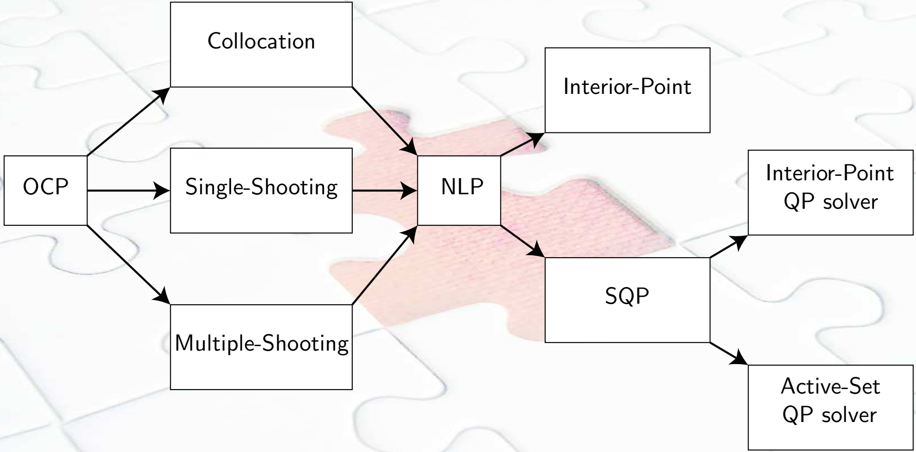
Numerical Optimal Control
Lecture 4: Shooting Methods

Sébastien Gros

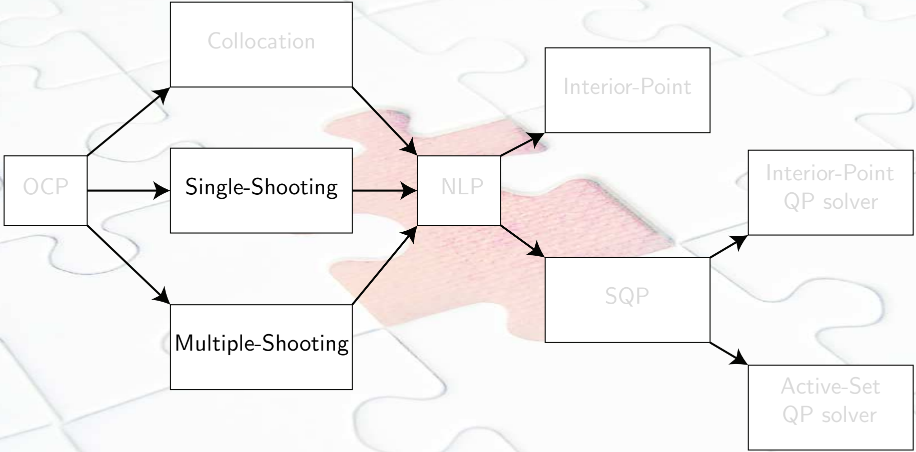
ITK NTNU

NTNU PhD course

Survival map of Direct Optimal Control



Survival map of Direct Optimal Control



One way of going from OCP to NLP

Outline

1. Single-Shooting
2. Multiple-Shooting
3. NLP from shooting methods

Outline

1. Single-Shooting
2. Multiple-Shooting
3. NLP from shooting methods

Discretization - Single-Shooting

Problem:

$$
\begin{aligned}
\min \quad & \phi\left(\mathbf{x}(.), \mathbf{u}(.)\right) \\
\text{s.t.} \quad & \dot{\mathbf{x}}\left(t\right) = \mathbf{F}\left(\mathbf{x}\left(t\right), \mathbf{u}\left(t\right)\right) \\
& \mathbf{h}\left(\mathbf{x}(t), \mathbf{u}(t), t\right) \leq 0 \\
& \mathbf{x}\left(t_0\right) = \mathbf{x}_0
\end{aligned}
$$

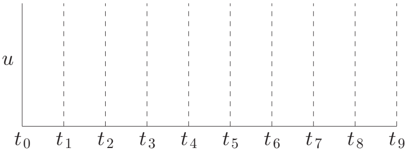
First discretize...

...then optimize:

Discretization - Single-Shooting

Problem:

$$\begin{aligned} \min \quad & \phi\left(\mathbf{x}(.), \mathbf{u}(.)\right) \\ \text{s.t.} \quad & \dot{\mathbf{x}}\left(t\right) = \mathbf{F}\left(\mathbf{x}\left(t\right), \mathbf{u}\left(t\right)\right) \\ & \mathbf{h}\left(\mathbf{x}(t), \mathbf{u}(t), t\right) \leq 0 \\ & \mathbf{x}\left(t_0\right) = \mathbf{x}_0 \end{aligned}$$



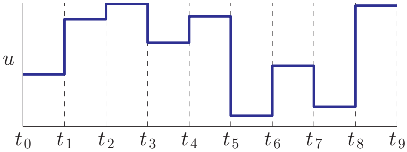
First discretize...

...then optimize:

Discretization - Single-Shooting

Problem:

$$\begin{aligned} \min \quad & \phi\left(\mathbf{x}(.), \mathbf{u}(.)\right) \\ \text{s.t.} \quad & \dot{\mathbf{x}}\left(t\right) = \mathbf{F}\left(\mathbf{x}\left(t\right), \mathbf{u}\left(t\right)\right) \\ & \mathbf{h}\left(\mathbf{x}(t), \mathbf{u}(t), t\right) \leq 0 \\ & \mathbf{x}\left(t_0\right) = \mathbf{x}_0 \end{aligned}$$



First discretize...

- Usually zero-order hold

$$\mathbf{u}\left(t \in [t_k, t_{k+1}]\right) = \mathbf{u}_k$$

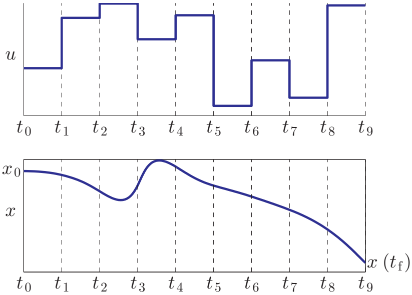
over a time grid t_0, ..., t_N

...then optimize:

Discretization - Single-Shooting

Problem:

$$\begin{aligned} \min \quad & \phi\left(\mathbf{x}(.), \mathbf{u}(.)\right) \\ \text{s.t.} \quad & \dot{\mathbf{x}}\left(t\right) = \mathbf{F}\left(\mathbf{x}\left(t\right), \mathbf{u}\left(t\right)\right) \\ & \mathbf{h}\left(\mathbf{x}(t), \mathbf{u}(t), t\right) \leq 0 \\ & \mathbf{x}\left(t_0\right) = \mathbf{x}_0 \end{aligned}$$



First discretize...

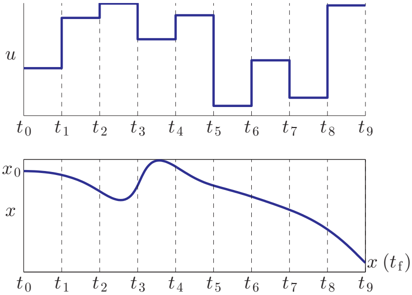
- Usually zero-order hold
$$\mathbf{u}\left(t \in [t_k, t_{k+1}]\right) = \mathbf{u}_k$$
over a time grid $t_0, ..., t_N$
- See $\mathbf{x}(.)$ as a *function* $\mathbf{f}$ of $\mathbf{w} = \{\mathbf{u}_0, ..., \mathbf{u}_{N-1}\}$, $\mathbf{x}_0$ and t:
$$\mathbf{f}\left(\mathbf{w}, \mathbf{x}_0, t\right) : \mathbf{w}, \mathbf{x}_0, t \mapsto \mathbf{x}(t)$$

...then optimize:

Discretization - Single-Shooting

Problem:

$$\begin{aligned} \min \quad & \phi\left(\mathbf{x}(.), \mathbf{u}(.)\right) \\ \text{s.t.} \quad & \dot{\mathbf{x}}\left(t\right) = \mathbf{F}\left(\mathbf{x}\left(t\right), \mathbf{u}\left(t\right)\right) \\ & \mathbf{h}\left(\mathbf{x}(t), \mathbf{u}(t), t\right) \leq 0 \\ & \mathbf{x}\left(t_0\right) = \mathbf{x}_0 \end{aligned}$$



First discretize...

- Usually zero-order hold

$$\mathbf{u}\left(t \in [t_k, t_{k+1}]\right) = \mathbf{u}_k$$

  over a time grid $t_0, ..., t_N$

- See $\mathbf{x}(.)$ as a *function* $\mathbf{f}$ of $\mathbf{w} = \{\mathbf{u}_0, ..., \mathbf{u}_{N-1}\}$, $\mathbf{x}_0$ and t:

$$\mathbf{f}\left(\mathbf{w}, \mathbf{x}_0, t\right) : \ \mathbf{w}, \mathbf{x}_0, t \mapsto \mathbf{x}(t)$$

  given by:

$$\frac{\partial}{\partial t}\mathbf{f}\left(\mathbf{w}, \mathbf{x}_0, t\right) = \mathbf{F}\left(\mathbf{f}\left(\mathbf{w}, \mathbf{x}_0, t\right), \mathbf{u}_k\right),$$

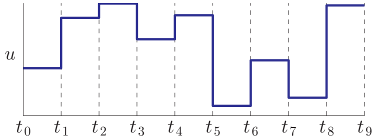
$$\mathbf{f}\left(\mathbf{w}, \mathbf{x}_0, t_0\right) = \mathbf{x}_0$$

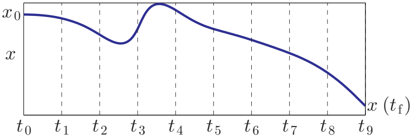
...then optimize:

Discretization - Single-Shooting

Problem:

$$\begin{aligned} \min \quad & \phi\left(\mathbf{x}(.), \mathbf{u}(.)\right) \\ \text{s.t.} \quad & \dot{\mathbf{x}}\left(t\right) = \mathbf{F}\left(\mathbf{x}\left(t\right), \mathbf{u}\left(t\right)\right) \\ & \mathbf{h}\left(\mathbf{x}(t), \mathbf{u}(t), t\right) \le 0 \\ & \mathbf{x}\left(t_0\right) = \mathbf{x}_0 \end{aligned}$$





First discretize...

- Usually zero-order hold
$$\mathbf{u}\left(t \in [t_k, t_{k+1}]\right) = \mathbf{u}_k$$
over a time grid $t_0, ..., t_N$
- See $\mathbf{x}(.)$ as a *function* $\mathbf{f}$ of $\mathbf{w} = \{\mathbf{u}_0, ..., \mathbf{u}_{N-1}\}$, $\mathbf{x}_0$ and t:
$$\mathbf{f}\left(\mathbf{w}, \mathbf{x}_0, t\right) : \ \mathbf{w}, \mathbf{x}_0, t \mapsto \mathbf{x}(t)$$

given by:

$$\frac{\partial}{\partial t}\mathbf{f}\left(\mathbf{w}, \mathbf{x}_0, t\right) = \mathbf{F}\left(\mathbf{f}\left(\mathbf{w}, \mathbf{x}_0, t\right), \mathbf{u}_k\right),$$
$$\mathbf{f}\left(\mathbf{w}, \mathbf{x}_0, t_0\right) = \mathbf{x}_0$$

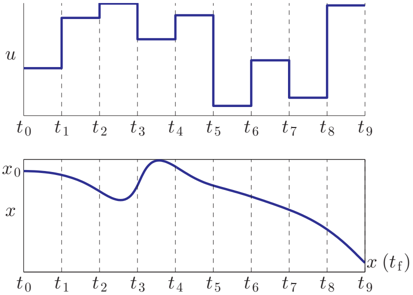
...then optimize: NLP

$$\begin{aligned} \min_{\mathbf{w}} \quad & \phi\left(\mathbf{f}\left(\mathbf{w}, \mathbf{x}_0, .\right), \mathbf{w}\right) \\ \text{s.t.} \quad & \mathbf{h}\left(\mathbf{f}\left(\mathbf{w}, \mathbf{x}_0, t_k\right), \mathbf{w}_k, t_k\right) \le 0 \\ & \mathbf{f}\left(\mathbf{w}, \mathbf{x}_0, t_f\right) = \mathbf{x}_f \end{aligned}$$

Discretization - Single-Shooting

Problem:

$$\begin{array}{ll} \min & \phi\left(\mathbf{x}(.), \mathbf{u}(.)\right) \\ \text{s.t.} & \dot{\mathbf{x}}\left(t\right) = \mathbf{F}\left(\mathbf{x}\left(t\right), \mathbf{u}\left(t\right)\right) \\ & \mathbf{h}\left(\mathbf{x}(t), \mathbf{u}(t), t\right) \leq 0 \\ & \mathbf{x}\left(t_0\right) = \mathbf{x}_0 \end{array}$$



First discretize...

- Usually zero-order hold
$$\mathbf{u}\left(t \in [t_k, t_{k+1}]\right) = \mathbf{u}_k$$
over a time grid t_0, ..., t_N
- See $\mathbf{x}(.)$ as a *function* $\mathbf{f}$ of $\mathbf{w} = \{\mathbf{u}_0, ..., \mathbf{u}_{N-1}\}$, $\mathbf{x}_0$ and t:
$$\mathbf{f}\left(\mathbf{w}, \mathbf{x}_0, t\right) : \ \mathbf{w}, \mathbf{x}_0, t \mapsto \mathbf{x}(t)$$
given by:
$$\frac{\partial}{\partial t}\mathbf{f}\left(\mathbf{w}, \mathbf{x}_0, t\right) = \mathbf{F}\left(\mathbf{f}\left(\mathbf{w}, \mathbf{x}_0, t\right), \mathbf{u}_k\right),$$
$$\mathbf{f}\left(\mathbf{w}, \mathbf{x}_0, t_0\right) = \mathbf{x}_0$$

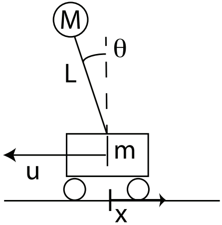
...then optimize: NLP, checkpoints t_k for $\mathbf{h}$

$$\begin{array}{ll} \min\limits_{\mathbf{w}} & \phi\left(\mathbf{f}\left(\mathbf{w}, \mathbf{x}_0, .\right), \mathbf{w}\right) \\ \text{s.t.} & \mathbf{h}\left(\mathbf{f}\left(\mathbf{w}, \mathbf{x}_0, t_k\right), \mathbf{w}_k, t_k\right) \leq 0 \\ & \mathbf{f}\left(\mathbf{w}, \mathbf{x}_0, t_f\right) = \mathbf{x}_f \end{array}$$

Single Shooting - Example: swing-up of a pendulum

OCP

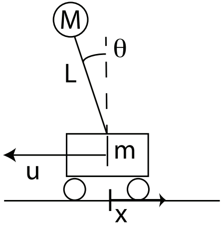
$$
\begin{aligned}
\min_{u_0,\dots,u_{N-1}} \quad & \sum_{k=0}^{N} u_k^2 \\
\text{s.t.} \quad & \dot{\mathbf{x}} = \mathbf{F}\left(\mathbf{x}, u_k\right), \quad \forall t \in [t_k, t_{k+1}] \\
& -20 \le u_k \le 20 \\
& \mathbf{x}(0) \equiv \mathbf{x}_0 = \begin{bmatrix} 0 & \pi & 0 & 0 \end{bmatrix}, \quad \mathbf{x}\left(t_\mathrm{f}\right) = 0
\end{aligned}
$$



Single Shooting - Example: swing-up of a pendulum

OCP

$$\begin{aligned}\min_{u_0,\dots,u_{N-1}} \quad & \sum_{k=0}^{N} u_k^2 \\ \text{s.t.} \quad & \dot{\mathbf{x}} = \mathbf{F}\left(\mathbf{x}, u_k\right), \quad \forall t \in [t_k, t_{k+1}] \\ & -20 \leq u_k \leq 20 \\ & \mathbf{x}(0) \equiv \mathbf{x}_0 = \begin{bmatrix} 0 & \pi & 0 & 0 \end{bmatrix}, \quad \mathbf{x}\left(t_f\right) = 0\end{aligned}$$



Integrator function provides $\mathbf{x}(t) = \mathbf{f}\left(\mathbf{u}_0, \dots, \mathbf{u}_{N-1}, \mathbf{x}_0, t\right)$

Single Shooting - Example: swing-up of a pendulum

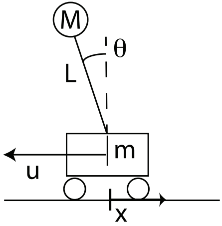
OCP

$$\min_{u_0,\dots,u_{N-1}} \quad \sum_{k=0}^{N} u_k^2$$

$$\text{s.t.} \quad \dot{\mathbf{x}} = \mathbf{F}\left(\mathbf{x}, u_k\right), \quad \forall t \in [t_k, t_{k+1}]$$

$$-20 \leq u_k \leq 20$$

$$\mathbf{x}(0) \equiv \mathbf{x}_0 = \begin{bmatrix} 0 & \pi & 0 & 0 \end{bmatrix}, \quad \mathbf{x}\left(t_f\right) = 0$$



Integrator function provides $\mathbf{x}(t) = \mathbf{f}\left(\mathbf{u}_0, \dots, \mathbf{u}_{N-1}, \mathbf{x}_0, t\right)$

An integrator is a function in the most rigorous sense of the term. E.g.

$$\frac{\partial \mathbf{f}}{\partial \mathbf{u}_k}\left(\mathbf{u}_0, \dots, \mathbf{u}_{N-1}, \mathbf{x}_0, t\right)$$

is well defined and computable

Single Shooting - Example: swing-up of a pendulum

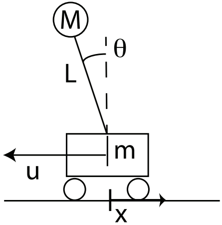
OCP

$$\min_{u_0,\dots,u_{N-1}} \quad \sum_{k=0}^{N} u_k^2$$

$$\text{s.t.} \quad \dot{\mathbf{x}} = \mathbf{F}(\mathbf{x}, u_k), \quad \forall t \in [t_k, t_{k+1}]$$

$$-20 \leq u_k \leq 20$$

$$\mathbf{x}(0) \equiv \mathbf{x}_0 = \begin{bmatrix} 0 & \pi & 0 & 0 \end{bmatrix}, \quad \mathbf{x}(t_f) = 0$$



Integrator function provides $\mathbf{x}(t) = \mathbf{f}(\mathbf{u}_0, \dots, \mathbf{u}_{N-1}, \mathbf{x}_0, t)$

NLP from single shooting

$$\min_{u_0,\dots,u_{N-1}} \quad \sum_{k=0}^{N-1} u_k^2$$

$$\text{s.t.} \quad -20 \leq u_k \leq 20$$

$$\mathbf{f}(u_0, \dots, u_{N-1}, \mathbf{x}_0, t_f) = 0$$

An integrator is a function in the most rigorous sense of the term. E.g.

$$\frac{\partial \mathbf{f}}{\partial \mathbf{u}_k}(\mathbf{u}_0, \dots, \mathbf{u}_{N-1}, \mathbf{x}_0, t)$$

is well defined and computable

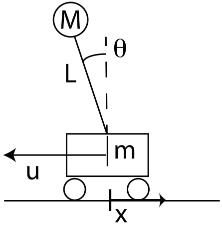
Single Shooting - Example: swing-up of a pendulum

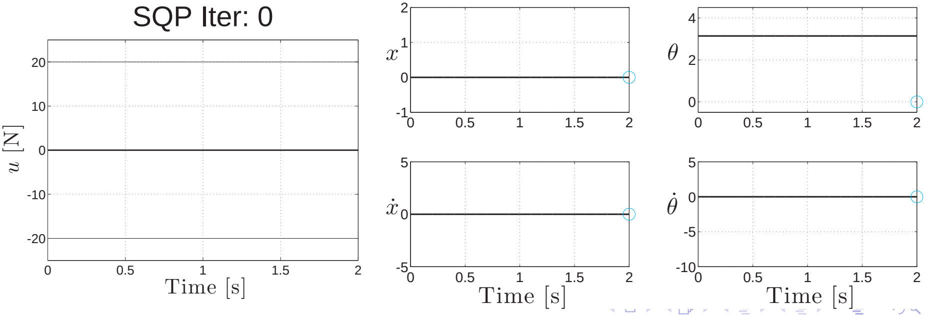
NLP from single shooting

$$\min_{u_0,\dots,u_{N-1}} \quad \sum_{k=0}^{N-1} u_k^2$$

$$\text{s.t.} \quad -20 \leq u_k \leq 20$$

$$\mathbf{f}\left(u_0, \dots, u_{N-1}, \mathbf{x}_0, t_{\mathrm{f}}\right) = 0$$

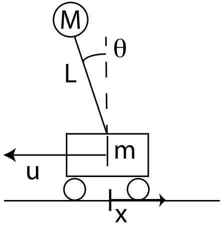


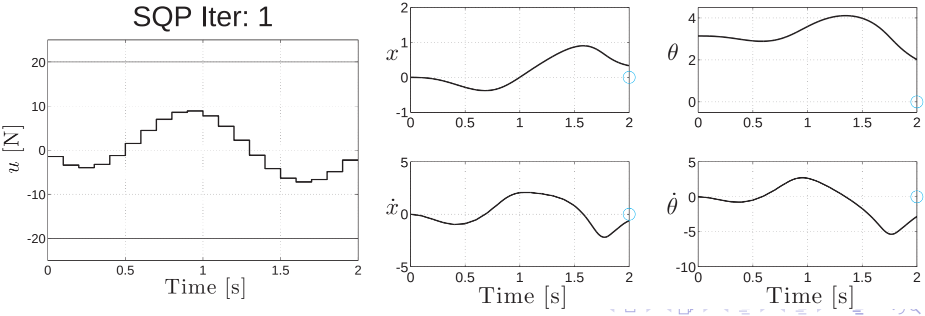


Single Shooting - Example: swing-up of a pendulum

NLP from single shooting

$$\begin{aligned}
\min_{u_0,\dots,u_{N-1}} \quad & \sum_{k=0}^{N-1} u_k^2 \\
\text{s.t.} \quad & -20 \leq u_k \leq 20 \\
& \mathbf{f}\left(u_0, \dots, u_{N-1}, \mathbf{x}_0, t_\mathrm{f}\right) = 0
\end{aligned}$$





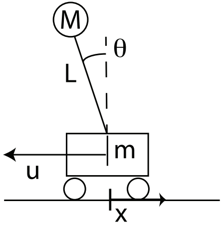
Single Shooting - Example: swing-up of a pendulum

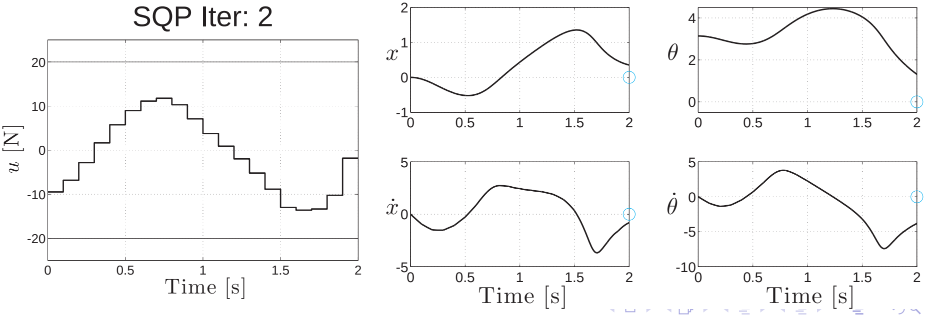
NLP from single shooting

$$\min_{u_0,\ldots,u_{N-1}} \quad \sum_{k=0}^{N-1} u_k^2$$

$$\text{s.t.} \quad -20 \leq u_k \leq 20$$

$$\mathbf{f}\left(u_0, ..., u_{N-1}, \mathbf{x}_0, t_{\mathrm{f}}\right) = 0$$





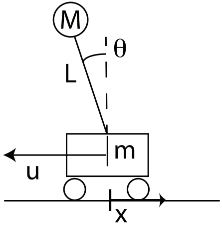
Single Shooting - Example: swing-up of a pendulum

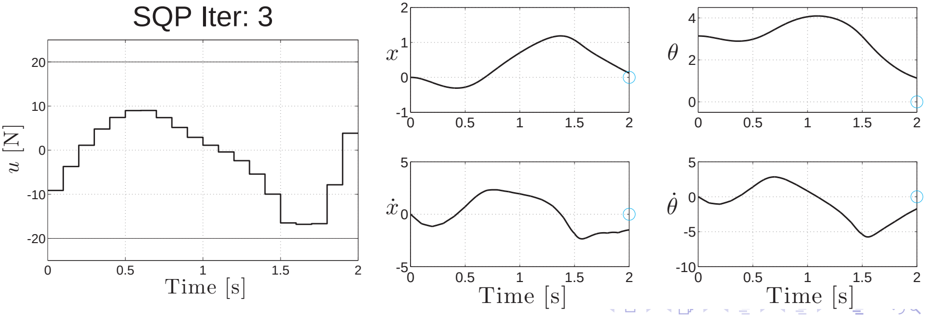
NLP from single shooting

$$\min_{u_0,\ldots,u_{N-1}} \quad \sum_{k=0}^{N-1} u_k^2$$

$$\text{s.t.} \quad -20 \leq u_k \leq 20$$

$$\mathbf{f}\left(u_0, \ldots, u_{N-1}, \mathbf{x}_0, t_\mathrm{f}\right) = 0$$

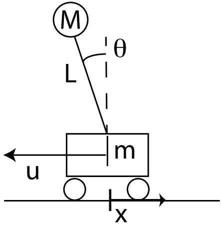


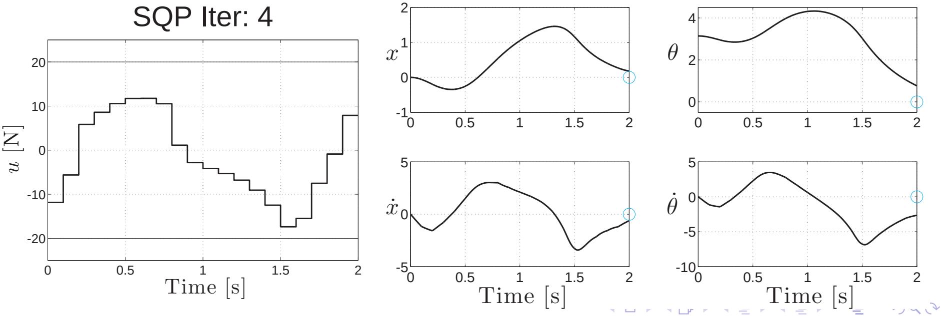


Single Shooting - Example: swing-up of a pendulum

NLP from single shooting

$$\min_{u_0,\dots,u_{N-1}} \quad \sum_{k=0}^{N-1} u_k^2$$
$$\text{s.t.} \quad -20 \leq u_k \leq 20$$
$$\mathbf{f}\left(u_0, \dots, u_{N-1}, \mathbf{x}_0, t_\mathrm{f}\right) = 0$$





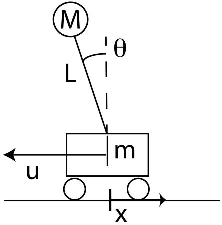
Single Shooting - Example: swing-up of a pendulum

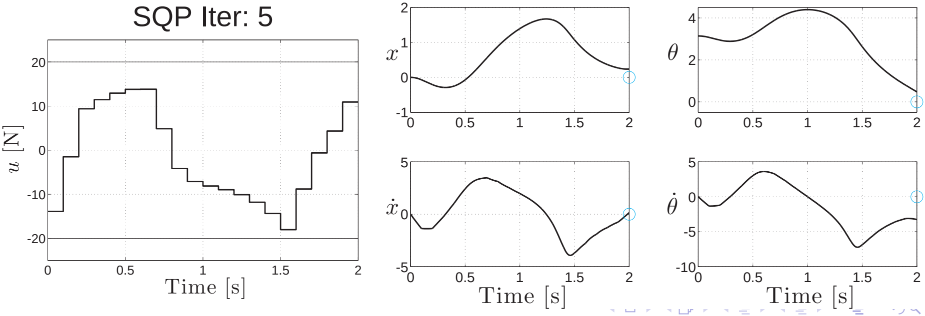
NLP from single shooting

$$\min_{u_0,\dots,u_{N-1}} \quad \sum_{k=0}^{N-1} u_k^2$$

$$\text{s.t.} \quad -20 \leq u_k \leq 20$$

$$\mathbf{f}\left(u_0, \dots, u_{N-1}, \mathbf{x}_0, t_\mathrm{f}\right) = 0$$

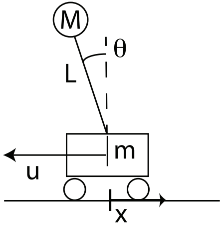


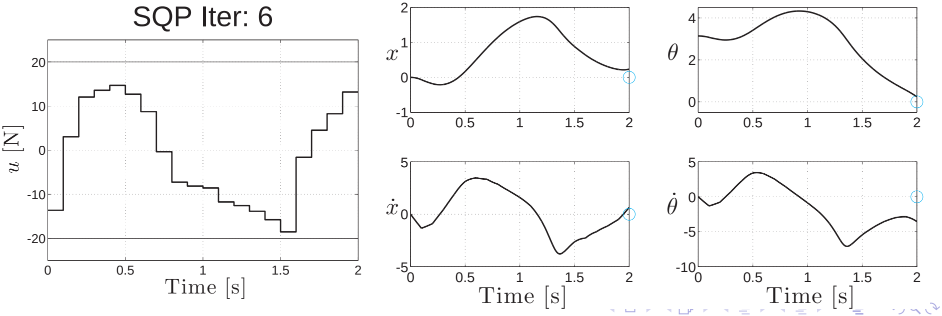


Single Shooting - Example: swing-up of a pendulum

NLP from single shooting

$$\min_{u_0,\dots,u_{N-1}} \quad \sum_{k=0}^{N-1} u_k^2$$
$$\text{s.t.} \quad -20 \le u_k \le 20$$
$$\mathbf{f}\left(u_0, \dots, u_{N-1}, \mathbf{x}_0, t_{\mathrm{f}}\right) = 0$$

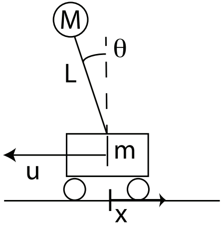


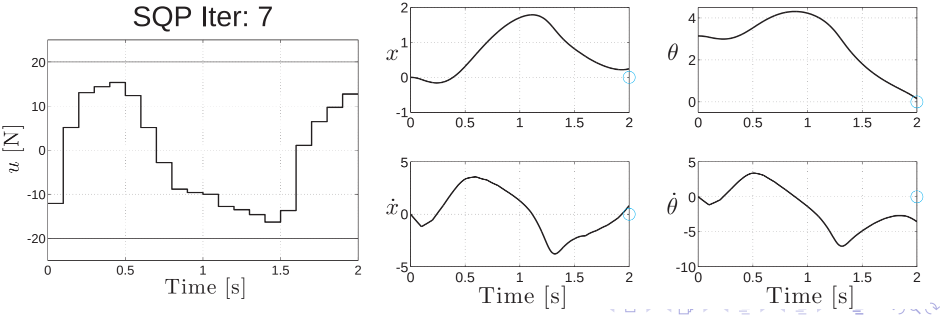


Single Shooting - Example: swing-up of a pendulum

NLP from single shooting

$$\begin{aligned} \min_{u_0,\ldots,u_{N-1}} \quad & \sum_{k=0}^{N-1} u_k^2 \\ \text{s.t.} \quad & -20 \leq u_k \leq 20 \\ & \mathbf{f}\left(u_0, \ldots, u_{N-1}, \mathbf{x}_0, t_\mathrm{f}\right) = 0 \end{aligned}$$

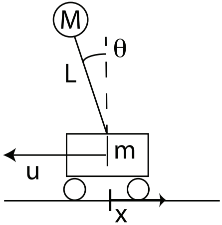


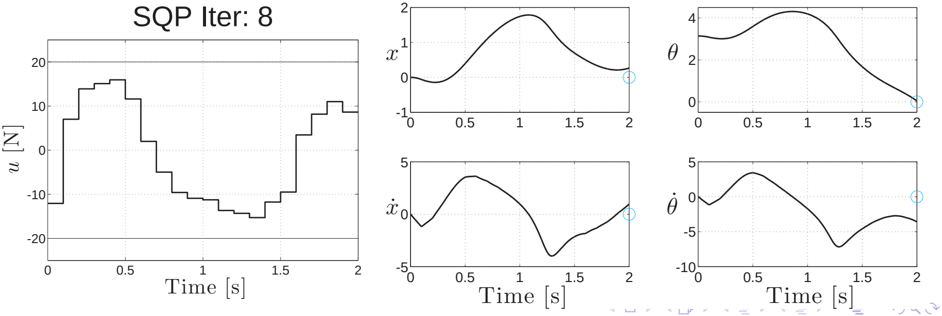


Single Shooting - Example: swing-up of a pendulum

NLP from single shooting

$$
\begin{aligned}
\min_{u_0,\dots,u_{N-1}} \quad & \sum_{k=0}^{N-1} u_k^2 \\
\text{s.t.} \quad & -20 \le u_k \le 20 \\
& \mathbf{f}\left(u_0,\dots,u_{N-1},\mathbf{x}_0,t_\mathrm{f}\right) = 0
\end{aligned}
$$

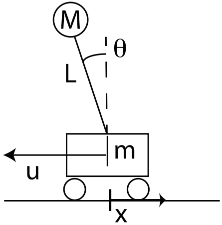


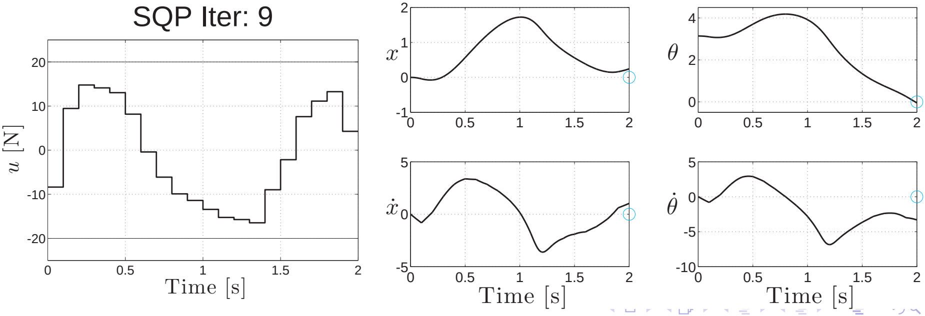


Single Shooting - Example: swing-up of a pendulum

NLP from single shooting

$$
\begin{aligned}
\min_{u_0,\dots,u_{N-1}} \quad & \sum_{k=0}^{N-1} u_k^2 \\
\text{s.t.} \quad & -20 \leq u_k \leq 20 \\
& \mathbf{f}\left(u_0, \dots, u_{N-1}, \mathbf{x}_0, t_{\mathrm{f}}\right) = 0
\end{aligned}
$$

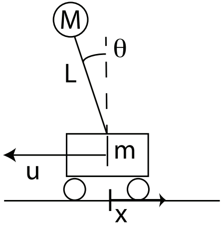


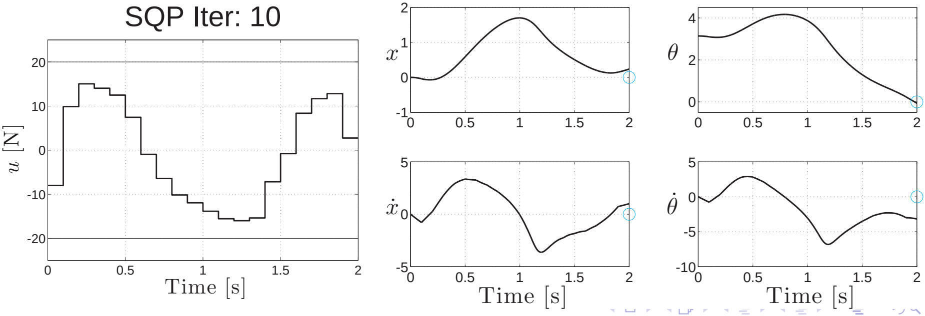


Single Shooting - Example: swing-up of a pendulum

NLP from single shooting

$$\min_{u_0,\dots,u_{N-1}} \quad \sum_{k=0}^{N-1} u_k^2$$
$$\text{s.t.} \quad -20 \leq u_k \leq 20$$
$$\mathbf{f}\left(u_0, \dots, u_{N-1}, \mathbf{x}_0, t_\mathrm{f}\right) = 0$$

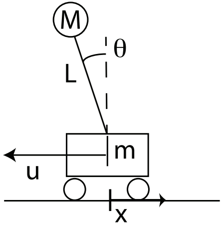


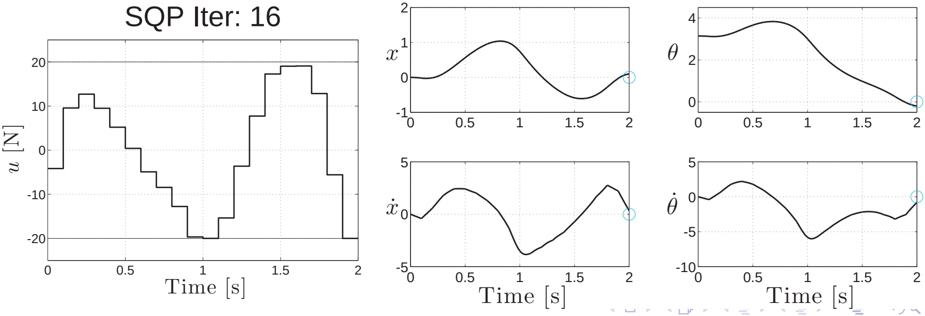


Single Shooting - Example: swing-up of a pendulum

NLP from single shooting

$$\min_{u_0,\ldots,u_{N-1}} \quad \sum_{k=0}^{N-1} u_k^2$$
$$\text{s.t.} \quad -20 \leq u_k \leq 20$$
$$\mathbf{f}\left(u_0, \ldots, u_{N-1}, \mathbf{x}_0, t_\mathrm{f}\right) = 0$$

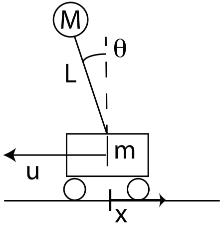


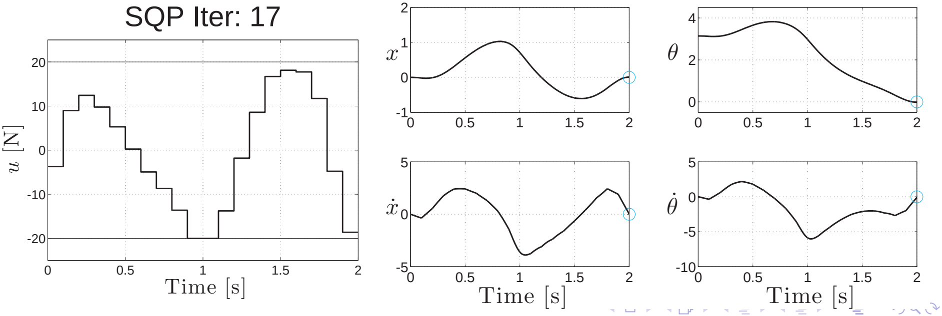


Single Shooting - Example: swing-up of a pendulum

NLP from single shooting

$$\begin{aligned} \min_{u_0,\dots,u_{N-1}} \quad & \sum_{k=0}^{N-1} u_k^2 \\ \text{s.t.} \quad & -20 \le u_k \le 20 \\ & \mathbf{f}\left(u_0, \dots, u_{N-1}, \mathbf{x}_0, t_{\mathrm{f}}\right) = 0 \end{aligned}$$

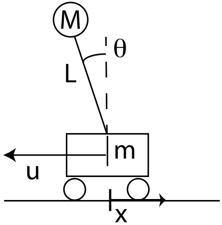


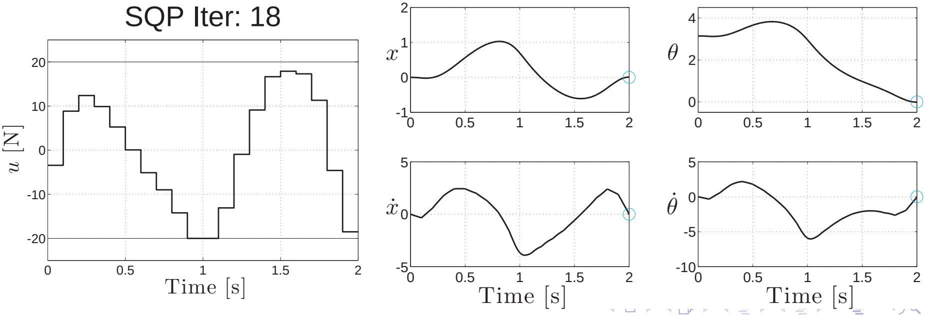


Single Shooting - Example: swing-up of a pendulum

NLP from single shooting

$$\begin{aligned}\min_{u_0,\ldots,u_{N-1}} \quad & \sum_{k=0}^{N-1} u_k^2 \\ \text{s.t.} \quad & -20 \leq u_k \leq 20 \\ & \mathbf{f}\left(u_0, \ldots, u_{N-1}, \mathbf{x}_0, t_{\mathrm{f}}\right) = 0\end{aligned}$$

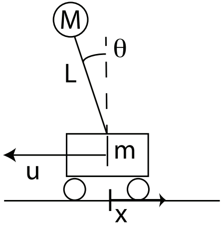


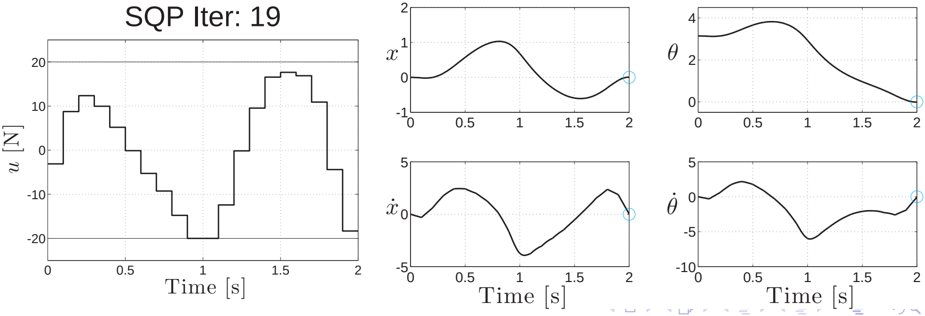


Single Shooting - Example: swing-up of a pendulum

NLP from single shooting

$$
\begin{aligned}
\min_{u_0,\dots,u_{N-1}} \quad & \sum_{k=0}^{N-1} u_k^2 \\
\text{s.t.} \quad & -20 \le u_k \le 20 \\
& \mathbf{f}\left(u_0, \dots, u_{N-1}, \mathbf{x}_0, t_\mathrm{f}\right) = 0
\end{aligned}
$$

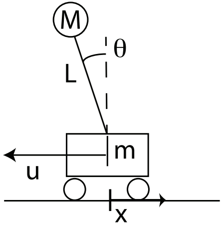


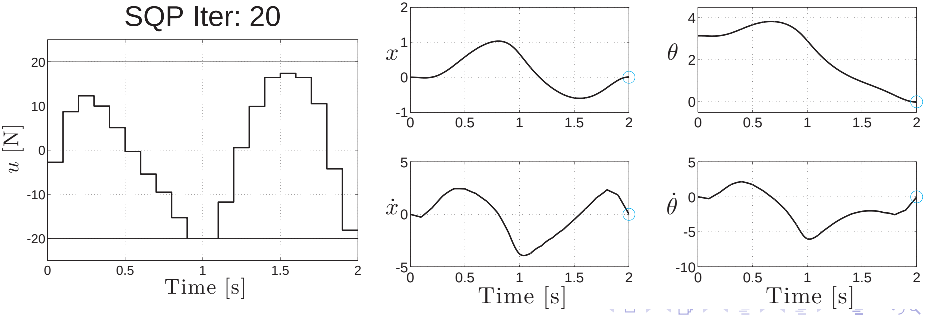


Single Shooting - Example: swing-up of a pendulum

NLP from single shooting

$$
\begin{aligned}
\min_{u_0,\dots,u_{N-1}} \quad & \sum_{k=0}^{N-1} u_k^2 \\
\text{s.t.} \quad & -20 \leq u_k \leq 20 \\
& \mathbf{f}\left(u_0, \dots, u_{N-1}, \mathbf{x}_0, t_\mathrm{f}\right) = 0
\end{aligned}
$$





Single-Shooting - Propagating continuous dynamics

Nonlinearity propagation: integrator function $\mathbf{f}$

$$\mathbf{f}\left(\mathbf{u}_0, ..., \mathbf{u}_{N-1}, \mathbf{x}_0, t\right) : \mathbf{u}_0, ..., \mathbf{u}_{N-1}, \mathbf{x}_0, t \longmapsto \mathbf{x}(t)$$

tends to become highly nonlinear for large t.

Single-Shooting - Propagating continuous dynamics

Nonlinearity propagation: integrator function $\mathbf{f}$

$$\mathbf{f}\left(\mathbf{u}_0, ..., \mathbf{u}_{N-1}, \mathbf{x}_0, t\right) : \mathbf{u}_0, ..., \mathbf{u}_{N-1}, \mathbf{x}_0, t \longmapsto \mathbf{x}(t)$$

tends to become highly nonlinear for large t.

Example

$$\begin{aligned}\dot{x} &= 10\left(y - x\right)\\ \dot{y} &= x\left(u - z\right) - y\\ \dot{z} &= xy - 3z\end{aligned}$$

State: $\mathbf{x}(.) = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$

Input: $u(.)$

Single-Shooting - Propagating continuous dynamics

Nonlinearity propagation: integrator function $\mathbf{f}$

$$\mathbf{f}\left(\mathbf{u}_0, ..., \mathbf{u}_{N-1}, \mathbf{x}_0, t\right) : \mathbf{u}_0, ..., \mathbf{u}_{N-1}, \mathbf{x}_0, t \longmapsto \mathbf{x}(t)$$

tends to become highly nonlinear for large t.

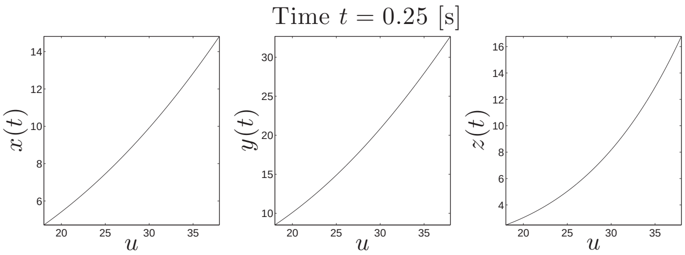
Example

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

State: $\mathbf{x}(.) = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$

Input: $u(.)$

States as a function of u (constant) at time t, for a fixed $\mathbf{x}_0$



Single-Shooting - Propagating continuous dynamics

Nonlinearity propagation: integrator function $\mathbf{f}$

$$\mathbf{f}\left(\mathbf{u}_0, ..., \mathbf{u}_{N-1}, \mathbf{x}_0, t\right) : \mathbf{u}_0, ..., \mathbf{u}_{N-1}, \mathbf{x}_0, t \longmapsto \mathbf{x}(t)$$

tends to become highly nonlinear for large t.

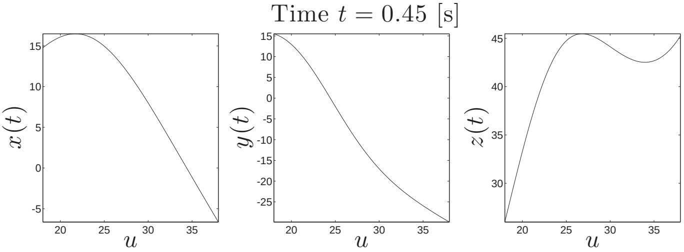
Example

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

State: $\mathbf{x}(.) = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$

Input: $u(.)$

States as a function of u (constant) at time t, for a fixed $\mathbf{x}_0$



Single-Shooting - Propagating continuous dynamics

Nonlinearity propagation: integrator function $\mathbf{f}$

$$\mathbf{f}\left(\mathbf{u}_0, ..., \mathbf{u}_{N-1}, \mathbf{x}_0, t\right) : \mathbf{u}_0, ..., \mathbf{u}_{N-1}, \mathbf{x}_0, t \longmapsto \mathbf{x}(t)$$

tends to become highly nonlinear for large t.

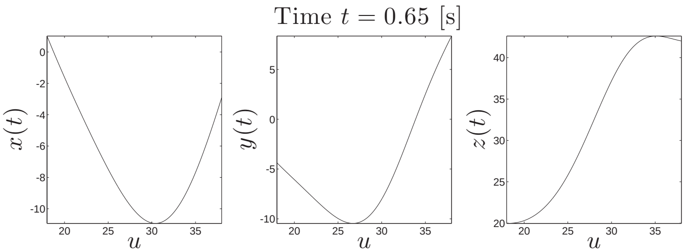
Example

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

State: $\mathbf{x}(.) = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$

Input: $u(.)$

States as a function of u (constant) at time t, for a fixed $\mathbf{x}_0$



Single-Shooting - Propagating continuous dynamics

Nonlinearity propagation: integrator function $\mathbf{f}$

$$\mathbf{f}\left(\mathbf{u}_0, ..., \mathbf{u}_{N-1}, \mathbf{x}_0, t\right) : \mathbf{u}_0, ..., \mathbf{u}_{N-1}, \mathbf{x}_0, t \longmapsto \mathbf{x}(t)$$

tends to become highly nonlinear for large t.

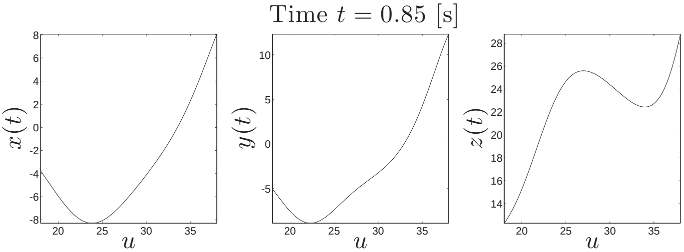
Example

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

State: $\mathbf{x}(.) = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$

Input: $u(.)$

States as a function of u (constant) at time t, for a fixed $\mathbf{x}_0$



Single-Shooting - Propagating continuous dynamics

Nonlinearity propagation: integrator function $\mathbf{f}$

$$\mathbf{f}\left(\mathbf{u}_0, ..., \mathbf{u}_{N-1}, \mathbf{x}_0, t\right) : \mathbf{u}_0, ..., \mathbf{u}_{N-1}, \mathbf{x}_0, t \longmapsto \mathbf{x}(t)$$

tends to become highly nonlinear for large t.

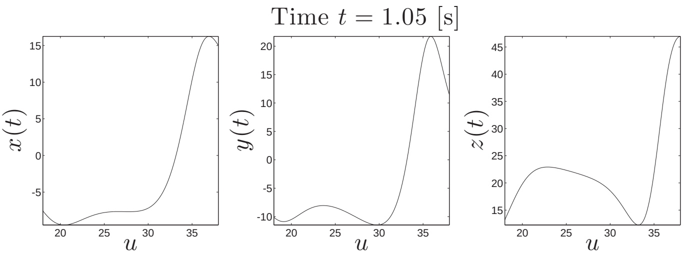
Example

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

State: $\mathbf{x}(.) = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$

Input: $u(.)$

States as a function of u (constant) at time t, for a fixed $\mathbf{x}_0$



Single-Shooting - Propagating continuous dynamics

Nonlinearity propagation: integrator function $\mathbf{f}$

$$\mathbf{f}\left(\mathbf{u}_0, ..., \mathbf{u}_{N-1}, \mathbf{x}_0, t\right) : \mathbf{u}_0, ..., \mathbf{u}_{N-1}, \mathbf{x}_0, t \longmapsto \mathbf{x}(t)$$

tends to become highly nonlinear for large t.

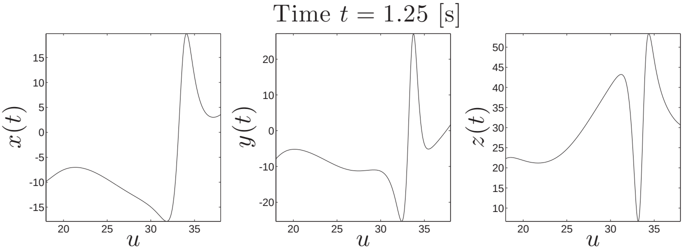
Example

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

State: $\mathbf{x}(.) = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$

Input: $u(.)$

States as a function of u (constant) at time t, for a fixed $\mathbf{x}_0$



Single-Shooting - Propagating continuous dynamics

Nonlinearity propagation: integrator function $\mathbf{f}$

$$\mathbf{f}\left(\mathbf{u}_0, ..., \mathbf{u}_{N-1}, \mathbf{x}_0, t\right) : \mathbf{u}_0, ..., \mathbf{u}_{N-1}, \mathbf{x}_0, t \longmapsto \mathbf{x}(t)$$

tends to become highly nonlinear for large t.

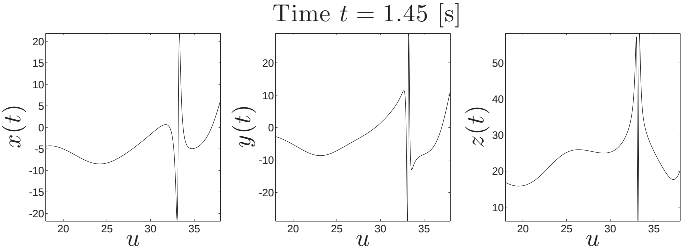
Example

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

State: $\mathbf{x}(.) = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$

Input: $u(.)$

States as a function of u (constant) at time t, for a fixed $\mathbf{x}_0$



Single-Shooting - Propagating continuous dynamics

Nonlinearity propagation: integrator function $\mathbf{f}$

$$\mathbf{f}\left(\mathbf{u}_0, ..., \mathbf{u}_{N-1}, \mathbf{x}_0, t\right) : \mathbf{u}_0, ..., \mathbf{u}_{N-1}, \mathbf{x}_0, t \longmapsto \mathbf{x}(t)$$

tends to become highly nonlinear for large t.

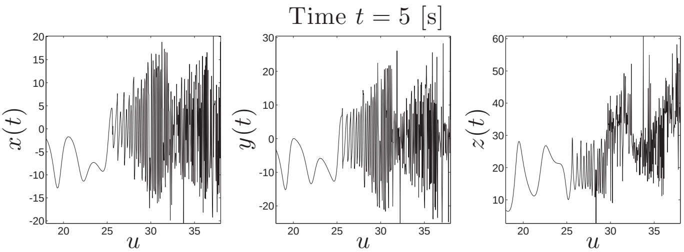
Example

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

State: $\mathbf{x}(.) = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$

Input: $u(.)$

States as a function of u (constant) at time t, for a fixed $\mathbf{x}_0$



Single-Shooting - Propagating continuous dynamics

Nonlinearity propagation: integrator function $\mathbf{f}$

$$\mathbf{f}\left(\mathbf{u}_0, ..., \mathbf{u}_{N-1}, \mathbf{x}_0, t\right) : \mathbf{u}_0, ..., \mathbf{u}_{N-1}, \mathbf{x}_0, t \longmapsto \mathbf{x}(t)$$

tends to become highly nonlinear for large t.

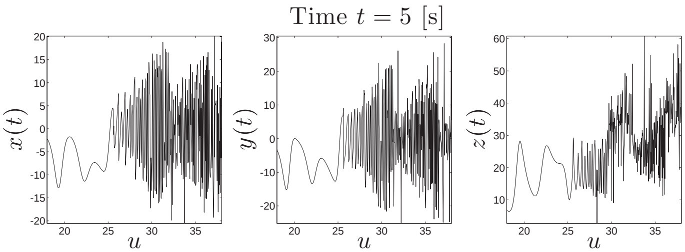
Example

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

State: $\mathbf{x}(.) = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$

Input: $u(.)$

States as a function of u (constant) at time t, for a fixed $\mathbf{x}_0$



What's this crazy system btw ?!?

Single-Shooting - Propagating continuous dynamics

Nonlinearity propagation: integrator function $\mathbf{f}$

$$\mathbf{f}\left(\mathbf{u}_0, ..., \mathbf{u}_{N-1}, \mathbf{x}_0, t\right) : \mathbf{u}_0, ..., \mathbf{u}_{N-1}, \mathbf{x}_0, t \longmapsto \mathbf{x}(t)$$

tends to become highly nonlinear for large t.

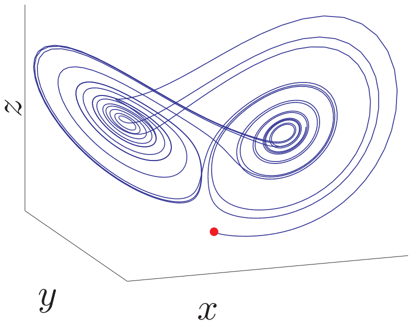
Example

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

State: $\mathbf{x}(.) = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$

Input: $u(.)$

Lorentz attractor ($u = 28$) stable but chaotic



Single-Shooting - Propagating continuous dynamics

Nonlinearity propagation: integrator function $\mathbf{f}$

$$\mathbf{f}\left(\mathbf{u}_0, \ldots, \mathbf{u}_{N-1}, \mathbf{x}_0, t\right) : \mathbf{u}_0, \ldots, \mathbf{u}_{N-1}, \mathbf{x}_0, t \longmapsto \mathbf{x}(t)$$

tends to become highly nonlinear for large t.

Example

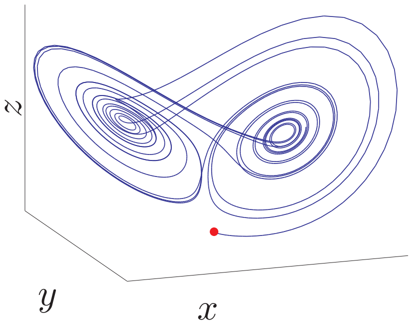
$$\dot{x} = 10\,(y - x)$$
$$\dot{y} = x\,(u - z) - y$$
$$\dot{z} = xy - 3z$$

State: $\mathbf{x}(.) = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$

Input: $u(.)$

In optimal control, **don't** simulate a nonlinear/unstable system over a *long* time horizon

Lorentz attractor ($u = 28$) stable but chaotic



Nonlinearity of the integrator function - Some formalism

Consider the dynamics $\dot{\mathbf{x}} = \mathbf{F}(\mathbf{x})$ and the corresponding **integrator function** $\mathbf{f}(\mathbf{x}, t)$ that maps the initial conditions $\mathbf{x} \in \mathbb{R}^n$ onto a trajectory $\mathbf{x}(t) \in \mathbb{R}^n$, i.e.

$$\mathbf{f}(\mathbf{x}, t) \,:\, \mathbf{x}, t \mapsto \mathbf{x}(t)$$

Nonlinearity of the integrator function - Some formalism

Consider the dynamics $\dot{\mathbf{x}} = \mathbf{F}(\mathbf{x})$ and the corresponding **integrator function** $\mathbf{f}(\mathbf{x}, t)$ that maps the initial conditions $\mathbf{x} \in \mathbb{R}^n$ onto a trajectory $\mathbf{x}(t) \in \mathbb{R}^n$, i.e.

$$\mathbf{f}(\mathbf{x}, t) \; : \; \mathbf{x}, t \; \mapsto \; \mathbf{x}(t)$$

Remarks:

- Here we omit the input, but $\dot{\mathbf{x}} = \mathbf{F}(\mathbf{x})$ is still general. Indeed, one can rewrite $\dot{\mathbf{x}} = \mathbf{F}(\mathbf{x}, \mathbf{u})$ (for $\mathbf{u}$ constant) as:

$$\mathbf{z} = \begin{bmatrix} \mathbf{x} \\ \mathbf{u} \end{bmatrix}, \qquad \dot{\mathbf{z}} = \begin{bmatrix} \mathbf{F}(\mathbf{z}) \\ 0 \end{bmatrix}$$

Nonlinearity of the integrator function - Some formalism

Consider the dynamics $\dot{\mathbf{x}} = \mathbf{F}(\mathbf{x})$ and the corresponding **integrator function** $\mathbf{f}(\mathbf{x}, t)$ that maps the initial conditions $\mathbf{x} \in \mathbb{R}^n$ onto a trajectory $\mathbf{x}(t) \in \mathbb{R}^n$, i.e.

$$\mathbf{f}(\mathbf{x}, t) \,:\, \mathbf{x}, t \mapsto \mathbf{x}(t)$$

Remarks:

- Here we omit the input, but $\dot{\mathbf{x}} = \mathbf{F}(\mathbf{x})$ is still general. Indeed, one can rewrite $\dot{\mathbf{x}} = \mathbf{F}(\mathbf{x}, \mathbf{u})$ (for $\mathbf{u}$ constant) as:

$$\mathbf{z} = \begin{bmatrix} \mathbf{x} \\ \mathbf{u} \end{bmatrix}, \qquad \dot{\mathbf{z}} = \begin{bmatrix} \mathbf{F}(\mathbf{z}) \\ 0 \end{bmatrix}$$

- How to measure the nonlinearity of $\mathbf{f}(\mathbf{x}, t)$ in $\mathbf{x}$?

Nonlinearity of the integrator function - Some formalism

Consider the dynamics $\dot{\mathbf{x}} = \mathbf{F}(\mathbf{x})$ and the corresponding **integrator function** $\mathbf{f}(\mathbf{x}, t)$ that maps the initial conditions $\mathbf{x} \in \mathbb{R}^n$ onto a trajectory $\mathbf{x}(t) \in \mathbb{R}^n$, i.e.

$$\mathbf{f}(\mathbf{x}, t) \,:\, \mathbf{x}, t \mapsto \mathbf{x}(t)$$

Remarks:

- Here we omit the input, but $\dot{\mathbf{x}} = \mathbf{F}(\mathbf{x})$ is still general. Indeed, one can rewrite $\dot{\mathbf{x}} = \mathbf{F}(\mathbf{x}, \mathbf{u})$ (for $\mathbf{u}$ constant) as:

$$\mathbf{z} = \begin{bmatrix} \mathbf{x} \\ \mathbf{u} \end{bmatrix}, \qquad \dot{\mathbf{z}} = \begin{bmatrix} \mathbf{F}(\mathbf{z}) \\ 0 \end{bmatrix}$$

- How to measure the nonlinearity of $\mathbf{f}(\mathbf{x}, t)$ in $\mathbf{x}$? What about:

$$\left\| \frac{\partial \mathbf{f}(\mathbf{x}, t)}{\partial \mathbf{x}} - \frac{\partial \mathbf{f}(\mathbf{y}, t)}{\partial \mathbf{y}} \right\| \leq L \left\| \mathbf{x} - \mathbf{y} \right\|$$

  If $\mathbf{f}(\mathbf{x}, t)$ is affine in $\mathbf{x}$, i.e. $\mathbf{f}(\mathbf{x}, t) = A(t)\mathbf{x} + \mathbf{b}(t)$, then $L = 0$... If L large, then $\mathbf{f}(\mathbf{x}, t)$ is very nonlinear...

Nonlinearity of the integrator function - Some formalism

Consider the dynamics $\dot{\mathbf{x}} = \mathbf{F}(\mathbf{x})$ and the corresponding **integrator function** $\mathbf{f}(\mathbf{x}, t)$ that maps the initial conditions $\mathbf{x} \in \mathbb{R}^n$ onto a trajectory $\mathbf{x}(t) \in \mathbb{R}^n$, i.e.

$$\mathbf{f}(\mathbf{x}, t) \; : \; \mathbf{x}, t \; \mapsto \; \mathbf{x}(t)$$

Proposition

Assume:

- Lipschitz ODE: $\left\|\mathbf{F}(\mathbf{x}) - \mathbf{F}(\mathbf{y})\right\| \leq L_0 \left\|\mathbf{x} - \mathbf{y}\right\|$
- Lipschitz sensitivity of the dynamics: $\left\| \frac{\partial \mathbf{F}(\mathbf{x})}{\partial \mathbf{x}} - \frac{\partial \mathbf{F}(\mathbf{y})}{\partial \mathbf{y}} \right\| \leq L_1 \left\|\mathbf{x} - \mathbf{y}\right\|$
- Bounded sensitivity of the dynamics: $\left\| \frac{\partial \mathbf{F}(\mathbf{x})}{\partial \mathbf{x}} \right\| \leq \beta$ for all $\mathbf{x}$

Nonlinearity of the integrator function - Some formalism

Consider the dynamics $\dot{\mathbf{x}} = \mathbf{F}(\mathbf{x})$ and the corresponding **integrator function** $\mathbf{f}(\mathbf{x}, t)$ that maps the initial conditions $\mathbf{x} \in \mathbb{R}^n$ onto a trajectory $\mathbf{x}(t) \in \mathbb{R}^n$, i.e.

$$\mathbf{f}(\mathbf{x}, t) \; : \; \mathbf{x}, t \mapsto \mathbf{x}(t)$$

Proposition

Assume:

- Lipschitz ODE: $\|\mathbf{F}(\mathbf{x}) - \mathbf{F}(\mathbf{y})\| \leq L_0 \|\mathbf{x} - \mathbf{y}\|$
- Lipschitz sensitivity of the dynamics: $\left\| \frac{\partial \mathbf{F}(\mathbf{x})}{\partial \mathbf{x}} - \frac{\partial \mathbf{F}(\mathbf{y})}{\partial \mathbf{y}} \right\| \leq L_1 \|\mathbf{x} - \mathbf{y}\|$
- Bounded sensitivity of the dynamics: $\left\| \frac{\partial \mathbf{F}(\mathbf{x})}{\partial \mathbf{x}} \right\| \leq \beta$ for all $\mathbf{x}$

Then the following holds:

- **Bounded divergence** of the solutions: $\|\mathbf{f}(\mathbf{x}, t) - \mathbf{f}(\mathbf{y}, t)\| \leq e^{L_0 t} \|\mathbf{x} - \mathbf{y}\|$

Nonlinearity of the integrator function - Some formalism

Consider the dynamics $\dot{\mathbf{x}} = \mathbf{F}(\mathbf{x})$ and the corresponding **integrator function** $\mathbf{f}(\mathbf{x}, t)$ that maps the initial conditions $\mathbf{x} \in \mathbb{R}^n$ onto a trajectory $\mathbf{x}(t) \in \mathbb{R}^n$, i.e.

$$\mathbf{f}(\mathbf{x}, t) \; : \; \mathbf{x}, t \; \mapsto \; \mathbf{x}(t)$$

Proposition

Assume:

- Lipschitz ODE: $\|\mathbf{F}(\mathbf{x}) - \mathbf{F}(\mathbf{y})\| \leq L_0 \|\mathbf{x} - \mathbf{y}\|$
- Lipschitz sensitivity of the dynamics: $\left\| \frac{\partial \mathbf{F}(\mathbf{x})}{\partial \mathbf{x}} - \frac{\partial \mathbf{F}(\mathbf{y})}{\partial \mathbf{y}} \right\| \leq L_1 \|\mathbf{x} - \mathbf{y}\|$
- Bounded sensitivity of the dynamics: $\left\| \frac{\partial \mathbf{F}(\mathbf{x})}{\partial \mathbf{x}} \right\| \leq \beta$ for all $\mathbf{x}$

Then the following holds:

- **Bounded divergence** of the solutions: $\|\mathbf{f}(\mathbf{x}, t) - \mathbf{f}(\mathbf{y}, t)\| \leq e^{L_0 t} \|\mathbf{x} - \mathbf{y}\|$
- **Bounded sensitivity**: $\left\| \frac{\partial \mathbf{f}(\mathbf{x}, t)}{\partial \mathbf{x}} \right\| \leq e^{\beta t}$

Nonlinearity of the integrator function - Some formalism

Consider the dynamics $\dot{\mathbf{x}} = \mathbf{F}(\mathbf{x})$ and the corresponding **integrator function** $\mathbf{f}(\mathbf{x}, t)$ that maps the initial conditions $\mathbf{x} \in \mathbb{R}^n$ onto a trajectory $\mathbf{x}(t) \in \mathbb{R}^n$, i.e.

$$\mathbf{f}(\mathbf{x}, t) \, : \, \mathbf{x}, t \mapsto \mathbf{x}(t)$$

Proposition

Assume:

- Lipschitz ODE: $\|\mathbf{F}(\mathbf{x}) - \mathbf{F}(\mathbf{y})\| \leq L_0 \|\mathbf{x} - \mathbf{y}\|$
- Lipschitz sensitivity of the dynamics: $\left\| \frac{\partial \mathbf{F}(\mathbf{x})}{\partial \mathbf{x}} - \frac{\partial \mathbf{F}(\mathbf{y})}{\partial \mathbf{y}} \right\| \leq L_1 \|\mathbf{x} - \mathbf{y}\|$
- Bounded sensitivity of the dynamics: $\left\| \frac{\partial \mathbf{F}(\mathbf{x})}{\partial \mathbf{x}} \right\| \leq \beta$ for all $\mathbf{x}$

Then the following holds:

- **Bounded divergence** of the solutions: $\|\mathbf{f}(\mathbf{x}, t) - \mathbf{f}(\mathbf{y}, t)\| \leq e^{L_0 t} \|\mathbf{x} - \mathbf{y}\|$
- **Bounded sensitivity**: $\left\| \frac{\partial \mathbf{f}(\mathbf{x},t)}{\partial \mathbf{x}} \right\| \leq e^{\beta t}$
- **Bounded nonlinearity**: $\left\| \frac{\partial \mathbf{f}(\mathbf{x},t)}{\partial \mathbf{x}} - \frac{\partial \mathbf{f}(\mathbf{y},t)}{\partial \mathbf{y}} \right\| \leq \frac{L_1}{L_0} e^{\beta t} \left(e^{L_0 t} - 1 \right) \|\mathbf{x} - \mathbf{y}\|$

Proof

Two useful mathematical tricks:

- The inequality
$$\frac{\mathrm{d}}{\mathrm{d}t}\left\|\mathbf{a}\right\| \leq \left\|\dot{\mathbf{a}}\right\|$$
holds on any vector space equipped with a (almost everywhere) differentiable norm $\|.\|$ (e.g. 2-norm, Frobenius).

Proof

Two useful mathematical tricks:

- The inequality
$$\frac{\mathrm{d}}{\mathrm{d}t}\|\mathbf{a}\| \leq \|\dot{\mathbf{a}}\|$$
holds on any vector space equipped with a (almost everywhere) differentiable norm $\|.\|$ (e.g. 2-norm, Frobenius).

- **Gronwall Lemma**: if
$$\frac{\mathrm{d}}{\mathrm{d}t}\|\mathbf{a}(t)\| \leq \alpha(t)\|\mathbf{a}(t)\| + \beta(t)$$

Proof

Two useful mathematical tricks:

- The inequality
$$\frac{\mathrm{d}}{\mathrm{d}t}\left\|\mathbf{a}\right\| \leq \left\|\dot{\mathbf{a}}\right\|$$
holds on any vector space equipped with a (almost everywhere) differentiable norm $\|.\|$ (e.g. 2-norm, Frobenius).

- **Gronwall Lemma**: if
$$\frac{\mathrm{d}}{\mathrm{d}t}\left\|\mathbf{a}(t)\right\| \leq \alpha(t)\left\|\mathbf{a}(t)\right\| + \beta(t)$$
then
$$\left\|\mathbf{a}(t)\right\| \leq \left\|\mathbf{a}(0)\right\| e^{\int_0^t \alpha(s)\mathrm{d}s} + \int_0^t e^{\int_s^t \alpha(s)}\beta(s)\mathrm{d}s$$
holds for all t.

Proof

Bounded divergence of solutions: let $\mathbf{e}(\mathbf{x},\mathbf{y},t) = \mathbf{f}(\mathbf{x},t) - \mathbf{f}(\mathbf{y},t)$, then:

$$\|\dot{\mathbf{e}}\| = \|\mathbf{F}(\mathbf{f}(\mathbf{x},t)) - \mathbf{F}(\mathbf{f}(\mathbf{y},t))\| \leq L_0 \|\mathbf{f}(\mathbf{x},t) - \mathbf{f}(\mathbf{y},t)\| = L_0\|\mathbf{e}\|$$

hence

$$\frac{\mathrm{d}}{\mathrm{d}t}\|\mathbf{e}\| \leq L_0\|\mathbf{e}\|$$

Using $\mathbf{e}(\mathbf{x},\mathbf{y},0) = \mathbf{x} - \mathbf{y}$, **Gronwall Lemma** ensures that

$$\|\mathbf{e}(\mathbf{x},\mathbf{y},t)\| \leq e^{L_0 t}\|\mathbf{x} - \mathbf{y}\|$$

i.e.:

$$\|\mathbf{f}(\mathbf{x},t) - \mathbf{f}(\mathbf{y},t)\| \leq e^{L_0 t}\|\mathbf{x} - \mathbf{y}\|$$

Proof

Let us write $\frac{\partial \mathbf{f}(\mathbf{x},t)}{\partial \mathbf{x}} = A(\mathbf{x},t)$. Here we will use the fact that $A(\mathbf{x},t)$ is given by

$$\dot{A}(\mathbf{x},t) = \frac{\partial \mathbf{F}(\mathbf{f}(\mathbf{x},t))}{\partial \mathbf{x}} A(\mathbf{x},t)$$

Bounded sensitivities: we use the Frobenius matrix norm and observe that

$$\frac{\mathrm{d}}{\mathrm{d}t}\left\|A(\mathbf{x},t)\right\| \leq \left\|\dot{A}(\mathbf{x},t)\right\| = \left\|\frac{\partial \mathbf{F}(\mathbf{f}(\mathbf{x},t))}{\partial \mathbf{x}} A(\mathbf{x},t)\right\| \leq \left\|\frac{\partial \mathbf{F}(\mathbf{f}(\mathbf{x},t))}{\partial \mathbf{x}}\right\| \left\|A(\mathbf{x},t)\right\|$$

such that

$$\frac{\mathrm{d}}{\mathrm{d}t}\left\|A(\mathbf{x},t)\right\| \leq \beta \left\|A(\mathbf{x},t)\right\|$$

Using $A(\mathbf{x},0) = I$, the **Gronwall Lemma** then ensures that:

$$\left\|A(\mathbf{x},t)\right\| \leq e^{\beta t}$$

Proof

Bounded nonlinearity: let us write $E(\mathbf{x},\mathbf{y},t) = A(\mathbf{x},t) - A(\mathbf{y},t)$ then

$$\frac{\mathrm{d}}{\mathrm{d}t}\|E\| \leq \left\|\dot{E}\right\| = \left\|\frac{\partial \mathbf{F}(\mathbf{f}(\mathbf{x},t))}{\partial \mathbf{x}}A(\mathbf{x},t) - \frac{\partial \mathbf{F}(\mathbf{f}(\mathbf{y},t))}{\partial \mathbf{y}}A(\mathbf{y},t)\right\|$$

Let us use the short notation $\xi_{.} = \frac{\partial \mathbf{F}(\mathbf{f}(.,t))}{\partial .}$. Then:

$$\left\|\dot{E}\right\| = \left\|\xi_x\left(A(\mathbf{x},t) - A(\mathbf{y},t)\right) + \left(\xi_x - \xi_y\right)A(\mathbf{y},t)\right\| \leq$$
$$\|\xi_x\|\,\|A(\mathbf{x},t) - A(\mathbf{y},t)\| + \|\xi_x - \xi_y\|\,\|A(\mathbf{y},t)\| \leq \beta\,\|E\| + \|\xi_x - \xi_y\|\,e^{\beta t}$$

Then we observe that:

$$\|\xi_x - \xi_y\| \leq L_1\,\|\mathbf{f}(\mathbf{x},t) - \mathbf{f}(\mathbf{y},t)\| \leq L_1 e^{L_0 t}\|\mathbf{x}-\mathbf{y}\|$$

We use $E(\mathbf{x},\mathbf{y},0) = 0$ and the **Gronwall Lemma** to conclude that:

$$\|E(\mathbf{x},\mathbf{y},t)\| \leq \int_0^t e^{\int_s^t \beta} L_1\,\|\mathbf{x}-\mathbf{y}\|\,e^{(\beta+L_0)s}\mathrm{d}s = \int_0^t e^{\beta(t-s)}L_1\,\|\mathbf{x}-\mathbf{y}\|\,e^{(\beta+L_0)s}\mathrm{d}s =$$
$$e^{\beta t}L_1\,\|\mathbf{x}-\mathbf{y}\|\int_0^t e^{L_0 s}\mathrm{d}s = \frac{L_1}{L_0}e^{\beta t}\,\|\mathbf{x}-\mathbf{y}\|\,e^{L_0 s}\Big|_0^t = \frac{L_1}{L_0}e^{\beta t}\,\|\mathbf{x}-\mathbf{y}\|\left(e^{L_0 t}-1\right)$$

Outline

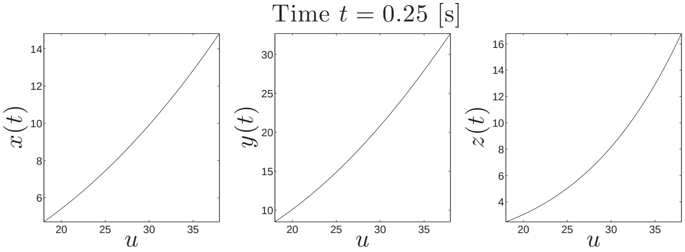
1. Single-Shooting
2. Multiple-Shooting
3. NLP from shooting methods

Multiple-Shooting - Key idea

Example

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

States as a function of u at time t

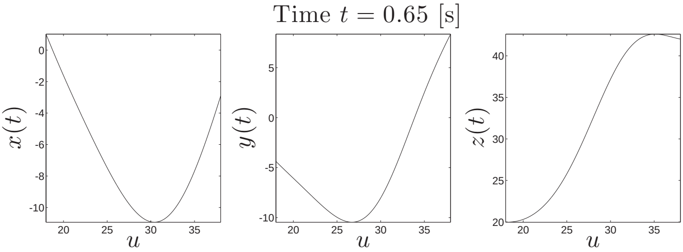


Multiple-Shooting - Key idea

Example

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

States as a function of u at time t

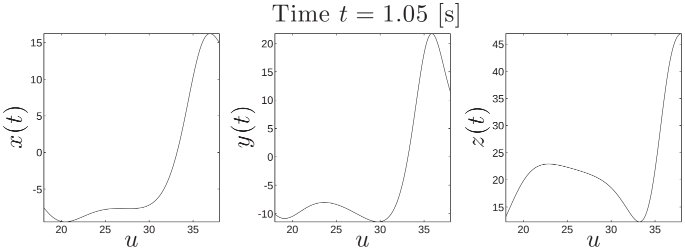


Multiple-Shooting - Key idea

Example

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

States as a function of u at time t

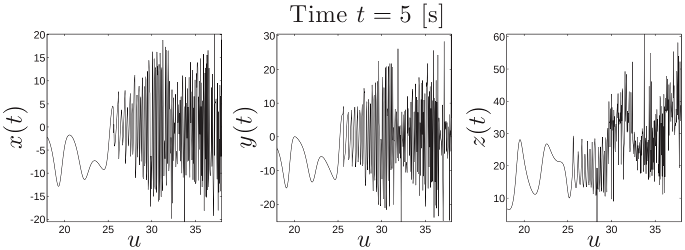


Multiple-Shooting - Key idea

Example

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

States as a function of u at time t

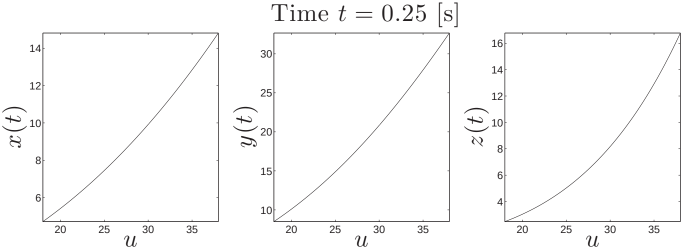


Multiple-Shooting - Key idea

Example

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

States as a function of u at time t

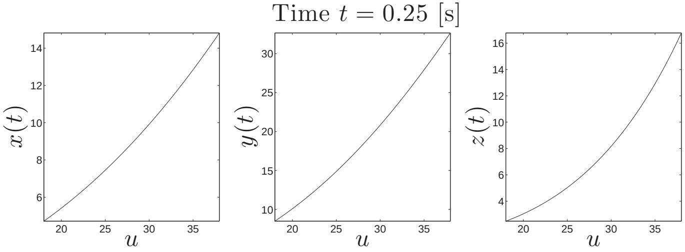


Multiple-Shooting - Key idea

Example

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

States as a function of u at time t



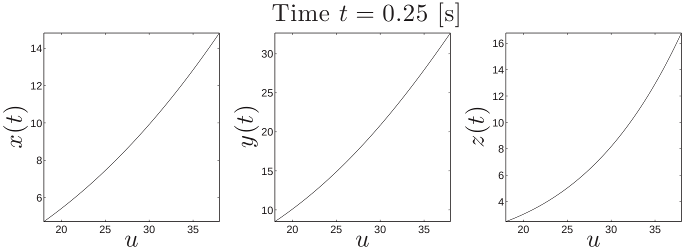
The integration function can be made "arbitrarily linear" by reducing the integration time

Multiple-Shooting - Key idea

Example

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

States as a function of u at time t

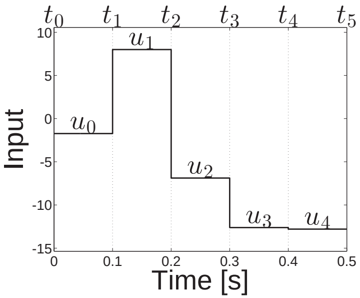


The integration function can be made "arbitrarily linear" by reducing the integration time

Multiple-shooting breaks down the system integration into short time intervals !!

Multiple-Shooting - key idea

Input ...

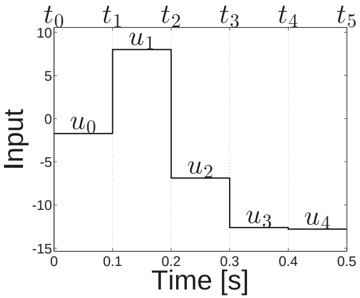


... discretised on the time grid

$$\{t_0,\ t_1,\ \dots,\ t_N\}$$

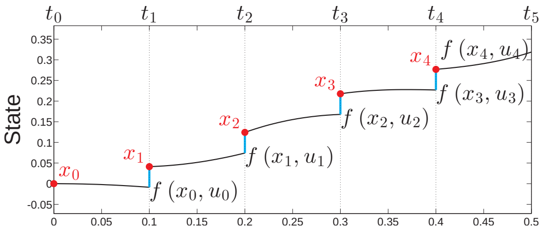
Multiple-Shooting - key idea

Input ...



... discretised on the time grid
$\{t_0, t_1, ..., t_N\}$

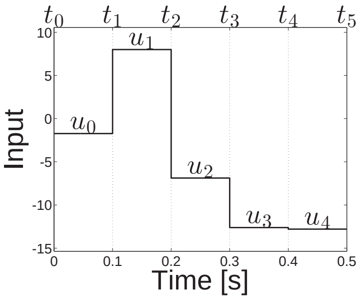
State...



...integration on the time intervals $[t_k, t_{k+1}]$

Multiple-Shooting - key idea

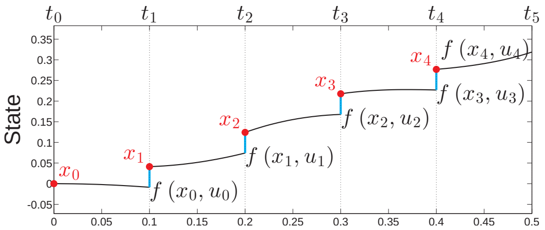
Input ...



... discretised on the time grid

$\{t_0,\ t_1,\ \ldots,\ t_N\}$

State...

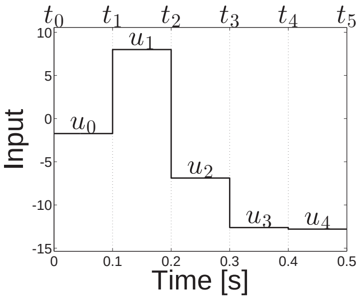


...integration on the time intervals $[t_k,\ t_{k+1}]$

- short integrations starting from *given* $\mathbf{x}_k$

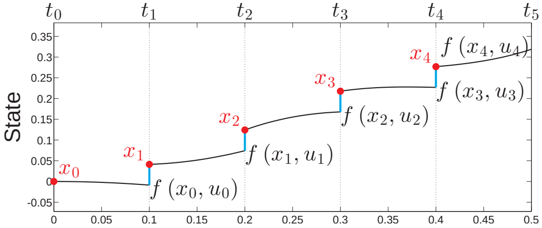
Multiple-Shooting - key idea

Input ...



... discretised on the time grid $\{t_0,\ t_1,\ ...,\ t_N\}$

State...

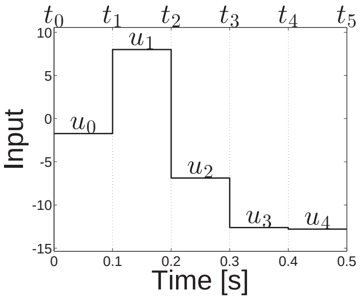


...integration on the time intervals $[t_k,\ t_{k+1}]$

- short integrations starting from *given* $\mathbf{x}_k$
- function $\mathbf{f}(\mathbf{x}_k, \mathbf{u}_k)$ can be made "as linear as we want" by reducing $t_{k+1} - t_k$

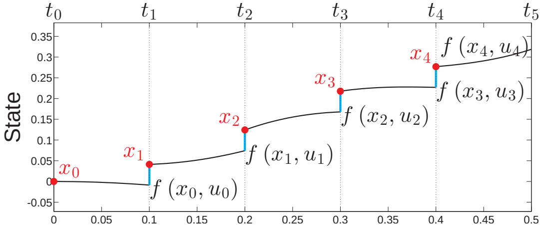
Multiple-Shooting - key idea

Input ...



... discretised on the time grid
$\{t_0, t_1, ..., t_N\}$

State...



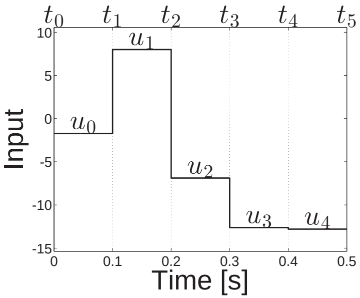
...integration on the time intervals $[t_k, t_{k+1}]$

- short integrations starting from *given* $\mathbf{x}_k$
- function $\mathbf{f}(\mathbf{x}_k, \mathbf{u}_k)$ can be made "as linear as we want" by reducing $t_{k+1} - t_k$
- the trajectory is physically meaningful when the shooting gaps are closed, i.e.

$$\mathbf{f}(\mathbf{x}_k, \mathbf{u}_k) - \mathbf{x}_{k+1} = 0, \quad k = 0, ..., N-1$$

Multiple-Shooting - key idea

Input ...

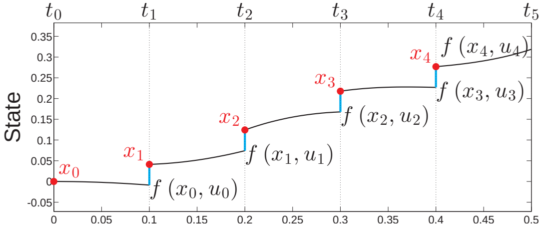


... discretised on the time grid

$\{t_0,\ t_1,\ ...,\ t_N\}$

- The $\mathbf{x}_k$ will become decision variables in the NLP
- The shooting gaps will be constraints in the NLP

State...



...integration on the time intervals $[t_k,\ t_{k+1}]$

- short integrations starting from *given* $\mathbf{x}_k$
- function $\mathbf{f}(\mathbf{x}_k, \mathbf{u}_k)$ can be made "as linear as we want" by reducing $t_{k+1} - t_k$
- the trajectory is physically meaningful when the shooting gaps are closed, i.e.

$$\mathbf{f}(\mathbf{x}_k, \mathbf{u}_k) - \mathbf{x}_{k+1} = 0, \quad k = 0, ..., N-1$$

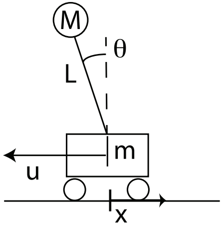
Multiple-shooting - Example

$$\min_{u,\mathbf{x}} \quad \sum_{k=0}^{N} u_k^2$$

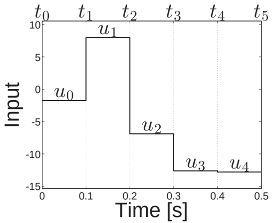
$$\text{s.t.} \quad \mathbf{f}\left(\mathbf{x}_k, \mathbf{u}_k\right) - \mathbf{x}_{k+1} = 0$$

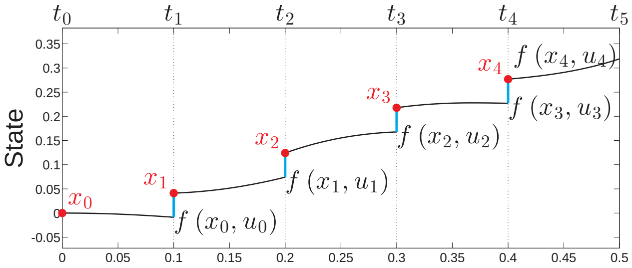
$$-20 \leq u_k \leq 20$$

$$\mathbf{x}_0 = \begin{bmatrix} 0 & \pi & 0 & 0 \end{bmatrix}, \quad \mathbf{x}_N = 0$$



$\mathbf{f}\left(\mathbf{x}_k, \mathbf{u}_k\right)$ integrates the dynamics over the time interval $[t_k, t_{k+1}]$





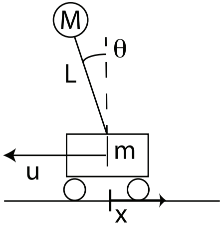
Multiple-shooting - Example

$$\min_{u,\mathbf{x}} \quad \sum_{k=0}^{N} u_k^2$$

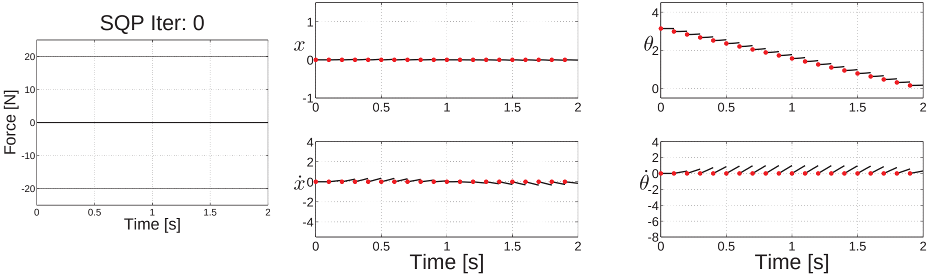
$$\text{s.t.} \quad \mathbf{f}\left(\mathbf{x}_k, \mathbf{u}_k\right) - \mathbf{x}_{k+1} = 0$$

$$-20 \leq u_k \leq 20$$

$$\mathbf{x}_0 = \begin{bmatrix} 0 & \pi & 0 & 0 \end{bmatrix}, \qquad \mathbf{x}_N = 0$$



$\mathbf{f}\left(\mathbf{x}_k, \mathbf{u}_k\right)$ integrates the dynamics over the time interval $[t_k, t_{k+1}]$



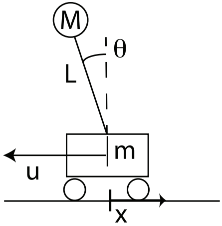
Multiple-shooting - Example

$$\min_{u,\mathbf{x}} \quad \sum_{k=0}^{N} u_k^2$$

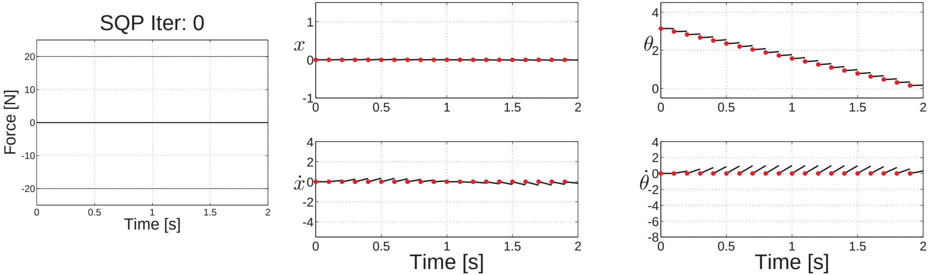
$$\text{s.t.} \quad \mathbf{f}\left(\mathbf{x}_k, \mathbf{u}_k\right) - \mathbf{x}_{k+1} = 0$$

$$-20 \leq u_k \leq 20$$

$$\mathbf{x}_0 = \begin{bmatrix} 0 & \pi & 0 & 0 \end{bmatrix}, \quad \mathbf{x}_N = 0$$



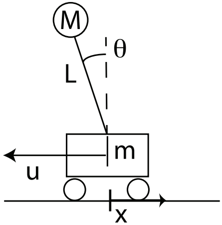
$\mathbf{f}\left(\mathbf{x}_k, \mathbf{u}_k\right)$ integrates the dynamics over the time interval $[t_k, t_{k+1}]$



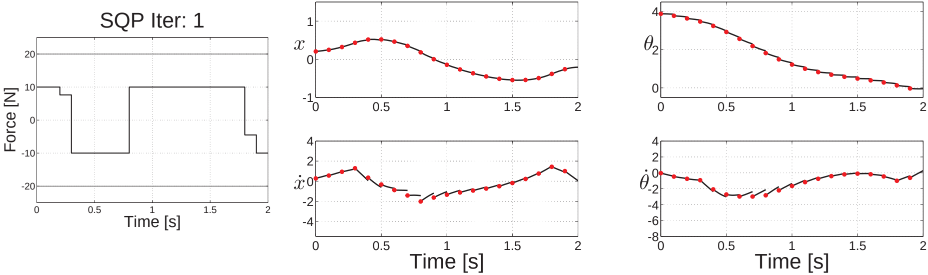
Note: one can provide a guess for the state trajectories in the form of the $\mathbf{x}_{0,\ldots,N}$!!

Multiple-shooting - Example

$$
\begin{aligned}
\min_{u,\mathbf{x}} \quad & \sum_{k=0}^{N} u_k^2 \\
\text{s.t.} \quad & \mathbf{f}\left(\mathbf{x}_k, \mathbf{u}_k\right) - \mathbf{x}_{k+1} = 0 \\
& -20 \le u_k \le 20 \\
& \mathbf{x}_0 = \begin{bmatrix} 0 & \pi & 0 & 0 \end{bmatrix}, \quad \mathbf{x}_N = 0
\end{aligned}
$$

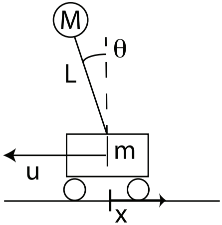


$\mathbf{f}\left(\mathbf{x}_k, \mathbf{u}_k\right)$ integrates the dynamics over the time interval $[t_k, t_{k+1}]$

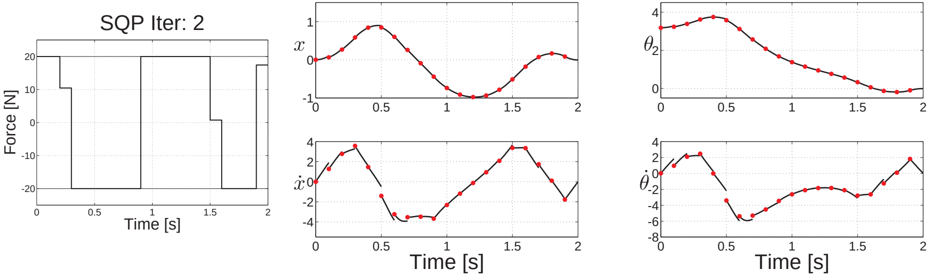


Multiple-shooting - Example

$$\begin{aligned} \min_{u,\mathbf{x}} \quad & \sum_{k=0}^{N} u_k^2 \\ \text{s.t.} \quad & \mathbf{f}\left(\mathbf{x}_k, \mathbf{u}_k\right) - \mathbf{x}_{k+1} = 0 \\ & -20 \leq u_k \leq 20 \\ & \mathbf{x}_0 = \left[\begin{array}{cccc} 0 & \pi & 0 & 0 \end{array}\right], \quad \mathbf{x}_N = 0 \end{aligned}$$

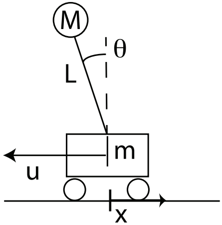


$\mathbf{f}\left(\mathbf{x}_k, \mathbf{u}_k\right)$ integrates the dynamics over the time interval $[t_k, t_{k+1}]$

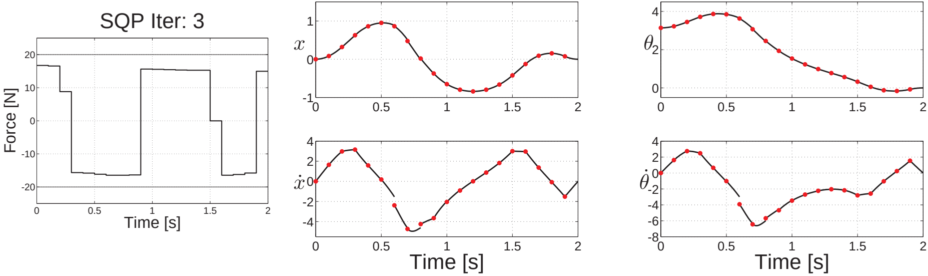


Multiple-shooting - Example

$$\begin{aligned} \min_{u,\mathbf{x}} \quad & \sum_{k=0}^{N} u_k^2 \\ \text{s.t.} \quad & \mathbf{f}\left(\mathbf{x}_k, \mathbf{u}_k\right) - \mathbf{x}_{k+1} = 0 \\ & -20 \leq u_k \leq 20 \\ & \mathbf{x}_0 = \begin{bmatrix} 0 & \pi & 0 & 0 \end{bmatrix}, \quad \mathbf{x}_N = 0 \end{aligned}$$



$\mathbf{f}\left(\mathbf{x}_k, \mathbf{u}_k\right)$ integrates the dynamics over the time interval $[t_k, t_{k+1}]$



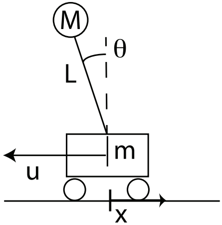
Multiple-shooting - Example

$$\min_{u,\mathbf{x}} \quad \sum_{k=0}^{N} u_k^2$$

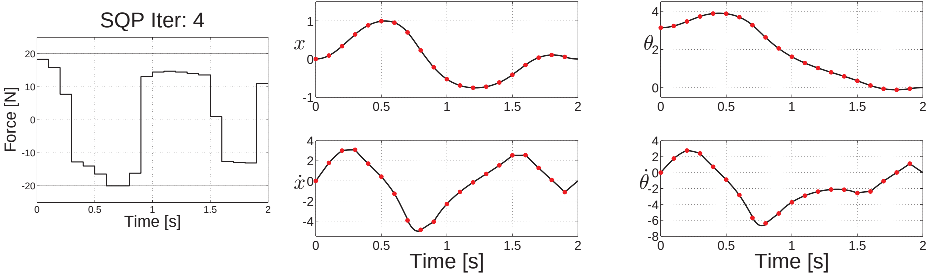
$$\text{s.t.} \quad \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k) - \mathbf{x}_{k+1} = 0$$

$$-20 \leq u_k \leq 20$$

$$\mathbf{x}_0 = \begin{bmatrix} 0 & \pi & 0 & 0 \end{bmatrix}, \quad \mathbf{x}_N = 0$$



$\mathbf{f}(\mathbf{x}_k, \mathbf{u}_k)$ integrates the dynamics over the time interval $[t_k, t_{k+1}]$



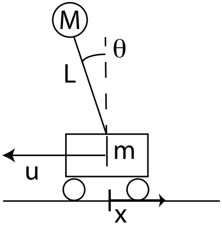
Multiple-shooting - Example

$$\min_{u,\mathbf{x}} \quad \sum_{k=0}^{N} u_k^2$$

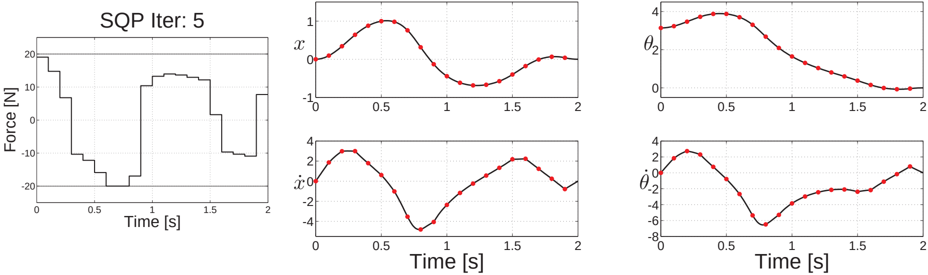
$$\text{s.t.} \quad \mathbf{f}\left(\mathbf{x}_k, \mathbf{u}_k\right) - \mathbf{x}_{k+1} = 0$$

$$-20 \leq u_k \leq 20$$

$$\mathbf{x}_0 = \left[\begin{array}{cccc} 0 & \pi & 0 & 0 \end{array}\right], \qquad \mathbf{x}_N = 0$$

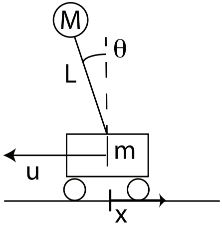


$\mathbf{f}\left(\mathbf{x}_k, \mathbf{u}_k\right)$ integrates the dynamics over the time interval $[t_k, t_{k+1}]$

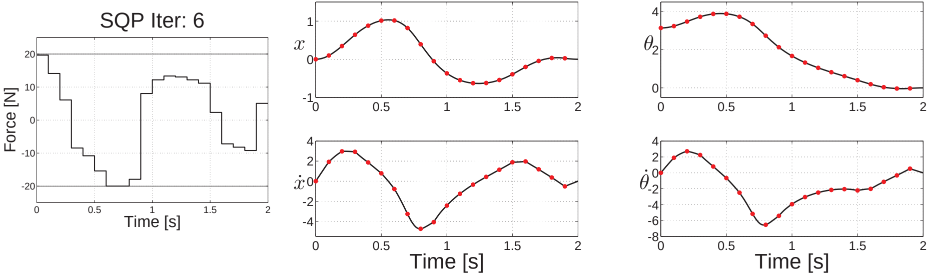


Multiple-shooting - Example

$$\begin{aligned} \min_{u,\mathbf{x}} \quad & \sum_{k=0}^{N} u_k^2 \\ \text{s.t.} \quad & \mathbf{f}\left(\mathbf{x}_k, \mathbf{u}_k\right) - \mathbf{x}_{k+1} = 0 \\ & -20 \leq u_k \leq 20 \\ & \mathbf{x}_0 = \left[\begin{array}{cccc} 0 & \pi & 0 & 0 \end{array}\right], \quad \mathbf{x}_N = 0 \end{aligned}$$

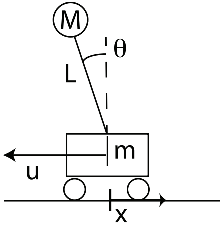


$\mathbf{f}\left(\mathbf{x}_k, \mathbf{u}_k\right)$ integrates the dynamics over the time interval $[t_k, t_{k+1}]$

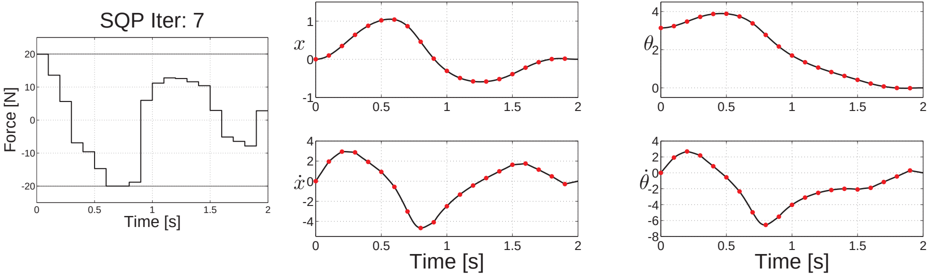


Multiple-shooting - Example

$$\begin{aligned} \min_{u,\mathbf{x}} \quad & \sum_{k=0}^{N} u_k^2 \\ \text{s.t.} \quad & \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k) - \mathbf{x}_{k+1} = 0 \\ & -20 \leq u_k \leq 20 \\ & \mathbf{x}_0 = \begin{bmatrix} 0 & \pi & 0 & 0 \end{bmatrix}, \quad \mathbf{x}_N = 0 \end{aligned}$$



$\mathbf{f}(\mathbf{x}_k, \mathbf{u}_k)$ integrates the dynamics over the time interval $[t_k, t_{k+1}]$



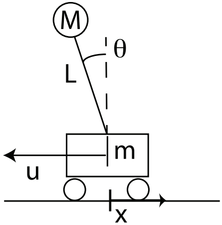
Multiple-shooting - Example

$$\min_{u,\mathbf{x}} \quad \sum_{k=0}^{N} u_k^2$$

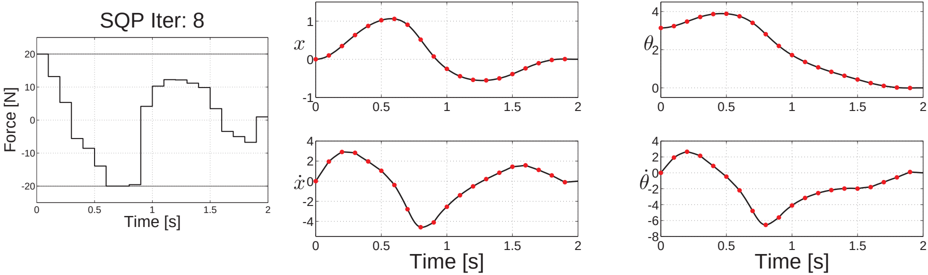
$$\text{s.t.} \quad \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k) - \mathbf{x}_{k+1} = 0$$

$$-20 \leq u_k \leq 20$$

$$\mathbf{x}_0 = \begin{bmatrix} 0 & \pi & 0 & 0 \end{bmatrix}, \quad \mathbf{x}_N = 0$$



$\mathbf{f}(\mathbf{x}_k, \mathbf{u}_k)$ integrates the dynamics over the time interval $[t_k, t_{k+1}]$



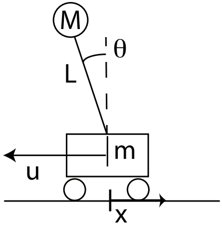
Multiple-shooting - Example

$$\min_{u,\mathbf{x}} \quad \sum_{k=0}^{N} u_k^2$$

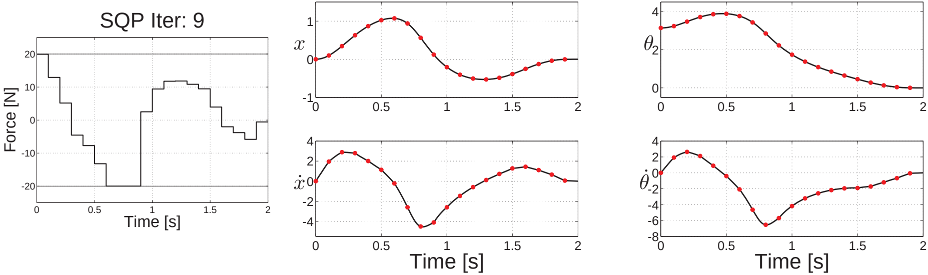
$$\text{s.t.} \quad \mathbf{f}\left(\mathbf{x}_k, \mathbf{u}_k\right) - \mathbf{x}_{k+1} = 0$$

$$-20 \leq u_k \leq 20$$

$$\mathbf{x}_0 = \left[\begin{array}{cccc} 0 & \pi & 0 & 0 \end{array}\right], \quad \mathbf{x}_N = 0$$

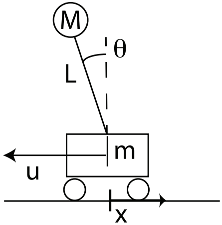


$\mathbf{f}\left(\mathbf{x}_k, \mathbf{u}_k\right)$ integrates the dynamics over the time interval $[t_k, t_{k+1}]$

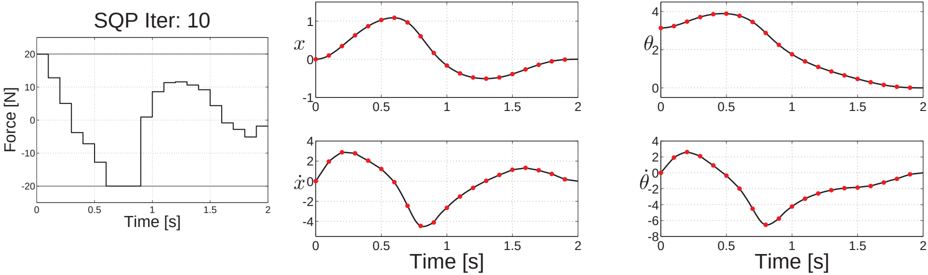


Multiple-shooting - Example

$$\min_{u,\mathbf{x}} \quad \sum_{k=0}^{N} u_k^2$$
$$\text{s.t.} \quad \mathbf{f}\left(\mathbf{x}_k, \mathbf{u}_k\right) - \mathbf{x}_{k+1} = 0$$
$$-20 \leq u_k \leq 20$$
$$\mathbf{x}_0 = \begin{bmatrix} 0 & \pi & 0 & 0 \end{bmatrix}, \quad \mathbf{x}_N = 0$$



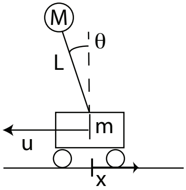
$\mathbf{f}\left(\mathbf{x}_k, \mathbf{u}_k\right)$ integrates the dynamics over the time interval $[t_k, t_{k+1}]$



Remember Single-shooting ?

An example

$$
\begin{aligned}
\min_{u} \quad & \sum_{k=0}^{N} u_k^2 \\
\text{s.t.} \quad & \dot{\mathbf{x}} = f\left(\mathbf{x}, u\right) \\
& -20 \leq u \leq 20 \\
& \mathbf{x}(0) = \left[\begin{array}{cccc} 0 & \pi & 0 & 0 \end{array}\right], \quad \mathbf{x}(T_{\mathrm{f}}) = 0
\end{aligned}
$$



Remember Single-shooting ?

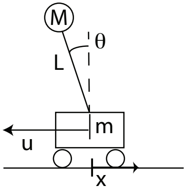
An example

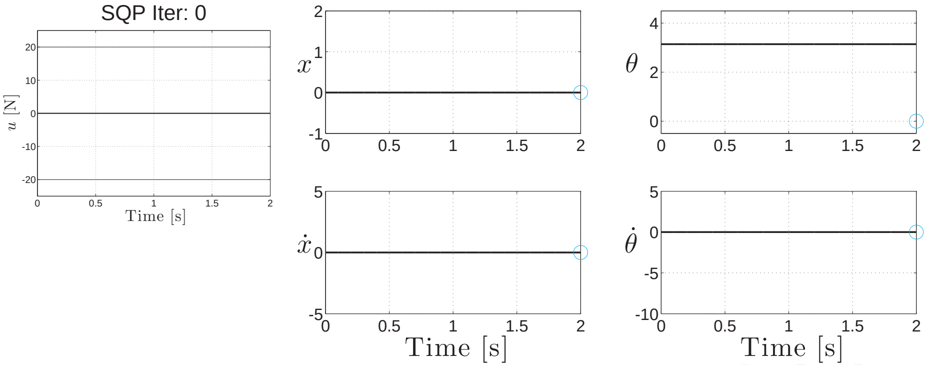
$$\min_{u} \quad \sum_{k=0}^{N} u_k^2$$

$$\text{s.t.} \quad \dot{\mathbf{x}} = f(\mathbf{x}, u)$$

$$-20 \le u \le 20$$

$$\mathbf{x}(0) = \begin{bmatrix} 0 & \pi & 0 & 0 \end{bmatrix}, \quad \mathbf{x}(T_f) = 0$$

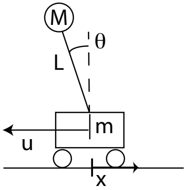


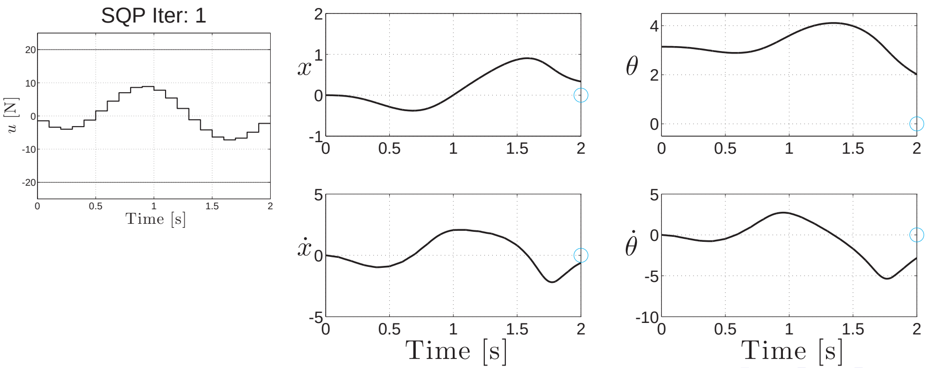


Remember Single-shooting ?

An example

$$\begin{aligned} \min_{u} \quad & \sum_{k=0}^{N} u_k^2 \\ \text{s.t.} \quad & \dot{\mathbf{x}} = f(\mathbf{x}, u) \\ & -20 \leq u \leq 20 \\ & \mathbf{x}(0) = \begin{bmatrix} 0 & \pi & 0 & 0 \end{bmatrix}, \quad \mathbf{x}(T_{\mathrm{f}}) = 0 \end{aligned}$$





Remember Single-shooting ?

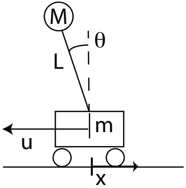
An example

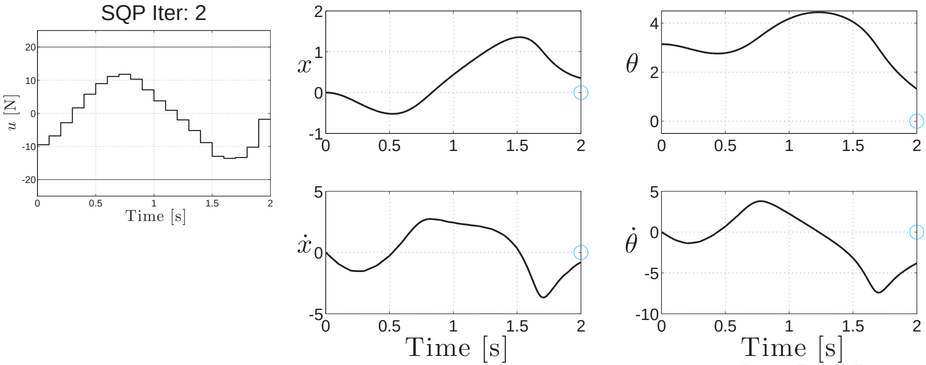
$$\min_{u} \quad \sum_{k=0}^{N} u_k^2$$

$$\text{s.t.} \quad \dot{\mathbf{x}} = f(\mathbf{x}, u)$$

$$-20 \leq u \leq 20$$

$$\mathbf{x}(0) = \begin{bmatrix} 0 & \pi & 0 & 0 \end{bmatrix}, \quad \mathbf{x}(T_\mathrm{f}) = 0$$





Remember Single-shooting ?

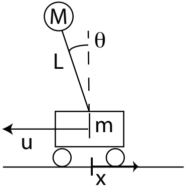
An example

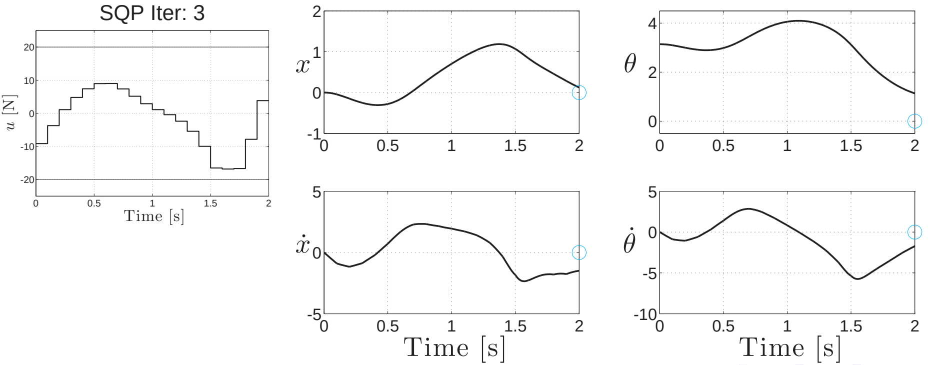
$$\min_{u} \quad \sum_{k=0}^{N} u_k^2$$

$$\text{s.t.} \quad \dot{\mathbf{x}} = f(\mathbf{x}, u)$$

$$-20 \leq u \leq 20$$

$$\mathbf{x}(0) = \begin{bmatrix} 0 & \pi & 0 & 0 \end{bmatrix}, \quad \mathbf{x}(T_\mathrm{f}) = 0$$



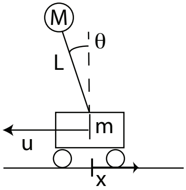


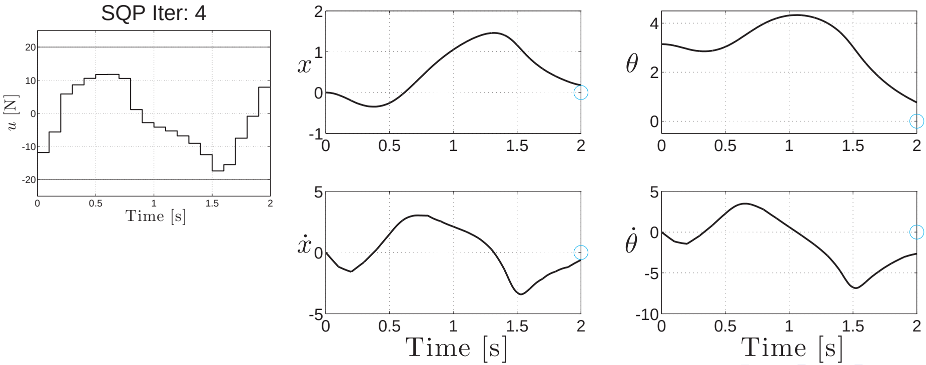
Remember Single-shooting ?

An example

$$\min_{u} \quad \sum_{k=0}^{N} u_k^2$$

$$\begin{aligned} \text{s.t.} \quad & \dot{\mathbf{x}} = f(\mathbf{x}, u) \\ & -20 \leq u \leq 20 \\ & \mathbf{x}(0) = \begin{bmatrix} 0 & \pi & 0 & 0 \end{bmatrix}, \quad \mathbf{x}(T_\mathrm{f}) = 0 \end{aligned}$$





Remember Single-shooting ?

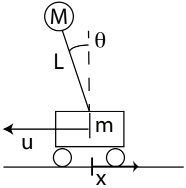
An example

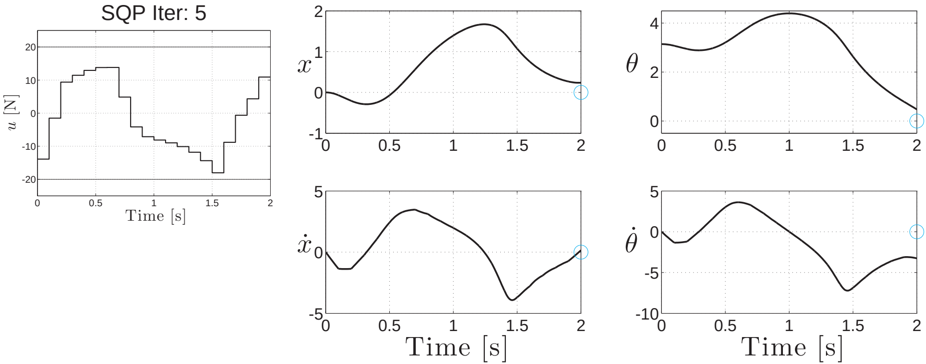
$$\min_{u} \quad \sum_{k=0}^{N} u_k^2$$

$$\text{s.t.} \quad \dot{\mathbf{x}} = f\left(\mathbf{x}, u\right)$$

$$-20 \leq u \leq 20$$

$$\mathbf{x}(0) = \begin{bmatrix} 0 & \pi & 0 & 0 \end{bmatrix}, \quad \mathbf{x}(T_\mathrm{f}) = 0$$



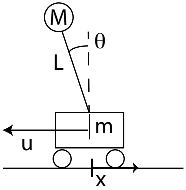


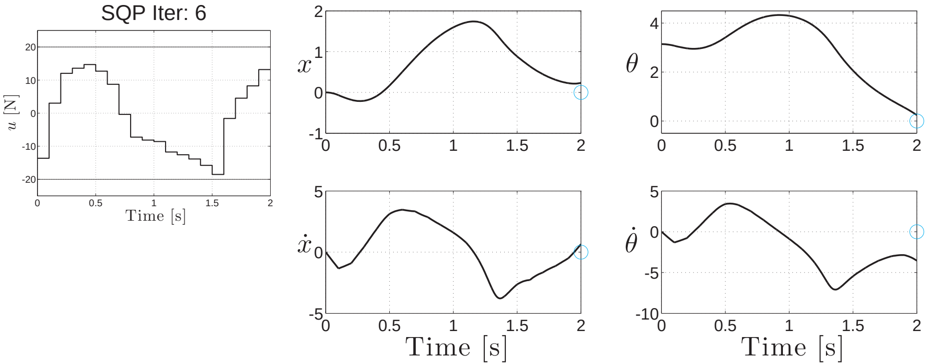
Remember Single-shooting ?

An example

$$\min_{u} \quad \sum_{k=0}^{N} u_k^2$$

$$\begin{aligned} \text{s.t.} \quad & \dot{\mathbf{x}} = f(\mathbf{x}, u) \\ & -20 \leq u \leq 20 \\ & \mathbf{x}(0) = \begin{bmatrix} 0 & \pi & 0 & 0 \end{bmatrix}, \quad \mathbf{x}(T_{\mathrm{f}}) = 0 \end{aligned}$$





Remember Single-shooting ?

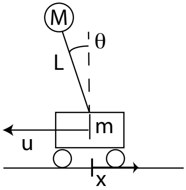
An example

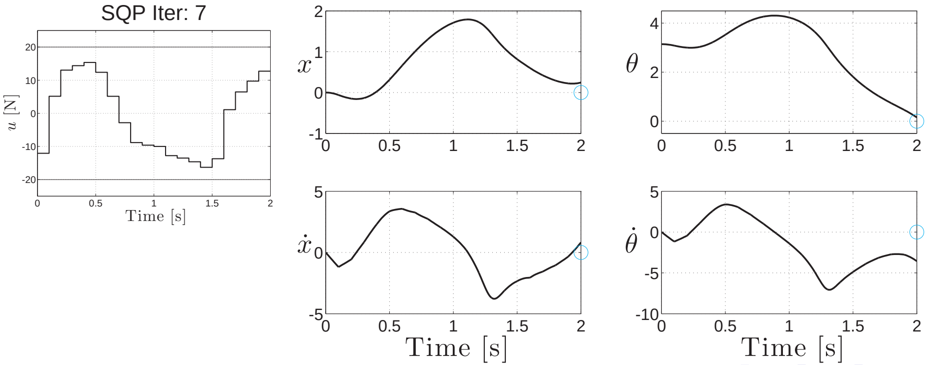
$$\min_{u} \quad \sum_{k=0}^{N} u_k^2$$

$$\text{s.t.} \quad \dot{\mathbf{x}} = f(\mathbf{x}, u)$$

$$-20 \leq u \leq 20$$

$$\mathbf{x}(0) = \begin{bmatrix} 0 & \pi & 0 & 0 \end{bmatrix}, \quad \mathbf{x}(T_{\mathrm{f}}) = 0$$





Remember Single-shooting ?

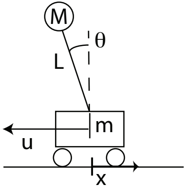
An example

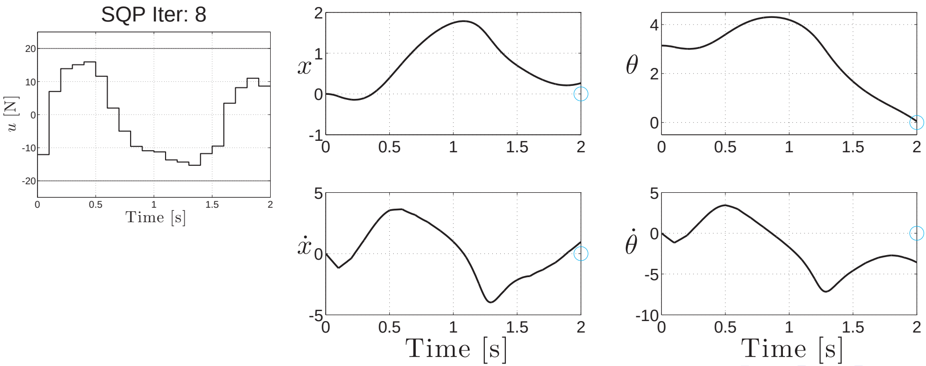
$$\min_{u} \quad \sum_{k=0}^{N} u_k^2$$

$$\text{s.t.} \quad \dot{\mathbf{x}} = f(\mathbf{x}, u)$$

$$-20 \leq u \leq 20$$

$$\mathbf{x}(0) = \begin{bmatrix} 0 & \pi & 0 & 0 \end{bmatrix}, \quad \mathbf{x}(T_f) = 0$$





Remember Single-shooting ?

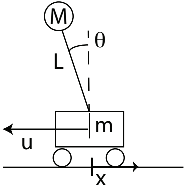
An example

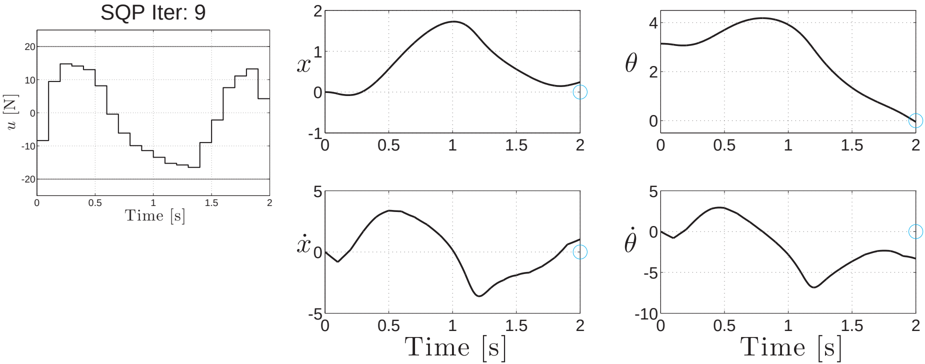
$$\min_{u} \quad \sum_{k=0}^{N} u_k^2$$

$$\text{s.t.} \quad \dot{\mathbf{x}} = f(\mathbf{x}, u)$$

$$-20 \leq u \leq 20$$

$$\mathbf{x}(0) = \begin{bmatrix} 0 & \pi & 0 & 0 \end{bmatrix}, \quad \mathbf{x}(T_\mathrm{f}) = 0$$





Remember Single-shooting ?

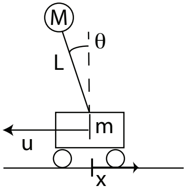
An example

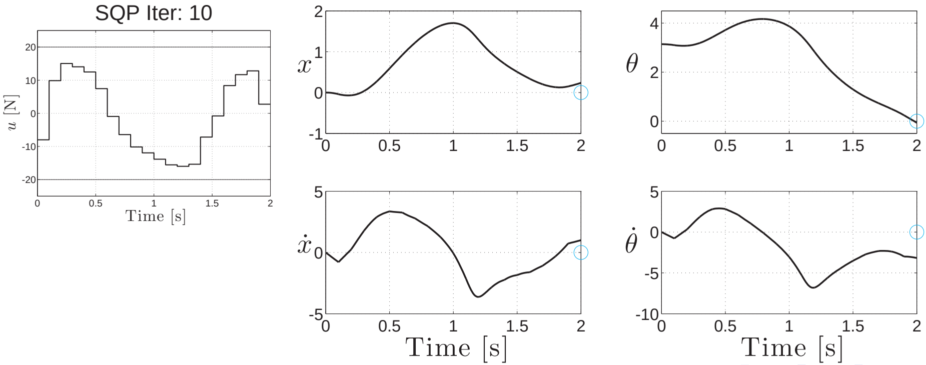
$$\min_{u} \quad \sum_{k=0}^{N} u_k^2$$

$$\text{s.t.} \quad \dot{\mathbf{x}} = f(\mathbf{x}, u)$$

$$-20 \leq u \leq 20$$

$$\mathbf{x}(0) = \begin{bmatrix} 0 & \pi & 0 & 0 \end{bmatrix}, \quad \mathbf{x}(T_{\mathrm{f}}) = 0$$

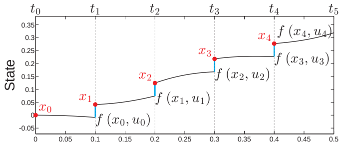




Cost and constraints discretisation in Multiple-shooting

OCP:

$$\begin{aligned} \min \quad & T\left(\mathbf{x}\left(t_{\mathrm{f}}\right)\right) + \int_0^{t_{\mathrm{f}}} L\left(\mathbf{x}\left(t\right), \mathbf{u}\left(t\right)\right) dt \\ \text{s.t.} \quad & \dot{\mathbf{x}} = \mathbf{F}\left(\mathbf{x}, \mathbf{u}\right) \\ & \mathbf{h}\left(\mathbf{x}\left(t\right), \mathbf{u}\left(t\right)\right) \leq 0 \end{aligned}$$



- Inequality constraints: $\mathbf{h}\left(\mathbf{x}\left(t\right), \mathbf{u}\left(t\right)\right) \leq 0$ are enforced on the the shooting nodes:

$$\mathbf{h}\left(\mathbf{x}_k, t_k, \mathbf{u}_k\right) \leq 0, \quad \forall\, k = 0, ..., N-1$$

- Cost function often approximated as (rectangular quadrature):

$$T\left(\mathbf{x}_N\right) + \sum_{k=0}^{N-1} \left(t_{k+1} - t_k\right) L\left(\mathbf{x}_k, \mathbf{u}_k\left(t\right)\right)$$

- Alternatively, integral cost function $L\left(\mathbf{x}, \mathbf{u}\right)$ can be implemented via a dynamic extension:

$$\frac{d}{dt}\left[\begin{array}{c} \mathbf{x} \\ \rho \end{array}\right] = \left[\begin{array}{c} \mathbf{F}\left(\mathbf{x}, \mathbf{u}\right) \\ L\left(\mathbf{x}, \mathbf{u}\right) \end{array}\right], \quad \Phi\left(\mathbf{w}\right) = T\left(\mathbf{x}_N\right) + \rho_N$$

Outline

NLP from Single-Shooting

OCP:

$$\begin{aligned} \min \quad & \Phi\left(\mathbf{x}(.), \mathbf{u}(.)\right) \\ \text{s.t.} \quad & \dot{\mathbf{x}}\left(t\right) = \mathbf{F}\left(\mathbf{x}\left(t\right), \mathbf{u}\left(t\right)\right) \\ & \mathbf{h}\left(\mathbf{x}(t), \mathbf{u}(t), t\right) \leq 0 \\ & \mathbf{x}\left(t_0\right) = \mathbf{x}_0 \end{aligned}$$

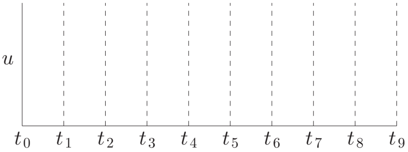
NLP:

with $\mathbf{w} = \{\mathbf{u}_0, ..., \mathbf{u}_{N-1}\}$

NLP from Single-Shooting

OCP:

$$\begin{aligned} \min \quad & \Phi\left(\mathbf{x}(.), \mathbf{u}(.)\right) \\ \text{s.t.} \quad & \dot{\mathbf{x}}\left(t\right) = \mathbf{F}\left(\mathbf{x}\left(t\right), \mathbf{u}\left(t\right)\right) \\ & \mathbf{h}\left(\mathbf{x}(t), \mathbf{u}(t), t\right) \leq 0 \\ & \mathbf{x}\left(t_0\right) = \mathbf{x}_0 \end{aligned}$$



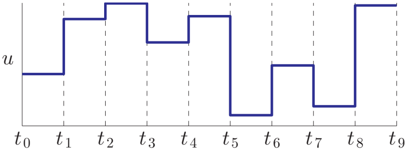
NLP:

with $\mathbf{w} = \{\mathbf{u}_0, ..., \mathbf{u}_{N-1}\}$

NLP from Single-Shooting

OCP:

$$\begin{aligned} \min \quad & \Phi\left(\mathbf{x}(.), \mathbf{u}(.)\right) \\ \text{s.t.} \quad & \dot{\mathbf{x}}\left(t\right) = \mathbf{F}\left(\mathbf{x}\left(t\right), \mathbf{u}\left(t\right)\right) \\ & \mathbf{h}\left(\mathbf{x}(t), \mathbf{u}(t), t\right) \leq 0 \\ & \mathbf{x}\left(t_0\right) = \mathbf{x}_0 \end{aligned}$$



NLP:

with $\mathbf{w} = \{\mathbf{u}_0, ..., \mathbf{u}_{N-1}\}$

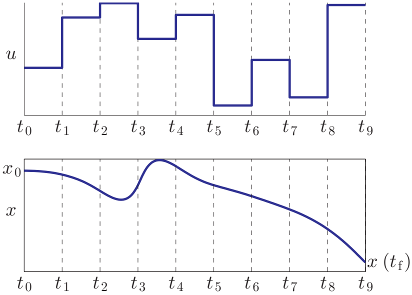
NLP from Single-Shooting

OCP:

$$\begin{aligned} \min \quad & \Phi\left(\mathbf{x}(.), \mathbf{u}(.)\right) \\ \text{s.t.} \quad & \dot{\mathbf{x}}\left(t\right) = \mathbf{F}\left(\mathbf{x}\left(t\right), \mathbf{u}\left(t\right)\right) \\ & \mathbf{h}\left(\mathbf{x}(t), \mathbf{u}(t), t\right) \leq 0 \\ & \mathbf{x}\left(t_0\right) = \mathbf{x}_0 \end{aligned}$$

NLP:

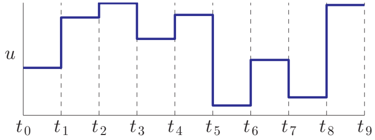
with $\mathbf{w} = \left\{\mathbf{u}_0, ..., \mathbf{u}_{N-1}\right\}$

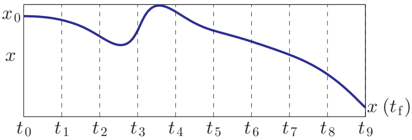


NLP from Single-Shooting

OCP:

$$\begin{aligned} \min \quad & \Phi\left(\mathbf{x}(.), \mathbf{u}(.)\right) \\ \text{s.t.} \quad & \dot{\mathbf{x}}\left(t\right) = \mathbf{F}\left(\mathbf{x}\left(t\right), \mathbf{u}\left(t\right)\right) \\ & \mathbf{h}\left(\mathbf{x}(t), \mathbf{u}(t), t\right) \leq 0 \\ & \mathbf{x}\left(t_0\right) = \mathbf{x}_0 \end{aligned}$$





NLP:

$$\begin{aligned} \min_{\mathbf{w}} \quad & \Phi\left(\mathbf{f}\left(\mathbf{w}, \mathbf{x}_0.\right), \mathbf{w}\right) \\ \text{s.t.} \quad & \mathbf{h}\left(\mathbf{f}\left(\mathbf{w}, \mathbf{x}_0, t_k\right), \mathbf{w}_k, t_k\right) \leq 0 \end{aligned}$$

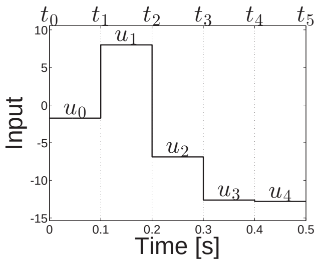
with $\mathbf{w} = \{\mathbf{u}_0, ..., \mathbf{u}_{N-1}\}$

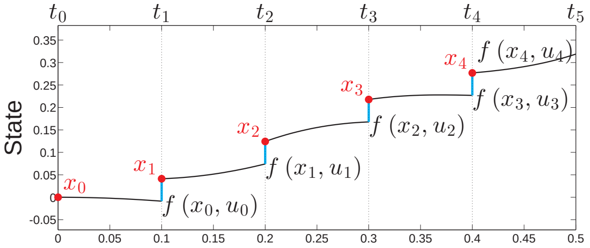
NLP from Multiple-Shooting

OCP:

$$\begin{aligned} \min \quad & \Phi\left(\mathbf{x}(.), \mathbf{u}(.)\right) \\ \text{s.t.} \quad & \dot{\mathbf{x}}\left(t\right) = \mathbf{F}\left(\mathbf{x}\left(t\right), \mathbf{u}\left(t\right)\right) \\ & \mathbf{h}\left(\mathbf{x}(t), \mathbf{u}(t), t\right) \leq 0 \\ & \mathbf{x}\left(t_0\right) = \bar{\mathbf{x}}_0 \end{aligned}$$

$\mathbf{f}\left(\mathbf{x}_k, \mathbf{u}_k\right)$ integrates the dynamics $\mathbf{F}$ over the time interval $[t_k,\ t_{k+1}]$





NLP from Multiple-Shooting

OCP:

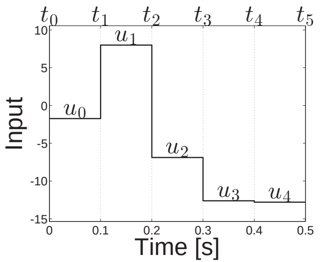
$$\begin{aligned} \min \quad & \Phi\left(\mathbf{x}(.), \mathbf{u}(.)\right) \\ \text{s.t.} \quad & \dot{\mathbf{x}}(t) = \mathbf{F}\left(\mathbf{x}(t), \mathbf{u}(t)\right) \\ & \mathbf{h}\left(\mathbf{x}(t), \mathbf{u}(t), t\right) \leq 0 \\ & \mathbf{x}(t_0) = \bar{\mathbf{x}}_0 \end{aligned}$$

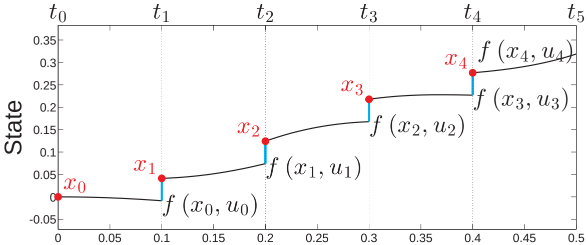
$\mathbf{f}\left(\mathbf{x}_k, \mathbf{u}_k\right)$ integrates the dynamics $\mathbf{F}$ over the time interval $[t_k,\ t_{k+1}]$

NLP with $\mathbf{w} = \{\mathbf{x}_0, \mathbf{u}_0, ..., \mathbf{x}_{N-1}, \mathbf{u}_{N-1}, \mathbf{x}_N\}$

$$\min_{\mathbf{w}} \quad \Phi\left(\mathbf{w}\right)$$

s.t.





NLP from Multiple-Shooting

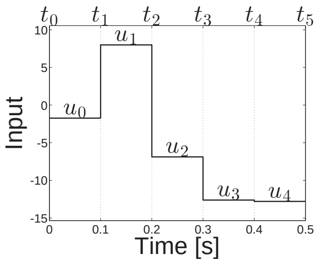
OCP:

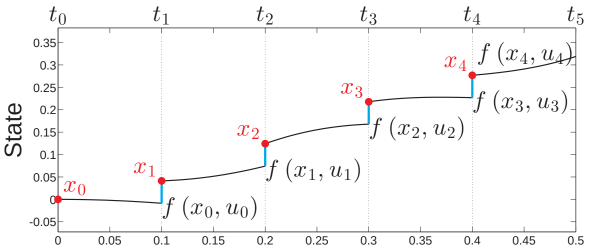
$$
\begin{aligned}
\min \quad & \Phi\left(\mathbf{x}(.), \mathbf{u}(.)\right) \\
\text{s.t.} \quad & \dot{\mathbf{x}}\left(t\right) = \mathbf{F}\left(\mathbf{x}\left(t\right), \mathbf{u}\left(t\right)\right) \\
& \mathbf{h}\left(\mathbf{x}(t), \mathbf{u}(t), t\right) \leq 0 \\
& \mathbf{x}\left(t_0\right) = \bar{\mathbf{x}}_0
\end{aligned}
$$

$\mathbf{f}\left(\mathbf{x}_k, \mathbf{u}_k\right)$ integrates the dynamics $\mathbf{F}$ over the time interval $[t_k,\ t_{k+1}]$

NLP with $\mathbf{w} = \{\mathbf{x}_0, \mathbf{u}_0, ..., \mathbf{x}_{N-1}, \mathbf{u}_{N-1}, \mathbf{x}_N\}$

$$
\begin{aligned}
\min_{\mathbf{w}} \quad & \Phi\left(\mathbf{w}\right) \\
\text{s.t.} \quad & \mathbf{g}\left(\mathbf{w}\right) = \begin{bmatrix} \bar{\mathbf{x}}_0 - \mathbf{x}_0 \\ \mathbf{f}\left(\mathbf{x}_0, \mathbf{u}_0\right) - \mathbf{x}_1 \\ \dots \\ \mathbf{f}\left(\mathbf{x}_{N-1}, \mathbf{u}_{N-1}\right) - \mathbf{x}_N \end{bmatrix} = 0
\end{aligned}
$$





NLP from Multiple-Shooting

OCP:

$$\begin{aligned}
\min \quad & \Phi\left(\mathbf{x}(.), \mathbf{u}(.)\right) \\
\text{s.t.} \quad & \dot{\mathbf{x}}\left(t\right) = \mathbf{F}\left(\mathbf{x}\left(t\right), \mathbf{u}\left(t\right)\right) \\
& \mathbf{h}\left(\mathbf{x}(t), \mathbf{u}(t), t\right) \leq 0 \\
& \mathbf{x}\left(t_0\right) = \bar{\mathbf{x}}_0
\end{aligned}$$

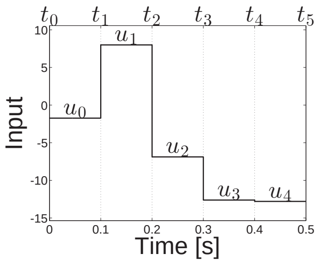
$\mathbf{f}\left(\mathbf{x}_k, \mathbf{u}_k\right)$ integrates the dynamics $\mathbf{F}$ over the time interval $[t_k,\ t_{k+1}]$

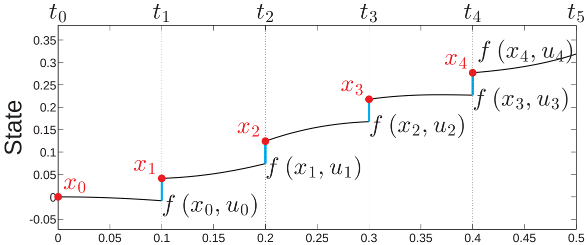
NLP with $\mathbf{w} = \{\mathbf{x}_0, \mathbf{u}_0, \ldots, \mathbf{x}_{N-1}, \mathbf{u}_{N-1}, \mathbf{x}_N\}$

$$\min_{\mathbf{w}} \quad \Phi\left(\mathbf{w}\right)$$

$$\text{s.t.} \quad \mathbf{g}\left(\mathbf{w}\right) = \begin{bmatrix} \bar{\mathbf{x}}_0 - \mathbf{x}_0 \\ \mathbf{f}\left(\mathbf{x}_0, \mathbf{u}_0\right) - \mathbf{x}_1 \\ \ldots \\ \mathbf{f}\left(\mathbf{x}_{N-1}, \mathbf{u}_{N-1}\right) - \mathbf{x}_N \end{bmatrix} = 0$$

$$\mathbf{h}\left(\mathbf{w}\right) = \begin{bmatrix} \mathbf{h}\left(\mathbf{x}_0, \mathbf{u}_0\right) \\ \ldots \\ \mathbf{h}\left(\mathbf{x}_{N-1}, \mathbf{u}_{N-1}\right) \\ \mathbf{h}\left(\mathbf{x}_N\right) \end{bmatrix} \leq 0$$





Lagrange function in Multiple-Shooting

NLP:

$$\min_{\mathbf{w}} \quad \Phi(\mathbf{w})$$

$$\text{s.t.} \quad \mathbf{g}(\mathbf{w}) = \begin{bmatrix} \bar{\mathbf{x}}_0 - \mathbf{x}_0 \\ \mathbf{f}(\mathbf{x}_0, \mathbf{u}_0) - \mathbf{x}_1 \\ \dots \\ \mathbf{f}(\mathbf{x}_{N-1}, \mathbf{u}_{N-1}) - \mathbf{x}_N \end{bmatrix} = 0$$

$$\mathbf{h}(\mathbf{w}) = \begin{bmatrix} \mathbf{h}(\mathbf{x}_0, \mathbf{u}_0) \\ \dots \\ \mathbf{h}(\mathbf{x}_{N-1}, \mathbf{u}_{N-1}) \\ \mathbf{h}(\mathbf{x}_N) \end{bmatrix} \leq 0$$

Lagrange function in Multiple-Shooting

NLP:

$$\min_{\mathbf{w}} \quad \Phi(\mathbf{w})$$

$$\text{s.t.} \quad \mathbf{g}(\mathbf{w}) = \begin{bmatrix} \bar{\mathbf{x}}_0 - \mathbf{x}_0 \\ \mathbf{f}(\mathbf{x}_0, \mathbf{u}_0) - \mathbf{x}_1 \\ \ldots \\ \mathbf{f}(\mathbf{x}_{N-1}, \mathbf{u}_{N-1}) - \mathbf{x}_N \end{bmatrix} = 0$$

$$\mathbf{h}(\mathbf{w}) = \begin{bmatrix} \mathbf{h}(\mathbf{x}_0, \mathbf{u}_0) \\ \ldots \\ \mathbf{h}(\mathbf{x}_{N-1}, \mathbf{u}_{N-1}) \\ \mathbf{h}(\mathbf{x}_N) \end{bmatrix} \leq 0$$

Lagrange function:

$$\mathcal{L}(\mathbf{w}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = \Phi(\mathbf{w}) + \boldsymbol{\lambda}^\top \mathbf{g}(\mathbf{w}) + \boldsymbol{\mu}^\top \mathbf{h}(\mathbf{w})$$

Lagrange function in Multiple-Shooting

NLP:

$$\min_{\mathbf{w}} \quad \Phi(\mathbf{w})$$

$$\text{s.t.} \quad \mathbf{g}(\mathbf{w}) = \begin{bmatrix} \bar{\mathbf{x}}_0 - \mathbf{x}_0 \\ \mathbf{f}(\mathbf{x}_0, \mathbf{u}_0) - \mathbf{x}_1 \\ \dots \\ \mathbf{f}(\mathbf{x}_{N-1}, \mathbf{u}_{N-1}) - \mathbf{x}_N \end{bmatrix} = 0$$

$$\mathbf{h}(\mathbf{w}) = \begin{bmatrix} \mathbf{h}(\mathbf{x}_0, \mathbf{u}_0) \\ \dots \\ \mathbf{h}(\mathbf{x}_{N-1}, \mathbf{u}_{N-1}) \\ \mathbf{h}(\mathbf{x}_N) \end{bmatrix} \leq 0$$

Then write:

$$\mathcal{L}(\mathbf{w}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = \underbrace{T(\mathbf{x}_N) + \sum_{k=0}^{N-1} L(\mathbf{x}_k, \mathbf{u}_k)}_{\Phi(\mathbf{w})}$$

Lagrange function:

$$\mathcal{L}(\mathbf{w}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = \Phi(\mathbf{w}) + \boldsymbol{\lambda}^\top \mathbf{g}(\mathbf{w}) + \boldsymbol{\mu}^\top \mathbf{h}(\mathbf{w})$$

Lagrange function in Multiple-Shooting

NLP:

$$\min_{\mathbf{w}} \quad \Phi(\mathbf{w})$$

$$\text{s.t.} \quad \mathbf{g}(\mathbf{w}) = \begin{bmatrix} \bar{\mathbf{x}}_0 - \mathbf{x}_0 \\ \mathbf{f}(\mathbf{x}_0, \mathbf{u}_0) - \mathbf{x}_1 \\ \dots \\ \mathbf{f}(\mathbf{x}_{N-1}, \mathbf{u}_{N-1}) - \mathbf{x}_N \end{bmatrix} = 0$$

$$\mathbf{h}(\mathbf{w}) = \begin{bmatrix} \mathbf{h}(\mathbf{x}_0, \mathbf{u}_0) \\ \dots \\ \mathbf{h}(\mathbf{x}_{N-1}, \mathbf{u}_{N-1}) \\ \mathbf{h}(\mathbf{x}_N) \end{bmatrix} \leq 0$$

Lagrange function:

$$\mathcal{L}(\mathbf{w}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = \Phi(\mathbf{w}) + \boldsymbol{\lambda}^\top \mathbf{g}(\mathbf{w}) + \boldsymbol{\mu}^\top \mathbf{h}(\mathbf{w})$$

Then write:

$$\mathcal{L}(\mathbf{w}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = \underbrace{T(\mathbf{x}_N) + \sum_{k=0}^{N-1} L(\mathbf{x}_k, \mathbf{u}_k)}_{\Phi(\mathbf{w})} + \underbrace{\boldsymbol{\lambda}_0^\top (\bar{\mathbf{x}}_0 - \mathbf{x}_0) + \sum_{k=0}^{N-1} \boldsymbol{\lambda}_{k+1}^\top (\mathbf{f}(\mathbf{x}_k, \mathbf{u}_k) - \mathbf{x}_{k+1})}_{\boldsymbol{\lambda}^\top \mathbf{g}(\mathbf{w})}$$

Lagrange function in Multiple-Shooting

NLP:

$$\min_{\mathbf{w}} \quad \Phi(\mathbf{w})$$

$$\text{s.t.} \quad \mathbf{g}(\mathbf{w}) = \begin{bmatrix} \bar{\mathbf{x}}_0 - \mathbf{x}_0 \\ \mathbf{f}(\mathbf{x}_0, \mathbf{u}_0) - \mathbf{x}_1 \\ \dots \\ \mathbf{f}(\mathbf{x}_{N-1}, \mathbf{u}_{N-1}) - \mathbf{x}_N \end{bmatrix} = 0$$

$$\mathbf{h}(\mathbf{w}) = \begin{bmatrix} \mathbf{h}(\mathbf{x}_0, \mathbf{u}_0) \\ \dots \\ \mathbf{h}(\mathbf{x}_{N-1}, \mathbf{u}_{N-1}) \\ \mathbf{h}(\mathbf{x}_N) \end{bmatrix} \leq 0$$

Lagrange function:

$$\mathcal{L}(\mathbf{w}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = \Phi(\mathbf{w}) + \boldsymbol{\lambda}^\top \mathbf{g}(\mathbf{w}) + \boldsymbol{\mu}^\top \mathbf{h}(\mathbf{w})$$

Then write:

$$\mathcal{L}(\mathbf{w}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = \underbrace{T(\mathbf{x}_N) + \sum_{k=0}^{N-1} L(\mathbf{x}_k, \mathbf{u}_k)}_{\Phi(\mathbf{w})} + \underbrace{\boldsymbol{\lambda}_0^\top (\bar{\mathbf{x}}_0 - \mathbf{x}_0) + \sum_{k=0}^{N-1} \boldsymbol{\lambda}_{k+1}^\top (\mathbf{f}(\mathbf{x}_k, \mathbf{u}_k) - \mathbf{x}_{k+1})}_{\boldsymbol{\lambda}^\top \mathbf{g}(\mathbf{w})}$$

$$+ \underbrace{\boldsymbol{\mu}_N^\top \mathbf{h}(\mathbf{x}_N) + \sum_{k=0}^{N-1} \boldsymbol{\mu}_k^\top \mathbf{h}(\mathbf{x}_k, \mathbf{u}_k)}_{\boldsymbol{\mu}^\top \mathbf{h}(\mathbf{w})}$$

Separability of the Lagrange function

$$\mathcal{L}(\mathbf{w}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = \underbrace{T(\mathbf{x}_N) + \sum_{k=0}^{N-1} L(\mathbf{x}_k, \mathbf{u}_k)}_{\Phi(\mathbf{w})} + \underbrace{\boldsymbol{\lambda}_0^\top (\bar{\mathbf{x}}_0 - \mathbf{x}_0) + \sum_{k=0}^{N-1} \boldsymbol{\lambda}_{k+1}^\top (\mathbf{f}(\mathbf{x}_k, \mathbf{u}_k) - \mathbf{x}_{k+1})}_{\boldsymbol{\lambda}^\top \mathbf{g}(\mathbf{w})}$$

$$+ \underbrace{\boldsymbol{\mu}_N^\top \mathbf{h}(\mathbf{x}_N) + \sum_{k=0}^{N-1} \boldsymbol{\mu}_k^\top \mathbf{h}(\mathbf{x}_k, \mathbf{u}_k)}_{\boldsymbol{\mu}^\top \mathbf{h}(\mathbf{w})}$$

Separability of the Lagrange function

$$\mathcal{L}(\mathbf{w},\boldsymbol{\lambda},\boldsymbol{\mu}) = T(\mathbf{x}_N) + \sum_{k=0}^{N-1} L(\mathbf{x}_k,\mathbf{u}_k) + \boldsymbol{\lambda}_0^\top(\bar{\mathbf{x}}_0 - \mathbf{x}_0) + \sum_{k=0}^{N-1}\boldsymbol{\lambda}_{k+1}^\top(\mathbf{f}(\mathbf{x}_k,\mathbf{u}_k) - \mathbf{x}_{k+1})$$
$$+ \boldsymbol{\mu}_N^\top \mathbf{h}(\mathbf{x}_N) + \sum_{k=0}^{N-1}\boldsymbol{\mu}_k^\top \mathbf{h}(\mathbf{x}_k,\mathbf{u}_k)$$

Separability of the Lagrange function

$$\mathcal{L}(\mathbf{w}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = T(\mathbf{x}_N) + \sum_{k=0}^{N-1} L(\mathbf{x}_k, \mathbf{u}_k) + \boldsymbol{\lambda}_0^\top \bar{\mathbf{x}}_0 - \boldsymbol{\lambda}_0^\top \mathbf{x}_0 + \sum_{k=0}^{N-1} \boldsymbol{\lambda}_{k+1}^\top \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k) - \sum_{k=0}^{N-1} \boldsymbol{\lambda}_{k+1}^\top \mathbf{x}_{k+1}$$

$$+ \boldsymbol{\mu}_N^\top \mathbf{h}(\mathbf{x}_N) + \sum_{k=0}^{N-1} \boldsymbol{\mu}_k^\top \mathbf{h}(\mathbf{x}_k, \mathbf{u}_k)$$

Separability of the Lagrange function

$$\mathcal{L}(\mathbf{w}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = T(\mathbf{x}_N) + \sum_{k=0}^{N-1} L(\mathbf{x}_k, \mathbf{u}_k) + \boldsymbol{\lambda}_0^\top \bar{\mathbf{x}}_0 - \boldsymbol{\lambda}_0^\top \mathbf{x}_0 + \sum_{k=0}^{N-1} \boldsymbol{\lambda}_{k+1}^\top \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k) - \sum_{k=0}^{N-1} \boldsymbol{\lambda}_{k+1}^\top \mathbf{x}_{k+1}$$

$$+ \boldsymbol{\mu}_N^\top \mathbf{h}(\mathbf{x}_N) + \sum_{k=0}^{N-1} \boldsymbol{\mu}_k^\top \mathbf{h}(\mathbf{x}_k, \mathbf{u}_k) =$$

$$T(\mathbf{x}_N) + \boldsymbol{\mu}_N^\top \mathbf{h}(\mathbf{x}_N) + \boldsymbol{\lambda}_0^\top \bar{\mathbf{x}}_0 - \boldsymbol{\lambda}_N^\top \mathbf{x}_N$$

$$+ \sum_{k=0}^{N-1} \Big(L(\mathbf{x}_k, \mathbf{u}_k) + \boldsymbol{\lambda}_{k+1}^\top \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k) - \boldsymbol{\lambda}_k^\top \mathbf{x}_k + \boldsymbol{\mu}_k^\top \mathbf{h}(\mathbf{x}_k, \mathbf{u}_k) \Big)$$

Separability of the Lagrange function

$$\mathcal{L}(\mathbf{w},\boldsymbol{\lambda},\boldsymbol{\mu}) = T(\mathbf{x}_N) + \sum_{k=0}^{N-1} L(\mathbf{x}_k,\mathbf{u}_k) + \boldsymbol{\lambda}_0^\top \bar{\mathbf{x}}_0 - \boldsymbol{\lambda}_0^\top \mathbf{x}_0 + \sum_{k=0}^{N-1} \boldsymbol{\lambda}_{k+1}^\top \mathbf{f}(\mathbf{x}_k,\mathbf{u}_k) - \sum_{k=0}^{N-1} \boldsymbol{\lambda}_{k+1}^\top \mathbf{x}_{k+1}$$

$$+ \boldsymbol{\mu}_N^\top \mathbf{h}(\mathbf{x}_N) + \sum_{k=0}^{N-1} \boldsymbol{\mu}_k^\top \mathbf{h}(\mathbf{x}_k,\mathbf{u}_k) =$$

$$T(\mathbf{x}_N) + \boldsymbol{\mu}_N^\top \mathbf{h}(\mathbf{x}_N) + \boldsymbol{\lambda}_0^\top \bar{\mathbf{x}}_0 - \boldsymbol{\lambda}_N^\top \mathbf{x}_N$$

$$+ \sum_{k=0}^{N-1} \Big(L(\mathbf{x}_k,\mathbf{u}_k) + \boldsymbol{\lambda}_{k+1}^\top \mathbf{f}(\mathbf{x}_k,\mathbf{u}_k) - \boldsymbol{\lambda}_k^\top \mathbf{x}_k + \boldsymbol{\mu}_k^\top \mathbf{h}(\mathbf{x}_k,\mathbf{u}_k) \Big)$$

Define:

$$\mathcal{L}_k(\mathbf{w}_k,\boldsymbol{\lambda},\boldsymbol{\mu}) = L(\mathbf{x}_k,\mathbf{u}_k) + \boldsymbol{\lambda}_{k+1}^\top \mathbf{f}(\mathbf{x}_k,\mathbf{u}_k) - \boldsymbol{\lambda}_k^\top \mathbf{x}_k + \boldsymbol{\mu}_k^\top \mathbf{h}(\mathbf{x}_k,\mathbf{u}_k), \qquad k = 1, \ldots, N-1$$

Separability of the Lagrange function

$$\mathcal{L}(\mathbf{w},\boldsymbol{\lambda},\boldsymbol{\mu}) = T(\mathbf{x}_N) + \sum_{k=0}^{N-1} L(\mathbf{x}_k,\mathbf{u}_k) + \boldsymbol{\lambda}_0^\top \bar{\mathbf{x}}_0 - \boldsymbol{\lambda}_0^\top \mathbf{x}_0 + \sum_{k=0}^{N-1} \boldsymbol{\lambda}_{k+1}^\top \mathbf{f}(\mathbf{x}_k,\mathbf{u}_k) - \sum_{k=0}^{N-1} \boldsymbol{\lambda}_{k+1}^\top \mathbf{x}_{k+1}$$

$$+ \boldsymbol{\mu}_N^\top \mathbf{h}(\mathbf{x}_N) + \sum_{k=0}^{N-1} \boldsymbol{\mu}_k^\top \mathbf{h}(\mathbf{x}_k,\mathbf{u}_k) =$$

$$T(\mathbf{x}_N) + \boldsymbol{\mu}_N^\top \mathbf{h}(\mathbf{x}_N) + \boldsymbol{\lambda}_0^\top \bar{\mathbf{x}}_0 - \boldsymbol{\lambda}_N^\top \mathbf{x}_N$$

$$+ \sum_{k=0}^{N-1} \left(L(\mathbf{x}_k,\mathbf{u}_k) + \boldsymbol{\lambda}_{k+1}^\top \mathbf{f}(\mathbf{x}_k,\mathbf{u}_k) - \boldsymbol{\lambda}_k^\top \mathbf{x}_k + \boldsymbol{\mu}_k^\top \mathbf{h}(\mathbf{x}_k,\mathbf{u}_k) \right)$$

Define:

$$\mathcal{L}_k(\mathbf{w}_k,\boldsymbol{\lambda},\boldsymbol{\mu}) = L(\mathbf{x}_k,\mathbf{u}_k) + \boldsymbol{\lambda}_{k+1}^\top \mathbf{f}(\mathbf{x}_k,\mathbf{u}_k) - \boldsymbol{\lambda}_k^\top \mathbf{x}_k + \boldsymbol{\mu}_k^\top \mathbf{h}(\mathbf{x}_k,\mathbf{u}_k), \qquad k = 1,\dots,N-1$$

$$\mathcal{L}_0(\mathbf{w}_0,\boldsymbol{\lambda},\boldsymbol{\mu}) = L(\mathbf{x}_0,\mathbf{u}_0) + \boldsymbol{\lambda}_1^\top \mathbf{f}(\mathbf{x}_0,\mathbf{u}_0) - \boldsymbol{\lambda}_0^\top \mathbf{x}_0 + \boldsymbol{\mu}_0^\top \mathbf{h}(\mathbf{x}_0,\mathbf{u}_0) + \boldsymbol{\lambda}_0^\top \bar{\mathbf{x}}_0$$

Separability of the Lagrange function

$$\mathcal{L}(\mathbf{w},\boldsymbol{\lambda},\boldsymbol{\mu}) = T(\mathbf{x}_N) + \sum_{k=0}^{N-1} L(\mathbf{x}_k,\mathbf{u}_k) + \boldsymbol{\lambda}_0^\top \bar{\mathbf{x}}_0 - \boldsymbol{\lambda}_0^\top \mathbf{x}_0 + \sum_{k=0}^{N-1} \boldsymbol{\lambda}_{k+1}^\top \mathbf{f}(\mathbf{x}_k,\mathbf{u}_k) - \sum_{k=0}^{N-1} \boldsymbol{\lambda}_{k+1}^\top \mathbf{x}_{k+1}$$

$$+ \boldsymbol{\mu}_N^\top \mathbf{h}(\mathbf{x}_N) + \sum_{k=0}^{N-1} \boldsymbol{\mu}_k^\top \mathbf{h}(\mathbf{x}_k,\mathbf{u}_k) =$$

$$T(\mathbf{x}_N) + \boldsymbol{\mu}_N^\top \mathbf{h}(\mathbf{x}_N) + \boldsymbol{\lambda}_0^\top \bar{\mathbf{x}}_0 - \boldsymbol{\lambda}_N^\top \mathbf{x}_N$$

$$+ \sum_{k=0}^{N-1} \Big(L(\mathbf{x}_k,\mathbf{u}_k) + \boldsymbol{\lambda}_{k+1}^\top \mathbf{f}(\mathbf{x}_k,\mathbf{u}_k) - \boldsymbol{\lambda}_k^\top \mathbf{x}_k + \boldsymbol{\mu}_k^\top \mathbf{h}(\mathbf{x}_k,\mathbf{u}_k) \Big)$$

Define:

$$\mathcal{L}_k(\mathbf{w}_k,\boldsymbol{\lambda},\boldsymbol{\mu}) = L(\mathbf{x}_k,\mathbf{u}_k) + \boldsymbol{\lambda}_{k+1}^\top \mathbf{f}(\mathbf{x}_k,\mathbf{u}_k) - \boldsymbol{\lambda}_k^\top \mathbf{x}_k + \boldsymbol{\mu}_k^\top \mathbf{h}(\mathbf{x}_k,\mathbf{u}_k), \qquad k = 1, \ldots, N-1$$

$$\mathcal{L}_0(\mathbf{w}_0,\boldsymbol{\lambda},\boldsymbol{\mu}) = L(\mathbf{x}_0,\mathbf{u}_0) + \boldsymbol{\lambda}_1^\top \mathbf{f}(\mathbf{x}_0,\mathbf{u}_0) - \boldsymbol{\lambda}_0^\top \mathbf{x}_0 + \boldsymbol{\mu}_0^\top \mathbf{h}(\mathbf{x}_0,\mathbf{u}_0) + \boldsymbol{\lambda}_0^\top \bar{\mathbf{x}}_0$$

$$\mathcal{L}_N(\mathbf{w}_N,\boldsymbol{\lambda},\boldsymbol{\mu}) = T(\mathbf{x}_N) - \boldsymbol{\lambda}_N^\top \mathbf{x}_N + \boldsymbol{\mu}_N^\top \mathbf{h}(\mathbf{x}_N)$$

Separability of the Lagrange function

$$\mathcal{L}(\mathbf{w},\boldsymbol{\lambda},\boldsymbol{\mu}) = T(\mathbf{x}_N) + \sum_{k=0}^{N-1} L(\mathbf{x}_k,\mathbf{u}_k) + \boldsymbol{\lambda}_0^\top \bar{\mathbf{x}}_0 - \boldsymbol{\lambda}_0^\top \mathbf{x}_0 + \sum_{k=0}^{N-1} \boldsymbol{\lambda}_{k+1}^\top \mathbf{f}(\mathbf{x}_k,\mathbf{u}_k) - \sum_{k=0}^{N-1} \boldsymbol{\lambda}_{k+1}^\top \mathbf{x}_{k+1}$$

$$+ \boldsymbol{\mu}_N^\top \mathbf{h}(\mathbf{x}_N) + \sum_{k=0}^{N-1} \boldsymbol{\mu}_k^\top \mathbf{h}(\mathbf{x}_k,\mathbf{u}_k) =$$

$$T(\mathbf{x}_N) + \boldsymbol{\mu}_N^\top \mathbf{h}(\mathbf{x}_N) + \boldsymbol{\lambda}_0^\top \bar{\mathbf{x}}_0 - \boldsymbol{\lambda}_N^\top \mathbf{x}_N$$

$$+ \sum_{k=0}^{N-1} \Big(L(\mathbf{x}_k,\mathbf{u}_k) + \boldsymbol{\lambda}_{k+1}^\top \mathbf{f}(\mathbf{x}_k,\mathbf{u}_k) - \boldsymbol{\lambda}_k^\top \mathbf{x}_k + \boldsymbol{\mu}_k^\top \mathbf{h}(\mathbf{x}_k,\mathbf{u}_k) \Big)$$

Define:

$$\mathcal{L}_k(\mathbf{w}_k,\boldsymbol{\lambda},\boldsymbol{\mu}) = L(\mathbf{x}_k,\mathbf{u}_k) + \boldsymbol{\lambda}_{k+1}^\top \mathbf{f}(\mathbf{x}_k,\mathbf{u}_k) - \boldsymbol{\lambda}_k^\top \mathbf{x}_k + \boldsymbol{\mu}_k^\top \mathbf{h}(\mathbf{x}_k,\mathbf{u}_k), \qquad k = 1,\ldots,N-1$$

$$\mathcal{L}_0(\mathbf{w}_0,\boldsymbol{\lambda},\boldsymbol{\mu}) = L(\mathbf{x}_0,\mathbf{u}_0) + \boldsymbol{\lambda}_1^\top \mathbf{f}(\mathbf{x}_0,\mathbf{u}_0) - \boldsymbol{\lambda}_0^\top \mathbf{x}_0 + \boldsymbol{\mu}_0^\top \mathbf{h}(\mathbf{x}_0,\mathbf{u}_0) + \boldsymbol{\lambda}_0^\top \bar{\mathbf{x}}_0$$

$$\mathcal{L}_N(\mathbf{w}_N,\boldsymbol{\lambda},\boldsymbol{\mu}) = T(\mathbf{x}_N) - \boldsymbol{\lambda}_N^\top \mathbf{x}_N + \boldsymbol{\mu}_N^\top \mathbf{h}(\mathbf{x}_N)$$

Then use $\mathbf{w}_k = \{\mathbf{x}_k, \mathbf{u}_k\}$ for $k = 0,\ldots,N-1$, and $\mathbf{w}_N \equiv \mathbf{x}_N$, so that

$$\boxed{\mathcal{L}(\mathbf{w},\boldsymbol{\lambda},\boldsymbol{\mu}) = \sum_{k=0}^{N} \mathcal{L}_k(\mathbf{w}_k,\boldsymbol{\lambda},\boldsymbol{\mu})}$$

QP structure from Multiple-Shooting

NLP:

$$\min_{\mathbf{w}} \quad \phi(\mathbf{w})$$

$$\text{s.t.} \quad \mathbf{g}(\mathbf{w}) = \left[\begin{array}{c} \bar{\mathbf{x}}_0 - \mathbf{x}_0 \\ \mathbf{f}(\mathbf{x}_0, \mathbf{u}_0) - \mathbf{x}_1 \\ \dots \\ \mathbf{f}(\mathbf{x}_{N-1}, \mathbf{u}_{N-1}) - \mathbf{x}_N \end{array}\right] = 0$$

$$\mathbf{h}(\mathbf{w}) = \left[\begin{array}{c} \mathbf{h}(\mathbf{x}_0, \mathbf{u}_0) \\ \dots \\ \mathbf{h}(\mathbf{x}_{N-1}, \mathbf{u}_{N-1}) \\ \mathbf{h}(\mathbf{x}_N) \end{array}\right] \leq 0$$

SQP recursively solves the QPs:

$$\begin{aligned} \min_{\Delta\mathbf{w}} \quad & \frac{1}{2}\Delta\mathbf{w}^\top H \Delta\mathbf{w} + \nabla\Phi^\top \Delta\mathbf{w} \\ \text{s.t.} \quad & \nabla\mathbf{g}^\top \Delta\mathbf{w} + \mathbf{g} = 0 \\ & \nabla\mathbf{h}^\top \Delta\mathbf{w} + \mathbf{h} \leq 0 \end{aligned}$$

QP structure from Multiple-Shooting

NLP:

$$\min_{\mathbf{w}} \quad \phi(\mathbf{w})$$

$$\text{s.t.} \quad \mathbf{g}(\mathbf{w}) = \begin{bmatrix} \bar{\mathbf{x}}_0 - \mathbf{x}_0 \\ \mathbf{f}(\mathbf{x}_0, \mathbf{u}_0) - \mathbf{x}_1 \\ \dots \\ \mathbf{f}(\mathbf{x}_{N-1}, \mathbf{u}_{N-1}) - \mathbf{x}_N \end{bmatrix} = 0$$

$$\mathbf{h}(\mathbf{w}) = \begin{bmatrix} \mathbf{h}(\mathbf{x}_0, \mathbf{u}_0) \\ \dots \\ \mathbf{h}(\mathbf{x}_{N-1}, \mathbf{u}_{N-1}) \\ \mathbf{h}(\mathbf{x}_N) \end{bmatrix} \leq 0$$

SQP recursively solves the QPs:

$$\min_{\Delta\mathbf{w}} \quad \frac{1}{2}\Delta\mathbf{w}^\top H \Delta\mathbf{w} + \nabla\Phi^\top \Delta\mathbf{w}$$

$$\text{s.t.} \quad \nabla\mathbf{g}^\top \Delta\mathbf{w} + \mathbf{g} = 0$$

$$\nabla\mathbf{h}^\top \Delta\mathbf{w} + \mathbf{h} \leq 0$$

Let's have a look at matrices H, $\nabla\mathbf{g}^\top$ and $\nabla\mathbf{h}^\top$ for this specific type of NLP

Constraints Jacobian - Dynamics

Constraints: $$\mathbf{g}(\mathbf{w}) = \begin{bmatrix} \mathbf{c}(\mathbf{x}_0) \\ \mathbf{f}(\mathbf{x}_0, \mathbf{u}_0) - \mathbf{x}_1 \\ \mathbf{f}(\mathbf{x}_1, \mathbf{u}_1) - \mathbf{x}_2 \\ \mathbf{f}(\mathbf{x}_2, \mathbf{u}_2) - \mathbf{x}_3 \\ \mathbf{f}(\mathbf{x}_3, \mathbf{u}_3) - \mathbf{x}_4 \\ \mathbf{f}(\mathbf{x}_4, \mathbf{u}_4) - \mathbf{x}_5 \end{bmatrix} \quad \text{with} \quad \mathbf{w} = \begin{bmatrix} \mathbf{x}_0 \\ \mathbf{u}_0 \\ \mathbf{x}_1 \\ \mathbf{u}_1 \\ \mathbf{x}_2 \\ \mathbf{u}_2 \\ \mathbf{x}_3 \\ \mathbf{u}_3 \\ \mathbf{x}_4 \\ \mathbf{u}_4 \\ \mathbf{x}_5 \end{bmatrix}$$

Constraints Jacobian - Dynamics

$$\text{Constraints:} \quad \mathbf{g}(\mathbf{w}) = \begin{bmatrix} \mathbf{c}(\mathbf{x}_0) \\ \mathbf{f}(\mathbf{x}_0, \mathbf{u}_0) - \mathbf{x}_1 \\ \mathbf{f}(\mathbf{x}_1, \mathbf{u}_1) - \mathbf{x}_2 \\ \mathbf{f}(\mathbf{x}_2, \mathbf{u}_2) - \mathbf{x}_3 \\ \mathbf{f}(\mathbf{x}_3, \mathbf{u}_3) - \mathbf{x}_4 \\ \mathbf{f}(\mathbf{x}_4, \mathbf{u}_4) - \mathbf{x}_5 \end{bmatrix} \quad \text{with} \quad \mathbf{w} = \begin{bmatrix} \mathbf{x}_0 \\ \mathbf{u}_0 \\ \mathbf{x}_1 \\ \mathbf{u}_1 \\ \mathbf{x}_2 \\ \mathbf{u}_2 \\ \mathbf{x}_3 \\ \mathbf{u}_3 \\ \mathbf{x}_4 \\ \mathbf{u}_4 \\ \mathbf{x}_5 \end{bmatrix}$$

Let's denote $\mathbf{f}_k = \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k)$, then the constraints derivative reads as:

$$\nabla \mathbf{g}(\mathbf{w})^{\mathrm{T}} = \begin{bmatrix} \frac{\partial \mathbf{c}}{\partial \mathbf{x}_0} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{\partial \mathbf{f}_0}{\partial \mathbf{x}_0} & \frac{\partial \mathbf{f}_0}{\partial \mathbf{u}_0} & -I & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{\partial \mathbf{f}_1}{\partial \mathbf{x}_1} & \frac{\partial \mathbf{f}_1}{\partial \mathbf{u}_1} & -I & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{\partial \mathbf{f}_2}{\partial \mathbf{x}_2} & \frac{\partial \mathbf{f}_2}{\partial \mathbf{u}_2} & -I & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{\partial \mathbf{f}_3}{\partial \mathbf{x}_3} & \frac{\partial \mathbf{f}_3}{\partial \mathbf{u}_3} & -I & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{\partial \mathbf{f}_4}{\partial \mathbf{x}_4} & \frac{\partial \mathbf{f}_4}{\partial \mathbf{u}_4} & -I \end{bmatrix}$$

Constraints Jacobian - Dynamics

Constraints: $$\mathbf{g}(\mathbf{w}) = \begin{bmatrix} \mathbf{c}(\mathbf{x}_0) \\ \mathbf{f}(\mathbf{x}_0, \mathbf{u}_0) - \mathbf{x}_1 \\ \mathbf{f}(\mathbf{x}_1, \mathbf{u}_1) - \mathbf{x}_2 \\ \mathbf{f}(\mathbf{x}_2, \mathbf{u}_2) - \mathbf{x}_3 \\ \mathbf{f}(\mathbf{x}_3, \mathbf{u}_3) - \mathbf{x}_4 \\ \mathbf{f}(\mathbf{x}_4, \mathbf{u}_4) - \mathbf{x}_5 \end{bmatrix} \quad \text{with} \quad \mathbf{w} = \begin{bmatrix} \mathbf{x}_0 \\ \mathbf{u}_0 \\ \mathbf{x}_1 \\ \mathbf{u}_1 \\ \mathbf{x}_2 \\ \mathbf{u}_2 \\ \mathbf{x}_3 \\ \mathbf{u}_3 \\ \mathbf{x}_4 \\ \mathbf{u}_4 \\ \mathbf{x}_5 \end{bmatrix}$$

Let's denote $\mathbf{f}_k = \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k)$, then the constraints derivative reads as:

$$\nabla \mathbf{g}(\mathbf{w})^{\mathrm{T}} = \begin{bmatrix} \frac{\partial \mathbf{c}}{\partial \mathbf{x}_0} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{\partial \mathbf{f}_0}{\partial \mathbf{x}_0} & \frac{\partial \mathbf{f}_0}{\partial \mathbf{u}_0} & -I & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{\partial \mathbf{f}_1}{\partial \mathbf{x}_1} & \frac{\partial \mathbf{f}_1}{\partial \mathbf{u}_1} & -I & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{\partial \mathbf{f}_2}{\partial \mathbf{x}_2} & \frac{\partial \mathbf{f}_2}{\partial \mathbf{u}_2} & -I & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{\partial \mathbf{f}_3}{\partial \mathbf{x}_3} & \frac{\partial \mathbf{f}_3}{\partial \mathbf{u}_3} & -I & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{\partial \mathbf{f}_4}{\partial \mathbf{x}_4} & \frac{\partial \mathbf{f}_4}{\partial \mathbf{u}_4} & -I \end{bmatrix}$$

Observe the **banded** structure of the **Jacobian** $\nabla \mathbf{g}(\mathbf{w})^{\mathrm{T}}$.
Note that this structure hinges on the ordering in $\mathbf{w}$ and $\mathbf{g}$!!!

Constraints Jacobian - Bounds

$$\text{Bounds:} \quad \mathbf{h}(\mathbf{w}) = \begin{bmatrix} \mathbf{h}_0(\mathbf{x}_0, \mathbf{u}_0) \\ \mathbf{h}_1(\mathbf{x}_1, \mathbf{u}_1) \\ \mathbf{h}_2(\mathbf{x}_2, \mathbf{u}_2) \\ \mathbf{h}_3(\mathbf{x}_3, \mathbf{u}_3) \\ \mathbf{h}_4(\mathbf{x}_4, \mathbf{u}_4) \\ \mathbf{h}_5(\mathbf{x}_5) \end{bmatrix} \quad \text{with} \quad \mathbf{w} = \begin{bmatrix} \mathbf{x}_0 \\ \mathbf{u}_0 \\ \mathbf{x}_1 \\ \mathbf{u}_1 \\ \mathbf{x}_2 \\ \mathbf{u}_2 \\ \mathbf{x}_3 \\ \mathbf{u}_3 \\ \mathbf{x}_4 \\ \mathbf{u}_4 \\ \mathbf{x}_5 \end{bmatrix}$$

Constraints Jacobian - Bounds

Bounds: $\mathbf{h}(\mathbf{w}) = \begin{bmatrix} \mathbf{h}_0(\mathbf{x}_0, \mathbf{u}_0) \\ \mathbf{h}_1(\mathbf{x}_1, \mathbf{u}_1) \\ \mathbf{h}_2(\mathbf{x}_2, \mathbf{u}_2) \\ \mathbf{h}_3(\mathbf{x}_3, \mathbf{u}_3) \\ \mathbf{h}_4(\mathbf{x}_4, \mathbf{u}_4) \\ \mathbf{h}_5(\mathbf{x}_5) \end{bmatrix}$ with $\mathbf{w} = \begin{bmatrix} \mathbf{x}_0 \\ \mathbf{u}_0 \\ \mathbf{x}_1 \\ \mathbf{u}_1 \\ \mathbf{x}_2 \\ \mathbf{u}_2 \\ \mathbf{x}_3 \\ \mathbf{u}_3 \\ \mathbf{x}_4 \\ \mathbf{u}_4 \\ \mathbf{x}_5 \end{bmatrix}$

Then:

$$\frac{\partial \mathbf{h}}{\partial \mathbf{w}} = \begin{bmatrix} \frac{\partial \mathbf{h}_0}{\partial \mathbf{x}_0} & \frac{\partial \mathbf{h}_0}{\partial \mathbf{u}_0} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{\partial \mathbf{h}_1}{\partial \mathbf{x}_1} & \frac{\partial \mathbf{h}_1}{\partial \mathbf{u}_1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{\partial \mathbf{h}_2}{\partial \mathbf{x}_2} & \frac{\partial \mathbf{h}_2}{\partial \mathbf{u}_2} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{\partial \mathbf{h}_3}{\partial \mathbf{x}_3} & \frac{\partial \mathbf{h}_3}{\partial \mathbf{u}_3} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{\partial \mathbf{h}_4}{\partial \mathbf{x}_4} & \frac{\partial \mathbf{h}_4}{\partial \mathbf{u}_4} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{\partial \mathbf{h}_5}{\partial \mathbf{x}_5} \end{bmatrix}$$

Constraints Jacobian - Bounds

$$\text{Bounds:}\quad \mathbf{h}(\mathbf{w}) = \begin{bmatrix} \mathbf{h}_0(\mathbf{x}_0, \mathbf{u}_0) \\ \mathbf{h}_1(\mathbf{x}_1, \mathbf{u}_1) \\ \mathbf{h}_2(\mathbf{x}_2, \mathbf{u}_2) \\ \mathbf{h}_3(\mathbf{x}_3, \mathbf{u}_3) \\ \mathbf{h}_4(\mathbf{x}_4, \mathbf{u}_4) \\ \mathbf{h}_5(\mathbf{x}_5) \end{bmatrix} \quad \text{with} \quad \mathbf{w} = \begin{bmatrix} \mathbf{x}_0 \\ \mathbf{u}_0 \\ \mathbf{x}_1 \\ \mathbf{u}_1 \\ \mathbf{x}_2 \\ \mathbf{u}_2 \\ \mathbf{x}_3 \\ \mathbf{u}_3 \\ \mathbf{x}_4 \\ \mathbf{u}_4 \\ \mathbf{x}_5 \end{bmatrix}$$

Then:

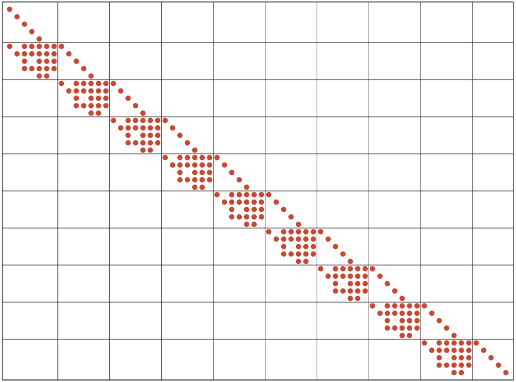
$$\frac{\partial \mathbf{h}}{\partial \mathbf{w}} = \begin{bmatrix} \frac{\partial \mathbf{h}_0}{\partial \mathbf{x}_0} & \frac{\partial \mathbf{h}_0}{\partial \mathbf{u}_0} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{\partial \mathbf{h}_1}{\partial \mathbf{x}_1} & \frac{\partial \mathbf{h}_1}{\partial \mathbf{u}_1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{\partial \mathbf{h}_2}{\partial \mathbf{x}_2} & \frac{\partial \mathbf{h}_2}{\partial \mathbf{u}_2} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{\partial \mathbf{h}_3}{\partial \mathbf{x}_3} & \frac{\partial \mathbf{h}_3}{\partial \mathbf{u}_3} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{\partial \mathbf{h}_4}{\partial \mathbf{x}_4} & \frac{\partial \mathbf{h}_4}{\partial \mathbf{u}_4} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{\partial \mathbf{h}_5}{\partial \mathbf{x}_5} \end{bmatrix}$$

Observe the **banded** structure of the **Jacobian** $\nabla \mathbf{h}(\mathbf{w})^{\mathrm{T}}$.
Note that this structure hinges on the ordering in $\mathbf{w}$ and $\mathbf{g}$!!!

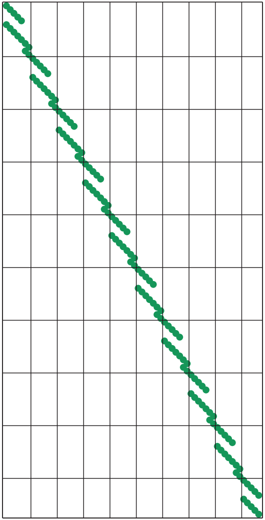
Constraints Jacobian sparsity pattern - Illustration

$$\begin{aligned} \min_{\Delta \mathbf{w}} \quad & \frac{1}{2}\Delta \mathbf{w}^\top B \Delta \mathbf{w} + \nabla \Phi^\top \Delta \mathbf{w} \\ \text{s.t.} \quad & \nabla \mathbf{g}^\top \Delta \mathbf{w} + \mathbf{g} = 0 \\ & \nabla \mathbf{h}^\top \Delta \mathbf{w} + \mathbf{h} \leq 0 \end{aligned}$$

$\nabla \mathbf{g}(\mathbf{w})^\top$



$\nabla \mathbf{h}(\mathbf{w})^\top$



Sparsity of the exact Hessian

Separability of the Lagrange function

$$\mathcal{L}\left(\mathbf{w}, \boldsymbol{\lambda}, \boldsymbol{\mu}\right) = \sum_{k=0}^{N} \mathcal{L}_k\left(\mathbf{w}_k, \boldsymbol{\lambda}, \boldsymbol{\mu}\right)$$

where $\mathbf{w}_k = \{\mathbf{x}_k, \mathbf{u}_k\}$ for $k = 0, ..., N-1$, and $\mathbf{w}_N \equiv \mathbf{x}_N$.

Sparsity of the exact Hessian

Separability of the Lagrange function

$$\mathcal{L}\left(\mathbf{w}, \boldsymbol{\lambda}, \boldsymbol{\mu}\right) = \sum_{k=0}^{N} \mathcal{L}_k\left(\mathbf{w}_k, \boldsymbol{\lambda}, \boldsymbol{\mu}\right)$$

where $\mathbf{w}_k = \{\mathbf{x}_k, \mathbf{u}_k\}$ for $k = 0, ..., N-1$, and $\mathbf{w}_N \equiv \mathbf{x}_N$. Hence:

$$\frac{\partial^2 \mathcal{L}\left(\mathbf{w}, \boldsymbol{\lambda}, \boldsymbol{\mu}\right)}{\partial \mathbf{w}_i \partial \mathbf{w}_j} = \sum_{k=0}^{N} \frac{\partial^2 \mathcal{L}_k\left(\mathbf{w}_k, \boldsymbol{\lambda}, \boldsymbol{\mu}\right)}{\partial \mathbf{w}_i \partial \mathbf{w}_j} = 0, \quad \forall i \neq j$$

Sparsity of the exact Hessian

Separability of the Lagrange function

$$\mathcal{L}(\mathbf{w}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = \sum_{k=0}^{N} \mathcal{L}_k(\mathbf{w}_k, \boldsymbol{\lambda}, \boldsymbol{\mu})$$

where $\mathbf{w}_k = \{\mathbf{x}_k, \mathbf{u}_k\}$ for $k = 0, ..., N-1$, and $\mathbf{w}_N \equiv \mathbf{x}_N$. Hence:

$$\frac{\partial^2 \mathcal{L}(\mathbf{w}, \boldsymbol{\lambda}, \boldsymbol{\mu})}{\partial \mathbf{w}_i \partial \mathbf{w}_j} = \sum_{k=0}^{N} \frac{\partial^2 \mathcal{L}_k(\mathbf{w}_k, \boldsymbol{\lambda}, \boldsymbol{\mu})}{\partial \mathbf{w}_i \partial \mathbf{w}_j} = 0, \quad \forall i \neq j$$

Hence the Hessian is **block diagonal**, i.e.

$$\nabla^2_{\mathbf{w}} \mathcal{L}(\mathbf{w}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = \begin{bmatrix} \nabla^2_{\mathbf{w}_0} \mathcal{L}_0(\mathbf{w}_0, \boldsymbol{\lambda}, \boldsymbol{\mu}) & 0 & 0 & 0 \\ 0 & \nabla^2_{\mathbf{w}_1} \mathcal{L}_1(\mathbf{w}_1, \boldsymbol{\lambda}, \boldsymbol{\mu}) & 0 & 0 \\ 0 & 0 & \ldots & 0 \\ 0 & 0 & 0 & \nabla^2_{\mathbf{w}_N} \mathcal{L}_N(\mathbf{w}_N, \boldsymbol{\lambda}, \boldsymbol{\mu}) \end{bmatrix}$$

Sparsity of the exact Hessian

Separability of the Lagrange function

$$\mathcal{L}\left(\mathbf{w},\boldsymbol{\lambda},\boldsymbol{\mu}\right)=\sum_{k=0}^{N}\mathcal{L}_k\left(\mathbf{w}_k,\boldsymbol{\lambda},\boldsymbol{\mu}\right)$$

where $\mathbf{w}_k=\{\mathbf{x}_k,\mathbf{u}_k\}$ for $k=0,...,N-1$, and $\mathbf{w}_N\equiv\mathbf{x}_N$. Hence:

$$\frac{\partial^2\mathcal{L}\left(\mathbf{w},\boldsymbol{\lambda},\boldsymbol{\mu}\right)}{\partial\mathbf{w}_i\partial\mathbf{w}_j}=\sum_{k=0}^{N}\frac{\partial^2\mathcal{L}_k\left(\mathbf{w}_k,\boldsymbol{\lambda},\boldsymbol{\mu}\right)}{\partial\mathbf{w}_i\partial\mathbf{w}_j}=0,\quad\forall i\neq j$$

Hence the Hessian is **block diagonal**, i.e.

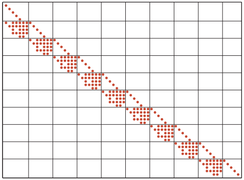
$$\nabla^2_{\mathbf{w}}\mathcal{L}\left(\mathbf{w},\boldsymbol{\lambda},\boldsymbol{\mu}\right)=\begin{bmatrix}\nabla^2_{\mathbf{w}_0}\mathcal{L}_0\left(\mathbf{w}_0,\boldsymbol{\lambda},\boldsymbol{\mu}\right) & 0 & 0 & 0\\ 0 & \nabla^2_{\mathbf{w}_1}\mathcal{L}_1\left(\mathbf{w}_1,\boldsymbol{\lambda},\boldsymbol{\mu}\right) & 0 & 0\\ 0 & 0 & ... & 0\\ 0 & 0 & 0 & \nabla^2_{\mathbf{w}_N}\mathcal{L}_N\left(\mathbf{w}_N,\boldsymbol{\lambda},\boldsymbol{\mu}\right)\end{bmatrix}$$

Careful: I assumed ϕ and $\mathbf{h}$ are *stage-wise*

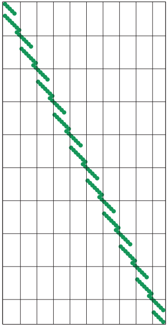
Sparsity pattern - Illustration

$$
\begin{aligned}
\min_{\Delta \mathbf{w}} \quad & \frac{1}{2}\Delta \mathbf{w}^\top B \Delta \mathbf{w} + \nabla \Phi^\top \Delta \mathbf{w} \\
\text{s.t.} \quad & \nabla \mathbf{g}^\top \Delta \mathbf{w} + \mathbf{g} = 0 \\
& \nabla \mathbf{h}^\top \Delta \mathbf{w} + \mathbf{h} \leq 0
\end{aligned}
$$

$\nabla \mathbf{g}(\mathbf{w})^\top$



$\nabla \mathbf{h}(\mathbf{w})^\top$



$B \equiv \nabla^2_{\mathbf{w}} \mathcal{L}(\mathbf{w}, \boldsymbol{\lambda})$