ALGORITHMS FOR EXACT AND APPROXIMATE IDENTICATION FROM FINITE TIME SERIES

Jan C. Willems

K.U. Leuven

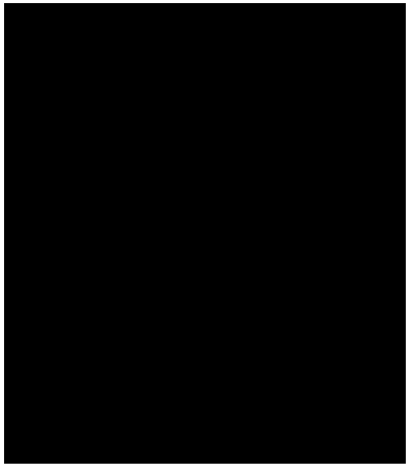
Kyoto University **May 17, 2005**

On-going joint research with

Ivan Markovsky (K.U. Leuven)

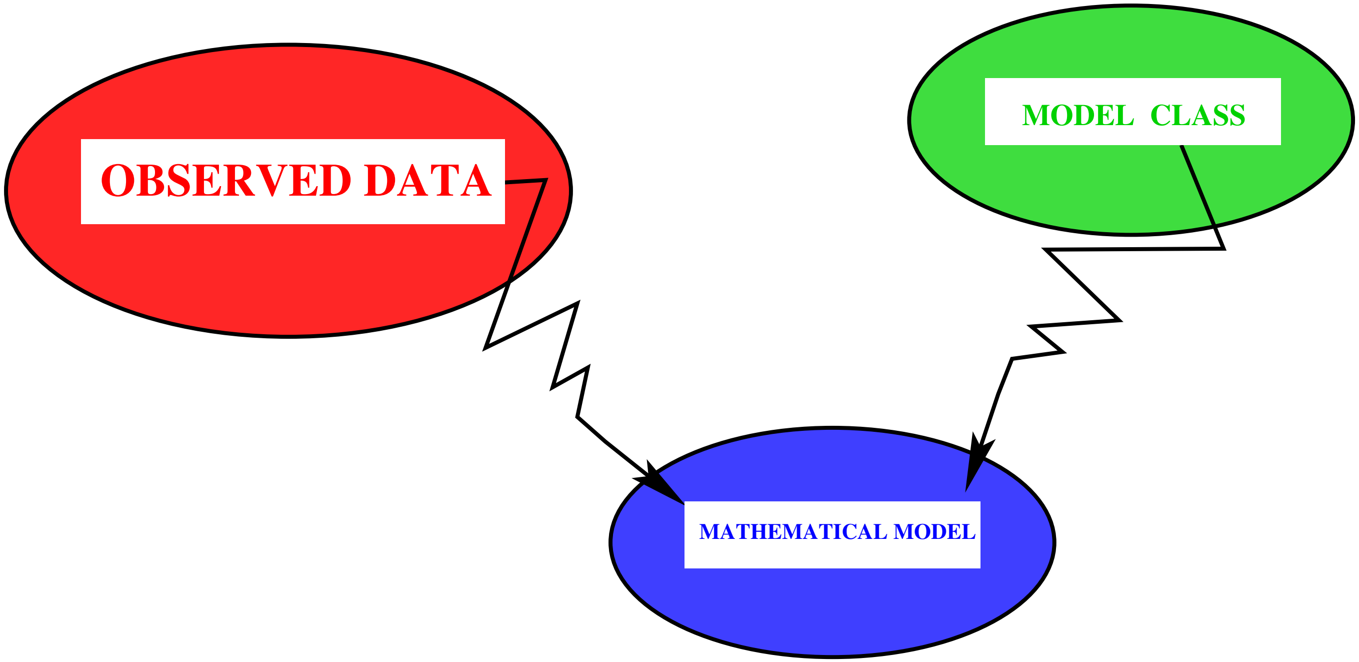
Paolo Rapisarda (Un. Maastricht)

& Bart De Moor (K.U. Leuven)



System ID

SYSID



SYSID

This is a very rich area. It involves

- **Algorithms:**

  Numerical data $\longmapsto$ model parameters

- **'Philosophical' issues:**

  How to deal with uncertainty

  Role of stochasticity

  How to deal with 'open' systems, etc.

- **Important area for applications, because of its**

  relevance in modeling

Case of interest today

Data: an 'observed' vector time-series

$$\tilde{w}(1), \tilde{w}(2), \ldots, \tilde{w}(T) \qquad \tilde{w}(t) \in \mathbb{R}^{\mathtt{w}}, T \text{ finite}$$

$$\Downarrow$$

A **dynamical model** from a model class,

e.g. a difference equation

$$R_0 w(t) + R_1 w(t+1) + \cdots + R_L w(t+L)$$

$$= 0$$

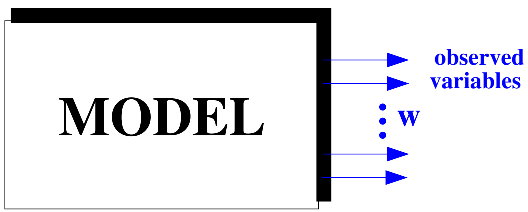
or

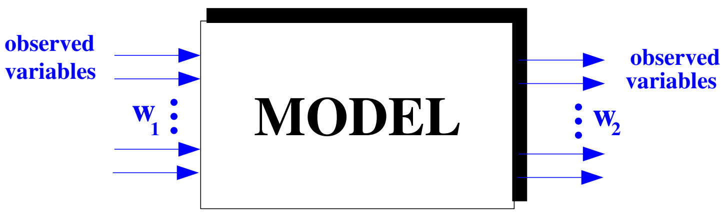
$$= M_0 \varepsilon(t) + M_1 \varepsilon(t+1) + \cdots + M_L \varepsilon(t+L)$$

Case of interest today

We discuss mainly the case:

'deterministic' ID

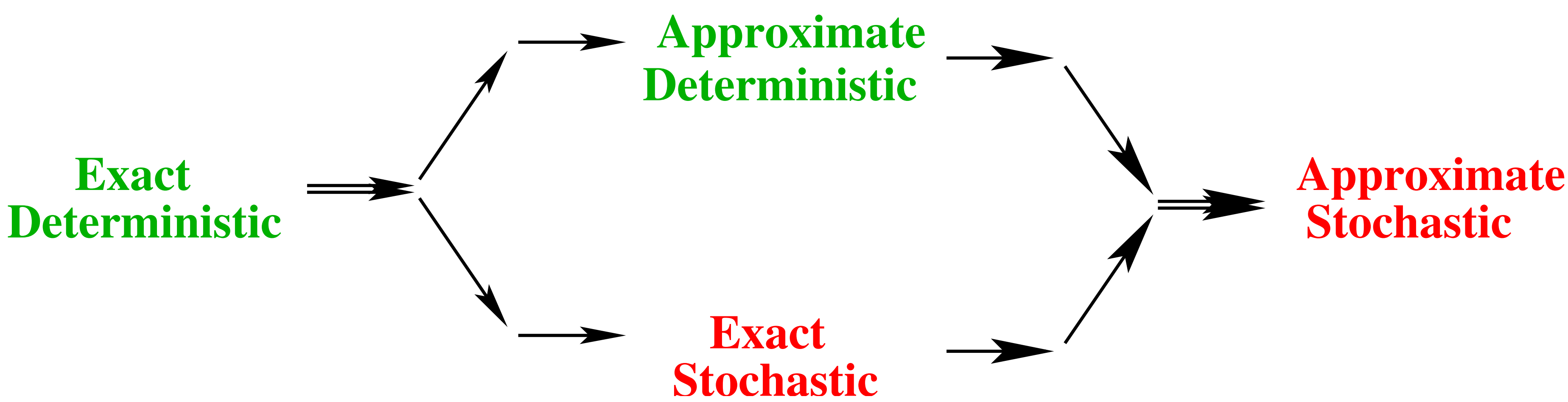




$$R_0 w(t) + R_1 w(t+1) + \cdots + R_L w(t+L) = 0$$

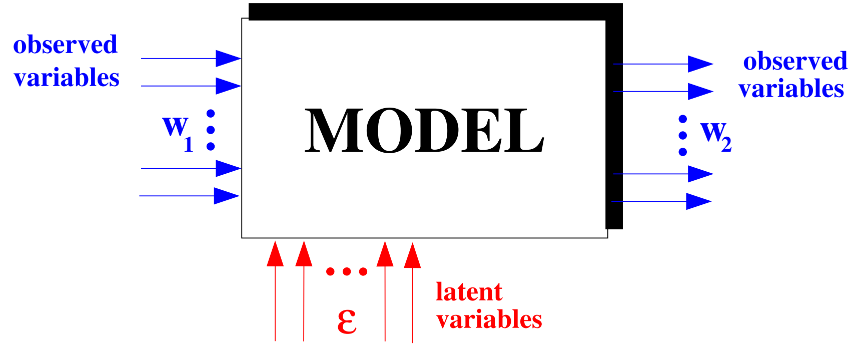
$$\tilde{w}(1), \tilde{w}(2), \ldots, \tilde{w}(T) \mapsto \hat{R}(\xi) = \hat{R}_0 + \hat{R}_1 \xi + \cdots + \hat{R}_{\hat{L}} \xi^{\hat{L}}$$

Case of interest today



Case of interest today

Towards the end, some remarks on ID with latent inputs



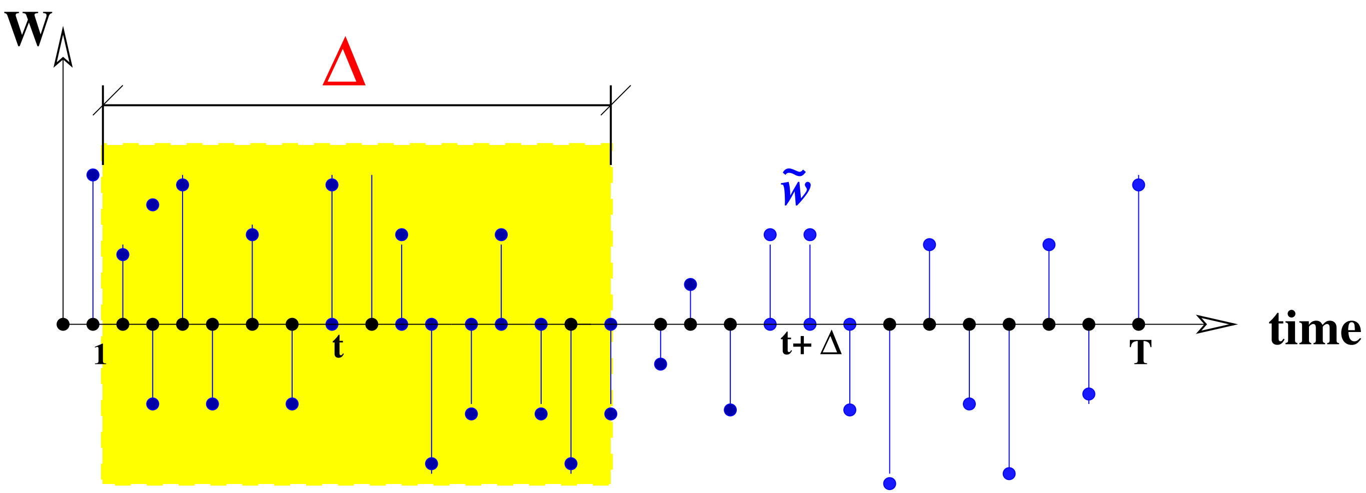
$$R_0 w(t) + R_1 w(t+1) + \cdots + R_L w(t+L)$$
$$= M_0 \varepsilon(t) + M_1 \varepsilon(t+1) + \cdots + M_L \varepsilon(t+L)$$

$$\tilde{w}(1), \tilde{w}(2), \ldots, \tilde{w}(T) \mapsto \left(\hat{R}(\xi), \hat{M}(\xi)\right)$$

$\tilde{w} \mapsto R$

$$\tilde{w}(1), \tilde{w}(2), \ldots, \tilde{w}(T)$$

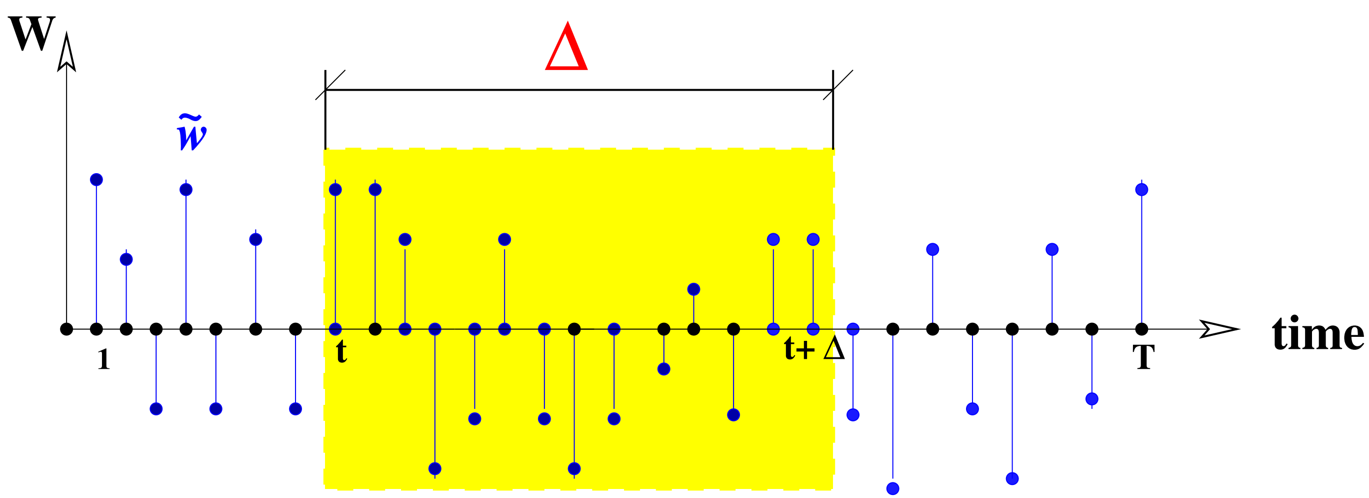
Basic idea: look through the window (with $\Delta > L$) in order to discover the system laws.



$\tilde{w} \mapsto R$

$$\tilde{w}(1), \tilde{w}(2), \ldots, \tilde{w}(T)$$

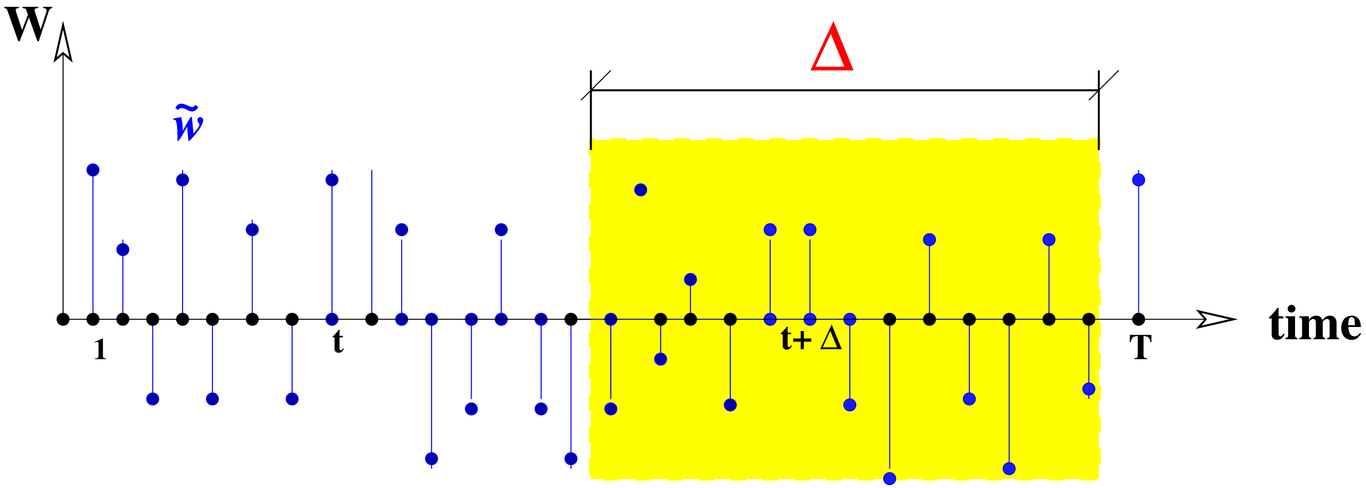
Basic idea: look through the window (with $\Delta > L$) in order to discover the system laws.



$\tilde{w} \mapsto R$

$$\tilde{w}(1), \tilde{w}(2), \ldots, \tilde{w}(T)$$

<u>Basic idea</u>: look through the window (with $\Delta > L$) in order to discover the system laws.



Is there a recursion, same for all these windows?

$\tilde{w} \mapsto R$

$$\tilde{w}(1), \tilde{w}(2), \ldots, \tilde{w}(T)$$

<u>Basic idea</u>: look through the window (with $\Delta > L$) in order to discover the system laws.

The windows lead linea recta to the Hankel matrix

$$\begin{bmatrix} \tilde{w}(1) & \tilde{w}(2) & \cdots & \tilde{w}(t) & \cdots & \tilde{w}(T-\Delta) \\ \tilde{w}(2) & \tilde{w}(3) & \cdots & \tilde{w}(t+1) & \cdots & \tilde{w}(T-\Delta+1) \\ \tilde{w}(3) & \tilde{w}(4) & \cdots & \tilde{w}(t+2) & \cdots & \tilde{w}(T-\Delta+2) \\ \vdots & \vdots & \vdots & \vdots & & \\ \tilde{w}(\Delta+1) & \tilde{w}(\Delta+2) & \cdots & \tilde{w}(t+\Delta) & \cdots & \tilde{w}(T) \end{bmatrix}$$

$\tilde{w} \mapsto R$

$$\tilde{w}(1), \tilde{w}(2), \ldots, \tilde{w}(T)$$

Basic idea: look through the window (with $\Delta > L$) in order to discover the system laws.

The windows lead linea recta to the Hankel matrix

$$\begin{bmatrix} \tilde{w}(1) & \tilde{w}(2) & \cdots & \tilde{w}(t) & \cdots & \tilde{w}(t-\Delta) \\ \tilde{w}(2) & \tilde{w}(3) & \cdots & \tilde{w}(t+1) & \cdots & \tilde{w}(t-\Delta+1) \\ \tilde{w}(3) & \tilde{w}(4) & \cdots & \tilde{w}(t+2) & \cdots & \tilde{w}(t-\Delta+2) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \tilde{w}(\Delta+1) & \tilde{w}(\Delta+2) & \cdots & \tilde{w}(t+\Delta) & \cdots & \tilde{w}(T) \end{bmatrix}$$

$\tilde{w} \mapsto R$

$$\tilde{w}(1), \tilde{w}(2), \ldots, \tilde{w}(T)$$

Basic idea: look through the window (with $\Delta > L$) in order to discover the system laws.

The windows lead linea recta to the Hankel matrix

$$\begin{bmatrix} \tilde{w}(1) & \tilde{w}(2) & \cdots & \tilde{w}(t) & \cdots & \tilde{w}(t-\Delta) \\ \tilde{w}(2) & \tilde{w}(3) & \cdots & \tilde{w}(t+1) & \cdots & \tilde{w}(t-\Delta+1) \\ \tilde{w}(3) & \tilde{w}(4) & \cdots & \tilde{w}(t+2) & \cdots & \tilde{w}(t-\Delta+2) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \tilde{w}(\Delta+1) & \tilde{w}(\Delta+2) & \cdots & \tilde{w}(t+\Delta) & \cdots & \tilde{w}(T) \end{bmatrix}$$

$\tilde{w} \mapsto R$

$$\tilde{w}(1), \tilde{w}(2), \ldots, \tilde{w}(T)$$

<u>Basic idea</u>: look through the window (with $\Delta > L$) in order to discover the system laws.

The windows lead linea recta to the Hankel matrix

$$\begin{bmatrix} \tilde{w}(1) & \tilde{w}(2) & \cdots & \tilde{w}(t) & \cdots & \tilde{w}(t-\Delta) \\ \tilde{w}(2) & \tilde{w}(3) & \cdots & \tilde{w}(t+1) & \cdots & \tilde{w}(t-\Delta+1) \\ \tilde{w}(3) & \tilde{w}(4) & \cdots & \tilde{w}(t+2) & \cdots & \tilde{w}(t-\Delta+2) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \tilde{w}(\Delta+1) & \tilde{w}(\Delta+2) & \cdots & \tilde{w}(t+\Delta) & \cdots & \tilde{w}(T) \end{bmatrix}$$

Are there left annihilitors, or approximate, or up to a stochastic interpretation, same for all these columns?

$\tilde{w} \mapsto R$

$$\tilde{w}(1), \tilde{w}(2), \ldots, \tilde{w}(T)$$

Basic idea: look through the window (with $\Delta > L$) in order to discover the system laws.

But first, some language: What do we mean by

a model, a model class, an unfalsified model, etc.?

The MPUM

The MPUM

- A **model**:= a subset $\mathfrak{B} \subseteq (\mathbb{R}^{\mathtt{w}})^{\mathbb{N}}$, the **'behavior'**

  A family of (vector) time series

  Recall notation $\mathfrak{B}_{|[1,T]}$

  := all 'prefixes' $w(1), w(2), \cdots, w(T)$ of $w \in \mathfrak{B}$

The MPUM

- A **model** := a subset $\mathfrak{B} \subseteq (\mathbb{R}^{\mathtt{w}})^{\mathbb{N}}$, the **'behavior'**
- $\mathfrak{B}$ is **unfalsified by** $\tilde{w}$:= $\tilde{w}(1), \tilde{w}(2), \ldots, \tilde{w}(T)$

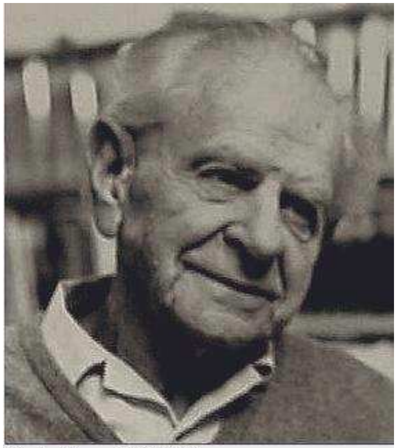
  $:\Leftrightarrow \tilde{w} \in \mathfrak{B}_{|[0,t]}$

The MPUM

- A model:= a subset $\mathfrak{B} \subseteq (\mathbb{R}^{\mathtt{w}})^{\mathbb{N}}$, the 'behavior'
- $\mathfrak{B}$ is unfalsified by $\tilde{w}$ $:\Leftrightarrow \tilde{w} \in \mathfrak{B}_{|[0,t]}$
- $\mathfrak{B}_1$ is more powerful than $\mathfrak{B}_2$ $:\Leftrightarrow \mathfrak{B}_1 \subseteq \mathfrak{B}_2$

Every model is prohibition.

The more a model forbids, the better it is.



Sir Karl Popper (1902-1994)

Karl Popper
(1902-1994)

The MPUM

- A **model**:= a subset $\mathfrak{B} \subseteq (\mathbb{R}^{\mathtt{w}})^{\mathbb{N}}$, the **'behavior'**
- $\mathfrak{B}$ is **unfalsified by** $\tilde{w}$ $\quad :\Leftrightarrow \tilde{w} \in \mathfrak{B}_{|[0,t]}$
- $\mathfrak{B}_1$ **is more powerful than** $\mathfrak{B}_2$ $:\Leftrightarrow \mathfrak{B}_1 \subseteq \mathfrak{B}_2$
- A **model class:** a family, $\mathbb{B}$, of models

The MPUM

- A model:= a subset $\mathfrak{B} \subseteq (\mathbb{R}^{\mathtt{w}})^{\mathbb{N}}$, the 'behavior'
- $\mathfrak{B}$ is unfalsified by $\tilde{w}$ $\quad :\Leftrightarrow \tilde{w} \in \mathfrak{B}_{|[0,t]}$
- $\mathfrak{B}_1$ is more powerful than $\mathfrak{B}_2$ $:\Leftrightarrow \mathfrak{B}_1 \subseteq \mathfrak{B}_2$
- A model class: a family, $\mathbb{B}$, of models
- The MPUM 'most powerful unfalsified model'

  in $\mathbb{B}$ for $\tilde{w}$, denoted $\mathfrak{B}^*_{\tilde{w}}$:

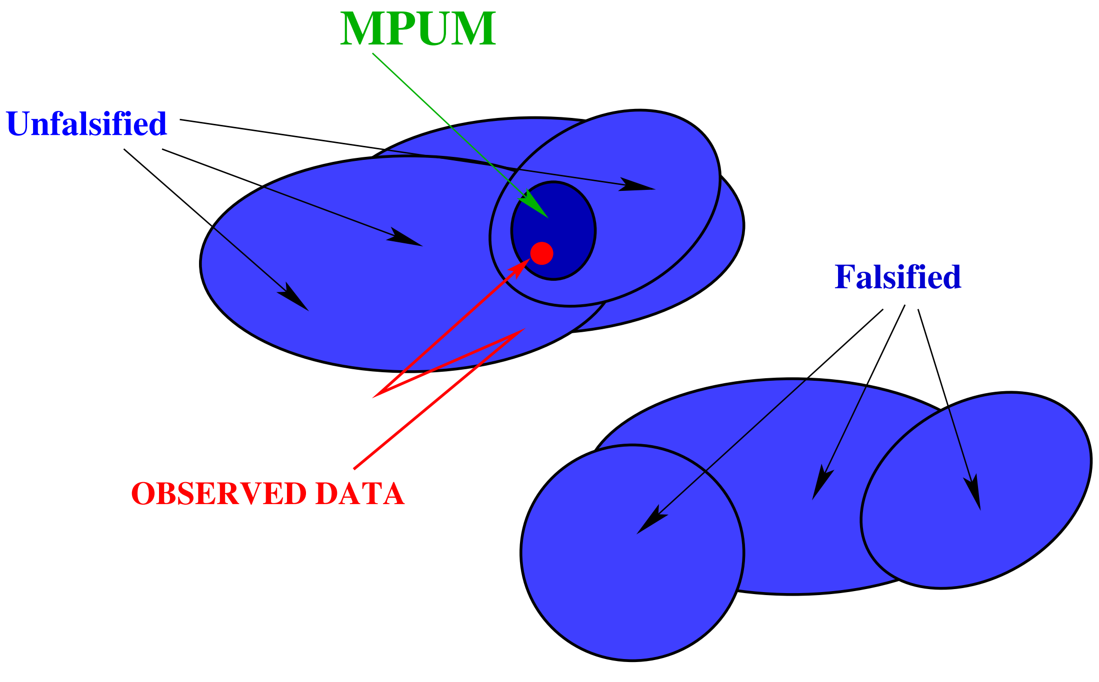
  1. $\mathfrak{B}^*_{\tilde{w}} \in \mathbb{B}$
  2. $\tilde{w} \in \mathfrak{B}^*_{\tilde{w}}{}_{|[1,T]}$
  3. $\mathfrak{B} \in \mathbb{B}$ and $\tilde{w} \in \mathfrak{B}_{|[1,T]} \Rightarrow \mathfrak{B}^*_{\tilde{w}} \subseteq \mathfrak{B}$

The MPUM

- A model:= a subset $\mathfrak{B} \subseteq \left(\mathbb{R}^{\mathtt{w}}\right)^{\mathbb{N}}$, the 'behavior'
- $\mathfrak{B}$ is unfalsified by $\tilde{w}$ $\quad :\Leftrightarrow \tilde{w} \in \mathfrak{B}_{|[0,t]}$
- $\mathfrak{B}_1$ is more powerful than $\mathfrak{B}_2$ $:\Leftrightarrow \mathfrak{B}_1 \subseteq \mathfrak{B}_2$
- A model class: a family, $\mathbb{B}$, of models
- The MPUM 'most powerful unfalsified model'

  in $\mathbb{B}$ for $\tilde{w}$, denoted $\mathfrak{B}^*_{\tilde{w}}$
- Given $\tilde{w}$ and $\mathbb{B}$, does $\mathfrak{B}^*_{\tilde{w}}$ exist?

The MPUM



The model class

The model class $\mathfrak{L}^{\mathtt{W}}$

Our model class (a family of subsets of $(\mathbb{R}^{\mathtt{W}})^{\mathbb{N}}$).

It is an exceedingly familiar one. First, $\mathfrak{L}^{\mathtt{W}}$.

$\mathfrak{B} \subseteq (\mathbb{R}^{\mathtt{W}})^{\mathbb{N}}$ belongs to $\mathfrak{L}^{\mathtt{W}}$ $:\Leftrightarrow$

The model class $\mathfrak{L}^{\mathtt{w}}$

$\mathfrak{B} \subseteq (\mathbb{R}^{\mathtt{w}})^{\mathbb{N}}$ belongs to $\mathfrak{L}^{\mathtt{w}}$ $:\Leftrightarrow$

- $\mathfrak{B}$ is linear, shift-invariant, and closed

shift-invariant $:\Leftrightarrow$ $\sigma\mathfrak{B} \subseteq \mathfrak{B}$

σ = the 'shift': $(\sigma f)(t) := f(t+1)$.

The model class $\mathfrak{L}^{\mathtt{w}}$

$\mathfrak{B} \subseteq (\mathbb{R}^{\mathtt{w}})^{\mathbb{N}}$ **belongs to** $\mathfrak{L}^{\mathtt{w}}$ **:**$\Leftrightarrow$

- $\mathfrak{B}$ **is linear, shift-invariant, and closed**
- $\exists$ **matrices** $R_0, R_1, \ldots, R_L$ **such that** $\mathfrak{B}$ **consists of all** w **that satisfy**

$$R_0 w(t) + R_1 w(t+1) + \cdots + R_L w(t+L) = 0$$

In obvious polynomial matrix notation

$$\boxed{R(\sigma)w = 0}$$

The model class $\mathfrak{L}^{\mathtt{w}}$

$\mathfrak{B} \subseteq (\mathbb{R}^{\mathtt{w}})^{\mathbb{N}}$ belongs to $\mathfrak{L}^{\mathtt{w}}$ $:\Leftrightarrow$

- $\mathfrak{B}$ is linear, shift-invariant, and closed
-

$$R(\sigma)w = 0$$

- Including input/output partition

$$P(\sigma)y = Q(\sigma)u, \quad w \cong \left[\begin{smallmatrix} u \\ y \end{smallmatrix}\right]$$

$\det(P) \neq 0$, $\mathtt{m}$ inputs, $\mathtt{p}$ outputs (= # of equations)

The model class $\mathfrak{L}^{\mathtt{w}}$

$\mathfrak{B} \subseteq (\mathbb{R}^{\mathtt{w}})^{\mathbb{N}}$ **belongs to** $\mathfrak{L}^{\mathtt{w}}$ $:\Leftrightarrow$

- $\mathfrak{B}$ **is linear, shift-invariant, and closed**

-

$$R(\sigma)w = 0$$

-

$$P(\sigma)y = Q(\sigma)u, \quad w \cong \begin{bmatrix} u \\ y \end{bmatrix}$$

- $\exists$ **matrices** A, B, C, D **such that**
  $\mathfrak{B}$ **consists of all** $w's$ **generated by**
  $\sigma x = Ax + Bu,\ y = Cx + Du, \quad w \cong \begin{bmatrix} u \\ y \end{bmatrix}$

The module structure

Let $\mathfrak{B} \in \mathfrak{L}^{\mathtt{w}}$. Define its annihilators by

$$\mathfrak{N}_{\mathfrak{B}} := \{n \in \mathbb{R}^{\mathtt{w}}[\xi] \mid n^\top(\frac{d}{dt})\mathfrak{B} = 0\}$$

Note: $\mathfrak{N}_{\mathfrak{B}}$ is a $\mathbb{R}[\xi]$ sub-module of $\mathbb{R}^{\mathtt{w}}[\xi]$. Means:

$$n_1, n_2 \in \mathbb{R}^{\mathtt{w}}[\xi], p \in \mathbb{R}[\xi]$$

$$\Rightarrow \quad n_1 + n_2 \in \mathfrak{N}_{\mathfrak{B}}, p\, n_1 \in \mathfrak{N}_{\mathfrak{B}}$$

The module structure

Let $\mathfrak{B} \in \mathfrak{L}^{\mathtt{w}}$. Define its annihilators by

$$\mathfrak{N}_{\mathfrak{B}} := \{n \in \mathbb{R}^{\mathtt{w}}[\xi] \mid n^{\top}(\frac{d}{dt})\mathfrak{B} = 0\}$$

Note: $\mathfrak{N}_{\mathfrak{B}}$ is a $\mathbb{R}[\xi]$ sub-module of $\mathbb{R}^{\mathtt{w}}[\xi]$. In fact,

$$\mathfrak{L}^{\mathtt{w}} \overset{\text{one-to-one}}{\longleftrightarrow} \text{ sub-modules of } \mathbb{R}^{\mathtt{w}}[\xi]$$

Consequence: since sub-module is finitely generated, $\mathfrak{B}$ is determined by finite number of generators.

For example, the rows of R, but this is non-unique.

The model class $\mathfrak{L}^{\mathtt{w}}_{\mathtt{L}}$

We now define our model class $\mathfrak{L}^{\mathtt{w}}_{\mathtt{L}}$.

It consists of all $\mathfrak{B} \in \mathfrak{L}^{\mathtt{w}}$ such that

$\exists$ matrices $R_0, R_1, \ldots, R_L$

with restricted lag: $L \leq \mathtt{L}$

such that $\mathfrak{B}$ consists of all w that satisfy

$$R_0 w(t) + R_1 w(t+1) + \cdots + R_L w(t+L) = 0.$$

Polynomial matrix in

$$R(\sigma) w = 0$$

has $\mathrm{degree}(R) \leq \mathtt{L}$.

The MPUM in $\mathfrak{L}^{\mathtt{w}}_{\mathrm{L}}$

For infinite observation interval, $T = \infty$, the MPUM for $\tilde{w}$ in $\mathfrak{L}^{\mathtt{w}}$ always exists.

In fact, it equals

$$\mathfrak{B}^*_{\tilde{w}} = \text{span}(\{\tilde{w}, \sigma\tilde{w}, \sigma^2\tilde{w}, \ldots\})^{\text{closure}}$$

$\exists$ effective computational algorithms to go from $\tilde{w}$ to the corresponding R.

The MPUM in $\mathfrak{L}^{\mathtt{w}}_{\mathrm{L}}$

For finite observation interval, $T < \infty$, the MPUM in $\mathfrak{L}^{\mathtt{w}}$ is not very useful.

We hence restrict attention to the MPUM in $\mathfrak{L}^{\mathtt{w}}_{\mathrm{L}}$.

Also here the MPUM may not exist. Example:

$$\tilde{w} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ 2 \end{bmatrix}$$

has no MPUM in $\mathfrak{L}^{\mathtt{w}}_{2}$. What is the issue?

The MPUM in $\mathfrak{L}_{\mathtt{L}}^{\mathtt{w}}$

The MPUM in $\mathfrak{L}_{\mathtt{L}}^{\mathtt{w}}$ $\rightsquigarrow$ left kernel of the Hankel matrix ('windows')

$$\begin{bmatrix} \tilde{w}(1) & \tilde{w}(2) & \cdots & \tilde{w}(T-\mathtt{L}) \\ \tilde{w}(2) & \tilde{w}(3) & \cdots & \tilde{w}(T-\mathtt{L}+1) \\ \tilde{w}(3) & \tilde{w}(4) & \cdots & \tilde{w}(T-\mathtt{L}+2) \\ \vdots & \vdots & \vdots & \vdots \\ \tilde{w}(\mathtt{L}+1) & \tilde{w}(\mathtt{L}+2) & \cdots & \tilde{w}(T) \end{bmatrix}$$

This must have a 'module-like' structure, i.e.

$$\begin{bmatrix} N_0 & N_1 & \cdots & N_{\mathtt{L}-1} & 0 \end{bmatrix} \text{ in left kernel}$$

$$\Rightarrow \begin{bmatrix} 0 & N_0 & \cdots & N_{\mathtt{L}-2} & N_{\mathtt{L}-1} \end{bmatrix} \text{ in left kernel}$$

The MPUM in $\mathfrak{L}^{\mathtt{w}}_{\mathtt{L}}$

Proposition: the MPUM in $\mathfrak{L}^{\mathtt{w}}_{\mathtt{L}}$ exits if

$$\operatorname{rank}\left(\begin{bmatrix} \tilde{w}(1) & \tilde{w}(2) & \cdots & \tilde{w}(T-\mathtt{L}) \\ \tilde{w}(2) & \tilde{w}(3) & \cdots & \tilde{w}(T-\mathtt{L}+1) \\ \tilde{w}(3) & \tilde{w}(4) & \cdots & \tilde{w}(T-\mathtt{L}+2) \\ \vdots & \vdots & \vdots & \vdots \\ \tilde{w}(\mathtt{L}) & \tilde{w}(\mathtt{L}+1) & \cdots & \tilde{w}(T-1) \end{bmatrix}\right)$$

$$= \operatorname{rank}\left(\begin{bmatrix} \tilde{w}(1) & \tilde{w}(2) & \cdots & \tilde{w}(T-\mathtt{L}) & \tilde{w}(T-\mathtt{L}+1) \\ \tilde{w}(2) & \tilde{w}(3) & \cdots & \tilde{w}(T-\mathtt{L}+1) & \tilde{w}(T-\mathtt{L}+2) \\ \tilde{w}(3) & \tilde{w}(4) & \cdots & \tilde{w}(T-\mathtt{L}+2) & \tilde{w}(T-\mathtt{L}+3) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \tilde{w}(\mathtt{L}) & \tilde{w}(\mathtt{L}+1) & \cdots & \tilde{w}(T-1) & \tilde{w}(T) \end{bmatrix}\right)$$

We henceforth assume this to be the case.

Computation of this MPUM

Recursive computation

We need to compute the left kernel of

$$\begin{bmatrix} \tilde{w}(1) & \tilde{w}(2) & \cdots & \tilde{w}(T-\mathtt{L}-1) & \tilde{w}(T-\mathtt{L}) \\ \tilde{w}(2) & \tilde{w}(3) & \cdots & \tilde{w}(T-\mathtt{L}) & \tilde{w}(T-\mathtt{L}+1) \\ \tilde{w}(3) & \tilde{w}(4) & \cdots & \tilde{w}(T-\mathtt{L}+1) & \tilde{w}(T-\mathtt{L}+2) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \tilde{w}(\mathtt{L}+1) & \tilde{w}(\mathtt{L}+2) & \cdots & \tilde{w}(T-1) & \tilde{w}(T) \end{bmatrix}$$

Suffices to compute a set of generators of the sub-module of annihilators of the MPUM. Also, we would like to do this computation

recursively and approximately.

Recursive in T

Idea derived from the case $T = \infty$.

Assume time-series data $\mathbb{D} = \{d_1, d_2, \cdots, d_N\}, \quad d_k \in (\mathbb{R}^{\mathtt{w}})^{\mathbb{N}}$.

! Compute the MPUM in $\mathfrak{L}^{\mathtt{w}} \quad \rightsquigarrow \quad$ polynomial matrix $R_{\mathbb{D}}$.

1. $R_0 = I$
2. from $R_k \mapsto R_{k+1}$:
   - Compute $e_{k+1} := R_k(\sigma) d_{k+1}$.
   - Compute E_{k+1} corresponding to the MPUM of e_{k+1}
   - $R_{k+1} = E_{k+1} R_k$
3. $R_{\mathbb{D}} = R_N$

Reduces pbm to the computation of the MPUM for one time series.

Recursive in T

MPUM with one time-series, d, time-axis $-\mathbb{N}$

$$d = (\cdots, d(t), \cdots, d(-1), d(0))$$

Use the previous algorithm with the time-series data

$$d_{-\mathtt{k}} = (\cdots, d(-\mathtt{k}-1), d(-\mathtt{k})), \quad -\mathtt{k} \in \mathbb{N}$$

1. $R_{\mathtt{k}_0}$ given, say $= I$
2. from $R_{-\mathtt{k}} \mapsto R_{-\mathtt{k}+1}$:
   - $e_{-\mathtt{k}+1} := R_{-\mathtt{k}}(\sigma^{-1})d_{-\mathtt{k}+1}$. Looks as $(\cdots, 0, \cdots, 0, *)$
   - Compute $E_{-\mathtt{k}+1}$ the MPUM of $e_{-\mathtt{k}+1}$. Very simple!
   - $R_{-\mathtt{k}+1} = E_{-\mathtt{k}+1} R_{-\mathtt{k}}$
3. $R_{\{d\}} = R_0$

Recursive in T

In order to apply this to

$$\tilde{w} = (\tilde{w}(1), \tilde{w}(2), \ldots, \tilde{w}(T))$$

we miss an initial condition. This may be circumvented by considering instead the extended time-series

$$\ldots, \quad \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \quad \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \quad \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad \begin{bmatrix} \tilde{w}(1) \\ 0 \end{bmatrix}, \quad \begin{bmatrix} \tilde{w}(2) \\ 0 \end{bmatrix}, \quad \ldots, \quad \begin{bmatrix} \tilde{w}(T) \\ 0 \end{bmatrix}$$

and discarding certain of the relations obtained.

Can be implemented usin approximate linear algebra computations.

Recursive in annihilators

We need to compute a 'module basis' of the left kernel of

$$\begin{bmatrix} \tilde{w}(1) & \tilde{w}(2) & \cdots & \tilde{w}(T-\mathtt{L}-1) & \tilde{w}(T-\mathtt{L}) \\ \tilde{w}(2) & \tilde{w}(3) & \cdots & \tilde{w}(T-\mathtt{L}) & \tilde{w}(T-\mathtt{L}+1) \\ \tilde{w}(3) & \tilde{w}(4) & \cdots & \tilde{w}(T-\mathtt{L}+1) & \tilde{w}(T-\mathtt{L}+2) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \tilde{w}(\mathtt{L}+1) & \tilde{w}(\mathtt{L}+2) & \cdots & \tilde{w}(T-1) & \tilde{w}(T) \end{bmatrix}$$

Recursive in annihilators

Consider the Hankel matrices

$$\begin{bmatrix} \tilde{w}(1) & \tilde{w}(2) & \cdots & \tilde{w}(T-\Delta-2) & \tilde{w}(T-\Delta-1) \\ \tilde{w}(2) & \tilde{w}(3) & \cdots & \tilde{w}(T-\Delta-1) & \tilde{w}(T-\Delta) \\ \tilde{w}(3) & \tilde{w}(4) & \cdots & \tilde{w}(T-\Delta) & \tilde{w}(T-\Delta+1) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \tilde{w}(\Delta) & \tilde{w}(\Delta+1) & \cdots & \tilde{w}(T-1) & \tilde{w}(T) \end{bmatrix}$$

and let Δ vary from 1 to $\mathtt{L}+1$.

Recursive in annihilators

Basic idea.

Step 1: Compute (SVD)! basis R_0 for left kernel of

$$\begin{bmatrix} \tilde{w}(1) & \tilde{w}(2) & \cdots & \tilde{w}(T-1) & \tilde{w}(T) \end{bmatrix}$$

and its orthogonal complement S_0.

Keep R_0 as valid zero-th order laws, and replace $\tilde{w}$ by

$$\tilde{w}' = S_0\tilde{w} = (\tilde{w}'(1), \tilde{w}'(2), \ldots, \tilde{w}'(T)), \; \tilde{w}'(t) \in \mathbb{R}^{\mathtt{w}'}$$

This has no more zero-th order laws.

Recursive in annihilators

Step 2: (SVD)! $R_1 = [\, n_0 \;\; n_1 \,], n_0, n_1 \in \mathbb{R}^{1\times \mathtt{w}'}$ in left kernel

$$\begin{bmatrix} \tilde{w}'(1) & \tilde{w}'(2) & \cdots & \tilde{w}'(T-2) & \tilde{w}'(T-1) \\ \tilde{w}'(2) & \tilde{w}'(3) & \cdots & \tilde{w}'(T-1) & \tilde{w}'(T) \end{bmatrix}$$

Organize R_1 as the polynomial row vector

$$n(\xi) = n_0 + n_1\xi = \begin{bmatrix} r_1(\xi) & r_2(\xi) & \cdots & r_{\mathtt{w}}(\xi) \end{bmatrix}$$

Compute (Bézout) $C \in \mathbb{R}^{(\mathtt{w}'-1)\times \mathtt{w}'}[\xi]$ such that $\begin{bmatrix} n[\xi] \\ C[\xi] \end{bmatrix}$ is unimodular.

Keep n as a valid first order law, and replace $\tilde{w}'$ by

$$\tilde{w}" = C(\sigma)\tilde{w}' = (\tilde{w}"(1), \tilde{w}"(2), \ldots, \tilde{w}"(T-1)), \tilde{w}"(t) \in \mathbb{R}^{\mathtt{w}'-1}$$

etc.

Recursive in annihilators

Both recursions can be combined, leading to very efficient ways of finding an MPUM.

This is effective for exact data (or in finite field case).

Behavior of the algorithm for T large

Consistency

Typical way of evaluate SYSID algorithms:

Assume that

$$\tilde{w}(1), \tilde{w}(2), \ldots, \tilde{w}(T)$$

is generated by an element of the model class.

Does the algorithm return the model that generated the data for large T, or in the limit as $T \to \infty$ (consistency)?

Identifiability

The MPUM in $\mathfrak{L}^{\mathtt{w}}_{\mathtt{L}}$ for

$$\tilde{w}(1), \tilde{w}(2), \ldots, \tilde{w}(T)$$

returns $\mathfrak{B}$ if

1. $\tilde{w} \in \mathfrak{B}_{|[1,T]}$
2. $\mathtt{L}$ is sufficiently large
3. $\mathfrak{B}$ is controllable
4. the input component in $\tilde{w}$ is persistently exciting of sufficiently high order

The left kernel of the Hankel matrix is then module-like.

Identifiability

Assume $\tilde{w} = (\tilde{u}, \tilde{y})$ generated by behavior $\mathfrak{B}$. Then

$$\begin{bmatrix} \tilde{u}(1) & \tilde{u}(2) & \tilde{u}(3) & \cdots & \tilde{u}(T-\Delta+1) \\ \tilde{y}(1) & \tilde{y}(2) & \tilde{y}(3) & \cdots & \tilde{y}(T-\Delta+1) \\ \tilde{u}(2) & \tilde{u}(3) & \tilde{u}(4) & \cdots & \tilde{u}(T-\Delta+2) \\ \tilde{y}(2) & \tilde{y}(3) & \tilde{y}(4) & \cdots & \tilde{y}(T-\Delta+2) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \tilde{u}(\Delta) & \tilde{u}(\Delta+1) & \tilde{u}(\Delta+2) & \cdots & \tilde{u}(T) \\ \tilde{y}(\Delta) & \tilde{y}(\Delta+1) & \tilde{y}(\Delta+2) & \cdots & \tilde{y}(T) \end{bmatrix}$$

has ‘correct’ kernel & image if

1. $\Delta >$ lag$(\mathfrak{B})$
2. $\mathfrak{B}$ controllable
3. $\tilde{u}$ is persistently exciting of order $> \Delta + \mathtt{n}(\mathfrak{B})$

Identifiability

$$\begin{bmatrix} \tilde{u}(1) & \tilde{u}(2) & \tilde{u}(3) & \cdots & \tilde{u}(T-\mathtt{L}(\mathfrak{B})) \\ \tilde{y}(1) & \tilde{y}(2) & \tilde{y}(3) & \cdots & \tilde{y}(T-\mathtt{L}(\mathfrak{B})) \\ \tilde{u}(2) & \tilde{u}(3) & \tilde{u}(4) & \cdots & \tilde{u}(T-\mathtt{L}(\mathfrak{B})+1) \\ \tilde{y}(2) & \tilde{y}(3) & \tilde{y}(4) & \cdots & \tilde{y}(T-\mathtt{L}(\mathfrak{B})+1) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \tilde{u}(\mathtt{L}(\mathfrak{B})+1) & \tilde{u}(\mathtt{L}(\mathfrak{B})+2) & \tilde{u}(\mathtt{L}(\mathfrak{B})+3) & \cdots & \tilde{u}(T) \\ \tilde{y}(\mathtt{L}(\mathfrak{B})+1) & \tilde{y}(\mathtt{L}(\mathfrak{B})+2) & \tilde{y}(\mathtt{L}(\mathfrak{B})+3) & \cdots & \tilde{y}(T) \end{bmatrix}$$

kernel det. laws of the system (has rank $\mathtt{m}(\mathfrak{B})(\mathtt{L}(\mathfrak{B})+1)+\mathtt{n}(\mathfrak{B})$ **if**

$$\begin{bmatrix} \tilde{u}(1) & \tilde{u}(2) & \cdots & \tilde{u}(T-\mathtt{L}(\mathfrak{B})-\mathtt{n}(\mathfrak{B})-1) \\ \tilde{u}(2) & \tilde{u}(3) & \cdots & \tilde{u}(T-\mathtt{L}(\mathfrak{B})-\mathtt{n}(\mathfrak{B})) \\ \vdots & \vdots & \vdots & \vdots \\ \tilde{u}(\mathtt{L}(\mathfrak{B})+\mathtt{n}(\mathfrak{B})+1) & \tilde{u}(\mathtt{L}(\mathfrak{B})+\mathtt{n}(\mathfrak{B})+2) & \cdots & \tilde{u}(T) \end{bmatrix}$$

has rank $\mathtt{m}(\mathfrak{B})(L(\mathfrak{B})+\mathtt{n}(\mathfrak{B})+1)$.

From the data to the state trajectory

$$\tilde{w} \mapsto \tilde{x} \mapsto \begin{bmatrix} A & B \\ C & D \end{bmatrix}$$

If it is possible to pass from the data

$$\tilde{w}(1), \tilde{w}(2), \ldots, \tilde{w}(T)$$

directly to the state trajectory

$$\tilde{x}(1), \tilde{x}(2), \ldots, \tilde{x}(T)$$

Then we can identify the model by solving

$$\begin{bmatrix} \tilde{x}(2) & \tilde{x}(3) & \cdots & \tilde{x}(T) \\ \tilde{y}(1) & \tilde{y}(2) & \cdots & \tilde{y}(T-1) \end{bmatrix} = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} \tilde{x}(1) & \tilde{x}(2) & \cdots & \tilde{x}(T-1) \\ \tilde{u}(1) & \tilde{u}(2) & \cdots & \tilde{u}(T-1) \end{bmatrix}$$

These algorithms go to (A, B, C, D) instead of to R or to (P, Q). They have realization algorithms as a special case.

$$\tilde{w} \mapsto \tilde{x} \mapsto \begin{bmatrix} A & B \\ C & D \end{bmatrix}$$

How does this work?

$$\tilde{w}(1), \tilde{w}(2), \ldots, \tilde{w}(T)$$

$$\Downarrow$$

$$\tilde{x}(1), \tilde{x}(2), \ldots, \tilde{x}(T)$$

Several algorithms. We give 3 of them.

Assume contr., $\Delta > \mathtt{L}(\mathfrak{B})$, and pers. of exc. as needed.

$$\tilde{w} \mapsto \tilde{x} \mapsto \begin{bmatrix} A & B \\ C & D \end{bmatrix}$$

1. Compute ‘the’ left annihilators of $\mathcal{H}$:

$$\begin{bmatrix} N_1 & N_2 & N_3 & \cdots & N_\Delta \end{bmatrix} \begin{bmatrix} \tilde{w}(1) & \tilde{w}(2) & \cdots & \tilde{w}(T-\Delta+1) \\ \tilde{w}(2) & \tilde{w}(3) & \cdots & \tilde{w}(T-\Delta+2) \\ \tilde{w}(3) & \tilde{w}(4) & \cdots & \tilde{w}(T-\Delta+3) \\ \vdots & \vdots & \vdots & \vdots \\ \tilde{w}(\Delta) & \tilde{w}(\Delta+1) & \cdots & \tilde{w}(T) \end{bmatrix} = 0$$

$$\tilde{w} \mapsto \tilde{x} \mapsto \begin{bmatrix} A & B \\ C & D \end{bmatrix}$$

1. Compute ‘the’ left annihilators of $\mathcal{H}$:

$$\begin{bmatrix} N_1 & N_2 & N_3 & \cdots & N_\Delta \end{bmatrix} \begin{bmatrix} \tilde{w}(1) & \tilde{w}(2) & \cdots & \tilde{w}(T-\Delta+1) \\ \tilde{w}(2) & \tilde{w}(3) & \cdots & \tilde{w}(T-\Delta+2) \\ \tilde{w}(3) & \tilde{w}(4) & \cdots & \tilde{w}(T-\Delta+3) \\ \vdots & \vdots & \vdots & \vdots \\ \tilde{w}(\Delta) & \tilde{w}(\Delta+1) & \cdots & \tilde{w}(T) \end{bmatrix} = 0$$

Then

$$\begin{bmatrix} \tilde{x}(1) & \tilde{x}(2) & \cdots & \tilde{x}(T-\Delta+1) \end{bmatrix}$$

$$= \begin{bmatrix} N_2 & N_3 & \cdots & N_\Delta & 0 \\ N_3 & N_4 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots\vdots\vdots & \vdots & \vdots \\ N_{\Delta-1} & N_\Delta & \cdots & 0 & 0 \\ N_\Delta & 0 & \cdots & 0 & 0 \end{bmatrix} \begin{bmatrix} \tilde{w}(1) & \tilde{w}(2) & \cdots & \tilde{w}(T-\Delta+1) \\ \tilde{w}(2) & \tilde{w}(3) & \cdots & \tilde{w}(T-\Delta+2) \\ \tilde{w}(3) & \tilde{w}(4) & \cdots & \tilde{w}(T-\Delta+3) \\ \vdots & \vdots & \vdots & \vdots \\ \tilde{w}(\Delta) & \tilde{w}(\Delta+1) & \cdots & \tilde{w}(T) \end{bmatrix}$$

$$\tilde{w} \mapsto \tilde{x} \mapsto \begin{bmatrix} A & B \\ C & D \end{bmatrix}$$

$$\left[\begin{array}{c} \mathcal{H}_- \\ \hline \mathcal{H}_+ \end{array}\right] = \left[\begin{array}{cccc} \tilde{w}(1) & \tilde{w}(2) & \cdots & \tilde{w}(T-2\Delta+1) \\ \tilde{w}(2) & \tilde{w}(3) & \cdots & \tilde{w}(T-2\Delta+2) \\ \vdots & \vdots & \vdots & \vdots \\ \tilde{w}(\Delta) & \tilde{w}(\Delta+1) & \cdots & \tilde{w}(T-\Delta) \\ \hline \tilde{w}(\Delta+1) & \tilde{w}(\Delta+2) & \cdots & \tilde{w}(T-\Delta+1) \\ \tilde{w}(\Delta+2) & \tilde{w}(\Delta+3) & \cdots & \tilde{w}(T-\Delta+2) \\ \vdots & \vdots & \vdots & \vdots \\ \tilde{w}(2\Delta) & \tilde{w}(2\Delta+1) & \cdots & \tilde{w}(T) \end{array}\right] \begin{array}{c} \uparrow \\ \uparrow \\ \uparrow \\ \text{PAST} \\ \hline \text{FUTURE} \\ \downarrow \\ \downarrow \\ \downarrow \end{array}$$

$$\tilde{w} \mapsto \tilde{x} \mapsto \begin{bmatrix} A & B \\ C & D \end{bmatrix}$$

$$\left[\begin{array}{c} \mathcal{H}_- \\ \hline \mathcal{H}_+ \end{array}\right] = \left[\begin{array}{cccc} \tilde{w}(1) & \tilde{w}(2) & \cdots & \tilde{w}(T-2\Delta+1) \\ \tilde{w}(2) & \tilde{w}(3) & \cdots & \tilde{w}(T-2\Delta+2) \\ \vdots & \vdots & \vdots & \vdots \\ \tilde{w}(\Delta) & \tilde{w}(\Delta+1) & \cdots & \tilde{w}(T-\Delta) \\ \hline \tilde{w}(\Delta+1) & \tilde{w}(\Delta+2) & \cdots & \tilde{w}(T-\Delta+1) \\ \tilde{w}(\Delta+2) & \tilde{w}(\Delta+3) & \cdots & \tilde{w}(T-\Delta+2) \\ \vdots & \vdots & \vdots & \vdots \\ \tilde{w}(2\Delta) & \tilde{w}(2\Delta+1) & \cdots & \tilde{w}(T) \end{array}\right] \begin{array}{c} \uparrow \\ \uparrow \\ \uparrow \\ \textbf{PAST} \\ \hline \textbf{FUTURE} \\ \downarrow \\ \downarrow \\ \downarrow \end{array}$$

2. The intersection of the span of the rows of $\mathcal{H}_-$

with the span of the rows of $\mathcal{H}_+$ equals

$$\begin{bmatrix} \tilde{x}(\Delta) & \tilde{x}(\Delta+1) & \cdots & \tilde{x}(T-\Delta) \end{bmatrix} \quad \textbf{PRESENT STATE}$$

Nice num. impl. (e.g. via left kernel) $\rightsquigarrow$ subspace ID

$$\tilde{w} \mapsto \tilde{x} \mapsto \begin{bmatrix} A & B \\ C & D \end{bmatrix}$$

3. Solve for G

$$\begin{bmatrix} \tilde{w}(1) & \cdots & \tilde{w}(T-2\Delta+1) \\ \vdots & \vdots & \vdots \\ \tilde{w}(\Delta) & \cdots & \tilde{w}(T-\Delta) \\ \hline \tilde{u}(\Delta+1) & \cdots & \tilde{u}(T-\Delta+1) \\ \vdots & \vdots & \vdots \\ \tilde{u}(2\Delta) & \cdots & \tilde{u}(T) \end{bmatrix} \boxed{G} = \begin{bmatrix} \tilde{w}(1) & \cdots & \tilde{w}(T-2\Delta+1) \\ \vdots & \vdots & \vdots \\ \tilde{w}(\Delta) & \cdots & \tilde{w}(T-\Delta) \\ \hline 0 & \cdots & 0 \\ \vdots & \vdots & \vdots \\ 0 & \cdots & 0 \end{bmatrix}$$

$$\begin{bmatrix} \tilde{y}(\Delta+1) & \cdots & \tilde{y}(T-\Delta+1) \\ \vdots & \vdots & \vdots \\ \tilde{y}(2\Delta) & \cdots & \tilde{y}(T) \end{bmatrix} G = \boxed{\begin{bmatrix} \tilde{x}(\Delta) & \cdots & \tilde{x}(T-\Delta) \end{bmatrix}}$$

Computes $\tilde{x}$!

$\cong$ **'oblique projection**

$\tilde{w} \mapsto R$ **or** $\begin{bmatrix} A & B \\ C & D \end{bmatrix}$

These algorithms, compute the left kernel of $\mathcal{H}$, etc. allow approximate implementations. For the state algorithms, this is worked out very well (subspace ID).

$$\textbf{SVD} \qquad \tilde{X} = \begin{bmatrix} \tilde{x}(1) & \tilde{x}(2) & \cdots & \tilde{x}(T) \end{bmatrix}$$

$$\rightsquigarrow \quad \tilde{X}^{\text{red}} = \begin{bmatrix} \tilde{x}^{\text{red}}(1) & \tilde{x}^{\text{red}}(2) & \cdots & \tilde{x}^{\text{red}}(T) \end{bmatrix}$$

followed by LS solution of

$$\begin{bmatrix} \tilde{x}^{\text{red}}(2) & \tilde{x}^{\text{red}}(3) & \cdots & \tilde{x}^{\text{red}}(T) \\ \tilde{y}(1) & \tilde{y}(2) & \cdots & \tilde{y}(T-1) \end{bmatrix} = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} \tilde{x}^{\text{red}}(1) & \tilde{x}^{\text{red}}(2) & \cdots & \tilde{x}^{\text{red}}(T-1) \\ \tilde{u}(1) & \tilde{u}(2) & \cdots & \tilde{u}(T-1) \end{bmatrix}$$

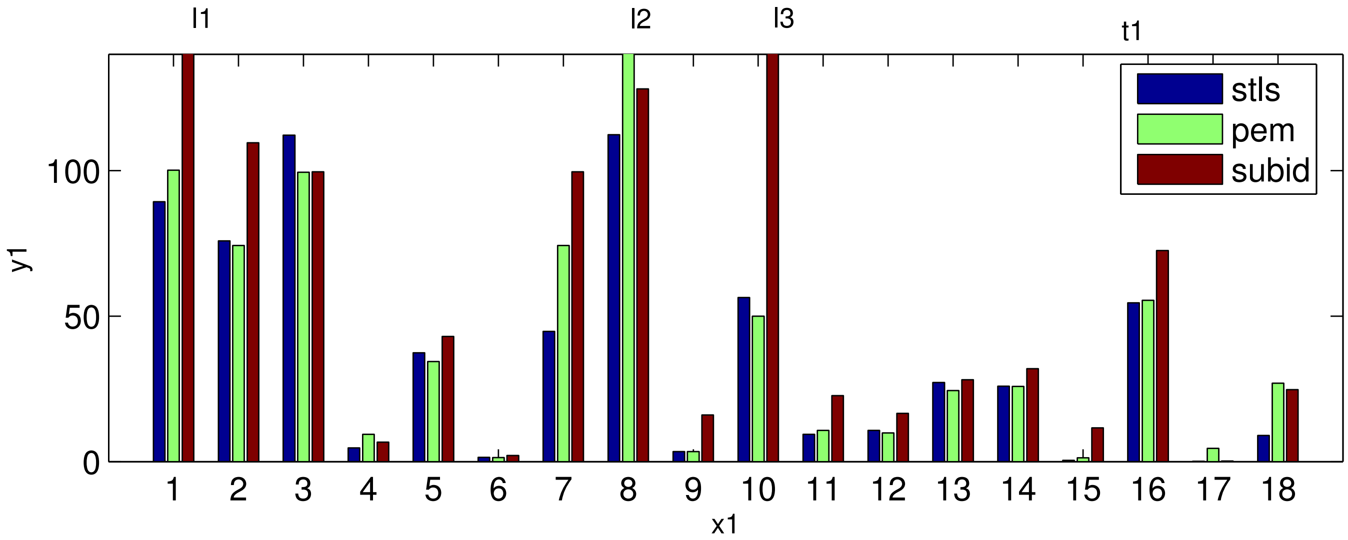
Performance

#	Data set name	T	m	p	l
1	Data of the western basin of Lake Erie	57	5	2	1
2	Data of Ethane-ethylene column	90	5	3	1
3	Data of a 120 MW power plant	200	5	3	2
4	Heating system	801	1	1	2
5	Data from an industrial dryer	867	3	3	1
6	Data of a hair dryer	1000	1	1	5
7	Data of the ball-and-beam setup in SISTA	1000	1	1	2
8	Wing flutter data	1024	1	1	5
9	Data from a flexible robot arm	1024	1	1	4
10	Data of a glass furnace (Philips)	1247	3	6	1
11	Heat flow density through a two layer wall	1680	2	1	2
12	Simulation of a pH neutralization process	2001	2	1	6
13	Data of a CD-player arm	2048	2	2	1
14	Data from an industrial winding process	2500	5	2	2
15	Liquid-saturated heat exchanger	4000	1	1	2
16	Data from an evaporator	6305	3	3	1
17	Continuous stirred tank reactor	7500	1	2	1
18	Model of a steam generator	9600	4	4	1

Performance

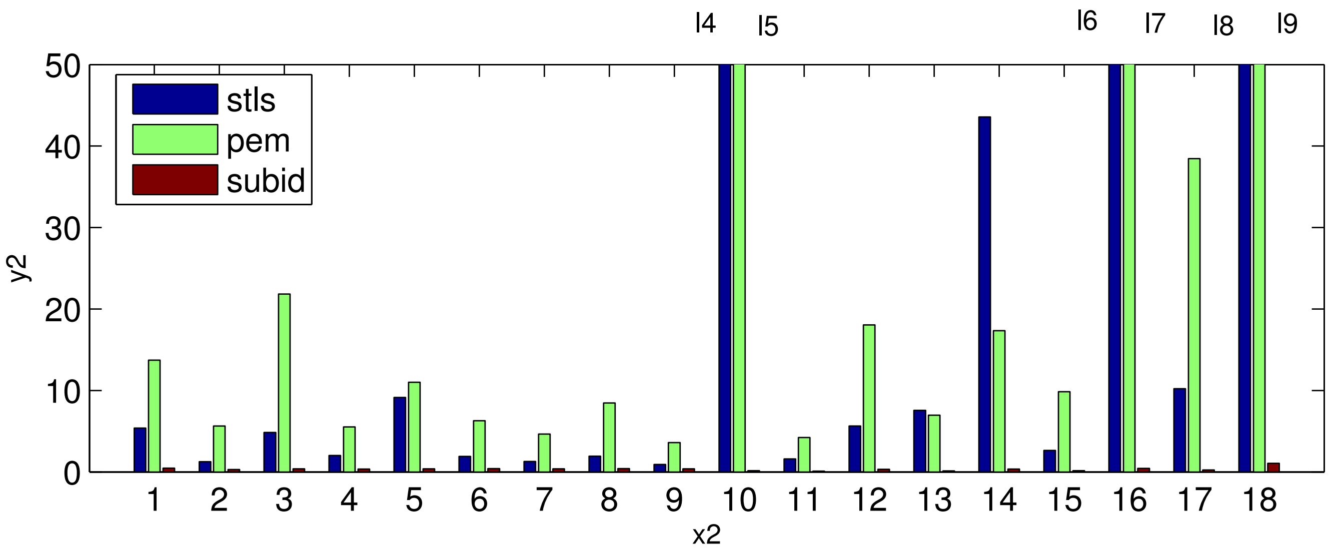
Compare the misfit on the last 30% of the outputs and the execution time for computing the ID model from the first 70% of the data.

Misfit

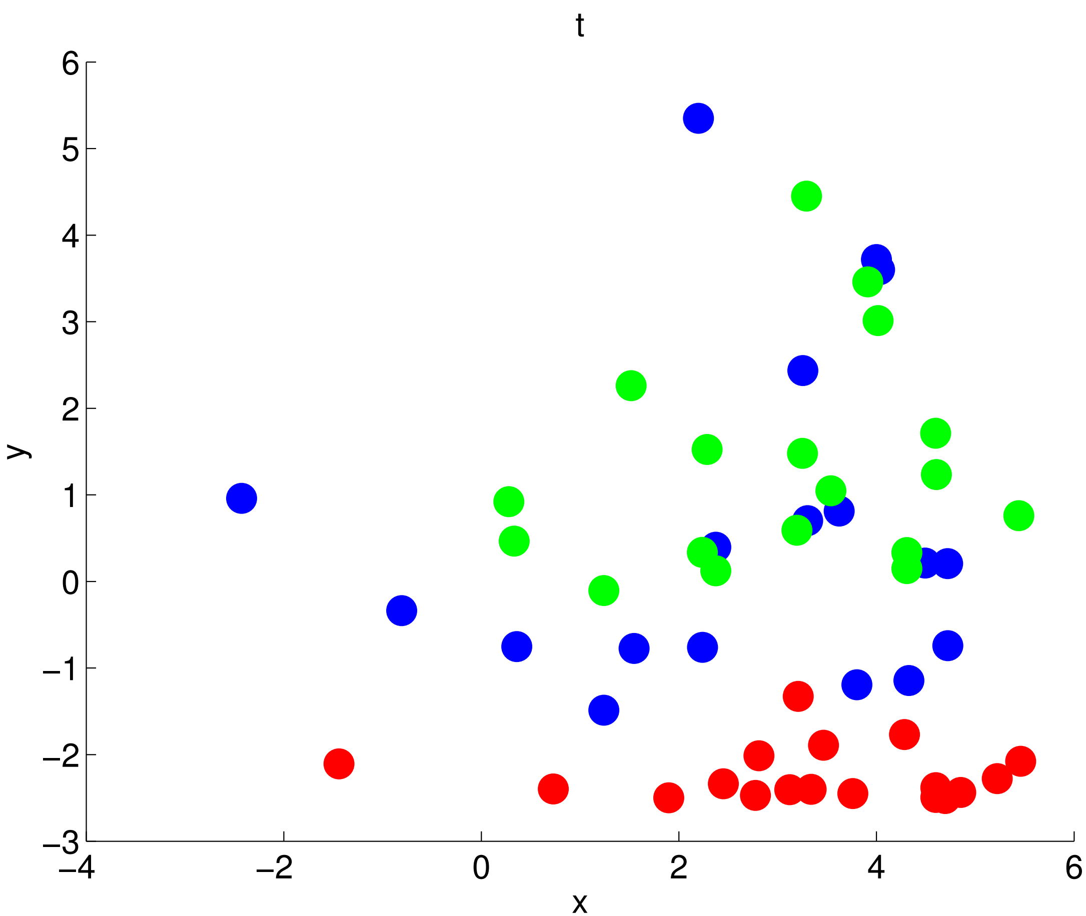


Performance

Execution time

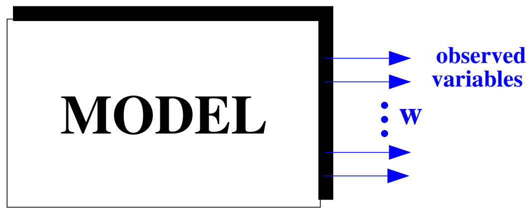


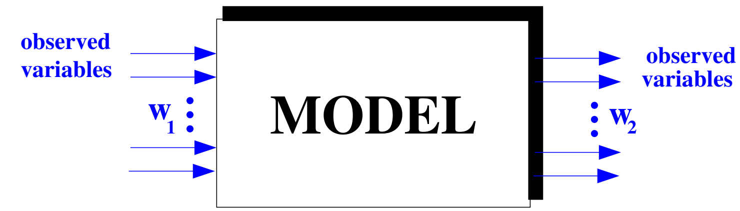
Performance



Latency minimization

Why latent variables?

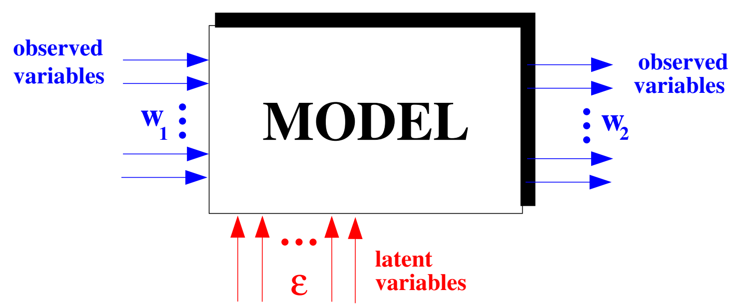




$$R_0 w(t) + R_1 w(t+1) + \cdots + R_L w(t+L) = 0$$

---

versus



$$R_0 w(t) + R_1 w(t+1) + \cdots + R_L w(t+L)$$
$$= M_0 \varepsilon(t) + M_1 \varepsilon(t+1) + \cdots + M_L \varepsilon(t+L)$$

Why latent variables?

For the w-behavior, this gives nothing new

($\Leftarrow$ **elimination theorem**).

So, what is the rationale for using latent variables ε ?

Why latent variables?

Data $\tilde{w}(t_1), \tilde{w}(t_1+1), \ldots, \tilde{w}(t_2)$ **with** $\tilde{w}(t) \in \mathbb{R}$

The model

$$R_0 w(t) + R_1 w(t+1) + \cdots + R_L w(t+L) = 0$$

$\rightsquigarrow$ **either** $w =$ **input, free,** $\mathfrak{B} = \mathbb{R}^{\mathbb{T}}$

or $w =$ **output,** $\rightsquigarrow \mathfrak{B} \cong$ **sums of 'exponentials'**

$\rightsquigarrow$ **very restrictive.**

Assuming unobserved inputs:

$$R_0 w(t) + \cdots + R_L w(t+L) = M_0 \varepsilon(t) + \cdots + M_L \varepsilon(t+L)$$

gives better possibilities, e.g. for prediction.

Latency minimization

Define the **'latency'**:

$$\text{latency}\ (\tilde{w}, \mathfrak{B}) := \ \text{minimum}\ ||\tilde{\varepsilon}||_{\ell^2}$$

with the minimum taken over all $\tilde{\varepsilon}$ such that

$$R_0\tilde{w}(t) + \cdots + R_L\tilde{w}(t+L) = M_0\tilde{\varepsilon}(t) + \cdots + M_L\tilde{\varepsilon}(t+L)$$

i.e. min. over all $\tilde{\varepsilon}$ that 'explain' $\tilde{w}(1), \tilde{w}(2), \ldots, \tilde{w}(T)$.

$\rightsquigarrow$ **system ID:** search for the optimal model,

in the sense of minimal latency

in a given model class.

Latency minimization

- How do we compute the latency, the optimal $\tilde{\varepsilon}$'s?
- Algorithms for minimization over (R, M)'s in model class.

Latency minimization is a deterministic Kalman filtering pbm

The latency is actually equal to the prediction error!

$\rightsquigarrow$ **deterministic interpretation, system ID toolbox, etc.**

Remarks on stochastic SYSID

Why stochastic interpretation?

$$R_0 w(t) + \cdots + R_L w(t+L) = M_0 \varepsilon(t) + \cdots + M_L \varepsilon(t+L)$$

We can consider ε as a stochastic disturbance.

If we take also u as a stochastic process, then w stochastic.

SYSID pbm is then a statistical one, leading to maximum likelihood estimation (very related to PEM).

It allows evaluation of algorithms in terms of $T \to \infty$. Nice statistical questions emerge, as consistency, asymptotic efficiency, etc.

$\rightsquigarrow$ deep theory of ARMAX systems.

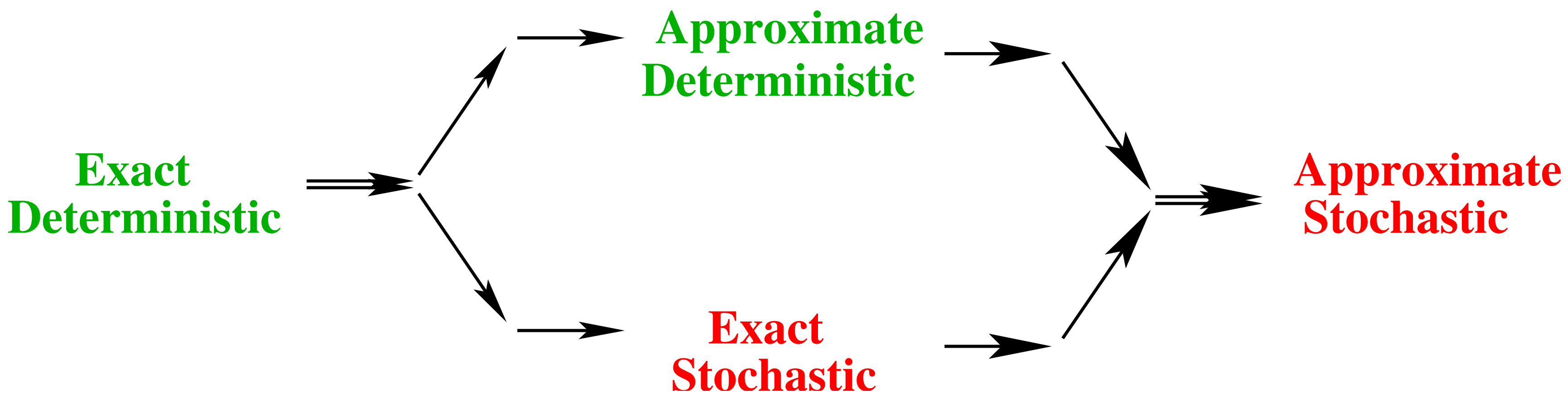
Why stochastic interpretation?

It is difficult to argue that stochastic unobserved disturbances offer a realistic explanation of the lack of fit between observations and the deterministic part.

This lack of fit is more likely a result of low order, linear models for nonlinear systems, neglected dynamics, approximation, in addition to unmeasured inputs, which may or may not be stochastic.

Stochastic methods offer the user a ‘certificate’ under which the algorithms work well.

Conclusions



- We concentrated on exact deterministic SYSID.
- Nice concepts, as MPUM.
- Realization theory as special case
- Subspace algorithms very effective

Thank you

Thank you

Thank you

Thank you

Thank you

Thank you

Thank you

Thank you