Regularized Moving-Horizon PWA Regression for LPV System Identification

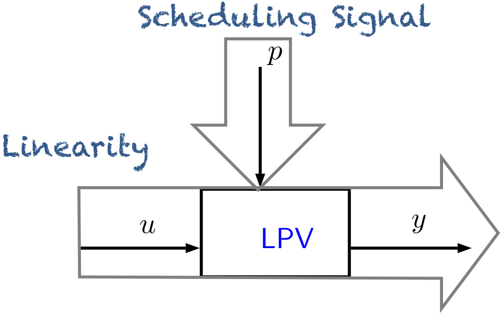
Manas Mejari[1], Vihangkumar V. Naik[1], Dario Piga[2] and Alberto Bemporad[1]
Presenter: Valentina Breschi[3]

[1]IMT School for Advanced Studies Lucca, Italy
{manas.mejari,vihangkumar.naik,alberto.bemporad}@imtlucca.it

[2]The IDSIA Dalle Molle Institute for Artificial Intelligence, Manno, Switzerland
dario.piga@supsi.ch

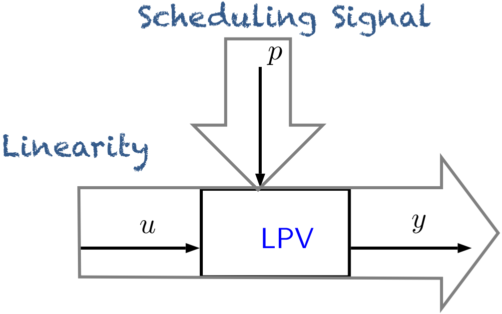
[3]Politecnico di Milano, Milano, Italy
valentina.breschi@polimi.it

Linear Parameter-Varying (LPV) Concept



- LPV paradigm: Linear dynamic relation between input and output
- Unlike Linear Time-Invariant (LTI), the relation changes over time according to a measurable time-varying 'scheduling signal' p.

Linear Parameter-Varying (LPV) Concept



- LPV paradigm: Linear dynamic relation between input and output
- Unlike Linear Time-Invariant (LTI), the relation changes over time according to a measurable time-varying 'scheduling signal' p.

LPV Model Identification

LPV-Input Output model:

$$y(k) = a_0(p(k)) + \sum_{j=1}^{n_a} a_j(p(k)) y(k-j) + \sum_{j=1}^{n_b} a_{j+n_a}(p(k)) u(k-j),$$

- $y(k), u(k)$ measured outputs and inputs at time k
- $p(k)$ measured scheduling variable, values in a set $\mathcal{P} \subseteq \mathbb{R}^{n_p}$

Objective

Given the dataset of N observations: $\{y(k), u(k), p(k)\}_{k=1}^{N}$,

- Estimate the p-dependent coefficient functions $a_j(p(k))$.

LPV Model Identification

LPV-Input Output model:

$$y(k) = a_0(p(k)) + \sum_{j=1}^{n_a} a_j(p(k)) y(k-j) + \sum_{j=1}^{n_b} a_{j+n_a}(p(k)) u(k-j),$$

- $y(k), u(k)$ measured outputs and inputs at time k
- $p(k)$ measured scheduling variable, values in a set $\mathcal{P} \subseteq \mathbb{R}^{n_p}$

Objective

Given the dataset of N observations: $\{y(k), u(k), p(k)\}_{k=1}^{N}$,

- Estimate the p-dependent coefficient functions $a_j(p(k))$.

LPV Model Identification

LPV-Input Output model:

$$y(k) = a_0(p(k)) + \sum_{j=1}^{n_a} a_j(p(k)) y(k-j) + \sum_{j=1}^{n_b} a_{j+n_a}(p(k)) u(k-j),$$

- $y(k), u(k)$ measured outputs and inputs at time k
- $p(k)$ measured scheduling variable, values in a set $\mathcal{P} \subseteq \mathbb{R}^{n_p}$

Objective

Given the dataset of N observations: $\{y(k), u(k), p(k)\}_{k=1}^{N}$,

- Estimate the p-dependent coefficient functions $a_j(p(k))$.

LPV Model Identification

LPV-Input Output model:

$$y(k) = a_0(p(k)) + \sum_{j=1}^{n_a} a_j(p(k)) y(k-j) + \sum_{j=1}^{n_b} a_{j+n_a}(p(k)) u(k-j),$$

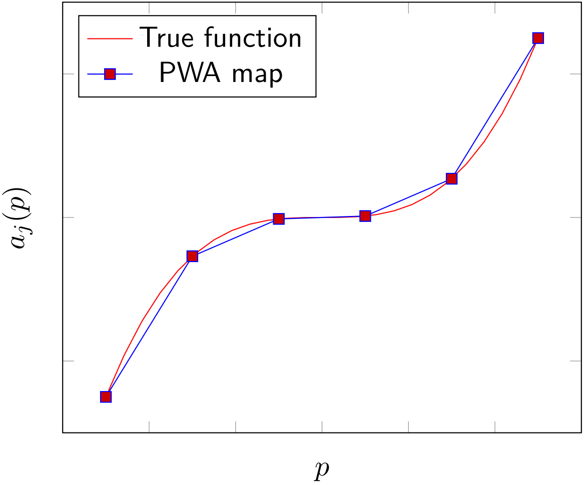
- $y(k), u(k)$ measured outputs and inputs at time k
- $p(k)$ measured scheduling variable, values in a set $\mathcal{P} \subseteq \mathbb{R}^{n_p}$

Objective

Given the dataset of N observations: $\{y(k), u(k), p(k)\}_{k=1}^{N}$,

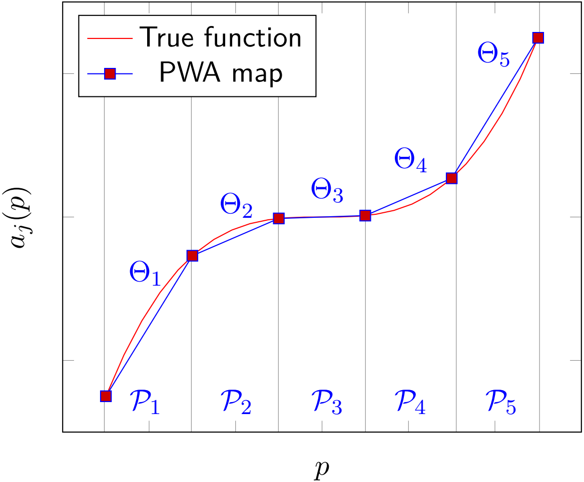
- Estimate the p-dependent coefficient functions $a_j(p(k))$.

LPV Identification as PWA regression



- Proposed idea: Approximate non-linear functions $a_j(p(k))$ with PieceWise Affine (PWA) maps

LPV Identification as PWA regression



$$a_j(p(k)) = \begin{cases} \Theta_{1,j}\ p & \text{if } p \in \mathcal{P}_1, \\ \vdots & \\ \Theta_{s,j}\ p & \text{if } p \in \mathcal{P}_s, \end{cases}$$

LPV Identification as PWA regression

- LPV-Input Output model

$$y(k) = a_0(p(k)) + \sum_{j=1}^{n_a} a_j(p(k)) y(k-j) + \sum_{j=1}^{n_b} a_{j+n_a}(p(k)) u(k-j),$$

can be written in the following PWA-LPV form:

$$y(k) = \begin{cases} \Theta_1 \; x(k) & \text{if } p(k) \in \mathcal{P}_1, \\ \vdots & \\ \Theta_s \; x(k) & \text{if } p(k) \in \mathcal{P}_s. \end{cases}$$

- $x(k)$ the regressor vector

$$x(k) = \begin{bmatrix} 1 & y'(k-1) & \cdots & y'(k-n_a) & u'(k-1) & \cdots & u'(k-n_b) \end{bmatrix}' \otimes \begin{bmatrix} 1 \\ p(k) \end{bmatrix}.$$

LPV Identification as PWA regression

- LPV-Input Output model

$$y(k) = a_0(p(k)) + \sum_{j=1}^{n_a} a_j(p(k)) y(k-j) + \sum_{j=1}^{n_b} a_{j+n_a}(p(k)) u(k-j),$$

can be written in the following PWA-LPV form:

$$y(k) = \begin{cases} \Theta_1 \; x(k) & \text{if } p(k) \in \mathcal{P}_1, \\ \vdots & \\ \Theta_s \; x(k) & \text{if } p(k) \in \mathcal{P}_s. \end{cases}$$

- $x(k)$ the regressor vector

$$x(k) = \begin{bmatrix} 1 & y'(k-1) & \cdots & y'(k-n_a) & u'(k-1) & \cdots & u'(k-n_b) \end{bmatrix}' \otimes \begin{bmatrix} 1 \\ p(k) \end{bmatrix}.$$

Problem: PWA regression

$$y(k) = \begin{cases} \Theta_1 \; x(k) & \text{if } p(k) \in \mathcal{P}_1, \\ \vdots & \\ \Theta_s \; x(k) & \text{if } p(k) \in \mathcal{P}_s. \end{cases}$$

Estimation of the PWA-LPV model consists of:

1. selecting the number of modes s
2. estimating the model parameter matrices Θ_i,
3. computing the polyhedra $\{\mathcal{P}_i\}_{i=1}^{s}$, defining partition of scheduling variable space

Problem: PWA regression

$$y(k) = \begin{cases} \Theta_1 \; x(k) & \text{if } p(k) \in \mathcal{P}_1, \\ \vdots & \\ \Theta_s \; x(k) & \text{if } p(k) \in \mathcal{P}_s. \end{cases}$$

Estimation of the PWA-LPV model consists of:

1. selecting the number of modes s
2. estimating the model parameter matrices Θ_i,
3. computing the polyhedra $\{\mathcal{P}_i\}_{i=1}^{s}$, defining partition of scheduling variable space

Problem: PWA regression

$$y(k) = \begin{cases} \Theta_1 \; x(k) & \text{if } p(k) \in \mathcal{P}_1, \\ \vdots & \\ \Theta_s \; x(k) & \text{if } p(k) \in \mathcal{P}_s. \end{cases}$$

Estimation of the PWA-LPV model consists of:

1. selecting the number of modes s
2. estimating the model parameter matrices Θ_i,
3. computing the polyhedra $\{\mathcal{P}_i\}_{i=1}^{s}$, defining partition of scheduling variable space

Problem: PWA regression

$$y(k) = \begin{cases} \Theta_1 \; x(k) & \text{if } p(k) \in \mathcal{P}_1, \\ \vdots & \\ \Theta_s \; x(k) & \text{if } p(k) \in \mathcal{P}_s. \end{cases}$$

Estimation of the PWA-LPV model consists of:

1. selecting the number of modes s
2. estimating the model parameter matrices Θ_i,
3. computing the polyhedra $\{\mathcal{P}_i\}_{i=1}^s$, defining partition of scheduling variable space

Problem: PWA regression

$$y(k) = \begin{cases} \Theta_1 \; x(k) & \text{if } p(k) \in \mathcal{P}_1, \\ \vdots & \\ \Theta_s \; x(k) & \text{if } p(k) \in \mathcal{P}_s. \end{cases}$$

Estimation of the PWA-LPV model consists of:

1. selecting the number of modes s (via cross-calibration)
2. estimating the model parameter matrices Θ_i,
3. computing the polyhedra $\{\mathcal{P}_i\}_{i=1}^{s}$, defining partition of scheduling variable space

PWA regression algorithm

The developed algorithm for PWA regression consists of the following two stages:

Stage S1.
- *estimation* of the model parameters Θ_i
- simultaneous *clustering* of the scheduling variables $\{p(k)\}_{k=1}^{N}$
- using regularized moving horizon regression algorithm

Stage S2.
- *Computation of polyhedral partitions* of the scheduling variable space $\mathcal{P}$
- using computationally efficient multi-category linear separation methods.

PWA regression algorithm

The developed algorithm for PWA regression consists of the following two stages:

Stage S1.
- *estimation* of the model parameters Θ_i
- simultaneous *clustering* of the scheduling variables $\{p(k)\}_{k=1}^{N}$
- using regularized moving horizon regression algorithm

Stage S2.
- *Computation of polyhedral partitions* of the scheduling variable space $\mathcal{P}$
- using computationally efficient multi-category linear separation methods.

PWA regression algorithm

The developed algorithm for PWA regression consists of the following two stages:

Stage S1.
- *estimation* of the model parameters Θ_i
- simultaneous *clustering* of the scheduling variables $\{p(k)\}_{k=1}^{N}$
- using regularized moving horizon regression algorithm

Stage S2.
- *Computation of polyhedral partitions* of the scheduling variable space $\mathcal{P}$
- using computationally efficient multi-category linear separation methods.

PWA regression algorithm

The developed algorithm for PWA regression consists of the following two stages:

Stage S1.
- *estimation* of the model parameters Θ_i
- simultaneous *clustering* of the scheduling variables $\{p(k)\}_{k=1}^{N}$
- using regularized moving horizon regression algorithm

Stage S2.
- *Computation of polyhedral partitions* of the scheduling variable space $\mathcal{P}$
- using computationally efficient multi-category linear separation methods.

PWA regression algorithm

The developed algorithm for PWA regression consists of the following two stages:

Stage S1.
- *estimation* of the model parameters Θ_i
- simultaneous *clustering* of the scheduling variables $\{p(k)\}_{k=1}^{N}$
- using regularized moving horizon regression algorithm

Stage S2.
- *Computation of polyhedral partitions* of the scheduling variable space $\mathcal{P}$
- using computationally efficient multi-category linear separation methods.

Stage S1: Regularized moving horizon identification algorithm

- At each time sample k, a moving-horizon window of length T containing training data samples $\{x(k), y(k), p(k)\}$ from time $k - T + 1$ to time k is considered.
- A *mixed-integer programming problem* is solved to simultaneously estimate:
  1. the model parameters Θ_i
  2. the active mode $\sigma(k) \in \{1, \dots, s\}$ which determines the polyhedral partition $\mathcal{P}_i$ in which $p(k)$ belongs to at time k, i.e., $\boxed{\sigma(k) = i^* \Rightarrow p(k) \in \mathcal{P}_{i^*}}$
- The training data samples $\{x(k), y(k), p(k)\}$ are processed iteratively by shifting the horizon window.

Stage S1: Regularized moving horizon identification algorithm

- At each time sample k, a moving-horizon window of length T containing training data samples $\{x(k), y(k), p(k)\}$ from time $k - T + 1$ to time k is considered.
- A *mixed-integer programming problem* is solved to simultaneously estimate:
  1. the model parameters Θ_i
  2. the active mode $\sigma(k) \in \{1, \ldots, s\}$ which determines the polyhedral partition $\mathcal{P}_i$ in which $p(k)$ belongs to at time k, i.e., $\boxed{\sigma(k) = i^* \Rightarrow p(k) \in \mathcal{P}_{i^*}}$
- The training data samples $\{x(k), y(k), p(k)\}$ are processed iteratively by shifting the horizon window.

Stage S1: Regularized moving horizon identification algorithm

- At each time sample k, a moving-horizon window of length T containing training data samples $\{x(k), y(k), p(k)\}$ from time $k - T + 1$ to time k is considered.
- A *mixed-integer programming problem* is solved to simultaneously estimate:
  1. the model parameters Θ_i
  2. the active mode $\sigma(k) \in \{1, \ldots, s\}$ which determines the polyhedral partition $\mathcal{P}_i$ in which $p(k)$ belongs to at time k, i.e., $\boxed{\sigma(k) = i^* \Rightarrow p(k) \in \mathcal{P}_{i^*}}$
- The training data samples $\{x(k), y(k), p(k)\}$ are processed iteratively by shifting the horizon window.

Stage S1: Regularized moving horizon identification algorithm

- At each time sample k, a moving-horizon window of length T containing training data samples $\{x(k), y(k), p(k)\}$ from time $k - T + 1$ to time k is considered.
- A *mixed-integer programming problem* is solved to simultaneously estimate:
  1. the model parameters Θ_i
  2. the active mode $\sigma(k) \in \{1, \ldots, s\}$ which determines the polyhedral partition $\mathcal{P}_i$ in which $p(k)$ belongs to at time k, i.e.,
     $\boxed{\sigma(k) = i^* \Rightarrow p(k) \in \mathcal{P}_{i^*}}$
- The training data samples $\{x(k), y(k), p(k)\}$ are processed iteratively by shifting the horizon window.

Stage S1: Regularized moving horizon identification algorithm

- At each time sample k, a moving-horizon window of length T containing training data samples $\{x(k), y(k), p(k)\}$ from time $k - T + 1$ to time k is considered.
- A *mixed-integer programming problem* is solved to simultaneously estimate:
  1. the model parameters Θ_i
  2. the active mode $\sigma(k) \in \{1, \ldots, s\}$ which determines the polyhedral partition $\mathcal{P}_i$ in which $p(k)$ belongs to at time k, i.e., $\boxed{\sigma(k) = i^* \Rightarrow p(k) \in \mathcal{P}_{i^*}}$
- The training data samples $\{x(k), y(k), p(k)\}$ are processed iteratively by shifting the horizon window.

Stage S1: Regularized moving horizon identification algorithm

- At each time sample k, a moving-horizon window of length T containing training data samples $\{x(k), y(k), p(k)\}$ from time $k-T+1$ to time k is considered.
- A *mixed-integer programming problem* is solved to simultaneously estimate:
  1. the model parameters Θ_i
  2. the active mode $\sigma(k) \in \{1, \ldots, s\}$ which determines the polyhedral partition $\mathcal{P}_i$ in which $p(k)$ belongs to at time k, i.e., $\boxed{\sigma(k) = i^* \Rightarrow p(k) \in \mathcal{P}_{i^*}}$
- The training data samples $\{x(k), y(k), p(k)\}$ are processed iteratively by shifting the horizon window.

Stage S1: Regularized moving horizon identification algorithm

At time k, we solve,

$$\min_{\Theta_i,\ \delta_i(k-t)} \sum_{i=1}^{s} \sum_{t=0}^{T-1} \left\| \left(y(k-t) - \Theta_i x(k-t)\right) \delta_i(k-t) \right\|^2$$

$$\text{s.t.}\ \ \delta_i(k-t) \in \{0,1\},\ \sum_{i=1}^{s} \delta_i(k-t) = 1,\ t = 0, \ldots, T-1.$$

- Fitting error term optimized over T horizon samples
- $\delta_i(k)$ indicates active mode at time k, i.e., $\boxed{\delta_i(k) = 1 \Rightarrow \sigma(k) = i \Rightarrow p(k) \in \mathcal{P}_i}$
- Computes both model parameters Θ_i and active modes $\sigma(k)$ at k simultaneously

Stage S1: Regularized moving horizon identification algorithm

At time k, we solve,

$$\min_{\Theta_i,\ \delta_i(k-t)} \sum_{i=1}^{s} \sum_{t=0}^{T-1} \|(y(k-t) - \Theta_i x(k-t))\, \delta_i(k-t)\|^2$$

$$\text{s.t.}\quad \delta_i(k-t) \in \{0,1\},\ \sum_{i=1}^{s} \delta_i(k-t) = 1,\ t = 0, \ldots, T-1.$$

- Fitting error term optimized over T horizon samples
- $\delta_i(k)$ indicates active mode at time k,
  i.e., $\boxed{\delta_i(k) = 1 \Rightarrow \sigma(k) = i \Rightarrow p(k) \in \mathcal{P}_i}$
- Computes both model parameters Θ_i and active modes $\sigma(k)$ at k simultaneously

Stage S1: Regularized moving horizon identification algorithm

At time k, we solve,

$$\min_{\Theta_i,\ \delta_i(k-t)} \sum_{i=1}^{s} \sum_{t=0}^{T-1} \|(y(k-t) - \Theta_i x(k-t))\, \delta_i(k-t)\|^2$$

$$\text{s.t.} \quad \delta_i(k-t) \in \{0,1\},\ \sum_{i=1}^{s} \delta_i(k-t) = 1,\ t = 0, \ldots, T-1.$$

- Fitting error term optimized over T horizon samples
- $\delta_i(k)$ indicates active mode at time k,
  i.e., $\boxed{\delta_i(k) = 1 \Rightarrow \sigma(k) = i \Rightarrow p(k) \in \mathcal{P}_i}$
- Computes both model parameters Θ_i and active modes $\sigma(k)$ at k simultaneously

Stage S1: Regularized moving horizon identification algorithm

At time k, we solve,

$$\min_{\Theta_i,\ \delta_i(k-t)} \sum_{i=1}^{s} \sum_{t=0}^{T-1} \|(y(k-t) - \Theta_i x(k-t))\,\delta_i(k-t)\|^2$$

$$\text{s.t.}\quad \delta_i(k-t) \in \{0,1\},\ \sum_{i=1}^{s} \delta_i(k-t) = 1,\ t = 0, \ldots, T-1.$$

- Fitting error term optimized over T horizon samples
- $\delta_i(k)$ indicates active mode at time k,
  i.e., $\boxed{\delta_i(k) = 1 \Rightarrow \sigma(k) = i \Rightarrow p(k) \in \mathcal{P}_i}$
- Computes both model parameters Θ_i and active modes $\sigma(k)$ at k simultaneously

Stage S1: Regularized moving horizon identification algorithm

$$\min_{\Theta_i,\ \delta_i(k-t)} \sum_{i=1}^{s} \sum_{t=0}^{T-1} \|(y(k-t) - \Theta_i x(k-t))\, \delta_i(k-t)\|^2$$

$$+ \boxed{\sum_{t=1}^{k-T} \left\| y(t) - \Theta_{\sigma(t)} x(t) \right\|^2} \quad \text{regularization on } \Theta_i$$

$$\text{s.t.} \quad \delta_i(k-t) \in \{0,1\}, \ \sum_{i=1}^{s} \delta_i(k-t) = 1, \quad t = 0, \ldots, T-1.$$

- takes into account the time history, outside the considered time window i.e. from time 1 to time $k-T$
- Fixed estimates $\{\sigma(t)\}_{t=1}^{k-T}$ are used, to refine the estimates of the model parameters Θ

Stage S1: Regularized moving horizon identification algorithm

$$\min_{\Theta_i,\ \delta_i(k-t)} \sum_{i=1}^{s} \sum_{t=0}^{T-1} \left\| \left(y(k-t) - \Theta_i x(k-t)\right) \delta_i(k-t) \right\|^2$$

$$+ \boxed{\sum_{t=1}^{k-T} \left\| y(t) - \Theta_{\sigma(t)} x(t) \right\|^2} \quad \text{regularization on } \Theta_i$$

$$\text{s.t.} \quad \delta_i(k-t) \in \{0,1\},\ \sum_{i=1}^{s} \delta_i(k-t) = 1, \quad t = 0, \ldots, T-1.$$

- takes into account the time history, outside the considered time window i.e. from time 1 to time $k-T$
- Fixed estimates $\{\sigma(t)\}_{t=1}^{k-T}$ are used, to refine the estimates of the model parameters Θ

Stage S1: Regularized moving horizon identification algorithm

$$\min_{\Theta_i,\ \delta_i(k-t)} \sum_{i=1}^{s} \sum_{t=0}^{T-1} \|(y(k-t) - \Theta_i x(k-t))\, \delta_i(k-t)\|^2$$

$$+ \boxed{\sum_{t=1}^{k-T} \left\|y(t) - \Theta_{\sigma(t)} x(t)\right\|^2} \quad \text{regularization on } \Theta_i$$

$$\text{s.t.} \quad \delta_i(k-t) \in \{0,1\},\ \sum_{i=1}^{s} \delta_i(k-t) = 1, \quad t = 0, \ldots, T-1.$$

- takes into account the time history, outside the considered time window i.e. from time 1 to time $k - T$
- Fixed estimates $\{\sigma(t)\}_{t=1}^{k-T}$ are used, to refine the estimates of the model parameters Θ

Stage S1: Regularized moving horizon identification algorithm

$$\min_{\Theta_i,\ \delta_i(k-t)} \sum_{i=1}^{s} \sum_{t=0}^{T-1} \|(y(k-t) - \Theta_i x(k-t))\,\delta_i(k-t)\|^2$$

$$+ \sum_{t=1}^{k-T} \left\|y(t) - \Theta_{\sigma(t)} x(t)\right\|^2 \quad \text{regularization on } \Theta_i$$

$$+ \boxed{\sum_{t=0}^{T-1} \sum_{i=1}^{s} \|(p(k-t) - c_i)\delta_i(k-t)\|^2} \quad \text{centroid distance}$$

$$\text{s.t.}\ \ \delta_i(k-t) \in \{0,1\},\ \sum_{i=1}^{s} \delta_i(k-t) = 1,\ t = 0, \ldots, T-1.$$

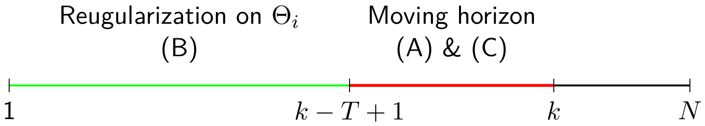
- Clustering: penalizes the error between $p(k)$ and centroids $\{c_i\}_{i=1}^{s}$ of clusters $\mathcal{P}_i$

Stage S1: Regularized moving horizon identification algorithm

$$
\begin{aligned}
\min_{\Theta_i,\ \delta_i(k-t)} \quad & \sum_{i=1}^{s} \sum_{t=0}^{T-1} \|(y(k-t) - \Theta_i x(k-t))\,\delta_i(k-t)\|^2 \\
& + \sum_{t=1}^{k-T} \left\|y(t) - \Theta_{\sigma(t)} x(t)\right\|^2 \quad \text{regularization on } \Theta_i \\
& + \boxed{\sum_{t=0}^{T-1} \sum_{i=1}^{s} \|(p(k-t) - c_i)\delta_i(k-t)\|^2} \quad \text{centroid distance} \\
\text{s.t.} \quad & \delta_i(k-t) \in \{0,1\},\ \sum_{i=1}^{s} \delta_i(k-t) = 1,\ t = 0, \ldots, T-1.
\end{aligned}
$$

- Clustering: penalizes the error between $p(k)$ and centroids $\{c_i\}_{i=1}^{s}$ of clusters $\mathcal{P}_i$

Stage S1: Regularized moving horizon identification algorithm



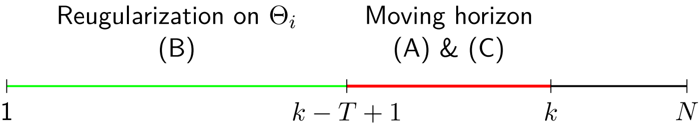
$$\min_{\Theta_i,\ \delta_i(k-t)} \sum_{i=1}^{s}\sum_{t=0}^{T-1} \left\|\left(y(k-t)-\Theta_i x(k-t)\right)\delta_i(k-t)\right\|^2 \qquad \text{(A)}$$

$$+\sum_{t=1}^{k-T} \left\|y(t)-\Theta_{\sigma(t)}x(t)\right\|^2 \qquad \text{(B)}$$

$$+\sum_{t=0}^{T-1}\sum_{i=1}^{s} \left\|(p(k-t)-c_i)\delta_i(k-t)\right\|^2 \qquad \text{(C)}$$

$$\text{s.t.}\ \ \delta_i(k-t)\in\{0,1\},\ \sum_{i=1}^{s}\delta_i(k-t)=1,\ t=0,\ldots,T-1.$$

Stage S1: Regularized moving horizon identification algorithm



Mixed-Integer Problem

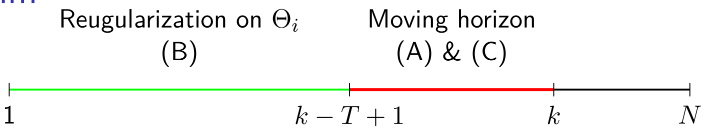
$$\min_{\Theta_i,\ \delta_i(k-t)} \sum_{i=1}^{s} \sum_{t=0}^{T-1} \left\|(y(k-t) - \Theta_i x(k-t))\, \delta_i(k-t)\right\|^2 \qquad \text{(A)}$$

$$+ \sum_{t=1}^{k-T} \left\|y(t) - \Theta_{\sigma(t)} x(t)\right\|^2 \qquad \text{(B)}$$

$$+ \sum_{t=0}^{T-1} \sum_{i=1}^{s} \left\|(p(k-t) - c_i)\delta_i(k-t)\right\|^2 \qquad \text{(C)}$$

$$\text{s.t.}\quad \delta_i(k-t) \in \{0,1\},\ \sum_{i=1}^{s} \delta_i(k-t) = 1,\ t = 0, \ldots, T-1.$$

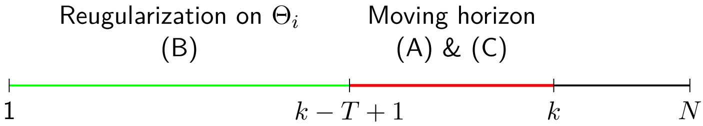
Stage S1: Regularized moving horizon identification algorithm



Mixed-Integer Problem

- Only the active mode $\sigma(k)$ at time k is kept
- the T-length time window is shifted forward to estimate the next mode $\sigma(k+1)$ solving the Mixed Integer Quadratic Programming (MIQP) problem.

Stage S1: Regularized moving horizon identification algorithm



Mixed-Integer Problem

- Only the active mode $\sigma(k)$ at time k is kept
- the T-length time window is shifted forward to estimate the next mode $\sigma(k+1)$ solving the Mixed Integer Quadratic Programming (MIQP) problem.

Iterative Refinement

- Initially for $k \ll N$, the clustering may be inaccurate
- It affects the evaluation of mode $\sigma(k)$
- As the regularization cost on Θ_i also depends on the estimated sequence $\{\sigma(t)\}_{t=1}^{k}$
- resulting estimates of model parameters Θ_i are inaccurate

Solution: To reduce the effect of initial classification error,

- We run the algorithm in Stage S1 multiple times (n_q) for iterative refinement by working in a batch mode

Iterative Refinement

- Initially for $k \ll N$, the clustering may be inaccurate
- It affects the evaluation of mode $\sigma(k)$
- As the regularization cost on Θ_i also depends on the estimated sequence $\{\sigma(t)\}_{t=1}^{k}$
- resulting estimates of model parameters Θ_i are inaccurate

Solution: To reduce the effect of initial classification error,

- We run the algorithm in Stage S1 multiple times (n_q) for iterative refinement by working in a batch mode

Iterative Refinement

- Initially for $k \ll N$, the clustering may be inaccurate
- It affects the evaluation of mode $\sigma(k)$
- As the regularization cost on Θ_i also depends on the estimated sequence $\{\sigma(t)\}_{t=1}^{k}$
- resulting estimates of model parameters Θ_i are inaccurate

Solution: To reduce the effect of initial classification error,

- We run the algorithm in Stage S1 multiple times (n_q) for iterative refinement by working in a batch mode

Iterative Refinement

- Initially for $k \ll N$, the clustering may be inaccurate
- It affects the evaluation of mode $\sigma(k)$
- As the regularization cost on Θ_i also depends on the estimated sequence $\{\sigma(t)\}_{t=1}^{k}$
- resulting estimates of model parameters Θ_i are inaccurate

Solution: To reduce the effect of initial classification error,

- We run the algorithm in Stage S1 multiple times (n_q) for iterative refinement by working in a batch mode

Iterative Refinement

- Initially for $k \ll N$, the clustering may be inaccurate
- It affects the evaluation of mode $\sigma(k)$
- As the regularization cost on Θ_i also depends on the estimated sequence $\{\sigma(t)\}_{t=1}^{k}$
- resulting estimates of model parameters Θ_i are inaccurate

Solution: To reduce the effect of initial classification error,

- We run the algorithm in Stage S1 multiple times (n_q) for iterative refinement by working in a batch mode

Iterative Refinement

- Initially for $k \ll N$, the clustering may be inaccurate
- It affects the evaluation of mode $\sigma(k)$
- As the regularization cost on Θ_i also depends on the estimated sequence $\{\sigma(t)\}_{t=1}^{k}$
- resulting estimates of model parameters Θ_i are inaccurate

Solution: To reduce the effect of initial classification error,

- We run the algorithm in Stage S1 multiple times (n_q) for iterative refinement by working in a batch mode

Iterative Refinement

- this is done by further adding a regularization term

$$\sum_{q=1}^{n_q-1} \lambda^{n_q-q-1} \sum_{t=1}^{N-T} \left\| y(t) - \Theta_{\sigma(t,q)} x(t) \right\|^2,$$

with $\sigma(t,q)$: estimate of the active mode at time t obtained at the q-th run of moving horizon algorithm

- A forgetting factor $\lambda \in \mathbb{R} : 0 < \lambda \leq 1$ is also included to exponentially downweight the estimates obtained at past runs
- For both the Regularization terms, a recursive update of the objective function is proposed to avoid the need to store the time history of the observations

Iterative Refinement

- this is done by further adding a regularization term

$$\sum_{q=1}^{n_q-1} \lambda^{n_q-q-1} \sum_{t=1}^{N-T} \left\| y(t) - \Theta_{\sigma(t,q)} x(t) \right\|^2,$$

  with $\sigma(t,q)$: estimate of the active mode at time t obtained at the q-th run of moving horizon algorithm
- A forgetting factor $\lambda \in \mathbb{R} : 0 < \lambda \leq 1$ is also included to exponentially downweight the estimates obtained at past runs
- For both the Regularization terms, a recursive update of the objective function is proposed to avoid the need to store the time history of the observations

Iterative Refinement

- this is done by further adding a regularization term

$$\sum_{q=1}^{n_q-1} \lambda^{n_q-q-1} \sum_{t=1}^{N-T} \left\| y(t) - \Theta_{\sigma(t,q)} x(t) \right\|^2,$$

  with $\sigma(t,q)$: estimate of the active mode at time t obtained at the q-th run of moving horizon algorithm
- A forgetting factor $\lambda \in \mathbb{R} : 0 < \lambda \leq 1$ is also included to exponentially downweight the estimates obtained at past runs
- For both the Regularization terms, a recursive update of the objective function is proposed to avoid the need to store the time history of the observations

Summary of Stage S1

- The model parameters Θ_i of PWA function have been estimated.
- Clustering of $\{p(k)\}_{k=1}^{N}$ into s clusters based on estimated mode sequence $\sigma(k)$ is obtained.
- Each cluster corresponds to a polyhedral partition $\mathcal{P}_i$.

- Next Step: Compute polyhedral partition of the clusters using Linear multicategory discrimination

Summary of Stage S1

- The model parameters Θ_i of PWA function have been estimated.
- Clustering of $\{p(k)\}_{k=1}^{N}$ into s clusters based on estimated mode sequence $\sigma(k)$ is obtained.
- Each cluster corresponds to a polyhedral partition $\mathcal{P}_i$.

- Next Step: Compute polyhedral partition of the clusters using Linear multicategory discrimination

Stage S2: Linear multicategory discrimination

Computation of a polyhedral partition $\mathcal{P}_i$ of the scheduling variable space $\mathcal{P}$ is done using the PWA separator function ϕ is defined as

$$\phi(p) = \max_{i=1,\ldots,s} \left(p'\omega^i - \gamma^i\right),$$

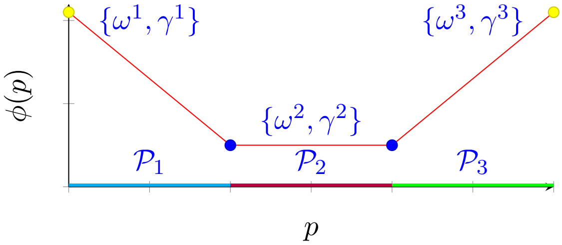
the polyhedra $\{\mathcal{P}_i\}_{i=1}^s$ are defined as

$$\mathcal{P}_i = \left\{p \in \mathbb{R}^{n_p} : [p' \quad -1]\left[\begin{smallmatrix}\omega^i - \omega^j \\ \gamma^i - \gamma^j\end{smallmatrix}\right] \geq 1,\ j = 1, \ldots, s,\ j \neq i\right\}.$$

Stage S2: Linear multicategory discrimination

Computation of a polyhedral partition $\mathcal{P}_i$ of the scheduling variable space $\mathcal{P}$ is done using the PWA separator function ϕ is defined as

$$\phi(p) = \max_{i=1,\ldots,s} \left(p'\omega^i - \gamma^i\right),$$



the polyhedra $\{\mathcal{P}_i\}_{i=1}^s$ are defined as

$$\mathcal{P}_i = \left\{p \in \mathbb{R}^{n_p} : \begin{bmatrix} p' & -1 \end{bmatrix} \begin{bmatrix} \omega^i - \omega^j \\ \gamma^i - \gamma^j \end{bmatrix} \geq 1,\ j = 1, \ldots, s,\ j \neq i\right\}.$$

Stage S2: Linear multicategory discrimination

The parameters $\{\omega^i, \gamma^i\}_{i=1}^s$ are calculated by solving the optimization problem which is convex, (Breschi, Piga and Bemporad, 2016)

$$\min_{\omega^i,\gamma^i} \frac{\kappa}{2} \sum_{i=1}^{s} \left(\|\omega^i\|_2^2 + (\gamma^i)^2 \right) +$$

$$\sum_{i=1}^{s} \sum_{\substack{j=1 \\ j \neq i}}^{s} \frac{1}{m_i} \left\| \left(\begin{bmatrix} M_i & -\mathbf{1}_{m_i} \end{bmatrix} \begin{bmatrix} \omega^j - \omega^i \\ \gamma^j - \gamma^i \end{bmatrix} + \mathbf{1}_{m_i} \right)_+ \right\|_2^2$$

Numerical Examples

Identification of SISO LPV-ARX system

Single-Input Single-Output (SISO) LPV-ARX system:

$$y(k) = a_1^{\mathrm{o}}(p(k))y(k-1)+a_2^{\mathrm{o}}(p(k))y(k-2)+b_1^{\mathrm{o}}(p(k))u(k-1)+e(k),$$

$$a_1^{\mathrm{o}}(p(k)) = \begin{cases} -0.5, & \text{if } p(k) > 0.5 \\ -p(k), & \text{if } -0.5 \leq p(k) \leq 0.5 \\ 0.5, & \text{if } p(k) < -0.5 \end{cases}$$

$$a_2^{\mathrm{o}}(p(k)) = p^3(k), \quad b_1^{\mathrm{o}}(p(k)) = \sin(\pi p(k)).$$

with $N = 6000$ training data, SNR on the output channel: 20 dB.

- A PWA model with $s = 6$ modes is considered.
- Stage S1 is run with prediction horizon $T = 6$.
- Stage S2 is executed with parameter $\kappa = 10^{-5}$

Identification of SISO LPV-ARX system

Single-Input Single-Output (SISO) LPV-ARX system:

$$y(k) = a_1^{\mathrm{o}}(p(k))y(k-1) + a_2^{\mathrm{o}}(p(k))y(k-2) + b_1^{\mathrm{o}}(p(k))u(k-1) + e(k),$$

$$a_1^{\mathrm{o}}(p(k)) = \begin{cases} -0.5, & \text{if } p(k) > 0.5 \\ -p(k), & \text{if } -0.5 \leq p(k) \leq 0.5 \\ 0.5, & \text{if } p(k) < -0.5 \end{cases}$$

$$a_2^{\mathrm{o}}(p(k)) = p^3(k), \quad b_1^{\mathrm{o}}(p(k)) = \sin(\pi p(k)).$$

with $N = 6000$ training data, SNR on the output channel: 20 dB.

- A PWA model with $s = 6$ modes is considered.
- Stage S1 is run with prediction horizon $T = 6$.
- Stage S2 is executed with parameter $\kappa = 10^{-5}$

Identification of SISO LPV-ARX system

Single-Input Single-Output (SISO) LPV-ARX system:

$$y(k) = a_1^{\mathrm{o}}(p(k))y(k-1)+a_2^{\mathrm{o}}(p(k))y(k-2)+b_1^{\mathrm{o}}(p(k))u(k-1)+e(k),$$

$$a_1^{\mathrm{o}}(p(k)) = \begin{cases} -0.5, & \text{if } p(k) > 0.5 \\ -p(k), & \text{if } -0.5 \leq p(k) \leq 0.5 \\ 0.5, & \text{if } p(k) < -0.5 \end{cases}$$

$$a_2^{\mathrm{o}}(p(k)) = p^3(k), \quad b_1^{\mathrm{o}}(p(k)) = \sin(\pi p(k)).$$

with $N = 6000$ training data, SNR on the output channel: 20 dB.

- A PWA model with $s = 6$ modes is considered.
- Stage S1 is run with prediction horizon $T = 6$.
- Stage S2 is executed with parameter $\kappa = 10^{-5}$

Identification of SISO LPV-ARX system

Single-Input Single-Output (SISO) LPV-ARX system:

$$y(k) = a_1^{o}(p(k))y(k-1) + a_2^{o}(p(k))y(k-2) + b_1^{o}(p(k))u(k-1) + e(k),$$

$$a_1^{o}(p(k)) = \begin{cases} -0.5, & \text{if } p(k) > 0.5 \\ -p(k), & \text{if } -0.5 \leq p(k) \leq 0.5 \\ 0.5, & \text{if } p(k) < -0.5 \end{cases}$$

$$a_2^{o}(p(k)) = p^3(k), \quad b_1^{o}(p(k)) = \sin(\pi p(k)).$$

with $N = 6000$ training data, SNR on the output channel: 20 dB.

- A PWA model with $s = 6$ modes is considered.
- Stage S1 is run with prediction horizon $T = 6$.
- Stage S2 is executed with parameter $\kappa = 10^{-5}$

Identification of SISO LPV-ARX system

- The resultant MIQP: 36 binary and 72 continuous variables, 144 inequality and 6 equality constraints.
- The mean time to solve: GUROBI (commercial) 0.09 sec, GPAD-B&B [1] 0.13 sec
- GPAD-B&B is simple library-free solver, yet rendered comparable performance w.r.t GUROBI

Best Fit Rate

- BFR on noise-free validation dataset $N_{\mathrm{val}} = 2000$ is 0.95

[1] accelerated dual gradient projection based branch and bound - (Naik and Bemporad, 2017)

Identification of SISO LPV-ARX system

- The resultant MIQP: 36 binary and 72 continuous variables, 144 inequality and 6 equality constraints.
- The mean time to solve: GUROBI (commercial) 0.09 sec, GPAD-B&B [1] 0.13 sec
- GPAD-B&B is simple library-free solver, yet rendered comparable performance w.r.t GUROBI

Best Fit Rate

- BFR on noise-free validation dataset $N_{\text{val}} = 2000$ is 0.95

[1] accelerated dual gradient projection based branch and bound - (Naik and Bemporad, 2017)

Identification of SISO LPV-ARX system

- The resultant MIQP: 36 binary and 72 continuous variables, 144 inequality and 6 equality constraints.
- The mean time to solve: GUROBI (commercial) 0.09 sec, GPAD-B&B [1] 0.13 sec
- GPAD-B&B is simple library-free solver, yet rendered comparable performance w.r.t GUROBI

Best Fit Rate

- BFR on noise-free validation dataset $N_{\text{val}} = 2000$ is 0.95

[1] accelerated dual gradient projection based branch and bound - (Naik and Bemporad, 2017)

Identification of SISO LPV-ARX system

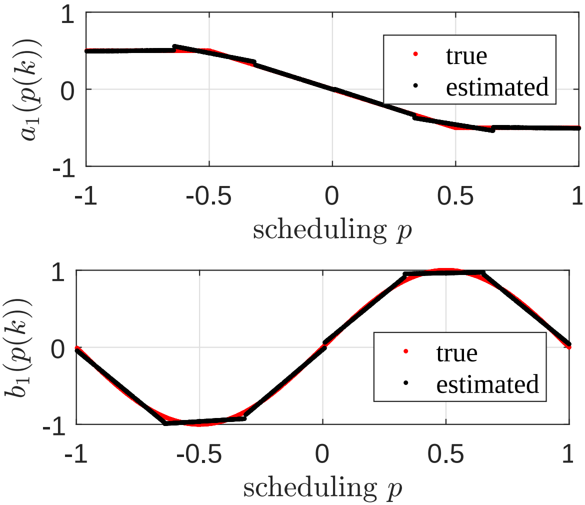
- The resultant MIQP: 36 binary and 72 continuous variables, 144 inequality and 6 equality constraints.
- The mean time to solve: GUROBI (commercial) 0.09 sec, GPAD-B&B [1] 0.13 sec
- GPAD-B&B is simple library-free solver, yet rendered comparable performance w.r.t GUROBI

Best Fit Rate

- BFR on noise-free validation dataset $N_{\text{val}} = 2000$ is 0.95

[1] accelerated dual gradient projection based branch and bound - (Naik and Bemporad, 2017)

Estimated LPV coefficient functions



Identification of MIMO LPV-ARX system

Multi-Input Multi-Output (SISO) LPV-ARX system:

$$\begin{bmatrix} y_1(k) \\ y_2(k) \end{bmatrix} = \begin{bmatrix} \bar{a}_{1,1}(p(k)) & \bar{a}_{1,2}(p(k)) \\ \bar{a}_{2,1}(p(k)) & \bar{a}_{2,2}(p(k)) \end{bmatrix} \begin{bmatrix} y_1(k-1) \\ y_2(k-1) \end{bmatrix} + \begin{bmatrix} \bar{b}_{1,1}(p(k)) & \bar{b}_{1,2}(p(k)) \\ \bar{b}_{2,1}(p(k)) & \bar{b}_{2,2}(p(k)) \end{bmatrix} \begin{bmatrix} u_1(k-1) \\ u_2(k-1) \end{bmatrix} + e(k),$$

with $N = 6000$ training data samples,
SNR on the output channels: $\mathrm{SNR}_1 = 23$ dB, $\mathrm{SNR}_2 = 23.5$ dB.

- A PWA model with $s = 10$ modes is considered.
- Algorithm S1 is run with prediction horizon $T = 6$, $n_\mathrm{q} = 2$ iterations.
- Algorithm S2 is executed with parameter $\kappa = 10^{-10}$

Identification of MIMO LPV-ARX system

Multi-Input Multi-Output (SISO) LPV-ARX system:

$$\begin{bmatrix} y_1(k) \\ y_2(k) \end{bmatrix} = \begin{bmatrix} \bar{a}_{1,1}(p(k)) & \bar{a}_{1,2}(p(k)) \\ \bar{a}_{2,1}(p(k)) & \bar{a}_{2,2}(p(k)) \end{bmatrix} \begin{bmatrix} y_1(k-1) \\ y_2(k-1) \end{bmatrix} + \begin{bmatrix} \bar{b}_{1,1}(p(k)) & \bar{b}_{1,2}(p(k)) \\ \bar{b}_{2,1}(p(k)) & \bar{b}_{2,2}(p(k)) \end{bmatrix} \begin{bmatrix} u_1(k-1) \\ u_2(k-1) \end{bmatrix} + e(k),$$

with $N = 6000$ training data samples,
SNR on the output channels: $\mathrm{SNR}_1 = 23$ dB, $\mathrm{SNR}_2 = 23.5$ dB.

- A PWA model with $s = 10$ modes is considered.
- Algorithm S1 is run with prediction horizon $T = 6$, $n_{\mathrm{q}} = 2$ iterations.
- Algorithm S2 is executed with parameter $\kappa = 10^{-10}$

Identification of MIMO LPV-ARX system

Multi-Input Multi-Output (SISO) LPV-ARX system:

$$\begin{bmatrix} y_1(k) \\ y_2(k) \end{bmatrix} = \begin{bmatrix} \bar{a}_{1,1}(p(k)) & \bar{a}_{1,2}(p(k)) \\ \bar{a}_{2,1}(p(k)) & \bar{a}_{2,2}(p(k)) \end{bmatrix} \begin{bmatrix} y_1(k-1) \\ y_2(k-1) \end{bmatrix} + \begin{bmatrix} \bar{b}_{1,1}(p(k)) & \bar{b}_{1,2}(p(k)) \\ \bar{b}_{2,1}(p(k)) & \bar{b}_{2,2}(p(k)) \end{bmatrix} \begin{bmatrix} u_1(k-1) \\ u_2(k-1) \end{bmatrix} + e(k),$$

with $N = 6000$ training data samples,
SNR on the output channels: $\mathrm{SNR}_1 = 23$ dB, $\mathrm{SNR}_2 = 23.5$ dB.

- A PWA model with $s = 10$ modes is considered.
- Algorithm S1 is run with prediction horizon $T = 6$, $n_{\mathrm{q}} = 2$ iterations.
- Algorithm S2 is executed with parameter $\kappa = 10^{-10}$

Identification of MIMO LPV-ARX system

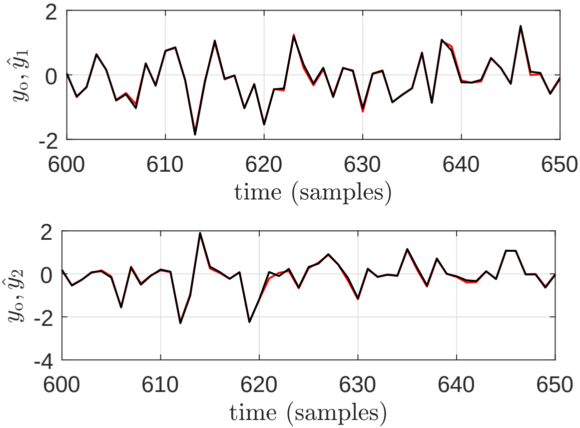
Multi-Input Multi-Output (SISO) LPV-ARX system:

$$\begin{bmatrix} y_1(k) \\ y_2(k) \end{bmatrix} = \begin{bmatrix} \bar{a}_{1,1}(p(k)) & \bar{a}_{1,2}(p(k)) \\ \bar{a}_{2,1}(p(k)) & \bar{a}_{2,2}(p(k)) \end{bmatrix} \begin{bmatrix} y_1(k-1) \\ y_2(k-1) \end{bmatrix} + \begin{bmatrix} \bar{b}_{1,1}(p(k)) & \bar{b}_{1,2}(p(k)) \\ \bar{b}_{2,1}(p(k)) & \bar{b}_{2,2}(p(k)) \end{bmatrix} \begin{bmatrix} u_1(k-1) \\ u_2(k-1) \end{bmatrix} + e(k),$$

with $N = 6000$ training data samples,
SNR on the output channels: $\mathrm{SNR}_1 = 23$ dB, $\mathrm{SNR}_2 = 23.5$ dB.

- A PWA model with $s = 10$ modes is considered.
- Algorithm S1 is run with prediction horizon $T = 6$, $n_{\mathrm{q}} = 2$ iterations.
- Algorithm S2 is executed with parameter $\kappa = 10^{-10}$

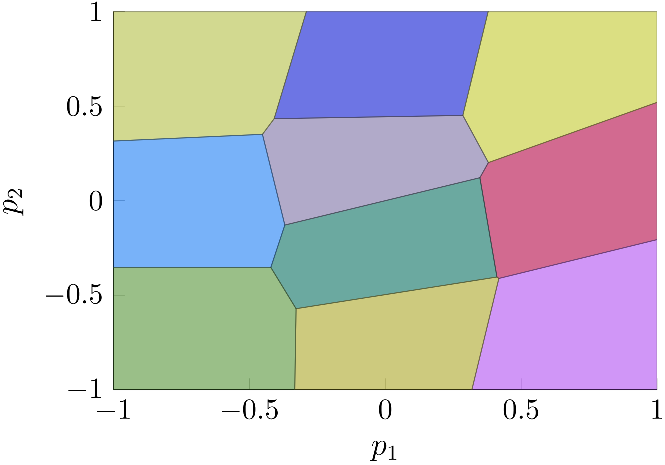
Identification of MIMO LPV-ARX system



Best Fit Rates

runs(n_q)	BFR y_1	BFR y_2
1	0.8793	0.8108
2	0.8817	0.8146

Scheduling space partition



Conclusions

- LPV identification is recast as *PieceWise Affine* (PWA) regression problem.
- A novel moving-horizon algorithm for PWA regression has been proposed.
- Simultaneous estimation of the model parameters and of the optimal sequence of active modes is achived.
- LPV systems can be modeled with arbitrary accuracy by the choice of number of PWA modes.

Conclusions

- LPV identification is recast as *PieceWise Affine* (PWA) regression problem.
- A novel moving-horizon algorithm for PWA regression has been proposed.
- Simultaneous estimation of the model parameters and of the optimal sequence of active modes is achived.
- LPV systems can be modeled with arbitrary accuracy by the choice of number of PWA modes.

Conclusions

- LPV identification is recast as *PieceWise Affine* (PWA) regression problem.
- A novel moving-horizon algorithm for PWA regression has been proposed.
- Simultaneous estimation of the model parameters and of the optimal sequence of active modes is achived.
- LPV systems can be modeled with arbitrary accuracy by the choice of number of PWA modes.

Conclusions

- LPV identification is recast as *PieceWise Affine* (PWA) regression problem.
- A novel moving-horizon algorithm for PWA regression has been proposed.
- Simultaneous estimation of the model parameters and of the optimal sequence of active modes is achived.
- LPV systems can be modeled with arbitrary accuracy by the choice of number of PWA modes.

Acknowledgment

The authors would like to thank Valentina Breschi for her help with the multicategory discrimination algorithm.

Thank You