := \left\{\boldsymbol{u} \in \mathbb{R}^N : \hat{x}_k \in X, \text{ for } k = 1, \ldots, N-1, \text{ and } \hat{x}_N \in X_f\right\}$$

be the set of control inputs $\boldsymbol{u} \in \mathbb{R}^N$ *which create trajectories over the prediction horizon which satisfy the constraints. These are called feasible or admissible control inputs.*

3. *Let* $\mathcal{X}_N = \{x \in X : \mathcal{U}_N(x) \neq \emptyset\}$ *be the set of system states for which there exists a control sequence to maintain feasibility.*

3.1 Closed loop MPC

- Can now formally define MPC feedback law and algorithm

Feedback law:

1. Suppose that
$$x \in \mathcal{X}_N \subset \mathbb{R}^n$$
2. Define optimal control inputs

$$\boldsymbol{u}^*(x) \in \operatorname{argmin}\Big\{V_N(x, \boldsymbol{u}) : \boldsymbol{u} \in \mathcal{U}_N(x)\Big\}. \qquad (P_N(x))$$

   where

$$\boldsymbol{u}^*(x) = (u_0^*(x), u_1^*(x), \ldots, u_{N-1}^*(x))$$

3. The MPC feedback law $\kappa_N : \mathbb{R}^n \to \mathbb{R}$ is defined by

$$\kappa_N(x) := u_0^*(x)$$

Closed loop MPC: Given the feedback law

$$x_{k+1} = f(x_k, \kappa_N(x_k)), \qquad k \geq 0$$

3.2 A visual overview of MPC

- The statement $x \in \mathcal{X}_N$ just says that the state can be drive to the final state constraint set in N steps.

- Since the number of steps is arbitrary, we can build up this idea using a sequence of nested subsets.

$$\mathcal{U}_j(x) = \{\boldsymbol{u} \in U^j : (\phi(k; x, \boldsymbol{u}))_{k=1}^{j-1} \subset X \text{ and } \phi(j; x, \boldsymbol{u}) \in X_f\}$$

$$\mathcal{X}_j = \{x \in X : \mathcal{U}_j(x) \neq 0\}$$

3.2 A visual overview of MPC

Definition 9 (Invariant Sets). *Given a system $x_{k+1} = f(x_k)$:*

1. *A set $X \subset \mathbb{R}^n$ is* **positive invariant** *if for any $x \in X$ it follows that $f(x) \in X$.*
2. *A set $X \subset \mathbb{R}^n$ is* **control invariant** *if for any $x \in X$ there exists $u \in U$ such that $f(x, u) \in X$.*

Lemma 1. *Suppose that X_f is control invariant. Define $\mathcal{X}_0 := X_f$. Then*

(i) For any $j \geq 1$,

$$\mathcal{X}_j = \{x \in X : \exists u \in U \text{ such that } f(x, u) \in \mathcal{X}_{j-1}\}$$

(ii) $\mathcal{X}_j \subseteq \mathcal{X}_{j+1}$, for any $j \geq 0$.

3.3 Dynamic Programming

- Given the nested sets, it is natural to define a series of MPC-like optimisation problems with different horizon lengths.

- Define the **stage costs** $V_j : \mathcal{X}_j \times \mathcal{U}_j \to \mathbb{R}$ by

$$V_j(x, \boldsymbol{u}) := \sum_{k=0}^{j-1} L(\hat{x}_k, u_k) + V_f(\hat{x}_j), \qquad j \geq 1,$$

- Define the **optimal costs** by

$$V_j^*(x) := \min \left\{ V_j(x, \boldsymbol{u}) : \boldsymbol{u} \in \mathcal{U}_j(x) \right\}$$

Dynamic Programming

For any $x \in \mathcal{X}_j$ the optimal cost is

$$V_j^*(x) = \min_{u \in U} \{ L(x, u) + V_{j-1}^*(f(x, u)) : f(x, u) \in \mathcal{X}_{j-1} \}$$

And the optimal control is

$$\kappa_j(x) = \operatorname{argmin}_{u \in U} \{ L(x, u) + V_{j-1}^*(f(x, u)) : f(x, u) \in \mathcal{X}_{j-1} \}$$

3.3 Towards MPC stability

- We now assume that a controller exists which can force the system to perform well locally in X_f.

- The dynamic programming relations then imply that this good behaviour is inherited on all the nested sets.

Lemma 2. *Suppose that for any $x \in X_f$, there exits $u \in U$ such that $f(x,u) \in X_f$ and*

$$V_f(f(x,u)) - V_f(x) \leq -L(x,u). \tag{15}$$

Then, $V_1^*(x) \leq V_f(x)$ and

$$V_{j+1}^*(x) \leq V_j^*(x), \qquad x \in \mathcal{X}_j, j \geq 1.$$

Taking more steps from a feasible point cannot cost more!

3.3 MPC Stability

- The point of this section is to study stability of the closed-loop MPC dynamics

$$x_{k+1} = f(x_i, \kappa_N(x_k)) =: g(x_k)$$

- Assumptions:

1. There exist $\alpha_L, \alpha_f \in \mathcal{K}_\infty$ such that

$$\alpha_L(\|x\|) \leq L(x,u), \qquad x \in X, u \in U,$$

and

$$V_f(x) \leq \alpha_f(\|x\|), \qquad x \in X_f$$

2. 0 is in the interior of X_f, and for any $x \in X_f$ exists $u \in U$ such that $f(x,u) \in X_f$ and

$$V_f(f(x,u)) - V_f(x) \leq -L(x,u)$$

- The aim is to show that the optimal cost

$$V_N^*(\cdot) : \mathbb{R}^N \to \mathbb{R}$$

is a Lyapunov function for the closed-loop system

3.3 MPC Stability

Theorem 9. *Consider the closed-loop MPC system* $x_{k+1} = f(x_k, \kappa_N(x_k))$. *with the above assumptions. Then*

(i) There exist $\alpha_1, \alpha_2 \in \mathcal{K}_\infty$ *such that*

$$\alpha_1(\|x\|) \leq V_N^*(x) \leq \alpha_2(\|x\|), \qquad x \in \mathcal{X}_N;$$

(ii) For any $x \in \mathcal{X}_N$, *it follows that* $g(x) = f(x, \kappa_N(x)) \in \mathcal{X}_N$;

(iii) There exists $\alpha_3 \in \mathcal{K}_\infty$ *such that*

$$V_N^*(g(x)) - V_N^*(x) \leq -\alpha_3(\|x\|), \qquad x \in \mathcal{X}_N.$$

3.3 MPC Stability

Corollary 1. *Suppose that the assumptions of Theorem 9 hold. Let* $x \in \mathcal{X}_N$ *and let* $(x_i)_{i \geq 0}$ *be the closed-loop MPC trajectory* $x_{i+1} = f(x_i, \kappa_N(x_i))$. *Then* $x_i \to 0$ *as* $i \to \infty$.