06_Optimization-Based Control

1. Optimal Control of a Class of Hybrid Systems
Example: Optimal Control of Hybrid Manufacturing Systems
Optimality Condition
Dealing with Non-differentiable: Generalized Gradient
2. MPC for MLD and PWA Systems
Classical Settings of MPC
MPC for MLD Systems
3. MPC for MMPS and Continuous PWA Systems
Note of PWA and MMPS Systems
Canonical Forms of MMPS Functions
MPC for MMPS Systems
4. Game-Theoretic Approach
Summary

1. Optimal Control of a Class of Hybrid Systems

Example: Optimal Control of Hybrid Manufacturing Systems

Manufacturing System

Jobs move through network of work centers

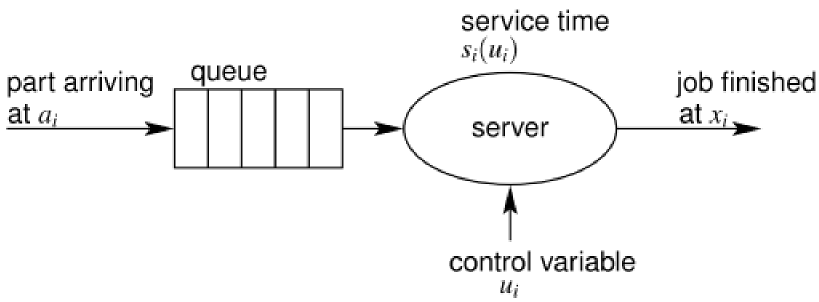
State

- **Temporal State (event-driven)**: waiting time, departure time, ...
- **Physical State (time-driven)**: temperature, size, weight, chemical composition

Trade-off:

- temporal requirements on job **completion times**
- physical requirements on **quality** of completed jobs

We always assume higher quality → longer processing times



- Single-stage, single-server queueing system
- N jobs (each job corresponds to mode)
- Buffer with capacity $> N$
- As job i is processed, physical state z_i evolves according to

$$\dot{z}_i = g_i(z_i, u_i, t) \qquad \text{with } z_i(\tau_i) = \zeta_i$$

with τ_i time instant at which processing begins

hs_opt_ctrl.4

- Control variable u_i is used to attain final desired physical state:
If $s_i(u_i)$ is *service time* and $\Gamma_i(u_i)$ is target quality set, then

$$s_i(u_i) = \min\{t \geqslant 0 \mid z_i(\tau_i + t) \in \Gamma_i(u_i)\}$$

- *Temporal state* x_i represents time when job is completed:
If a_i is arrival time of job i, then

$$x_i = \max(x_{i-1}, a_i) + s_i(u_i) \qquad \text{(Lindley equation)}$$

hs_opt_ctrl.5

Optimal control problem:

$$\min_{u_1,\dots,u_N} J = \sum_{i=1}^{N} L_i(x_i, u_i)$$

subject to evolution equations for z_i and x_i

where $L(x_i, u_i)$ is cost function associated with job i

$\rightarrow$ classical discrete-time optimal control problems except for

- i does not count time steps
  $\rightarrow$ not really an issue
- max is non-differentiable for $a_i = x_{i-1}$
  $\rightarrow$ prevents use of standard gradient-based techniques
  $\rightarrow$ use non-differentiable calculus, generalized gradient

Example: heating/annealing manufacturing processes

See Lecture Slides

Optimality Condition

Augmented Cost

$$\bar{J}(x, \lambda, u) = \sum_{i=1}^{N} \left(L_i\left(x_i, u_i\right) + \lambda_i\left(\max\left(x_{i-1}, a_i\right) + s_i\left(u_i\right) - x_i\right)\right)$$

where λ is co-state

- Assume costs L_i and s_i are continuously differentiable
- First-Order Necessary Conditions of Optimality

$$\frac{\partial \bar{J}}{\partial u_i} = 0, \quad \frac{\partial \bar{J}}{\partial \lambda_i} = 0, \quad \frac{\partial \bar{J}}{\partial x_i} = 0 \quad \text{for } i = 1, \ldots, N$$

  Then we have:

  - **Stationarity condition**: $\frac{\partial L_i(x_i,u_i)}{\partial u_i} + \lambda_i \frac{ds_i(u_i)}{du_i} = 0$
  - **Temporal state equation**: $x_i = \max\left(x_{i-1}, a_i\right) + s_i\left(u_i\right)$ with $x_0 = -\infty$
  - **Co-state equation**: $\lambda_i = \frac{\partial L_i(x_i,u_i)}{\partial x_i} + \lambda_{i+1} \frac{d \max(x_i,a_{i+1})}{dx_i}$ with final boundary condition $\lambda_N = \frac{\partial L_N(x_N,u_N)}{\partial x_N}$
  - However, $\max$ function most time are not differentiable

Dealing with Non-differentiable: Generalized Gradient

- $\max$ is **Lipschitz continuous + differentiable** except for $x_i = a_{i+1}$

$$\frac{d \max\left(x_i, a_{i+1}\right)}{dx_i} = \begin{cases} 0 & \text{if } x_i < a_{i+1} \\ 1 & \text{if } x_i > a_{i+1} \end{cases}$$

Generalized Gradient

Let $f : \mathbb{R}^n \to \mathbb{R}$ be **locally Lipschitz continuous**, and let $S(u)$ denote set of all sequences $\{u_m\}_{m=1}^{\infty}$ that satisfy

- $u_m \to u$ as $m \to \infty$
- gradient $\nabla f\left(u_m\right)$ exists for all m
- $\lim_{m\to\infty} \nabla f\left(u_m\right) = \phi$ exists

Then **generalized gradient** $\partial f(u)$ is defined as **convex hull** of all limits ϕ corresponding to some sequence $\{u_m\}_{m=1}^{\infty}$ in $S(u)$

Property

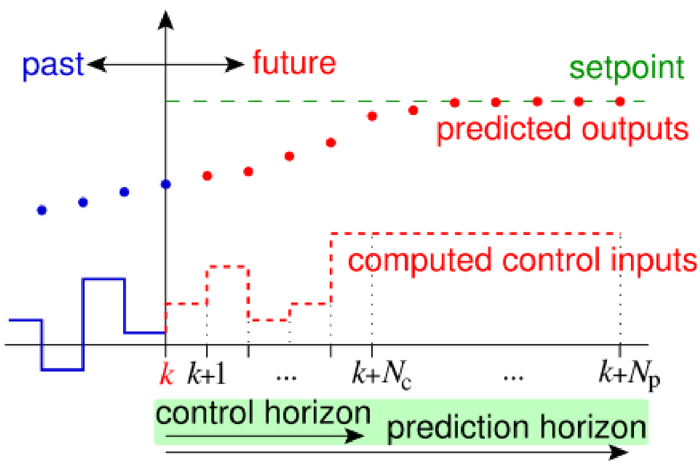
- if f is **continuously differentiable** in some open set containing u, then $\partial f(u) = \{\nabla f(u)\}$
- if u is **local minimum, then** $0 \in \partial f(u)$
  $\longrightarrow$ this becomes **first-order optimality condition in non-smooth optimization**

2. MPC for MLD and PWA Systems

Classical Settings of MPC

Classical MPC is quite familiar for me. Here I will not introduce MPC in detail, I will just mention some key-points.

One important thing is in this course, w have two **horizon: control horizon and prediction horizon**



Extra condition to reduce **computational complexity**: control horizon N_{c}

$$u(k+j) = u\left(k+N_{\mathrm{c}}-1\right) \quad \text{for } j = N_{\mathrm{c}}, \ldots, N_{\mathrm{p}}-1$$

$\longrightarrow$ smoother controller signal & stabilizing effect

MPC for MLD Systems

<u>System Model</u>

Consider MLD Systems

$$\begin{gathered}
x(k+1) = Ax(k) + B_1 u(k) + B_2 \delta(k) + B_3 z(k) \\
y(k) = Cx(k) + D_1 u(k) + D_2 \delta(k) + D_3 z(k) \\
E_1 x(k) + E_2 u(k) + E_3 \delta(k) + E_4 z(k) \leqslant g_5
\end{gathered}$$

- Consider **equilibrium state/input/output**: $(x_{\mathrm{eq}}, u_{\mathrm{eq}}, y_{\mathrm{eq}}) \rightarrow (\delta_{\mathrm{eq}}, z_{\mathrm{eq}})$
- $\hat{x}(k+j \mid k)$: estimate of x at sample step $k+j$ based on information available at sample step k

<u>MPC Model</u>

We tried to stabilize system to equilibrium state

$$
\begin{aligned}
J(k)=\sum_{j=1}^{N_{\mathrm{p}}} & \|\hat{x}(k+j \mid k)-x_{\mathrm{eq}}\|_{Qx}^2+\|u(k+j-1)-u_{\mathrm{eq}}\|_{Qu}^2+ \\
& \|\hat{y}(k+j \mid k)-y_{\mathrm{eq}}\|_{Qy}^2+\left\|\hat{\delta}(k+j-1 \mid k)-\delta_{\mathrm{eq}}\right\|_{Q\delta}^2+ \\
& \|\hat{z}(k+j-1 \mid k)-z_{\mathrm{eq}}\|_{Qz}^2
\end{aligned}
$$

with $Q. > 0$

- End-point condition: $\hat{x}\left(k+N_{\mathrm{p}} \mid k\right)=x_{\mathrm{eq}}$
- Control horizon constraint:

  $u(k+j)=u\left(k+N_{\mathrm{c}}-1\right)$ for $j=N_{\mathrm{c}}, \ldots, N_{\mathrm{p}}-1$

Property

If feasible solution exists for $x(0)$, then MPC input stabilizes system, i.e.,

$$
\begin{array}{lll}
\lim_{k \rightarrow \infty} x(k)=x_{\mathrm{eq}} & \lim_{k \rightarrow \infty}\|y(k)-y_{\mathrm{eq}}\|_{Q_y}=0 & \lim_{k \rightarrow \infty}\|z(k)-z_{\mathrm{eq}}\|_{Q_z}=0 \\
\lim_{k \rightarrow \infty} u(k)=u_{\mathrm{eq}} & \lim_{k \rightarrow \infty}\|\delta(k)-\delta_{\mathrm{eq}}\|_{Q_\delta}=0 &
\end{array}
$$

Algorithms for MLD-MPC: mixed-integer quadratic programming (MIQP)

- **Successive substitution** of system equations: → $\hat{x}(k+j|k)$ is linear function of $x(k), \hat{u}, \hat{\delta}$ and $\hat{z}$ Also holds for $y(k \hat{+} j|k)$
  - When substitution, we assume u_k is known, we are now proving u and x to compute J, which is necessary for optimization steps
- Define $\tilde{V}(k)=\left[\begin{array}{lll}\tilde{u}^{\top}(k) & \tilde{\delta}^{\top}(k) & \tilde{z}^{\top}(k)\end{array}\right]^{\top}$
- Results in an **MIQP** Problem

$$
\begin{aligned}
& \min_{\tilde{V}(k)} \tilde{V}^{\top}(k) S_1 \tilde{V}(k)+2\left(S_2+x^{\top}(k) S_3\right) \tilde{V}(k) \\
& \text{subject to } F_1 \tilde{V}(k) \leqslant F_2+F_3 x(k),
\end{aligned}
$$

3. MPC for MMPS and Continuous PWA Systems

Note of PWA and MMPS Systems

- **Continuous** PWA model can be used as **approximation** of general nonlinear continuous state space model
- **General** PWA systems are equivalent to **constrained** MMPS systems
- **Continuous** PWA functions and **unconstrained** MMPS functions coincide.

  i.e., a continuous PWA function f can be rewritten as

$$f = \max_j \min_i \left(\alpha_i^T x + \beta_i\right)$$

Canonical Forms of MMPS Functions

Any MMPS function $f : \mathbb{R}^n \to \mathbb{R}$ can be rewritten into **min-max canonical form**

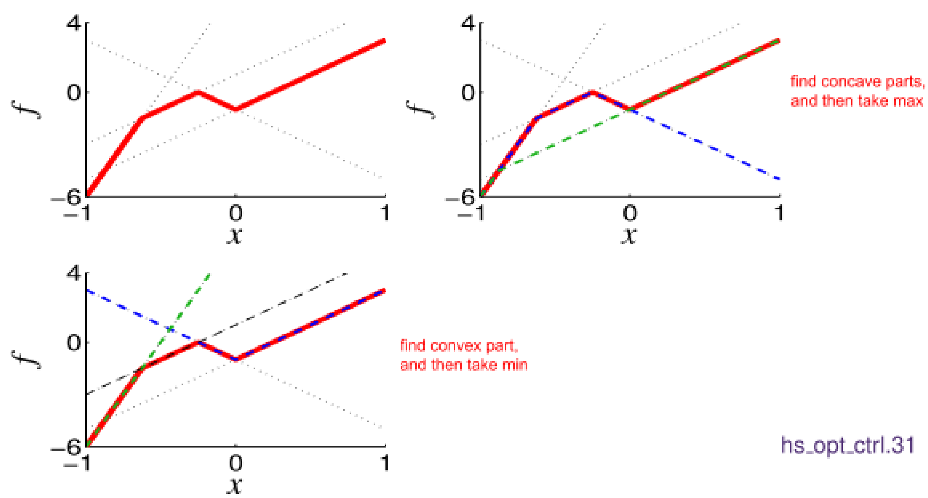
$$f = \min_i \max_j \left(\alpha_{(i,j)}^T x + \beta_{(i,j)}\right)$$

or into **max-min canonical form**

$$f = \max_i \min_j \left(\gamma_{(i,j)}^T x + \delta_{(i,j)}\right)$$

Example

$$\begin{aligned} f(x) &= \min(8x+6,1) - 2\max\left(\min(2x+1,1-2x),-2x\right) \\ &= \max\left(\min(12x+6,4x+1,-4x-1),\min(12x+6,4x-1)\right) \\ &= \min\left(\max(4x-1,-4x-1),12x+6,4x+1\right) \end{aligned}$$



<u>**Some Transformation Method: (From Lecture Notes)**</u>

- Max-Min and Min-Max
  - Minimization is distributive w.r.t. maximization, i.e., $\min(\alpha, \max(\beta,\gamma)) = \max(\min(\alpha,\beta),$ $\min(\alpha,\gamma))$, which results in:

$$\min(\max(\alpha,\beta),\max(\gamma,\delta)) = \max(\min(\alpha,\gamma),\min(\alpha,\delta),\min(\beta,\gamma),\min(\beta,\delta)).$$

  - The max operation is distributive w.r.t. min. Hence,

$$\max(\min(\alpha,\beta),\min(\gamma,\delta)) = \min(\max(\alpha,\gamma),\max(\alpha,\delta),\max(\beta,\gamma),\max(\beta,\delta)).$$

- We have

$$
\begin{aligned}
& \min (\alpha, \beta)+\min (\gamma, \delta)=\min (\alpha+\gamma, \alpha+\delta, \beta+\gamma, \beta+\delta) \\
& \max (\alpha, \beta)+\max (\gamma, \delta)=\max (\alpha+\gamma, \alpha+\delta, \beta+\gamma, \beta+\delta)
\end{aligned}
$$

- The min and max operators are related as follows:

$$
\max (\alpha, \beta)=-\min (-\alpha,-\beta)
$$

- If $\rho \in \mathbb{R}$ is positive, then

$$
\rho \max (\alpha, \beta)=\max (\rho \alpha, \rho \beta), \quad \rho \min (\alpha, \beta)=\min (\rho \alpha, \rho \beta)
$$

MPC for MMPS Systems

MMPS Model

$$
\begin{aligned}
& x(k)=\mathscr{M}_x(x(k-1), u(k)) \\
& y(k)=\mathscr{M}_y(x(k), u(k))
\end{aligned}
$$

- Prediction Horizon: N_p
- Estimate $\hat{y}(k+j \mid k)$ of output at sample step $k+j$:

$$
\hat{y}(k+j \mid k)=F_j(x(k-1), u(k), \ldots, u(k+j))
$$

$\rightarrow F_j$ **is MMPS function**! (if we expand the formulation with higher-order x, we can find the combination is still MMPS equation)

MPC Model

- Some possible **cost functions** (need to meet MMPS format)

$$
\begin{array}{ll}
J_{\text {out }, 1}(k)=\|\tilde{y}(k)-\tilde{r}(k)\|_1 & J_{\text {out }, \infty}(k)=\|\tilde{y}(k)-\tilde{r}(k)\|_{\infty} \\
J_{\text {in }, 1}(k)=\|\tilde{u}(k)\|_1 & J_{\text {in }, \infty}(k)=\|\tilde{u}(k)\|_{\infty}
\end{array}
$$

with

$$
\begin{aligned}
& \tilde{u}(k)=\left[\begin{array}{lll}
u^T(k) & \ldots & u^T(k+N \mathrm{p}-1)
\end{array}\right]^T \\
& \tilde{y}(k)=\left[\begin{array}{lll}
\hat{y}^{\top}(k \mid k) & \ldots & \hat{y}^{\top}\left(k+N_{\mathrm{p}}-1 \mid k\right)
\end{array}\right]^T \\
& \tilde{r}(k)=\left[\begin{array}{lll}
r^T(k) & \ldots & r^T\left(k+N_{\mathrm{p}}-1\right)
\end{array}\right]^T
\end{aligned}
$$

Note: $|x|=\max (x,-x) \rightarrow$ cost functions are MMPS functions

- **Constraints**

$$
C_{\mathrm{c}}(k, x(k-1), \tilde{u}(k), \tilde{y}(k)) \geqslant 0
$$

Algorithms for Model Solution

3.4 Algorithms for MMPS-MPC

- Nonlinear optimization (SQP, ELCP):
  $\rightarrow$ local minima, excessive computation time
- MPC for mixed logical-dynamical (MLD) systems [Bemporad, Morari]:
  $\rightarrow$ mixed real-integer quadratic programming problems
- New approach based on canonical forms:
  $\rightarrow$ collection of linear programming problems

We will introduce the LP method in detail, we are familiar with LP and we will see how to **transform MMPS MPC to an LP** problem

Recall: $J(k)$ is MMPS function

$$
\begin{aligned}
\Rightarrow J(k) &= \max_i \left(\min_j \left(\gamma_{(i,j)}^T \tilde{u} + \delta_{(i,j)} \right) \right) \\
&= \min_i \left(\max_j \left(\alpha_{(i,j)}^T \tilde{u} + \beta_{(i,j)} \right) \right) \\
\Rightarrow \min_{\tilde{u}} J(k) &= \min_{\tilde{u}} \min_i \left(\max_j \left(\alpha_{(i,j)}^T \tilde{u} + \beta_{(i,j)} \right) \right) \\
&= \min_i \underbrace{\min_{\tilde{u}} \left(\max_j \left(\alpha_{(i,j)}^T \tilde{u} + \beta_{(i,j)} \right) \right)}_{\rightarrow \text{LP!}}
\end{aligned}
$$

Which means we will have i **LP problem** because the min-max part is equal to:

LP i :

$$
\begin{aligned}
&\min_{\tilde{u}} t \\
&\text{s.t.} \begin{cases} t \geqslant \alpha_{(i,j)}^T \tilde{u} + \beta_{(i,j)} & \text{for all } j \\ P\tilde{u} + q \geqslant 0 \end{cases}
\end{aligned}
$$

4. Game-Theoretic Approach

We tried to use **Game-Theoretic Perspective** for **Safety-Critical Applications** such as collision avoidance in free flight or automated highways

- guarantee safety even in case intentions of other aircraft/vehicle are not known (**non-cooperative game**)
- if (partial) communication possible $\rightarrow$ **cooperative game**

Game-Theoretic Model

- **System Model:**

$$\dot{x} = f(x, u, d)$$

  - u control inputs (corresponding to 1st player)
  - d disturbance inputs (corresponding to 2nd player/adversary)

- **Safety Constraints** be represented by:

$$F = \{x \in X \mid S(x) \geqslant 0\}$$

- Cost Function
  - Let $t_0 \leqslant t_{\text{end}}$ and consider cost function

$$\begin{gathered} J : X \times \mathscr{U} \times \mathscr{D} \times [t_0, t_{\text{end}}] \rightarrow \mathbb{R} \\ i.e. \quad (x, u(\cdot), d(\cdot), t) \mapsto S\left(x\left(t_{\text{end}}\right)\right) \end{gathered}$$

    where $\mathscr{U}$ and $\mathscr{D}$ denote admissible control and disturbance functions

    J **is cost associated with trajectory starting at** x **at time** $t \in [t_0, t_{\text{end}}]$ **with inputs** $u(\cdot)$ **and** $d(\cdot)$**, and ending at time** $t = t_{\text{end}}$ **at the final state** $x\left(t_{\text{end}}\right)$

- **Optimization Target**

$$J^\star(x, t) = \max_{u \in \mathscr{U}} \min_{d \in \mathscr{D}} J(x, u, d, t)$$

Model Solution

- **Feasible Set(not officially)**

$$\left\{x \in X \mid \min_{\tau \in [t, t_{\text{end}}]} J^\star(x, \tau) \geqslant 0\right\}$$

  - contains all states for which system can be forced by control u to remain in safe set F for at least $|t_{\text{end}} - t|$ time units, irrespective 0 disturbance function d

- Value function $J^\star$ can be computed using **Hamilton-Jacobi equations**
  - Computation Tremendous Task
  - Provides systematic way to check safety properties for continuous-time system

Summary

5. Summary

- Optimal control of hybrid systems
- MPC for MLD and PWA systems
- MPC for MMPS and continuous PWA systems
- Game-theoretic approaches

- Optimal Control of Hybrid Manufacturing Systems
  - **temporal state (event-driven)**
  - physical state (time-driven)
  - event-driven part may cause **non-differentiable** when computing optimality condition of the cost function
- **Generalized Gradient** to deal with non-differentiable
- MPC for MLD and PWA
  - Control Horizon + Prediction Horizon
  - MLD: **mixed-integer quadratic programming (MIQP)**
  - PWA: transform to MLD
- MPC for MMPS and Continuous PWA
  - **Continuous PWA** $\leftrightarrow$ **Unconstrained MMPS**
  - Canonical form of MMPS function
  - MPC for MMPS: **canonical form + multiple LP problem**
- Gamte-Theoretical Approach
  - Safety-Critical
  - Hamilton-Jacobi equations