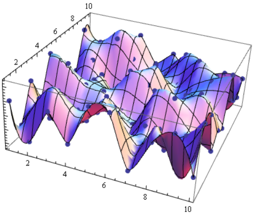
Direct data-driven control of constrained linear systems

Simone Formentin

Dept. of Electronics, Information and Bioengineering, Politecnico di Milano, Italy



ERNSI Workshop @Lyon, France, September 25, 2017

Joint work with R. Tóth, D. Piga, A. Bemporad and S.M. Savaresi

Outline

- Problem formulation
- Direct design of LPV controllers from data
- Constraint management and performance boost
- Concluding remarks

Outline

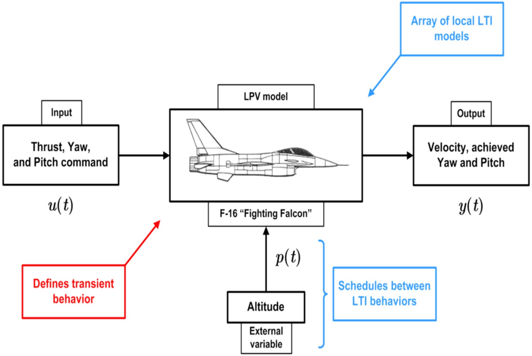
- Problem formulation
- Direct design of LPV controllers from data
- Constraint management and performance boost
- Concluding remarks

Introduction

- Many nonlinear and time-varying plants can be described by LPV models

Introduction

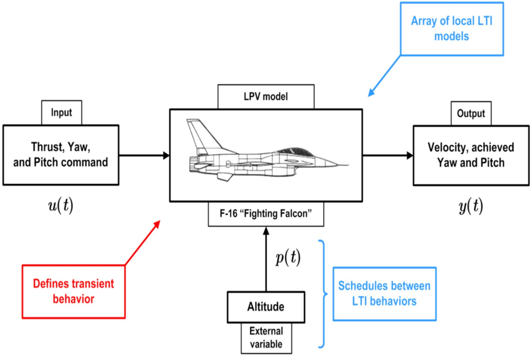
- Many nonlinear and time-varying plants can be described by LPV models



- The dynamic relationship between $u(t)$ and $y(t)$
  - ✓ is **linear**
  - ✓ depends on a **measurable** signal, the so-called scheduling variable $p(t)$

Introduction

- Many nonlinear and time-varying plants can be described by LPV models



- The dynamic relationship between $u(t)$ and $y(t)$
  - ✓ is **linear**
  - ✓ depends on a **measurable** signal, the so-called scheduling variable $p(t)$
- **Linear** robust and gain-scheduled control solution can be employed for effective control of complex nonlinear systems!

State of the art

- Standard approach: system identification + model-based control

State of the art

- Standard approach: system identification + model-based control

- Identification (global approaches)
  - ✓ **State-space (SS) models**: (Nemani et al, 1995), (Lee and Poolla, 1997 & 1999) (Lovera et al., 1998), (Sznaier and Mazzaro, 2001 & 2003), (Felici et al., 2007), (van Wingerden and Verhaegen, 2008), (Rizvi et al., 2017)
  - ✓ **Input/output (IO) models**: (Bamieh and Giarré, 1999 & 2002), (Previdi and Lovera, 2003 & 2004), (Toth et al., 2007 & 2008), (Piga et al., 2015)

State of the art

- Standard approach: system identification + model-based control

- Identification (global approaches)
  - ✓ **State-space (SS) models**: (Nemani et al, 1995), (Lee and Poolla, 1997 & 1999) (Lovera et al., 1998), (Sznaier and Mazzaro, 2001 & 2003), (Felici et al., 2007), (van Wingerden and Verhaegen, 2008), (Rizvi et al., 2017)
  - ✓ **Input/output (IO) models**: (Bamieh and Giarré, 1999 & 2002), (Previdi and Lovera, 2003 & 2004), (Toth et al., 2007 & 2008), (Piga et al., 2015)

- Model-based control
  - ✓ **SS models**: many approaches since J. Shamma's thesis in 1988
  - ✓ **IO models**: some recent studies, e.g., (Ali et al., 2010), (Cerone et al., 2012), (Wollnack et al., 2013), (Abbas et al., 2016 & 2017)

Problem Statement

- What to do?
  - ✓ **IO identification** is often favored as it represents an extension of the classical LTI PEM framework and its capability to handle high order systems

Problem Statement

- What to do?
  - ✓ **IO identification** is often favored as it represents an extension of the classical LTI PEM framework and its capability to handle high order systems
- Problems:
  - ✓ Exact SS realizations of the identified IO models (aimed to **SS control**) often introduce undesired complexity in the scheduling dependencies, which may prevent controller synthesis or its hardware implementation
  - ✓ **IO control** design is mainly based on gradient-based BMI solvers and generally suffers from high computational cost
  - ✓ Assessment of performance is often done in continuous-time but **discretization** of LPV models is still an open problem (Toth et al., 2008)

Problem Statement

- What to do?
  - ✓ **IO identification** is often favored as it represents an extension of the classical LTI PEM framework and its capability to handle high order systems
- Problems:
  - ✓ Exact SS realizations of the identified IO models (aimed to **SS control**) often introduce undesired complexity in the scheduling dependencies, which may prevent controller synthesis or its hardware implementation
  - ✓ **IO control** design is mainly based on gradient-based BMI solvers and generally suffers from high computational cost
  - ✓ Assessment of performance is often done in continuous-time but **discretization** of LPV models is still an open problem (Toth et al., 2008)
  - ✓ The way the **modeling errors** affect the control performance is unknown for most design methods
  - ✓ Little work has been done on including information about the control objectives into the identification setting

Problem Statement

- What to do?
  - ✓ **IO identification** is often favored as it represents an extension of the classical LTI PEM framework and its capability to handle high order systems
- Problems:
  - ✓ Exact SS realizations of the identified IO models (aimed to **SS control**) often introduce undesired complexity in the scheduling dependencies, which may prevent controller synthesis or its hardware implementation
  - ✓ **IO control** design is mainly based on gradient-based BMI solvers and generally suffers from high computational cost
  - ✓ Assessment of performance is often done in continuous-time but **discretization** of LPV models is still an open problem (Toth et al., 2008)
  - ✓ The way the **modeling errors** affect the control performance is unknown for most design methods
  - ✓ Little work has been done on including information about the control objectives into the identification setting
- **Idea: direct identification of the discrete-time controller in its IO representation** ("from data to implementation")

Outline

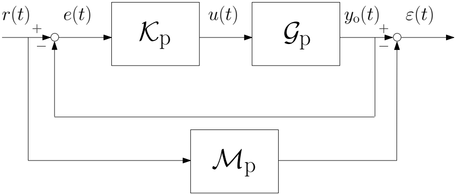
- Problem formulation
- Direct design of LPV controllers from data
- Constraint management and performance boost
- Concluding remarks

Direct design of LPV controllers from data

- Many possible ways to address the problem. Here:
  - ✓ Stochastic framework
  - ✓ Model reference control formulation

Direct design of LPV controllers from data

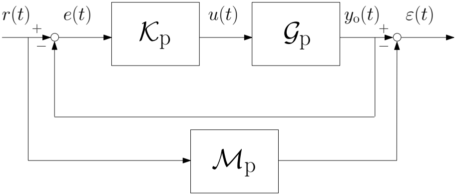
- Many possible ways to address the problem. Here:
  - ✓ Stochastic framework
  - ✓ Model reference control formulation



$$
\begin{aligned}
\theta_o = \arg\min_{\theta,\varepsilon} \|\varepsilon\|^2 & \\
\text{s.t.} \quad \varepsilon(t) &= y_o(t) - M(p,t,q^{-1})r(t) \\
A(p,t,q^{-1})y_o(t) &= B(p,t,q^{-1})u(t) \\
A_K(p,t,q^{-1},\theta)u(t) &= B_K(p,t,q^{-1},\theta)(r(t) - y_o(t))
\end{aligned}
$$

Direct design of LPV controllers from data

- Many possible ways to address the problem. Here:
  - ✓ Stochastic framework
  - ✓ Model reference control formulation



$$
\begin{aligned}
\theta_o = \arg\min_{\theta,\varepsilon} \|\varepsilon\|^2 & \\
\text{s.t.} \quad \varepsilon(t) &= y_o(t) - M(p,t,q^{-1})r(t) \\
A(p,t,q^{-1})y_o(t) &= B(p,t,q^{-1})u(t) \\
A_K(p,t,q^{-1},\theta)u(t) &= B_K(p,t,q^{-1},\theta)(r(t)-y_o(t))
\end{aligned}
$$

Model-dependent! (unique model-based contribution: Abdullah and Zribi, 2009)

Assumptions

A1. The objective is **achievable**, *i.e.*, $\exists\ \theta$ such that the closed-loop behavior corresponds to $\mathcal{M}$ for any trajectory of $p(t)$.

Assumptions

A1. The objective is **achievable**, *i.e.*, $\exists\ \theta$ such that the closed-loop behavior corresponds to $\mathcal{M}$ for any trajectory of $p(t)$.

A2. $\mathcal{G}$ is stable.

Definition D1. An LPV system $\mathcal{G}$ is **stable** if, for all trajectories $\{u(t), y(t), p(t)\}$ satisfying $\mathcal{G}$ with $u(t) = 0,\ t \geq 0$, $\exists\ \delta > 0$ s.t. $|y(t)| \leq \delta, \forall t \geq 0$.

Assumptions

A1. The objective is **achievable**, *i.e.*, $\exists\ \theta$ such that the closed-loop behavior corresponds to $\mathcal{M}$ for any trajectory of $p(t)$.

A2. $\mathcal{G}$ is stable.

A3. $\mathcal{M}$ is left-invertible.

Definition D1. An LPV system $\mathcal{G}$ is **stable** if, for all trajectories $\{u(t), y(t), p(t)\}$ satisfying $\mathcal{G}$ with $u(t) = 0,\ t \geq 0$, $\exists\ \delta > 0$ s.t. $|y(t)| \leq \delta, \forall t \geq 0$.

Definition D2. Given a causal LPV map $\mathcal{M}$ with input r, scheduling signal p and output y, the causal LPV mapping $M^{\dagger}$ that gives r as output when fed by y, for any trajectory of p, is called the **left inverse** of $\mathcal{M}$.

The main idea

- Problem reformulation:

$$\theta_o = \arg\min_{\theta,\varepsilon} \|\varepsilon\|^2 \quad \text{s.t.}$$
$$\begin{aligned}
\varepsilon(t) &= y_o(t) - M(p,t,q^{-1})r(t) \\
A(p,t,q^{-1})y_o(t) &= B(p,t,q^{-1})u(t) \\
A_K(p,t,q^{-1},\theta)u(t) &= B_K(p,t,q^{-1},\theta)(r(t)-y_o(t))
\end{aligned}$$

The main idea

- Problem reformulation:

$$\theta_o = \arg\min_{\theta,\varepsilon} \|\varepsilon\|^2 \quad \text{s.t.}$$

$$\begin{aligned} \varepsilon(t) &= y_o(t) - M(p,t,q^{-1})r(t) \\ A(p,t,q^{-1})y_o(t) &= B(p,t,q^{-1})u(t) \\ A_K(p,t,q^{-1},\theta)u(t) &= B_K(p,t,q^{-1},\theta)(r(t) - y_o(t)) \end{aligned}$$

- $r(t) = M^{\dagger}(p,t,q^{-1})\varepsilon(t) + M^{\dagger}(p,t,q^{-1})y_o(t)$, depends on the user defined $M^{\dagger}$, the unknown ε and the noiseless output

The main idea

- Problem reformulation:

$$\theta_o = \arg\min_{\theta,\varepsilon} \|\varepsilon\|^2 \quad \text{s.t.}$$

$$\begin{aligned} A(p,t,q^{-1})y_o(t) &= B(p,t,q^{-1})u(t) \\ A_K(p,t,q^{-1},\theta)u(t) &= B_K(p,t,q^{-1},\theta)(M^\dagger\varepsilon(t) + M^\dagger y_o(t) - y_o(t)) \end{aligned}$$

- $r(t) = M^\dagger(p,t,q^{-1})\varepsilon(t) + M^\dagger(p,t,q^{-1})y_o(t)$, depends on the user defined $M^\dagger$, the unknown ε and the noiseless output

The main idea

- Problem reformulation:

$$\theta_o = \arg\min_{\theta,\varepsilon} \|\varepsilon\|^2 \quad \text{s.t.}$$

$$A_K(p,t,q^{-1},\theta)u(t) \quad = \quad B_K(p,t,q^{-1},\theta)(M^\dagger \varepsilon(t) + M^\dagger y(t) - y(t))$$
$$\text{for} \quad t = 1,\dots,N$$

- $r(t) = M^\dagger(p,t,q^{-1})\varepsilon(t) + M^\dagger(p,t,q^{-1})y_o(t)$, depends on the user defined $M^\dagger$, the unknown ε and the noiseless output

- Use of IO data instead of constraint 2

The main idea

- Problem reformulation:

$$\theta_o = \arg\min_{\theta,\varepsilon} \|\varepsilon\|^2 \quad \text{s.t.}$$

$$A_K(p,t,q^{-1},\theta)u(t) \quad = \quad B_K(p,t,q^{-1},\theta)(M^\dagger\varepsilon(t) + M^\dagger y(t) - y(t))$$
$$\text{for} \quad t = 1,\ldots,N$$

  - $r(t) = M^\dagger(p,t,q^{-1})\varepsilon(t) + M^\dagger(p,t,q^{-1})y_o(t)$, depends on the user defined $M^\dagger$, the unknown ε and the noiseless output

  - Use of IO data instead of constraint 2

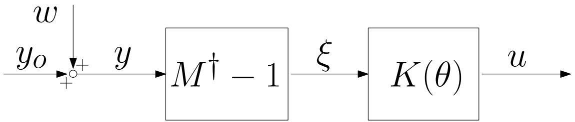
- N finite and $y(t) = y_o(t) + w(t)$, with $w(t)$ any zero mean stationary noise

The data-driven optimization problem

- Final control design problem:

$$
\begin{aligned}
&\min_{\theta,\varepsilon} \|\varepsilon\|^2 \quad \text{s.t.} \\
A_K(p,\theta)u(t) \quad &= \quad B_K(p,\theta)(M^\dagger(p)\varepsilon(t) + M^\dagger(p)y(t) - y(t)) \\
&\quad\ \text{for} \quad t = 1,\ldots,N
\end{aligned}
$$

is an **identification problem**



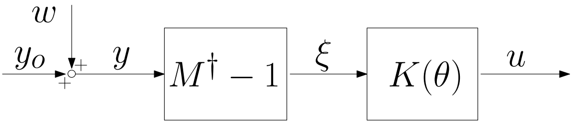
- ✓ ξ is the (virtual) noisy input
- ✓ u is the noiseless output
- ✓ ε is the residual
- ✓ $K(\theta)$ is the system to identify

The data-driven optimization problem

- Final control design problem:

$$\min_{\theta,\varepsilon} \|\varepsilon\|^2 \quad \text{s.t.}$$
$$A_K(p,\theta)u(t) = B_K(p,\theta)(M^\dagger(p)\varepsilon(t) + \xi(t))$$
$$\text{for} \quad t = 1, \ldots, N$$

is an **identification problem**



✓ ξ is the (virtual) noisy input
✓ u is the noiseless output
✓ ε is the residual
✓ $K(\theta)$ is the system to identify

The data-driven optimization problem

- Controller identification problem:

$$\min_{\theta,\varepsilon} \|\varepsilon\|^2 \quad \text{s.t.}$$
$$\begin{aligned} A_K(p,\theta)u(t) &= B_K(p,\theta)(M^\dagger(p)\varepsilon(t) + M^\dagger(p)y(t) - y(t)) \\ & \text{for} \quad t = 1,\ldots,N \end{aligned}$$

The data-driven optimization problem

- Controller identification problem:

$$\min_{\theta,\varepsilon} \|\varepsilon\|^2 \quad \text{s.t.}$$
$$\begin{aligned} A_K(p,\theta)u(t) &= B_K(p,\theta)(M^\dagger(p)\varepsilon(t) + M^\dagger(p)y(t) - y(t)) \\ &\quad \text{for} \quad t = 1,\ldots,N \end{aligned}$$

- Purely data-driven optimization problem with N constraints

The data-driven optimization problem

- Controller identification problem:

$$\min_{\theta,\varepsilon} \|\varepsilon\|^2 \quad \text{s.t.}$$
$$\begin{aligned} A_K(p,\theta)u(t) &= B_K(p,\theta)(M^\dagger(p)\varepsilon(t) + M^\dagger(p)y(t) - y(t)) \\ &\quad \text{for} \quad t = 1,\ldots,N \end{aligned}$$

- Purely data-driven optimization problem with N constraints
- Critical points:
  - ✓ the original and the current problems are equivalent when $y(t)$ is noisy?

The data-driven optimization problem

- Controller identification problem:

$$\begin{aligned} \min_{\theta,\varepsilon} \|\varepsilon\|^2 \quad \text{s.t.} & \\ A_K(p,\theta)u(t) &= B_K(p,\theta)(M^\dagger(p)\varepsilon(t) + M^\dagger(p)y(t) - y(t)) \\ & \text{for} \quad t = 1,\dots,N \end{aligned}$$

- Purely data-driven optimization problem with N constraints
- Critical points:
  - ✓ the original and the current problems are equivalent when $y(t)$ is noisy?
  - ✓ problem complexity depends on the controller parameterization

The data-driven optimization problem

- Controller identification problem:

$$
\begin{aligned}
&\min_{\theta,\varepsilon} \|\varepsilon\|^2 \quad \text{s.t.} \\
A_K(p,\theta)u(t) \quad &= \quad B_K(p,\theta)(M^{\dagger}(p)\varepsilon(t) + M^{\dagger}(p)y(t) - y(t)) \\
&\quad \text{for} \quad t = 1,\ldots,N
\end{aligned}
$$

- Purely data-driven optimization problem with N constraints
- Critical points:
  - ✓ the original and the current problems are equivalent when $y(t)$ is noisy?
  - ✓ problem complexity depends on the controller parameterization
  - ✓ computation of $M^{\dagger}(p)$ from $\mathcal{M}$ defined as

$$
\begin{aligned}
x_M(t+1) &= A_M(p,t)x_M(t) + B_M(p,t)r(t) \\
y(t) &= C_M(p,t)x_M(t) + D_M(p,t)r(t)
\end{aligned}
$$

Computation of $M^{\dagger}(p)$

- Analytical solution:

Proposition

Assume that $D_M(p,t) \neq 0, \forall p$ such that $\exists\ D_M^{-1}(p,t)$ with $D_M^{-1}(p,t) D_M(p,t) = 1,\ \forall p$. Define the state-space representation of $M^{\dagger}$ as

$$
\begin{aligned}
x_{M^\dagger}(t+1) &= A_{M^\dagger}(p,t) x_{M^\dagger}(t) + B_{M^\dagger}(p,t) y(t) \\
r(t) &= C_{M^\dagger}(p,t) x_{M^\dagger}(t) + D_{M^\dagger}(p,t) y(t)
\end{aligned}
$$

The matrices describing $M^{\dagger}$ can be computed as

$$
\begin{aligned}
A_{M^\dagger}(p,t) &= A_M(p,t) - B_M(p,t) D_M^{-1}(p,t) C_M(p,t), \\
B_{M^\dagger}(p,t) &= B_M(p,t) D_M^{-1}(p,t), \\
C_{M^\dagger}(p,t) &= -D_M^{-1}(p,t) C_M(p,t), \\
D_{M^\dagger}(p,t) &= D_M^{-1}(p,t).
\end{aligned}
$$

Two different scenarios

- In real applications, we encounter two different scenarios

  ✓ CASE A: **Fixed-structure controller tuning** (namely, parameter optimization, e.g. gain-scheduled PID tuning with affine parameterization)

  ✓ CASE B: **LPV controller design** (both structure selection and parameter optimization are needed, e.g. gain-scheduled PID tuning with unknown parameterization)

Two different scenarios - cont'd

- Controller parameterization:

$$A_K(p,t,q^{-1},\theta)u(t) = B_K(p,t,q^{-1},\theta)(r(t)-y(t))$$

$$A_K(p,t,q^{-1}) = 1 + \sum_{i=1}^{n_{\mathrm{a}_K}} a_i^K(p,t)q^{-i}$$

$$B_K(p,t,q^{-1}) = \sum_{i=0}^{n_{\mathrm{b}_K}} b_i^K(p,t)q^{-i}$$

$$a_i^K(p,t) = \sum_{j=1}^{n_0} a_{i,j}^K f_{i,j}(p,t)$$

$$b_i^K(p,t) = \sum_{j=0}^{m_i} b_{i,j}^K g_{i,j}(p,t)$$

Two different scenarios - cont'd

- Controller parameterization:

$$A_K(p, t, q^{-1}, \theta)u(t) = B_K(p, t, q^{-1}, \theta)(r(t) - y(t))$$

$$A_K(p, t, q^{-1}) = 1 + \sum_{i=1}^{n_{\mathrm{a}_K}} a_i^K(p, t)q^{-i}$$

$$B_K(p, t, q^{-1}) = \sum_{i=0}^{n_{\mathrm{b}_K}} b_i^K(p, t)q^{-i}$$

$$a_i^K(p, t) = \sum_{j=1}^{n_0} a_{i,j}^K f_{i,j}(p, t)$$

$$b_i^K(p, t) = \sum_{j=0}^{m_i} b_{i,j}^K g_{i,j}(p, t)$$

- CASE A: $f_{i,j}(p, t)$ and $g_{i,j}(p, t)$ are **a-priori** defined nonlinear (possibly dynamic) functions of p.

Fixed-structure controller tuning

- CASE A:

$$\theta = [\underline{a}_1^\top \; \dots \; \underline{a}_{n_{\mathrm{a}_K}}^\top \; \underline{b}_0^\top \; \dots \; \underline{b}_{n_{\mathrm{b}_K}}^\top]^\top$$

$$\underline{a}_i = [a_{i,1}^K \; \dots \; a_{i,n_i}^K]^\top, \; \underline{b}_i = [b_{i,1}^K \; \dots \; b_{i,m_i}^K]^\top$$

Fixed-structure controller tuning

- CASE A:

$$\theta = [\underline{a}_1^\top \ \ldots \ \underline{a}_{n_{\mathrm{a}_K}}^\top \ \underline{b}_0^\top \ \ldots \ \underline{b}_{n_{\mathrm{b}_K}}^\top]^\top$$

$$\underline{a}_i = [a_{i,1}^K \ \ldots \ a_{i,n_i}^K]^\top, \ \underline{b}_i = [b_{i,1}^K \ \ldots \ b_{i,m_i}^K]^\top$$

- The problem of interest

$$\begin{aligned} &\min_{\theta,\varepsilon} \|\varepsilon\|^2 \quad \text{s.t.} \\ A_K(p,\theta)u(t) \quad &= \quad B_K(p,\theta)(M^\dagger(p)\varepsilon(t) + M^\dagger(p)y(t) - y(t)), \forall t \end{aligned}$$

✓ is **convex** if $B_K(q^{-1}, p, \theta)$ is independent of θ

Fixed-structure controller tuning

- CASE A:

$$\theta = [\underline{a}_1^\top \ \ldots \ \underline{a}_{n_{\mathrm{a}_K}}^\top \ \underline{b}_0^\top \ \ldots \ \underline{b}_{n_{\mathrm{b}_K}}^\top]^\top$$

$$\underline{a}_i = [a_{i,1}^K \ \ldots \ a_{i,n_i}^K]^\top, \ \underline{b}_i = [b_{i,1}^K \ \ldots \ b_{i,m_i}^K]^\top$$

- The problem of interest

$$\min_{\theta,\varepsilon} \|\varepsilon\|^2 \quad \text{s.t.}$$
$$A_K(p,\theta)u(t) \quad = \quad B_K(p,\theta)(M^\dagger(p)\varepsilon(t) + M^\dagger(p)y(t) - y(t)), \forall t$$

✓ is **convex** if $B_K(q^{-1}, p, \theta)$ is independent of θ

✓ is **bi-convex** if $B_K(q^{-1}, p, \theta)$ is linear in θ

Fixed-structure controller tuning - cont'd

- Controller dynamics:

$$A_K(p,\theta)u(t) = B_K(p,\theta)\xi(t) + B_K(p,\theta)M^{\dagger}(p)\varepsilon(t)$$

Fixed-structure controller tuning - cont'd

- Controller dynamics:

$$A_K(p,\theta)u(t) = B_K(p,\theta)\xi(t) + B_K(p,\theta)M^{\dagger}(p)\varepsilon(t)$$

- Define:

$$\phi(\xi,t) = [-u(t-1)f_{1,0}(p,t),\ldots,\xi(t)g_{0,0}(p,t),\xi(t-n_{\mathrm{b}_K})g_{n_{\mathrm{b}_K},1}(p,t),\ldots]^{\top}$$

Fixed-structure controller tuning - cont'd

- Controller dynamics:

$$A_K(p,\theta)u(t) = B_K(p,\theta)\xi(t) + B_K(p,\theta)M^{\dagger}(p)\varepsilon(t)$$

- Define:

$$\phi(\xi,t) = [-u(t-1)f_{1,0}(p,t),\ldots,\xi(t)g_{0,0}(p,t),\xi(t-n_{\mathrm{b}_K})g_{n_{\mathrm{b}_K},1}(p,t),\ldots]^{\top}$$

- "Regression-like" form:

$$u(t) = \phi^{\top}(\xi,t)\theta + B_K(p,t,\theta)M^{\dagger}(p,t)\varepsilon(t)$$

Fixed-structure controller tuning - cont'd

- Controller dynamics:

$$A_K(p,\theta)u(t) = B_K(p,\theta)\xi(t) + B_K(p,\theta)M^{\dagger}(p)\varepsilon(t)$$

- Define:

$$\phi(\xi,t) = [-u(t-1)f_{1,0}(p,t),\ldots,\xi(t)g_{0,0}(p,t),\xi(t-n_{\mathrm{b}_K})g_{n_{\mathrm{b}_K},1}(p,t),\ldots]^{\top}$$

- "Regression-like" form:

$$u(t) = \phi^{\top}(\xi,t)\theta + B_K(p,t,\theta)M^{\dagger}(p,t)\varepsilon(t)$$

- θ-dependent residual (and noisy input) $\rightarrow$ instrumental variables

Fixed-structure controller tuning - cont'd

- Controller dynamics:

$$A_K(p,\theta)u(t) = B_K(p,\theta)\xi(t) + B_K(p,\theta)M^{\dagger}(p)\varepsilon(t)$$

- Define:

$$\phi(\xi,t) = [-u(t-1)f_{1,0}(p,t),\ldots,\xi(t)g_{0,0}(p,t),\xi(t-n_{\mathrm{b}_K})g_{n_{\mathrm{b}_K},1}(p,t),\ldots]^{\top}$$

- "Regression-like" form:

$$u(t) = \phi^{\top}(\xi,t)\theta + B_K(p,t,\theta)M^{\dagger}(p,t)\varepsilon(t)$$

- θ-dependent residual (and noisy input) $\rightarrow$ instrumental variables
- "Least squares"-like solution $\hat{\theta}_{\mathrm{IV}}$

Proposition

$$\lim_{N\to\infty} \hat{\theta}_{\mathrm{IV}} = \theta_o, \textit{ w.p. 1.}$$

Illustrative example

- LPV system $\mathcal{G}$ (with $\mathbb{P} = [-0.4, 0.4]$)

$$
\begin{aligned}
x_G(t+1) &= p(t)x_G(t) + u(t) \\
y(t) &= x_G(t)
\end{aligned}
$$

Illustrative example

- LPV system $\mathcal{G}$ (with $\mathbb{P} = [-0.4, 0.4]$)

$$\begin{aligned} x_G(t+1) &= p(t)x_G(t) + u(t) \\ y(t) &= x_G(t) \end{aligned}$$

- Reference LPV behaviour $\mathcal{M}$

$$\begin{aligned} x_M(t+1) &= A_M(p,t)x_M(t) + B_M(p,t)r(t) \\ y_M(t) &= C_M(p,t)x_M(t) + D_M(p,t)r(t) \end{aligned}$$

$$A_M(p,t) = \begin{bmatrix} -1 & 1 \\ -1-\Delta p(t) & 1 \end{bmatrix}, \; B_M(p,t) = \begin{bmatrix} 1+p(t) \\ 1+\Delta p(t) \end{bmatrix},$$

$$C_M = [1 \;\; 0], \; D_M = 0, \Delta p(t) = p(t) - p(t-1)$$

Illustrative example

- LPV system $\mathcal{G}$ (with $\mathbb{P} = [-0.4, 0.4]$)

$$
\begin{array}{rcl}
x_G(t+1) & = & p(t)x_G(t) + u(t) \\
y(t) & = & x_G(t)
\end{array}
$$

- Reference LPV behaviour $\mathcal{M}$

$$
\begin{array}{rcl}
x_M(t+1) & = & A_M(p,t)x_M(t) + B_M(p,t)r(t) \\
y_M(t) & = & C_M(p,t)x_M(t) + D_M(p,t)r(t)
\end{array}
$$

$$
A_M(p,t) = \left[\begin{array}{cc} -1 & 1 \\ -1-\Delta p(t) & 1 \end{array}\right], \; B_M(p,t) = \left[\begin{array}{c} 1+p(t) \\ 1+\Delta p(t) \end{array}\right],
$$

$$
C_M = [1 \;\; 0], \; D_M = 0, \Delta p(t) = p(t) - p(t-1)
$$

- Gain-scheduled PI controller $\mathcal{K}$

$$
\begin{array}{rcl}
x_K(t+1) & = & x_K(t) + \left(\theta_0(p,t) + \theta_1(p,t)\right)\left(r(t) - y(t)\right) \\
u(t) & = & x_K(t) + \theta_0(p,t)\left(r(t) - y(t)\right)
\end{array}
$$

$$
\theta_0(p,t) = \theta_{00} + \theta_{01}p(t), \; \theta_1(p,t) = \theta_{10} + \theta_{11}p(t-1)
$$

Illustrative example

- LPV system $\mathcal{G}$ (with $\mathbb{P} = [-0.4, 0.4]$)

$$\begin{array}{rcl} x_G(t+1) & = & p(t)x_G(t) + u(t) \\ y(t) & = & x_G(t) \end{array}$$

- Reference LPV behaviour $\mathcal{M}$ (achievable!)

$$\begin{array}{rcl} x_M(t+1) & = & A_M(p,t)x_M(t) + B_M(p,t)r(t) \\ y_M(t) & = & C_M(p,t)x_M(t) + D_M(p,t)r(t) \end{array}$$

$$A_M(p,t) = \begin{bmatrix} -1 & 1 \\ -1-\Delta p(t) & 1 \end{bmatrix}, \; B_M(p,t) = \begin{bmatrix} 1+p(t) \\ 1+\Delta p(t) \end{bmatrix},$$

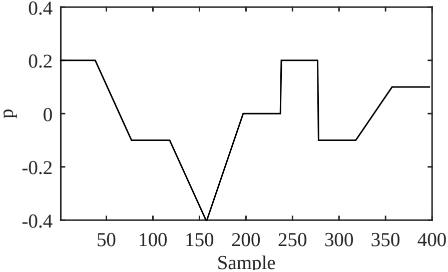
$$C_M = [1 \;\; 0], \; D_M = 0, \Delta p(t) = p(t) - p(t-1)$$

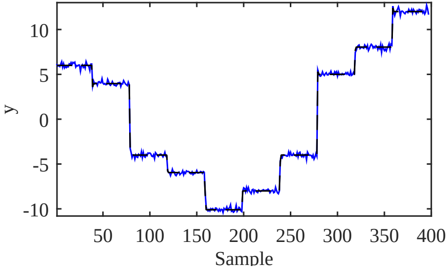
- Gain-scheduled PI controller $\mathcal{K}$

$$\begin{array}{rcl} x_K(t+1) & = & x_K(t) + (\theta_0(p,t) + \theta_1(p,t))\,(r(t) - y(t)) \\ u(t) & = & x_K(t) + \theta_0(p,t)\,(r(t) - y(t)) \end{array}$$

$$\theta_0(p,t) = \theta_{00} + \theta_{01}p(t), \; \theta_1(p,t) = \theta_{10} + \theta_{11}p(t-1)$$

Illustrative example - cont'd





Mean tracking error: **less than 1%**

$$\theta_0^\circ(p,t) = 1 + p(t) \quad \rightarrow \quad \hat{\theta}_0(p,t) = 0.9852 + 1.0166p(t)$$
$$\theta_1^\circ(p,t) = -p(t-1) \quad \rightarrow \quad \hat{\theta}_1(p,t) = -0.0153 - 0.9860p(t-1)$$

Price to pay: two experiments plus a least squares estimation

Two different scenarios - cont'd

- Controller parameterization:

$$A_K(p,t,q^{-1},\theta)u(t) = B_K(p,t,q^{-1},\theta)(r(t)-y(t))$$

$$A_K(p,t,q^{-1}) = 1 + \sum_{i=1}^{n_{\mathrm{a}_K}} a_i^K(p,t)q^{-i}$$

$$B_K(p,t,q^{-1}) = \sum_{i=0}^{n_{\mathrm{b}_K}} b_i^K(p,t)q^{-i}$$

$$a_i^K(p,t) = \sum_{j=1}^{n_0} a_{i,j}^K f_{i,j}(p,t)$$

$$b_i^K(p,t) = \sum_{j=0}^{m_i} b_{i,j}^K g_{i,j}(p,t)$$

Two different scenarios - cont'd

- Controller parameterization:

$$A_K(p,t,q^{-1},\theta)u(t) = B_K(p,t,q^{-1},\theta)(r(t)-y(t))$$

$$A_K(p,t,q^{-1}) = 1 + \sum_{i=1}^{n_{\mathrm{a}_K}} a_i^K(p,t)q^{-i}$$

$$B_K(p,t,q^{-1}) = \sum_{i=0}^{n_{\mathrm{b}_K}} b_i^K(p,t)q^{-i}$$

$$a_i^K(p,t) = \sum_{j=1}^{n_0} a_{i,j}^K f_{i,j}(p,t)$$

$$b_i^K(p,t) = \sum_{j=0}^{m_i} b_{i,j}^K g_{i,j}(p,t)$$

- CASE B: $f_{i,j}(p,t)$ and $g_{i,j}(p,t)$ are **unknown** nonlinear (possibly dynamic) functions of p.

Nonparametric tuning

- CASE B:

$$a_i^K(p,t) = \theta_i^\top \psi_i(p,t) \quad i = 1,\ldots,n_{\mathrm{a}_K}$$
$$b_i^K(p,t) = \theta_{i+n_{\mathrm{a}_K}+1}^\top \psi_{i+n_{\mathrm{a}_K}+1}(p,t) \quad i = 0,\ldots,n_{\mathrm{b}_K}$$

- $\theta_i \in \mathbb{R}^{n_{\mathrm{H}}}$ is a vector of unknown parameters and $\psi_i(p,t)$ (with $i = 1,\ldots,n_{\mathrm{a}_K}+n_{\mathrm{b}_K}+1$) is a nonlinear map from $\mathbb{P}$ to an n_{H}-dimensional space, commonly referred to as the **feature space**

Nonparametric tuning

- CASE B:

$$a_i^K(p,t) = \theta_i^\top \psi_i(p,t) \quad i = 1, \ldots, n_{\mathrm{a}_K}$$

$$b_i^K(p,t) = \theta_{i+n_{\mathrm{a}_K}+1}^\top \psi_{i+n_{\mathrm{a}_K}+1}(p,t) \quad i = 0, \ldots, n_{\mathrm{b}_K}$$

- $\theta_i \in \mathbb{R}^{n_\mathrm{H}}$ is a vector of unknown parameters and $\psi_i(p,t)$ (with $i = 1, \ldots, n_{\mathrm{a}_K} + n_{\mathrm{b}_K} + 1$) is a nonlinear map from $\mathbb{P}$ to an n_H-dimensional space, commonly referred to as the **feature space**

- Neither ψ_i nor n_H are specified. Potentially, θ_i and $\psi_i(p,t)$ can be infinite-dimensional vectors!

Nonparametric tuning - cont'd

- Controller dynamics:

$$A_K(p,\theta)u(t) = B_K(p,\theta)\xi(t) + B_K(p,\theta)M^{\dagger}(p)\varepsilon(t)$$

Nonparametric tuning - cont'd

- Controller dynamics:

$$A_K(p,\theta)u(t) = B_K(p,\theta)\xi(t) + B_K(p,\theta)M^{\dagger}(p)\varepsilon(t)$$

- Define

$$x(\xi,t)=[-u(t-1)\ \ \ldots - u(t-n_{\mathrm{a}_K})\ \ \xi(t)\ \ \ldots\ \ \xi(t-n_{\mathrm{b}_K})]^{\top}.$$

Nonparametric tuning - cont'd

- Controller dynamics:

$$A_K(p,\theta)u(t) = B_K(p,\theta)\xi(t) + B_K(p,\theta)M^{\dagger}(p)\varepsilon(t)$$

- Define

$$x(\xi,t)=[-u(t-1)\ \ldots -u(t-n_{\mathrm{a}_K})\ \ \xi(t)\ \ldots\ \xi(t-n_{\mathrm{b}_K})]^{\top}.$$

- "Regression-like" form:

$$u(t) = \sum_{i=1}^{n_{\mathrm{f}}} \theta_i^{\top}\psi_i(p,t)x_i(\xi,t) + \underbrace{B_K(p,\theta,\psi_i)M^{\dagger}(p)\varepsilon(t)}_{\varepsilon_u(t)}$$

Nonparametric tuning - cont'd

- Controller dynamics:

$$A_K(p,\theta)u(t) = B_K(p,\theta)\xi(t) + B_K(p,\theta)M^{\dagger}(p)\varepsilon(t)$$

- Define

$$x(\xi,t)=[-u(t-1)\ \ldots -u(t-n_{\mathrm{a}_K})\ \ \xi(t)\ \ldots\ \xi(t-n_{\mathrm{b}_K})]^{\top}.$$

- "Regression-like" form:

$$u(t) = \sum_{i=1}^{n_{\mathrm{f}}} \theta_i^{\top}\psi_i(p,t)x_i(\xi,t) + \underbrace{B_K(p,\theta,\psi_i)M^{\dagger}(p)\varepsilon(t)}_{\varepsilon_u(t)}$$

- θ- and ψ_i-dependent residual (and noisy input) $\rightarrow$ instrumental variables

Nonparametric tuning - cont'd

- Different problem formulation ("primal" form - still convex):

$$
\min_{\theta_i, \varepsilon_u} \frac{1}{2} \sum_{i=1}^{n_\mathrm{f}} \theta_i^\top \theta_i \; + \; \frac{\gamma}{2N^2} \sum_{i=1}^{n_\mathrm{f}} \left\| \sum_{t=1}^{N} z_i(t) \varepsilon_u(t) \right\|_2^2
$$

$$
\text{s.t. } \varepsilon_u(t) = u(t) - \sum_{i=1}^{n_\mathrm{f}} \theta_i^\top \psi_i(p, t) x_i(\xi, t), \quad \forall t \in \mathcal{I}_1^N
$$

Nonparametric tuning - cont'd

- Different problem formulation ("primal" form - still convex):

$$\min_{\theta_i,\varepsilon_u} \frac{1}{2}\sum_{i=1}^{n_\mathrm{f}} \theta_i^\top \theta_i \; + \; \frac{\gamma}{2N^2}\sum_{i=1}^{n_\mathrm{f}} \left\| \sum_{t=1}^{N} z_i(t)\varepsilon_u(t) \right\|_2^2$$

$$\text{s.t. } \varepsilon_u(t) = u(t) - \sum_{i=1}^{n_\mathrm{f}} \theta_i^\top \psi_i(p,t) x_i(\xi,t), \quad \forall t \in \mathcal{I}_1^N$$

- Identification as a **regularized optimization problem**

Nonparametric tuning - cont'd

- Different problem formulation ("primal" form - still convex):

$$\min_{\theta_i, \varepsilon_u} \frac{1}{2} \sum_{i=1}^{n_\mathrm{f}} \theta_i^\top \theta_i \;+\; \frac{\gamma}{2N^2} \sum_{i=1}^{n_\mathrm{f}} \left\| \sum_{t=1}^{N} z_i(t) \varepsilon_u(t) \right\|_2^2$$

$$\text{s.t. } \varepsilon_u(t) = u(t) - \sum_{i=1}^{n_\mathrm{f}} \theta_i^\top \psi_i(p,t) x_i(\xi,t), \quad \forall t \in \mathcal{I}_1^N$$

- Identification as a **regularized optimization problem**
- $z_i(t) \in \mathbb{R}^{n_\mathrm{H}}$: instrument

Nonparametric tuning - cont'd

- Different problem formulation ("primal" form - still convex):

$$\min_{\theta_i,\varepsilon_u} \frac{1}{2}\sum_{i=1}^{n_\mathrm{f}} \theta_i^\top \theta_i \; + \; \frac{\gamma}{2N^2}\sum_{i=1}^{n_\mathrm{f}} \left\| \sum_{t=1}^{N} z_i(t)\varepsilon_u(t) \right\|_2^2$$

$$\text{s.t. } \varepsilon_u(t) = u(t) - \sum_{i=1}^{n_\mathrm{f}} \theta_i^\top \psi_i(p,t) x_i(\xi,t), \quad \forall t \in \mathcal{I}_1^N$$

- Identification as a **regularized optimization problem**
- $z_i(t) \in \mathbb{R}^{n_\mathrm{H}}$: instrument

Proposition

$$\lim_{N\to\infty} \hat{\theta}_{\mathrm{NP,IV}} = \theta^\circ - R^{-1}\gamma^{-1}\theta^\circ,$$

$$R = \lim_{N\to\infty} \gamma^{-1} I + \frac{1}{N^2}\Psi^\top Z Z^\top \Psi$$

Nonparametric tuning - cont'd

- Problem: $\hat{\theta}_{\mathrm{NP,IV}}$ cannot be computed since an explicit representation of $\psi_i(p, t)$ and $z_i(t)$ would be needed

Nonparametric tuning - cont'd

- Problem: $\hat{\theta}_{\mathrm{NP,IV}}$ cannot be computed since an explicit representation of $\psi_i(p,t)$ and $z_i(t)$ would be needed
- Dual problem:

$$\mathcal{L}(\alpha,\theta,E) = \frac{1}{2}\sum_{i=1}^{n_{\mathrm{f}}}\theta_i^\top\theta_i \;+\; \frac{\gamma}{2N^2}\sum_{i=1}^{n_{\mathrm{f}}}\left\|\Psi_i^\top X_i(\hat{\xi})E\right\|_2^2 + \\ -\alpha^\top\left(E - U + \sum_{i=1}^{n_{\mathrm{f}}} X_i(\xi)\Psi_i\theta_i\right),$$

where $\alpha \in \mathbb{R}^N$ are **Lagrangian multipliers**

Nonparametric tuning - cont'd

- Problem: $\hat{\theta}_{\mathrm{NP,IV}}$ cannot be computed since an explicit representation of $\psi_i(p,t)$ and $z_i(t)$ would be needed
- Dual problem:

$$\begin{aligned}\mathcal{L}(\alpha,\theta,E) =& \frac{1}{2}\sum_{i=1}^{n_\mathrm{f}} \theta_i^\top \theta_i \; + \; \frac{\gamma}{2N^2}\sum_{i=1}^{n_\mathrm{f}} \left\| \Psi_i^\top X_i(\hat{\xi}) E \right\|_2^2 + \\ & -\alpha^\top \left(E - U + \sum_{i=1}^{n_\mathrm{f}} X_i(\xi)\Psi_i \theta_i \right),\end{aligned}$$

  where $\alpha \in \mathbb{R}^N$ are **Lagrangian multipliers**
- From *Karush-Kuhn-Tucker* (KKT) conditions for all $i = 1, \ldots, n_\mathrm{f}$:

$$\alpha = R_\mathrm{D}^{-1}(\Psi_i)\frac{1}{N^2}\sum_{i=1}^{n_\mathrm{f}} X_i(\hat{\xi})\Psi_i\Psi_i^\top X_i(\hat{\xi})U,$$

$$R_\mathrm{D}(\Psi_i) = \gamma^{-1} I + \frac{1}{N^2}\sum_{i=1}^{n_\mathrm{f}} X_i(\hat{\xi})\Psi_i\Psi_i^\top X_i(\hat{\xi}) \sum_{j=1}^{n_\mathrm{f}} X_j(\xi)\Psi_j\Psi_j^\top X_j(\xi).$$

Nonparametric tuning - cont'd

- Define the *Grammian* matrix as $\Omega_i = \Psi_i \Psi_i^\top$.
- According to the Mercer's theorem, the generic (t, k)-th entry of Ω_i can be described by a positive definite *kernel* function $\kappa_i(p, t, k)$, i.e.,

$$[\Omega_i]_{t,k} = \left\langle \psi_i(p, t), \psi_i(p, k) \right\rangle = \kappa_i(p, t, k).$$

Nonparametric tuning - cont'd

- Define the *Grammian* matrix as $\Omega_i = \Psi_i \Psi_i^\top$.
- According to the Mercer's theorem, the generic (t, k)-th entry of Ω_i can be described by a positive definite *kernel* function $\kappa_i(p, t, k)$, i.e.,

$$[\Omega_i]_{t,k} = \left\langle \psi_i(p, t), \psi_i(p, k) \right\rangle = \kappa_i(p, t, k).$$

- Specification of the kernels instead of the maps ψ_i is called the **kernel trick** (Vapnik, 1998)

Nonparametric tuning - cont'd

- Define the *Grammian* matrix as $\Omega_i = \Psi_i \Psi_i^\top$.
- According to the Mercer's theorem, the generic (t, k)-th entry of Ω_i can be described by a positive definite *kernel* function $\kappa_i(p, t, k)$, i.e.,

$$[\Omega_i]_{t,k} = \left\langle \psi_i(p, t), \psi_i(p, k) \right\rangle = \kappa_i(p, t, k).$$

- Specification of the kernels instead of the maps ψ_i is called the **kernel trick** (Vapnik, 1998)
- In our case:

$$\alpha = R_{\mathrm{D}}^{-1}(\Omega_i) \frac{1}{N^2} \sum_{i=1}^{n_{\mathrm{f}}} X_i(\hat{\xi}) \Psi_i \Psi_i^\top X_i(\hat{\xi}) U$$

with

$$R_{\mathrm{D}}(\Omega_i) = \gamma^{-1} I + \frac{1}{N^2} \sum_{i=1}^{n_{\mathrm{f}}} X_i(\hat{\xi}) \Psi_i \Psi_i^\top X_i(\hat{\xi}) \sum_{j=1}^{n_{\mathrm{f}}} X_j(\xi) \Psi_i \Psi_i^\top X_j(\xi).$$

Nonparametric tuning - cont'd

- Define the *Grammian* matrix as $\Omega_i = \Psi_i \Psi_i^\top$.
- According to the Mercer's theorem, the generic (t,k)-th entry of Ω_i can be described by a positive definite *kernel* function $\kappa_i(p,t,k)$, i.e.,

$$[\Omega_i]_{t,k} = \langle \psi_i(p,t), \psi_i(p,k) \rangle = \kappa_i(p,t,k).$$

- Specification of the kernels instead of the maps ψ_i is called the **kernel trick** (Vapnik, 1998)
- In our case:

$$\alpha = R_{\mathrm{D}}^{-1}(\Omega_i) \frac{1}{N^2} \sum_{i=1}^{n_{\mathrm{f}}} X_i(\hat{\xi}) \Omega_i X_i(\hat{\xi}) U$$

with

$$R_{\mathrm{D}}(\Omega_i) = \gamma^{-1} I + \frac{1}{N^2} \sum_{i=1}^{n_{\mathrm{f}}} X_i(\hat{\xi}) \Omega_i X_i(\hat{\xi}) \sum_{j=1}^{n_{\mathrm{f}}} X_j(\xi) \Omega_i X_j(\xi).$$

Nonparametric tuning - cont'd

- A typical choice of kernel is the **Radial Basis Function** (RBF) kernel:

$$\kappa_i(p,t,k) = \exp\left(-\frac{\|p(t)-p(k)\|_2^2}{\sigma_i^2}\right),$$

Nonparametric tuning - cont'd

- A typical choice of kernel is the **Radial Basis Function** (RBF) kernel:

$$\kappa_i(p,t,k) = \exp\left(-\frac{\|p(t)-p(k)\|_2^2}{\sigma_i^2}\right),$$

- $\sigma_i > 0$ is a **hyper-parameter** characterizing the width of the RBF and it is tuned by the user (e.g., through cross-validation).

Nonparametric tuning - cont'd

- A typical choice of kernel is the **Radial Basis Function** (RBF) kernel:

$$\kappa_i(p,t,k) = \exp\left(-\frac{\|p(t)-p(k)\|_2^2}{\sigma_i^2}\right),$$

- $\sigma_i > 0$ is a **hyper-parameter** characterizing the width of the RBF and it is tuned by the user (e.g., through cross-validation).
- Once the Lagrangian multipliers α are computed, the p-dependent coefficient functions $a_i^K(p,t)$ and $b_i^K(p,t)$ characterizing the LPV controller are obtained as

$$a_i^K(\cdot) = \psi_i^\top(\cdot)\theta_i = \psi_i^\top(\cdot)\Psi_i^\top X_i(\xi)\alpha = \sum_{t=1}^{N} \underbrace{\psi_i^\top(\cdot)\psi_i(p,t)}_{\kappa_i(p,t,\cdot)} x_i(\xi,t)\alpha_t,$$

$$b_i^K(\cdot) = \psi_{i+n_{a_K}+1}^\top(\cdot)\theta_{i+n_{a_K}+1} = \psi_{i+n_{a_K}+1}^\top(\cdot)\Psi_{i+n_{a_K}+1}^\top X_{i+n_{a_K}+1}(\xi)\alpha =$$

$$= \sum_{t=1}^{N} \underbrace{\psi_{i+n_{a_K}+1}^\top(\cdot)\psi_{i+n_{a_K}+1}(p,t)}_{\kappa_{i+n_{a_K}+1}(p,t,\cdot)} x_{i+n_{a_K}+1}(\xi,t)\alpha_t.$$

Illustrative example Reprise

- LPV system $\mathcal{G}$ (with $\mathbb{P} = [-0.4, 0.4]$)

$$\begin{aligned} x_G(t+1) &= p(t)x_G(t) + u(t) \\ y(t) &= x_G(t) \end{aligned}$$

- Reference achievable LPV behaviour $\mathcal{M}$

$$\begin{aligned} x_M(t+1) &= A_M(p,t)x_M(t) + B_M(p,t)r(t) \\ y_M(t) &= C_M(p,t)x_M(t) + D_M(p,t)r(t) \end{aligned}$$

$$A_M(p,t) = \begin{bmatrix} -1 & 1 \\ -1-\Delta p(t) & 1 \end{bmatrix}, \ B_M(p,t) = \begin{bmatrix} 1+p(t) \\ 1+\Delta p(t) \end{bmatrix},$$

$$C_M = [1 \ \ 0], \ D_M = 0, \Delta p(t) = p(t) - p(t-1)$$

Illustrative example Reprise

- LPV system $\mathcal{G}$ (with $\mathbb{P} = [-0.4, 0.4]$)

$$\begin{array}{rcl} x_G(t+1) & = & p(t)x_G(t) + u(t) \\ y(t) & = & x_G(t) \end{array}$$

- Reference achievable LPV behaviour $\mathcal{M}$

$$\begin{array}{rcl} x_M(t+1) & = & A_M(p,t)x_M(t) + B_M(p,t)r(t) \\ y_M(t) & = & C_M(p,t)x_M(t) + D_M(p,t)r(t) \end{array}$$

$$A_M(p,t) = \begin{bmatrix} -1 & 1 \\ -1-\Delta p(t) & 1 \end{bmatrix}, \; B_M(p,t) = \begin{bmatrix} 1+p(t) \\ 1+\Delta p(t) \end{bmatrix},$$

$$C_M = [1 \;\; 0], \; D_M = 0, \Delta p(t) = p(t) - p(t-1)$$

- Nonparametric controller $\mathcal{K}$

$$\begin{aligned} u(t) = a_1^K(p(t), p(t-1))u(t-1) + \\ + b_0^K(p(t), p(t-1))\,(r(t) - y(t)) + \\ + b_1^K(p(t), p(t-1))\,(r(t-1) - y(t-1)) \end{aligned}$$

Illustrative example Reprise - cont'd

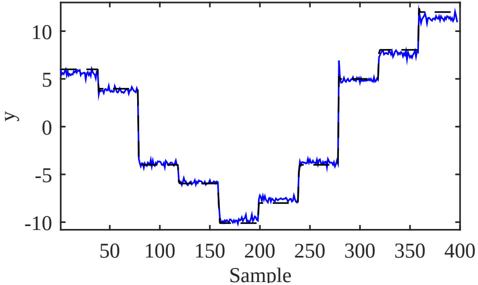
- Same IO data as before
- Now the dependence of a_1^K, b_0^K and b_1^K on $p(t)$ and $p(t-1)$ is **not** a-priori specified (neither the integral action is fixed)

Illustrative example Reprise - cont'd

- Same IO data as before
- Now the dependence of a_1^K, b_0^K and b_1^K on $p(t)$ and $p(t-1)$ is **not** a-priori specified (neither the integral action is fixed)
- γ and σ_i $(i = 1, 2, 3)$ are chosen through cross-validation

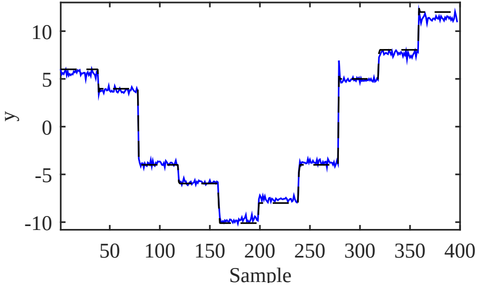
Illustrative example Reprise - cont'd

- Same IO data as before
- Now the dependence of a_1^K, b_0^K and b_1^K on $p(t)$ and $p(t-1)$ is **not** a-priori specified (neither the integral action is fixed)
- γ and σ_i $(i = 1, 2, 3)$ are chosen through cross-validation



Illustrative example Reprise - cont'd

- Same IO data as before
- Now the dependence of a_1^K, b_0^K and b_1^K on $p(t)$ and $p(t-1)$ is **not** a-priori specified (neither the integral action is fixed)
- γ and σ_i $(i = 1, 2, 3)$ are chosen through cross-validation



	parametric	**nonparametric**
MSE	0.0407	0.1565

A case study

- In a real-world application,
  - ✓ Assumption **A1** may not be verified ($\mathcal{M}$ is usually given as a - barely achievable - LTI model)
  - ✓ The system dynamics is much more complex

A case study

- In a real-world application,
  - ✓ Assumption **A1** may not be verified ($\mathcal{M}$ is usually given as a - barely achievable - LTI model)
  - ✓ The system dynamics is much more complex
- A more realistic case study :

Voltage-controlled DC motor with an unbalanced disk

A case study - cont'd

- Continuous-time system dynamics

$$\begin{bmatrix} \dot{\theta}(\tau) \\ \dot{\omega}(\tau) \\ \dot{I}(\tau) \end{bmatrix} = \left(\begin{bmatrix} 0 & 1 & 0 \\ 0 & -\frac{b}{J} & \frac{K}{J} \\ 0 & -\frac{K}{L} & -\frac{R}{L} \end{bmatrix} + \begin{bmatrix} 0 & 1 & 0 \\ \frac{mgl}{J} & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \frac{\sin(\theta(\tau))}{\theta(\tau)} \right) \begin{bmatrix} \theta(\tau) \\ \omega(\tau) \\ I(\tau) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \frac{1}{L} \end{bmatrix} V(\tau)$$

$$y(\tau) = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \theta(\tau) \\ \omega(\tau) \\ I(\tau) \end{bmatrix},$$

A case study - cont'd

- Continuous-time system dynamics

$$\begin{bmatrix} \dot{\theta}(\tau) \\ \dot{\omega}(\tau) \\ \dot{I}(\tau) \end{bmatrix} = \left(\begin{bmatrix} 0 & 1 & 0 \\ 0 & -\frac{b}{J} & \frac{K}{J} \\ 0 & -\frac{K}{L} & -\frac{R}{L} \end{bmatrix} + \begin{bmatrix} 0 & 1 & 0 \\ \frac{mgl}{J} & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \frac{\sin(\theta(\tau))}{\theta(\tau)} \right) \begin{bmatrix} \theta(\tau) \\ \omega(\tau) \\ I(\tau) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \frac{1}{L} \end{bmatrix} V(\tau)$$

$$y(\tau) = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \theta(\tau) \\ \omega(\tau) \\ I(\tau) \end{bmatrix},$$

- Third order system with dynamics depending on the load angular position θ

A case study - cont'd

- Continuous-time system dynamics

$$\begin{bmatrix} \dot{\theta}(\tau) \\ \dot{\omega}(\tau) \\ \dot{I}(\tau) \end{bmatrix} = \left(\begin{bmatrix} 0 & 1 & 0 \\ 0 & -\frac{b}{J} & \frac{K}{J} \\ 0 & -\frac{K}{L} & -\frac{R}{L} \end{bmatrix} + \begin{bmatrix} 0 & 1 & 0 \\ \frac{mgl}{J} & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \frac{\sin(\theta(\tau))}{\theta(\tau)} \right) \begin{bmatrix} \theta(\tau) \\ \omega(\tau) \\ I(\tau) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \frac{1}{L} \end{bmatrix} V(\tau)$$

$$y(\tau) = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \theta(\tau) \\ \omega(\tau) \\ I(\tau) \end{bmatrix},$$

- Third order system with dynamics depending on the load angular position θ
- Model is unknown: $p(\tau) = \theta(\tau)$ and functional dependency to be learnt

A case study - cont'd

- Continuous-time system dynamics

$$\begin{bmatrix} \dot{\theta}(\tau) \\ \dot{\omega}(\tau) \\ \dot{I}(\tau) \end{bmatrix} = \left(\begin{bmatrix} 0 & 1 & 0 \\ 0 & -\frac{b}{J} & \frac{K}{J} \\ 0 & -\frac{K}{L} & -\frac{R}{L} \end{bmatrix} + \begin{bmatrix} 0 & 1 & 0 \\ \frac{mgl}{J} & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \frac{\sin(\theta(\tau))}{\theta(\tau)} \right) \begin{bmatrix} \theta(\tau) \\ \omega(\tau) \\ I(\tau) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \frac{1}{L} \end{bmatrix} V(\tau)$$

$$y(\tau) = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \theta(\tau) \\ \omega(\tau) \\ I(\tau) \end{bmatrix},$$

- Third order system with dynamics depending on the load angular position θ
- Model is unknown: $p(\tau) = \theta(\tau)$ and functional dependency to be learnt

- The system is **quasi LPV** (more realistic setting)

A case study - cont'd

- Data-driven controller tuning:
  - ✓ 1st order LTI reference model (≈ 0.15 Hz dynamics)

$$x_M(t+1) = 0.99x_M(t) + 0.01r(t)$$
$$\theta_M(t) = x_M(t)$$

  - ✓ 4th order nonparametric controller with integral action

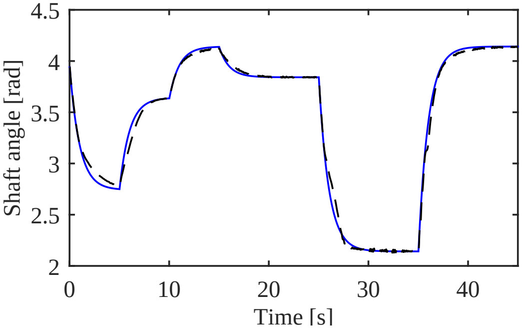
$$u(t) = \sum_{i=1}^{4} a_i^K(\Pi(t))u(t-i) + \sum_{j=0}^{4} b_j^K(\Pi(t))e_{int}(t-j)$$
$$e_{int}(t) = e_{int}(t-1) + (r(t) - y(t)),$$
$$\Pi(t) = [p(t-1)\ p(t-2)\ p(t-3)\ p(t-4)]^\top,$$

A case study - cont'd

- γ and $\sigma_i, \ \forall i$ are chosen through cross-validation

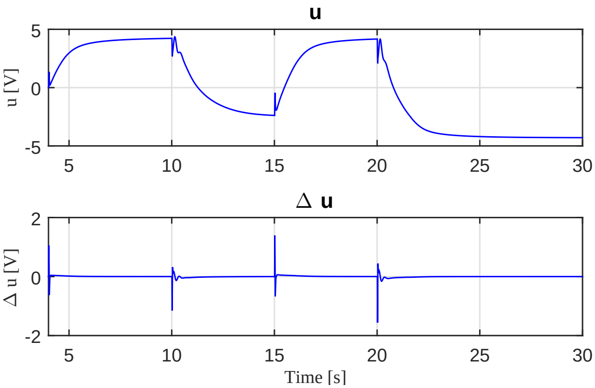
A case study - cont'd

- γ and σ_i, $\forall i$ are chosen through cross-validation



Closed-loop matching satisfactory almost everywhere (...but is the reference model acceptable?)

Moreover...

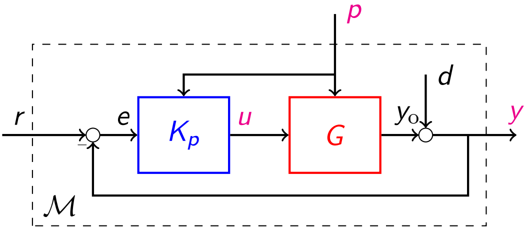


Constraints on the input signals cannot be handled

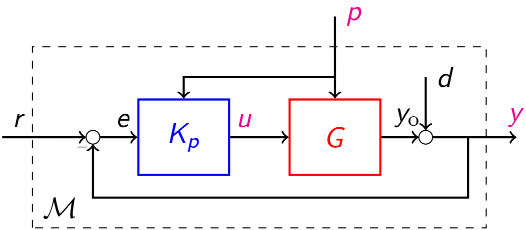
Outline

- Problem formulation
- Direct design of LPV controllers from data
- Constraint management and performance boost
- Concluding remarks

Hierarchical control architecture

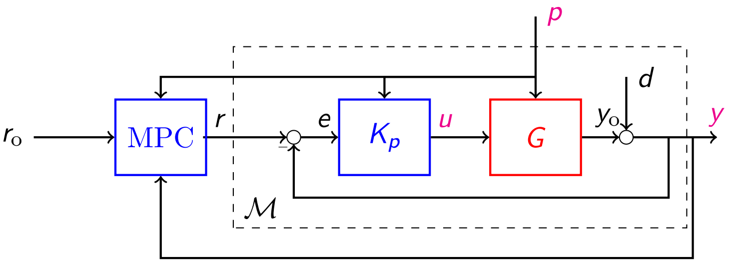


Hierarchical control architecture

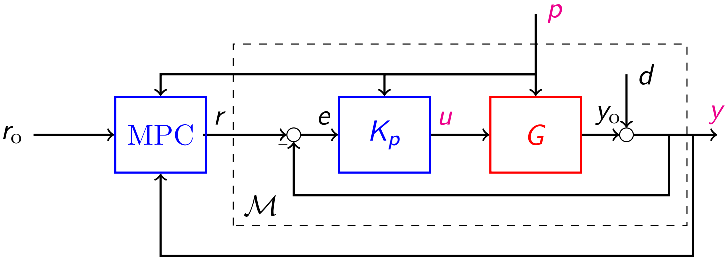


The model $\mathcal{M}$ describes the relation between r and y!

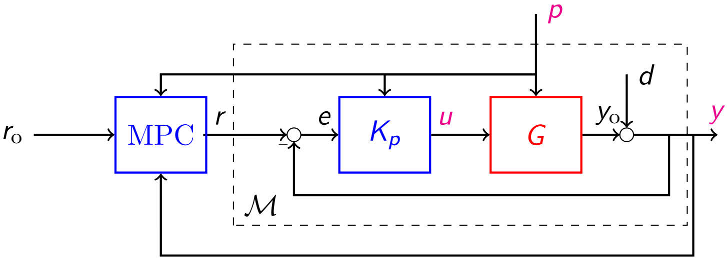
Hierarchical control architecture



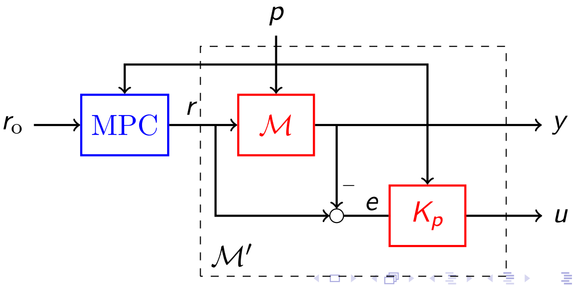
Hierarchical control architecture



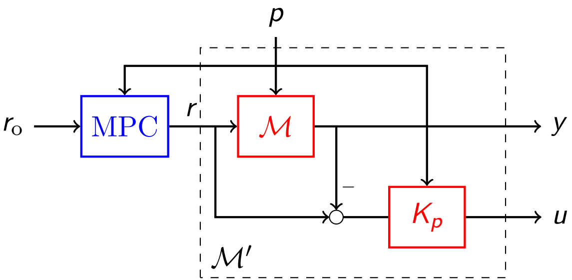
Hierarchical control architecture



Control design scheme:

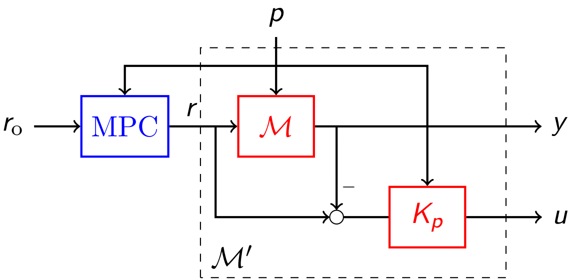


MPC design



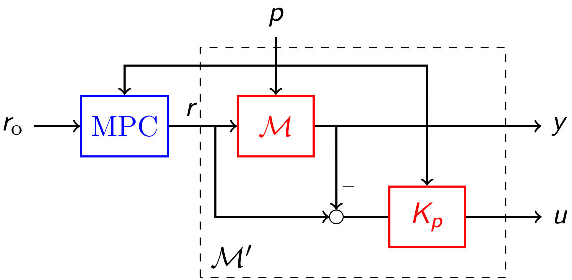
$$\min_{\color{red}r} \sum_{t=1}^{N} Q_y \left(y(t+k|k) - r_o(t+k)\right)^2 +$$

MPC design



$$\min_{r} \sum_{t=1}^{N} Q_y \left(y(t+k|k) - r_{\mathrm{o}}(t+k)\right)^2 + \sum_{t=1}^{N_c} Q_r \left(r_{\mathrm{o}}(t+k) - r(t+k|k)\right)^2$$

MPC design

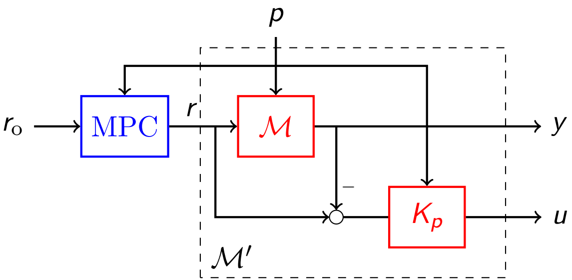


$$\min_{r} \sum_{t=1}^{N} Q_y \left(y(t+k|k) - r_o(t+k)\right)^2 + \sum_{t=1}^{N_c} Q_r \left(r_o(t+k) - r(t+k|k)\right)^2$$

$$\text{s.t.}\, u_{\min} \leq u(t+k|k) \leq u_{\max} \quad t = 1, \ldots, N$$

$$\Delta u_{\min} \leq u(t+k+1|k) - u(t+k|k) \leq \Delta u_{\max} \quad t = 1, \ldots, N-1$$

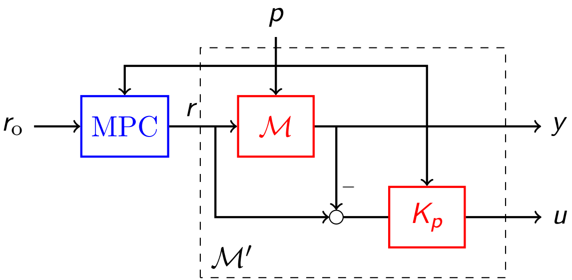
MPC design



$$\min_{r} \sum_{t=1}^{N} Q_y \left(y(t+k|k) - r_{\rm o}(t+k)\right)^2 + \sum_{t=1}^{N_c} Q_r \left(r_{\rm o}(t+k) - r(t+k|k)\right)^2$$

$$\begin{aligned}\text{s.t.}\, & u_{\min} \leq u(t+k|k) \leq u_{\max} \quad t=1,\dots,N \\ & \Delta u_{\min} \leq u(t+k+1|k) - u(t+k|k) \leq \Delta u_{\max} \quad t=1,\dots,N-1 \\ & y_{\min} \leq y(t+k|k) \leq y_{\max} \quad t=1,\dots,N\end{aligned}$$

MPC design

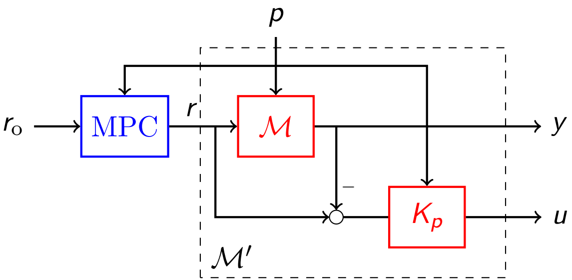


$$\min_{r} \sum_{t=1}^{N} Q_y \left(y(t+k|k) - r_o(t+k)\right)^2 + \sum_{t=1}^{N_c} Q_r \left(r_o(t+k) - r(t+k|k)\right)^2$$

$$\text{s.t.}\, u_{\min} \leq u(t+k|k) \leq u_{\max} \quad t = 1, \dots, N$$

$$\Delta u_{\min} \leq u(t+k+1|k) - u(t+k|k) \leq \Delta u_{\max} \quad t = 1, \dots, N-1$$

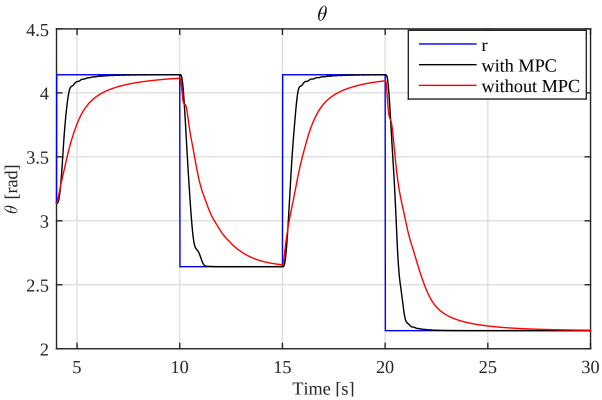
$$y_{\min} \leq y(t+k|k) \leq y_{\max} \quad t = 1, \dots, N$$



$$\min_{r} \sum_{t=1}^{N} Q_y \left(y(t+k|k) - r_o(t+k)\right)^2 + \sum_{t=1}^{N_c} Q_r \left(r_o(t+k) - r(t+k|k)\right)^2$$

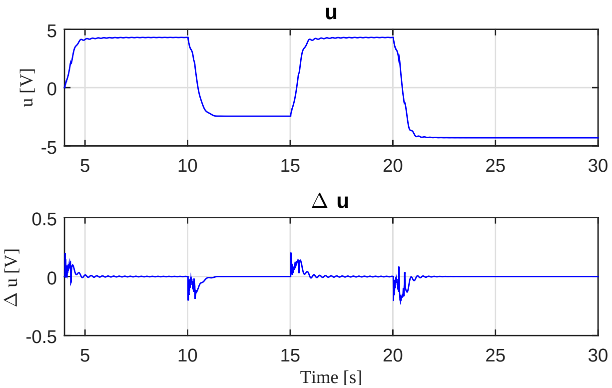
$$\begin{aligned} \text{s.t.}\, & u_{\min} \leq u(t+k|k) \leq u_{\max} \quad t = 1, \ldots, N \\ & \Delta u_{\min} \leq u(t+k+1|k) - u(t+k|k) \leq \Delta u_{\max} \quad t = 1, \ldots, N-1 \\ & y_{\min} \leq y(t+k|k) \leq y_{\max} \quad t = 1, \ldots, N \\ & x_{M'}(t+k+1|k) = A_{M'}(p(k+t))x_{M'}(t+k|k) + B_{M'}(p(k+t))r(t+k|k) \\ & \begin{bmatrix} y(t+k|k) \\ u(t+k|k) \end{bmatrix} = C_{M'}(p(k+t))x_{M'}(t+k|k) \end{aligned}$$

DC motor: obtained results



Average time required to compute the control law: 11 ms (MPC toolbox, Matlab)

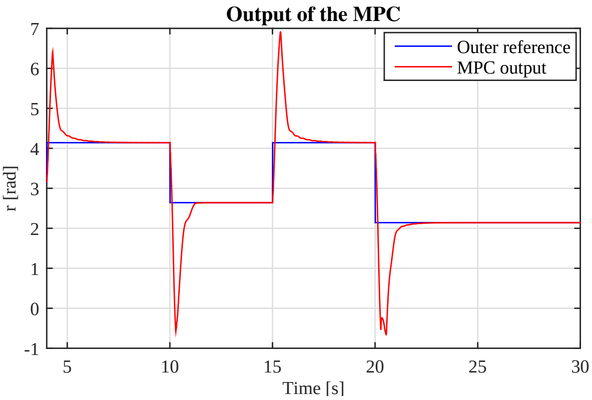
DC motor: obtained results



$|\Delta u| \leq 0.2$ V

Constraints on the input signal $|\Delta u| \leq 0.2$ V are fulfilled

DC motor: obtained results



Outline

- Problem formulation
- Direct design of LPV controllers from data
- Constraint management and performance boost
- Concluding remarks

Conclusions and future works

- An LPV controller K_p is designed from data (without using a model of the plant) to achieve a desired closed loop behaviour $\mathcal{M}$ a-priori specified by the user.

Conclusions and future works

- An LPV controller K_p is designed from data (without using a model of the plant) to achieve a desired closed loop behaviour $\mathcal{M}$ a-priori specified by the user.
- The desired closed-loop behavior should not be too demanding, otherwise it may not be achieved.

Conclusions and future works

- An LPV controller K_p is designed from data (without using a model of the plant) to achieve a desired closed loop behaviour $\mathcal{M}$ a-priori specified by the user.
- The desired closed-loop behavior should not be too demanding, otherwise it may not be achieved.
- The known model $\mathcal{M}$ is used to design an outer MPC controller, which allows one to handle input/output constraints and to improve the closed-loop performance.

Conclusions and future works

- An LPV controller K_p is designed from data (without using a model of the plant) to achieve a desired closed loop behaviour $\mathcal{M}$ a-priori specified by the user.
- The desired closed-loop behavior should not be too demanding, otherwise it may not be achieved.
- The known model $\mathcal{M}$ is used to design an outer MPC controller, which allows one to handle input/output constraints and to improve the closed-loop performance.

- Use explicit MPC to reduce the online computational time.

Conclusions and future works

- An LPV controller K_p is designed from data (without using a model of the plant) to achieve a desired closed loop behaviour $\mathcal{M}$ a-priori specified by the user.
- The desired closed-loop behavior should not be too demanding, otherwise it may not be achieved.
- The known model $\mathcal{M}$ is used to design an outer MPC controller, which allows one to handle input/output constraints and to improve the closed-loop performance.

- Use explicit MPC to reduce the online computational time.
- Design robust MPC able to handle uncertainties due to a possible mismatch between the desired inner closed loop behaviour $\mathcal{M}$ and the actual one.

Conclusions and future works

- An LPV controller K_p is designed from data (without using a model of the plant) to achieve a desired closed loop behaviour $\mathcal{M}$ a-priori specified by the user.
- The desired closed-loop behavior should not be too demanding, otherwise it may not be achieved.
- The known model $\mathcal{M}$ is used to design an outer MPC controller, which allows one to handle input/output constraints and to improve the closed-loop performance.

- Use explicit MPC to reduce the online computational time.
- Design robust MPC able to handle uncertainties due to a possible mismatch between the desired inner closed loop behaviour $\mathcal{M}$ and the actual one.
- Real-world applications?