AA 203
Optimal and Learning-Based Control
Direct methods

Autonomous Systems Laboratory

Stanford University

April 12, 2023
(last updated April 12, 2023)

Stanford
University

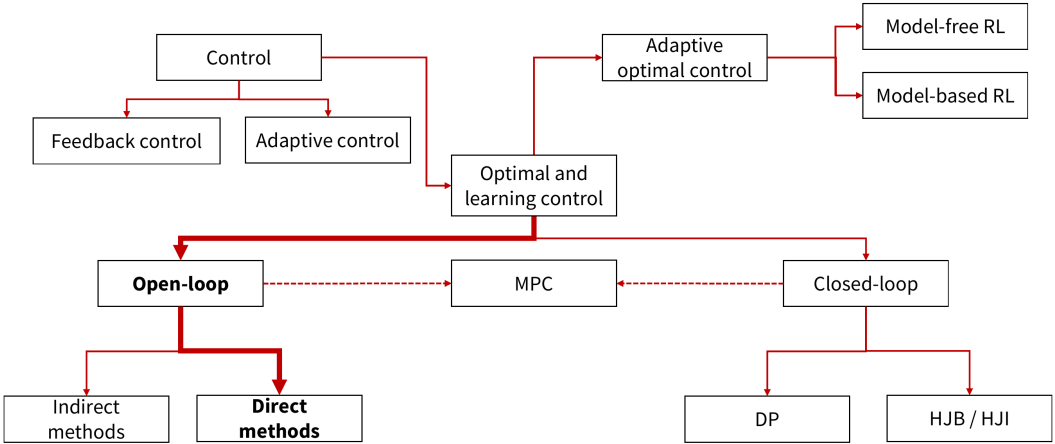
Agenda

1. Indirect versus direct methods
2. Direct multiple shooting
3. Direct collocation
4. Sequential convex programming
5. Direct methods in practice

Agenda

1. Indirect versus direct methods
2. Direct multiple shooting
3. Direct collocation
4. Sequential convex programming
5. Direct methods in practice

Course overview



Optimal control problem (continuous-time)

Consider the continuous-time optimal control problem (OCP)

$$
\begin{aligned}
\underset{x,u}{\text{minimize}} \quad & \ell_T(x(T)) + \int_0^T \ell(t, x(t), u(t))\, dt && \text{cost (terminal + stage)} \\
\text{subject to} \quad & \dot{x}(t) = f(t, x(t), u(t)),\ \forall t \in [0, T] && \text{dynamical feasibility} \\
& x(0) = x_0 && \text{initial condition} \\
& x(T) \in \mathcal{X}_T && \text{terminal condition} \\
& u(t) \in \mathcal{U},\ \forall t \in [0, T] && \text{input constraints}
\end{aligned}
$$

An optimal control $u^*(t)$ for a specific initial state x_0 is an *open-loop* input.

An optimal control of the form $u^*(t) = \pi^*(t, x(t))$ is a *closed-loop* input.

Review: Indirect methods

Use the PMP (i.e., first-order NOCs) to construct a BVP of the form

$$\begin{pmatrix} \dot{x}^* \\ \dot{p}^* \end{pmatrix} = \begin{pmatrix} f(t, x^*, u^*(t)) \\ -\nabla_x H_\eta(t, x^*, u^*(t), p^*) \end{pmatrix} \qquad x^*(0) = x_0$$

$$u^*(t) = \arg\max_{u \in \mathcal{U}} H_\eta(t, x^*(t), u^*(t), p^*(t)) \qquad h(T, x^*(T), p^*(T)) = 0$$

where $h(T, x^*(T), p^*(T)) \in \mathbb{R}^n$ if T is fixed, or $h(T, x^*(T), p^*(T)) \in \mathbb{R}^{n+1}$ if T is free.

The boundary conditions are determined by $x^*(0) = x_0$, $x^*(T) \in \mathcal{X}_T$, the transversality condition, and the boundary condition on the Hamiltonian (for free final time).

In *indirect single shooting*, we guess $p^*(0)$ and possibly T, solve the resulting IVP, then apply a root-finding method to the boundary condition $h(T, x^*(T), p^*(T)) = 0$.

In *indirect multiple shooting*, we "shoot" from many different times $\{t_k\}_{k=0}^{N-1} \subset [0, T)$ with $t_0 = 0$ to form a sparse system of equations, then apply a root-finding method.

Direct methods

In *direct methods*, we *transcribe* the OCP into a *finite-dimensional* nonlinear optimization problem that we then solve directly with nonlinear programming.

$$\left.\begin{aligned} \underset{x,u}{\text{minimize}} \;& \ell_T(x(T)) + \int_0^T \ell(t, x(t), u(t))\, dt \\ \text{subject to} \;& \dot{x}(t) = f(t, x(t), u(t)), \;\; \forall t \in [0, T] \\ & x(0) = x_0 \\ & x(T) \in \mathcal{X}_T \\ & u(t) \in \mathcal{U}, \;\; \forall t \in [0, T] \end{aligned}\right\} \implies \left\{\begin{aligned} \underset{z \in \mathcal{S} \subseteq \mathbb{R}^d}{\text{minimize}} \;& f(z) \\ \text{subject to} \;& h(z) = 0 \\ & g(z) \preceq 0 \end{aligned}\right.$$

Each direct method uses some manner of *transcription*, which also determines what z, h, g, and $\mathcal{S}$ represent.

Agenda

1. Indirect versus direct methods
2. Direct multiple shooting
3. Direct collocation
4. Sequential convex programming
5. Direct methods in practice

Direct multiple shooting via zero-order hold control

Construct u as *piecewise constant* over $t \in [0, T]$ with N intervals, i.e.,

$$u(t) = u(t_k), \ \forall t \in [t_k, t_{k+1}), \forall k \in \{0, 1, \ldots, N-1\},$$

with $t_0 = 0$ and $t_N = T$. This is known as a *zero-order hold* for u, as it is piecewise-$\mathcal{C}^0$.

Approximate the true discrete-time dynamics, i.e., enforce

$$x(t_{k+1}) = \underbrace{x(t_k) + \int_{t_k}^{t_{k+1}} f(t, x(t), u_k)\, dt}_{\text{“solve this IVP numerically”}} \approx f_{\mathsf{d}}(t_k, t_{k+1}, x(t_k), u(t_k))$$

For example, use Euler integration, i.e.,

$$f_{\mathsf{d}}(t_k, t_{k+1}, x(t_k), u(t_k)) \coloneqq x(t_k) + (t_{k+1} - t_k) f(t_k, x(t_k), u(t_k)),$$

or a Runge-Kutta scheme.

Direct multiple shooting via zero-order hold control

The resulting discretized OCP is

$$
\begin{aligned}
\underset{x,u}{\text{minimize}} \;\; & \ell_T(x(t_N)) + \sum_{k=0}^{N-1} (t_{k+1} - t_k)\ell(t_k, x(t_k), u(t_k)) \\
\text{subject to} \;\; & x(t_{k+1}) = f_{\mathsf{d}}(t_k, t_{k+1}, x(t_k), u(t_k)), \;\; \forall k \in \{0, 1, \ldots, N-1\} \\
& x(t_0) = x_0 \\
& x(t_N) \in \mathcal{X}_T \\
& u(t_k) \in \mathcal{U}, \;\; \forall k \in \{0, 1, \ldots, N-1\}
\end{aligned}
$$

This is a nonlinear program with decision variable

$$
z \coloneqq \left(\{x(t_k)\}_{k=0}^{N}, \, \{u(t_k)\}_{k=0}^{N-1} \right) \in \mathbb{R}^{Nn+(N-1)m}.
$$

Direct multiple shooting via zero-order hold control (time-optimal)

For time-optimal problems, introduce $T \geq 0$ as a variable and define

$$t_k = \frac{k}{N} T, \ \forall k \in \{0, 1, \ldots, N\}.$$

The resulting discretized time-optimal OCP is then

$$\begin{aligned}
\underset{x,u,T \geq 0}{\text{minimize}} \ & \ell_T(x(t_N)) + \sum_{k=0}^{N-1} \frac{T}{N} \ell(t_k, x(t_k), u(t_k)) \\
\text{subject to} \ & x(t_{k+1}) = f_{\mathsf{d}}(t_k, t_{k+1}, x(t_k), u(t_k)), \ \forall k \in \{0, 1, \ldots, N-1\} \\
& x(t_0) = x_0 \\
& x(t_N) \in \mathcal{X}_T \\
& u(t_k) \in \mathcal{U}, \ \forall k \in \{0, 1, \ldots, N-1\}
\end{aligned}$$

This is a nonlinear program with decision variable

$$z := \left(\{x(t_k)\}_{k=0}^{N}, \{u(t_k)\}_{k=0}^{N-1}, T \right) \in \mathbb{R}^{Nn + (N-1)m + 1}.$$

Limitations of direct multiple shooting

While the transcription in direct multiple shooting is straightforward,

- we do not have a parameterization of $x(t)$ for all $t \in [0, T]$,
- we are limited to piecewise-constant u, and
- we rely on explicit numerical differentiation of the dynamics (which might compose f multiple times in higher-order schemes).

The last point can cause computational bottlenecks when nonlinear solvers must compute gradients of terms in the transcribed problem. For example, a fourth-order Runge-Kutta scheme (i.e., RK4) would compose f with itself 3 times.

Agenda

1. Indirect versus direct methods
2. Direct multiple shooting
3. Direct collocation
4. Sequential convex programming
5. Direct methods in practice

Direct collocation (Hermite-Simpson)

Suppose we construct $x(t)$ to be a *continuous cubic spline*, i.e.,

$$x(t) = c_{k0} + \tau_k(t)c_{k1} + \tau_k(t)^2 c_{k2} + \tau_k(t)^3 c_{k3}, \ \forall t \in [t_k, t_{k+1}].$$

for each $k \in \{0, 1, \ldots, N-1\}$, where $c_{kd} \in \mathbb{R}^n$ for $d \in \{0, 1, 2, 3\}$ and

$$\tau_k(t) \coloneqq \frac{t - t_k}{\Delta t_k} \in [0, 1], \quad \Delta t_k \coloneqq t_{k+1} - t_k > 0.$$

Then

$$\dot{x}(t) = \frac{1}{\Delta t_k} c_{k1} + \frac{2}{\Delta t_k} \tau_k(t) c_{k2} + \frac{3}{\Delta t_k} \tau_k(t)^2 c_{k3}, \ \forall t \in [t_k, t_{k+1}].$$

Substituting in $t = t_k$ and $t = t_{k+1}$ for $x(t)$ and $\dot{x}(t)$ yields the linear system

$$\begin{pmatrix} x(t_k) \\ \dot{x}(t_k) \\ x(t_{k+1}) \\ \dot{x}(t_{k+1}) \end{pmatrix} = \begin{bmatrix} I & 0 & 0 & 0 \\ 0 & \Delta t_k^{-1} I & 0 & 0 \\ I & I & I & I \\ 0 & \Delta t_k^{-1} I & 2\Delta t_k^{-1} I & 3\Delta t_k^{-1} I \end{bmatrix} \begin{pmatrix} c_{k0} \\ c_{k1} \\ c_{k2} \\ c_{k3} \end{pmatrix}$$

Direct collocation (Hermite-Simpson)

Substituting in $t = t_k$ and $t = t_{k+1}$ for $x(t)$ and $\dot{x}(t)$ yields the linear system

$$\begin{pmatrix} x(t_k) \\ \dot{x}(t_k) \\ x(t_{k+1}) \\ \dot{x}(t_{k+1}) \end{pmatrix} = \begin{bmatrix} I & 0 & 0 & 0 \\ 0 & \Delta t_k^{-1} I & 0 & 0 \\ I & I & I & I \\ 0 & \Delta t_k^{-1} I & 2\Delta t_k^{-1} I & 3\Delta t_k^{-1} I \end{bmatrix} \begin{pmatrix} c_{k0} \\ c_{k1} \\ c_{k2} \\ c_{k3} \end{pmatrix}$$

Solving this system and using $\dot{x} = f(t, x, u)$ gives us

$$\begin{pmatrix} c_{k0} \\ c_{k1} \\ c_{k2} \\ c_{k3} \end{pmatrix} = \begin{bmatrix} I & 0 & 0 & 0 \\ 0 & \Delta t_k I & 0 & 0 \\ -3I & -2\Delta t_k I & 3I & -\Delta t_k I \\ 2I & \Delta t_k I & -2I & \Delta t_k I \end{bmatrix} \begin{pmatrix} x(t_k) \\ f(t_k, x(t_k), u(t_k)) \\ x(t_{k+1}) \\ f(t_{k+1}, x(t_{k+1}), u(t_{k+1})) \end{pmatrix}$$

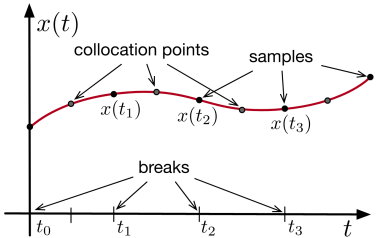
So if we determine $(x(t_k), u(t_k))$ for each $k \in \{0, 1, \ldots, N\}$, then we can solve for the spline coefficients to get an explicit parameterization of $x(t)$ over $t \in [0, T]$.

Direct collocation (Hermite-Simpson)

We have not yet enforced the dynamics constraints

$$x(t) = x(t_k) + \int_{t_k}^{t} f(s, x(s), u(s))\, ds,\ \forall t \in [t_k, t_{k+1}).$$

It is infeasible to enforce this at every time along each spline segment. We choose to enforce this constraint at only a single time $\bar{t}_k \in [t_k, t_{k+1})$ for each segment. Altogether, $\{\bar{t}_k\}_{k=0}^{N-1}$ are called *collocation points*.



Specifically, we choose the segment midpoints $\bar{t}_k := \frac{1}{2}(t_k + t_{k+1})$, which gives us

$$x(\bar{t}_k) = \frac{1}{2}(x(t_k) + x(t_{k+1})) + \frac{\Delta t_k}{8}(f(t_k, x(t_k), u(t_k)) - f(t_{k+1}, x(t_{k+1}), u(t_{k+1})))$$

$$\dot{x}(\bar{t}_k) = -\frac{3}{2\Delta t_k}(x(t_k) - x(t_{k+1})) - \frac{1}{4}(f(t_k, x(t_k), u(t_k)) + f(t_{k+1}, x(t_{k+1}), u(t_{k+1})))$$

Direct collocation (Hermite-Simpson)

The resulting discretized OCP is

$$\begin{aligned}
\underset{x,u}{\text{minimize}} \;\; & \ell_T(x(t_N)) + \sum_{k=0}^{N-1} (t_{k+1} - t_k)\ell(t_k, x(t_k), u(t_k)) \\
\text{subject to} \;\; & \dot{x}(\bar{t}_k) = f(\bar{t}_k, x(\bar{t}_k), u(\bar{t}_k)), \;\; \forall k \in \{0, 1, \ldots, N-1\} \\
& x(t_0) = x_0 \\
& x(t_N) \in \mathcal{X}_T \\
& u(t_k) \in \mathcal{U}, \;\; \forall k \in \{0, 1, \ldots, N-1\}
\end{aligned}$$

where the dynamics f are composed only once, since

$$x(\bar{t}_k) = \frac{1}{2}(x(t_k) + x(t_{k+1})) + \frac{\Delta t_k}{8}(f(t_k, x(t_k), u(t_k)) - f(t_{k+1}, x(t_{k+1}), u(t_{k+1})))$$

$$\dot{x}(\bar{t}_k) = -\frac{3}{2\Delta t_k}(x(t_k) - x(t_{k+1})) - \frac{1}{4}(f(t_k, x(t_k), u(t_k)) + f(t_{k+1}, x(t_{k+1}), u(t_{k+1})))$$

We have $u(\bar{t}_k) = u(t_k)$ if u is piecewise-constant. Alternatively, we can set u as *piecewise-linear* to get $u(\bar{t}_k) = \frac{1}{2}(u(t_k) + u(t_{k+1}))$.

Agenda

1. Indirect versus direct methods
2. Direct multiple shooting
3. Direct collocation
4. Sequential convex programming
5. Direct methods in practice

Sequential convex programming (SCP)

Consider the non-convex problem

$$\begin{aligned} \underset{z \in \mathbb{R}^d}{\text{minimize}} \; & f(z) \\ \text{subject to } & h(z) = 0 \\ & g(z) \preceq 0 \end{aligned}$$

The basic idea of SCP is to iteratively solve for $z^{(k)}$ via the convex sub-problem

$$\begin{aligned} \underset{z \in \mathbb{R}^d}{\text{minimize}} \; & \hat{f}^{(k)}(z) \\ \text{subject to } & \hat{h}^{(k)}(z) \coloneqq \hat{A}^{(k)} z - \hat{b}^{(k)} = 0 \\ & \hat{g}^{(k)}(z) \preceq 0 \\ & z \in \mathcal{T}^{(k)} \coloneqq \{ z \in \mathbb{R}^d \mid \| z - z^{(k)} \|_\infty \leq \rho^{(k)} \} \end{aligned}$$

where $(\hat{f}^{(k)}, \hat{g}^{(k)})$ and $\hat{h}^{(k)}$ are convex and affine, respectively, *approximations* of (f, g) and h, respectively, over a convex *trust region* $\mathcal{T}^{(k)}$ around $z^{(k)}$ for some $\rho^{(k)} > 0$.

Consider the discrete-time OCP

$$\begin{aligned} \underset{x,u}{\text{minimize}} \;\; & \ell_T(x_T) + \sum_{t=0}^{T-1} \ell(t, x_t, u_t) \\ \text{subject to} \;\; & x_{t+1} = f(t, x_t, u_t), \;\; \forall t \in \{0, 1, \ldots, T-1\} \\ & x_0 = \bar{x}_0 \\ & x_T \in \mathcal{X}_T \\ & u_t \in \mathcal{U}, \;\; \forall t \in \{0, 1, \ldots, T-1\} \end{aligned}$$

Often we can assume $\mathcal{X}_T$ and $\mathcal{U}$ are convex sets (e.g., polyhedra and norm balls). We also are often interested in convex cost functions, e.g.,

$$\ell_T(x_T) = x_T^{\mathsf{T}} Q_T x_T, \quad \ell(t, x_t, u_t) = x_t^{\mathsf{T}} Q_t x_t + u_t^{\mathsf{T}} R_t u_t,$$

where $Q_T \succ 0$, $Q_t \succ 0$, and $R_t \succ 0$.

Consider the discrete-time OCP

$$\begin{aligned}
\underset{x,u}{\text{minimize}} \;\; & \ell_T(x_T) + \sum_{t=0}^{T-1} \ell(t, x_t, u_t) \\
\text{subject to} \;\; & x_{t+1} = f(t, x_t, u_t), \;\; \forall t \in \{0, 1, \ldots, T-1\} \\
& x_0 = \bar{x}_0 \\
& x_T \in \mathcal{X}_T \\
& u_t \in \mathcal{U}, \;\; \forall t \in \{0, 1, \ldots, T-1\}
\end{aligned}$$

Assume $\mathcal{X}_T$ and $\mathcal{U}$ are convex sets, and ℓ_T and ℓ are convex in x_T and (x_t, u_t), respectively. Then the remaining non-convexity is due to the nonlinear dynamics constraints.

SCP for OCPs

Let $\{x_t^{(k)}\}_{t=0}^{T}$ and $\{u_t^{(k)}\}_{t=0}^{T-1}$ represent our current solution iterate. Linearize the dynamics around this iterate to get the estimate

$$\hat{f}^{(k)}(t, x_t, u_t) \coloneqq f(t, x_t^{(k)}, u_t^{(k)}) + \frac{\partial f}{\partial x}(t, x_t^{(k)}, u_t^{(k)})(x_t - x_t^{(k)}) + \frac{\partial f}{\partial u}(t, x_t^{(k)}, u_t^{(k)})(u_t - u_t^{(k)})$$

which allows us to construct the *convex* OCP

$$\begin{aligned}
\underset{x,u}{\text{minimize}} \quad & \ell_T(x_T) + \sum_{t=0}^{T-1} \ell(t, x_t, u_t) \\
\text{subject to} \quad & x_{t+1} = \hat{f}^{(k)}(t, x_t, u_t), \ \forall t \in \{0, 1, \ldots, T-1\} \\
& x_0 = \bar{x}_0 \\
& x_T \in \mathcal{X}_T \\
& u_t \in \mathcal{U}, \ \forall t \in \{0, 1, \ldots, T-1\} \\
& \|x_t - x_t^{(k)}\|_\infty \leq \rho_x^{(k)}, \ \forall t \in \{0, 1, \ldots, T-1\} \\
& \|u_t - u_t^{(k)}\|_\infty \leq \rho_u^{(k)}, \ \forall t \in \{0, 1, \ldots, T-1\}
\end{aligned}$$

Agenda

1. Indirect versus direct methods
2. Direct multiple shooting
3. Direct collocation
4. Sequential convex programming
5. Direct methods in practice

Direct methods in practice

"As you begin to play with these algorithms on your own problems, you might feel like you're on an emotional roller-coaster." - Russ Tedrake, *Underactuated Robotics*

In general, there are no guarantees for solving nonlinear optimization problems. You can converge to a bad local minimum, or not at all.

You may need to spend some time tuning, e.g., your cost function and trust region radii, or perhaps adding slack variables.

It is also a good idea to try "warm-starting" your initial guess in SCP with the solution to an easier problem (e.g., one with looser constraints).

Next class

Dynamic programming
(i.e., optimal *closed-loop* control via recursion)