

AA203

Optimal and Learning-based Control

Direct methods for optimal control: direct collocation and SCP*

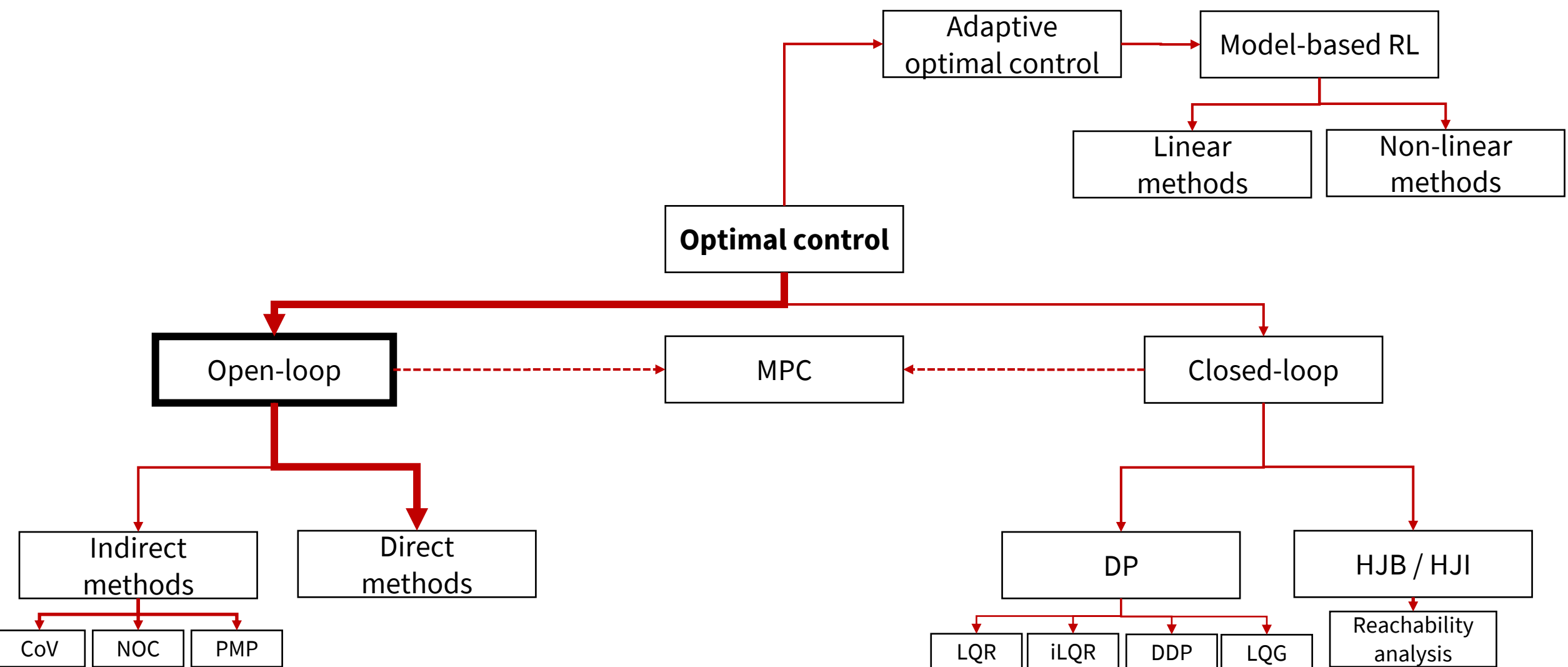


Stanford
University

*Slides prepared by Riccardo Bonalli,
rbonalli@stanford.edu



Roadmap



Optimal control problem

$$\min \int_0^{t_f} g(\mathbf{x}(t), \mathbf{u}(t), t) dt$$

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), t), \quad t \in [0, t_f]$$

(OCP)

$$\mathbf{x}(0) = \mathbf{x}_0,$$

$$\mathbf{x}(t_f) \in M_f = \{\mathbf{x} \in \mathbb{R}^n : F(\mathbf{x}) = 0\}$$

$$\mathbf{u}(t) \in U \subseteq \mathbb{R}^m, \quad t \in [0, t_f]$$

For simplicity:

- We assume the terminal cost h is equal to 0
- We assume $t_0 = 0$

- Direct Methods:

1. Transcribe (OCP) into a nonlinear, constrained optimization problem
2. Solve the optimization problem via nonlinear programming

- Specific Methods:

1. High Order Direct Methods, i.e., Direct Collocation Methods
2. Sequential Convex Programming (SCP)

Direct Collocation: High-Order Direct Method

Hermite-Simpson Method

- Select a discretization $0 = t_0 < t_1 < \dots < t_N = t_f$ for the interval $[0, t_f]$ and denote $h_i = t_{i+1} - t_i$
- In every subinterval $[t_i, t_{i+1}]$, approximate $\mathbf{x}(t)$ with the **cubic polynomial** $\mathbf{x}(t) = \mathbf{c}_0^i + \mathbf{c}_1^i(t - t_i) + \mathbf{c}_2^i(t - t_i)^2 + \mathbf{c}_3^i(t - t_i)^3$ so that its derivative is given by $\dot{\mathbf{x}}(t) = \mathbf{c}_1^i + 2\mathbf{c}_2^i(t - t_i) + 3\mathbf{c}_3^i(t - t_i)^2$
- By denoting $\mathbf{x}_i = \mathbf{x}(t_i)$, $\mathbf{x}_{i+1} = \mathbf{x}(t_{i+1})$, $\dot{\mathbf{x}}_i = \dot{\mathbf{x}}(t_i)$ and $\dot{\mathbf{x}}_{i+1} = \dot{\mathbf{x}}(t_{i+1})$, the relations above give us the coefficients

$$\begin{bmatrix} \mathbf{x}_i \\ \dot{\mathbf{x}}_i \\ \mathbf{x}_{i+1} \\ \dot{\mathbf{x}}_{i+1} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & h_i & h_i^2 & h_i^3 \\ 0 & 1 & 2h_i & 3h_i^2 \end{bmatrix} \begin{bmatrix} \mathbf{c}_0^i \\ \mathbf{c}_1^i \\ \mathbf{c}_2^i \\ \mathbf{c}_3^i \end{bmatrix} \quad \longrightarrow \quad \begin{bmatrix} \mathbf{c}_0^i \\ \mathbf{c}_1^i \\ \mathbf{c}_2^i \\ \mathbf{c}_3^i \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -\frac{3}{h_i^2} & -\frac{2}{h_i} & \frac{3}{h_i^2} & -\frac{1}{h_i} \\ \frac{2}{h_i^2} & \frac{1}{h_i^2} & -\frac{2}{h_i^3} & \frac{1}{h_i^2} \end{bmatrix} \begin{bmatrix} \mathbf{x}_i \\ \dot{\mathbf{x}}_i \\ \mathbf{x}_{i+1} \\ \dot{\mathbf{x}}_{i+1} \end{bmatrix}$$

- Choose intermediate times $t_i^c = t_i + \frac{h_i}{2}$, i.e., **collocation points**, and define interpolated controls $\mathbf{u}_i^c = \frac{\mathbf{u}_i + \mathbf{u}_{i+1}}{2}$. From above:

$$\mathbf{x}_i^c := \mathbf{x}\left(t_i + \frac{h_i}{2}\right) = \frac{1}{2}(\mathbf{x}_i + \mathbf{x}_{i+1}) + \frac{h_i}{8}(\mathbf{f}(\mathbf{x}_i, \mathbf{u}_i, t_i) - \mathbf{f}(\mathbf{x}_{i+1}, \mathbf{u}_{i+1}, t_{i+1}))$$

$$\dot{\mathbf{x}}_i^c := \dot{\mathbf{x}}\left(t_i + \frac{h_i}{2}\right) = -\frac{3}{2h_i}(\mathbf{x}_i - \mathbf{x}_{i+1}) - \frac{1}{4}(\mathbf{f}(\mathbf{x}_i, \mathbf{u}_i, t_i) + \mathbf{f}(\mathbf{x}_{i+1}, \mathbf{u}_{i+1}, t_{i+1}))$$

Direct Collocation: High-Order Direct Method

Hermite-Simpson Method

(OCP)

$$\min \int_0^{t_f} g(\mathbf{x}(t), \mathbf{u}(t), t) dt$$

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), t), t \in [0, t_f]$$

$$\mathbf{x}(0) = \mathbf{x}_0,$$

$$\mathbf{x}(t_f) \in M_f = \{\mathbf{x} \in \mathbb{R}^n : F(\mathbf{x}) = 0\}$$

$$\mathbf{u}(t) \in U \subseteq \mathbb{R}^m, t \in [0, t_f]$$



(NLOP)

$$\min_{(\mathbf{x}_i, \mathbf{u}_i)} \sum_{i=0}^{N-1} h_i g(t_i, \mathbf{x}_i, \mathbf{u}_i)$$

$$\mathbf{u}_i \in U, i = 0, \dots, N - 1, \quad F(\mathbf{x}_N) = 0$$

$$\dot{\mathbf{x}}_i^c - \mathbf{f}(t_i^c, \mathbf{x}_i^c, \mathbf{u}_i^c) = 0, i = 0, \dots, N - 1$$

Where, from the previous computations:

$$\dot{\mathbf{x}}_i^c - \mathbf{f}(t_i^c, \mathbf{x}_i^c, \mathbf{u}_i^c) = \mathbf{x}_i - \mathbf{x}_{i+1} + \frac{h_i}{6} (\mathbf{f}(t_i, \mathbf{x}_i, \mathbf{u}_i) + 4\mathbf{f}(t_i^c, \mathbf{x}_i^c, \mathbf{u}_i^c) + \mathbf{f}(t_{i+1}, \mathbf{x}_{i+1}, \mathbf{u}_{i+1}))$$

Implicit constraints that improve robustness!

Practical Example

Designing Hermite-Simpson Method in Matlab

Solve the following optimal control problem via the Hermite-Simpson method, using the Matlab function “**fmincon**”

Modified Zermelo Problem

(OCP)

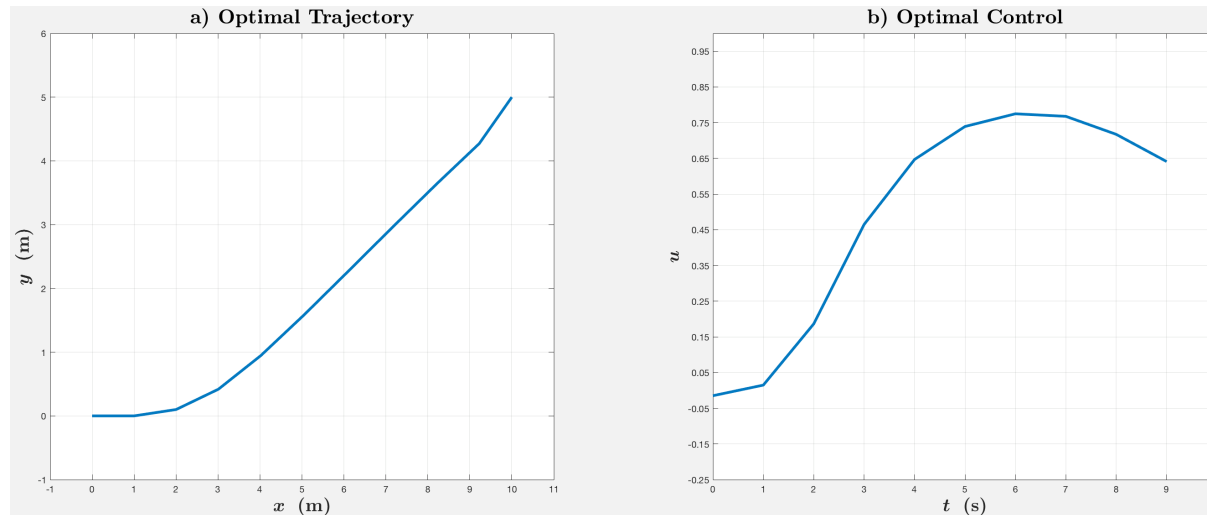
$$\min \int_0^{t_f} u(t)^2 dt$$
$$\dot{x}(t) = v \cos(u(t)) + \text{flow}(y(t)), t \in [0, t_f]$$
$$\dot{y}(t) = v \sin(u(t)), t \in [0, t_f]$$
$$(x, y)(0) = 0, (x, y)(t_f) = (M, \ell)$$
$$|u(t)| \leq u_{\max}, t \in [0, t_f]$$



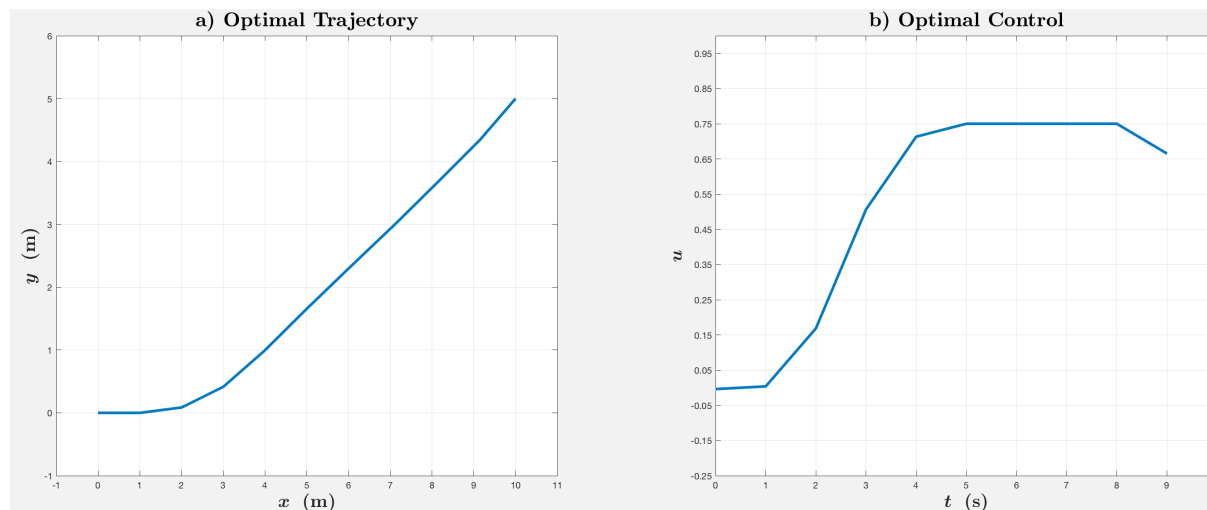
(NLOP)

$$\min_{(x_i, y_i, u_i)} \sum_{i=0}^{N-1} u_i^2$$
$$|u_i| \leq u_{\max}, (x_0, y_0) = 0, (x_N, y_N) = (M, \ell)$$
$$\dot{x}_i^c - v \cos(u_i^c) - \text{flow}(y_i^c) = 0$$
$$\dot{y}_i^c - v \sin(u_i^c) = 0$$

Practical Example



$|u(t)| \leq 1$
N = 10 (20)
12 iterations (28)



$|u(t)| \leq 0.75$
N = 10 (20)
12 iterations (23)

Sequential Convex Programming

The sources of nonconvexities are the dynamics and (possibly) the cost. **Idea: linearizing them around nominal curves!**

$$\min \int_0^{t_f} g(\mathbf{x}(t), \mathbf{u}(t), t) dt$$

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), t), \quad t \in [0, t_f]$$

(OCP)

$$\mathbf{x}(0) = \mathbf{x}_0, \quad \mathbf{x}(t_f) = \mathbf{x}_f$$

$$\mathbf{u}(t) \in U \subseteq \mathbb{R}^m, \quad t \in [0, t_f]$$

1. Assume that g is convex. Let $(\mathbf{x}_0(\cdot), \mathbf{u}_0(\cdot))$ be a nominal tuple of trajectory and control. **$(\mathbf{x}_0(\cdot), \mathbf{u}_0(\cdot))$ does not need to be feasible!**

Sequential Convex Programming

The sources of nonconvexities are the dynamics and (possibly) the cost. **Idea: linearizing them around nominal curves!**

$$\min \int_0^{t_f} g(\mathbf{x}(t), \mathbf{u}(t), t) dt$$

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), t), \quad t \in [0, t_f]$$

(OCP)

$$\mathbf{x}(0) = \mathbf{x}_0, \quad \mathbf{x}(t_f) = \mathbf{x}_f$$

$$\mathbf{u}(t) \in U \subseteq \mathbb{R}^m, \quad t \in [0, t_f]$$

1. Assume that g is convex. Let $(\mathbf{x}_0(\cdot), \mathbf{u}_0(\cdot))$ be a nominal tuple of trajectory and control. **$(\mathbf{x}_0(\cdot), \mathbf{u}_0(\cdot))$ does not need to be feasible!**

2. Linearize \mathbf{f} around $(\mathbf{x}_0(\cdot), \mathbf{u}_0(\cdot))$:

$$\begin{aligned} \mathbf{f}_1(\mathbf{x}, \mathbf{u}, t) \\ = \mathbf{f}(\mathbf{x}_0(t), \mathbf{u}_0(t), t) + \frac{\partial \mathbf{f}}{\partial \mathbf{x}}(\mathbf{x}_0(t), \mathbf{u}_0(t), t)(\mathbf{x} - \mathbf{x}_0(t)) + \frac{\partial \mathbf{f}}{\partial \mathbf{u}}(\mathbf{x}_0(t), \mathbf{u}_0(t), t)(\mathbf{u} - \mathbf{u}_0(t)) \end{aligned}$$

Sequential Convex Programming

The sources of nonconvexities are the dynamics and (possibly) the cost. **Idea: linearizing them around nominal curves!**

$$\min \int_0^{t_f} g(\mathbf{x}(t), \mathbf{u}(t), t) dt$$

$$\dot{\mathbf{x}}(t) = \mathbf{f}_1(\mathbf{x}(t), \mathbf{u}(t), t), \quad t \in [0, t_f]$$

(LOCP)₁

$$\mathbf{x}(0) = \mathbf{x}_0, \quad \mathbf{x}(t_f) = \mathbf{x}_f$$

$$\mathbf{u}(t) \in U \subseteq \mathbb{R}^m, \quad t \in [0, t_f]$$

1. Assume that g is convex. Let $(\mathbf{x}_0(\cdot), \mathbf{u}_0(\cdot))$ be a nominal tuple of trajectory and control. **$(\mathbf{x}_0(\cdot), \mathbf{u}_0(\cdot))$ does not need to be feasible!**
2. Linearize \mathbf{f} around $(\mathbf{x}_0(\cdot), \mathbf{u}_0(\cdot))$:
$$\begin{aligned} \mathbf{f}_1(\mathbf{x}, \mathbf{u}, t) &= \mathbf{f}(\mathbf{x}_0(t), \mathbf{u}_0(t), t) + \frac{\partial \mathbf{f}}{\partial \mathbf{x}}(\mathbf{x}_0(t), \mathbf{u}_0(t), t)(\mathbf{x} - \mathbf{x}_0(t)) + \frac{\partial \mathbf{f}}{\partial \mathbf{u}}(\mathbf{x}_0(t), \mathbf{u}_0(t), t)(\mathbf{u} - \mathbf{u}_0(t)) \end{aligned}$$
3. Solve the new **problem (LOCP)₁** for $(\mathbf{x}_1(\cdot), \mathbf{u}_1(\cdot))$

Sequential Convex Programming

The sources of nonconvexities are the dynamics and (possibly) the cost. **Idea: linearizing them around nominal curves!**

$$\min \int_0^{t_f} g(\mathbf{x}(t), \mathbf{u}(t), t) dt$$

$$\dot{\mathbf{x}}(t) = \mathbf{f}_{k+1}(\mathbf{x}(t), \mathbf{u}(t), t), \quad t \in [0, t_f]$$

(LOCP)_{k+1}

$$\mathbf{x}(0) = \mathbf{x}_0, \quad \mathbf{x}(t_f) = \mathbf{x}_f$$

$$\mathbf{u}(t) \in U \subseteq \mathbb{R}^m, \quad t \in [0, t_f]$$

\mathbf{f}_{k+1} is linear in \mathbf{x} and \mathbf{u} !

1. Assume that g is convex. Let $(\mathbf{x}_0(\cdot), \mathbf{u}_0(\cdot))$ be a nominal tuple of trajectory and control. **$(\mathbf{x}_0(\cdot), \mathbf{u}_0(\cdot))$ does not need to be feasible!**

2. Linearize \mathbf{f} around $(\mathbf{x}_0(\cdot), \mathbf{u}_0(\cdot))$:

$$\begin{aligned} \mathbf{f}_1(\mathbf{x}, \mathbf{u}, t) \\ = \mathbf{f}(\mathbf{x}_0(t), \mathbf{u}_0(t), t) + \frac{\partial \mathbf{f}}{\partial \mathbf{x}}(\mathbf{x}_0(t), \mathbf{u}_0(t), t)(\mathbf{x} - \mathbf{x}_0(t)) + \frac{\partial \mathbf{f}}{\partial \mathbf{u}}(\mathbf{x}_0(t), \mathbf{u}_0(t), t)(\mathbf{u} - \mathbf{u}_0(t)) \end{aligned}$$

3. Solve the new **problem (LOCP)₁** for $(\mathbf{x}_1(\cdot), \mathbf{u}_1(\cdot))$

4. Iterate this procedure until convergence is achieved: linearize \mathbf{f} around the solution $(\mathbf{x}_k(\cdot), \mathbf{u}_k(\cdot))$ at iteration k :

$$\begin{aligned} \mathbf{f}_{k+1}(\mathbf{x}, \mathbf{u}, t) \\ = \mathbf{f}(\mathbf{x}_k(t), \mathbf{u}_k(t), t) + \frac{\partial \mathbf{f}}{\partial \mathbf{x}}(\mathbf{x}_k(t), \mathbf{u}_k(t), t)(\mathbf{x} - \mathbf{x}_k(t)) + \frac{\partial \mathbf{f}}{\partial \mathbf{u}}(\mathbf{x}_k(t), \mathbf{u}_k(t), t)(\mathbf{u} - \mathbf{u}_k(t)) \end{aligned}$$

and solve the **problem (LOCP)_{k+1}** for $(\mathbf{x}_{k+1}(\cdot), \mathbf{u}_{k+1}(\cdot))$

Sequential Convex Programming

Discretize and Solve a Convex Problem at Each Iteration

$$\min \int_0^{t_f} g(\mathbf{x}(t), \mathbf{u}(t), t) dt$$

$$\dot{\mathbf{x}}(t) = \mathbf{f}_{k+1}(\mathbf{x}(t), \mathbf{u}(t), t), t \in [0, t_f]$$

(LOCP)_{k+1}

$$\mathbf{x}(0) = \mathbf{x}_0, \quad \mathbf{x}(t_f) = \mathbf{x}_f$$

$$\mathbf{u}(t) \in U \subseteq \mathbb{R}^m, \quad t \in [0, t_f]$$

f_{k+1} is linear in x and u !

1. Select a discretization $0 = t_0 < t_1 < \dots < t_N = t_f$ for the interval $[0, t_f]$ and, for every $i = 0, \dots, N - 1$, define $\mathbf{x}_{i+1} \sim \mathbf{x}(t)$, $\mathbf{u}_i \sim \mathbf{u}(t)$, $t \in (t_i, t_{i+1}]$ and $\mathbf{x}_0 \sim \mathbf{x}(0)$
2. By denoting $h_i = t_{i+1} - t_i$, **(LOCP)**_{k+1} is transcribed into the following **convex optimization problem**

(DLOCP)_{k+1}

$$\min_{(\mathbf{x}_i, \mathbf{u}_i)} \sum_{i=0}^{N-1} h_i g(\mathbf{x}_i, \mathbf{u}_i, t_i)$$

$$\mathbf{x}_{i+1} = \mathbf{x}_i + h_i \mathbf{f}_{k+1}(\mathbf{x}_i, \mathbf{u}_i, t_i), i = 0, \dots, N - 1$$

$$\mathbf{u}_i \in U, i = 0, \dots, N - 1, \quad \mathbf{x}_N = \mathbf{x}_f$$

Sequential Convex Programming

Methodological Scheme for SCP

At each iteration k :

Linearize \mathbf{f} around the solution $(\mathbf{x}_k(\cdot), \mathbf{u}_k(\cdot))$

Define the continuous time problem $(\mathbf{LOCP})_{k+1}$

Solve $(\mathbf{DLOCP})_{k+1}$ via convex programming for a discretized version of $(\mathbf{x}_{k+1}(\cdot), \mathbf{u}_{k+1}(\cdot))$

Discretize $(\mathbf{LOCP})_{k+1}$ in time and define the convex optimization problem $(\mathbf{DLOCP})_{k+1}$

$$\min \int_0^{t_f} g(\mathbf{x}(t), \mathbf{u}(t), t) dt$$

$$\dot{\mathbf{x}}(t) = \mathbf{f}_{k+1}(\mathbf{x}(t), \mathbf{u}(t), t), t \in [0, t_f]$$

$(\mathbf{LOCP})_{k+1}$

$$\mathbf{x}(0) = \mathbf{x}_0, \quad \mathbf{x}(t_f) = \mathbf{x}_f$$

$$\mathbf{u}(t) \in U \subseteq \mathbb{R}^m, \quad t \in [0, t_f]$$

\mathbf{f}_{k+1} is linear in \mathbf{x} and \mathbf{u} !

$(\mathbf{DLOCP})_{k+1}$

$$\min_{(\mathbf{x}_i, \mathbf{u}_i)} \sum_{i=0}^{N-1} h_i g(\mathbf{x}_i, \mathbf{u}_i, t_i)$$

$$\mathbf{x}_{i+1} = \mathbf{x}_i + h_i \mathbf{f}_{k+1}(\mathbf{x}_i, \mathbf{u}_i, t_i), i = 0, \dots, N - 1$$

$$\mathbf{u}_i \in U, i = 0, \dots, N - 1, \quad \mathbf{x}_N = \mathbf{x}_f$$

Practical Example

Designing SCP Method in Matlab

Solve the following optimal control problem via SCP, using the Matlab framework “**Cvx**”

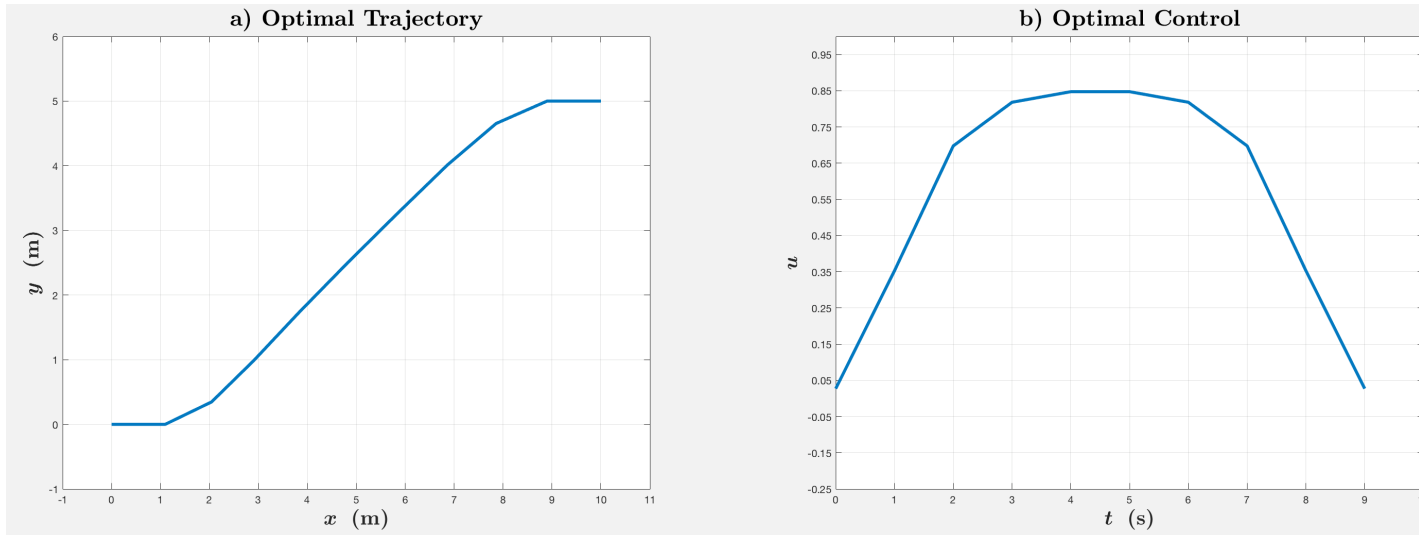
Modified Zermelo Problem

$$\begin{aligned} \text{(OCP)} \quad & \min \int_0^{t_f} u(t)^2 dt \\ & \dot{x}(t) = v \cos(u(t)) + \text{flow}(y(t)), t \in [0, t_f] \\ & \dot{y}(t) = v \sin(u(t)), t \in [0, t_f] \\ & (x, y)(0) = 0, (x, y)(t_f) = (M, \ell) \\ & |u(t)| \leq u_{\max}, t \in [0, t_f] \end{aligned}$$

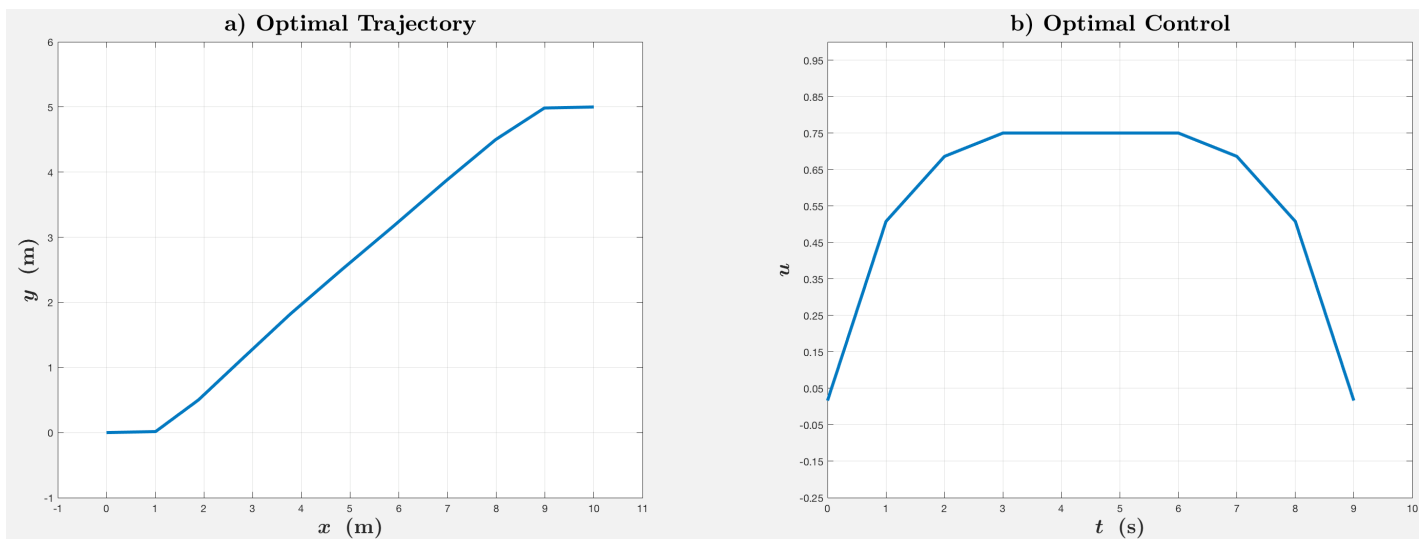


$$\begin{aligned} \text{(DLOCP)}_{k+1} \quad & \min_{(x_i, y_i, u_i)} \sum_{i=0}^{N-1} u_i^2 \\ & |u_i| \leq u_{\max}, (x_0, y_0) = 0, (x_N, y_N) = (M, \ell) \\ & \begin{pmatrix} x_{i+1} \\ y_{i+1} \end{pmatrix} = \begin{pmatrix} x_i \\ y_i \end{pmatrix} + h_i \mathbf{f}_{k+1}(x_i, y_i, u_i, t_i) \\ & \frac{\partial \mathbf{f}}{\partial (x, y)}(x, y, u) = \begin{pmatrix} 0 & \frac{\partial \text{flow}}{\partial y}(y) \\ 0 & 0 \end{pmatrix}, \quad \frac{\partial \mathbf{f}}{\partial u}(x, y, u) = \begin{pmatrix} -v \sin u \\ v \cos u \end{pmatrix} \end{aligned}$$

Practical Example



$|u(t)| \leq 1$
 $N = 10$
2 SCP iterations (12)



$|u(t)| \leq 0.75$
 $N = 10$
3 SCP iterations (12)

Next time

- MPC