# Numerical Optimal Control
# From linear MPC to real-time NMPC

Sébastien Gros

ITK, NTNU

NTNU PhD course

# Outline

# NMPC

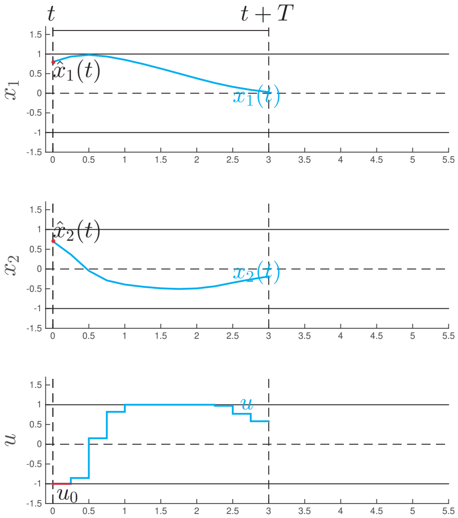
**Looking ahead in the future**...

...design a control sequence minimizing a penalty function

## NMPC problem

$$P\left(\hat{\mathbf{x}}(t)\right): \min_{u,s} \quad \int_{t}^{t+T} L\left(\mathbf{x},\mathbf{u}\right) d\tau$$
$$\text{s.t.} \quad \dot{\mathbf{x}} = \mathbf{F}\left(\mathbf{x},\mathbf{u}\right), \quad \mathbf{x}\left(t\right) = \hat{\mathbf{x}}(t)$$
$$\mathbf{h}\left(\mathbf{x},\mathbf{u}\right) \leq 0$$

- Inputs discretized (z.o.h.) over an ad-hoc time grid $t_0, ..., t_N$



# NMPC

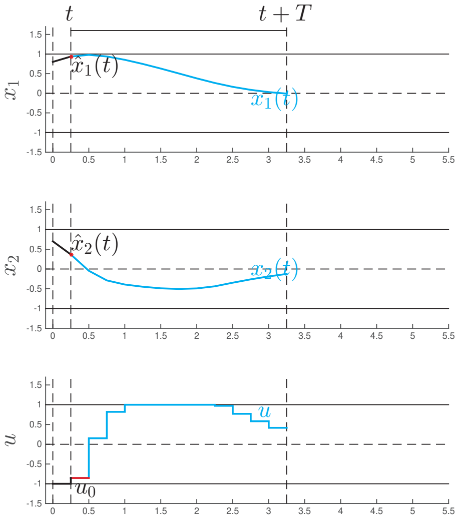
**Looking ahead in the future**...

...design a control sequence minimizing a penalty function

## NMPC problem

$$P\left(\hat{\mathbf{x}}(t)\right):\ \min_{u,s} \quad \int_t^{t+T} L\left(\mathbf{x},\mathbf{u}\right) d\tau$$
$$\text{s.t.} \quad \dot{\mathbf{x}} = \mathbf{F}\left(\mathbf{x},\mathbf{u}\right), \quad \mathbf{x}\left(t\right) = \hat{\mathbf{x}}(t)$$
$$\mathbf{h}\left(\mathbf{x},\mathbf{u}\right) \leq 0$$

- Inputs discretized (z.o.h.) over an ad-hoc time grid $t_0, ..., t_N$



# NMPC

**Looking ahead in the future**...

...design a control sequence minimizing a penalty function

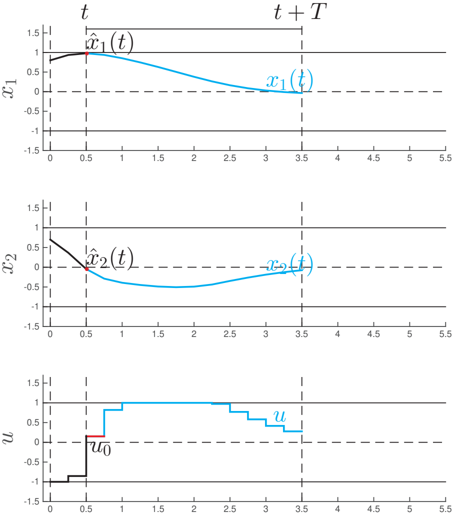
## NMPC problem

$$P\left(\hat{\mathbf{x}}(t)\right): \min_{u,s} \quad \int_t^{t+T} L\left(\mathbf{x},\mathbf{u}\right) d\tau$$

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

$$\mathbf{h}\left(\mathbf{x},\mathbf{u}\right) \leq 0$$

- Inputs discretized (z.o.h.) over an ad-hoc time grid $t_0, ..., t_N$



# NMPC

**Looking ahead in the future**...

...design a control sequence minimizing a penalty function

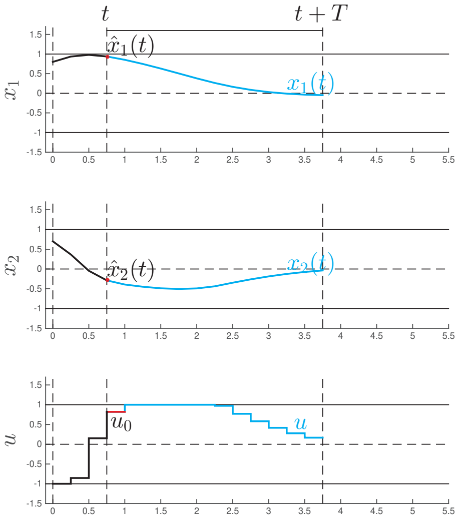
## NMPC problem

$$P\left(\hat{\mathbf{x}}(t)\right): \min_{u,s} \quad \int_t^{t+T} L\left(\mathbf{x},\mathbf{u}\right) d\tau$$

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

$$\mathbf{h}\left(\mathbf{x},\mathbf{u}\right) \leq 0$$

- Inputs discretized (z.o.h.) over an ad-hoc time grid $t_0, ..., t_N$



# NMPC

**Looking ahead in the future**...

...design a control sequence minimizing a penalty function

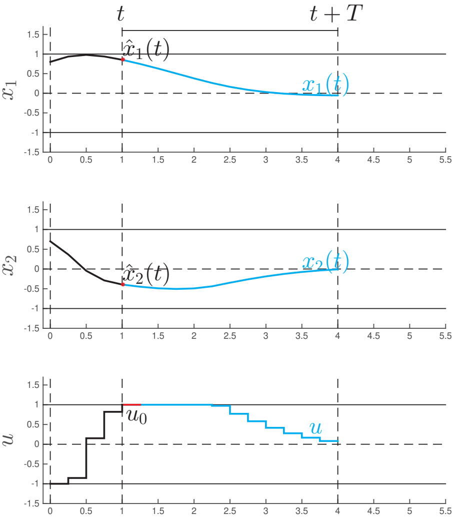
## NMPC problem

$$P\left(\hat{\mathbf{x}}(t)\right): \min_{u,s} \quad \int_{t}^{t+T} L\left(\mathbf{x},\mathbf{u}\right) d\tau$$

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

$$\mathbf{h}\left(\mathbf{x},\mathbf{u}\right) \leq 0$$

- Inputs discretized (z.o.h.) over an ad-hoc time grid $t_0, ..., t_N$



# NMPC

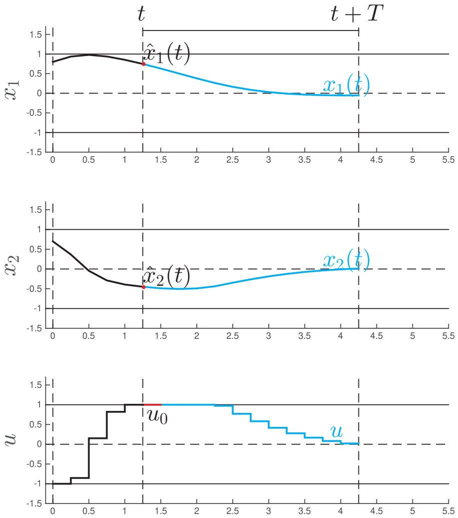
**Looking ahead in the future...**

...design a control sequence minimizing a penalty function

## NMPC problem

$$P\left(\hat{\mathbf{x}}(t)\right): \min_{u,s} \quad \int_t^{t+T} L\left(\mathbf{x},\mathbf{u}\right) d\tau$$
$$\text{s.t.} \quad \dot{\mathbf{x}} = \mathbf{F}\left(\mathbf{x},\mathbf{u}\right), \quad \mathbf{x}\left(t\right) = \hat{\mathbf{x}}(t)$$
$$\mathbf{h}\left(\mathbf{x},\mathbf{u}\right) \leq 0$$

- Inputs discretized (z.o.h.) over an ad-hoc time grid $t_0, ..., t_N$



# NMPC

**Looking ahead in the future**...

...design a control sequence minimizing a penalty function

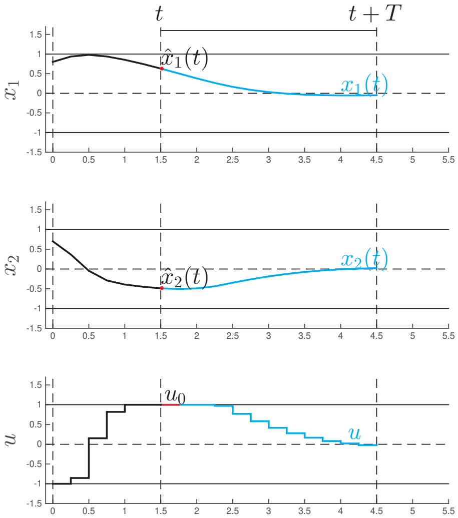
## NMPC problem

$$P\left(\hat{\mathbf{x}}(t)\right): \min_{u,s} \quad \int_t^{t+T} L\left(\mathbf{x},\mathbf{u}\right) d\tau$$

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

$$\mathbf{h}\left(\mathbf{x},\mathbf{u}\right) \leq 0$$

- Inputs discretized (z.o.h.) over an ad-hoc time grid $t_0, ..., t_N$



# NMPC

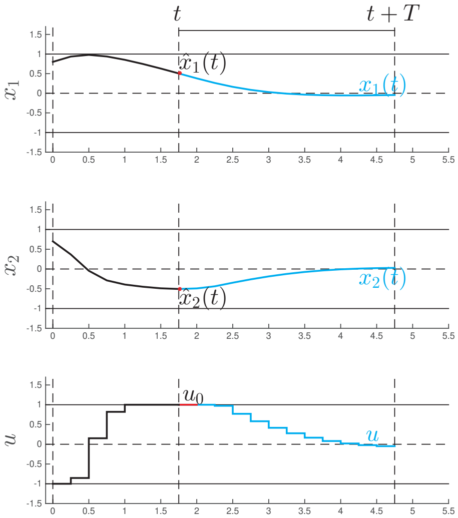
**Looking ahead in the future...**

...design a control sequence minimizing a penalty function

## NMPC problem

$$P\left(\hat{\mathbf{x}}(t)\right): \min_{u,s} \quad \int_t^{t+T} L\left(\mathbf{x},\mathbf{u}\right) d\tau$$
$$\text{s.t.} \quad \dot{\mathbf{x}} = \mathbf{F}\left(\mathbf{x},\mathbf{u}\right), \quad \mathbf{x}\left(t\right) = \hat{\mathbf{x}}(t)$$
$$\mathbf{h}\left(\mathbf{x},\mathbf{u}\right) \leq 0$$

- Inputs discretized (z.o.h.) over an ad-hoc time grid $t_0, ..., t_N$



# NMPC

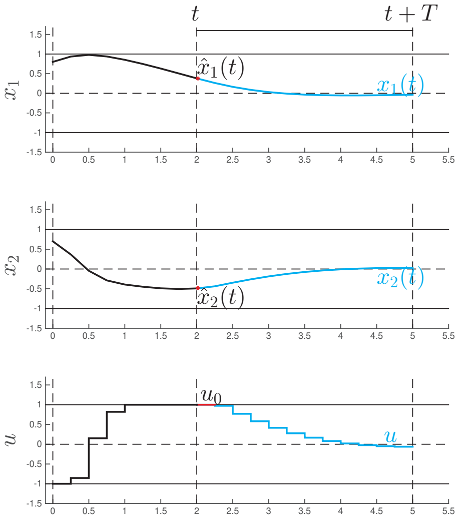
**Looking ahead in the future...**

...design a control sequence minimizing a penalty function

## NMPC problem

$$
\begin{aligned}
P\left(\hat{\mathbf{x}}(t)\right):\ \min_{u,s} \quad & \int_t^{t+T} L\left(\mathbf{x},\mathbf{u}\right) d\tau \\
\text{s.t.} \quad & \dot{\mathbf{x}} = \mathbf{F}\left(\mathbf{x},\mathbf{u}\right), \quad \mathbf{x}\left(t\right) = \hat{\mathbf{x}}(t) \\
& \mathbf{h}\left(\mathbf{x},\mathbf{u}\right) \leq 0
\end{aligned}
$$

- Inputs discretized (z.o.h.) over an ad-hoc time grid $t_0, ..., t_N$



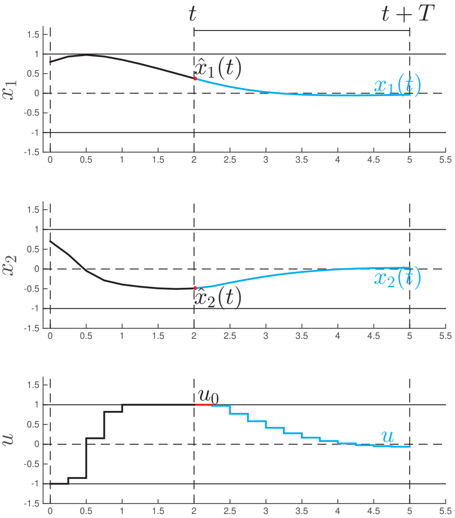
# NMPC

**Looking ahead in the future**...

...design a control sequence minimizing a penalty function

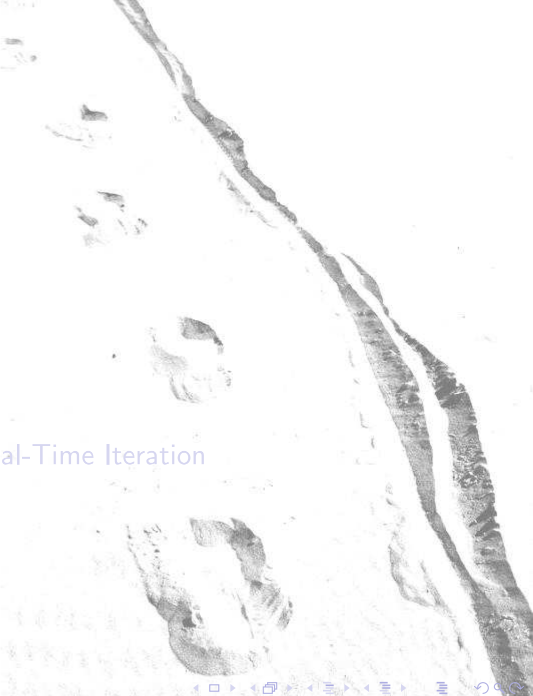
## NMPC problem

$$P\left(\hat{\mathbf{x}}(t)\right): \min_{u,s} \quad \int_t^{t+T} L\left(\mathbf{x},\mathbf{u}\right) d\tau$$
$$\text{s.t.} \quad \dot{\mathbf{x}} = \mathbf{F}\left(\mathbf{x},\mathbf{u}\right), \quad \mathbf{x}\left(t\right) = \hat{\mathbf{x}}(t)$$
$$\mathbf{h}\left(\mathbf{x},\mathbf{u}\right) \leq 0$$




- Inputs discretized (z.o.h.) over an ad-hoc time grid $t_0, ..., t_N$
- Alternative formulations:
  - Terminal cost
  - Terminal constraint

# Outline



# Parametric Embedding (cheap trick to code pred.-corr.)

Generic parametric NLPs:

$$\begin{aligned} \mathrm{P}\left(\mathbf{p}\right):\ \min_{\mathbf{w}} \quad & \Phi\left(\mathbf{w},\mathbf{p}\right) \\ \text{s.t.} \quad & \mathbf{g}\left(\mathbf{w},\mathbf{p}\right) = 0 \\ & \mathbf{h}\left(\mathbf{w},\mathbf{p}\right) \leq 0 \end{aligned}$$

Parametric embedding :

$$\begin{aligned} \mathrm{P}_{\mathrm{E}}\left(\mathbf{p}\right):\ \min_{\mathbf{w},\boldsymbol{\theta}} \quad & \Phi\left(\mathbf{w},\boldsymbol{\theta}\right) \\ \text{s.t.} \quad & \mathbf{g}\left(\mathbf{w},\boldsymbol{\theta}\right) = 0 \\ & \mathbf{h}\left(\mathbf{w},\boldsymbol{\theta}\right) \leq 0 \\ & \mathbf{p} - \boldsymbol{\theta} = 0 \end{aligned}$$

**What difference does this make?!?** Consider QP for $\mathrm{P}_{\mathrm{E}}\left(\mathbf{p}\right)$:

# Parametric Embedding (cheap trick to code pred.-corr.)

Generic parametric NLPs:

$$\begin{aligned} \mathrm{P}\left(\mathbf{p}\right): \min_{\mathbf{w}} \quad & \Phi\left(\mathbf{w},\mathbf{p}\right) \\ \text{s.t.} \quad & \mathbf{g}\left(\mathbf{w},\mathbf{p}\right) = 0 \\ & \mathbf{h}\left(\mathbf{w},\mathbf{p}\right) \leq 0 \end{aligned}$$

Parametric embedding :

$$\begin{aligned} \mathrm{P}_{\mathrm{E}}\left(\mathbf{p}\right): \min_{\mathbf{w},\boldsymbol{\theta}} \quad & \Phi\left(\mathbf{w},\boldsymbol{\theta}\right) \\ \text{s.t.} \quad & \mathbf{g}\left(\mathbf{w},\boldsymbol{\theta}\right) = 0 \\ & \mathbf{h}\left(\mathbf{w},\boldsymbol{\theta}\right) \leq 0 \\ & \mathbf{p} - \boldsymbol{\theta} = 0 \end{aligned}$$

**What difference does this make?!?** Consider QP for $\mathrm{P}_{\mathrm{E}}\left(\mathbf{p}\right)$:

# Parametric Embedding (cheap trick to code pred.-corr.)

Generic parametric NLPs:

$$\begin{aligned} \mathrm{P}\left(\mathbf{p}\right): \min_{\mathbf{w}} \quad & \Phi\left(\mathbf{w},\mathbf{p}\right) \\ \text{s.t.} \quad & \mathbf{g}\left(\mathbf{w},\mathbf{p}\right) = 0 \\ & \mathbf{h}\left(\mathbf{w},\mathbf{p}\right) \le 0 \end{aligned}$$

Parametric embedding :

$$\begin{aligned} \mathrm{P}_{\mathrm{E}}\left(\mathbf{p}\right): \min_{\mathbf{w},\boldsymbol{\theta}} \quad & \Phi\left(\mathbf{w},\boldsymbol{\theta}\right) \\ \text{s.t.} \quad & \mathbf{g}\left(\mathbf{w},\boldsymbol{\theta}\right) = 0 \\ & \mathbf{h}\left(\mathbf{w},\boldsymbol{\theta}\right) \le 0 \\ & \mathbf{p} - \boldsymbol{\theta} = 0 \end{aligned}$$

**What difference does this make?!?** Consider QP for $\mathrm{P}_{\mathrm{E}}\left(\mathbf{p}\right)$:

$$\begin{aligned} \min_{\Delta\mathbf{w},\Delta\boldsymbol{\theta}} \quad & \frac{1}{2}\Delta\mathbf{w}^\top H \Delta\mathbf{w} + \nabla_{\mathbf{w}}\Phi^\top\Delta\mathbf{w} + \nabla_{\boldsymbol{\theta}}\Phi^\top\Delta\boldsymbol{\theta} + \Delta\boldsymbol{\theta}^\top\nabla_{\mathbf{pw}}\mathcal{L}\Delta\mathbf{w} + \frac{1}{2}\Delta\boldsymbol{\theta}^\top\nabla_{\mathbf{pp}}\mathcal{L}\Delta\boldsymbol{\theta} \\ & \mathbf{g} + \nabla_{\mathbf{w}}\mathbf{g}^\top\Delta\mathbf{w} + \nabla_{\mathbf{p}}\mathbf{g}^\top\Delta\boldsymbol{\theta} = 0 \\ & \mathbf{h} + \nabla_{\mathbf{w}}\mathbf{h}^\top\Delta\mathbf{w} + \nabla_{\mathbf{p}}\mathbf{h}^\top\Delta\boldsymbol{\theta} \le 0 \\ & \mathbf{p} - \boldsymbol{\theta} - \Delta\boldsymbol{\theta} = 0 \end{aligned}$$

# Parametric Embedding (cheap trick to code pred.-corr.)

Generic parametric NLPs:

$$\begin{aligned} \mathrm{P}(\mathbf{p}): \min_{\mathbf{w}} \quad & \Phi(\mathbf{w},\mathbf{p}) \\ \text{s.t.} \quad & \mathbf{g}(\mathbf{w},\mathbf{p}) = 0 \\ & \mathbf{h}(\mathbf{w},\mathbf{p}) \leq 0 \end{aligned}$$

Parametric embedding :

$$\begin{aligned} \mathrm{P}_{\mathrm{E}}(\mathbf{p}): \min_{\mathbf{w},\boldsymbol{\theta}} \quad & \Phi(\mathbf{w},\boldsymbol{\theta}) \\ \text{s.t.} \quad & \mathbf{g}(\mathbf{w},\boldsymbol{\theta}) = 0 \\ & \mathbf{h}(\mathbf{w},\boldsymbol{\theta}) \leq 0 \\ & \mathbf{p} - \boldsymbol{\theta} = 0 \end{aligned}$$

**What difference does this make?!?** Consider QP for $\mathrm{P}_{\mathrm{E}}(\mathbf{p})$:

$$\begin{aligned} \min_{\Delta\mathbf{w},\Delta\boldsymbol{\theta}} \quad & \frac{1}{2}\Delta\mathbf{w}^\top H \Delta\mathbf{w} + \nabla_{\mathbf{w}}\Phi^\top\Delta\mathbf{w} + \cancel{\nabla_{\boldsymbol{\theta}}\Phi^\top\Delta\boldsymbol{\theta}} + \Delta\boldsymbol{\theta}^\top\nabla_{\mathbf{pw}}\mathcal{L}\Delta\mathbf{w} + \cancel{\tfrac{1}{2}\Delta\boldsymbol{\theta}^\top\nabla_{\mathbf{pp}}\mathcal{L}\Delta\boldsymbol{\theta}} \\ & \mathbf{g} + \nabla_{\mathbf{w}}\mathbf{g}^\top\Delta\mathbf{w} + \nabla_{\mathbf{p}}\mathbf{g}^\top\Delta\boldsymbol{\theta} = 0 \\ & \mathbf{h} + \nabla_{\mathbf{w}}\mathbf{h}^\top\Delta\mathbf{w} + \nabla_{\mathbf{p}}\mathbf{h}^\top\Delta\boldsymbol{\theta} \leq 0 \\ & \mathbf{p} - \boldsymbol{\theta} - \Delta\boldsymbol{\theta} = 0 \end{aligned}$$

# Parametric Embedding (cheap trick to code pred.-corr.)

Generic parametric NLPs:

$$\begin{aligned} \mathrm{P}(\mathbf{p}): \min_{\mathbf{w}} \quad & \Phi(\mathbf{w}, \mathbf{p}) \\ \text{s.t.} \quad & \mathbf{g}(\mathbf{w}, \mathbf{p}) = 0 \\ & \mathbf{h}(\mathbf{w}, \mathbf{p}) \leq 0 \end{aligned}$$

Parametric embedding :

$$\begin{aligned} \mathrm{P_E}(\mathbf{p}): \min_{\mathbf{w}, \boldsymbol{\theta}} \quad & \Phi(\mathbf{w}, \boldsymbol{\theta}) \\ \text{s.t.} \quad & \mathbf{g}(\mathbf{w}, \boldsymbol{\theta}) = 0 \\ & \mathbf{h}(\mathbf{w}, \boldsymbol{\theta}) \leq 0 \\ & \mathbf{p} - \boldsymbol{\theta} = 0 \end{aligned}$$

**What difference does this make?!?** Consider QP for $\mathrm{P_E}(\mathbf{p})$:

$$\begin{aligned} \min_{\Delta\mathbf{w}, \Delta\boldsymbol{\theta}} \quad & \frac{1}{2}\Delta\mathbf{w}^\top H \Delta\mathbf{w} + \nabla_{\mathbf{w}}\Phi^\top \Delta\mathbf{w} + \Delta\boldsymbol{\theta}^\top \nabla_{\mathbf{pw}}\mathcal{L}\Delta\mathbf{w} \\ & \mathbf{g} + \nabla_{\mathbf{w}}\mathbf{g}^\top \Delta\mathbf{w} + \nabla_{\mathbf{p}}\mathbf{g}^\top \Delta\boldsymbol{\theta} = 0 \\ & \mathbf{h} + \nabla_{\mathbf{w}}\mathbf{h}^\top \Delta\mathbf{w} + \nabla_{\mathbf{p}}\mathbf{h}^\top \Delta\boldsymbol{\theta} \leq 0 \\ & \mathbf{p} - \boldsymbol{\theta} - \Delta\boldsymbol{\theta} = 0 \end{aligned}$$

# Parametric Embedding (cheap trick to code pred.-corr.)

Generic parametric NLPs:

$$\begin{aligned} \mathrm{P}(\mathbf{p}):\ \min_{\mathbf{w}} \quad & \Phi(\mathbf{w},\mathbf{p}) \\ \text{s.t.} \quad & \mathbf{g}(\mathbf{w},\mathbf{p}) = 0 \\ & \mathbf{h}(\mathbf{w},\mathbf{p}) \leq 0 \end{aligned}$$

Parametric embedding :

$$\begin{aligned} \mathrm{P}_{\mathrm{E}}(\mathbf{p}):\ \min_{\mathbf{w},\boldsymbol{\theta}} \quad & \Phi(\mathbf{w},\boldsymbol{\theta}) \\ \text{s.t.} \quad & \mathbf{g}(\mathbf{w},\boldsymbol{\theta}) = 0 \\ & \mathbf{h}(\mathbf{w},\boldsymbol{\theta}) \leq 0 \\ & \mathbf{p} - \boldsymbol{\theta} = 0 \end{aligned}$$

**What difference does this make?!?** Consider QP for $\mathrm{P}_{\mathrm{E}}(\mathbf{p})$:

Predictor-corrector for parametric NLP $\mathrm{P}(\mathbf{p})$, with $\Delta\mathbf{p} = \mathbf{p} - \boldsymbol{\theta}$:

$$\begin{aligned} \min_{\Delta\mathbf{w}} \quad & \frac{1}{2}\Delta\mathbf{w}^\top H \Delta\mathbf{w} + \nabla_{\mathbf{w}}\Phi^\top \Delta\mathbf{w} + \Delta\mathbf{p}^\top \nabla_{\mathbf{pw}}\mathcal{L}\Delta\mathbf{w} \\ & \mathbf{g} + \nabla_{\mathbf{w}}\mathbf{g}^\top \Delta\mathbf{w} + \nabla_{\mathbf{p}}\mathbf{g}^\top \Delta\mathbf{p} = 0 \\ & \mathbf{h} + \nabla_{\mathbf{w}}\mathbf{h}^\top \Delta\mathbf{w} + \nabla_{\mathbf{p}}\mathbf{h}^\top \Delta\mathbf{p} \leq 0 \end{aligned}$$

# Parametric Embedding (cheap trick to code pred.-corr.)

Generic parametric NLPs:

$$\begin{aligned} \mathrm{P}\left(\mathbf{p}\right): \min_{\mathbf{w}} \quad & \Phi\left(\mathbf{w},\mathbf{p}\right) \\ \text{s.t.} \quad & \mathbf{g}\left(\mathbf{w},\mathbf{p}\right) = 0 \\ & \mathbf{h}\left(\mathbf{w},\mathbf{p}\right) \leq 0 \end{aligned}$$

Parametric embedding :

$$\begin{aligned} \mathrm{P}_{\mathrm{E}}\left(\mathbf{p}\right): \min_{\mathbf{w},\boldsymbol{\theta}} \quad & \Phi\left(\mathbf{w},\boldsymbol{\theta}\right) \\ \text{s.t.} \quad & \mathbf{g}\left(\mathbf{w},\boldsymbol{\theta}\right) = 0 \\ & \mathbf{h}\left(\mathbf{w},\boldsymbol{\theta}\right) \leq 0 \\ & \mathbf{p} - \boldsymbol{\theta} = 0 \end{aligned}$$

**What difference does this make?!?** Consider QP for $\mathrm{P}_{\mathrm{E}}\left(\mathbf{p}\right)$:

Predictor-corrector for parametric NLP $\mathrm{P}\left(\mathbf{p}\right)$, with $\Delta\mathbf{p} = \mathbf{p} - \boldsymbol{\theta}$:

$$\begin{aligned} \min_{\Delta\mathbf{w}} \quad & \frac{1}{2}\Delta\mathbf{w}^\top H \Delta\mathbf{w} + \nabla_{\mathbf{w}}\Phi^\top \Delta\mathbf{w} + \Delta\mathbf{p}^\top \nabla_{\mathbf{p}\mathbf{w}}\mathcal{L}\Delta\mathbf{w} \\ & \mathbf{g} + \nabla_{\mathbf{w}}\mathbf{g}^\top \Delta\mathbf{w} + \nabla_{\mathbf{p}}\mathbf{g}^\top \Delta\mathbf{p} = 0 \\ & \mathbf{h} + \nabla_{\mathbf{w}}\mathbf{h}^\top \Delta\mathbf{w} + \nabla_{\mathbf{p}}\mathbf{h}^\top \Delta\mathbf{p} \leq 0 \end{aligned}$$

- $\boldsymbol{\theta}$ is "old parameter", where we linearize

## Parametric Embedding (cheap trick to code pred.-corr.)

Generic parametric NLPs:

$$\begin{aligned} \mathrm{P}\left(\mathbf{p}\right):\ \min_{\mathbf{w}} \quad & \Phi\left(\mathbf{w},\mathbf{p}\right) \\ \text{s.t.} \quad & \mathbf{g}\left(\mathbf{w},\mathbf{p}\right) = 0 \\ & \mathbf{h}\left(\mathbf{w},\mathbf{p}\right) \leq 0 \end{aligned}$$

Parametric embedding :

$$\begin{aligned} \mathrm{P}_{\mathrm{E}}\left(\mathbf{p}\right):\ \min_{\mathbf{w},\boldsymbol{\theta}} \quad & \Phi\left(\mathbf{w},\boldsymbol{\theta}\right) \\ \text{s.t.} \quad & \mathbf{g}\left(\mathbf{w},\boldsymbol{\theta}\right) = 0 \\ & \mathbf{h}\left(\mathbf{w},\boldsymbol{\theta}\right) \leq 0 \\ & \mathbf{p} - \boldsymbol{\theta} = 0 \end{aligned}$$

**What difference does this make?!?** Consider QP for $\mathrm{P}_{\mathrm{E}}\left(\mathbf{p}\right)$:

Predictor-corrector for parametric NLP $\mathrm{P}\left(\mathbf{p}\right)$, with $\Delta\mathbf{p} = \mathbf{p} - \boldsymbol{\theta}$:

$$\begin{aligned} \min_{\Delta\mathbf{w}} \quad & \frac{1}{2}\Delta\mathbf{w}^\top H \Delta\mathbf{w} + \nabla_{\mathbf{w}}\Phi^\top \Delta\mathbf{w} + \Delta\mathbf{p}^\top \nabla_{\mathbf{pw}}\mathcal{L}\Delta\mathbf{w} \\ & \mathbf{g} + \nabla_{\mathbf{w}}\mathbf{g}^\top \Delta\mathbf{w} + \nabla_{\mathbf{p}}\mathbf{g}^\top \Delta\mathbf{p} = 0 \\ & \mathbf{h} + \nabla_{\mathbf{w}}\mathbf{h}^\top \Delta\mathbf{w} + \nabla_{\mathbf{p}}\mathbf{h}^\top \Delta\mathbf{p} \leq 0 \end{aligned}$$

- $\boldsymbol{\theta}$ is "old parameter", where we linearize
- $\mathbf{p}$ is "new parameter", where we solve

# Parametric Embedding (cheap trick to code pred.-corr.)

Generic parametric NLPs:

$$
\begin{aligned}
\mathrm{P}(\mathbf{p}):\ \min_{\mathbf{w}} \quad & \Phi(\mathbf{w},\mathbf{p}) \\
\text{s.t.} \quad & \mathbf{g}(\mathbf{w},\mathbf{p}) = 0 \\
& \mathbf{h}(\mathbf{w},\mathbf{p}) \le 0
\end{aligned}
$$

Parametric embedding :

$$
\begin{aligned}
\mathrm{P}_{\mathrm{E}}(\mathbf{p}):\ \min_{\mathbf{w},\boldsymbol{\theta}} \quad & \Phi(\mathbf{w},\boldsymbol{\theta}) \\
\text{s.t.} \quad & \mathbf{g}(\mathbf{w},\boldsymbol{\theta}) = 0 \\
& \mathbf{h}(\mathbf{w},\boldsymbol{\theta}) \le 0 \\
& \mathbf{p} - \boldsymbol{\theta} = 0
\end{aligned}
$$

**What difference does this make?!?** Consider QP for $\mathrm{P}_{\mathrm{E}}(\mathbf{p})$:

Predictor-corrector for parametric NLP $\mathrm{P}(\mathbf{p})$, with $\Delta\mathbf{p} = \mathbf{p} - \boldsymbol{\theta}$:

$$
\begin{aligned}
\min_{\Delta\mathbf{w}} \quad & \frac{1}{2}\Delta\mathbf{w}^\top H \Delta\mathbf{w} + \nabla_{\mathbf{w}}\Phi^\top \Delta\mathbf{w} + \Delta\mathbf{p}^\top \nabla_{\mathbf{pw}}\mathcal{L}\Delta\mathbf{w} \\
& \mathbf{g} + \nabla_{\mathbf{w}}\mathbf{g}^\top \Delta\mathbf{w} + \nabla_{\mathbf{p}}\mathbf{g}^\top \Delta\mathbf{p} = 0 \\
& \mathbf{h} + \nabla_{\mathbf{w}}\mathbf{h}^\top \Delta\mathbf{w} + \nabla_{\mathbf{p}}\mathbf{h}^\top \Delta\mathbf{p} \le 0
\end{aligned}
$$

- $\boldsymbol{\theta}$ is "old parameter", where we linearize
- $\mathbf{p}$ is "new parameter", where we solve

**Parametric embedding**:

- Embed parameters in NLP
- Classic SQP step is a predictor-corrector
- Cheap *coding* for path-following

# Outline

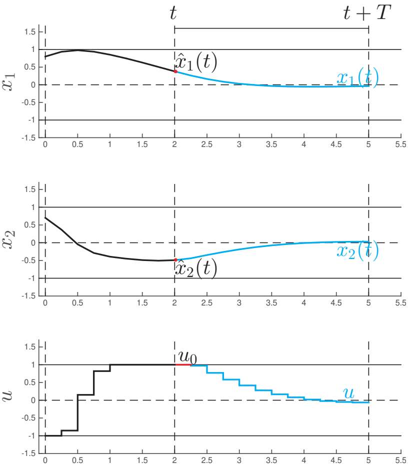
# NMPC is a parametric NLP

## NMPC problem

$$\begin{aligned} \mathrm{NMPC}\left(\hat{\mathbf{x}}(t)\right): \quad \min_{\mathbf{u},\mathbf{x}} \quad & \int_t^{t+T} \phi\left(\mathbf{x},\mathbf{u}\right) \mathrm{d}\tau \\ \text{s.t.} \quad & \dot{\mathbf{x}} = \mathbf{F}\left(\mathbf{x},\mathbf{u}\right), \\ & \mathbf{x}\left(t\right) = \hat{\mathbf{x}}(t) \\ & \mathbf{h}\left(\mathbf{x},\mathbf{u}\right) \leq 0 \end{aligned}$$

## NLP discretization

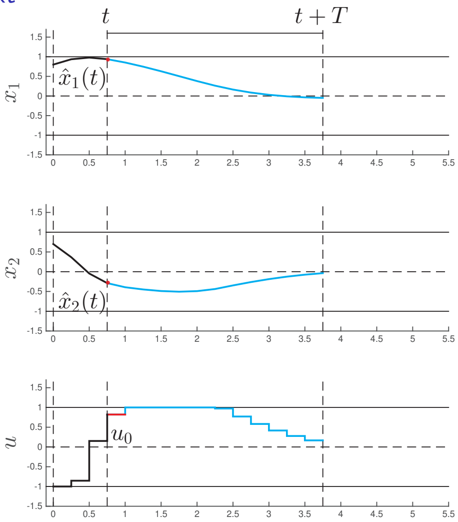
$$\begin{aligned} \mathrm{NLP}\left(\hat{\mathbf{x}}(t)\right): \quad \min_{\mathbf{w}} \quad & \Phi\left(\mathbf{w}, \hat{\mathbf{x}}(t)\right) \\ \text{s.t.} \quad & \mathbf{g}\left(\mathbf{w}, \hat{\mathbf{x}}(t)\right) = 0 \\ & \mathbf{h}\left(\mathbf{w}, \hat{\mathbf{x}}(t)\right) \leq 0 \end{aligned}$$



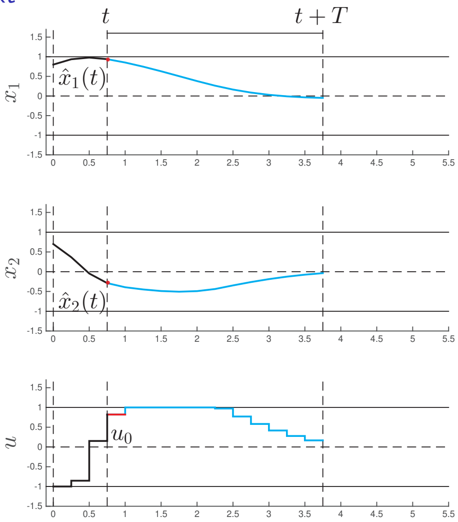

- Initial conditions play the role of parameters
- Solution for $\hat{\mathbf{x}}\left(t\right)$ is almost solution for $\hat{\mathbf{x}}\left(t+\Delta t\right)$
- Predictor-corrector approach for solving problem $\hat{\mathbf{x}}\left(t+\Delta t\right)$?

# Shifting - From one solution to the next

- If model is good, system will be close to predicted trajectory



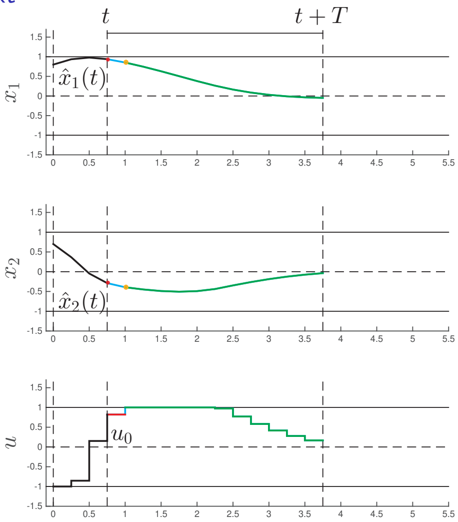
# Shifting - From one solution to the next




- If model is good, system will be close to predicted trajectory
- $\mathbf{x}(t+\Delta t) \approx \mathbf{x}_{\text{pred}}(t+\Delta t)$

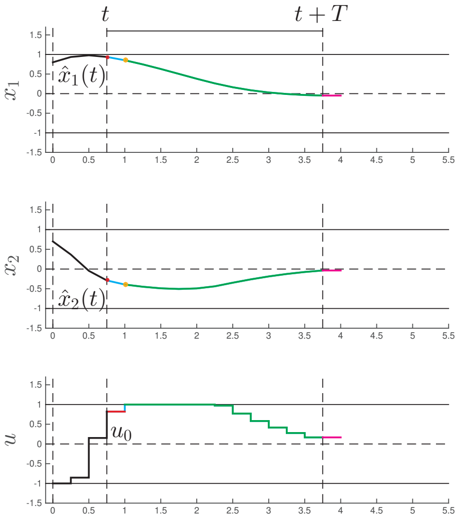
# Shifting - From one solution to the next

- If model is good, system will be close to predicted trajectory
- $\mathbf{x}(t+\Delta t) \approx \mathbf{x}_{\mathrm{pred}}(t+\Delta t)$
- Prediction $\mathbf{x}_{\mathrm{pred}}(.)$ on $[t+\Delta t, t+T]$ is close to solution



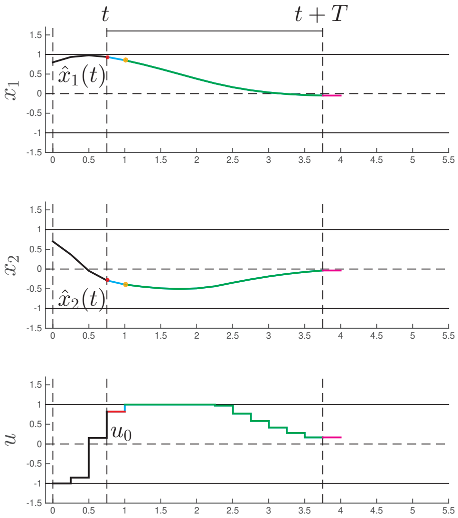
# Shifting - From one solution to the next

- If model is good, system will be close to predicted trajectory
- $\mathbf{x}(t + \Delta t) \approx \mathbf{x}_{\text{pred}}(t + \Delta t)$
- Prediction $\mathbf{x}_{\text{pred}}(.)$ on $[t + \Delta t, t + T]$ is close to solution
  - Requires "completion"

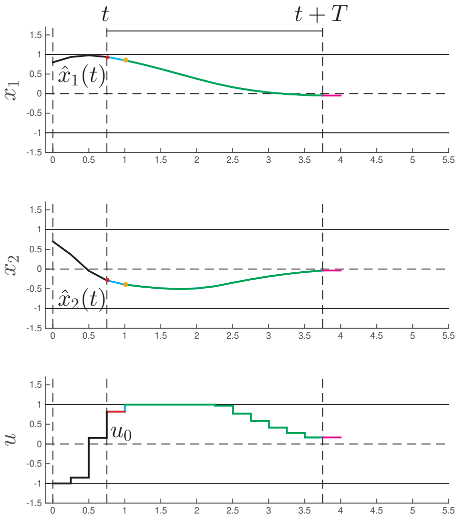


# Shifting - From one solution to the next

- If model is good, system will be close to predicted trajectory
- $\mathbf{x}(t + \Delta t) \approx \mathbf{x}_{\text{pred}}(t + \Delta t)$
- Prediction $\mathbf{x}_{\text{pred}}(.)$ on $[t + \Delta t, t + T]$ is close to solution
  - Requires "completion"
  - It is suboptimal



# Shifting - From one solution to the next



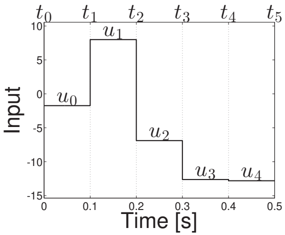

- If model is good, system will be close to predicted trajectory
- $\mathbf{x}(t+\Delta t) \approx \mathbf{x}_{\text{pred}}(t+\Delta t)$
- Prediction $\mathbf{x}_{\text{pred}}(.)$ on $[t+\Delta t, t+T]$ is close to solution
  - Requires "completion"
  - It is suboptimal

**$\mathbf{x}_{\text{pred}}(.)$ + "completion" is called shifting, generates very good guess for NMPC at time $t+\Delta t$, provided that solution at time $t$ is good.**

# NMPC & Parametric NLPs for simultaneous methods

$$\mathrm{NMPC}\left(\hat{\mathbf{x}}(t)\right):$$

$$\begin{aligned}
\min_{\mathbf{u},\mathbf{x}} \quad & \int_{t}^{t+T} \phi\left(\mathbf{x},\mathbf{u}\right)\mathrm{d}\tau \\
\text{s.t.} \quad & \dot{\mathbf{x}} = \mathbf{F}\left(\mathbf{x},\mathbf{u}\right) \\
& \mathbf{x}\left(t\right) = \hat{\mathbf{x}}(t) \\
& \mathbf{h}\left(\mathbf{x},\mathbf{u}\right) \leq 0
\end{aligned}$$

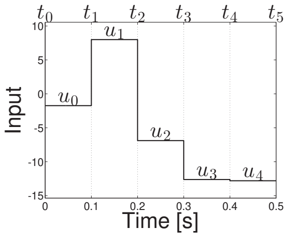


# NMPC & Parametric NLPs for simultaneous methods

$$\mathrm{NMPC}\left(\hat{\mathbf{x}}(t)\right):$$

$$\begin{aligned} \min_{\mathbf{u},\mathbf{x}} \quad & \int_t^{t+T} \phi\left(\mathbf{x},\mathbf{u}\right) \mathrm{d}\tau \\ \text{s.t.} \quad & \dot{\mathbf{x}} = \mathbf{F}\left(\mathbf{x},\mathbf{u}\right) \\ & \mathbf{x}\left(t\right) = \hat{\mathbf{x}}(t) \\ & \mathbf{h}\left(\mathbf{x},\mathbf{u}\right) \leq 0 \end{aligned}$$

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

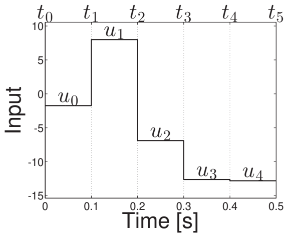


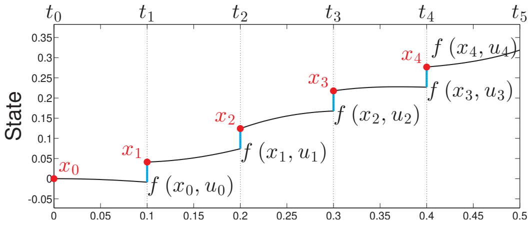
# NMPC & Parametric NLPs for simultaneous methods

$$\text{NMPC}\left(\hat{\mathbf{x}}(t)\right):$$

$$\begin{aligned} \min_{\mathbf{u},\mathbf{x}} \quad & \int_t^{t+T} \phi\left(\mathbf{x},\mathbf{u}\right) \mathrm{d}\tau \\ \text{s.t.} \quad & \dot{\mathbf{x}} = \mathbf{F}\left(\mathbf{x},\mathbf{u}\right) \\ & \mathbf{x}\left(t\right) = \hat{\mathbf{x}}(t) \\ & \mathbf{h}\left(\mathbf{x},\mathbf{u}\right) \leq 0 \end{aligned}$$

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



# NMPC & Parametric NLPs for simultaneous methods

$$\text{NMPC}\left(\hat{\mathbf{x}}(t)\right):$$

$$\min_{\mathbf{u},\mathbf{x}} \quad \int_t^{t+T} \phi\left(\mathbf{x},\mathbf{u}\right)\mathrm{d}\tau$$

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

$$\mathbf{x}\left(t\right) = \hat{\mathbf{x}}(t)$$

$$\mathbf{h}\left(\mathbf{x},\mathbf{u}\right) \leq 0$$

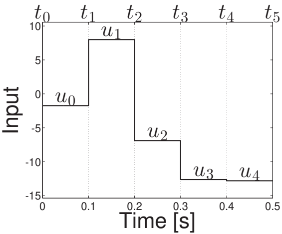
$\mathbf{f}\left(\mathbf{x}_k,\mathbf{u}_k\right)$ integrates the dynamics $\mathbf{F}$ over the time interval $\left[t_k,\ t_{k+1}\right]$

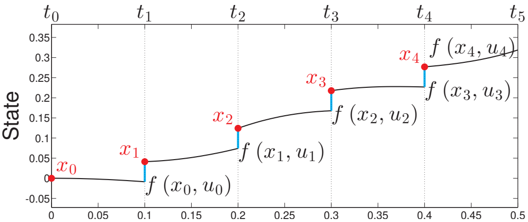
**NLP** with $\mathbf{w} = \left\{\mathbf{x}_0, \mathbf{u}_0, \ldots, \mathbf{x}_{N-1}, \mathbf{u}_{N-1}, \mathbf{x}_N\right\}$

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

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

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



# Outline

## Real-time Path Following - The real-time dilemma

Suppose $\mathbf{p}(t)$ is **continuously changing with time** $t$ and "continuously" measured...
How should we use this information?

$$\begin{aligned} \mathrm{NLP}\left(\hat{\mathbf{x}}(t)\right): \quad \min_{\mathbf{w}} \quad & \Phi\left(\mathbf{w}\right) \\ \text{s.t.} \quad & \mathbf{g}\left(\mathbf{w}, \hat{\mathbf{x}}(t)\right) = 0 \\ & \mathbf{h}\left(\mathbf{w}\right) \leq 0 \end{aligned}$$

**Real-time dilemma**:

- Solve $\mathrm{NLP}\left(\hat{\mathbf{x}}(t)\right)$ to full convergence
  $\rightarrow$ good solution but outdated
- Iterate $\mathrm{NLP}\left(\hat{\mathbf{x}}(t)\right)$ on latest $\hat{\mathbf{x}}(t)$
  $\rightarrow$ approximate solution but up-to-date

## Real-time Path Following - The real-time dilemma

Suppose $\mathbf{p}(t)$ is **continuously changing with time** $t$ and "continuously" measured...
How should we use this information?

- **NMPC**: $\mathbf{p}\left(t\right)$ is a state estimation $\hat{\mathbf{x}}(t)$

$$\begin{aligned} \mathrm{NLP}\left(\hat{\mathbf{x}}(t)\right): \min_{\mathbf{w}} \quad & \Phi\left(\mathbf{w}\right) \\ \text{s.t.} \quad & \mathbf{g}\left(\mathbf{w}, \hat{\mathbf{x}}(t)\right) = 0 \\ & \mathbf{h}\left(\mathbf{w}\right) \leq 0 \end{aligned}$$

**Real-time dilemma**:

- Solve $\mathrm{NLP}\left(\hat{\mathbf{x}}(t)\right)$ to full convergence
  → good solution but outdated
- Iterate $\mathrm{NLP}\left(\hat{\mathbf{x}}(t)\right)$ on latest $\hat{\mathbf{x}}(t)$
  → approximate solution but up-to-date

## Real-time Path Following - The real-time dilemma

Suppose $\mathbf{p}(t)$ is **continuously changing with time** $t$ and "continuously" measured...
How should we use this information?

- **NMPC**: $\mathbf{p}(t)$ is a state estimation $\hat{\mathbf{x}}(t)$
- $\mathbf{w}(\hat{\mathbf{x}}(t))$ contains the control $\mathbf{u}_0, \ldots, \mathbf{u}_{N-1}$, control $\mathbf{u}_0$ is delivered to the system

$$\begin{aligned} \mathrm{NLP}(\hat{\mathbf{x}}(t)) : \min_{\mathbf{w}} \quad & \Phi(\mathbf{w}) \\ \text{s.t.} \quad & \mathbf{g}(\mathbf{w}, \hat{\mathbf{x}}(t)) = 0 \\ & \mathbf{h}(\mathbf{w}) \leq 0 \end{aligned}$$

**Real-time dilemma**:

- Solve $\mathrm{NLP}(\hat{\mathbf{x}}(t))$ to full convergence
  $\rightarrow$ good solution but outdated
- Iterate $\mathrm{NLP}(\hat{\mathbf{x}}(t))$ on latest $\hat{\mathbf{x}}(t)$
  $\rightarrow$ approximate solution but up-to-date

## Real-time Path Following - The real-time dilemma

Suppose $\mathbf{p}(t)$ is **continuously changing with time** $t$ and "continuously" measured...
How should we use this information?

- **NMPC**: $\mathbf{p}(t)$ is a state estimation $\hat{\mathbf{x}}(t)$
- $\mathbf{w}(\hat{\mathbf{x}}(t))$ contains the control $\mathbf{u}_0, \ldots, \mathbf{u}_{N-1}$, control $\mathbf{u}_0$ is delivered to the system
- Feedback arises from $\hat{\mathbf{x}}(t) \xrightarrow{\text{NMPC}} \mathbf{u}_0$

$$\begin{aligned} \text{NLP}(\hat{\mathbf{x}}(t)): \quad \min_{\mathbf{w}} \quad & \Phi(\mathbf{w}) \\ \text{s.t.} \quad & \mathbf{g}(\mathbf{w}, \hat{\mathbf{x}}(t)) = 0 \\ & \mathbf{h}(\mathbf{w}) \leq 0 \end{aligned}$$

**Real-time dilemma**:

- Solve $\text{NLP}(\hat{\mathbf{x}}(t))$ to full convergence
  $\rightarrow$ good solution but outdated
- Iterate $\text{NLP}(\hat{\mathbf{x}}(t))$ on latest $\hat{\mathbf{x}}(t)$
  $\rightarrow$ approximate solution but up-to-date

## Real-time Path Following - The real-time dilemma

Suppose $\mathbf{p}(t)$ is **continuously changing with time** $t$ and "continuously" measured...
How should we use this information?

- **NMPC**: $\mathbf{p}(t)$ is a state estimation $\hat{\mathbf{x}}(t)$
- $\mathbf{w}(\hat{\mathbf{x}}(t))$ contains the control $\mathbf{u}_0, \dots, \mathbf{u}_{N-1}$, control $\mathbf{u}_0$ is delivered to the system
- Feedback arises from $\hat{\mathbf{x}}(t) \xrightarrow{\text{NMPC}} \mathbf{u}_0$
- $\xrightarrow{\text{NMPC}}$ is control delay!! Minimize time for updating solution

$$\begin{aligned} \mathrm{NLP}(\hat{\mathbf{x}}(t)): \quad \min_{\mathbf{w}} \quad & \Phi(\mathbf{w}) \\ \text{s.t.} \quad & \mathbf{g}(\mathbf{w}, \hat{\mathbf{x}}(t)) = 0 \\ & \mathbf{h}(\mathbf{w}) \leq 0 \end{aligned}$$

**Real-time dilemma**:

- Solve $\mathrm{NLP}(\hat{\mathbf{x}}(t))$ to full convergence
  $\rightarrow$ good solution but outdated
- Iterate $\mathrm{NLP}(\hat{\mathbf{x}}(t))$ on latest $\hat{\mathbf{x}}(t)$
  $\rightarrow$ approximate solution but up-to-date

# Real-time Path Following - The real-time dilemma

Suppose $\mathbf{p}(t)$ is **continuously changing with time** $t$ and "continuously" measured...
How should we use this information?

- **NMPC**: $\mathbf{p}\left(t\right)$ is a state estimation $\hat{\mathbf{x}}(t)$
- $\mathbf{w}\left(\hat{\mathbf{x}}(t)\right)$ contains the control $\mathbf{u}_0, \ldots, \mathbf{u}_{N-1}$, control $\mathbf{u}_0$ is delivered to the system
- Feedback arises from $\hat{\mathbf{x}}(t) \xrightarrow{\text{NMPC}} \mathbf{u}_0$
- $\xrightarrow{\text{NMPC}}$ is control delay!! Minimize time for updating solution

**Algorithm:** Path-following SQP for NMPC

**Input:** Solution estimate $\hat{\mathbf{z}}(\hat{\mathbf{x}})$, $\hat{\mathbf{x}}_+$

Predictor-corrector with $\Delta\hat{\mathbf{x}} = \hat{\mathbf{x}}_+ - \hat{\mathbf{x}}$

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

Update $\hat{\mathbf{z}}\left(\hat{\mathbf{x}}_+\right) = \hat{\mathbf{z}}\left(\hat{\mathbf{x}}\right) + \Delta\mathbf{z}$

**return** $\hat{\mathbf{z}}\left(\hat{\mathbf{x}}_+\right)$

## Real-time Path Following - The real-time dilemma

Suppose $\mathbf{p}(t)$ is **continuously changing with time** $t$ and "continuously" measured...
How should we use this information?

- **NMPC**: $\mathbf{p}\,(t)$ is a state estimation $\hat{\mathbf{x}}(t)$
- $\mathbf{w}\,(\hat{\mathbf{x}}(t))$ contains the control $\mathbf{u}_0, \ldots, \mathbf{u}_{N-1}$, control $\mathbf{u}_0$ is delivered to the system
- Feedback arises from $\hat{\mathbf{x}}(t) \xrightarrow{\text{NMPC}} \mathbf{u}_0$
- $\xrightarrow{\text{NMPC}}$ is control delay!! Minimize time for updating solution

**Algorithm:** Path-following SQP for NMPC

**Input:** Solution estimate $\hat{\mathbf{z}}(\hat{\mathbf{x}})$, $\hat{\mathbf{x}}_+$

Predictor-corrector with $\Delta\hat{\mathbf{x}} = \hat{\mathbf{x}}_+ - \hat{\mathbf{x}}$

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

Update $\hat{\mathbf{z}}\,(\hat{\mathbf{x}}_+) = \hat{\mathbf{z}}\,(\hat{\mathbf{x}}) + \Delta\mathbf{z}$

**return** $\hat{\mathbf{z}}\,(\hat{\mathbf{x}}_+)$

**Note:**

- $\Delta\hat{\mathbf{x}}$ enters linearly and only in $\mathbf{g}$

## Real-time Path Following - The real-time dilemma

Suppose $\mathbf{p}(t)$ is **continuously changing with time** $t$ and "continuously" measured...
How should we use this information?

- **NMPC**: $\mathbf{p}\left(t\right)$ is a state estimation $\hat{\mathbf{x}}(t)$
- $\mathbf{w}\left(\hat{\mathbf{x}}(t)\right)$ contains the control $\mathbf{u}_0, \ldots, \mathbf{u}_{N-1}$, control $\mathbf{u}_0$ is delivered to the system
- Feedback arises from $\hat{\mathbf{x}}(t) \xrightarrow{\text{NMPC}} \mathbf{u}_0$
- $\xrightarrow{\text{NMPC}}$ is control delay!! Minimize time for updating solution

**Algorithm:** Path-following SQP for NMPC

**Input:** Solution estimate $\hat{\mathbf{z}}(\hat{\mathbf{x}})$, $\hat{\mathbf{x}}_+$

Predictor-corrector with $\Delta\hat{\mathbf{x}} = \hat{\mathbf{x}}_+ - \hat{\mathbf{x}}$

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

Update $\hat{\mathbf{z}}\left(\hat{\mathbf{x}}_+\right) = \hat{\mathbf{z}}\left(\hat{\mathbf{x}}\right) + \Delta\mathbf{z}$

**return** $\hat{\mathbf{z}}\left(\hat{\mathbf{x}}_+\right)$

**Note:**

- $\Delta\hat{\mathbf{x}}$ enters linearly and only in $\mathbf{g}$
- $\hat{\mathbf{x}}$ is previous state estimate, $\hat{\mathbf{x}}_+$ most recent one

## Real-time Path Following - The real-time dilemma

Suppose $\mathbf{p}(t)$ is **continuously changing with time** $t$ and "continuously" measured...
How should we use this information?

- **NMPC**: $\mathbf{p}(t)$ is a state estimation $\hat{\mathbf{x}}(t)$
- $\mathbf{w}(\hat{\mathbf{x}}(t))$ contains the control $\mathbf{u}_0, \ldots, \mathbf{u}_{N-1}$, control $\mathbf{u}_0$ is delivered to the system
- Feedback arises from $\hat{\mathbf{x}}(t) \xrightarrow{\text{NMPC}} \mathbf{u}_0$
- $\xrightarrow{\text{NMPC}}$ is control delay!! Minimize time for updating solution

**Algorithm:** Path-following SQP for NMPC

**Input:** Solution estimate $\hat{\mathbf{z}}(\hat{\mathbf{x}})$, $\hat{\mathbf{x}}_+$

Predictor-corrector with $\Delta\hat{\mathbf{x}} = \hat{\mathbf{x}}_+ - \hat{\mathbf{x}}$

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

Update $\hat{\mathbf{z}}(\hat{\mathbf{x}}_+) = \hat{\mathbf{z}}(\hat{\mathbf{x}}) + \Delta\mathbf{z}$

**return** $\hat{\mathbf{z}}(\hat{\mathbf{x}}_+)$

**Note:**

- $\Delta\hat{\mathbf{x}}$ enters linearly and only in $\mathbf{g}$
- $\hat{\mathbf{x}}$ is previous state estimate, $\hat{\mathbf{x}}_+$ most recent one
- $\hat{\mathbf{z}}(\hat{\mathbf{x}})$ comes from shifting previous solution

# Real-time Path Following - The real-time dilemma

Suppose $\mathbf{p}(t)$ is **continuously changing with time** $t$ and "continuously" measured... How should we use this information?

- **NMPC**: $\mathbf{p}(t)$ is a state estimation $\hat{\mathbf{x}}(t)$
- $\mathbf{w}(\hat{\mathbf{x}}(t))$ contains the control $\mathbf{u}_0, \dots, \mathbf{u}_{N-1}$, control $\mathbf{u}_0$ is delivered to the system
- Feedback arises from $\hat{\mathbf{x}}(t) \xrightarrow{\text{NMPC}} \mathbf{u}_0$
- $\xrightarrow{\text{NMPC}}$ is control delay!! Minimize time for updating solution

**Algorithm:** Path-following SQP for NMPC

**Input:** Solution estimate $\hat{\mathbf{z}}(\hat{\mathbf{x}})$, $\hat{\mathbf{x}}_+$

Predictor-corrector with $\Delta\hat{\mathbf{x}} = \hat{\mathbf{x}}_+ - \hat{\mathbf{x}}$

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

Update $\hat{\mathbf{z}}(\hat{\mathbf{x}}_+) = \hat{\mathbf{z}}(\hat{\mathbf{x}}) + \Delta\mathbf{z}$

**return** $\hat{\mathbf{z}}(\hat{\mathbf{x}}_+)$

**Note:**

- $\Delta\hat{\mathbf{x}}$ enters linearly and only in $\mathbf{g}$
- $\hat{\mathbf{x}}$ is previous state estimate, $\hat{\mathbf{x}}_+$ most recent one
- $\hat{\mathbf{z}}(\hat{\mathbf{x}})$ comes from shifting previous solution
- $\Delta\hat{\mathbf{x}} \equiv$ prediction error

# Real-time Path Following - The real-time dilemma

Suppose $\mathbf{p}(t)$ is **continuously changing with time** $t$ and "continuously" measured...
How should we use this information?

- **NMPC**: $\mathbf{p}(t)$ is a state estimation $\hat{\mathbf{x}}(t)$
- $\mathbf{w}(\hat{\mathbf{x}}(t))$ contains the control $\mathbf{u}_0, \ldots, \mathbf{u}_{N-1}$, control $\mathbf{u}_0$ is delivered to the system
- Feedback arises from $\hat{\mathbf{x}}(t) \xrightarrow{\text{NMPC}} \mathbf{u}_0$
- $\xrightarrow{\text{NMPC}}$ is control delay!! Minimize time for updating solution

**Algorithm:** Path-following SQP for NMPC

**Input:** Solution estimate $\hat{\mathbf{z}}(\hat{\mathbf{x}})$, $\hat{\mathbf{x}}_+$

Predictor-corrector with $\Delta\hat{\mathbf{x}} = \hat{\mathbf{x}}_+ - \hat{\mathbf{x}}$

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

Update $\hat{\mathbf{z}}(\hat{\mathbf{x}}_+) = \hat{\mathbf{z}}(\hat{\mathbf{x}}) + \Delta\mathbf{z}$

**return** $\hat{\mathbf{z}}(\hat{\mathbf{x}}_+)$

**Note:**

- $\Delta\hat{\mathbf{x}}$ enters linearly and only in $\mathbf{g}$
- $\hat{\mathbf{x}}$ is previous state estimate, $\hat{\mathbf{x}}_+$ most recent one
- $\hat{\mathbf{z}}(\hat{\mathbf{x}})$ comes from shifting previous solution
- $\Delta\hat{\mathbf{x}} \equiv$ prediction error
- All can be done via initial condition embedding

# Real-time Path Following - The real-time dilemma

Suppose $\mathbf{p}(t)$ is **continuously changing with time** $t$ and "continuously" measured...
How should we use this information?

- **NMPC**: $\mathbf{p}\left(t\right)$ is a state estimation $\hat{\mathbf{x}}(t)$
- $\mathbf{w}\left(\hat{\mathbf{x}}(t)\right)$ contains the control $\mathbf{u}_0, \ldots, \mathbf{u}_{N-1}$, control $\mathbf{u}_0$ is delivered to the system
- Feedback arises from $\hat{\mathbf{x}}(t) \xrightarrow{\text{NMPC}} \mathbf{u}_0$
- $\xrightarrow{\text{NMPC}}$ is control delay!! Minimize time for updating solution

**Algorithm:** Path-following SQP for NMPC

**Input:** Solution estimate $\hat{\mathbf{z}}(\hat{\mathbf{x}})$, $\hat{\mathbf{x}}_+$

Predictor-corrector with $\Delta\hat{\mathbf{x}} = \hat{\mathbf{x}}_+ - \hat{\mathbf{x}}$

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

Update $\hat{\mathbf{z}}\left(\hat{\mathbf{x}}_+\right) = \hat{\mathbf{z}}\left(\hat{\mathbf{x}}\right) + \Delta\mathbf{z}$

**return** $\hat{\mathbf{z}}\left(\hat{\mathbf{x}}_+\right)$

**Workload:**

- Form $H, \nabla_{\mathbf{w}}\Phi, \mathbf{g}, \nabla_{\mathbf{w}}\mathbf{g}, \mathbf{h}, \nabla_{\mathbf{w}}\mathbf{h}$ at $\hat{\mathbf{z}}(\hat{\mathbf{x}})$
- Solve QP

# Real-time Path Following - The real-time dilemma

Suppose $\mathbf{p}(t)$ is **continuously changing with time** $t$ and "continuously" measured...
How should we use this information?

- **NMPC**: $\mathbf{p}\left(t\right)$ is a state estimation $\hat{\mathbf{x}}(t)$
- $\mathbf{w}\left(\hat{\mathbf{x}}(t)\right)$ contains the control $\mathbf{u}_0, \ldots, \mathbf{u}_{N-1}$, control $\mathbf{u}_0$ is delivered to the system
- Feedback arises from $\hat{\mathbf{x}}(t) \xrightarrow{\text{NMPC}} \mathbf{u}_0$
- $\xrightarrow{\text{NMPC}}$ is control delay!! Minimize time for updating solution

**Algorithm:** Path-following SQP for NMPC

**Input:** Solution estimate $\hat{\mathbf{z}}(\hat{\mathbf{x}})$, $\hat{\mathbf{x}}_+$

Predictor-corrector with $\Delta\hat{\mathbf{x}} = \hat{\mathbf{x}}_+ - \hat{\mathbf{x}}$

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

Update $\hat{\mathbf{z}}\left(\hat{\mathbf{x}}_+\right) = \hat{\mathbf{z}}\left(\hat{\mathbf{x}}\right) + \Delta\mathbf{z}$

**return** $\hat{\mathbf{z}}\left(\hat{\mathbf{x}}_+\right)$

**Workload:**

- Form $H, \nabla_{\mathbf{w}}\Phi, \mathbf{g}, \nabla_{\mathbf{w}}\mathbf{g}, \mathbf{h}, \nabla_{\mathbf{w}}\mathbf{h}$ at $\hat{\mathbf{z}}(\hat{\mathbf{x}})$
- Solve QP

**Note:**

- Approximation for $H$ is often used

# Real-time Path Following - The real-time dilemma

Suppose $\mathbf{p}(t)$ is **continuously changing with time** $t$ and "continuously" measured...
How should we use this information?

- **NMPC**: $\mathbf{p}\left(t\right)$ is a state estimation $\hat{\mathbf{x}}(t)$
- $\mathbf{w}\left(\hat{\mathbf{x}}(t)\right)$ contains the control $\mathbf{u}_0, \ldots, \mathbf{u}_{N-1}$, control $\mathbf{u}_0$ is delivered to the system
- Feedback arises from $\hat{\mathbf{x}}(t) \xrightarrow{\text{NMPC}} \mathbf{u}_0$
- $\xrightarrow{\text{NMPC}}$ is control delay!! Minimize time for updating solution

**Algorithm:** Path-following SQP for NMPC

**Input:** Solution estimate $\hat{\mathbf{z}}(\hat{\mathbf{x}})$, $\hat{\mathbf{x}}_+$

Predictor-corrector with $\Delta\hat{\mathbf{x}} = \hat{\mathbf{x}}_+ - \hat{\mathbf{x}}$

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

Update $\hat{\mathbf{z}}\left(\hat{\mathbf{x}}_+\right) = \hat{\mathbf{z}}\left(\hat{\mathbf{x}}\right) + \Delta\mathbf{z}$

**return** $\hat{\mathbf{z}}\left(\hat{\mathbf{x}}_+\right)$

**Workload:**

- Form $H, \nabla_{\mathbf{w}}\Phi, \mathbf{g}, \nabla_{\mathbf{w}}\mathbf{g}, \mathbf{h}, \nabla_{\mathbf{w}}\mathbf{h}$ at $\hat{\mathbf{z}}(\hat{\mathbf{x}})$
- Solve QP

**Note:**

- Approximation for $H$ is often used
- $\nabla_{\mathbf{w}}\Phi, \nabla_{\mathbf{w}}\mathbf{h}$ cheap, $\nabla_{\hat{\mathbf{x}}(t)}\mathbf{g}$ is constant, $\nabla_{\mathbf{w}}\mathbf{g}$ is expensive...

# Real-time Path Following - The real-time dilemma

Suppose $\mathbf{p}(t)$ is **continuously changing with time** $t$ and "continuously" measured...
How should we use this information?

- **NMPC**: $\mathbf{p}(t)$ is a state estimation $\hat{\mathbf{x}}(t)$
- $\mathbf{w}(\hat{\mathbf{x}}(t))$ contains the control $\mathbf{u}_0, \dots, \mathbf{u}_{N-1}$, control $\mathbf{u}_0$ is delivered to the system
- Feedback arises from $\hat{\mathbf{x}}(t) \xrightarrow{\text{NMPC}} \mathbf{u}_0$
- $\xrightarrow{\text{NMPC}}$ is control delay!! Minimize time for updating solution

**Algorithm:** Path-following SQP for NMPC

**Input:** Solution estimate $\hat{\mathbf{z}}(\hat{\mathbf{x}})$, $\hat{\mathbf{x}}_+$

Predictor-corrector with $\Delta\hat{\mathbf{x}} = \hat{\mathbf{x}}_+ - \hat{\mathbf{x}}$

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

Update $\hat{\mathbf{z}}(\hat{\mathbf{x}}_+) = \hat{\mathbf{z}}(\hat{\mathbf{x}}) + \Delta\mathbf{z}$

**return** $\hat{\mathbf{z}}(\hat{\mathbf{x}}_+)$

**Workload:**

- Form $H, \nabla_{\mathbf{w}}\Phi, \mathbf{g}, \nabla_{\mathbf{w}}\mathbf{g}, \mathbf{h}, \nabla_{\mathbf{w}}\mathbf{h}$ at $\hat{\mathbf{z}}(\hat{\mathbf{x}})$
- Solve QP

**Note:**

- Approximation for $H$ is often used
- $\nabla_{\mathbf{w}}\Phi, \nabla_{\mathbf{w}}\mathbf{h}$ cheap, $\nabla_{\hat{\mathbf{x}}(t)}\mathbf{g}$ is constant, $\nabla_{\mathbf{w}}\mathbf{g}$ is expensive...

# Real-time Path Following - The real-time dilemma

Suppose $\mathbf{p}(t)$ is **continuously changing with time** $t$ and "continuously" measured...
How should we use this information?

- **NMPC**: $\mathbf{p}\left(t\right)$ is a state estimation $\hat{\mathbf{x}}(t)$
- $\mathbf{w}\left(\hat{\mathbf{x}}(t)\right)$ contains the control $\mathbf{u}_0, \ldots, \mathbf{u}_{N-1}$, control $\mathbf{u}_0$ is delivered to the system
- Feedback arises from $\hat{\mathbf{x}}(t) \xrightarrow{\text{NMPC}} \mathbf{u}_0$
- $\xrightarrow{\text{NMPC}}$ is control delay!! Minimize time for updating solution

---

**Algorithm:** Path-following SQP for NMPC

---

**Input:** Solution estimate $\hat{\mathbf{z}}(\hat{\mathbf{x}})$, $\hat{\mathbf{x}}_+$

Predictor-corrector with $\Delta\hat{\mathbf{x}} = \hat{\mathbf{x}}_+ - \hat{\mathbf{x}}$

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

Update $\hat{\mathbf{z}}\left(\hat{\mathbf{x}}_+\right) = \hat{\mathbf{z}}\left(\hat{\mathbf{x}}\right) + \Delta\mathbf{z}$

**return** $\hat{\mathbf{z}}\left(\hat{\mathbf{x}}_+\right)$

---

**Workload:**

- Form $H, \nabla_{\mathbf{w}}\Phi, \mathbf{g}, \nabla_{\mathbf{w}}\mathbf{g}, \mathbf{h}, \nabla_{\mathbf{w}}\mathbf{h}$ at $\hat{\mathbf{z}}(\hat{\mathbf{x}})$
- Solve QP

**Note:**

- Approximation for $H$ is often used
- $\nabla_{\mathbf{w}}\Phi, \nabla_{\mathbf{w}}\mathbf{h}$ cheap, $\nabla_{\hat{\mathbf{x}}(t)}\mathbf{g}$ is constant, $\nabla_{\mathbf{w}}\mathbf{g}$ is expensive...
- **All matrices are independent of $\hat{\mathbf{x}}_+$!!**

# Real-Time Iteration (RTI) for NMPC

**Algorithm:** Path-following for NMPC

**Input:** Solution estimate $\hat{\mathbf{z}}(\hat{\mathbf{x}})$, $\hat{\mathbf{x}}_+$

Predictor-corrector with $\Delta\hat{\mathbf{x}} = \hat{\mathbf{x}}_+ - \hat{\mathbf{x}}$

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

Update $\hat{\mathbf{z}}(\hat{\mathbf{x}}_+) = \hat{\mathbf{z}}(\hat{\mathbf{x}}) + \Delta\mathbf{z}$

**return** $\hat{\mathbf{z}}(\hat{\mathbf{x}}_+)$

## Real-Time Iteration (RTI) for NMPC

**Algorithm:** Path-following for NMPC

**Input:** Solution estimate $\hat{\mathbf{z}}(\hat{\mathbf{x}})$, $\hat{\mathbf{x}}_+$

Predictor-corrector with $\Delta\hat{\mathbf{x}} = \hat{\mathbf{x}}_+ - \hat{\mathbf{x}}$

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

Update $\hat{\mathbf{z}}\left(\hat{\mathbf{x}}_+\right) = \hat{\mathbf{z}}\left(\hat{\mathbf{x}}\right) + \Delta\mathbf{z}$

**return** $\hat{\mathbf{z}}\left(\hat{\mathbf{x}}_+\right)$

**Time-split of the algorithm...**

# Real-Time Iteration (RTI) for NMPC

**Algorithm:** Path-following for NMPC

**Input:** Solution estimate $\hat{\mathbf{z}}(\hat{\mathbf{x}})$, $\hat{\mathbf{x}}_+$

Predictor-corrector with $\Delta\hat{\mathbf{x}} = \hat{\mathbf{x}}_+ - \hat{\mathbf{x}}$

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

Update $\hat{\mathbf{z}}(\hat{\mathbf{x}}_+) = \hat{\mathbf{z}}(\hat{\mathbf{x}}) + \Delta\mathbf{z}$

**return** $\hat{\mathbf{z}}(\hat{\mathbf{x}}_+)$

**Time-split of the algorithm...**

Perform *between* $\hat{\mathbf{x}}$ and $\hat{\mathbf{x}}_+$:

**Algorithm:** Preparation phase

**Input:** $\hat{\mathbf{z}}(\hat{\mathbf{x}})$ and $\hat{\mathbf{x}}$

Compute $H, \mathbf{h}, \nabla_{\mathbf{w}}\mathbf{h}, \nabla_{\mathbf{w}}\mathbf{g}, \nabla_{\mathbf{w}}\Phi$ and

$$\tilde{\mathbf{g}} = \begin{bmatrix} \mathbf{f}(\mathbf{x}_0, \mathbf{u}_0) - \mathbf{x}_1 \\ \vdots \\ \mathbf{f}(\mathbf{x}_{N-1}, \mathbf{u}_{N-1}) - \mathbf{x}_N \end{bmatrix}$$

**return** $H, \mathbf{h}, \nabla\mathbf{h}, \nabla\mathbf{g}, \nabla\Phi$ and $\tilde{\mathbf{g}}$

# Real-Time Iteration (RTI) for NMPC

**Algorithm:** Path-following for NMPC

**Input:** Solution estimate $\hat{\mathbf{z}}(\hat{\mathbf{x}})$, $\hat{\mathbf{x}}_+$

Predictor-corrector with $\Delta\hat{\mathbf{x}} = \hat{\mathbf{x}}_+ - \hat{\mathbf{x}}$

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

Update $\hat{\mathbf{z}}(\hat{\mathbf{x}}_+) = \hat{\mathbf{z}}(\hat{\mathbf{x}}) + \Delta\mathbf{z}$

**return** $\hat{\mathbf{z}}(\hat{\mathbf{x}}_+)$

**Time-split of the algorithm...**

Perform *between* $\hat{\mathbf{x}}$ and $\hat{\mathbf{x}}_+$:

**Algorithm:** Preparation phase

**Input:** $\hat{\mathbf{z}}(\hat{\mathbf{x}})$ and $\hat{\mathbf{x}}$

Compute $H, \mathbf{h}, \nabla_{\mathbf{w}}\mathbf{h}, \nabla_{\mathbf{w}}\mathbf{g}, \nabla_{\mathbf{w}}\Phi$ and

$$\tilde{\mathbf{g}} = \begin{bmatrix} \mathbf{f}(\mathbf{x}_0, \mathbf{u}_0) - \mathbf{x}_1 \\ \vdots \\ \mathbf{f}(\mathbf{x}_{N-1}, \mathbf{u}_{N-1}) - \mathbf{x}_N \end{bmatrix}$$

**return** $H, \mathbf{h}, \nabla\mathbf{h}, \nabla\mathbf{g}, \nabla\Phi$ and $\tilde{\mathbf{g}}$

Perform when receiving $\hat{\mathbf{x}}_+$

**Algorithm:** Feedback phase

**Input:** $\hat{\mathbf{x}}_+$, $H, \mathbf{h}, \nabla\mathbf{h}, \nabla\mathbf{g}, \nabla\Phi$ and $\tilde{\mathbf{g}}$

Form
$$\mathbf{g}(\mathbf{w}, \hat{\mathbf{x}}_i) = \begin{bmatrix} \mathbf{x}_0 - \hat{\mathbf{x}}_+ \\ \tilde{\mathbf{g}} \end{bmatrix}$$

Solve QP

Update $\hat{\mathbf{z}}(\hat{\mathbf{x}}_+) = \hat{\mathbf{z}}(\hat{\mathbf{x}}) + \Delta\mathbf{z}$

# Real-Time Iteration (RTI) for NMPC

**Algorithm:** Path-following for NMPC

**Input:** Solution estimate $\hat{\mathbf{z}}(\hat{\mathbf{x}})$, $\hat{\mathbf{x}}_+$

Predictor-corrector with $\Delta\hat{\mathbf{x}} = \hat{\mathbf{x}}_+ - \hat{\mathbf{x}}$

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

Update $\hat{\mathbf{z}}(\hat{\mathbf{x}}_+) = \hat{\mathbf{z}}(\hat{\mathbf{x}}) + \Delta\mathbf{z}$

**return** $\hat{\mathbf{z}}(\hat{\mathbf{x}}_+)$

**Time-split of the algorithm...**

- RTI reduces control delay by doing linearization before receiving the state estimation (Preparation phase)
- Feedback "reduces" to solving a QP

Perform *between* $\hat{\mathbf{x}}$ and $\hat{\mathbf{x}}_+$:

**Algorithm:** Preparation phase

**Input:** $\hat{\mathbf{z}}(\hat{\mathbf{x}})$ and $\hat{\mathbf{x}}$

Compute $H, \mathbf{h}, \nabla_{\mathbf{w}}\mathbf{h}, \nabla_{\mathbf{w}}\mathbf{g}, \nabla_{\mathbf{w}}\Phi$ and

$$
\tilde{\mathbf{g}} = \begin{bmatrix} \mathbf{f}(\mathbf{x}_0, \mathbf{u}_0) - \mathbf{x}_1 \\ \vdots \\ \mathbf{f}(\mathbf{x}_{N-1}, \mathbf{u}_{N-1}) - \mathbf{x}_N \end{bmatrix}
$$

**return** $H, \mathbf{h}, \nabla\mathbf{h}, \nabla\mathbf{g}, \nabla\Phi$ and $\tilde{\mathbf{g}}$

Perform when receiving $\hat{\mathbf{x}}_+$

**Algorithm:** Feedback phase

**Input:** $\hat{\mathbf{x}}_+$, $H, \mathbf{h}, \nabla\mathbf{h}, \nabla\mathbf{g}, \nabla\Phi$ and $\tilde{\mathbf{g}}$

Form

$$
\mathbf{g}(\mathbf{w}, \hat{\mathbf{x}}_i) = \begin{bmatrix} \mathbf{x}_0 - \hat{\mathbf{x}}_+ \\ \tilde{\mathbf{g}} \end{bmatrix}
$$

Solve QP

Update $\hat{\mathbf{z}}(\hat{\mathbf{x}}_+) = \hat{\mathbf{z}}(\hat{\mathbf{x}}) + \Delta\mathbf{z}$

# Outline

1. Preliminaries
2. Parametric Embedding
3. Parametric NLPs & NMPC
4. Real-time dilemma and the Real-Time Iteration
5. From Linear MPC to NMPC

# Deploying Linear MPC

- Nonlinear system from shooting:

$$\mathbf{x}_{k+1} = \mathbf{f}\left(\mathbf{x}_k, \mathbf{u}_k\right)$$

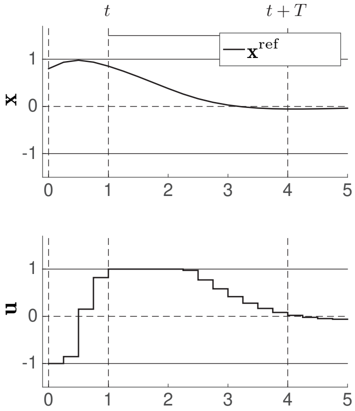
# Deploying Linear MPC

- Nonlinear system from shooting:

$$\mathbf{x}_{k+1} = \mathbf{f}\left(\mathbf{x}_k, \mathbf{u}_k\right)$$

- Reference trajectory:

$$\mathbf{x}_k^{\mathrm{ref}},\ \mathbf{u}_k^{\mathrm{ref}}$$



# Deploying Linear MPC

- Nonlinear system from shooting:

$$\mathbf{x}_{k+1} = \mathbf{f}\left(\mathbf{x}_k, \mathbf{u}_k\right)$$

- Reference trajectory:

$$\mathbf{x}_k^{\mathrm{ref}},\ \mathbf{u}_k^{\mathrm{ref}}$$

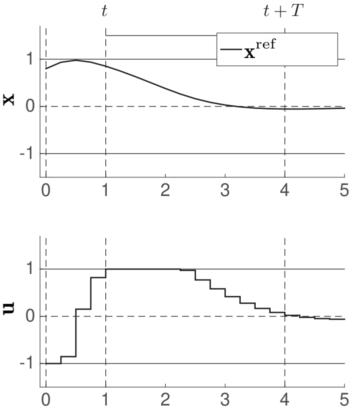
- Affine Time-Varying model:

$$\Delta\mathbf{x}_{k+1} = A_k \Delta\mathbf{x}_k + B_k \Delta\mathbf{u}_k + \mathbf{r}_k$$

where

$$\Delta\mathbf{x}_k = \mathbf{x}_k - \mathbf{x}_k^{\mathrm{ref}}, \quad \Delta\mathbf{u}_k = \mathbf{u}_k - \mathbf{u}_k^{\mathrm{ref}}$$

$$A_k = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\mathbf{x}_k^{\mathrm{ref}}, \mathbf{u}_k^{\mathrm{ref}}}, \quad B_k = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{u}}\right|_{\mathbf{x}_k^{\mathrm{ref}}, \mathbf{u}_k^{\mathrm{ref}}}$$



# Deploying Linear MPC

- Nonlinear system from shooting:

$$\mathbf{x}_{k+1} = \mathbf{f}\left(\mathbf{x}_k, \mathbf{u}_k\right)$$

- Reference trajectory:

$$\mathbf{x}_k^{\mathrm{ref}},\ \mathbf{u}_k^{\mathrm{ref}}$$

- Affine Time-Varying model:

$$\Delta\mathbf{x}_{k+1} = A_k \Delta\mathbf{x}_k + B_k \Delta\mathbf{u}_k + \mathbf{r}_k$$

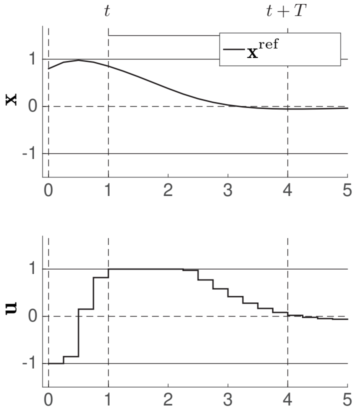
where

$$\Delta\mathbf{x}_k = \mathbf{x}_k - \mathbf{x}_k^{\mathrm{ref}}, \quad \Delta\mathbf{u}_k = \mathbf{u}_k - \mathbf{u}_k^{\mathrm{ref}}$$

$$A_k = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\mathbf{x}_k^{\mathrm{ref}}, \mathbf{u}_k^{\mathrm{ref}}}, \quad B_k = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{u}}\right|_{\mathbf{x}_k^{\mathrm{ref}}, \mathbf{u}_k^{\mathrm{ref}}}$$

$$\mathbf{r}_k = \mathbf{f}\left(\mathbf{x}_k^{\mathrm{ref}}, \mathbf{u}_k^{\mathrm{ref}}\right) - \mathbf{x}_{k+1}^{\mathrm{ref}}$$

note $\mathbf{r}_k = 0$ if reference trajectory $\mathbf{x}_k^{\mathrm{ref}}, \mathbf{u}_k^{\mathrm{ref}}$ is feasible



# Deploying Linear MPC

- Nonlinear system from shooting:

$$\mathbf{x}_{k+1} = \mathbf{f}\left(\mathbf{x}_k, \mathbf{u}_k\right)$$

- Reference trajectory:

$$\mathbf{x}_k^{\mathrm{ref}}, \ \mathbf{u}_k^{\mathrm{ref}}$$

- Affine Time-Varying model:

$$\Delta\mathbf{x}_{k+1} = A_k \Delta\mathbf{x}_k + B_k \Delta\mathbf{u}_k + \mathbf{r}_k$$

where

$$\Delta\mathbf{x}_k = \mathbf{x}_k - \mathbf{x}_k^{\mathrm{ref}}, \quad \Delta\mathbf{u}_k = \mathbf{u}_k - \mathbf{u}_k^{\mathrm{ref}}$$

$$A_k = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\mathbf{x}_k^{\mathrm{ref}}, \mathbf{u}_k^{\mathrm{ref}}}, \quad B_k = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{u}}\right|_{\mathbf{x}_k^{\mathrm{ref}}, \mathbf{u}_k^{\mathrm{ref}}}$$

$$\mathbf{r}_k = \mathbf{f}\left(\mathbf{x}_k^{\mathrm{ref}}, \mathbf{u}_k^{\mathrm{ref}}\right) - \mathbf{x}_{k+1}^{\mathrm{ref}}$$

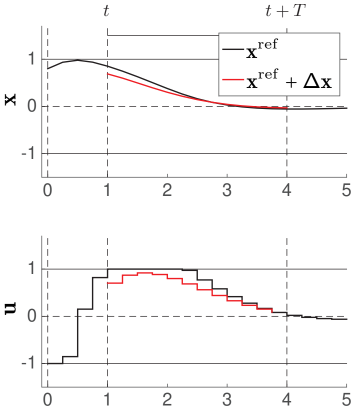
note $\mathbf{r}_k = 0$ if reference trajectory $\mathbf{x}_k^{\mathrm{ref}}, \mathbf{u}_k^{\mathrm{ref}}$ is feasible

MPC $\left(\hat{\mathbf{x}}_i, \mathbf{x}^{\mathrm{ref}}, \mathbf{u}^{\mathrm{ref}}\right)$:

$$\min_{\Delta\mathbf{x}, \Delta\mathbf{u}} \sum_{k=i}^{i+N-1} \frac{1}{2} \begin{bmatrix} \Delta\mathbf{x}_k \\ \Delta\mathbf{u}_k \end{bmatrix}^\top W_k \begin{bmatrix} \Delta\mathbf{x}_k \\ \Delta\mathbf{u}_k \end{bmatrix}$$

$$\text{s.t.} \quad \Delta\mathbf{x}_i = \hat{\mathbf{x}}_i - \mathbf{x}_i^{\mathrm{ref}}$$

$$\Delta\mathbf{x}_{k+1} = A_k \Delta\mathbf{x}_k + B_k \Delta\mathbf{u}_k + \mathbf{r}_k$$



# Deploying Linear MPC

- Nonlinear system from shooting:

$$\mathbf{x}_{k+1} = \mathbf{f}\left(\mathbf{x}_k, \mathbf{u}_k\right)$$

- Reference trajectory:

$$\mathbf{x}_k^{\mathrm{ref}}, \mathbf{u}_k^{\mathrm{ref}}$$

- Affine Time-Varying model:

$$\Delta\mathbf{x}_{k+1} = A_k \Delta\mathbf{x}_k + B_k \Delta\mathbf{u}_k + \mathbf{r}_k$$

where

$$\Delta\mathbf{x}_k = \mathbf{x}_k - \mathbf{x}_k^{\mathrm{ref}}, \quad \Delta\mathbf{u}_k = \mathbf{u}_k - \mathbf{u}_k^{\mathrm{ref}}$$

$$A_k = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\mathbf{x}_k^{\mathrm{ref}}, \mathbf{u}_k^{\mathrm{ref}}}, \quad B_k = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{u}}\right|_{\mathbf{x}_k^{\mathrm{ref}}, \mathbf{u}_k^{\mathrm{ref}}}$$

$$\mathbf{r}_k = \mathbf{f}\left(\mathbf{x}_k^{\mathrm{ref}}, \mathbf{u}_k^{\mathrm{ref}}\right) - \mathbf{x}_{k+1}^{\mathrm{ref}}$$

note $\mathbf{r}_k = 0$ if reference trajectory $\mathbf{x}_k^{\mathrm{ref}}, \mathbf{u}_k^{\mathrm{ref}}$ is feasible

$\mathrm{MPC}\left(\hat{\mathbf{x}}_i, \mathbf{x}^{\mathrm{ref}}, \mathbf{u}^{\mathrm{ref}}\right)$:

$$\begin{aligned} \min_{\Delta\mathbf{x}, \Delta\mathbf{u}} \quad & \sum_{k=i}^{i+N-1} \frac{1}{2} \begin{bmatrix} \Delta\mathbf{x}_k \\ \Delta\mathbf{u}_k \end{bmatrix}^\top W_k \begin{bmatrix} \Delta\mathbf{x}_k \\ \Delta\mathbf{u}_k \end{bmatrix} \\ \mathrm{s.t.} \quad & \Delta\mathbf{x}_i = \hat{\mathbf{x}}_i - \mathbf{x}_i^{\mathrm{ref}} \\ & \Delta\mathbf{x}_{k+1} = A_k \Delta\mathbf{x}_k + B_k \Delta\mathbf{u}_k + \mathbf{r}_k \end{aligned}$$

**What about deploying NMPC?**

$\mathrm{NMPC}\left(\hat{\mathbf{x}}_i, \mathbf{x}^{\mathrm{ref}}, \mathbf{u}^{\mathrm{ref}}\right)$:

$$\begin{aligned} \min_{\mathbf{x}, \mathbf{u}} \quad & \sum_{k=i}^{i+N-1} \frac{1}{2} \begin{bmatrix} \mathbf{x}_k - \mathbf{x}_k^{\mathrm{ref}} \\ \mathbf{u}_k - \mathbf{u}_k^{\mathrm{ref}} \end{bmatrix}^\top W_k \begin{bmatrix} \mathbf{x}_k - \mathbf{x}_k^{\mathrm{ref}} \\ \mathbf{u}_k - \mathbf{u}_k^{\mathrm{ref}} \end{bmatrix} \\ \mathrm{s.t.} \quad & \mathbf{x}_i = \hat{\mathbf{x}}_i \\ & \mathbf{x}_{k+1} = \mathbf{f}\left(\mathbf{x}_k, \mathbf{u}_k\right) \end{aligned}$$

# Deploying NMPC

$$\begin{aligned} \mathrm{NMPC}\left(\hat{\mathbf{x}}_i,\, \mathbf{x}^{\mathrm{ref}},\, \mathbf{u}^{\mathrm{ref}}\right) :\ \min_{\mathbf{x},\mathbf{u}} \ & \sum_{k=i}^{i+N-1} \frac{1}{2} \begin{bmatrix} \mathbf{x}_k - \mathbf{x}_k^{\mathrm{ref}} \\ \mathbf{u}_k - \mathbf{u}_k^{\mathrm{ref}} \end{bmatrix}^\top W_k \begin{bmatrix} \mathbf{x}_k - \mathbf{x}_k^{\mathrm{ref}} \\ \mathbf{u}_k - \mathbf{u}_k^{\mathrm{ref}} \end{bmatrix} \\ \mathrm{s.t.} \quad & \mathbf{x}_i = \hat{\mathbf{x}}_i \\ & \mathbf{x}_{k+1} = \mathbf{f}\left(\mathbf{x}_k, \mathbf{u}_k\right) \end{aligned}$$

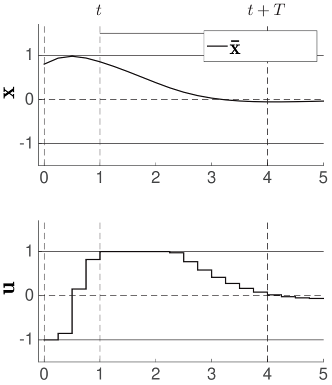
Given "guess" $\bar{\mathbf{x}}, \bar{\mathbf{u}}$, SQP iterates:

$$\begin{aligned} \min_{\Delta\mathbf{x},\Delta\mathbf{u}} \ & \sum_{k=i}^{i+N-1} \frac{1}{2} \begin{bmatrix} \Delta\mathbf{x}_k \\ \Delta\mathbf{u}_k \end{bmatrix} H_k \begin{bmatrix} \Delta\mathbf{x}_k \\ \Delta\mathbf{u}_k \end{bmatrix} + J_k^T \begin{bmatrix} \Delta\mathbf{x}_k \\ \Delta\mathbf{u}_k \end{bmatrix} \\ \mathrm{s.t.} \ & \Delta\mathbf{x}_i = \hat{\mathbf{x}}_i - \bar{\mathbf{x}}_i \\ & \Delta\mathbf{x}_{k+1} = A_k \Delta\mathbf{x}_k + B_k \Delta\mathbf{u}_k + \mathbf{r}_k \end{aligned}$$

where $\Delta\mathbf{x}, \Delta\mathbf{u}$ is the Newton step and:

$$A_k = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\bar{\mathbf{x}}_k,\bar{\mathbf{u}}_k}, \quad B_k = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{u}}\right|_{\bar{\mathbf{x}}_k,\bar{\mathbf{u}}_k}$$

$$\mathbf{r}_k = \mathbf{f}\left(\bar{\mathbf{x}}_k, \bar{\mathbf{u}}_k\right) - \bar{\mathbf{x}}_{k+1}, \qquad J_k = W_k \begin{bmatrix} \bar{\mathbf{x}}_k - \mathbf{x}_k^{\mathrm{ref}} \\ \bar{\mathbf{u}}_k - \mathbf{u}_k^{\mathrm{ref}} \end{bmatrix}$$



# Deploying NMPC

$$\text{NMPC}\left(\hat{\mathbf{x}}_i,\, \mathbf{x}^{\text{ref}},\, \mathbf{u}^{\text{ref}}\right) : \min_{\mathbf{x},\mathbf{u}} \sum_{k=i}^{i+N-1} \frac{1}{2} \begin{bmatrix} \mathbf{x}_k - \mathbf{x}_k^{\text{ref}} \\ \mathbf{u}_k - \mathbf{u}_k^{\text{ref}} \end{bmatrix}^\top W_k \begin{bmatrix} \mathbf{x}_k - \mathbf{x}_k^{\text{ref}} \\ \mathbf{u}_k - \mathbf{u}_k^{\text{ref}} \end{bmatrix}$$

$$\text{s.t.} \quad \mathbf{x}_i = \hat{\mathbf{x}}_i$$

$$\mathbf{x}_{k+1} = \mathbf{f}\left(\mathbf{x}_k, \mathbf{u}_k\right)$$

Given "guess" $\bar{\mathbf{x}}, \bar{\mathbf{u}}$, SQP iterates:

$$\min_{\Delta\mathbf{x},\Delta\mathbf{u}} \sum_{k=i}^{i+N-1} \frac{1}{2} \begin{bmatrix} \Delta\mathbf{x}_k \\ \Delta\mathbf{u}_k \end{bmatrix} H_k \begin{bmatrix} \Delta\mathbf{x}_k \\ \Delta\mathbf{u}_k \end{bmatrix} + J_k^T \begin{bmatrix} \Delta\mathbf{x}_k \\ \Delta\mathbf{u}_k \end{bmatrix}$$

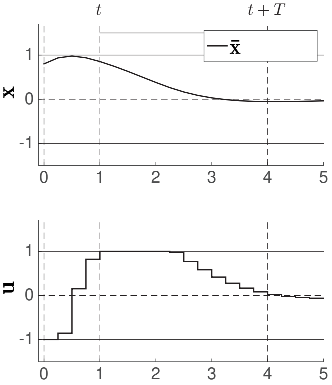
$$\text{s.t.} \quad \Delta\mathbf{x}_i = \hat{\mathbf{x}}_i - \bar{\mathbf{x}}_i$$

$$\Delta\mathbf{x}_{k+1} = A_k \Delta\mathbf{x}_k + B_k \Delta\mathbf{u}_k + \mathbf{r}_k$$

where $\Delta\mathbf{x}, \Delta\mathbf{u}$ is the Newton step and:

$$A_k = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\bar{\mathbf{x}}_k, \bar{\mathbf{u}}_k}, \quad B_k = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{u}}\right|_{\bar{\mathbf{x}}_k, \bar{\mathbf{u}}_k}$$

$$\mathbf{r}_k = \mathbf{f}\left(\bar{\mathbf{x}}_k, \bar{\mathbf{u}}_k\right) - \bar{\mathbf{x}}_{k+1}, \qquad J_k = W_k \begin{bmatrix} \bar{\mathbf{x}}_k - \mathbf{x}_k^{\text{ref}} \\ \bar{\mathbf{u}}_k - \mathbf{u}_k^{\text{ref}} \end{bmatrix}$$



# Deploying NMPC

$$\mathrm{NMPC}\left(\hat{\mathbf{x}}_i,\, \mathbf{x}^{\mathrm{ref}},\, \mathbf{u}^{\mathrm{ref}}\right) : \min_{\mathbf{x},\mathbf{u}} \sum_{k=i}^{i+N-1} \frac{1}{2} \begin{bmatrix} \mathbf{x}_k - \mathbf{x}_k^{\mathrm{ref}} \\ \mathbf{u}_k - \mathbf{u}_k^{\mathrm{ref}} \end{bmatrix}^\top W_k \begin{bmatrix} \mathbf{x}_k - \mathbf{x}_k^{\mathrm{ref}} \\ \mathbf{u}_k - \mathbf{u}_k^{\mathrm{ref}} \end{bmatrix}$$

$$\text{s.t.} \quad \mathbf{x}_i = \hat{\mathbf{x}}_i$$

$$\mathbf{x}_{k+1} = \mathbf{f}\left(\mathbf{x}_k, \mathbf{u}_k\right)$$

Given "guess" $\bar{\mathbf{x}}, \bar{\mathbf{u}}$, SQP iterates:

$$\min_{\Delta\mathbf{x},\Delta\mathbf{u}} \sum_{k=i}^{i+N-1} \frac{1}{2} \begin{bmatrix} \Delta\mathbf{x}_k \\ \Delta\mathbf{u}_k \end{bmatrix} H_k \begin{bmatrix} \Delta\mathbf{x}_k \\ \Delta\mathbf{u}_k \end{bmatrix} + J_k^T \begin{bmatrix} \Delta\mathbf{x}_k \\ \Delta\mathbf{u}_k \end{bmatrix}$$

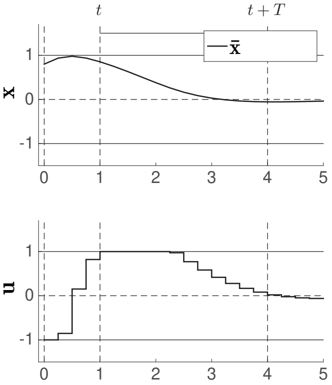
$$\text{s.t.} \quad \Delta\mathbf{x}_i = \hat{\mathbf{x}}_i - \bar{\mathbf{x}}_i$$

$$\Delta\mathbf{x}_{k+1} = A_k \Delta\mathbf{x}_k + B_k \Delta\mathbf{u}_k + \mathbf{r}_k$$

where $\Delta\mathbf{x}, \Delta\mathbf{u}$ is the Newton step and:

$$A_k = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\bar{\mathbf{x}}_k,\bar{\mathbf{u}}_k}, \quad B_k = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{u}}\right|_{\bar{\mathbf{x}}_k,\bar{\mathbf{u}}_k}$$

$$\mathbf{r}_k = \mathbf{f}\left(\bar{\mathbf{x}}_k, \bar{\mathbf{u}}_k\right) - \bar{\mathbf{x}}_{k+1}, \qquad J_k = W_k \begin{bmatrix} \bar{\mathbf{x}}_k - \mathbf{x}_k^{\mathrm{ref}} \\ \bar{\mathbf{u}}_k - \mathbf{u}_k^{\mathrm{ref}} \end{bmatrix}$$



# Deploying NMPC

$$\mathrm{NMPC}\left(\hat{\mathbf{x}}_i,\, \mathbf{x}^{\mathrm{ref}},\, \mathbf{u}^{\mathrm{ref}}\right) :\ \min_{\mathbf{x},\mathbf{u}} \ \sum_{k=i}^{i+N-1} \frac{1}{2} \begin{bmatrix} \mathbf{x}_k - \mathbf{x}_k^{\mathrm{ref}} \\ \mathbf{u}_k - \mathbf{u}_k^{\mathrm{ref}} \end{bmatrix}^\top W_k \begin{bmatrix} \mathbf{x}_k - \mathbf{x}_k^{\mathrm{ref}} \\ \mathbf{u}_k - \mathbf{u}_k^{\mathrm{ref}} \end{bmatrix}$$

$$\text{s.t.} \quad \mathbf{x}_i = \hat{\mathbf{x}}_i$$

$$\mathbf{x}_{k+1} = \mathbf{f}\left(\mathbf{x}_k, \mathbf{u}_k\right)$$

Given "guess" $\bar{\mathbf{x}}, \bar{\mathbf{u}}$, SQP iterates:

$$\min_{\Delta\mathbf{x},\Delta\mathbf{u}} \ \sum_{k=i}^{i+N-1} \frac{1}{2} \begin{bmatrix} \Delta\mathbf{x}_k \\ \Delta\mathbf{u}_k \end{bmatrix} H_k \begin{bmatrix} \Delta\mathbf{x}_k \\ \Delta\mathbf{u}_k \end{bmatrix} + J_k^T \begin{bmatrix} \Delta\mathbf{x}_k \\ \Delta\mathbf{u}_k \end{bmatrix}$$

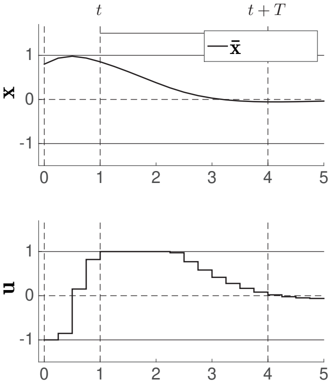
$$\text{s.t.} \quad \Delta\mathbf{x}_i = \hat{\mathbf{x}}_i - \bar{\mathbf{x}}_i$$

$$\Delta\mathbf{x}_{k+1} = A_k \Delta\mathbf{x}_k + B_k \Delta\mathbf{u}_k + \mathbf{r}_k$$

where $\Delta\mathbf{x}, \Delta\mathbf{u}$ is the Newton step and:

$$A_k = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\bar{\mathbf{x}}_k, \bar{\mathbf{u}}_k}, \quad B_k = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{u}}\right|_{\bar{\mathbf{x}}_k, \bar{\mathbf{u}}_k}$$

$$\mathbf{r}_k = \mathbf{f}\left(\bar{\mathbf{x}}_k, \bar{\mathbf{u}}_k\right) - \bar{\mathbf{x}}_{k+1}, \qquad J_k = W_k \begin{bmatrix} \bar{\mathbf{x}}_k - \mathbf{x}_k^{\mathrm{ref}} \\ \bar{\mathbf{u}}_k - \mathbf{u}_k^{\mathrm{ref}} \end{bmatrix}$$



# Deploying NMPC

$$\mathrm{NMPC}\left(\hat{\mathbf{x}}_i,\, \mathbf{x}^{\mathrm{ref}},\, \mathbf{u}^{\mathrm{ref}}\right)\;:\;\min_{\mathbf{x},\mathbf{u}}\;\sum_{k=i}^{i+N-1}\frac{1}{2}\begin{bmatrix}\mathbf{x}_k-\mathbf{x}_k^{\mathrm{ref}}\\ \mathbf{u}_k-\mathbf{u}_k^{\mathrm{ref}}\end{bmatrix}^\top W_k\begin{bmatrix}\mathbf{x}_k-\mathbf{x}_k^{\mathrm{ref}}\\ \mathbf{u}_k-\mathbf{u}_k^{\mathrm{ref}}\end{bmatrix}$$

$$\begin{aligned}\mathrm{s.t.}\quad & \mathbf{x}_i=\hat{\mathbf{x}}_i\\ & \mathbf{x}_{k+1}=\mathbf{f}\left(\mathbf{x}_k,\mathbf{u}_k\right)\end{aligned}$$

Given "guess" $\bar{\mathbf{x}}, \bar{\mathbf{u}}$, SQP iterates:

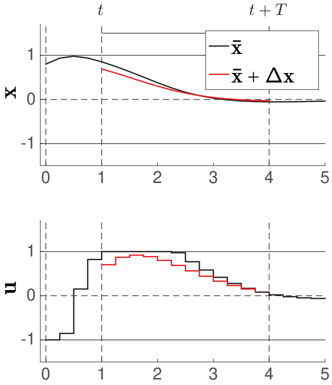
$$\min_{\Delta\mathbf{x},\Delta\mathbf{u}}\;\sum_{k=i}^{i+N-1}\frac{1}{2}\begin{bmatrix}\Delta\mathbf{x}_k\\ \Delta\mathbf{u}_k\end{bmatrix}H_k\begin{bmatrix}\Delta\mathbf{x}_k\\ \Delta\mathbf{u}_k\end{bmatrix}+J_k^T\begin{bmatrix}\Delta\mathbf{x}_k\\ \Delta\mathbf{u}_k\end{bmatrix}$$

$$\begin{aligned}\mathrm{s.t.}\quad & \Delta\mathbf{x}_i=\hat{\mathbf{x}}_i-\bar{\mathbf{x}}_i\\ & \Delta\mathbf{x}_{k+1}=A_k\Delta\mathbf{x}_k+B_k\Delta\mathbf{u}_k+\mathbf{r}_k\end{aligned}$$

where $\Delta\mathbf{x}, \Delta\mathbf{u}$ is the Newton step and:

$$A_k=\left.\frac{\partial\mathbf{f}}{\partial\mathbf{x}}\right|_{\bar{\mathbf{x}}_k,\bar{\mathbf{u}}_k},\quad B_k=\left.\frac{\partial\mathbf{f}}{\partial\mathbf{u}}\right|_{\bar{\mathbf{x}}_k,\bar{\mathbf{u}}_k}$$

$$\mathbf{r}_k=\mathbf{f}\left(\bar{\mathbf{x}}_k,\bar{\mathbf{u}}_k\right)-\bar{\mathbf{x}}_{k+1},\qquad J_k=W_k\begin{bmatrix}\bar{\mathbf{x}}_k-\mathbf{x}_k^{\mathrm{ref}}\\ \bar{\mathbf{u}}_k-\mathbf{u}_k^{\mathrm{ref}}\end{bmatrix}$$



# Deploying NMPC

$$\mathrm{NMPC}\left(\hat{\mathbf{x}}_i,\, \mathbf{x}^{\mathrm{ref}},\, \mathbf{u}^{\mathrm{ref}}\right) : \min_{\mathbf{x},\mathbf{u}} \sum_{k=i}^{i+N-1} \frac{1}{2} \left[\begin{array}{c} \mathbf{x}_k - \mathbf{x}_k^{\mathrm{ref}} \\ \mathbf{u}_k - \mathbf{u}_k^{\mathrm{ref}} \end{array}\right]^\top W_k \left[\begin{array}{c} \mathbf{x}_k - \mathbf{x}_k^{\mathrm{ref}} \\ \mathbf{u}_k - \mathbf{u}_k^{\mathrm{ref}} \end{array}\right]$$

$$\text{s.t.} \quad \mathbf{x}_i = \hat{\mathbf{x}}_i$$

$$\mathbf{x}_{k+1} = \mathbf{f}\left(\mathbf{x}_k, \mathbf{u}_k\right)$$

Given "guess" $\bar{\mathbf{x}}, \bar{\mathbf{u}}$, SQP iterates:

$$\min_{\Delta\mathbf{x},\Delta\mathbf{u}} \sum_{k=i}^{i+N-1} \frac{1}{2} \left[\begin{array}{c} \Delta\mathbf{x}_k \\ \Delta\mathbf{u}_k \end{array}\right] H_k \left[\begin{array}{c} \Delta\mathbf{x}_k \\ \Delta\mathbf{u}_k \end{array}\right] + J_k^T \left[\begin{array}{c} \Delta\mathbf{x}_k \\ \Delta\mathbf{u}_k \end{array}\right]$$

$$\text{s.t.} \quad \Delta\mathbf{x}_i = \hat{\mathbf{x}}_i - \bar{\mathbf{x}}_i$$

$$\Delta\mathbf{x}_{k+1} = A_k \Delta\mathbf{x}_k + B_k \Delta\mathbf{u}_k + \mathbf{r}_k$$

where $\Delta\mathbf{x}, \Delta\mathbf{u}$ is the Newton step and:

$$A_k = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\bar{\mathbf{x}}_k,\bar{\mathbf{u}}_k}, \quad B_k = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{u}}\right|_{\bar{\mathbf{x}}_k,\bar{\mathbf{u}}_k}$$

$$\mathbf{r}_k = \mathbf{f}\left(\bar{\mathbf{x}}_k, \bar{\mathbf{u}}_k\right) - \bar{\mathbf{x}}_{k+1}, \qquad J_k = W_k \left[\begin{array}{c} \bar{\mathbf{x}}_k - \mathbf{x}_k^{\mathrm{ref}} \\ \bar{\mathbf{u}}_k - \mathbf{u}_k^{\mathrm{ref}} \end{array}\right]$$

$\mathrm{MPC}\left(\hat{\mathbf{x}}_i,\, \mathbf{x}^{\mathrm{ref}},\, \mathbf{u}^{\mathrm{ref}}\right)$:

$$\min_{\Delta\mathbf{x},\Delta\mathbf{u}} \sum_{k=i}^{i+N-1} \frac{1}{2} \left[\begin{array}{c} \Delta\mathbf{x}_k \\ \Delta\mathbf{u}_k \end{array}\right]^\top W_k \left[\begin{array}{c} \Delta\mathbf{x}_k \\ \Delta\mathbf{u}_k \end{array}\right]$$

$$\text{s.t.} \quad \Delta\mathbf{x}_i = \hat{\mathbf{x}}_i - \mathbf{x}_i^{\mathrm{ref}}$$

$$\Delta\mathbf{x}_{k+1} = A_k \Delta\mathbf{x}_k + B_k \Delta\mathbf{u}_k + \mathbf{r}_k$$

# Deploying NMPC

$$
\begin{aligned}
\mathrm{NMPC}\left(\hat{\mathbf{x}}_i,\, \mathbf{x}^{\mathrm{ref}},\, \mathbf{u}^{\mathrm{ref}}\right) :\ \min_{\mathbf{x},\mathbf{u}} &\quad \sum_{k=i}^{i+N-1} \frac{1}{2} \begin{bmatrix} \mathbf{x}_k - \mathbf{x}_k^{\mathrm{ref}} \\ \mathbf{u}_k - \mathbf{u}_k^{\mathrm{ref}} \end{bmatrix}^\top W_k \begin{bmatrix} \mathbf{x}_k - \mathbf{x}_k^{\mathrm{ref}} \\ \mathbf{u}_k - \mathbf{u}_k^{\mathrm{ref}} \end{bmatrix} \\
\mathrm{s.t.} &\quad \mathbf{x}_i = \hat{\mathbf{x}}_i \\
&\quad \mathbf{x}_{k+1} = \mathbf{f}\left(\mathbf{x}_k, \mathbf{u}_k\right)
\end{aligned}
$$

Given "guess" $\bar{\mathbf{x}}, \bar{\mathbf{u}}$, SQP iterates:

$$
\begin{aligned}
\min_{\Delta\mathbf{x},\Delta\mathbf{u}} &\quad \sum_{k=i}^{i+N-1} \frac{1}{2} \begin{bmatrix} \Delta\mathbf{x}_k \\ \Delta\mathbf{u}_k \end{bmatrix} H_k \begin{bmatrix} \Delta\mathbf{x}_k \\ \Delta\mathbf{u}_k \end{bmatrix} + J_k^T \begin{bmatrix} \Delta\mathbf{x}_k \\ \Delta\mathbf{u}_k \end{bmatrix} \\
\mathrm{s.t.} &\quad \Delta\mathbf{x}_i = \hat{\mathbf{x}}_i - \bar{\mathbf{x}}_i \\
&\quad \Delta\mathbf{x}_{k+1} = A_k \Delta\mathbf{x}_k + B_k \Delta\mathbf{u}_k + \mathbf{r}_k
\end{aligned}
$$

where $\Delta\mathbf{x}, \Delta\mathbf{u}$ is the Newton step and:

$$
A_k = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\bar{\mathbf{x}}_k, \bar{\mathbf{u}}_k}, \quad B_k = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{u}}\right|_{\bar{\mathbf{x}}_k, \bar{\mathbf{u}}_k}
$$

$$
\mathbf{r}_k = \mathbf{f}\left(\bar{\mathbf{x}}_k, \bar{\mathbf{u}}_k\right) - \bar{\mathbf{x}}_{k+1}, \qquad J_k = W_k \begin{bmatrix} \bar{\mathbf{x}}_k - \mathbf{x}_k^{\mathrm{ref}} \\ \bar{\mathbf{u}}_k - \mathbf{u}_k^{\mathrm{ref}} \end{bmatrix}
$$

$\mathrm{MPC}\left(\hat{\mathbf{x}}_i,\, \mathbf{x}^{\mathrm{ref}},\, \mathbf{u}^{\mathrm{ref}}\right)$:

$$
\begin{aligned}
\min_{\Delta\mathbf{x},\Delta\mathbf{u}} &\quad \sum_{k=i}^{i+N-1} \frac{1}{2} \begin{bmatrix} \Delta\mathbf{x}_k \\ \Delta\mathbf{u}_k \end{bmatrix}^\top W_k \begin{bmatrix} \Delta\mathbf{x}_k \\ \Delta\mathbf{u}_k \end{bmatrix} \\
\mathrm{s.t.} &\quad \Delta\mathbf{x}_i = \hat{\mathbf{x}}_i - \mathbf{x}_i^{\mathrm{ref}} \\
&\quad \Delta\mathbf{x}_{k+1} = A_k \Delta\mathbf{x}_k + B_k \Delta\mathbf{u}_k + \mathbf{r}_k
\end{aligned}
$$

**What if guess = reference? I.e. if $\bar{\mathbf{x}} = \mathbf{x}^{\mathrm{ref}}$ ?**

# Deploying NMPC

$$\mathrm{NMPC}\left(\hat{\mathbf{x}}_i,\, \mathbf{x}^{\mathrm{ref}},\, \mathbf{u}^{\mathrm{ref}}\right) : \min_{\mathbf{x},\mathbf{u}} \sum_{k=i}^{i+N-1} \frac{1}{2}\left[\begin{array}{c} \mathbf{x}_k - \mathbf{x}_k^{\mathrm{ref}} \\ \mathbf{u}_k - \mathbf{u}_k^{\mathrm{ref}} \end{array}\right]^\top W_k \left[\begin{array}{c} \mathbf{x}_k - \mathbf{x}_k^{\mathrm{ref}} \\ \mathbf{u}_k - \mathbf{u}_k^{\mathrm{ref}} \end{array}\right]$$

$$\begin{aligned} \text{s.t.} \quad & \mathbf{x}_i = \hat{\mathbf{x}}_i \\ & \mathbf{x}_{k+1} = \mathbf{f}\left(\mathbf{x}_k, \mathbf{u}_k\right) \end{aligned}$$

Given "guess" $\bar{\mathbf{x}}, \bar{\mathbf{u}}$, SQP iterates:

$$\min_{\Delta\mathbf{x},\Delta\mathbf{u}} \sum_{k=i}^{i+N-1} \frac{1}{2}\left[\begin{array}{c} \Delta\mathbf{x}_k \\ \Delta\mathbf{u}_k \end{array}\right] H_k \left[\begin{array}{c} \Delta\mathbf{x}_k \\ \Delta\mathbf{u}_k \end{array}\right] + J_k^T \left[\begin{array}{c} \Delta\mathbf{x}_k \\ \Delta\mathbf{u}_k \end{array}\right]$$

$$\begin{aligned} \text{s.t.} \quad & \Delta\mathbf{x}_i = \hat{\mathbf{x}}_i - \bar{\mathbf{x}}_i \\ & \Delta\mathbf{x}_{k+1} = A_k \Delta\mathbf{x}_k + B_k \Delta\mathbf{u}_k + \mathbf{r}_k \end{aligned}$$

where $\Delta\mathbf{x}, \Delta\mathbf{u}$ is the Newton step and:

$$A_k = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\bar{\mathbf{x}}_k, \bar{\mathbf{u}}_k}, \quad B_k = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{u}}\right|_{\bar{\mathbf{x}}_k, \bar{\mathbf{u}}_k}$$

$$\mathbf{r}_k = \mathbf{f}\left(\bar{\mathbf{x}}_k, \bar{\mathbf{u}}_k\right) - \bar{\mathbf{x}}_{k+1}, \qquad J_k = W_k \left[\begin{array}{c} \bar{\mathbf{x}}_k - \mathbf{x}_k^{\mathrm{ref}} \\ \bar{\mathbf{u}}_k - \mathbf{u}_k^{\mathrm{ref}} \end{array}\right]$$

$\mathrm{MPC}\left(\hat{\mathbf{x}}_i,\, \mathbf{x}^{\mathrm{ref}},\, \mathbf{u}^{\mathrm{ref}}\right)$:

$$\min_{\Delta\mathbf{x},\Delta\mathbf{u}} \sum_{k=i}^{i+N-1} \frac{1}{2}\left[\begin{array}{c} \Delta\mathbf{x}_k \\ \Delta\mathbf{u}_k \end{array}\right]^\top W_k \left[\begin{array}{c} \Delta\mathbf{x}_k \\ \Delta\mathbf{u}_k \end{array}\right]$$

$$\begin{aligned} \text{s.t.} \quad & \Delta\mathbf{x}_i = \hat{\mathbf{x}}_i - \mathbf{x}_i^{\mathrm{ref}} \\ & \Delta\mathbf{x}_{k+1} = A_k \Delta\mathbf{x}_k + B_k \Delta\mathbf{u}_k + \mathbf{r}_k \end{aligned}$$

**What if guess = reference? I.e. if $\bar{\mathbf{x}} = \mathbf{x}^{\mathrm{ref}}$ ?**

SQP step = QP correction if $W_k = H_k$ (Gauss-Newton approx.)

# Deploying NMPC

$$\text{NMPC}\left(\hat{\mathbf{x}}_i,\, \mathbf{x}^{\text{ref}},\, \mathbf{u}^{\text{ref}}\right) : \min_{\mathbf{x},\mathbf{u}} \sum_{k=i}^{i+N-1} \frac{1}{2} \left[ \begin{array}{c} \mathbf{x}_k - \mathbf{x}_k^{\text{ref}} \\ \mathbf{u}_k - \mathbf{u}_k^{\text{ref}} \end{array} \right]^\top W_k \left[ \begin{array}{c} \mathbf{x}_k - \mathbf{x}_k^{\text{ref}} \\ \mathbf{u}_k - \mathbf{u}_k^{\text{ref}} \end{array} \right]$$

$$\text{s.t.} \quad \mathbf{x}_i = \hat{\mathbf{x}}_i$$

$$\mathbf{x}_{k+1} = \mathbf{f}\left(\mathbf{x}_k, \mathbf{u}_k\right)$$

Given "guess" $\bar{\mathbf{x}}, \bar{\mathbf{u}}$, SQP iterates:

$$\min_{\Delta\mathbf{x},\Delta\mathbf{u}} \sum_{k=i}^{i+N-1} \frac{1}{2} \left[ \begin{array}{c} \Delta\mathbf{x}_k \\ \Delta\mathbf{u}_k \end{array} \right] H_k \left[ \begin{array}{c} \Delta\mathbf{x}_k \\ \Delta\mathbf{u}_k \end{array} \right] + J_k^T \left[ \begin{array}{c} \Delta\mathbf{x}_k \\ \Delta\mathbf{u}_k \end{array} \right]$$

$$\text{s.t.} \quad \Delta\mathbf{x}_i = \hat{\mathbf{x}}_i - \bar{\mathbf{x}}_i$$

$$\Delta\mathbf{x}_{k+1} = A_k \Delta\mathbf{x}_k + B_k \Delta\mathbf{u}_k + \mathbf{r}_k$$

where $\Delta\mathbf{x}, \Delta\mathbf{u}$ is the Newton step and:

$$A_k = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\bar{\mathbf{x}}_k,\bar{\mathbf{u}}_k}, \quad B_k = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{u}}\right|_{\bar{\mathbf{x}}_k,\bar{\mathbf{u}}_k}$$

$$\mathbf{r}_k = \mathbf{f}\left(\bar{\mathbf{x}}_k, \bar{\mathbf{u}}_k\right) - \bar{\mathbf{x}}_{k+1}, \qquad J_k = W_k \left[ \begin{array}{c} \bar{\mathbf{x}}_k - \mathbf{x}_k^{\text{ref}} \\ \bar{\mathbf{u}}_k - \mathbf{u}_k^{\text{ref}} \end{array} \right]$$

**Bottom line**:

- NMPC with RTI "re-linearize" the dynamics at every time step based on the previous (shifted) solution.
- Regular MPC linearizes at the reference

# From linear MPC to NMPC

$$\mathrm{NMPC}\left(\hat{\mathbf{x}}_i,\, \mathbf{x}^{\mathrm{ref}},\, \mathbf{u}^{\mathrm{ref}}\right) :\ \min_{\mathbf{x},\mathbf{u}} \sum_{k=i}^{i+N-1} \frac{1}{2} \begin{bmatrix} \mathbf{x}_k - \mathbf{x}_k^{\mathrm{ref}} \\ \mathbf{u}_k - \mathbf{u}_k^{\mathrm{ref}} \end{bmatrix}^\top W_k \begin{bmatrix} \mathbf{x}_k - \mathbf{x}_k^{\mathrm{ref}} \\ \mathbf{u}_k - \mathbf{u}_k^{\mathrm{ref}} \end{bmatrix}$$

$$\begin{aligned} \mathrm{s.t.} \quad & \mathbf{x}_i = \hat{\mathbf{x}}_i \\ & \mathbf{x}_{k+1} = \mathbf{f}\left(\mathbf{x}_k, \mathbf{u}_k\right) \end{aligned}$$

$\mathrm{MPC}\left(\hat{\mathbf{x}}_i,\, \mathbf{x}^{\mathrm{ref}},\, \mathbf{u}^{\mathrm{ref}}\right)$:

$$\min_{\Delta\mathbf{x},\Delta\mathbf{u}} \sum_{k=i}^{i+N-1} \frac{1}{2} \begin{bmatrix} \Delta\mathbf{x}_k \\ \Delta\mathbf{u}_k \end{bmatrix}^\top W_k \begin{bmatrix} \Delta\mathbf{x}_k \\ \Delta\mathbf{u}_k \end{bmatrix}$$

$$\begin{aligned} \mathrm{s.t.} \quad & \Delta\mathbf{x}_i = \hat{\mathbf{x}}_i - \mathbf{x}_i^{\mathrm{ref}} \\ & \Delta\mathbf{x}_{k+1} = A_k \Delta\mathbf{x}_k + B_k \Delta\mathbf{u}_k + \mathbf{r}_k \end{aligned}$$

**Linear MPC uses** $\bar{\mathbf{x}}_k = \mathbf{x}^{\mathrm{ref}}, \bar{\mathbf{u}}_k = \mathbf{u}^{\mathrm{ref}}$:

- Only once, offline form at :

$$A_k, \quad B_k,\ J_k = 0,\ \mathbf{r}_k$$

- Online: solve QP

**At every time instant, RTI does**:

- Shift previous solution to get $\bar{\mathbf{x}}_k, \bar{\mathbf{u}}_k$
- Form at $\bar{\mathbf{x}}_k, \bar{\mathbf{u}}_k$:

$$A_k,\ B_k,\ J_k,\ \mathbf{r}_k$$

- Solve QP

- RTI ≡ linear MPC + update of the linearization based on our best guess
- If integrators are fast, then NMPC is “as fast as” linear MPC

From Linear to Nonlinear MPC: Bridging the Gap via the Real-Time Iteration, S. Gros, M. Zanon, R. Quirynen, A. Bemporad, M. Diehl