Aalto University

Continuous-time optimal control: Shooting

CHEM-E7225 (was E7195), 2020-2021

Francesco Corona (¬_¬)

Chemical and Metallurgical Engineering
School of Chemical Engineering

Overview

We combined the notions on dynamic systems and simulation with the notions on nonlinear programming, to formulate a general **discrete-time optimal control** problem

- We understood and treated them as special forms of nonlinear programs

$$
\begin{aligned}
\min_{\substack{x_0, x_1, \ldots, x_K \\ u_0, u_1, \ldots, u_{K-1}}} \quad & E\left(x_K\right) + \sum_{k=0}^{K-1} L\left(x_k, u_k\right) \\
\text{subject to} \quad & x_{k+1} - f\left(x_k, u_k | \theta_x\right) = 0, \quad && k = 0, 1, \ldots, K-1 \\
& h\left(x_k, u_k\right) \leq 0, && k = 0, 1, \ldots, K-1 \\
& r\left(x_0, x_K\right) \leq 0
\end{aligned}
$$

In general, the system dynamics are defined in continuous time

⇝ The control inputs are continuous functions of time

We are interested in the continuous-time formulation

- We discuss more precisely the discretisation

Formulation

Continuous-time optimal control

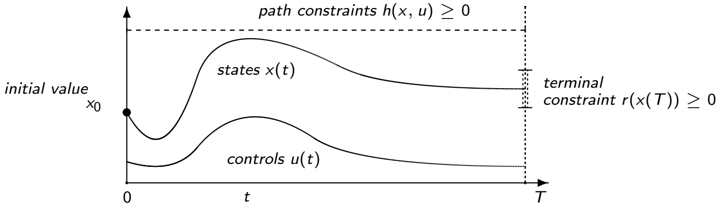
Formulation

The simplest form of continuous-time optimal control lets all functions be continuous

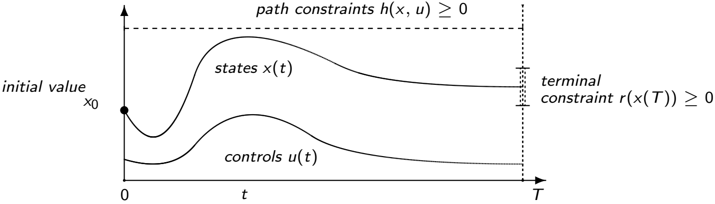
$$\min_{\substack{x(0 \rightsquigarrow T) \\ u(0 \rightsquigarrow T)}} \quad E\left(x(T)\right) + \int_0^T L\left(x(t), u(t)\right) dt$$

$$\begin{aligned} \text{subject to} \quad & \dot{x}(t) - f\left(x(t), u(t)\right) = 0, && t \in [0, T] && \text{(Dynamics)} \\ & h\left(x(t), u(t)\right) \leq 0, && t \in [0, T] && \text{(Path constraints)} \\ & x_0 - x(0) = 0 && (t = 0) && \text{(Initial value)} \\ & r\left(x(T)\right) \leq 0 && (t = T) && \text{Terminal constraint} \end{aligned}$$

That is, the optimisation is over state ad control trajectories, $x(0 \rightsquigarrow T)$ and $u(0 \rightsquigarrow T)$



Formulation (cont.)



The state are a continuous and differentiable function of time over the interval $[0, T]$

Similarly, also the controls are function of time over $[0, T]$

⇝ Though, they can be rough or jumpy fuctions

Formulation (cont.)

$$\min_{\substack{x(0 \rightsquigarrow T) \\ u(0 \rightsquigarrow T)}} \underbrace{\underbrace{E\left(x(T)\right)}_{\text{Mayer term}} + \underbrace{\int_0^T L\left(x(t), u(t)\right) dt}_{\text{Lagrange term}}}_{\text{Bolza objective function}}$$

$$\begin{aligned}
\text{subject to} \quad & \dot{x}(t) - f\left(x(t), u(t)\right) = 0, && t \in [0, T] && \text{(Dynamics)} \\
& h\left(x(t), u(t)\right) \leq 0, && t \in [0, T] && \text{(Path constraints)} \\
& x_0 - x(0) = 0 && (t = 0) && \text{(Initial value)} \\
& r\left(x(T)\right) \leq 0 && (t = T) && \text{Terminal constraint}
\end{aligned}$$

We constrain the initial value of the state to be x_0, by explicitly setting $x(0) = x_0$

Moreover, we constrain the state to satisfy the continuous-time dynamics in $[0, T]$

$$\dot{x}(t) - f\left(x(t), u(t)\right) = 0, \qquad t \in [0, T]$$

When the initial state $x(0)$ is fixed and the trajectory of the controls $u(t)$ are known in $[0, T]$, the dynamic constraint will determine the trajectory of the state $x(t)$ in $[0, T]$

Formulation (cont.)

$$\dot{x}(t) - f\left(x(t), u(t)\right) = 0, \qquad t \in [0, T]$$

In discrete-time, we have expressed the dynamic constraint as a vector of constraints

$$\underbrace{\begin{bmatrix} x_1 - f\left(x_0, u_0\right) \\ x_2 - f\left(x_1, u_1\right) \\ \vdots \\ x_{k+1} - f\left(x_k, u_k\right) \\ \vdots \\ x_{K-1} - f\left(x_{K-2}, u_{K-2}\right) \\ x_K - f\left(x_{K-1}, u_{K-1}\right) \end{bmatrix}}_{K \times N_x}$$

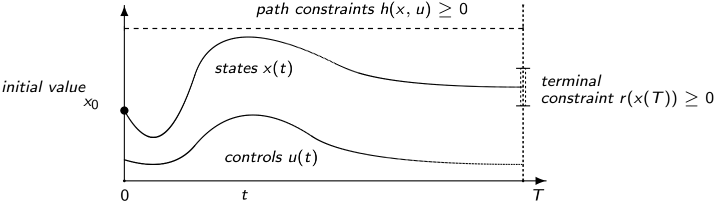
In continuous-time, the dynamic constraint is understood as an infinitely long vector

Formulation (cont.)

$$\min_{\substack{x(0 \rightsquigarrow T) \\ u(0 \rightsquigarrow T)}} \underbrace{\underbrace{E\left(x(T)\right)}_{\text{Mayer term}} + \underbrace{\int_0^T L\left(x(t), u(t)\right) dt}_{\text{Lagrange term}}}_{\text{Bolza objective function}}$$

$$\begin{aligned} \text{subject to} \quad & \dot{x}(t) - f\left(x(t), u(t)\right) = 0, && t \in [0, T] && \text{(Dynamics)} \\ & h\left(x(t), u(t)\right) \leq 0, && t \in [0, T] && \text{(Path constraints)} \\ & x_0 - x(0) = 0 && (t = 0) && \text{(Initial value)} \\ & r\left(x(T)\right) \leq 0 && (t = T) && \text{Terminal constraint} \end{aligned}$$

We constrain trajectories along the path, by explicitly setting an inequality constraint



Formulation (cont.)

$$h\left(x(t), u(t)\right) \leq 0, \qquad t \in [0, T]$$

In discrete-time, we have expressed the path constraint as a vector of constraints

$$\begin{bmatrix} h\left(x_0, u_0\right) \\ h\left(x_1, u_1\right) \\ \vdots \\ h\left(x_k, u_k\right) \\ \vdots \\ h\left(x_K, \cancel{u_K}\right) \end{bmatrix}$$

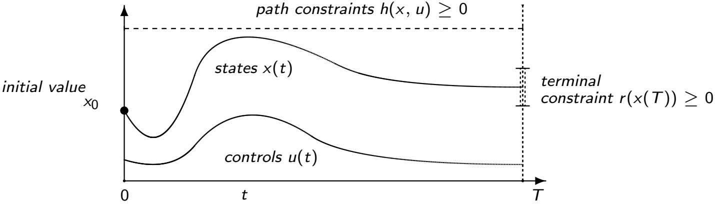
In continuous-time, the path constraint is understood as an infinitely long vector

Formulation (cont.)

$$\min_{\substack{x(0 \rightsquigarrow T)\\ u(0 \rightsquigarrow T)}} \underbrace{\underbrace{E\left(x(T)\right)}_{\text{Mayer term}} + \underbrace{\int_0^T L\left(x(t), u(t)\right) dt}_{\text{Lagrange term}}}_{\text{Bolza objective function}}$$

$$\begin{aligned} \text{subject to} \quad & \dot{x}(t) - f\left(x(t), u(t)\right) = 0, && t \in [0, T] && \text{(Dynamics)} \\ & h\left(x(t), u(t)\right) \leq 0, && t \in [0, T] && \text{(Path constraints)} \\ & x_0 - x(0) = 0 && (t = 0) && \text{(Initial value)} \\ & r\left(x(T)\right) \leq 0 && (t = T) && \text{Terminal constraint} \end{aligned}$$

A terminal constraint is expressed as inequality constraint on the terminal state $x(T)$



Numerics

Continuous-time optimal control

Overview of numerical approaches

There exist three main classes of approaches to solve continuous-time optimal control

- **State-space methods** are based on the Bellman's **principle of optimality**
  - The **Hamilton-Jocobi-Bellman** equation, HJB
  - (Continuous-time dynamic programming)
- **Indirect methods** are based on the Pontryangin's **minimum principle**
  - First-optimise, then discretise
- **Direct methods** are based on transcriptions as **nonlinear programs**
  - First-discretise, then optimise

Direct methods | Single-shooting

$$\min_{\substack{x(0 \rightsquigarrow T) \\ u(0 \rightsquigarrow T)}} \quad E\left(x(T)\right) + \int_0^T L\left(x(t), u(t)\right) dt$$

$$\begin{aligned}
\text{subject to} \quad & \dot{x}(t) - f\left(x(t), u(t)\right) = 0, && t \in [0, T] && \text{(Dynamics)} \\
& h\left(x(t), u(t)\right) \leq 0, && t \in [0, T] && \text{(Path constraints)} \\
& x_0 - x(0) = 0 && (t = 0) && \text{(Initial value)} \\
& r\left(x(T)\right) \leq 0 && (t = T) && \text{Terminal constraint}
\end{aligned}$$

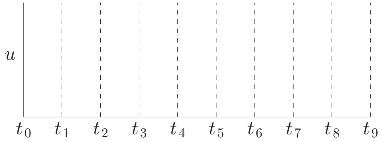
The general idea of single shooting methods is common to all the shooting methods

- Use an embedded integrator of the differential model
- To eliminate the continuous-time dynamics

Direct methods | Single-shooting

$$\min_{\substack{x(0\rightsquigarrow T)\\ u(0\rightsquigarrow T)}} \quad E\left(x(T)\right) + \int_0^T L\left(x(t), u(t)\right) dt$$

$$\begin{aligned}
\text{subject to} \quad & \dot{x}(t) - f\left(x(t), u(t)\right) = 0, && t \in [0, T] && \text{(Dynamics)}\\
& h\left(x(t), u(t)\right) \leq 0, && t \in [0, T] && \text{(Path constraints)}\\
& x_0 - x(0) = 0 && (t = 0) && \text{(Initial value)}\\
& r\left(x(T)\right) \leq 0 && (t = T) && \text{Terminal constraint}
\end{aligned}$$



First-discretise, then optimise

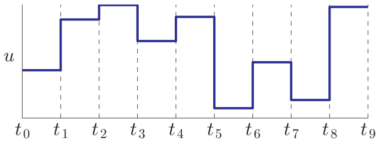
- Define a fixed time-grid for $[0, T]$

$$0 = t_0 < t_1 < \cdots < t_{K-1} < t_K = T$$

The time-intervals do not need to be necessarily equally spaced, though this is common

Direct methods | Single-shooting (cont.)

$$
\begin{aligned}
\min_{\substack{x(0\rightsquigarrow T)\\ u(0\rightsquigarrow T)}} \quad & E\left(x(T)\right) + \int_0^T L\left(x(t), u(t)\right) dt \\
\text{subject to} \quad & \dot{x}(t) - f\left(x(t), u(t)\right) = 0, && t \in [0, T] && \text{(Dynamics)} \\
& h\left(x(t), u(t)\right) \leq 0, && t \in [0, T] && \text{(Path constraints)} \\
& x_0 - x(0) = 0 && (t = 0) && \text{(Initial value)} \\
& r\left(x(T)\right) \leq 0 && (t = T) && \text{Terminal constraint}
\end{aligned}
$$



First-discretise, then optimise

- Define a fixed time-grid for $[0, T]$

$$0 = t_0 < t_1 < \cdots < t_{K-1} < t_K = T$$

- Discretise the controls $u(t)$

$$u(t \in [t_k, t_{k+1}]) = u_k$$

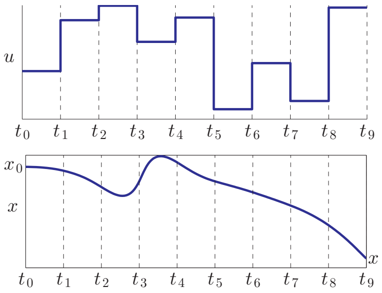
The control trajectory $u(t)$ is commonly parameterised by piecewise constant functions

- Other parameterisations are possible (other piecewise polynomials)

Direct methods | Single-shooting (cont.)

$$\min_{\substack{x(0 \rightsquigarrow T)\\ u(0 \rightsquigarrow T)}} \quad E\left(x(T)\right) + \int_0^T L\left(x(t), u(t)\right) dt$$

$$\begin{aligned}
\text{subject to} \quad & \dot{x}(t) - f\left(x(t), u(t)\right) = 0, && t \in [0, T] && \text{(Dynamics)}\\
& h\left(x(t), u(t)\right) \leq 0, && t \in [0, T] && \text{(Path constraints)}\\
& x_0 - x(0) = 0 && (t = 0) && \text{(Initial value)}\\
& r\left(x(T)\right) \leq 0 && (t = T) && \text{Terminal constraint}
\end{aligned}$$



First-discretise, then optimise

- Define a fixed time-grid for $[0, T]$

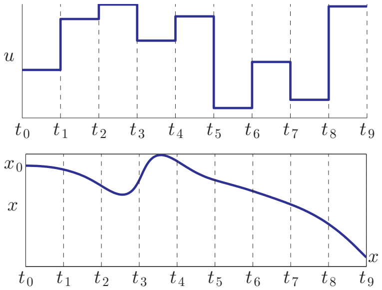
$$0 = t_0 < t_1 < \cdots < t_{K-1} < t_K = T$$

- Discretise the controls $u(t)$

$$u(t \in [t_k, t_{k+1}]) = u_k$$

- Treat the states $x(t)$ as function of discretised controls $\{u_k\}$ and x_0

Direct methods | Single-shooting (cont.)



- Define a fixed time-grid for $[0, T]$

$$0 = t_0 < t_1 < \cdots < t_{K-1} < t_K = T$$

- Discretise the controls $u(t)$

$$u(t \in [t_k, t_{k+1}]) = u_k$$

- Treat the states $x(t)$ as function of discretised controls $\{u_k\}$ and x_0

Consider the time $t \in [t_k, t_{k+1}]$, the zero-order hold control active on the interval is u_k

We denoted the state trajectory over the short interval $[t_k, t_{k+1}]$ as the **solution map**

$$\widetilde{x}_k(t|x_k, u_k), \quad t \in [t_k, t_{k+1}]$$

The final value of the short trajectory is the output of the **transition function**

$$\widetilde{x}_k(t_{k+1}|x_k, u_k) = f_{\Delta t}(x_k, u_k)$$

Direct methods | Single-shooting (cont.)

$$\begin{aligned}
\min_{\substack{x_0, x_1, \dots, x_K \\ u_0, u_1, \dots, u_{K-1}}} \quad & E(x_K) + \sum_{k=0}^{K-1} L_k(x_k, u_k) \\
\text{subject to} \quad & x_{k+1} - f_{\Delta t}(x_k, u_k | \theta_x) = 0, \qquad k = 0, 1, \dots, K-1 \\
& h(x_k, u_k) \leq 0, \qquad k = 0, 1, \dots, K-1 \\
& r(x_0, x_K) \leq 0
\end{aligned}$$

Discretising the controls transcribes the infinite dimensional problem into a finite one

Single-shooting regards the states x_k as dependent variables obtained by integration

- From the initial state x_0, under the sequence of controls $\{u_k\}$

$$\begin{aligned}
x_0 &= \underbrace{x_0}_{\overline{x}_0(x_0)} \\
x_1 &= \underbrace{f_{\Delta t}(x_0, u_0)}_{\overline{x}_1(x_0, u_0)} \\
x_2 &= f_{\Delta t}(x_1, u_1) \\
&= \underbrace{f_{\Delta t}(f_{\Delta t}(x_0, u_0), u_1)}_{\overline{x}_2(x_0, u_0, u_1)} \\
\cdots &= \cdots
\end{aligned}$$

Direct methods | Single-shooting (cont.)

$$
\begin{aligned}
\min_{\substack{x_0 \\ u_0, u_1, \dots, u_{K-1}}} \quad & E\left(\overline{x}_K\left(x_0, u_0, u_1, \dots, u_{K-1}\right)\right) + \sum_{k=0}^{K-1} L\left(\overline{x}_k\left(x_0, u_0, u_1, \dots, u_{K-1}\right), u_k\right) \\
\text{subject to} \quad & h\left(\overline{x}_k\left(x_0, u_0, u_1, \dots, u_{K-1}\right), u_k\right) \leq 0, \quad k = 0, 1, \dots, K-1 \\
& r\left(x_0, \overline{x}_N\left(x_0, u_0, u_1, \dots, u_{K-1}\right)\right) = 0
\end{aligned}
$$

Simulation and optimisation are solved sequentially, the approach is the sequential one

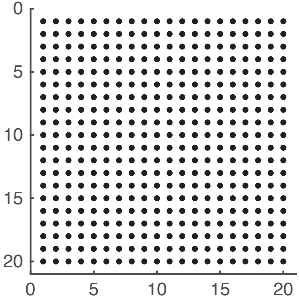
The only decision variable in the nonlinear program is the collection of control vectors

- The decision variable influences all of the problem functions

$$
\underbrace{u_0, u_1, \dots, u_{K-1}}_{K \times N_u}
$$

Direct methods | Single-shooting (cont.)

The nonlinear program is dense, any generic solver can be used for the task



The Lagrangian function $\mathcal{L}(w, \lambda, \mu)$

$$\mathcal{L}(w, \lambda, \mu) = f(w) + \lambda^T g(w) + \mu^T h(w)$$

The Hessian of the Lagrangian

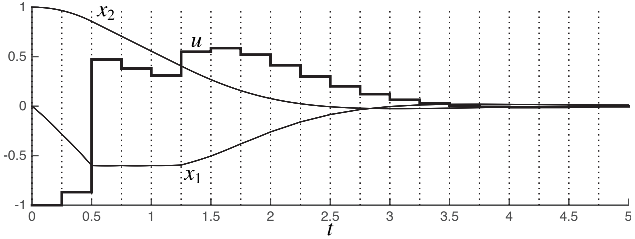
$$\nabla_w^2 \mathcal{L}(w, \lambda, \mu)$$

In general, there is no structure in $\nabla_w^2 \mathcal{L}(w, \lambda, \mu)$

$$\frac{\partial^2 \mathcal{L}(w, \lambda, \mu)}{\partial w_i \partial w_k} \neq 0$$

Direct methods | Single-shooting (cont.)

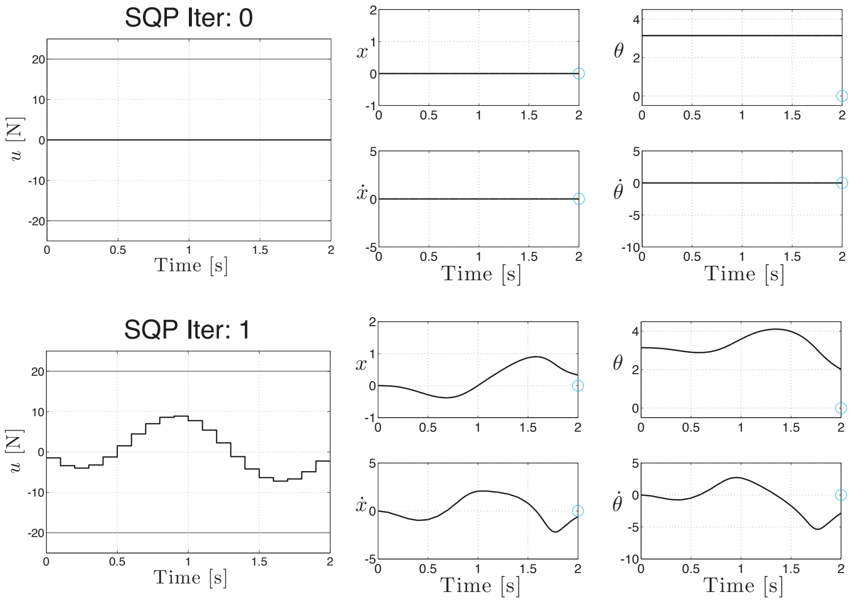
Single shooting solution using simulation based on a order-4 Runge-Kutta integrator



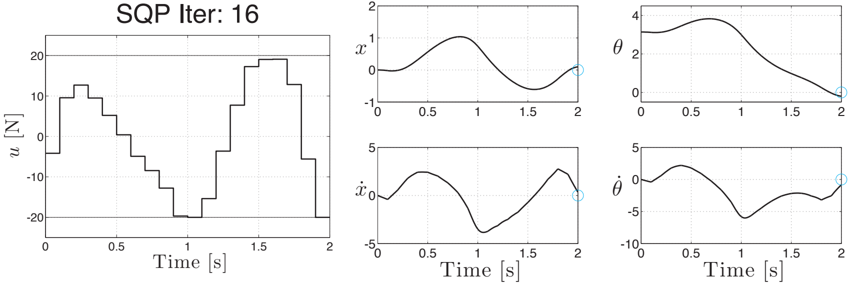
The state trajectory can be computed during the iterations of the optimisation scheme

- The model equations are satisfied by definition

Direct methods | Single-shooting (cont.)



Direct methods | Single-shooting (cont.)



Direct methods | Single-shooting (cont.)

The **forward integrator map** of the system dynamics is formally defined as a function

$$
\begin{aligned}
f_{\text{int}} &: \mathcal{R}^{N_x+(K\times N_u)} \times \mathcal{R} \rightarrow \mathcal{R}^{N_x} \\
&: (x_0, u_0, u_1, \ldots, u_{K-1}, t) \mapsto x(t)
\end{aligned}
$$

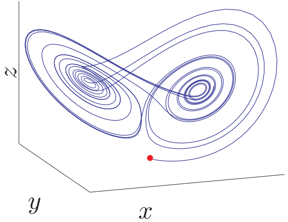
Function f_{int} propagates continuous dynamics, it may get highly nonlinear for large T

Direct methods | Single-shooting (cont.)

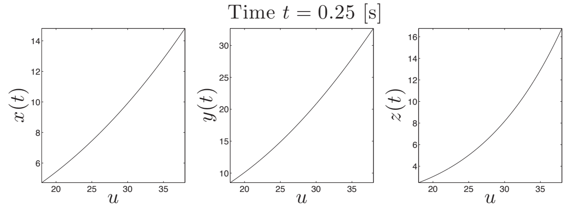
Example

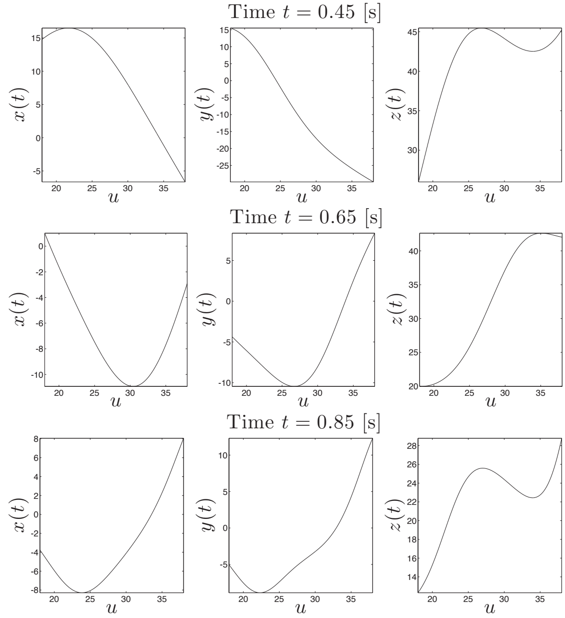
$$\dot{x}(t) = 10\,(y(t) - x(t))$$
$$\dot{y}(t) = x(t)\,(u(t) - z(t)) - y(t)$$
$$\dot{z}(t) = x(t)y(t) - 3z(t)$$

From some fixed initial condition (x_0, y_0, z_0) and constant control $u(t)$



System's states $x(t)$ as a function of the controls $u(t) = \text{const}$, at simulation time t

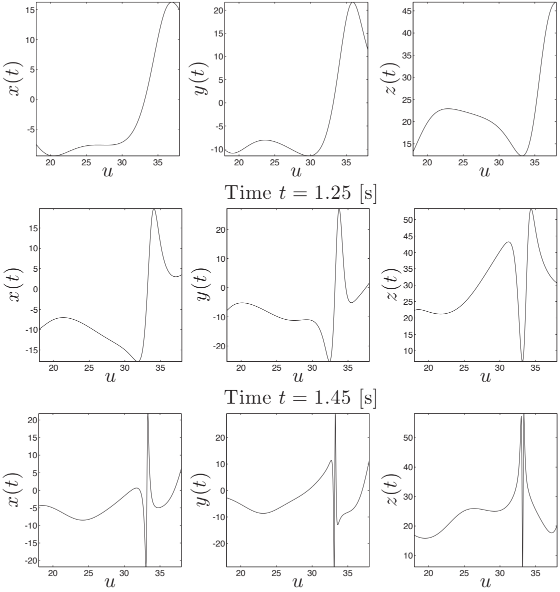



Time $t = 0.45$ [s]
Time $t = 0.65$ [s]
Time $t = 0.85$ [s]
$x(t)$
$y(t)$
$z(t)$
u

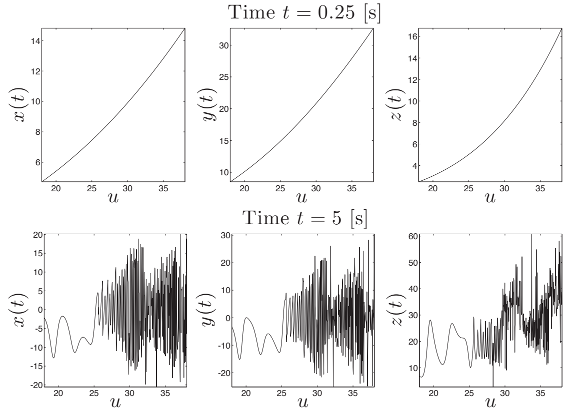
Time $t = 1.05$ [s]

Time $t = 1.25$ [s]

Time $t = 1.45$ [s]



Direct methods | Single-shooting (cont.)



For short integration times, the relationship between states and controls is close to linear and as the becomes highly nonlinear with the duration of the simulation time

Direct methods | Multiple-shooting

$$\begin{aligned} \min_{\substack{x(0 \rightsquigarrow T) \\ u(0 \rightsquigarrow T)}} \quad & E\left(x(T)\right) + \int_0^T L\left(x(t), u(t)\right) dt \\ \text{subject to} \quad & \dot{x}(t) - f\left(x(t), u(t)\right) = 0, && t \in [0, T] && \text{(Dynamics)} \\ & h\left(x(t), u(t)\right) \leq 0, && t \in [0, T] && \text{(Path constraints)} \\ & x_0 - x(0) = 0 && (t = 0) && \text{(Initial value)} \\ & r\left(x(T)\right) \leq 0 && (t = T) && \text{Terminal constraint} \end{aligned}$$

The general idea of multiple shooting methods is common to all the shooting methods

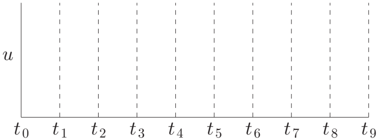
- Use an embedded integrator of the differential model
- To eliminate the continuous-time dynamics

Yet, the integration of the dynamics over a long period of time can be counterproductive

$\rightsquigarrow$ Restrict the integration to relatively shorter inntervals

Direct methods | Multiple-shooting (cont.)

$$
\begin{aligned}
\min_{\substack{x(0 \rightsquigarrow T)\\ u(0 \rightsquigarrow T)}} \quad & E\left(x(T)\right) + \int_0^T L\left(x(t), u(t)\right) dt \\
\text{subject to} \quad & \dot{x}(t) - f\left(x(t), u(t)\right) = 0, && t \in [0, T] && \text{(Dynamics)} \\
& h\left(x(t), u(t)\right) \leq 0, && t \in [0, T] && \text{(Path constraints)} \\
& x_0 - x(0) = 0 && (t = 0) && \text{(Initial value)} \\
& r\left(x(T)\right) \leq 0 && (t = T) && \text{Terminal constraint}
\end{aligned}
$$



First-discretise, then optimise

- Define a fixed time-grid for $[0, T]$

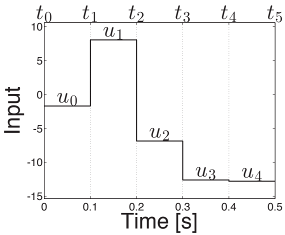
$$0 = t_0 < t_1 < \cdots < t_{K-1} < t_K = T$$

The time-intervals do not need to be necessarily equally spaced, though this is common

Direct methods | Multiple-shooting (cont.)

$$\min_{\substack{x(0 \rightsquigarrow T)\\ u(0 \rightsquigarrow T)}} \quad E\left(x(T)\right) + \int_0^T L\left(x(t), u(t)\right) dt$$

$$\begin{aligned} \text{subject to} \quad & \dot{x}(t) - f\left(x(t), u(t)\right) = 0, && t \in [0, T] && \text{(Dynamics)} \\ & h\left(x(t), u(t)\right) \leq 0, && t \in [0, T] && \text{(Path constraints)} \\ & x_0 - x(0) = 0 && (t = 0) && \text{(Initial value)} \\ & r\left(x(T)\right) \leq 0 && (t = T) && \text{Terminal constraint} \end{aligned}$$



First-discretise, then optimise

- Define a fixed time-grid for $[0, T]$

$$0 = t_0 < t_1 < \cdots < t_{K-1} < t_K = T$$

- Discretise the controls $u(t)$

$$u(t \in [t_k, t_{k+1}]) = u_k$$

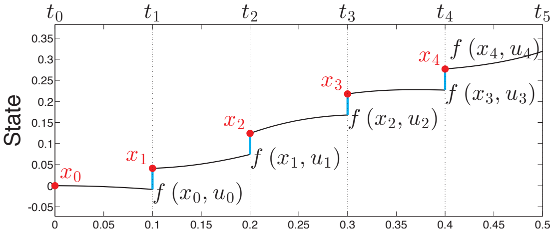
The control trajectory $u(t)$ is commonly parameterised by piecewise constant functions

- Other parameterisations are possible (other piecewise polynomials)

Direct methods | Multiple-shooting (cont.)

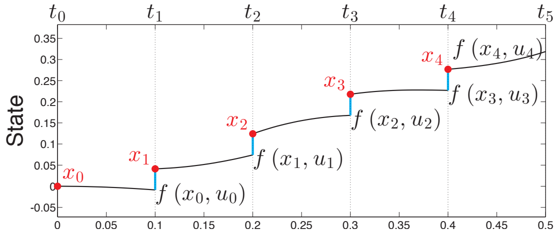
$$\min_{\substack{x(0 \rightsquigarrow T)\\ u(0 \rightsquigarrow T)}} \quad E\left(x(T)\right) + \int_0^T L\left(x(t), u(t)\right) dt$$

$$\begin{aligned}
\text{subject to} \quad & \dot{x}(t) - f\left(x(t), u(t)\right) = 0, && t \in [0, T] && \text{(Dynamics)} \\
& h\left(x(t), u(t)\right) \leq 0, && t \in [0, T] && \text{(Path constraints)} \\
& x_0 - x(0) = 0 && (t = 0) && \text{(Initial value)} \\
& r\left(x(T)\right) \leq 0 && (t = T) && \text{Terminal constraint}
\end{aligned}$$



Treat the states $x(t)$ as function of discretised controls u_k, starting from a given x_k

Direct methods | Multiple-shooting (cont.)



The integration of the dynamics is performed over the much sorter interval $[t_k, t_{k+1}]$

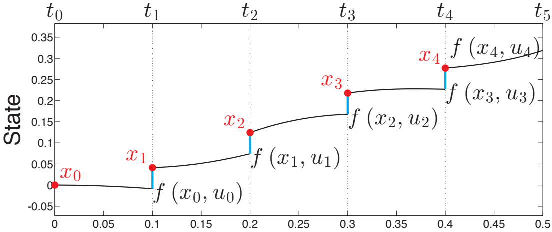
- The forward integrator is only mildly nonlinear

$$x_{k+1} = f_{\Delta t}(x_k, u_k | \theta_x)$$

The states x_k used in the integration become decision variables of the optimisation

$$\begin{aligned}
\min_{\substack{x_0, x_1, \dots, x_K \\ u_0, u_1, \dots, u_{K-1}}} \quad & E(x_K) + \sum_{k=0}^{K-1} L(x_k, u_k) \\
\text{subject to} \quad & x_{k+1} - f_{\Delta t}(x_k, u_k | \theta_x) = 0, \qquad k = 0, 1, \dots, K-1 \\
& h(x_k, u_k) \leq 0, \qquad k = 0, 1, \dots, K-1 \\
& r(x_0, x_N) = 0
\end{aligned}$$

Direct methods | Multiple-shooting (cont.)



The integration over the interval $[t_k, t_{k+1}]$ is meaningful if the shooting gap is closed

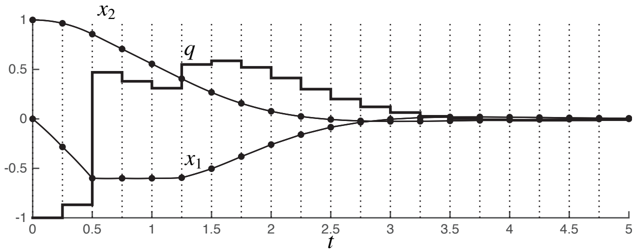
$$x_{k+1} - f_{\Delta t}(x_k, u_k | \theta_x) = 0$$

To ensure continuity, the shooting gaps become equality constraints of the optimisation

$$\begin{aligned} \min_{\substack{x_0, x_1, \dots, x_K \\ u_0, u_1, \dots, u_{K-1}}} \quad & E(x_K) + \sum_{k=0}^{K-1} L(x_k, u_k) \\ \text{subject to} \quad & x_{k+1} - f_{\Delta t}(x_k, u_k | \theta_x) = 0, \qquad k = 0, 1, \dots, K-1 \\ & h(x_k, u_k) \leq 0, \qquad k = 0, 1, \dots, K-1 \\ & r(x_0, x_N) = 0 \end{aligned}$$

Direct methods | Mutliple-shooting (cont.)

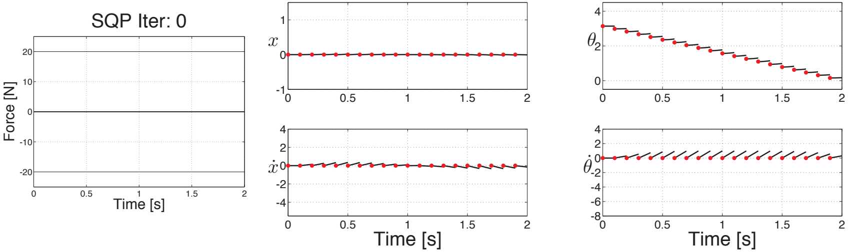
Multiple shooting solution using simulation based on a order-4 Runge-Kutta integrator

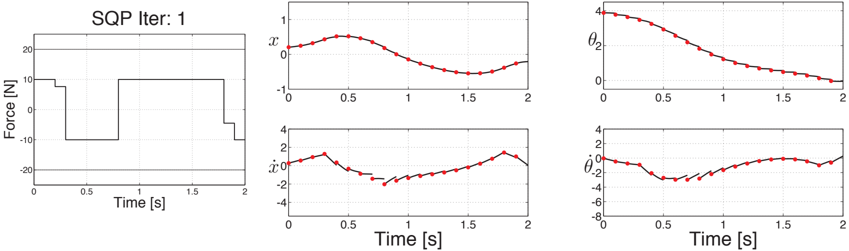


Because the continuity conditions hold, the short-interval simulations join at time nodes

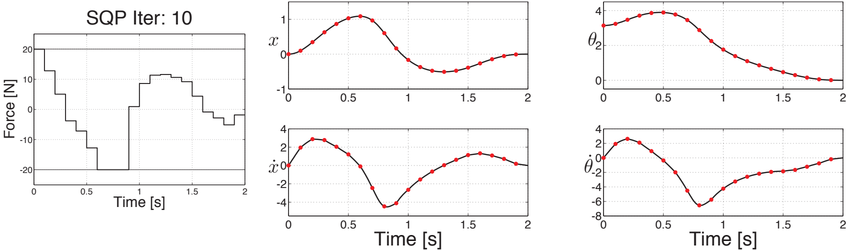
- The model equations satisfied only once the nonlinear program has converged

Direct methods | Multiple-shooting (cont.)



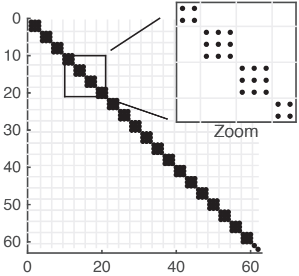


Direct methods | Multiple-shooting (cont.)



Direct methods | Multiple-shooting (cont.)

The nonlinear program is sparse, structure-exploiting solvers should be used



The Lagrangian function $\mathcal{L}(w, \lambda, \mu)$

$$\mathcal{L}(w, \lambda, \mu) = f(w) + \lambda^T g(w) + \mu^T h(w)$$

The Hessian of the Lagrangian

$$\nabla_w^2 \mathcal{L}(w, \lambda, \mu)$$

The Hessian of the Lagrangian is block-diagonal, with small symmetric blocks

- All the other second derivatives are zero

The block-diagonality property of the Hessian is extremely favourable

- ⇝ Hessian approximations
- ⇝ QP subproblems