
Topics on Fast NMPC . . .

The control updating period issue . . .

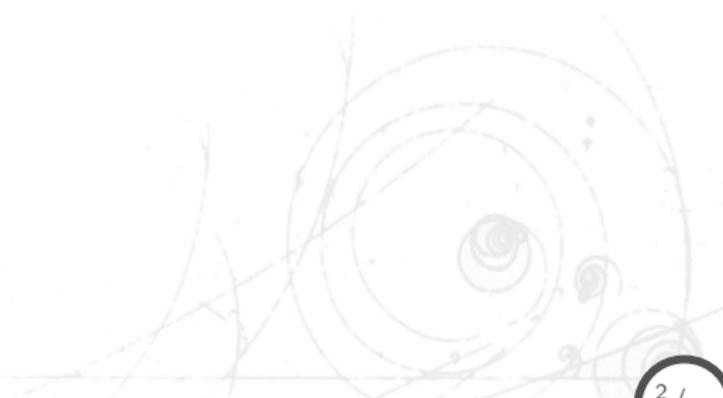
Mazen Alamir

GIPSA-Lab, CNRS, University of Grenoble, France.



Outline

- Fast NMPC
- The Control Updating Period Paradigm
- The Proposed Updating Scheme
- Fast Gradient Related Considerations
- Numerical Investigation
- Conclusion & future works



Dynamical Systems

$$\dot{x} = f(x, u) \quad \rightarrow \text{sampling time } \tau$$

Control parameterization (over some prediction horizon)

$$\mathcal{U}_{pwc}(p) := (\textcolor{red}{u}^{(1)}(p), \dots, u^{(N)}(p)) \in \mathbb{U} \subset \mathbb{R}^{N \cdot n_u}$$

Open-Loop optimization problem

$$\mathcal{P}(x) : \hat{p}(x) := \arg \min_p J(p, x) \quad \text{under } C(p, x) \leq 0$$

Ideal NMPC feedback

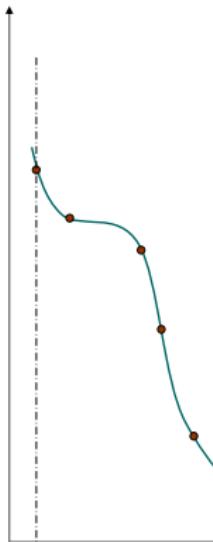
$$u(k) := \textcolor{red}{u}^{(1)}(\hat{p}(x(k)))$$

This suppose that the computation of $\hat{p}(x)$ can be done in a fraction of the sampling period τ . We call Fast systems those systems requiring sampling periods for which this is not true.

Real-Time NMPC

The principle

$$J(x(t_k), T_p, \textcolor{red}{p})$$



Real-Time NMPC

The principle

$$J(x(t_k), T_p, \textcolor{red}{p})$$

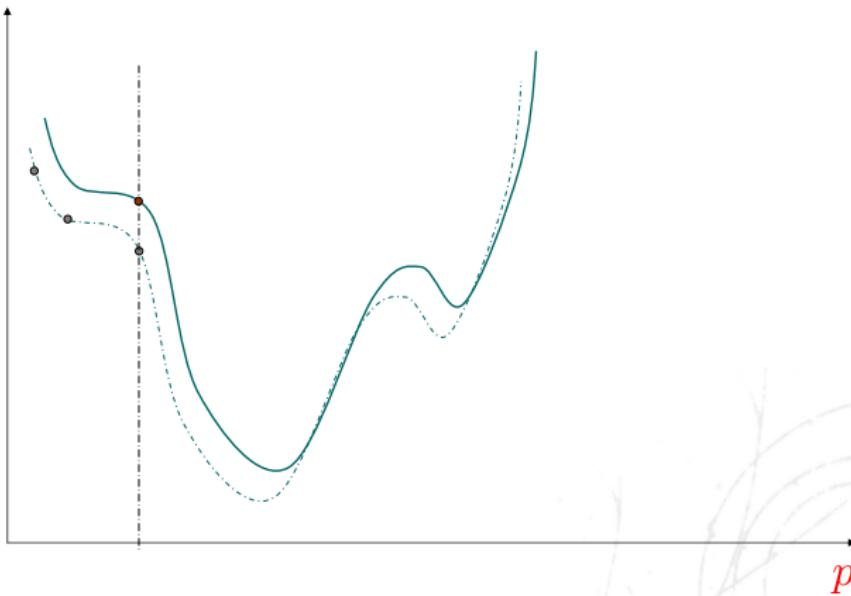


$\textcolor{red}{p}$

Real-Time NMPC

The principle

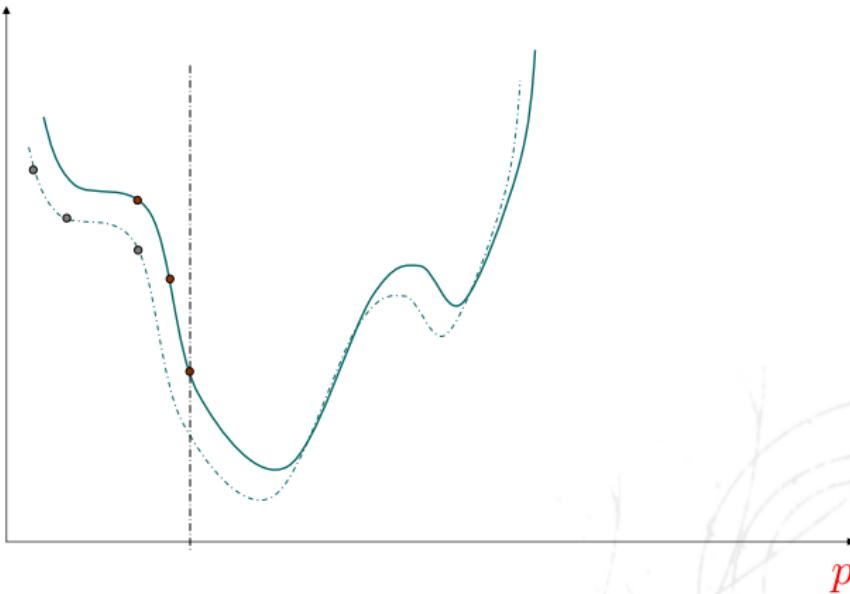
$$J(x(t_{k+1}), T_p, \textcolor{red}{p})$$



Real-Time NMPC

The principle

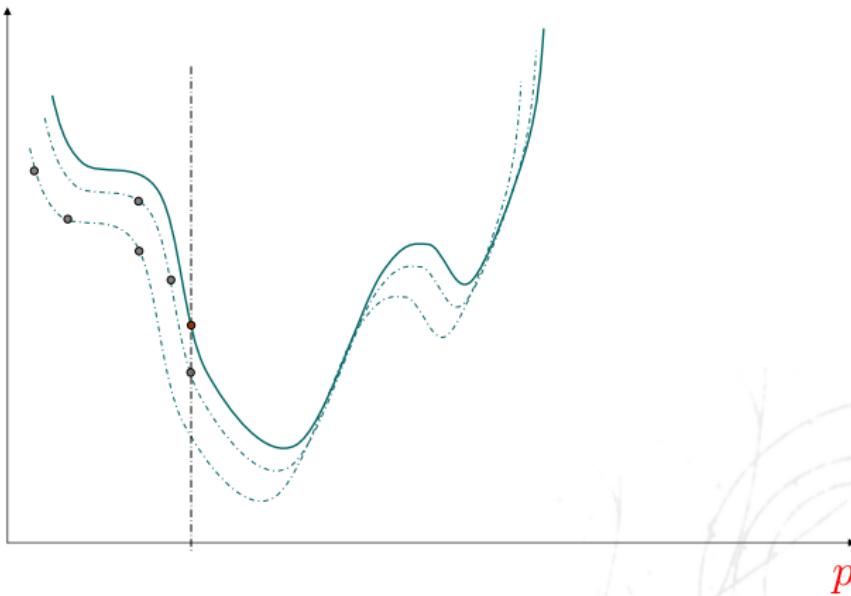
$$J(x(t_{k+1}), T_p, \textcolor{red}{p})$$



Real-Time NMPC

The principle

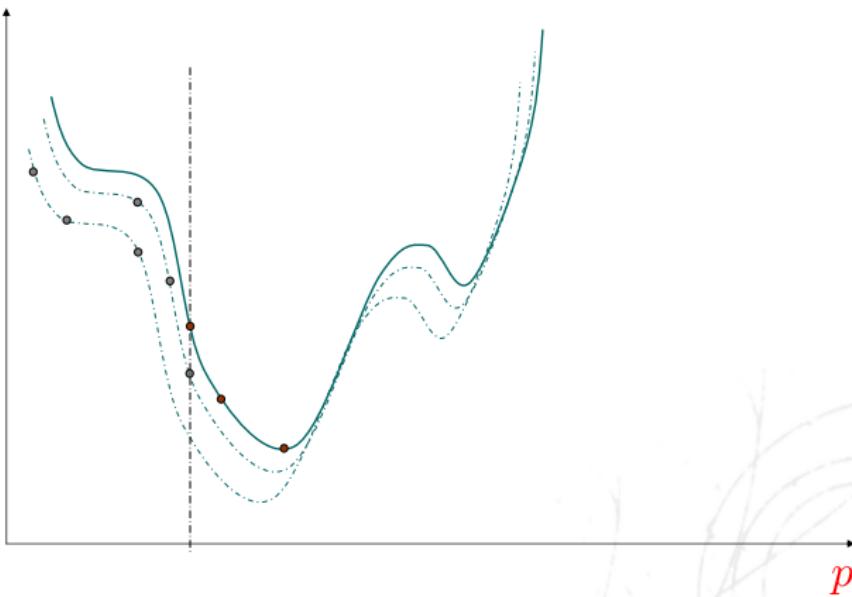
$$J(x(t_{k+2}), T_p, \textcolor{red}{p})$$



Real-Time NMPC

The principle

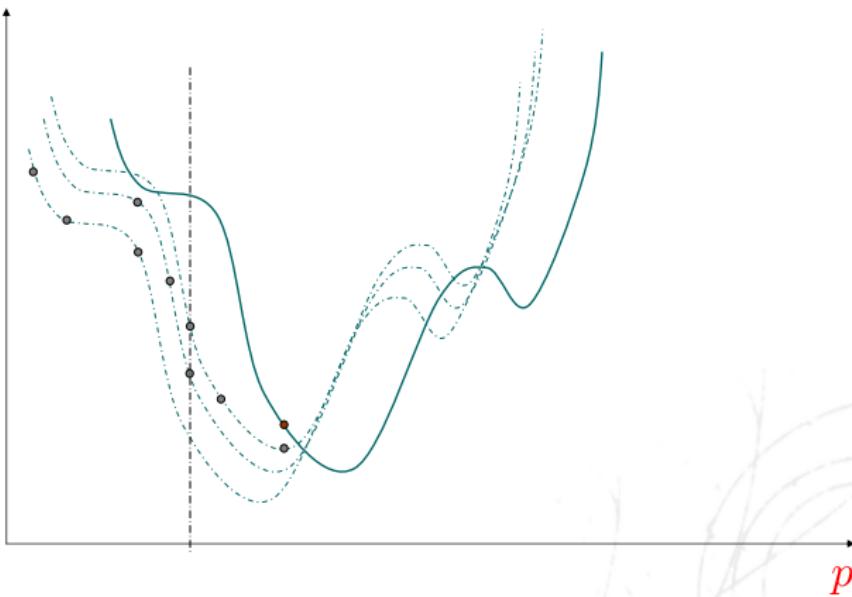
$$J(x(t_{k+2}), T_p, \textcolor{red}{p})$$



Real-Time NMPC

The principle

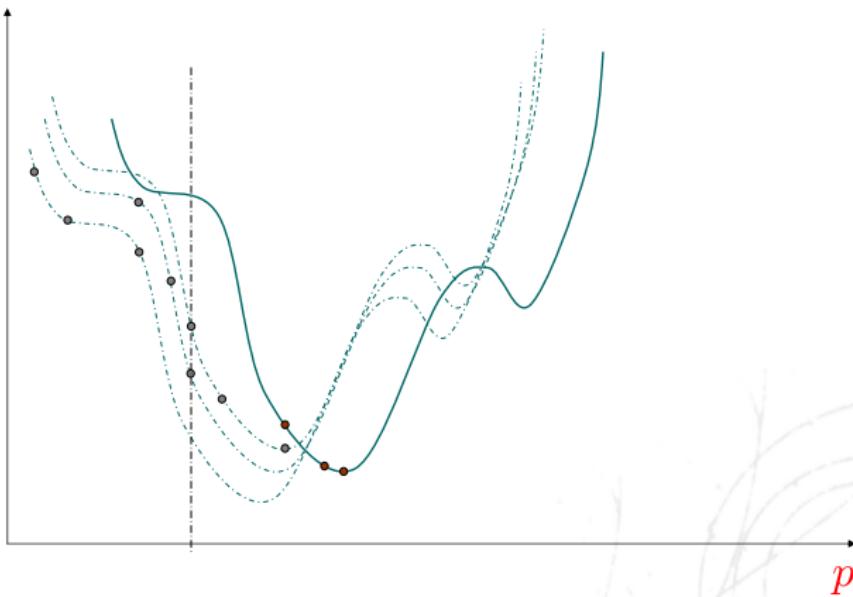
$$J(x(t_{k+3}), T_p, \textcolor{red}{p})$$



Real-Time NMPC

The principle

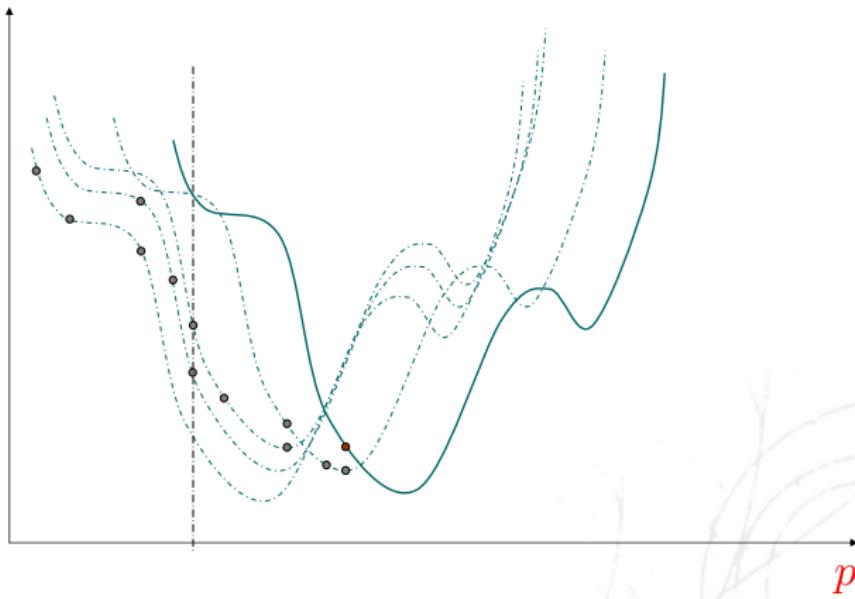
$$J(x(t_{k+3}), T_p, \textcolor{red}{p})$$



Real-Time NMPC

The principle

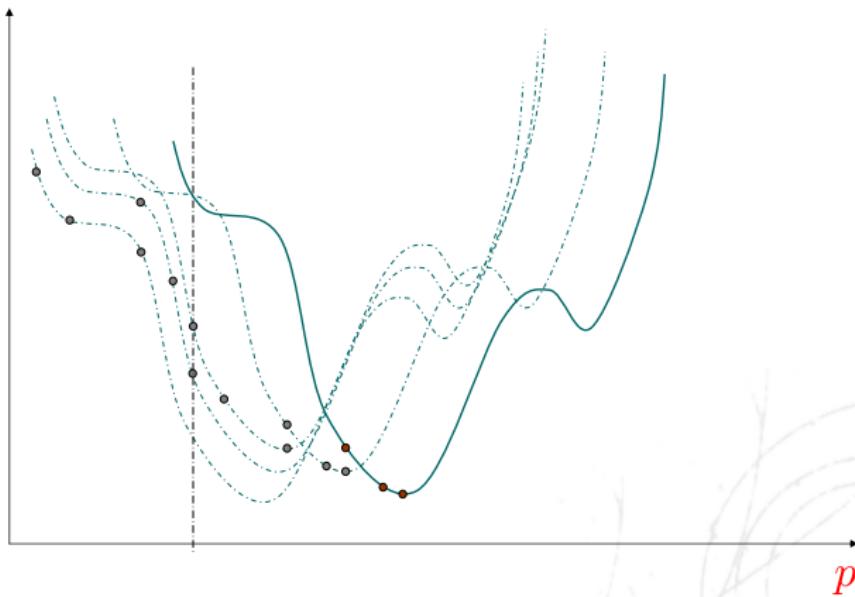
$$J(x(t_{k+4}), T_p, \textcolor{red}{p})$$



Real-Time NMPC

The principle

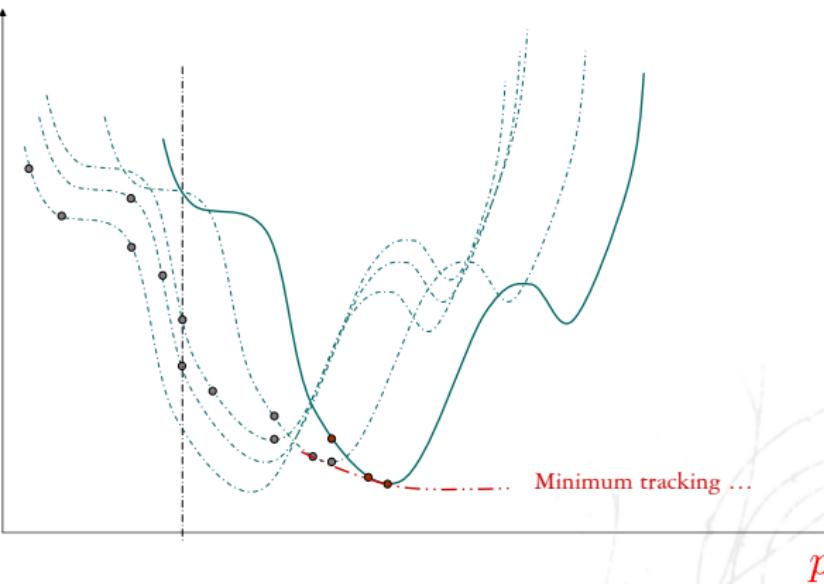
$$J(x(t_{k+4}), T_p, \textcolor{red}{p})$$



Real-Time NMPC

The principle

$$J(x(t_{k+4}), T_p, \textcolor{red}{p})$$





PERGAMON

Control Engineering Practice 9 (2001) 107–116

CONTROL ENGINEERING
PRACTICE

www.elsevier.com/locate/conengprac

Nonlinear receding horizon sub-optimal guidance law for the minimum interception time problem

Mazen Alamir*

Laboratoire d'Automatique de Grenoble, UMR 5528, CNRS-INPG-UJF, BP 46 Domaine Universitaire, Saint Martin d'Hères, France

Received 8 September 1999; accepted 18 November 1999

Page 5 : "In the *distributed* Newton Method, the Newton iterations are distributed over the successive sampling periods. One step at each sampling period."

Real-Time NMPC

A bit of history ...

M. Alamir, Control Engineering Practice, **2001**.

T. Ohtsuka, Automatica, **2004**.

M. Diehl et al. SIAM J. Control and Optimization, **2005**.

IFAC Workshop on NMPC for Fast Systems, Grenoble, **2006**.

M. Alamir, Springer, **2006**. Embedding numerical libraries is an issue in applications for memory, certification and reliability reasons.

D. DeHaan, M. Guay, Springer LNCIS, **2007**.

Y. Wang, S. Boyd, IEEE CST, **2010**.

B. Houska et al. OCAM, **2010**.

...

Fast Gradient (Y. Nesterov, 1983) related NMPC implementation :

P. Zometa et al. ACC, **2012**

A. Bemporad, P. Patrinos, IFAC NMPC2012

C. Jones et al., IFAC NMPC2012

...

Examples Diesel engine

- BMW M47D Diesel Engine (Linz)
- Sampling time $\tau = 50$ ms
- Prediction Horizon $30 \cdot \tau$
- Observation Horizon $N_O = 10\tau$
- Solver SQP/Trust region
- Number of iteration $q = 30$
- NL model, 11 states, 2 controls



André Murilo



R. Fürhapter

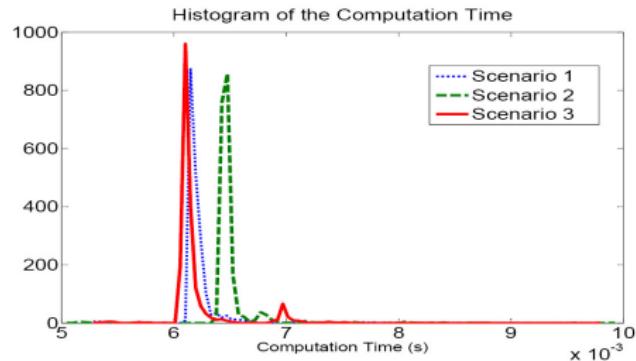


Peter Ortner

Alamir et al. Automotive MPC: Models, Methods & Applications, Linz, Austria, 2009

Examples Diesel engine

- BMW M47D Diesel Engine (Linz)
- Sampling time $\tau = 50$ ms
- Prediction Horizon $30 \cdot \tau$
- Observation Horizon $N_O = 10\tau$
- Solver SQP/Trust region
- Number of iteration $q = 30$
- NL model, 11 states, 2 controls



André Murilo



R. Fürhapter



Peter Ortner

Alamir et al. Automotive MPC: Models, Methods & Applications, Linz, Austria, 2009

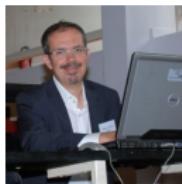
Examples AMT (Automated Manual Transmission)

- Only 0.3 Mb for control software
- Base sampling time 1 ms
- Preparation + 1 iteration $\approx 50\mu s$
- Number of iteration $q = 21$

Hybridation légère d'un véhicule roulant au gaz naturel



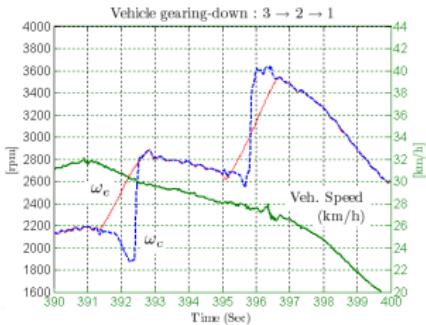
Rachid Amari



Paolo Tona

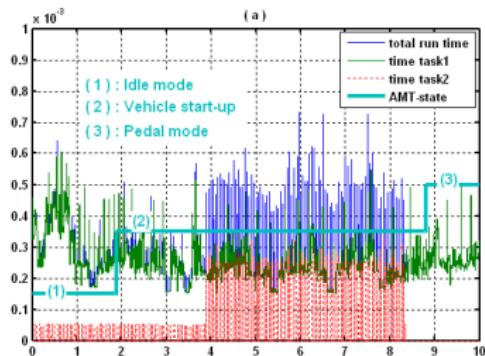
Amari et al. IFAC WC, Seoul, 2008

Alamir et al. Automotive MPC: Models, Methods & Applications, Linz, Austria, 2009



Examples AMT (Automated Manual Transmission)

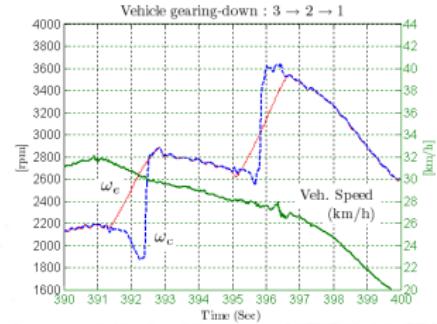
- Only 0.3 Mb for control software
- Base sampling time 1 ms
- Preparation + 1 iteration $\approx 50\mu s$
- Number of iteration $q = 21$



Rachid Amari



Paolo Tona



Amari et al. IFAC WC, Seoul, 2008

Alamir et al. Automotive MPC: Models, Methods & Applications, Linz, Austria, 2009

Examples

Twin Pendulum

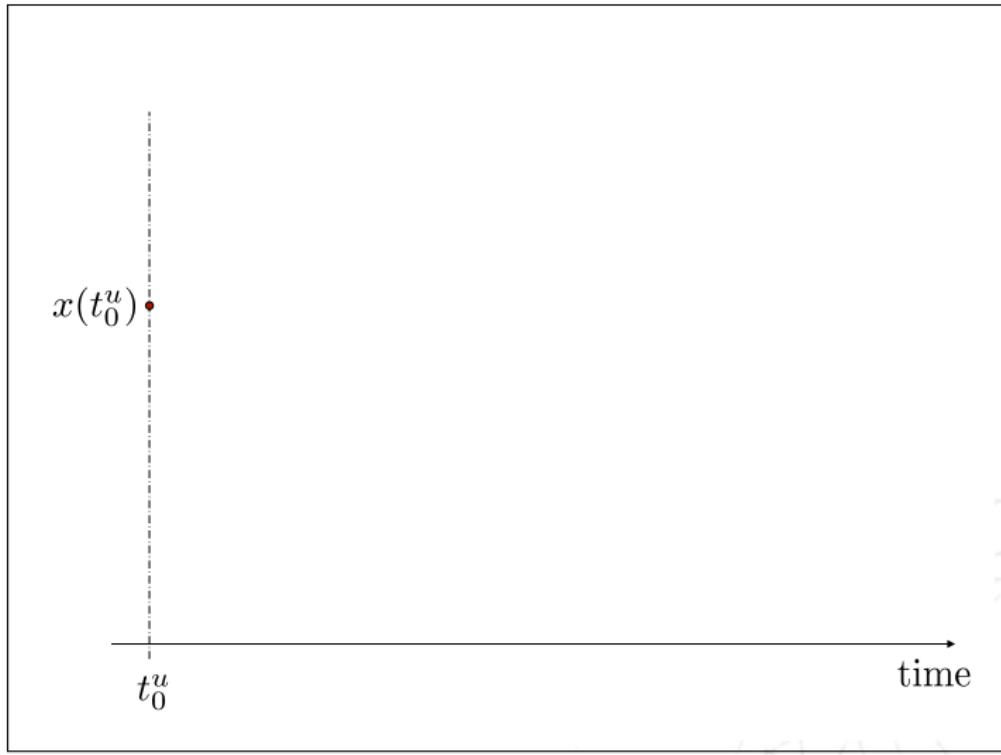
- NL system, 6 state, 1 control
- Highly unstable
- Sampling period 200 ms
- Number of iterations $q = 20$.



M. Alamir, A. Murilo, Automatica, 2008.

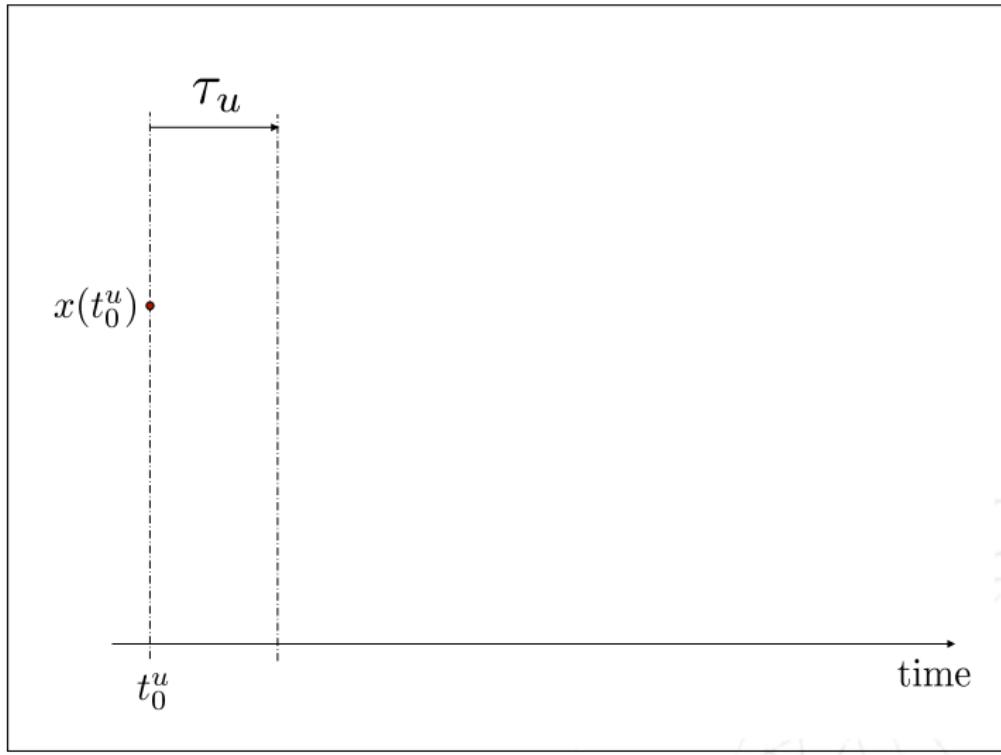
Real-Time NMPC

More precisely



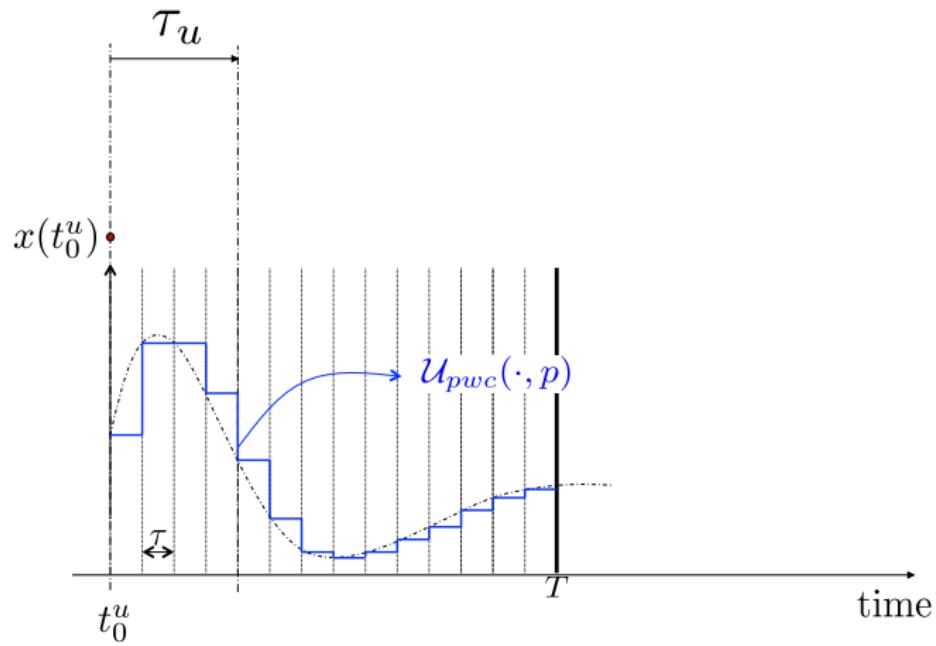
Real-Time NMPC

More precisely



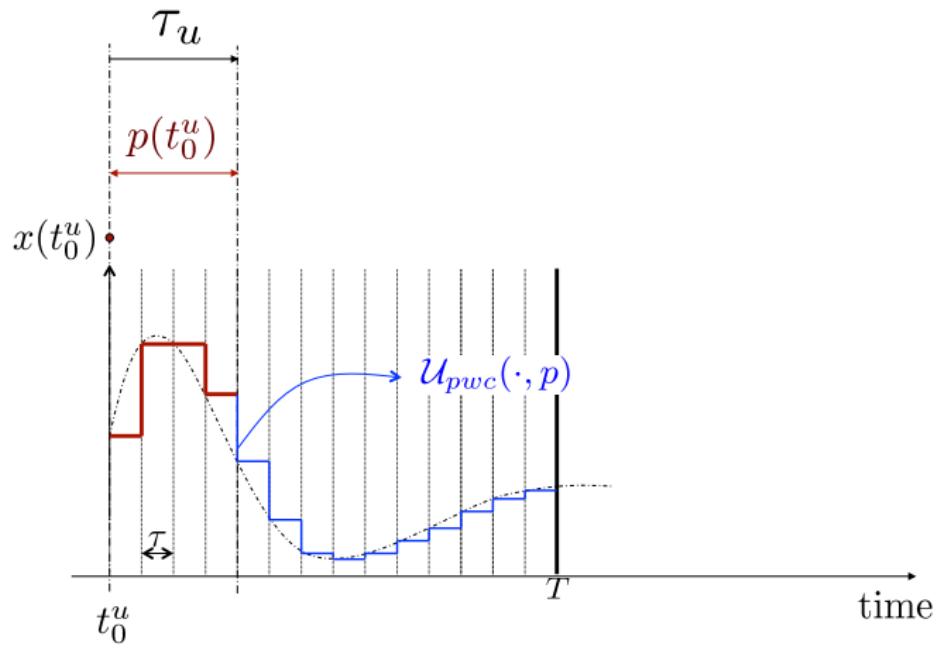
Real-Time NMPC

More precisely



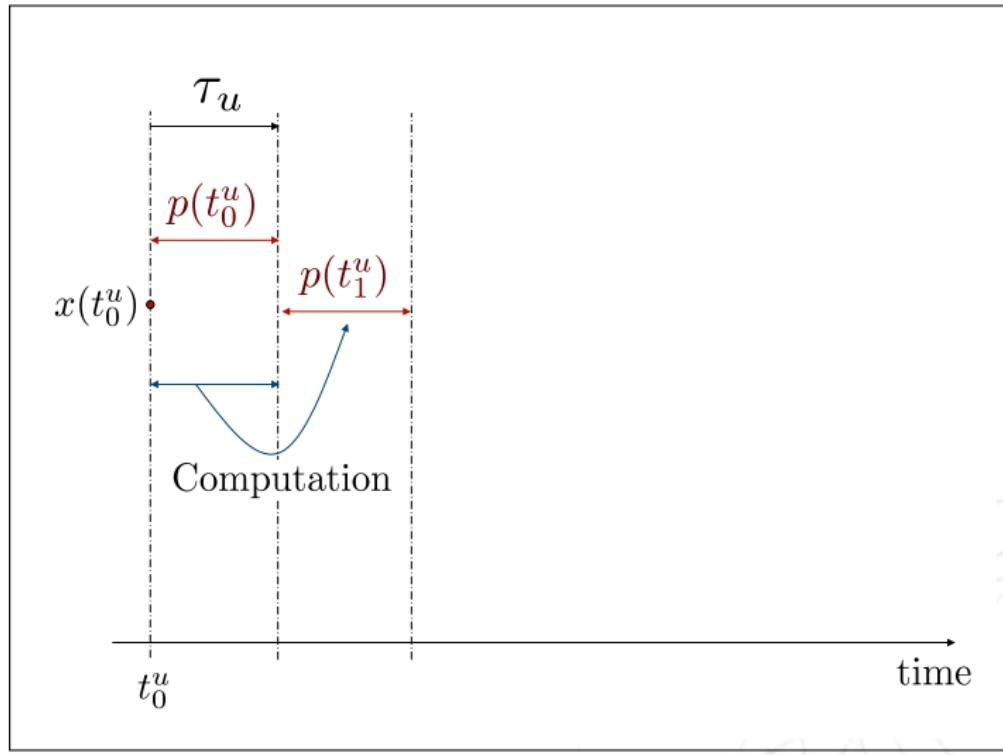
Real-Time NMPC

More precisely



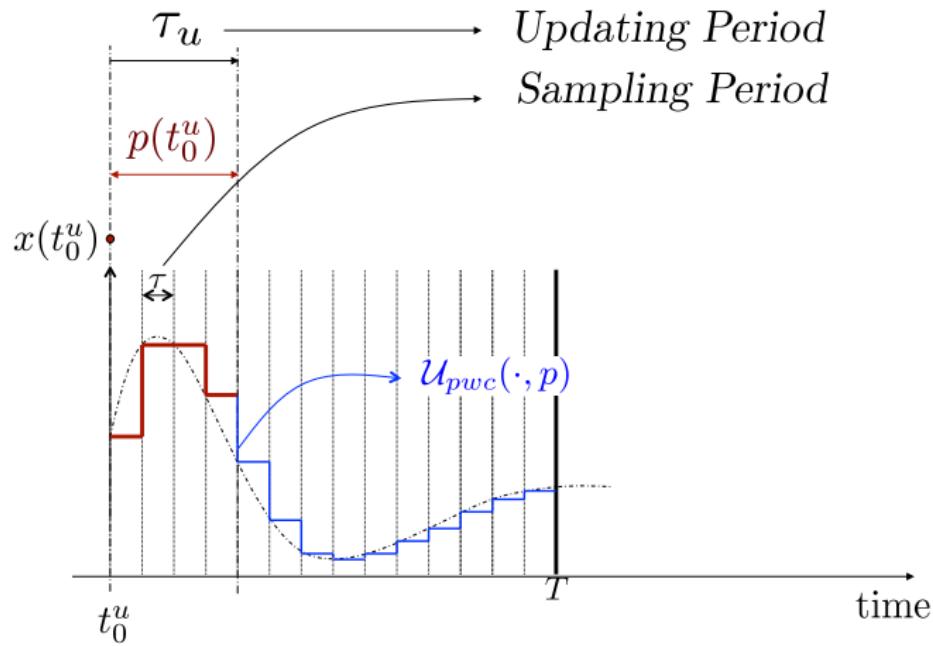
Real-Time NMPC

More precisely



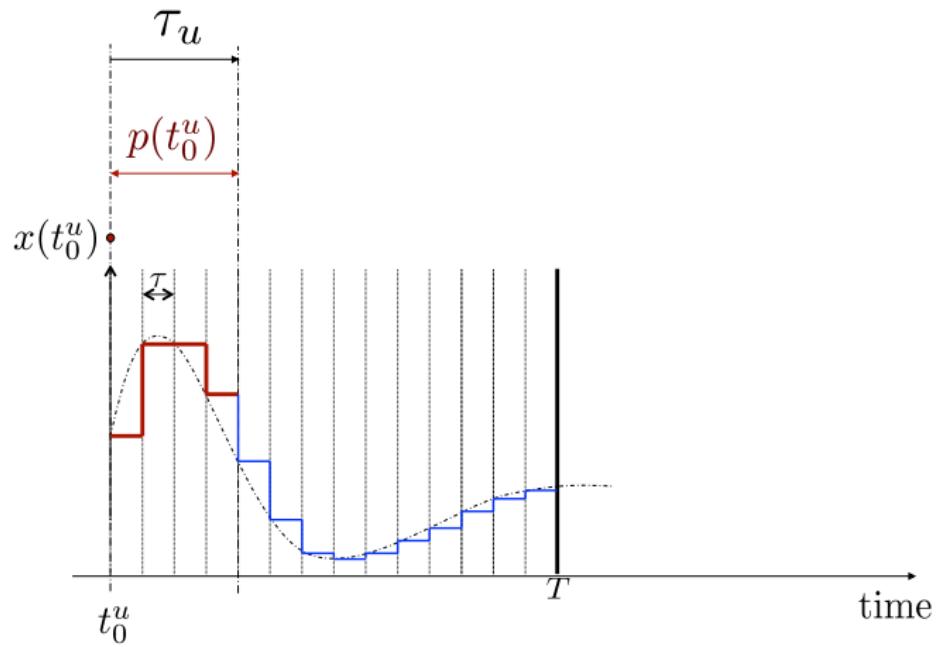
Real-Time NMPC

More precisely



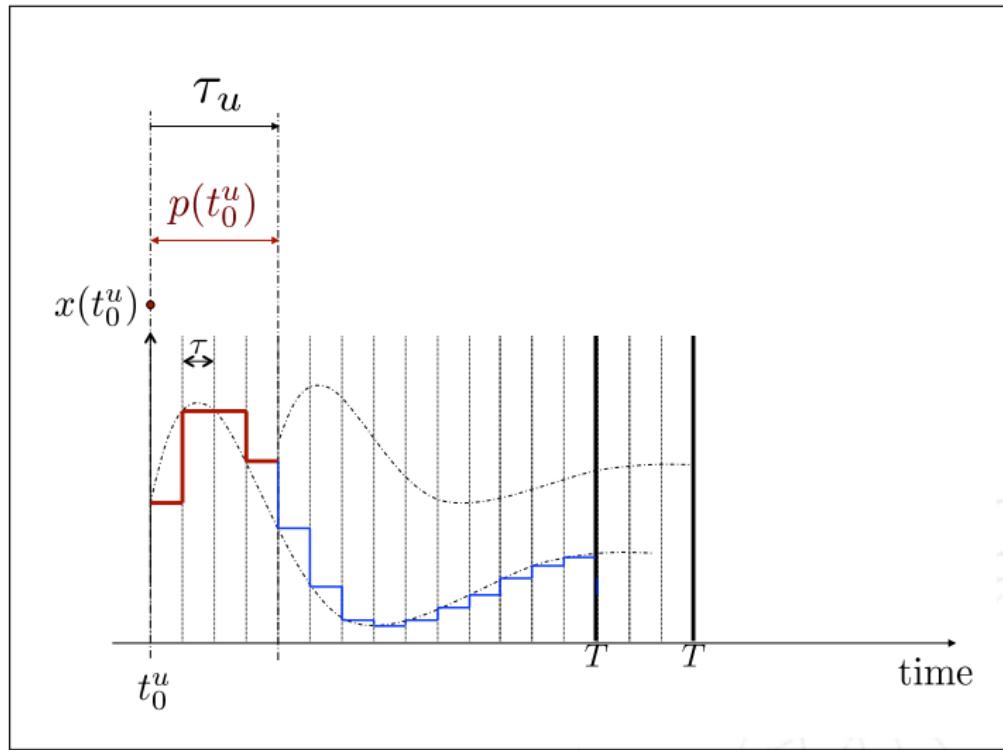
Real-Time NMPC

More precisely



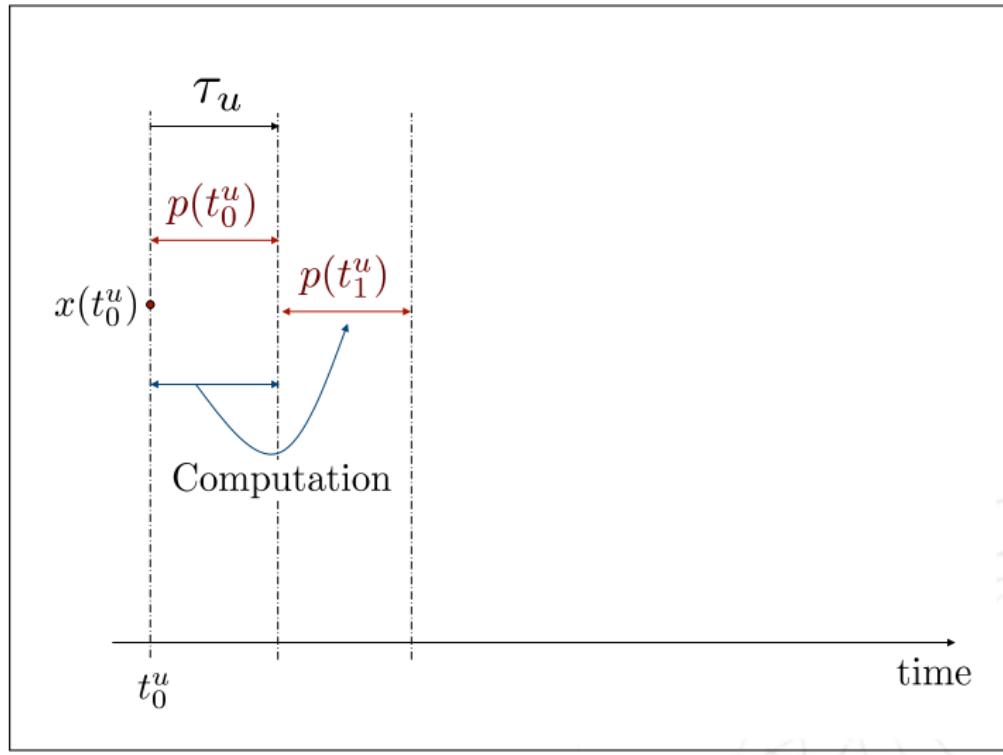
Real-Time NMPC

More precisely



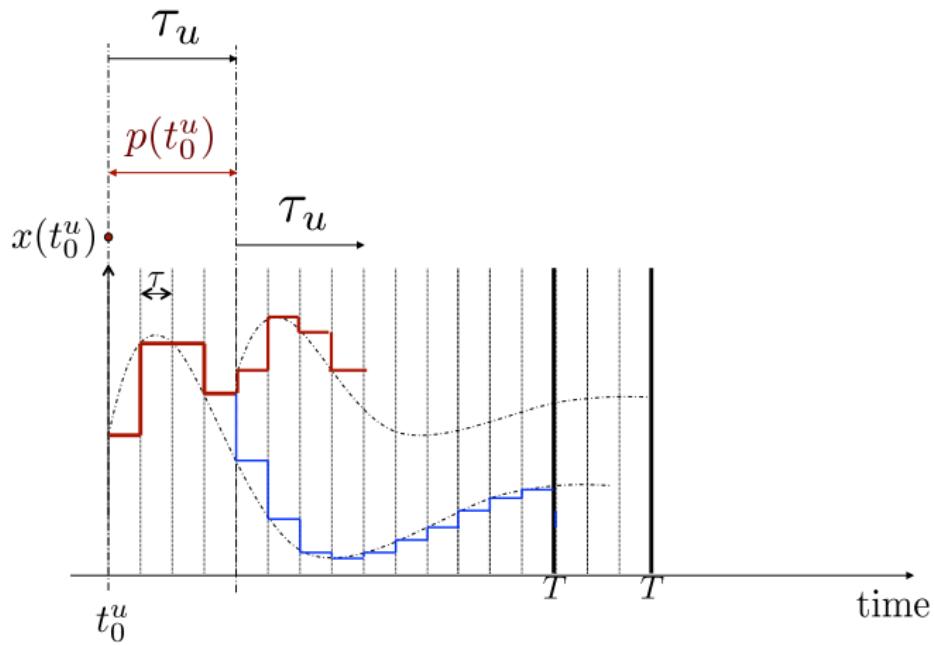
Real-Time NMPC

More precisely



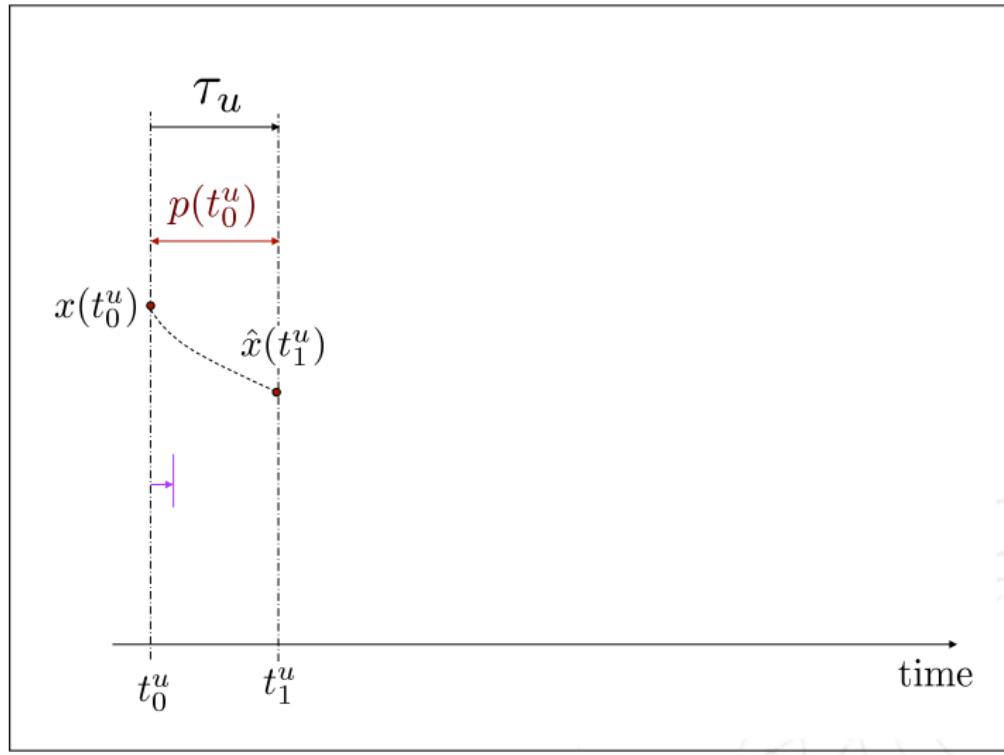
Real-Time NMPC

More precisely



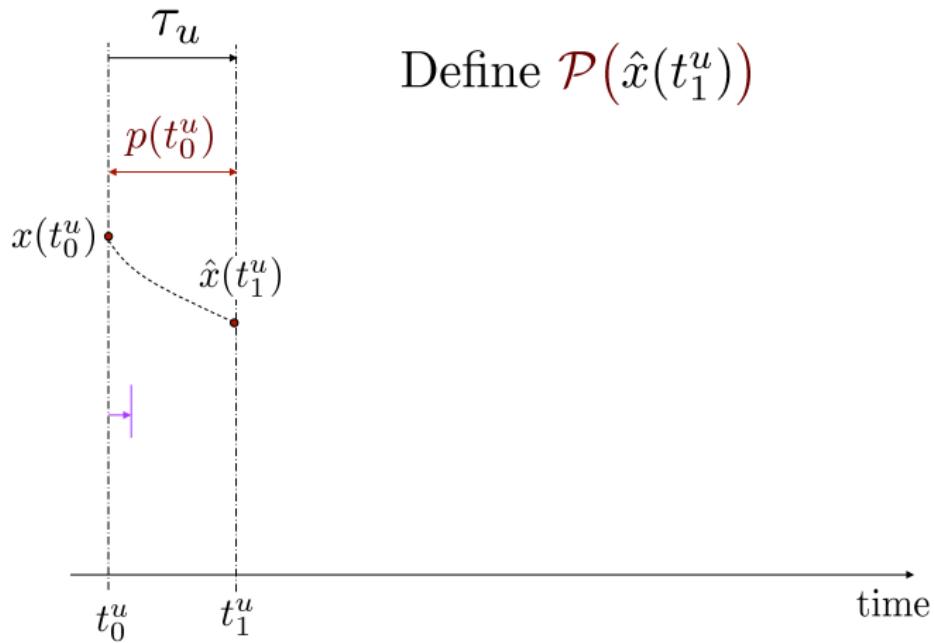
Real-Time NMPC

More precisely



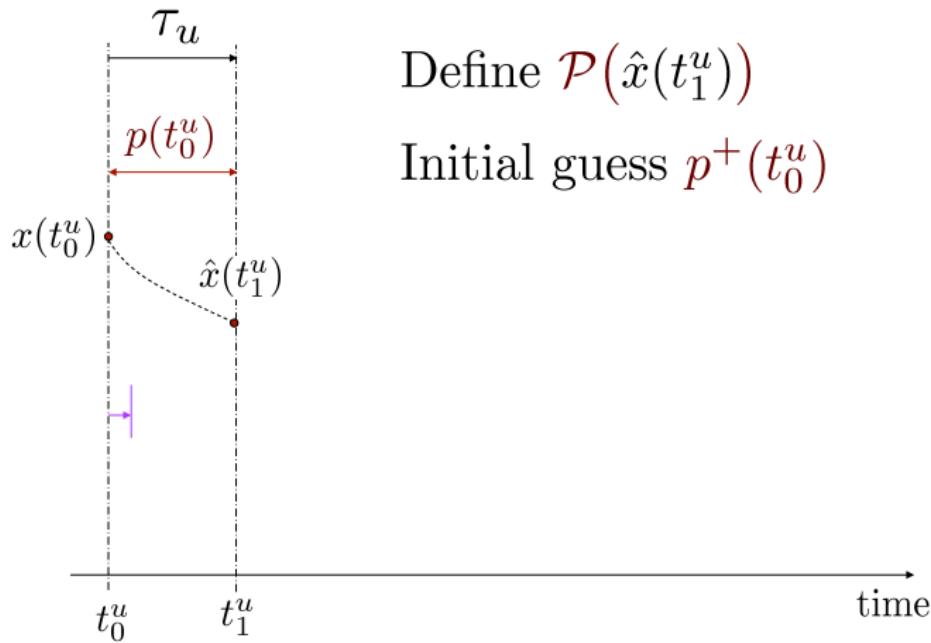
Real-Time NMPC

More precisely



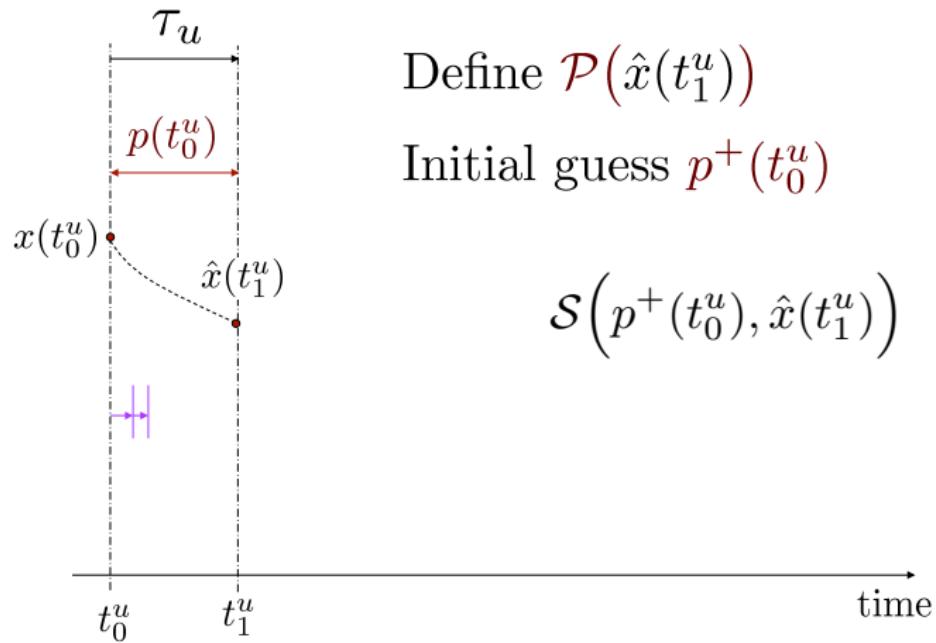
Real-Time NMPC

More precisely



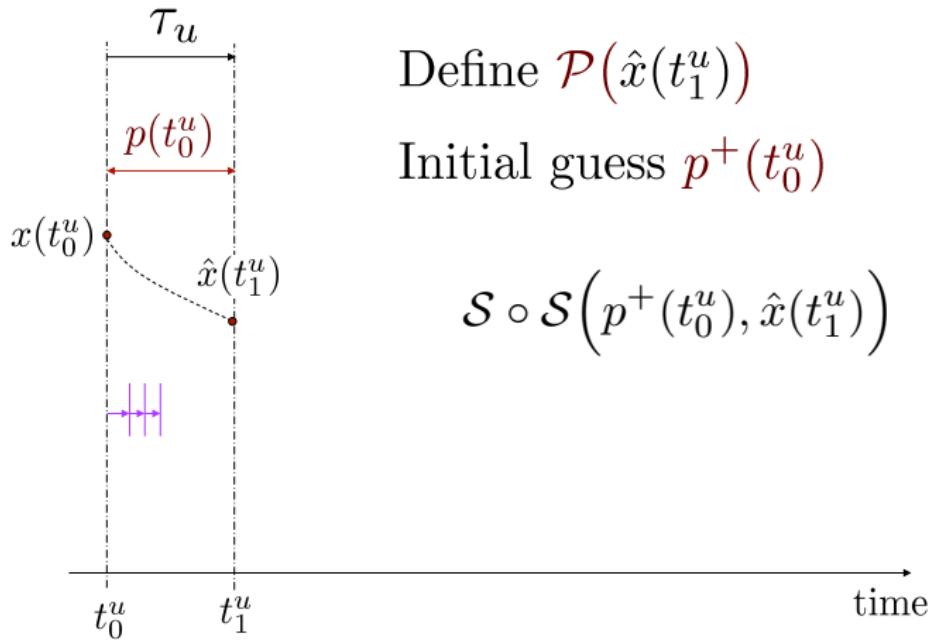
Real-Time NMPC

More precisely



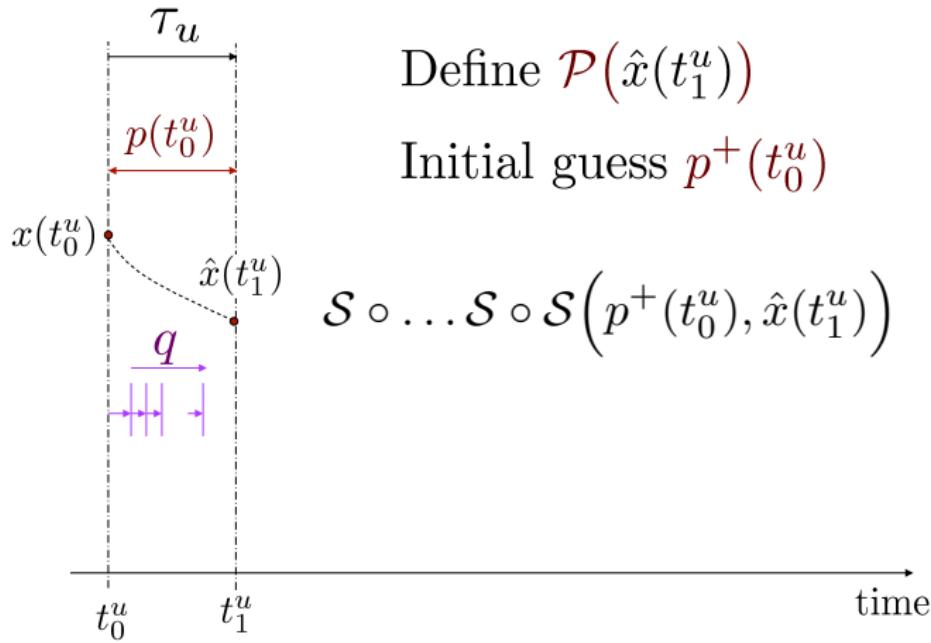
Real-Time NMPC

More precisely



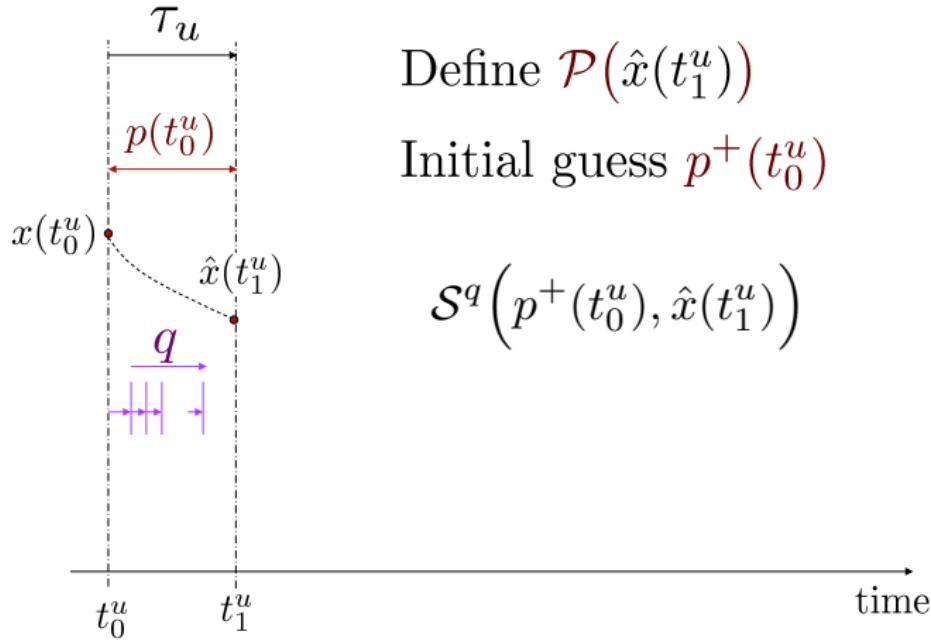
Real-Time NMPC

More precisely



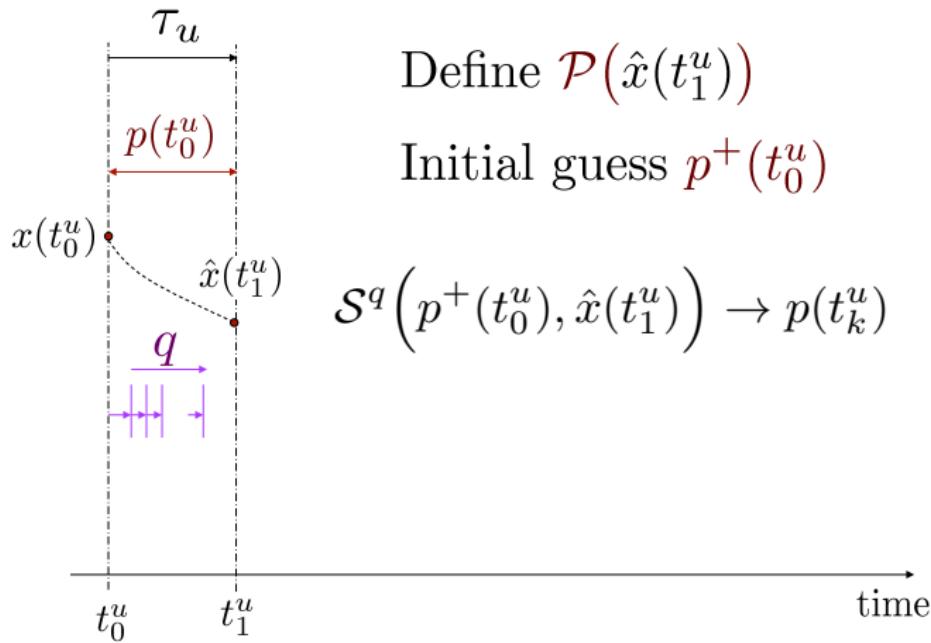
Real-Time NMPC

More precisely



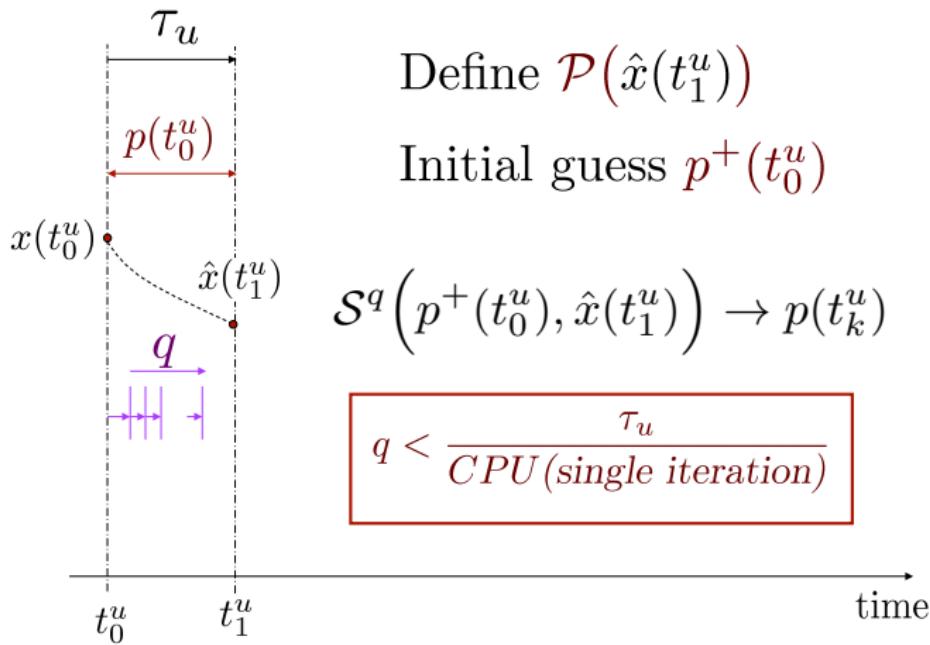
Real-Time NMPC

More precisely



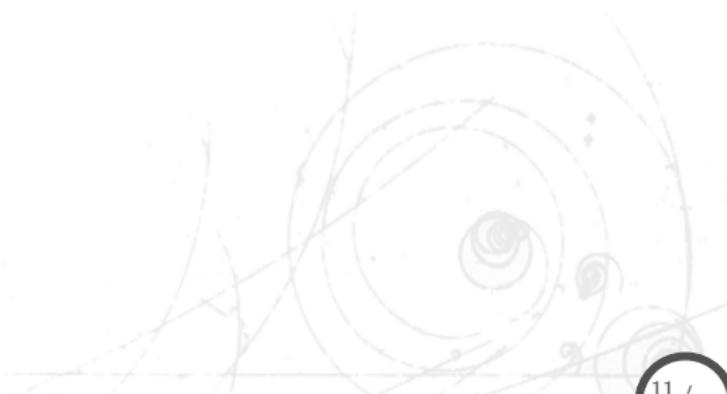
Real-Time NMPC

More precisely

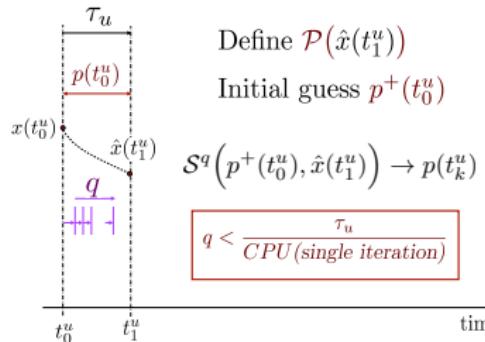


Outline

- Fast NMPC
- **The Control Updating Period Paradigm**
- The Proposed Updating Scheme
- Fast Gradient Related Considerations
- Numerical Investigation
- Conclusion & future works



The Control Updating Period Paradigm



This results in the extended dynamic system:

$$\begin{aligned}x(t_{k+1}^u) &= X^r(q\tau, \mathbf{x}(t_k^u), \mathbf{p}(t_k^u), \mathbf{w}) \\p(t_{k+1}^u) &= \mathcal{S}^q\left(p^+(t_k^u), \underbrace{X(q\tau, \mathbf{x}(t_k^u), \mathbf{p}(t_k^u))}_{\hat{x}(t_{k+1}^u)}\right)\end{aligned}$$

The Control Updating Period Paradigm

Given \mathcal{S}

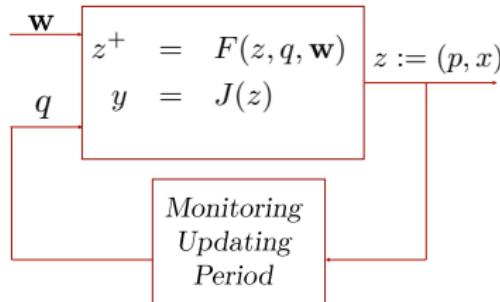
$$\begin{array}{lcl} \text{w} & z^+ = F(z, q, \mathbf{w}) \\ q & y = J(z) \end{array} \quad z := (p, x)$$

This results in the extended dynamic system:

$$\begin{aligned} x(t_{k+1}^u) &= X^r(q\tau, \color{red}{x(t_k^u)}, \color{teal}{p(t_k^u)}, \mathbf{w}) \\ p(t_{k+1}^u) &= \mathcal{S}^q \left(p^+(t_k^u), \underbrace{X(q\tau, \color{red}{x(t_k^u)}, \color{teal}{p(t_k^u)})}_{\hat{x}(t_{k+1}^u)} \right) \end{aligned}$$

The Control Updating Period Paradigm

Given \mathcal{S}

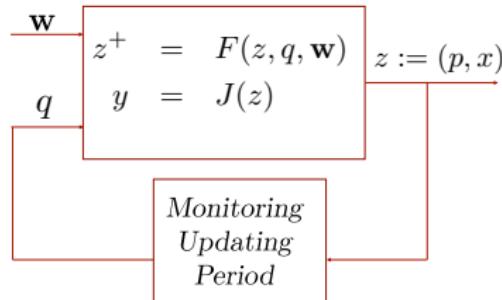


This results in the extended dynamic system:

$$\begin{aligned} x(t_{k+1}^u) &= X^r(q\tau, \textcolor{red}{x}(t_k^u), \textcolor{teal}{p}(t_k^u), w) \\ p(t_{k+1}^u) &= \mathcal{S}^q \left(p^+(t_k^u), \underbrace{X(q\tau, \textcolor{red}{x}(t_k^u), \textcolor{teal}{p}(t_k^u))}_{\hat{x}(t_{k+1}^u)} \right) \end{aligned}$$

The Control Updating Period Paradigm

Given \mathcal{S}



Problem Statement

Given a solver \mathcal{S} , propose a concrete (on-line) feedback:

$$q(t_k^u) = K(z(t_k^u), \dots)$$

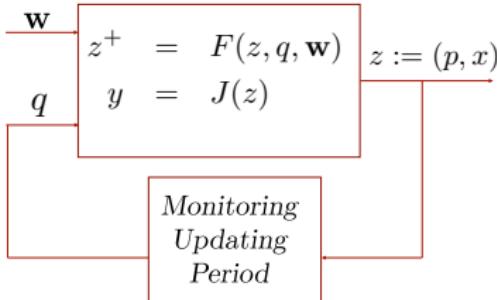
steering y to its minimum value.

This results in the extended dynamic system:

$$\begin{aligned} x(t_{k+1}^u) &= X^r(q\tau, \textcolor{red}{x}(t_k^u), \textcolor{teal}{p}(t_k^u), w) \\ p(t_{k+1}^u) &= \mathcal{S}^q \left(p^+(t_k^u), \underbrace{X(q\tau, \textcolor{red}{x}(t_k^u), \textcolor{teal}{p}(t_k^u))}_{\hat{x}(t_{k+1}^u)} \right) \end{aligned}$$

The Control Updating Period Paradigm

Given \mathcal{S}



Problem Statement

Given a solver \mathcal{S} , propose a concrete (on-line) feedback:

$$q(t_k^u) = K(z(t_k^u), \dots)$$

steering y to its minimum value.

NOTA The computation time needed for this feedback must be negligible.

This results in the extended dynamic system:

$$\begin{aligned} x(t_{k+1}^u) &= X^r(q\tau, \textcolor{red}{x}(t_k^u), \textcolor{teal}{p}(t_k^u), w) \\ p(t_{k+1}^u) &= \mathcal{S}^q \left(p^+(t_k^u), \underbrace{X(q\tau, \textcolor{red}{x}(t_k^u), \textcolor{teal}{p}(t_k^u))}_{\hat{x}(t_{k+1}^u)} \right) \end{aligned}$$

A Small Gain Paradigm

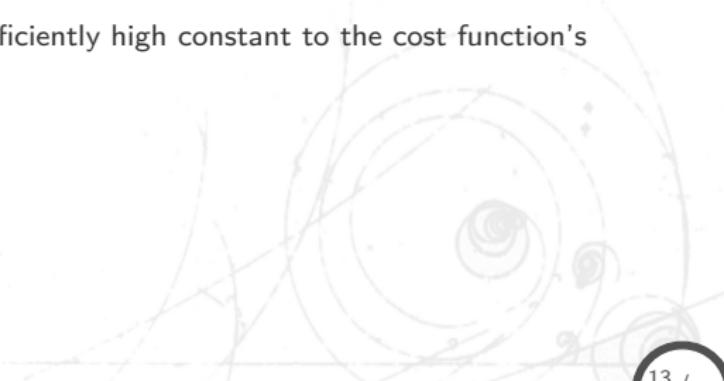
Preliminary Assumption

There is a positive $\underline{J} > 0$ such that for all (p, x) of interest, the inequality:

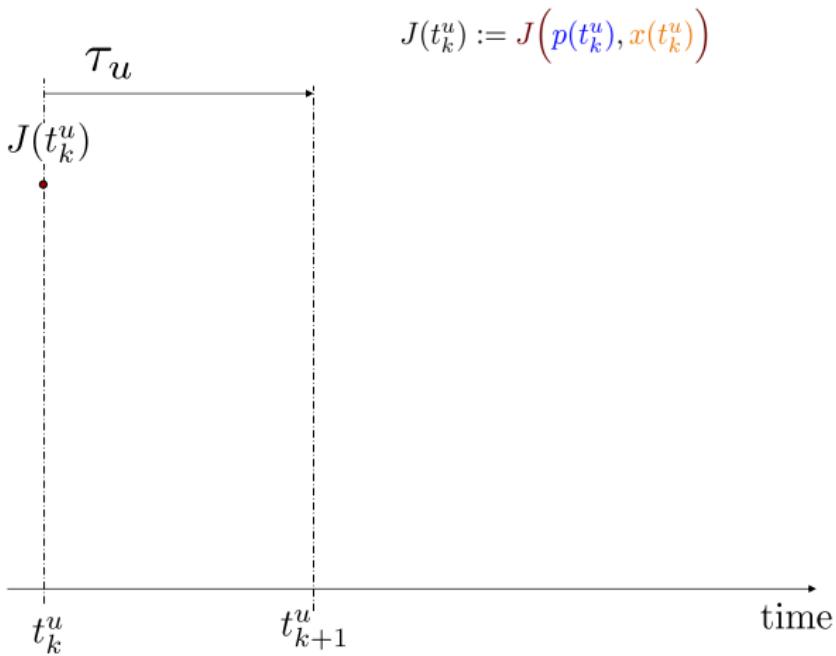
$$J(p, x) \geq \underline{J} > 0$$

is satisfied

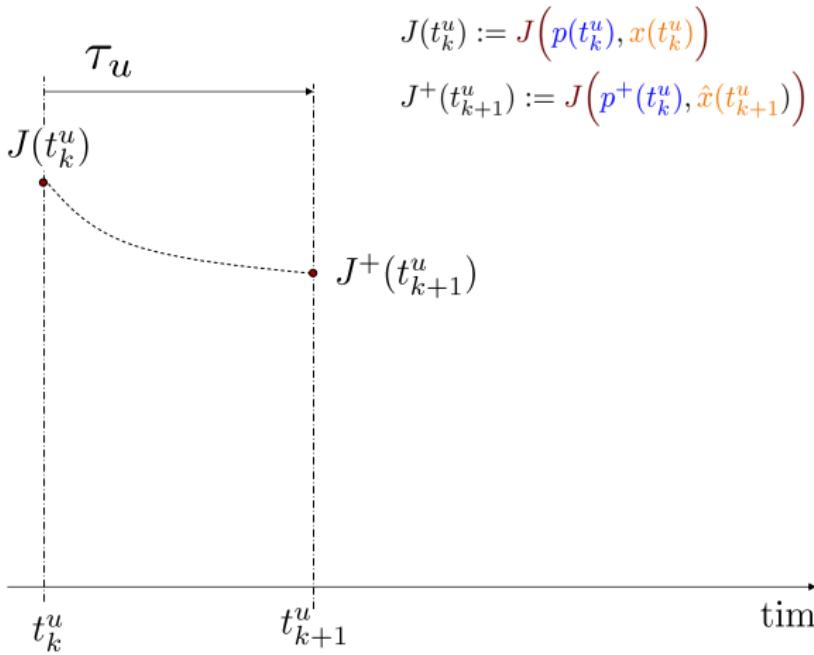
This can always be satisfied by adding sufficiently high constant to the cost function's definition.



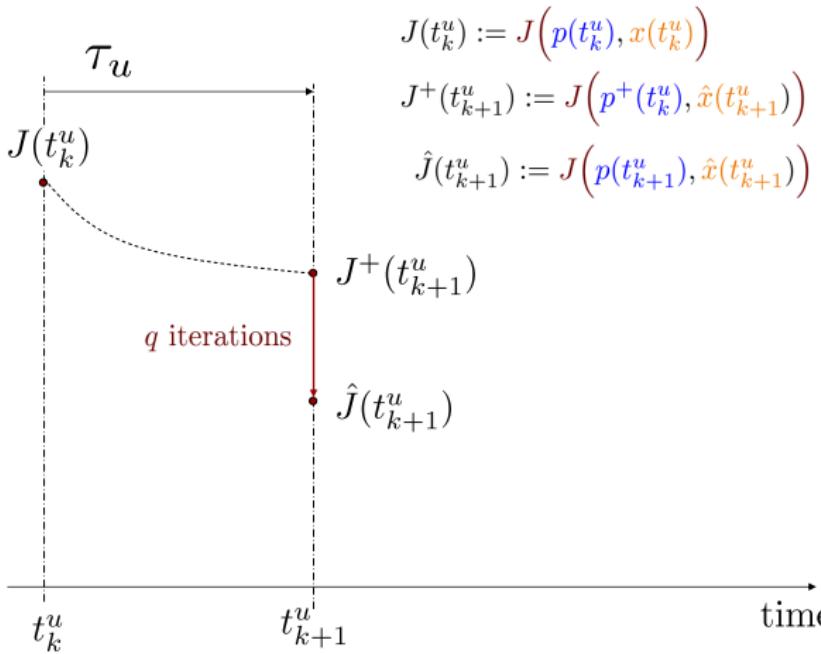
A Small Gain Paradigm



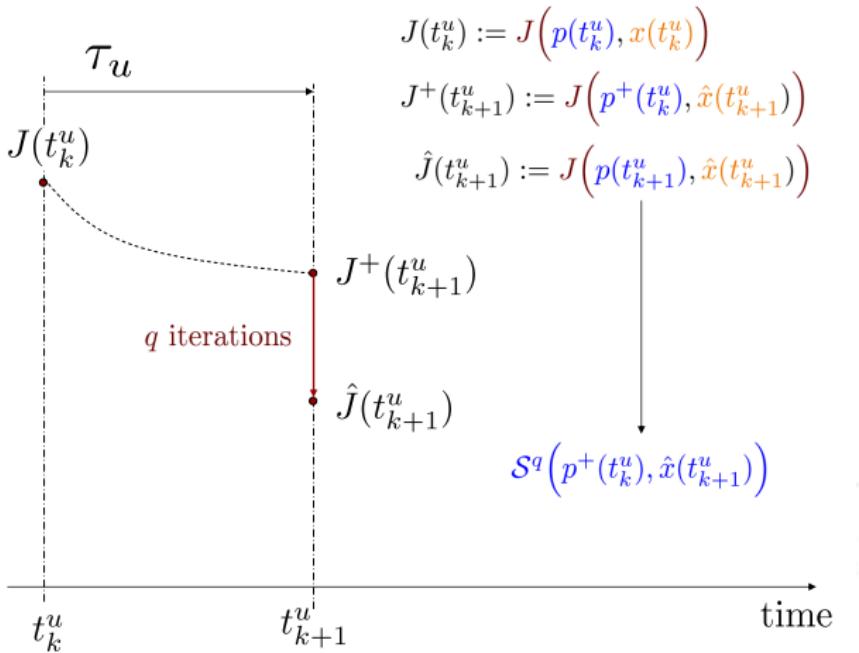
A Small Gain Paradigm



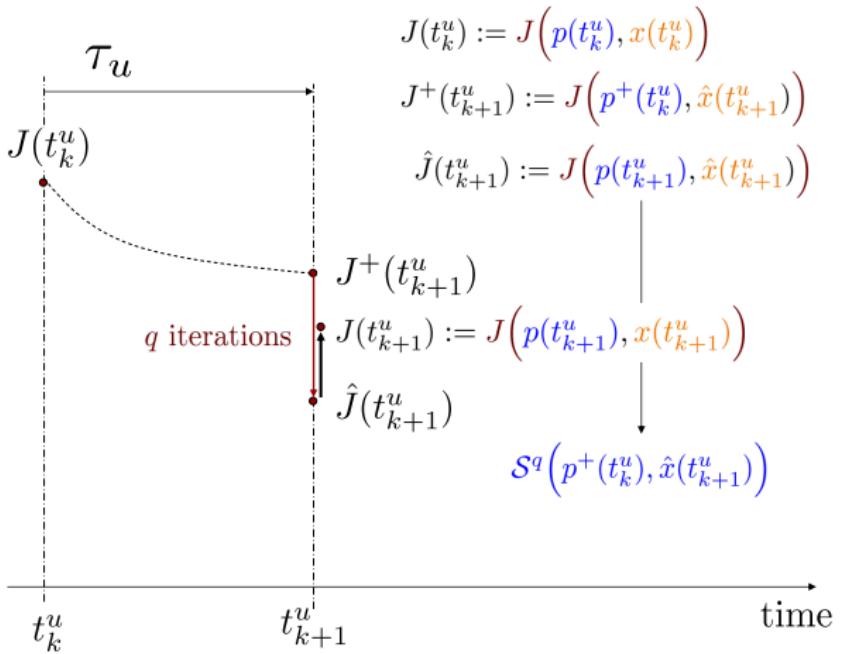
A Small Gain Paradigm



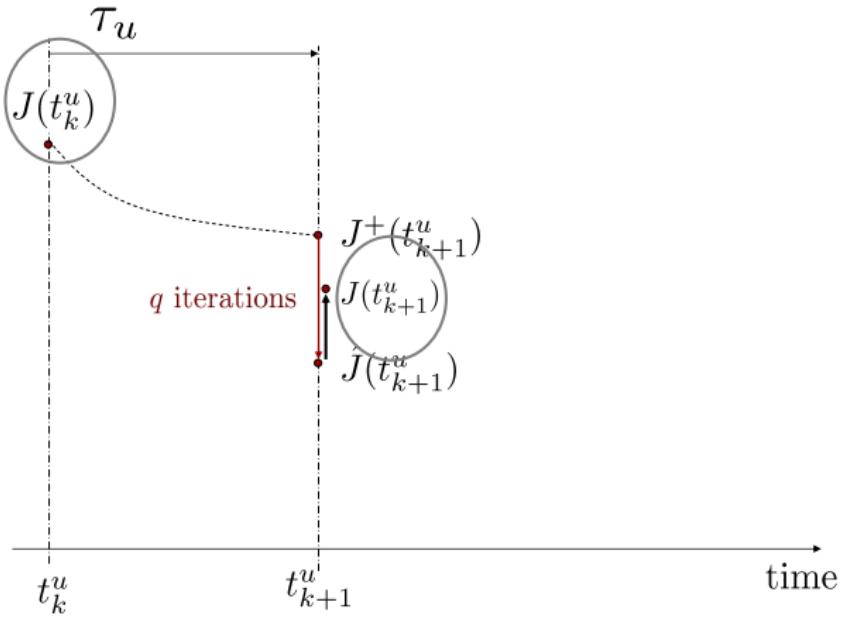
A Small Gain Paradigm



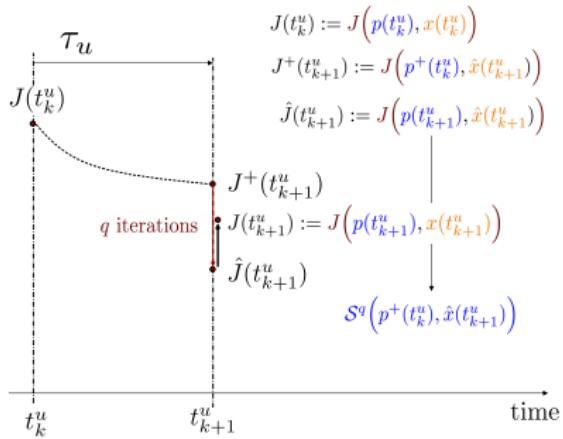
A Small Gain Paradigm



A Small Gain Paradigm

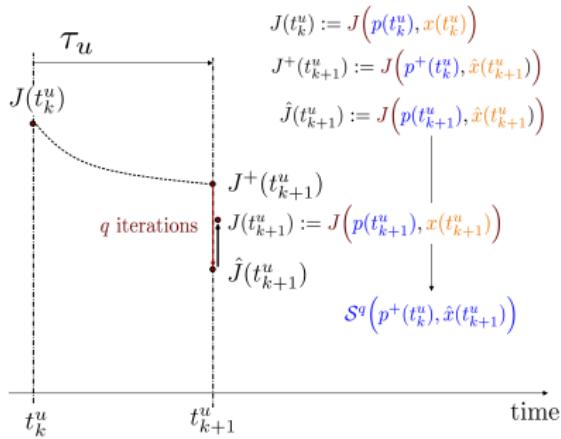


A Small Gain Paradigm



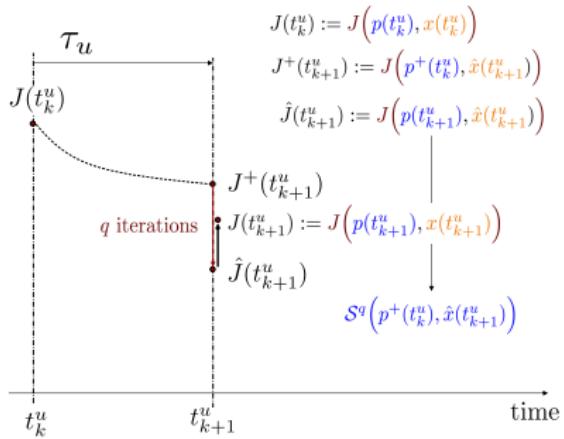
$$\frac{J(t_{k+1}^u)}{J(t_k^u)} = \left[\frac{J(t_{k+1}^u)}{\hat{J}(t_{k+1}^u)} \right] \times \left[\frac{\hat{J}(t_{k+1}^u)}{J^+(t_{k+1}^u)} \right] \times \left[\frac{J^+(t_{k+1}^u)}{J(t_k^u)} \right]$$

A Small Gain Paradigm



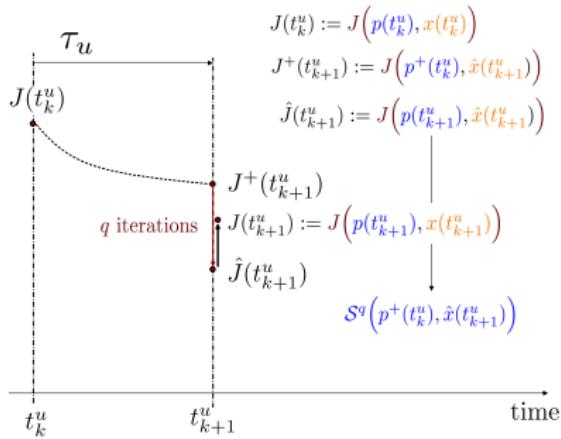
$$\frac{J(t_{k+1}^u)}{J(t_k^u)} = \underbrace{\left[\frac{J(t_{k+1}^u)}{\hat{J}(t_{k+1}^u)} \right]}_{E_k(q(t_k^u))} \times \left[\frac{\hat{J}(t_{k+1}^u)}{J^+(t_{k+1}^u)} \right] \times \left[\frac{J^+(t_{k+1}^u)}{J(t_k^u)} \right]$$

A Small Gain Paradigm



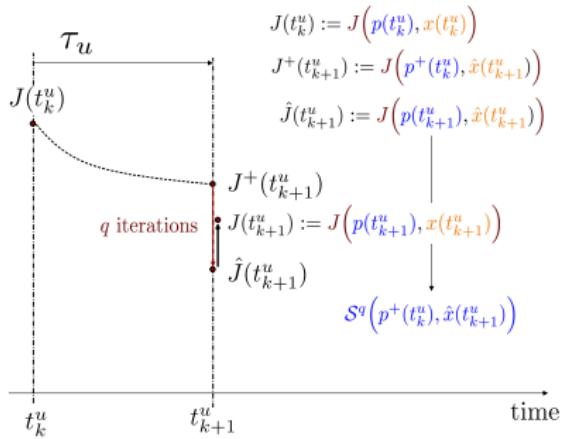
$$\frac{J(t_{k+1}^u)}{J(t_k^u)} = \underbrace{\left[\frac{J(t_{k+1}^u)}{\hat{J}(t_{k+1}^u)} \right]}_{\text{Uncertainty}} \times \underbrace{\left[\frac{\hat{J}(t_{k+1}^u)}{J^+(t_{k+1}^u)} \right]}_{E_k(q(t_k^u))} \times \left[\frac{J^+(t_{k+1}^u)}{J(t_k^u)} \right]$$

A Small Gain Paradigm



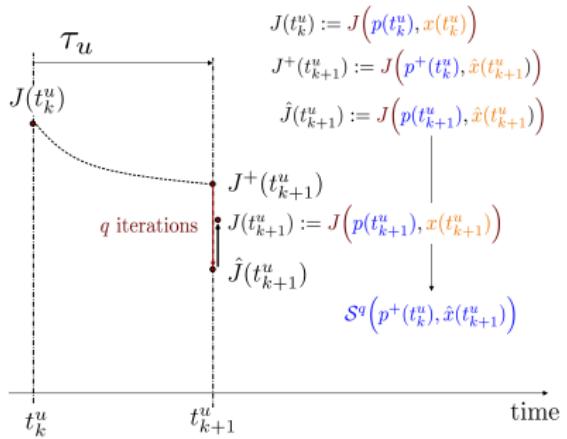
$$\frac{J(t_{k+1}^u)}{J(t_k^u)} = \underbrace{\left[\frac{J(t_{k+1}^u)}{\hat{J}(t_{k+1}^u)} \right]}_{\text{Uncertainty}} \times \underbrace{\left[\frac{\hat{J}(t_{k+1}^u)}{J^+(t_{k+1}^u)} \right]}_{E_k(q(t_k^u))} \times \underbrace{\left[\frac{J^+(t_{k+1}^u)}{J(t_k^u)} \right]}_{\text{Parametrization}}$$

A Small Gain Paradigm



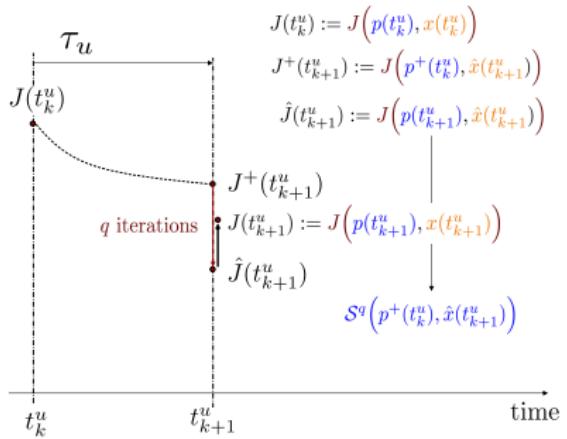
$$\frac{J(t_{k+1}^u)}{J(t_k^u)} = \underbrace{\left[\frac{\hat{J}(t_{k+1}^u)}{J^+(t_{k+1}^u)} \right]}_{E_k(q(t_k^u))} \times \underbrace{\left[\frac{J(t_{k+1}^u)}{\hat{J}(t_{k+1}^u)} \right]}_{\text{Uncertainty}} \times \underbrace{\left[\frac{J^+(t_{k+1}^u)}{J(t_k^u)} \right]}_{\text{Parametrization}}$$

A Small Gain Paradigm



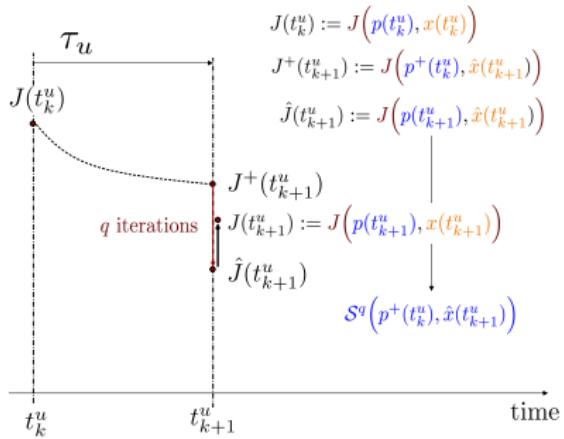
$$\frac{J(t_{k+1}^u)}{J(t_k^u)} = \underbrace{\left[\frac{\hat{J}(t_{k+1}^u)}{J^+(t_{k+1}^u)} \right]}_{E_k(q(t_k^u))} \times \underbrace{\left[\frac{J(t_{k+1}^u)}{\hat{J}(t_{k+1}^u)} \right]}_{D_k(q(t_k^u))} \times \left[\frac{J^+(t_{k+1}^u)}{J(t_k^u)} \right]$$

A Small Gain Paradigm



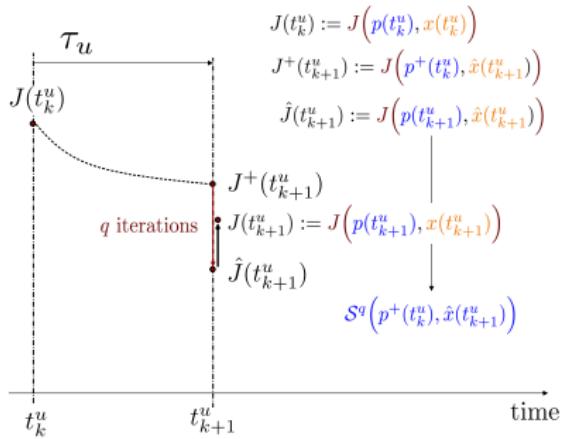
$$\frac{J(t_{k+1}^u)}{J(t_k^u)} = \underbrace{\left[\frac{\hat{J}(t_{k+1}^u)}{J^+(t_{k+1}^u)} \right]}_{E_k(q(t_k^u))} \times \underbrace{\left[\frac{J(t_{k+1}^u)}{\hat{J}(t_{k+1}^u)} \right]}_{D_k(q(t_k^u))} \times \left[\frac{J^+(t_{k+1}^u)}{J(t_k^u)} \right]$$

A Small Gain Paradigm



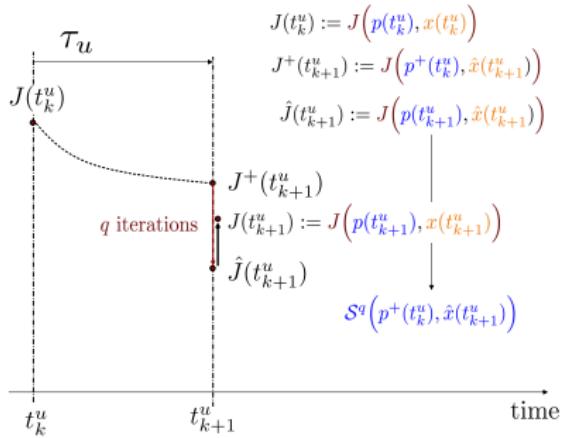
$$J(t_{k+1}^u) = [E_k(q(t_k^u))] \cdot [D_k(q(t_k^u))] \cdot J(t_k^u)$$

A Small Gain Paradigm



$$\begin{aligned} J(t_{k+1}^u) &= [E_k(q(t_k^u))] \cdot [D_k(q(t_k^u))] \cdot J(t_k^u) \\ &= [K_k(q(t_k^u))] \cdot J(t_k^u) \end{aligned}$$

A Small Gain Paradigm

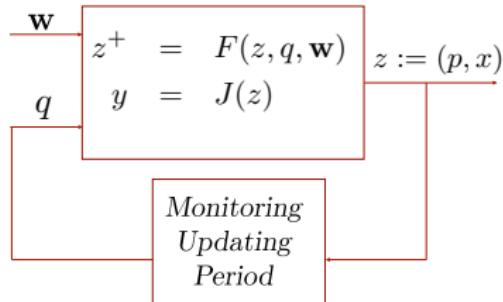


$$\begin{aligned}
 J(t_{k+1}^u) &= \left[E_k(q(t_k^u)) \right] \cdot \left[D_k(q(t_k^u)) \right] \cdot J(t_k^u) \\
 &= \left[K_k(q(t_k^u)) \right] \cdot J(t_k^u)
 \end{aligned}$$

NOTA: If $K_k < 1$, the settling time is $t_r(q(t_k^u)) \sim \frac{q(t_k^u)}{|\log(K_k(q(t_k^u)))|}$

The Ideal Updating Scheme Principle

Given \mathcal{S}



Problem Statement

Given a solver \mathcal{S} , propose a concrete (on-line) feedback:

$$q(t_k^u) = K(z(t_k^u), \dots)$$

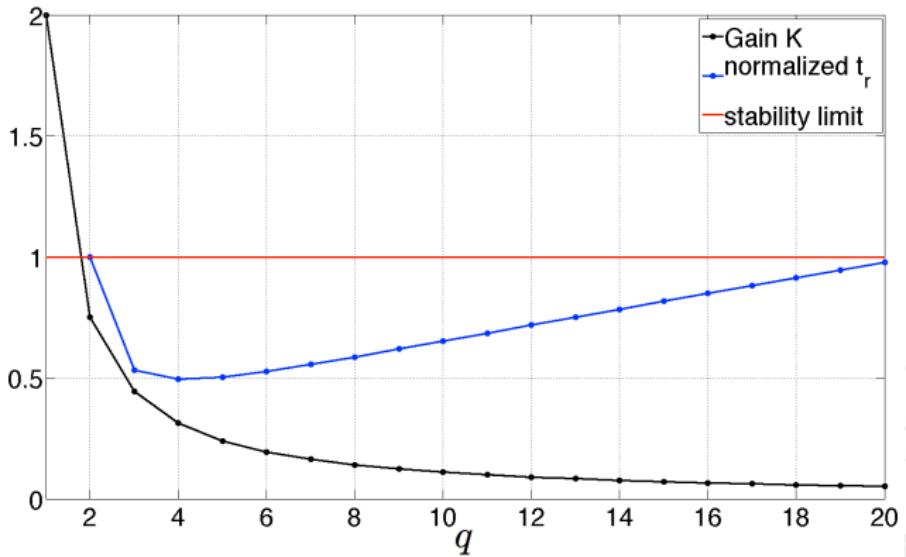
steering y to its minimum value.

$$q(t_{k+1}^u) := \begin{cases} \arg \min_{q \in \{1, q_{max}\}} \frac{q}{|\log(K_k(q))|} & \text{if } K_k(q(t_k^u)) < 1 \\ \arg \min_{q \in \{1, q_{max}\}} K_k(q) & \text{otherwise} \end{cases}$$

A one-step scalar predictive control . . . !!

Numerical Investigation Small gain paradigm illustration

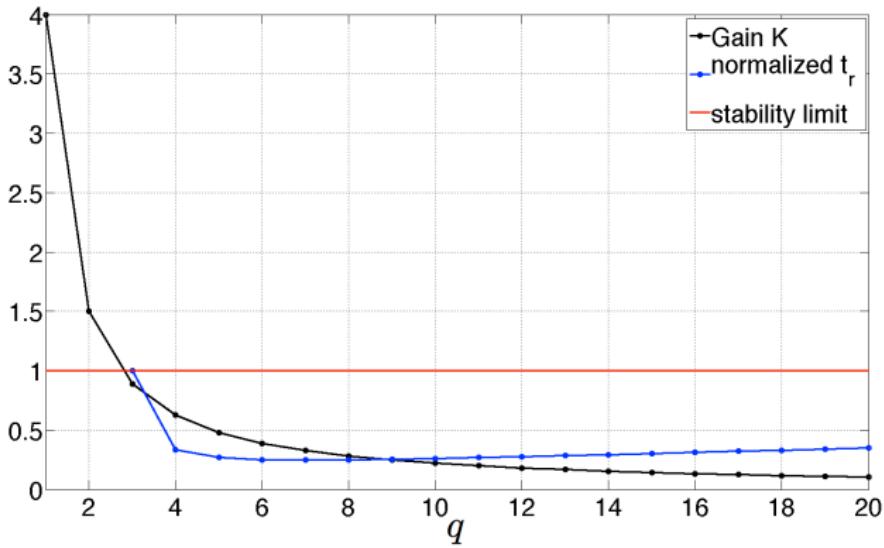
$$E(q) := \frac{1}{q^2} \quad ; \quad D(q) = (1 + 1 \cdot q)$$



⇒ Contraction needs $q \geq 2$, optimal $q = 4$

Numerical Investigation Small gain paradigm illustration

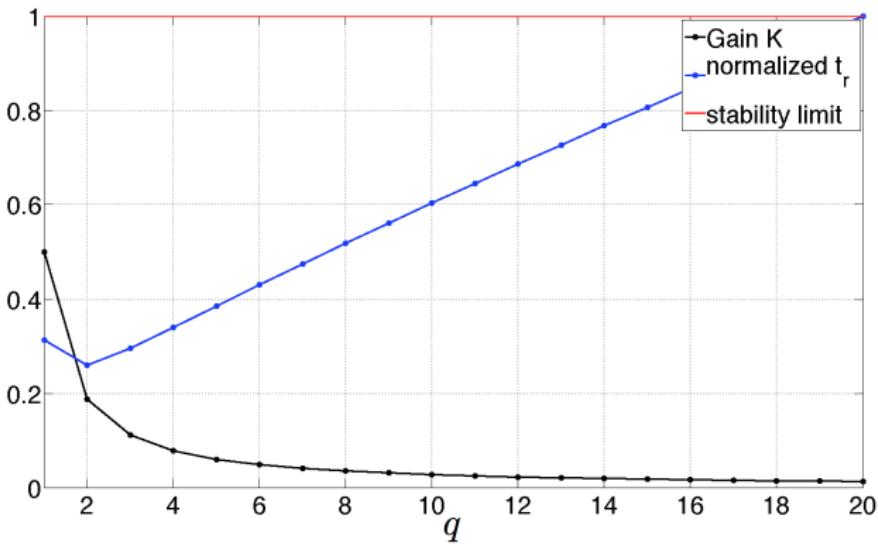
$$E(q) := \frac{2}{q^2} \quad ; \quad D(q) = (1 + 1 \cdot q)$$



⇒ Contraction needs $q \geq 3$, optimal $q = 6$

Numerical Investigation Small gain paradigm illustration

$$E(q) := \frac{0.25}{q^2} \quad ; \quad D(q) = (1 + 1 \cdot q)$$



⇒ Contraction needs $q \geq 1$, optimal $q = 2$

Outline

- Fast NMPC
- The Control Updating Period Paradigm
- **The Proposed Updating Scheme**
- Fast Gradient Related Considerations
- Numerical Investigation
- Conclusion & future works

The Ideal Updating Scheme

Discussion ...

$$q(t_{k+1}^u) := \begin{cases} \arg \min_{q \in \{1, q_{max}\}} \frac{q}{|\log(K_k(q))|} & \text{if } K_k(q(t_k^u)) < 1 \\ \arg \min_{q \in \{1, q_{max}\}} K_k(q) & \text{otherwise} \end{cases}$$

Recall that:

$$K_k(q) := E_k(q) \times D_k(q)$$

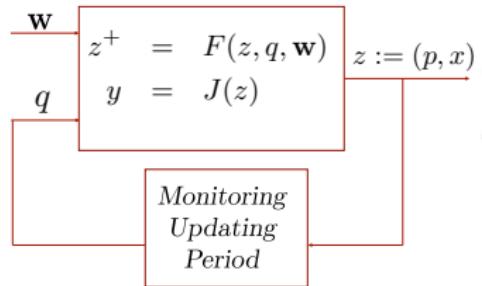
- $E_k(q)$ is badly known
- On-line identification based on a parametric structure:

$$E_k(q) = \mathcal{E}(q, p_E(k))$$

[Alamir, Springer, 2008]

- Critical when $q(t_k^u)$ is close to 1.

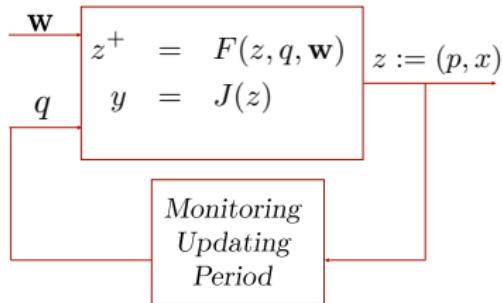
Given \mathcal{S}



The Ideal Updating Scheme

Idea ...

Given \mathcal{S}



Problem Statement

Given a solver \mathcal{S} , propose a concrete (on-line) feedback:

$$q(t_k^u) = K(z(t_k^u), \dots)$$

stabilizing $y = 0$.

$$q(t_{k+1}^u) := \arg \min_{q \in \{1, q_{max}\}} \begin{cases} \frac{q}{|\log(K_k(q))|} & \text{if } K_k(q(t_k^u)) < 1 \\ K_k(q) & \text{otherwise} \end{cases}$$

A one-step scalar predictive control ... !!

→ Distribute the optimization over time ... AGAIN !

Updating Algorithm

$$q(t_{k+1}^u) := \arg \min_{q \in \{1, q_{max}\}} \begin{cases} \frac{q}{|\log(K_k(q))|} & \text{if } K_k(q(t_k^u)) < 1 \\ K_k(q) & \text{otherwise} \end{cases}$$

Algorithm 1 Updating rule $q(t_{k+1}^u) = U(q(t_k^u), t_k^u)$

1: **If** ($K_k \geq 1$) **then**

2: $\Gamma \leftarrow \frac{\Delta K_k}{\Delta q}(q(t_k^u))$

3: **Else**

4: $\Gamma \leftarrow \frac{\Delta(q/|\log(K_k(q))|)}{\Delta q} = \frac{-\log(K_k) + \frac{q}{K_k} \times \frac{\Delta K_k}{\Delta q}(q(t_k^u))}{[\log(K_k)]^2}$

5: **End If**

6: $q(t_{k+1}^u) \leftarrow \max\left\{2, \min\left\{q_{max}, \color{red}q(t_k^u) - \delta \cdot sign(\Gamma)\right\}\right\}$

Updating Algorithm

$$q(t_{k+1}^u) := \arg \min_{q \in \{1, q_{max}\}} \begin{cases} \frac{q}{|\log(K_k(q))|} & \text{if } K_k(q(t_k^u)) < 1 \\ K_k(q) & \text{otherwise} \end{cases}$$

Algorithm 4 Updating rule $q(t_{k+1}^u) = U(q(t_k^u), t_k^u)$

1: **If** ($K_k \geq 1$) **then**

2: $\Gamma \leftarrow \frac{\Delta K_k}{\Delta q}(q(t_k^u))$

3: **Else**

4: $\Gamma \leftarrow \frac{\Delta(q/|\log(K_k(q))|)}{\Delta q} = \frac{-\log(K_k) + \frac{q}{K_k} \times \frac{\Delta K_k}{\Delta q}(q(t_k^u))}{[\log(K_k)]^2}$

5: **End If**

6: $q(t_{k+1}^u) \leftarrow \max\left\{2, \min\left\{q_{max}, \color{red}{q(t_k^u) - \delta \cdot sign(\Gamma)}$

Need to compute $K_k(q(t_k^u))$ and $\frac{\Delta K_k}{\Delta q}(q(t_k^u))$

Sensitivity Computation

E_k -related quantities

Recall that:

$$E_k(q) := \frac{J(p^{(q)}, \hat{x}(t_{k+1}^u))}{J(p^{(0)}, \hat{x}(t_{k+1}^u))} \quad ; \quad p^{(0)} := p^+(t_k^u)$$

therefore:

$$\frac{\Delta E_k}{\Delta q}(q(t_k^u)) \approx \frac{J(p^{(q(t_k^u))}, \hat{x}(t_{k+1}^u)) - J(p^{(q(t_k^u)-1)}, \hat{x}(t_{k+1}^u))}{J(p^{(0)}, \hat{x}(t_{k+1}^u))}$$

→ q need to be greater than 2.

→ $E_k(q(t_k^u))$ and $\frac{\Delta E_k}{\Delta q}(q(t_k^u))$ are computable from available computer data at the cost of **2 divisions and 1 addition**

Sensitivity Computation

D_k -related quantities

Consider the simple uncertainty model:

$$D_k(q) = 1 + [\alpha_D] \cdot q$$

recall that one has by definition:

$$D_k(q(t_k^u)) := \frac{J_{k+1} \times J_k^+}{J_k \times \hat{J}_{k+1}} = \frac{1}{E_k} \frac{J_{k+1}}{J_k}$$

$$\Rightarrow \alpha_D := \frac{\Delta D_k}{\Delta q}(q(t_k^u)) \approx \frac{1}{q(t_k^u)} \times \left[\frac{1}{E_k} \frac{J_{k+1}}{J_k} - 1 \right]$$

$\rightarrow D_k(q(t_k^u))$ and $\frac{\Delta D_k}{\Delta q}(q(t_k^u))$ are computable from available computer data at the cost of **2 divisions and 1 multiplication**

$$\rightarrow \frac{\Delta K_k}{\Delta q} = E_k \frac{\Delta D_k}{\Delta q} + D_k \frac{\Delta E_k}{\Delta q}$$

Convergence

Convergence depends on the issue of a **COMPETITION** between:

1. A **DECREASE** in $t_r(q)$ due to the quantized gradient descent
2. A *potential* **INCREASE** due to change between t_k^u and t_{k+1}^u .

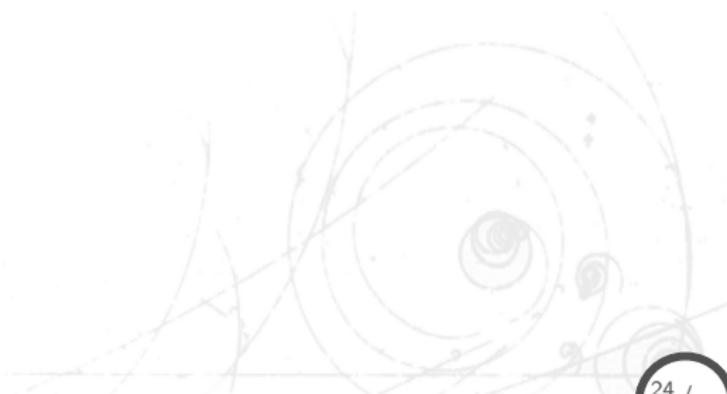
Does $t_r(q)$ show a unique local minimum on $[2, q_{max}]$?

This is more likely to be true if $E_k(\cdot)$ is strictly monotonically decreasing.

What about $E_k(\cdot)$ in the case of fast gradient . . . ?

Outline

- Fast NMPC
- The Control Updating Period Paradigm
- The Proposed Updating Scheme
- **Fast Gradient Related Considerations**
- Numerical Investigation
- Conclusion & future works



Algorithm 2 Fast gradient iterations $\bar{p} = \mathcal{S}^{(q)}(p, x)$

- 1: Initialization. $p^{(0)} \leftarrow p$, $r \leftarrow p$
- 2: **for** $i = 1, q$ **do**
- 3: $p^{(i)} \leftarrow P_C\left(r - \frac{1}{L}[\nabla J(r, x)]\right)$
- 4: $r \leftarrow p^{(i)} + c(p^{(i)} - p^{(i-1)})$
- 5: **end for**
- 6: $\bar{p} \leftarrow p^{(q)}$

- $\forall(x, p_1, p_2)$, $\|\nabla J(p_2, x) - \nabla J(p_1, x)\| \leq L\|p_2 - p_1\|$
- $c \in [0, 1]$:
 - $c = 0 \rightarrow$ pure (slow) gradient descent
 - For quadratic problems $c = \frac{\sqrt{\lambda_{max}(H)} - \sqrt{\lambda_{min}(H)}}{\sqrt{\lambda_{max}(H)} + \sqrt{\lambda_{min}(H)}}$
- $P_C(\cdot)$: projection on the admissible set.

Algorithm 2 Fast gradient iterations $\bar{p} = \mathcal{S}^{(q)}(p, x)$

- 1: Initialization. $p^{(0)} \leftarrow p$, $r \leftarrow p$
- 2: **for** $i = 1, q$ **do**
- 3: $p^{(i)} \leftarrow P_C\left(r - \frac{1}{L}[\nabla J(r, x)]\right)$
- 4: $r \leftarrow p^{(i)} + c(p^{(i)} - p^{(i-1)})$
- 5: **end for**
- 6: $\bar{p} \leftarrow p^{(q)}$

- $J(p^{(k)}, x) - J^*(x) \leq -\frac{\alpha(\|x - x^*\|)}{(k+2)^2}$ (optimal c)
- $r \leftarrow P_C(r - (\nabla J)/L) + \dots$ [r is somehow an integrator state]
- Induced oscillations for high values of $c \in [0, 1]$.
 - B. O'Donoghue and A. Candes. Adaptive restart for accelerated gradient schemes. arxiv:1204.3982. April 2012.
- **Efficiency map $E_k(\cdot)$ of original Fast Gradient is not monotonic**

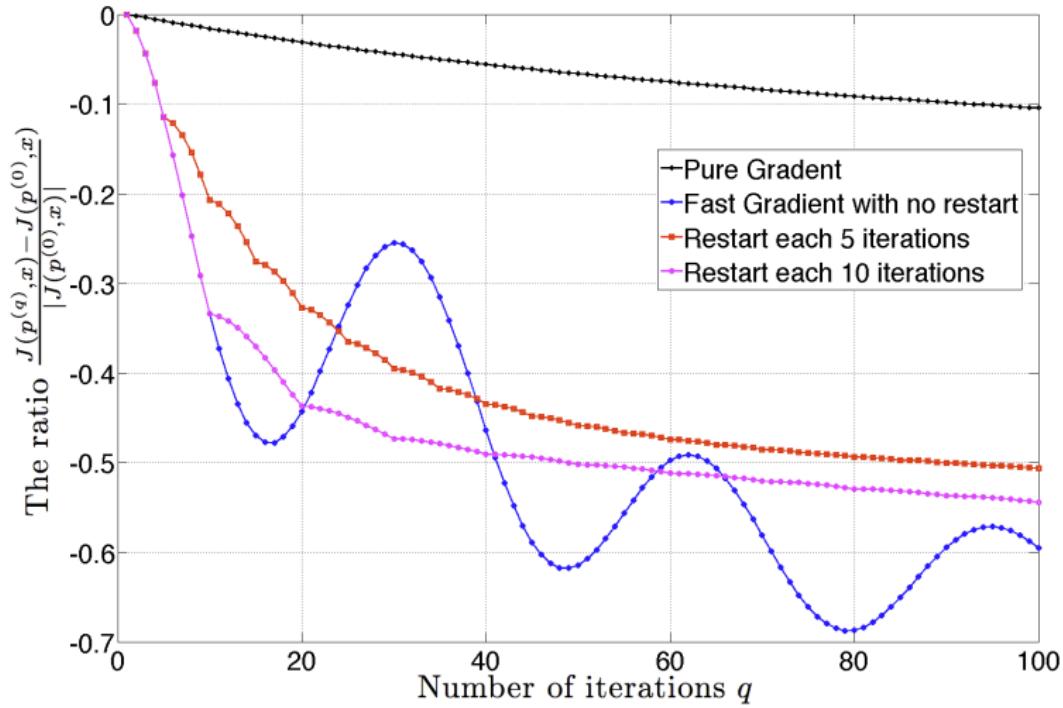
Algorithm 3 Fast gradient iterations $\bar{p} = \mathcal{S}^{(q)}(p, x)$ with constant restarting strategy

- 1: Initialization. $p^{(0)} \leftarrow p$, $r \leftarrow p$, $s \leftarrow 0$
- 2: **for** $i = 1, q$ **do**
- 3: $s \leftarrow s + 1$
- 4: $p^{(i)} \leftarrow P_C\left(r - \frac{1}{L}[\nabla J(r, x)]\right)$
- 5: $r \leftarrow p^{(i)} + c(p^{(i)} - p^{(i-1)})$
- 6: **if** $(s = s_{max})$ **then** $r \leftarrow p^{(i)}$, $s = 0$ **End if**
- 7: **end for**
- 8: $\bar{p} \leftarrow p^{(q)}$

- $J(p^{(k)}, x) - J^*(x) \leq -\frac{\alpha(\|x - x^*\|)}{(k+2)^2}$ (optimal c)
- $r \leftarrow P_C(r - (\nabla J)/L) + \dots$ [r is somehow an integrator state]
- Induced oscillations for high values of $c \in [0, 1]$.
 - B. O'Donoghue and A. Candes. Adaptive restart for accelerated gradient schemes. arxiv:1204.3982. April 2012.
- **Efficiency map $E_k(\cdot)$ of original Fast Gradient is not monotonic**

Fast Gradient with Restart Mechanism

Typical behavior of successive cost function values under pure gradient and fast gradient descent method without and with restarting mechanism



Outline

- Fast NMPC
- The Control Updating Period Paradigm
- The Proposed Updating Scheme
- Fast Gradient Related Considerations
- **Numerical Investigation**
- Conclusion & future works

Illustrative example MPC-Controlled system with adaptive control updating period

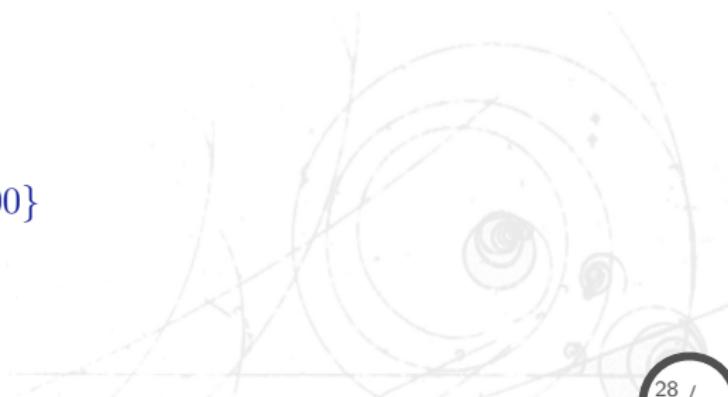
Consider the following triple integrator system

$$\dot{x}_1 = x_2 ; \quad \dot{x}_2 = x_3 ; \quad \dot{x}_3 = u \quad ; \quad |u| \leq 1$$

Consider MPC scheme based on the following cost function:

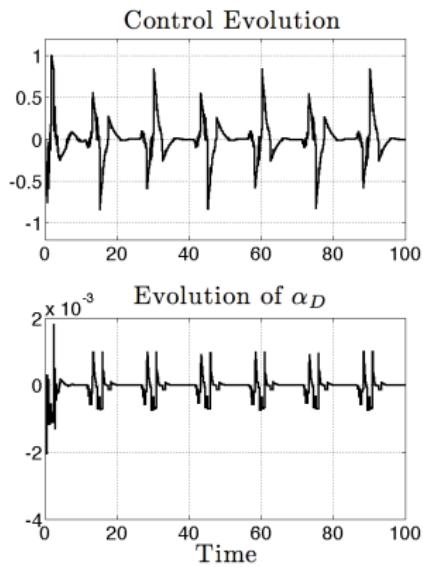
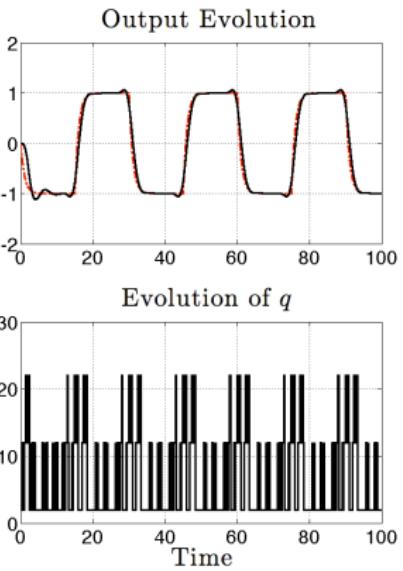
$$J = \sum_{k=1}^N \|y(k) - y^{ref}(k)\|_Q^2 + \|u(k)\|_R^2$$

- Standard p.w.c parametrization
- $\tau = 0.02$, $Q = 100$, $R = 1$
- Restarting parameter $s_{max} = 8$
- Gradient step for q , $\delta = 10$
- Prediction horizon $N \in \{100, 200\}$
- $q_{max} = 100$



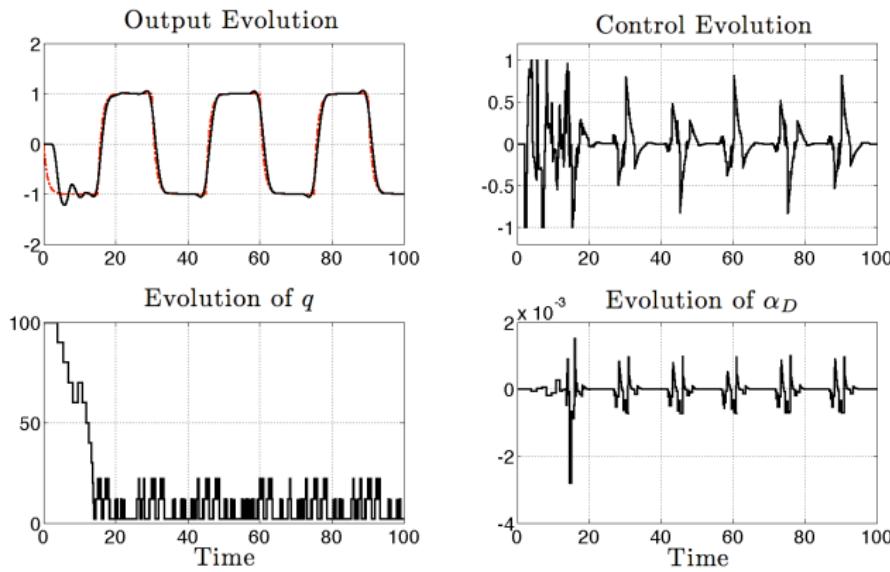
Illustrative example MPC-Controlled system with adaptive control updating period

- Adapt. ON
- $q(0) = 2$
- $N = 200$
- $\delta = 10$



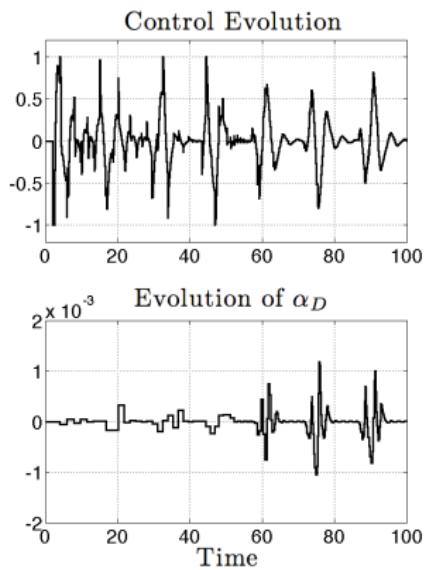
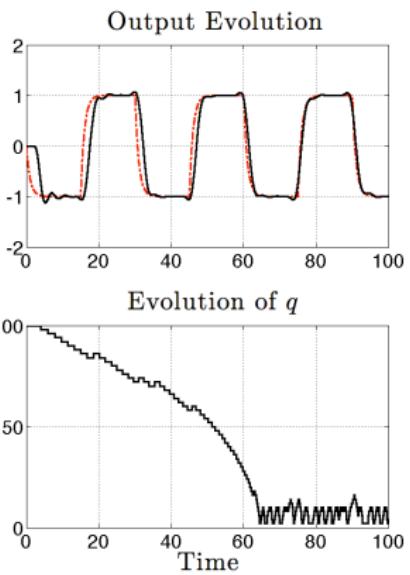
Illustrative example MPC-Controlled system with adaptive control updating period

- Adapt. ON
- $q(0) = 100$
- $N = 200$
- $\delta = 10$



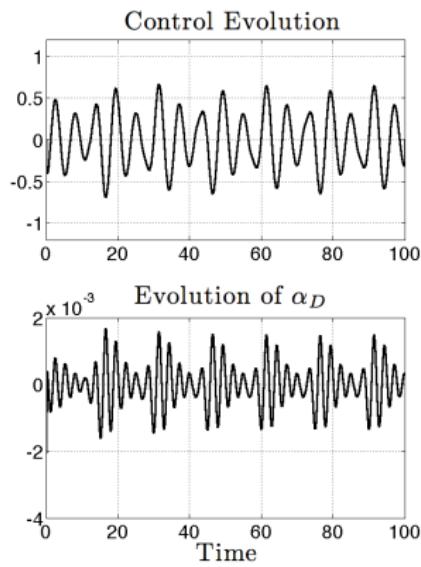
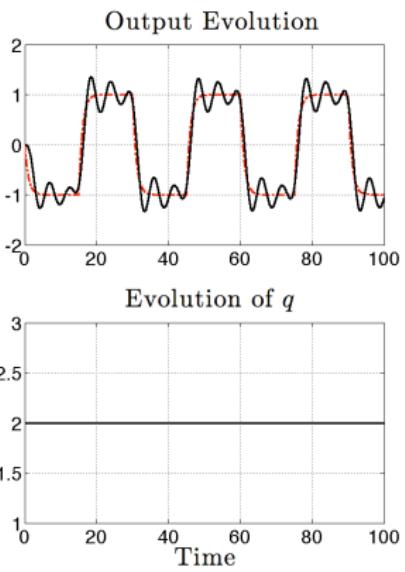
Illustrative example MPC-Controlled system with adaptive control updating period

- Adapt. ON
- $q(0) = 2$
- $N = 200$
- $\delta = 2$



Illustrative example MPC-Controlled system with adaptive control updating period

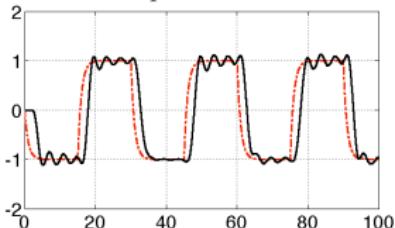
- Adapt. OFF
- $q \equiv 2$
- $N = 200$
- $\delta = 10$



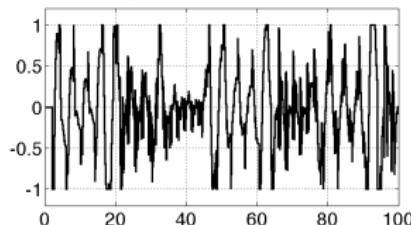
Illustrative example MPC-Controlled system with adaptive control updating period

- Adapt. OFF
- $q \equiv 100$
- $N = 200$
- $\delta = 10$

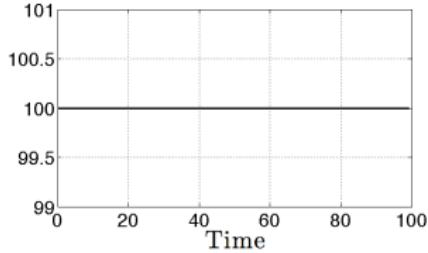
Output Evolution



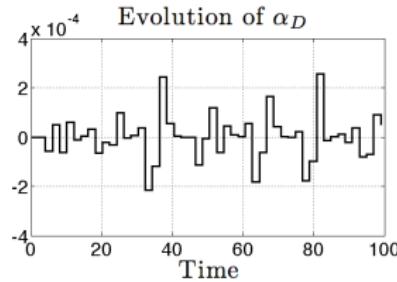
Control Evolution



Evolution of q

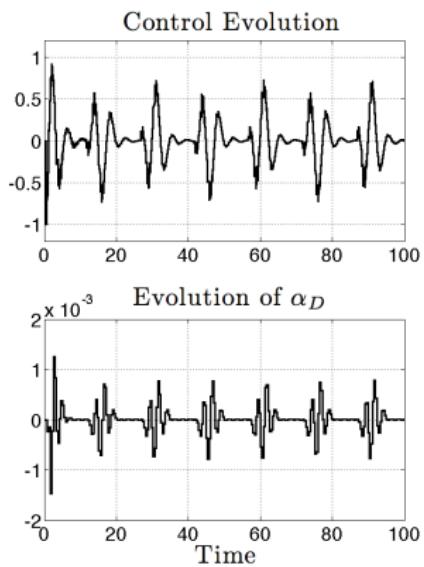
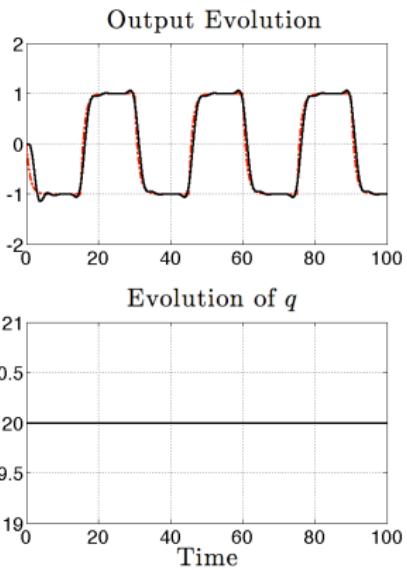


Evolution of α_D



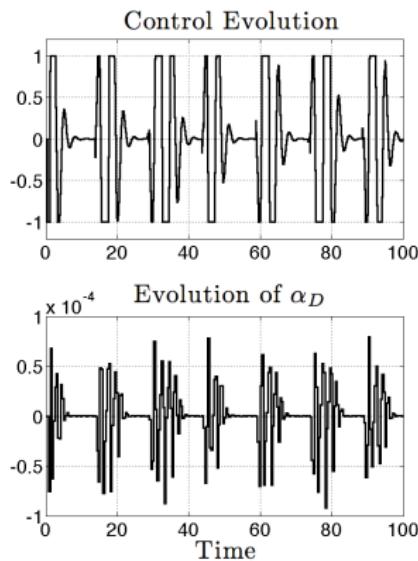
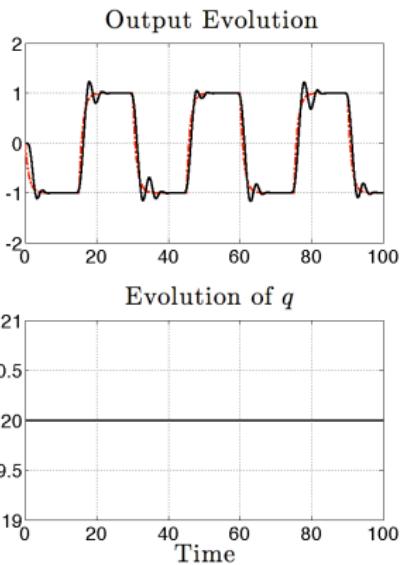
Illustrative example MPC-Controlled system with adaptive control updating period

- Adapt. OFF
- $q \equiv 20$
- $N = 200$
- $\delta = 10$



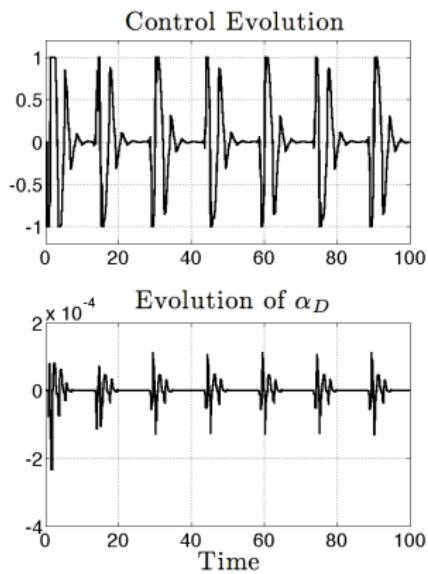
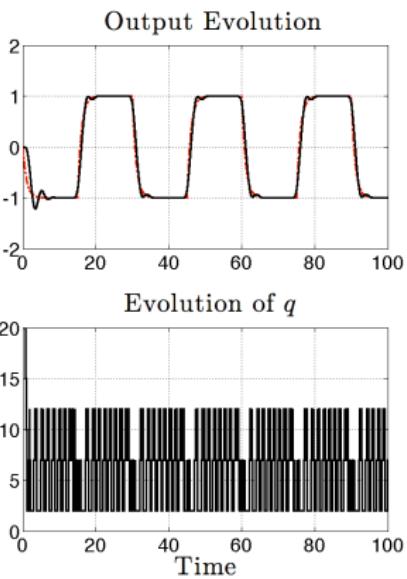
Illustrative example MPC-Controlled system with adaptive control updating period

- Adapt. OFF
- $q \equiv 20$
- $N = 100$
- $\delta = 10$



Illustrative example MPC-Controlled system with adaptive control updating period

- Adapt. ON
- $q(0) = 20$
- $N = 100$
- $\delta = 10$



Conclusion & Future Work

- Simple and computationally cheap solution for adapting control updating rate in distributed-in-time NMPC.
- General adaptation layer that can be added to your favorite descent method.
- Two tuning parameters (δ and s_{max})
- Avoid worst case choice of updating period [Allow less control CPU use].

- Dual formulation for Moving-Horizon Observers
- Potential use in a task management under computational resource sharing context.
- Interaction with event-based control and sensing paradigm.

For more details

<http://arxiv.1209.4922>