



# *Identification of continuous-time linear models*

-----

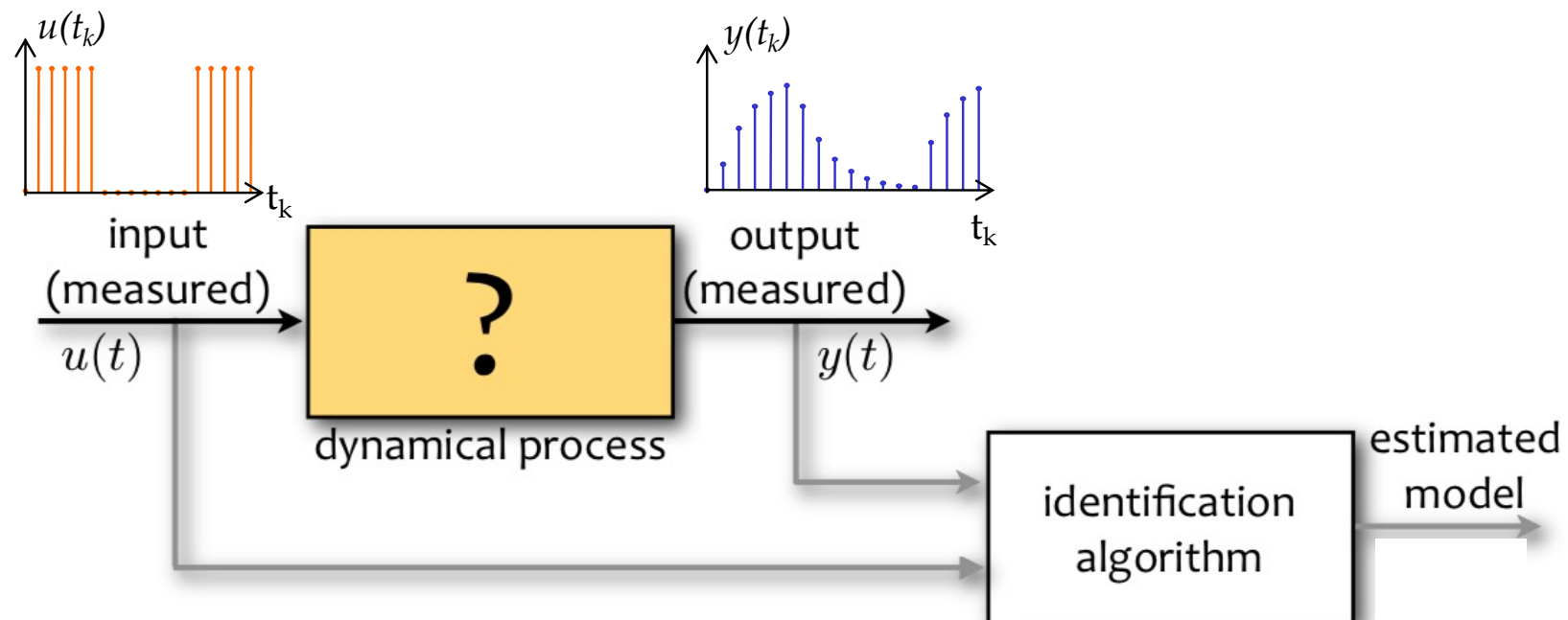
*Practical aspects and the CONTSID toolbox*

Hugues GARNIER

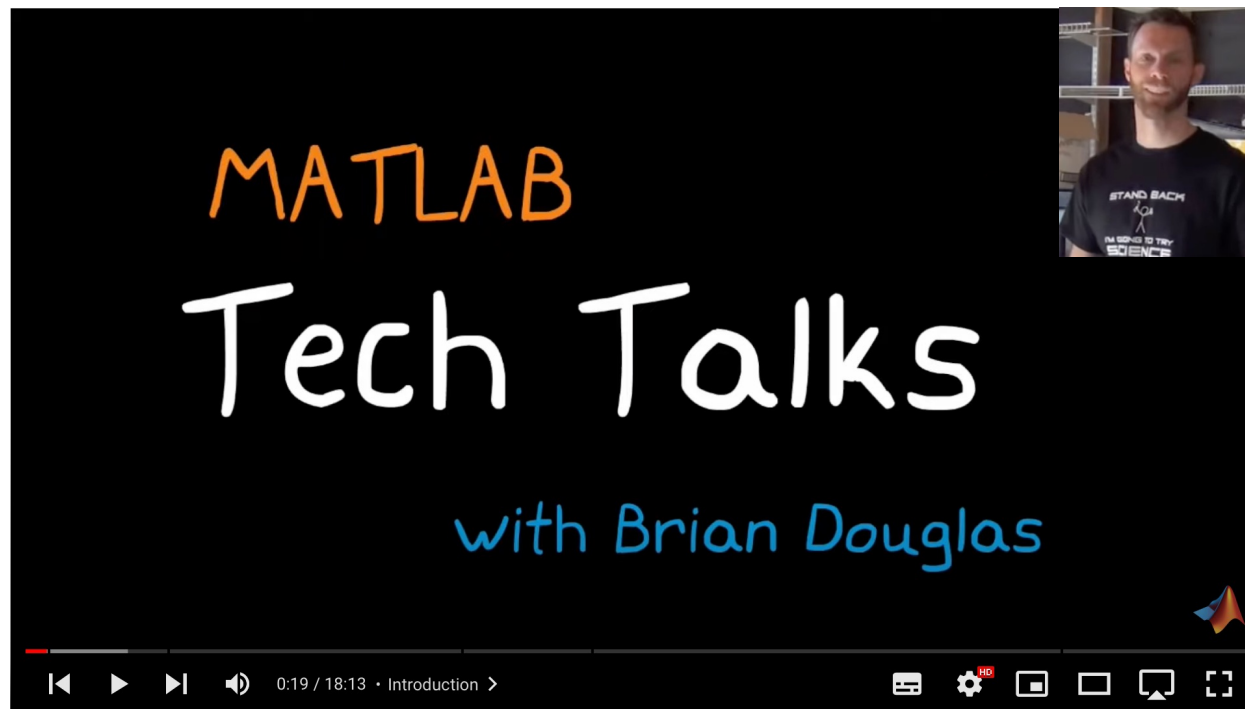
[hugues.garnier@univ-lorraine.fr](mailto:hugues.garnier@univ-lorraine.fr)

## System identification – Brief recap

- System identification is an iterative process, where you identify models with different structures from sampled data and compare model performance
- Ultimately, choose the simplest model that best describes the dynamics of your system
- **A priori physical knowledge** about the system is often a key for success



## Linear System Identification – A introduction from Brian



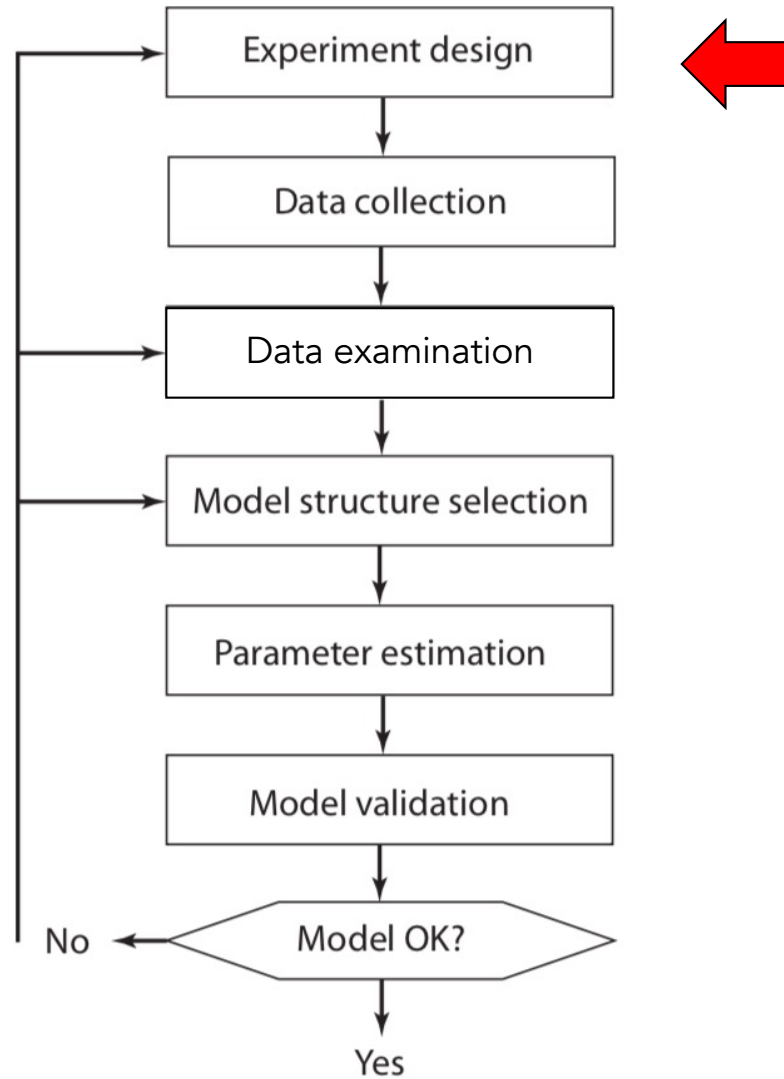
Linear System Identification | System Identification, Part 2

16 mn

[www.youtube.com/watch?v=qC\\_C04SEV1E](https://www.youtube.com/watch?v=qC_C04SEV1E)

# Data-driven system identification

## An iterative procedure



## Experiment design for data collection

- To obtain a good model of your system, you must have measured data that reflects the dynamic behavior of the system
- The accuracy of the model depends on the quality of the measurement data, which in turn depends on the experiment design

Often good to use a two-stage approach

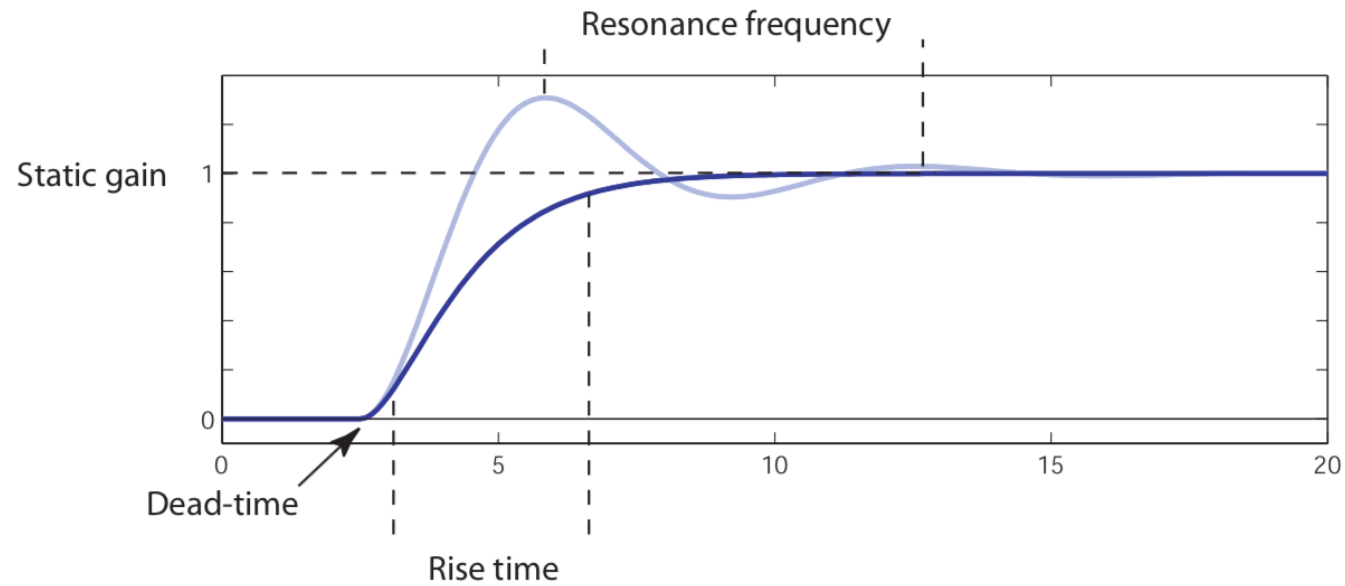
### 1. Preliminary experiments

- step/impulse response tests to get basic understanding of system dynamics
- linearity, stationary gains, time delays, time constants, sampling interval

### 2. Data collection for model estimation

- carefully designed experiment to enable good model fit
- operating point, input signal type, number of data points to collect, etc.

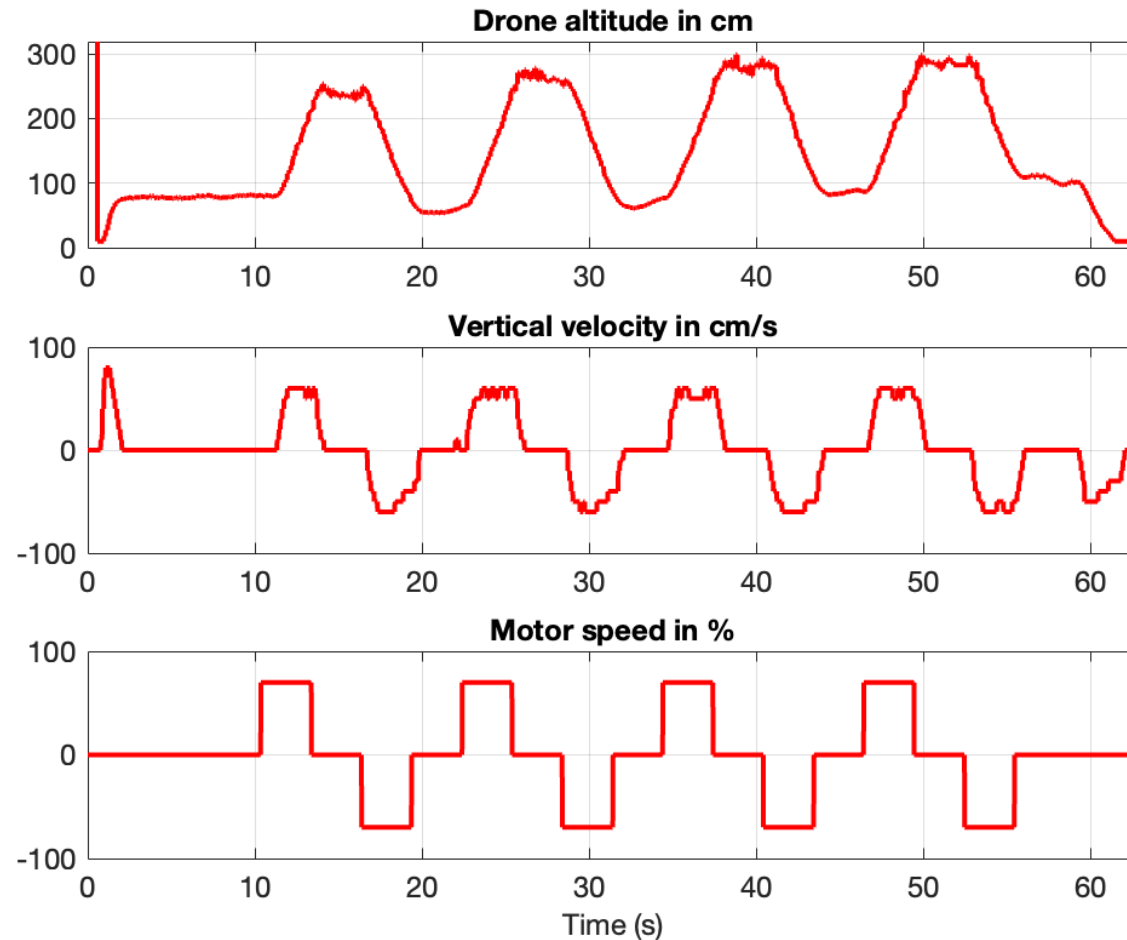
## Preliminary test: step response experiment



Useful for obtaining qualitative information about system

- indicates dead-times, static gain, time constants and resonances

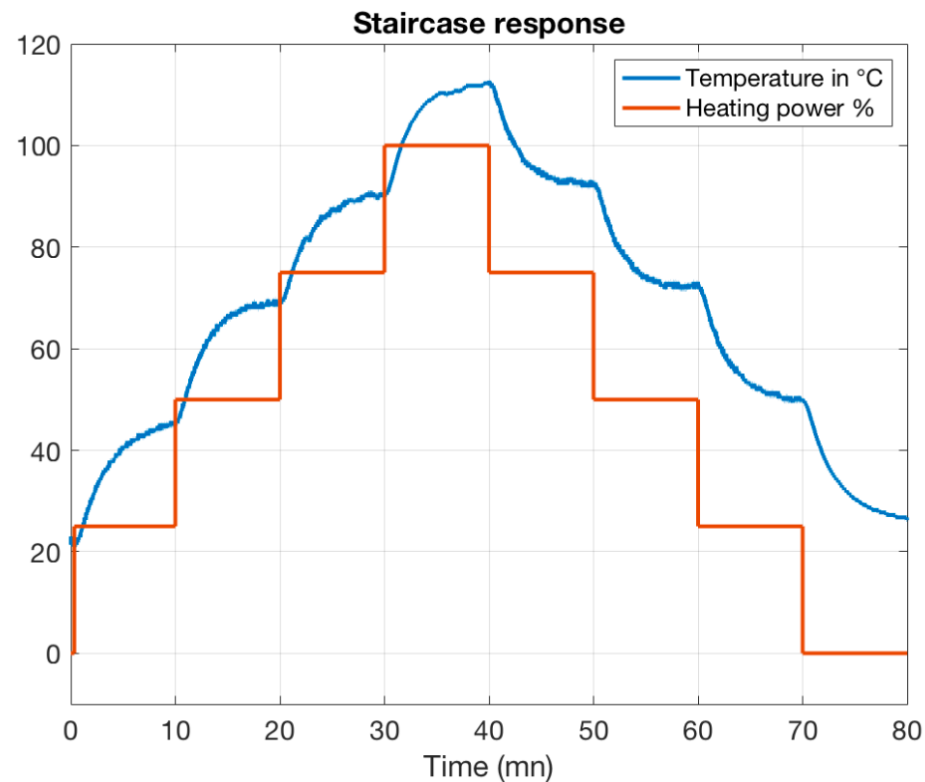
Preliminary test: square wave response experiment  
 Apply, when ever possible, periodic square wave input



Give insights about non-linearity effects in the system

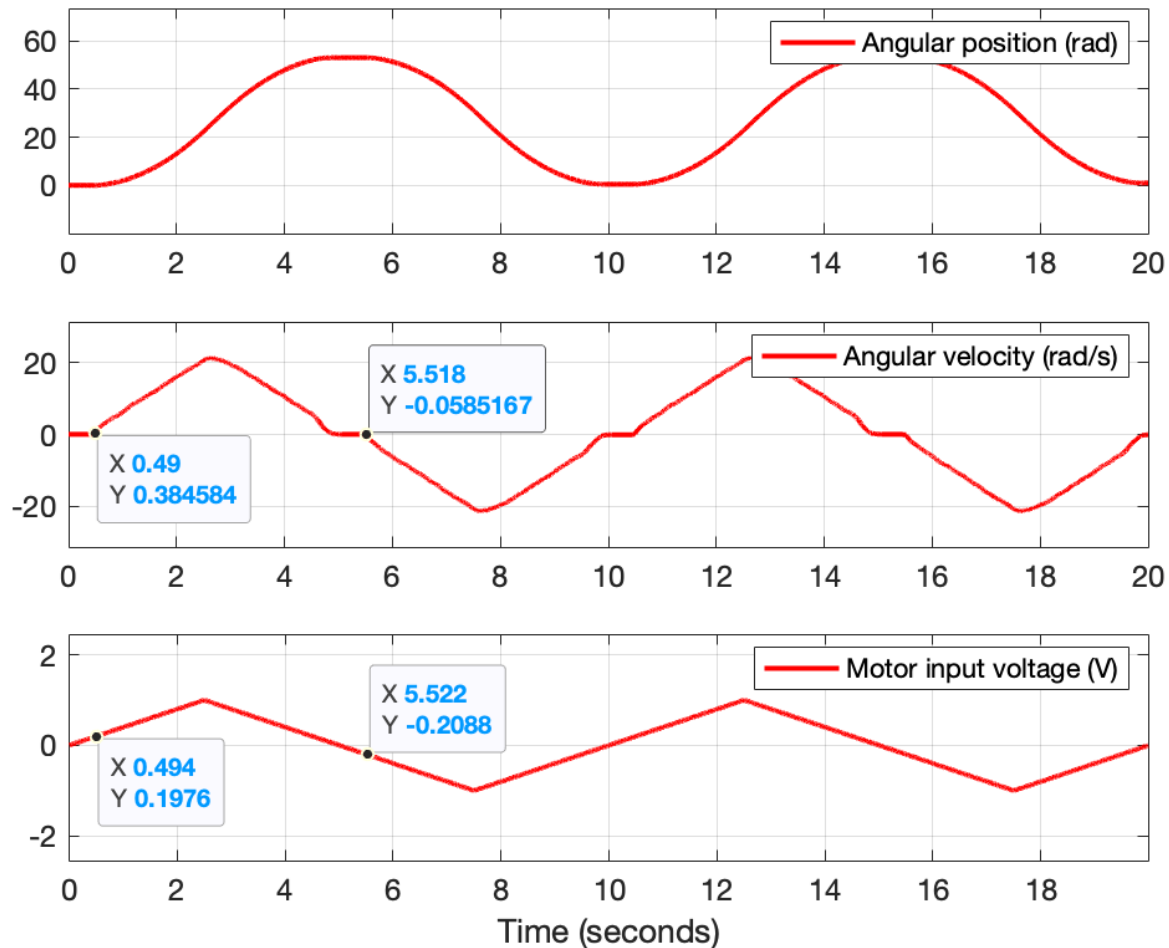
## Tests for verifying linearity

- a linear system has the **same response independent of the operative point**
- test step response in different operating points





## Tests for detecting dry friction



A simple linear model will not be able to capture the friction effects from these data

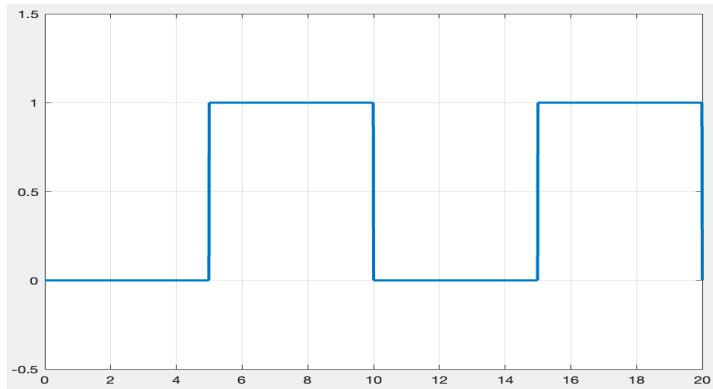
- Use a nonlinear model to better capture the friction effects
- Use amplitude of the steps  $>0.2$  to cancel out the friction effects

## Choice of inputs for informative data

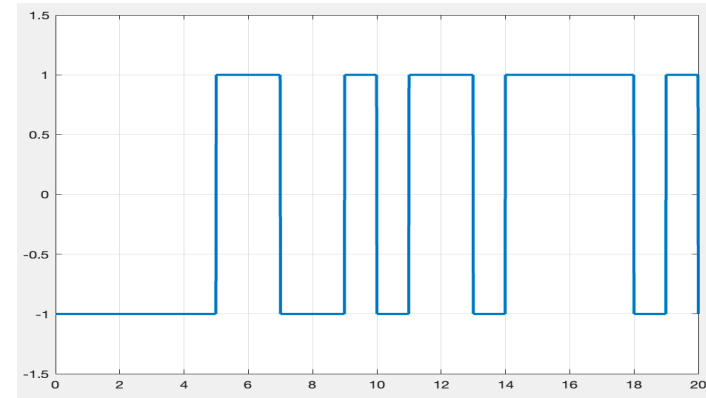
- Needs to be sufficiently “rich.”
  - Input signal is needed to excite the system
  - The experiment should be carried out under conditions that are similar to those under which the model is going to be used
- Amplitude
  - Trade-off is needed
    - Large amplitude gives good signal-to-noise ratio, low parameter estimate variance
    - But most systems enter into nonlinear regimes for large input amplitude
- Number of data points
  - The larger, the better (...if data is informative)

# Common choices of input for linear system identification

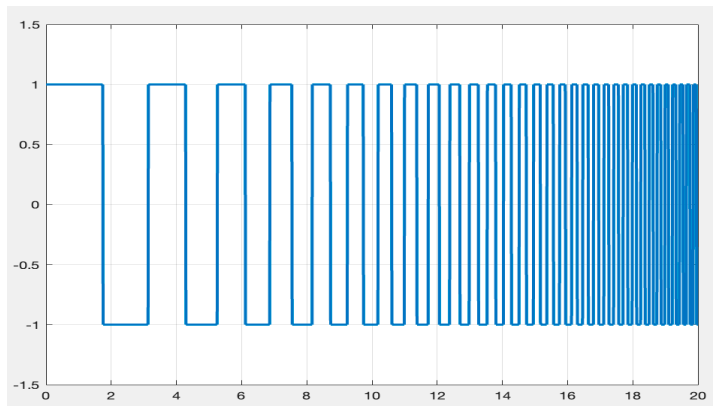
Step input/square wave



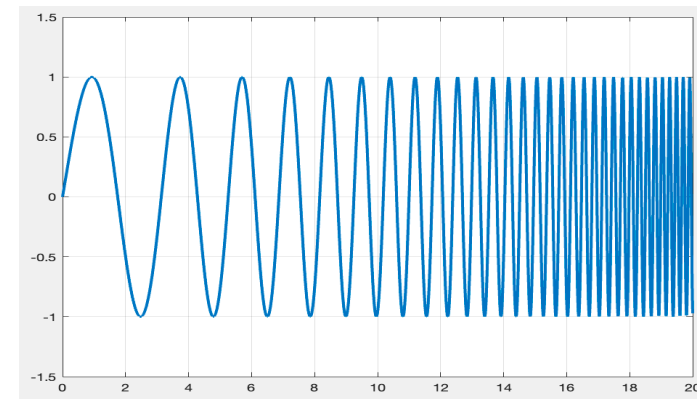
PRBS



Square chirp



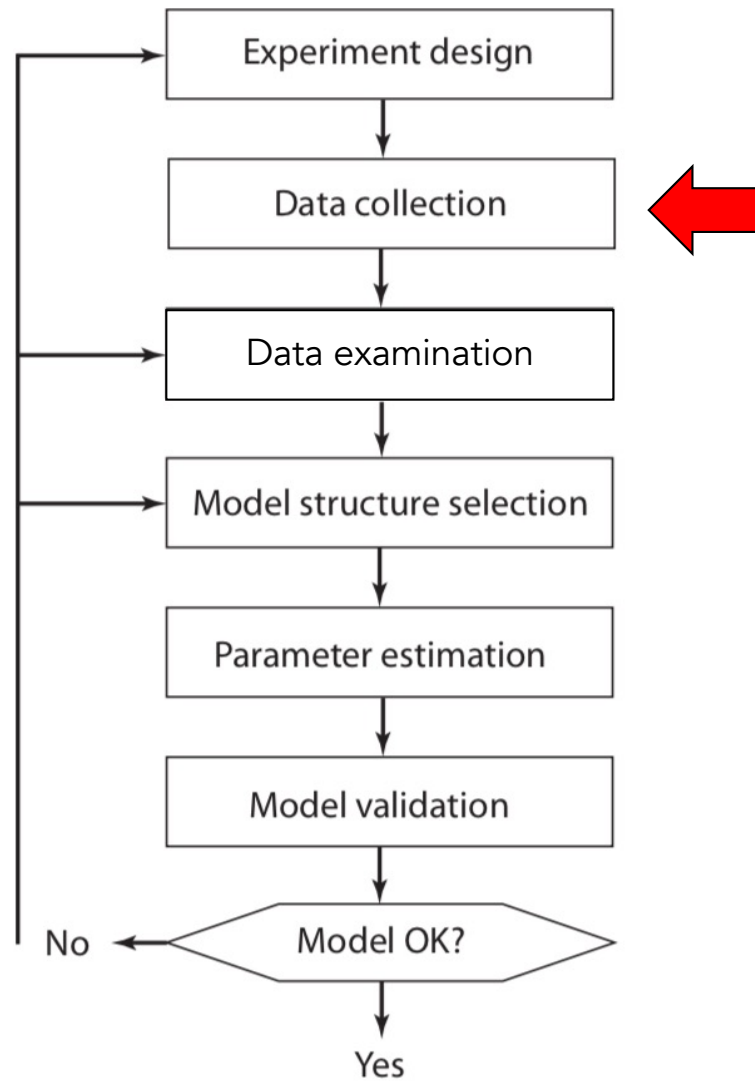
Sine chirp



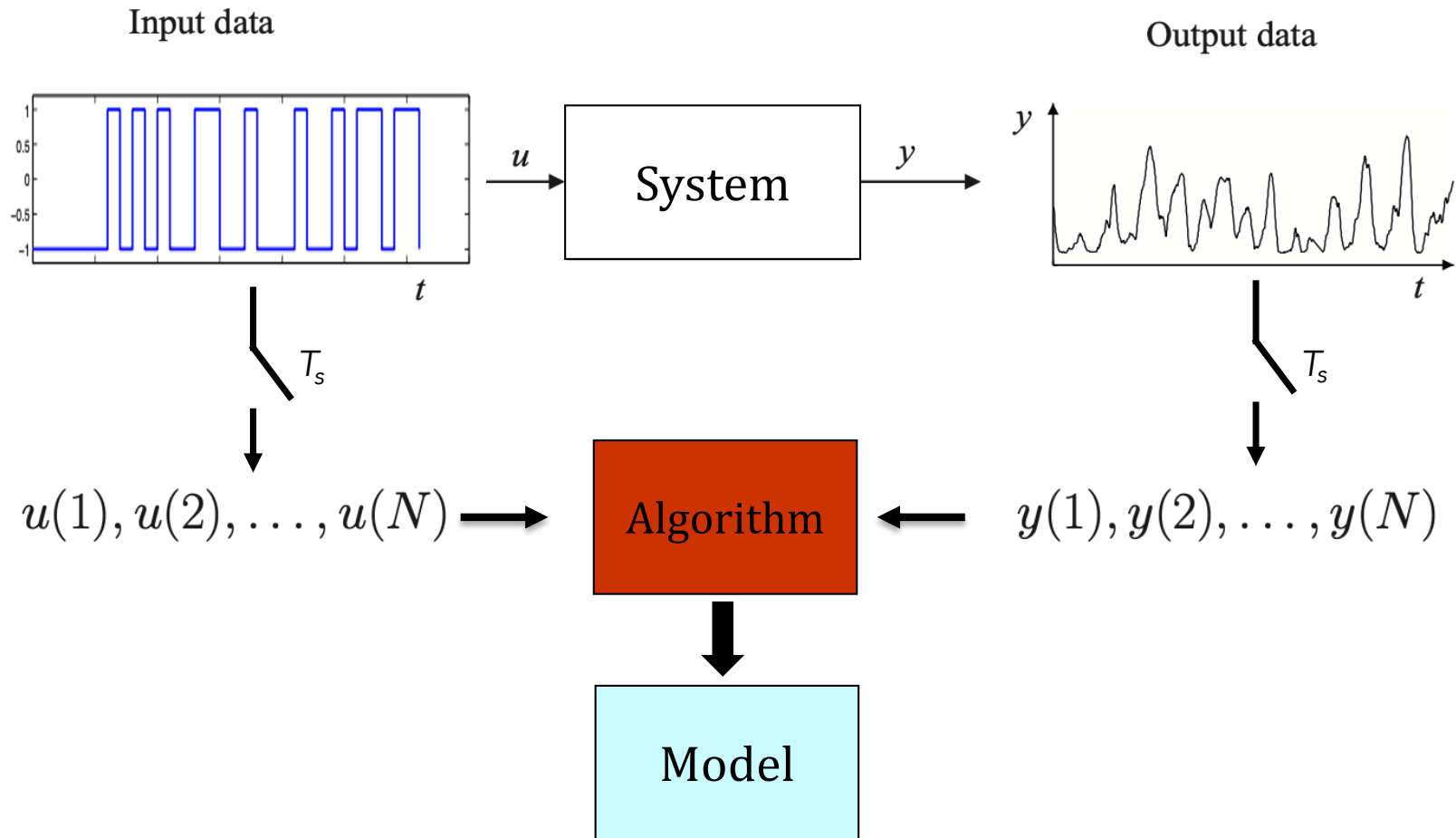
*Multisine are also common*

# Data-driven system identification

## An iterative procedure

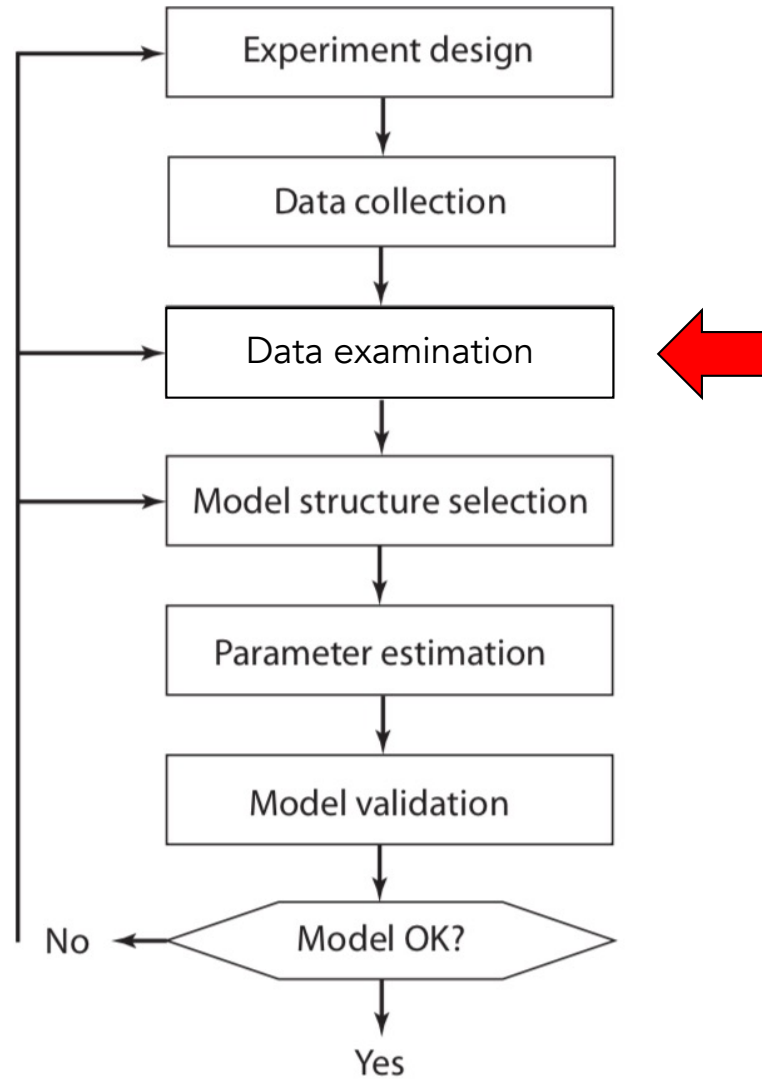


# Data collection



# Data-driven system identification

## An iterative procedure



## Examination of the collected data

**ALWAYS** plot **FIRST** the input/output data !

Examine carefully the measured data and identify possible problems

- Offset and drift (low-frequency disturbances)
- Occasional bursts and outliers
- High-frequency noise/disturbance

Select good/relevant segments of data for model estimation and validation

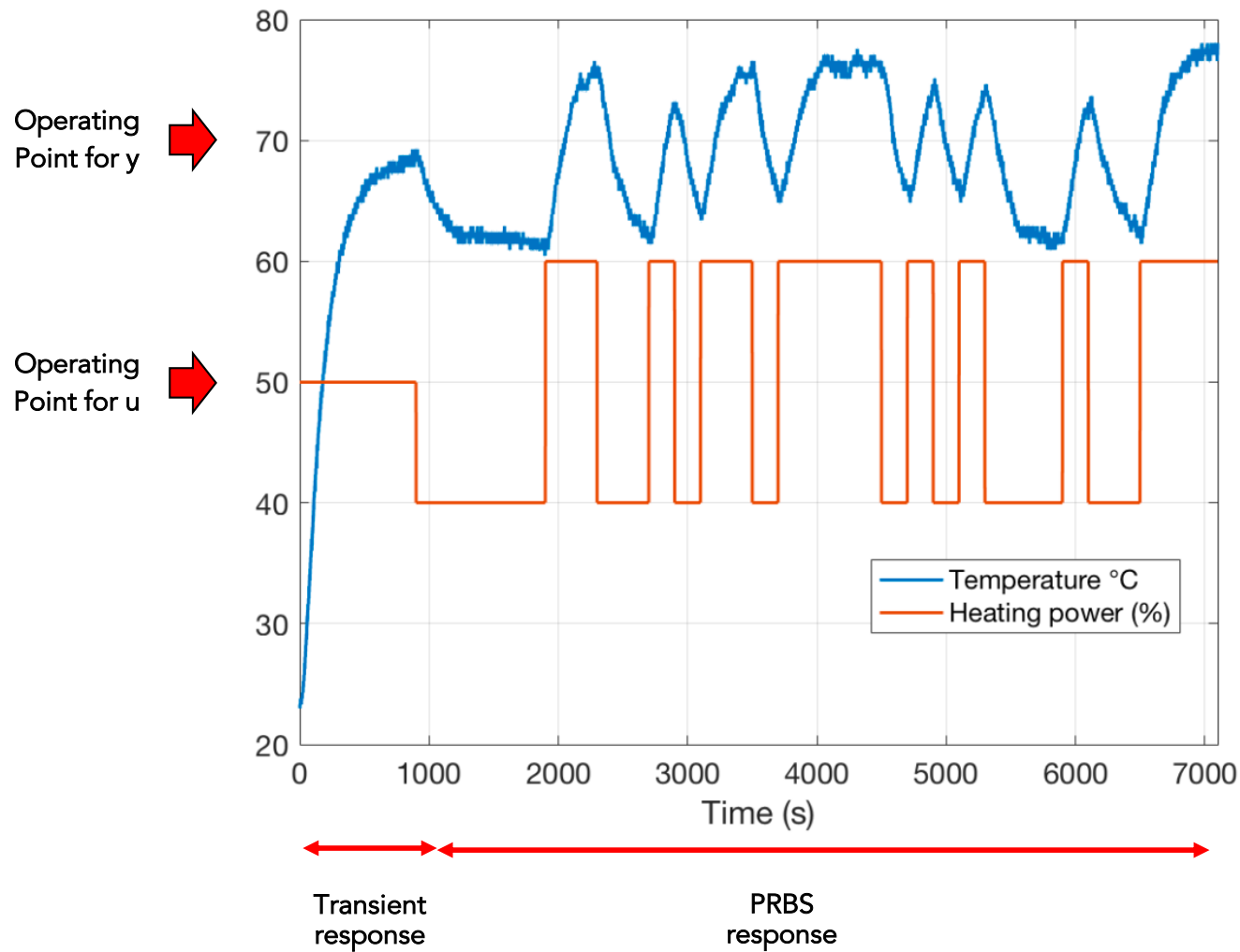
## Post processing of the collected data

- Input and output should be scaled to have approximately the same magnitude to avoid the numerical problems
- Look at the data:
  - Proper signal levels, frequencies, disturbance. . .
- Remove
  - transients before reaching the correct operating point (for nonlinear
  - signal mean (if not a physical model)
  - drift and slow trends (detrend in MATLAB)
  - high frequency noise should have been removed by antialiasing filter. In case apply a low-pass filter (keeping the breakpoints of the Bode plot)
  - "Outliers" (manifestly erroneous measurements): check plot of residuals

$$y(t) - \hat{y}(t|\theta)$$

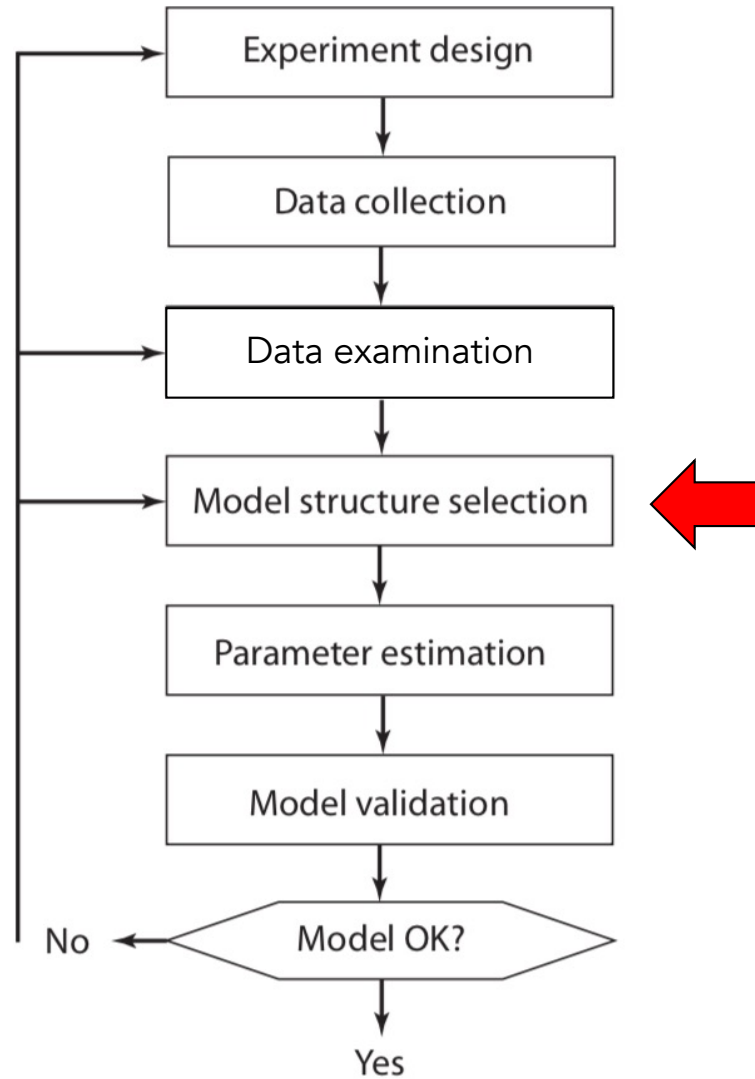


# Examination of the collected data What do you observed?



# Data-driven system identification

## An iterative procedure



## Family of linear model structures

- A model structure is a mathematical relationship between input and output variables that contains unknown parameters

- Common examples of linear model structures are:

- Input/output polynomial models  $Ay = \frac{B}{F}u + \frac{C}{D}e$  CT/DT

- Low-order process models plus delay  $G(s) = \frac{K_p}{1 + T_{p1}s} e^{-T_d s}$  CT

- Transfer function models plus delay  $G(s) = \frac{(b_0 + b_1s + b_2s^2 + \dots)}{(1 + f_1s + f_2s^2 + \dots)} e^{-T_d s}$  CT/DT

- State-space models  $\dot{x} = Ax + Bu$   
 $y = Cx + Du$  CT/DT

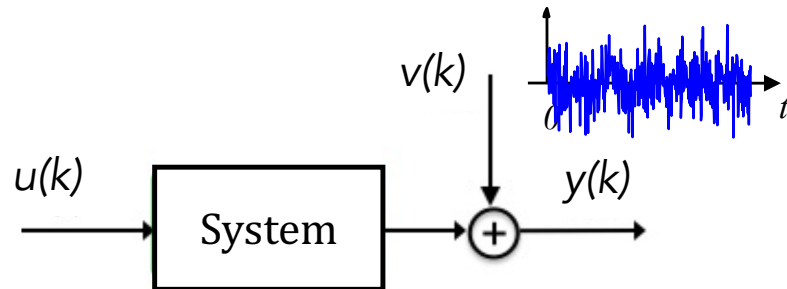
- Most of these model structures can be expressed in continuous-time (CT) and in discrete-time (DT)

## Discrete-time (DT) versus continuous-time (CT) model ?

- ✓ For several decades, the general mainstream identification approach has been to identify DT models from sampled data
  
- ✓ To obtain satisfying results, **DT model identification**
  - Often requires the active participation of an experienced practitioner
  
- ✓ **Direct CT model identification** includes many advantages and is therefore recommended:
  - well adapted to the current sampling situations: **fast or irregular**
  - requires less participation from the user (*inherent pre-filtering*)
  - Easier to interpret in a physical sense
  - makes the *application* of the SYSID procedure much *easier*

## Models for measurement noise

So far: only deterministic models



$u(k)$  = input

$v(k)$  = measurement noise

$y(k)$  = output

- Measurement noise modelled as a stochastic discrete-time signal
- Stochastic models:
  - means, covariances
  - spectra (energy or power)

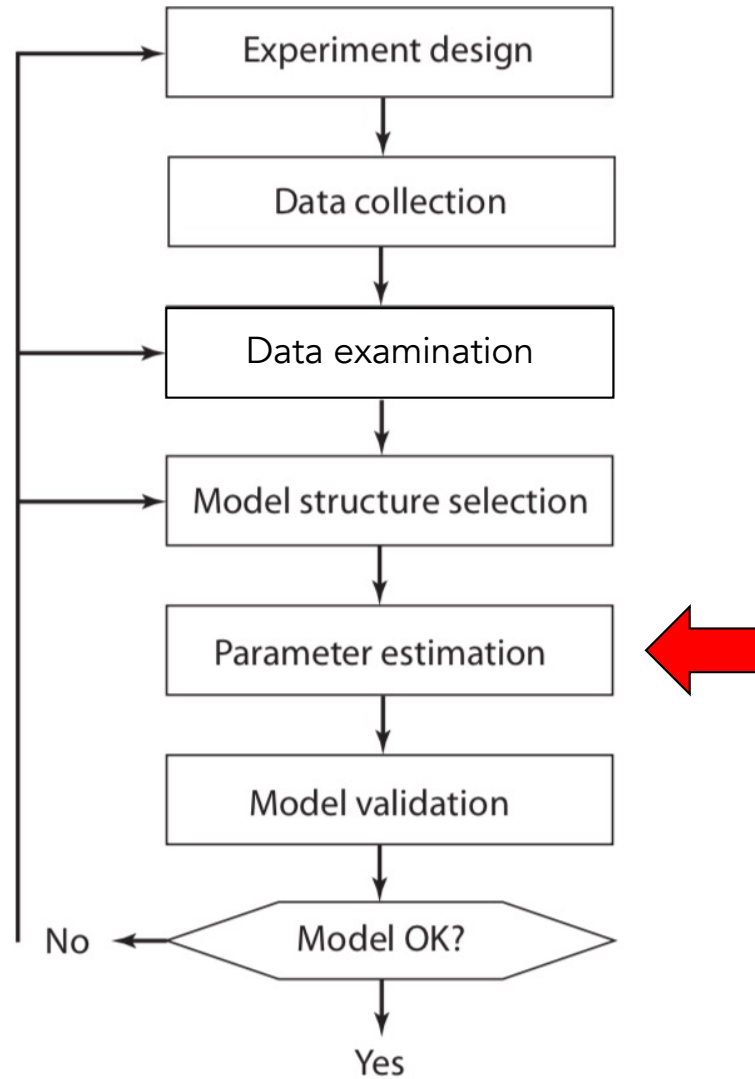
$$v(k) = H(q)e(k) = \frac{C(q^{-1})}{D(q^{-1})} e(k)$$

$e(k)$  is a white Gaussian noise

$q^{-1}$ : delay operator

# Data-driven system identification

## An iterative procedure



## Optimization methods for parameter model estimation

- Common optimization algorithms for estimating the parameters of the models are:
  - Least-squares (LS) method
  - Instrumental variable (IV) method
  - Prediction error method (PEM)
  - Subspace method

## Least squares (LS) method

System:

$$y(t) = \varphi^T(t)\boldsymbol{\theta} + v(t), \quad t = 1, \dots, N$$

$$\mathbf{Y} = \boldsymbol{\Phi}\boldsymbol{\theta} + \mathbf{v}$$

where  $v(t)$  is a disturbance and  $E\mathbf{v} = 0$ ,  $E\mathbf{v}\mathbf{v}^T = \mathbf{R}$ .

**Estimate:**

$$\hat{\boldsymbol{\theta}} = (\boldsymbol{\Phi}^T \boldsymbol{\Phi})^{-1} \boldsymbol{\Phi}^T \mathbf{Y} = \left[ \sum_{t=1}^N \varphi(t) \varphi^T(t) \right]^{-1} \left[ \sum_{t=1}^N \varphi(t) y(t) \right]$$



## Instrumental variable (IV) methods

System:

$$y(t) = \varphi^T(t)\boldsymbol{\theta} + v(t), \quad t = 1, \dots, N$$

where  $v(t)$  is a disturbance with  $E\boldsymbol{v} = 0$ .

**Estimate:** Modify the least squares solution. We get:

$$\hat{\boldsymbol{\theta}} = \left[ \sum_{t=1}^N \boldsymbol{z}(t)\varphi^T(t) \right]^{-1} \left[ \sum_{t=1}^N \boldsymbol{z}(t)y(t) \right]$$

where  $\boldsymbol{z}(t)$  is the vector of instruments.

- ✓ Amongst the different IV versions, one is particularly recommended:
  - **SRIVC**: Simple Refined Instrumental Variable algorithm for Continuous models
    - robust to noise assumptions and algorithmic aspects

## Prediction error methods (PEM)

Idea: Model the noise as well. General methodology applicable to a broad range of models.

The following choices have to be made:

- Choice of model structure. Ex: ARMAX, OE.
- Choice of predictor  $\hat{y}(t|t-1, \boldsymbol{\theta})$ .
- Choice of criterion function. Ex:  $V(\boldsymbol{\theta}) = \frac{1}{N} \sum \varepsilon^2(t, \boldsymbol{\theta})$ .

**Estimate:**

$$\hat{\boldsymbol{\theta}} = \arg \min_{\boldsymbol{\theta}} V(\boldsymbol{\theta})$$

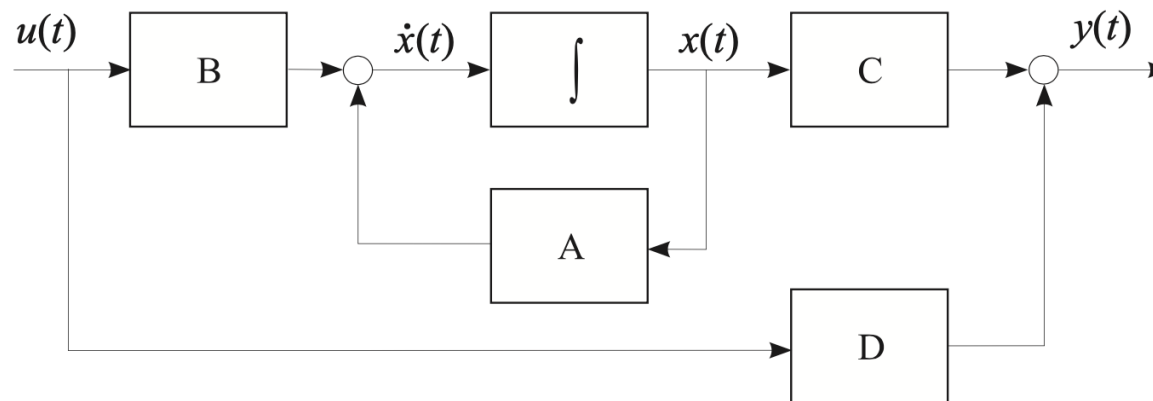
- ✓ The parameters are often estimated by iterative optimization methods like the gradient descent method

## Subspace methods


- Use linear algebra to estimate state-space models
  - Well adapted to multivariable-input multivariable-output (MIMO) systems

$$\dot{x}(t) = Ax(t) + Bu(t)$$

$$y(t) = Cx(t) + Du(t)$$



## Main estimation routines in the SID and CONTSID toolboxes

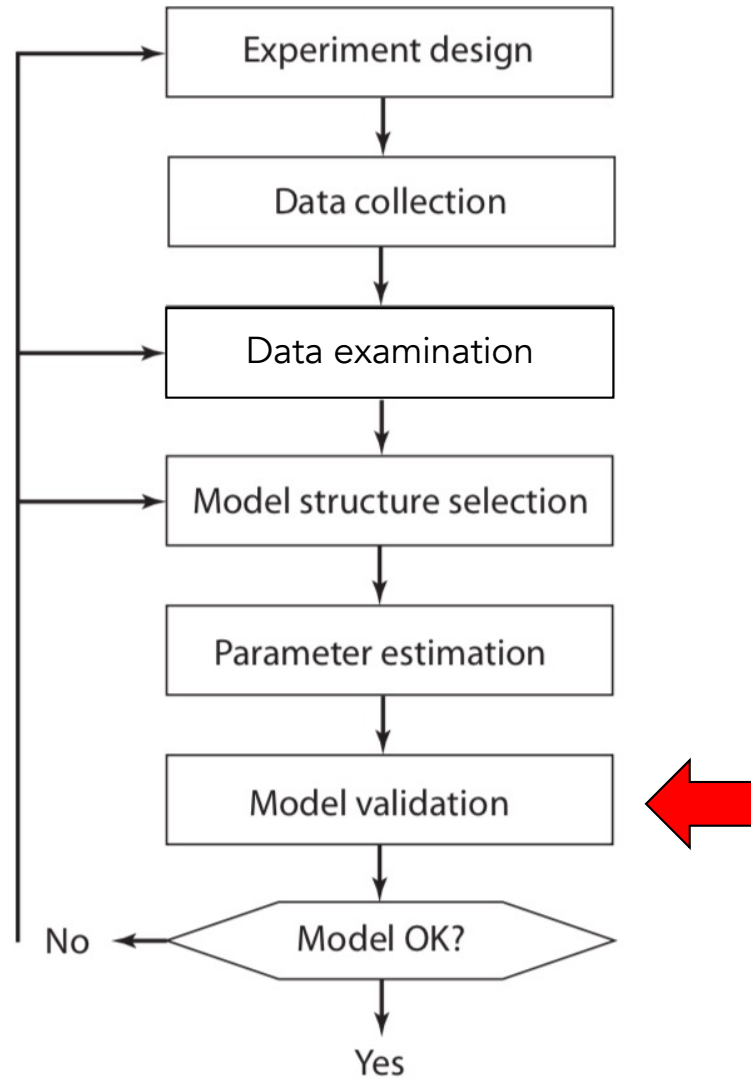


| Commands for Offline Estimation   | Matlab                     | SID toolbox   | CONTSID toolbox                                      |
|---|----------------------------|---|--|
| <b>Model Type</b>   | <b>Estimation Commands</b> |   |  |
| Transfer function models  |                            | tfest   | tfsrvc (/tfrvc/tfcoe)                                |
| Process models (low-order transfer functions expressed in time-constant form) |                            | procest   | procsrvc   |
| Linear input-output polynomial models   |                            | armax (ARMAX and ARIMAX models)<br>arx (ARX and ARIX models)<br>bj (BJ only)<br>iv4 (ARX only)<br>ivx (ARX only)<br>oe (OE only)<br>polyst (for all models) | lssvf (CARX)<br>rvc (CBJ)<br>srvc (COE)<br>coe (COE) |
| State-space models  |                            | n4sid<br>ssest<br>ssregist  | sidgpmf<br>ssivgpmf                                  |

- The majority of these optimization algorithms are iterative

# Data-driven system identification

## An iterative procedure



## Recommended workflow of model validation

1. Compare simulated model output with **estimation data**
  - use a model fit criterion: e.g. the FIT value or the coefficient of determination  $R_T^2$
2. Compare simulated model output with **validation data**
3. **Interpret in a physical sense** the main features of the identified model
  - Steady-state gain, time-constants, time-delay, damping coefficient, natural frequencies
4. **When ever possible**, perform statistical tests on prediction errors
  - Plot the autocorrelation of the residuals and the cross-correlation between the input and the residuals

*This statistical test is often difficult to pass for real-life data*

- *Non-linear effects, parameter varying effects, non stationary noise effects, ...*

## Estimation data versus validation data

Split your data:

### Estimation data

data use for parameter estimation



compute  $\hat{\theta}_N$

### Validation data

data not used for computing  $\hat{\theta}_N$

use them to evaluate the quality of  
the fit



Cross-validation

## Common model accuracy measures

Let  $\hat{y}_k$  be the model prediction. Calculate the *residuals/prediction errors* as

$$\varepsilon_k = y_k - \hat{y}_k$$

Common model accuracy measures are:

- the **FIT** percentage

$$\text{FIT} = 100 \times \left( 1 - \frac{\|y_k - \hat{y}_k\|}{\|y_k - \bar{y}_k\|} \right) \text{ (expressed in \%)}$$

- indicates the agreement between the model and measured output
  - 100% means a perfect fit, and 0 indicates a poor fit

- the **coefficient of determination**

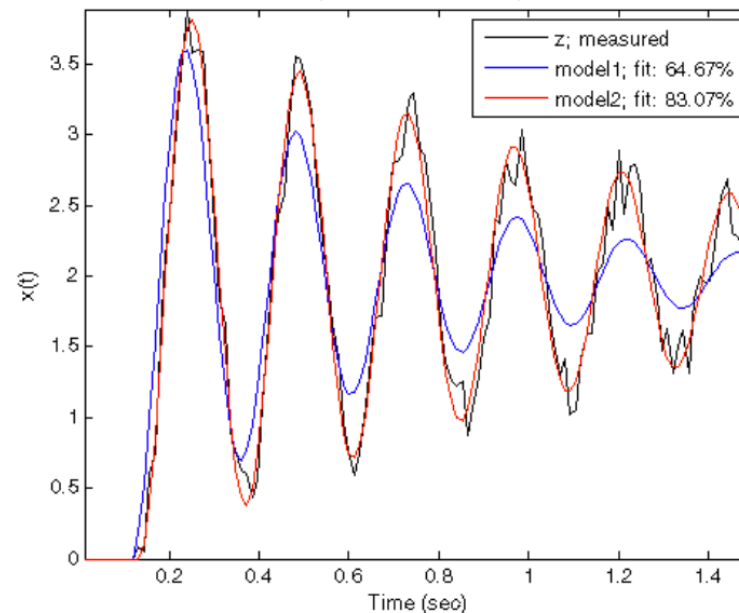
$$R_T^2 = 1 - \frac{\sigma_\varepsilon^2}{\sigma_y^2}$$

- indicates the agreement between the model and measured output
  - the closer  $R_T^2$  to 1, the better the fit



## Comparison of model response to measured response with the estimation data

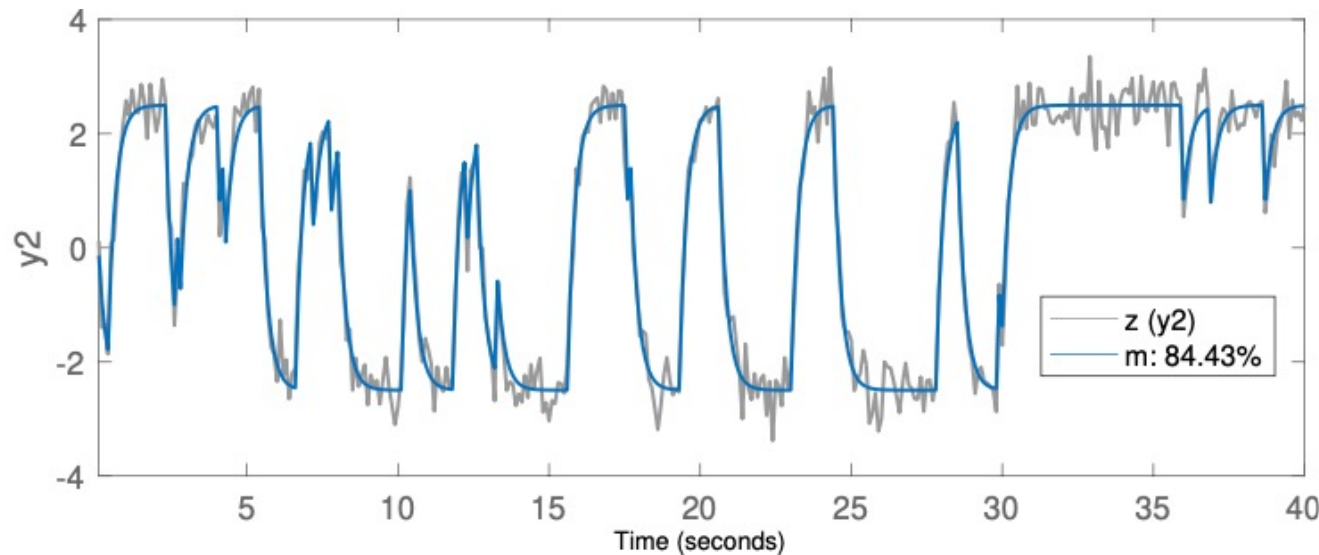
- Typically, you evaluate **first** the quality of models by comparing their model responses to the measured output with the **estimation data**



- model2 above is better than model1 because model2 better fits the data (83% vs. 65%)

## Comparison of model response to measured response with the estimation data: **Warning message**

- Do not be impressed by a good fit to data on a simulation test with the **estimation data**

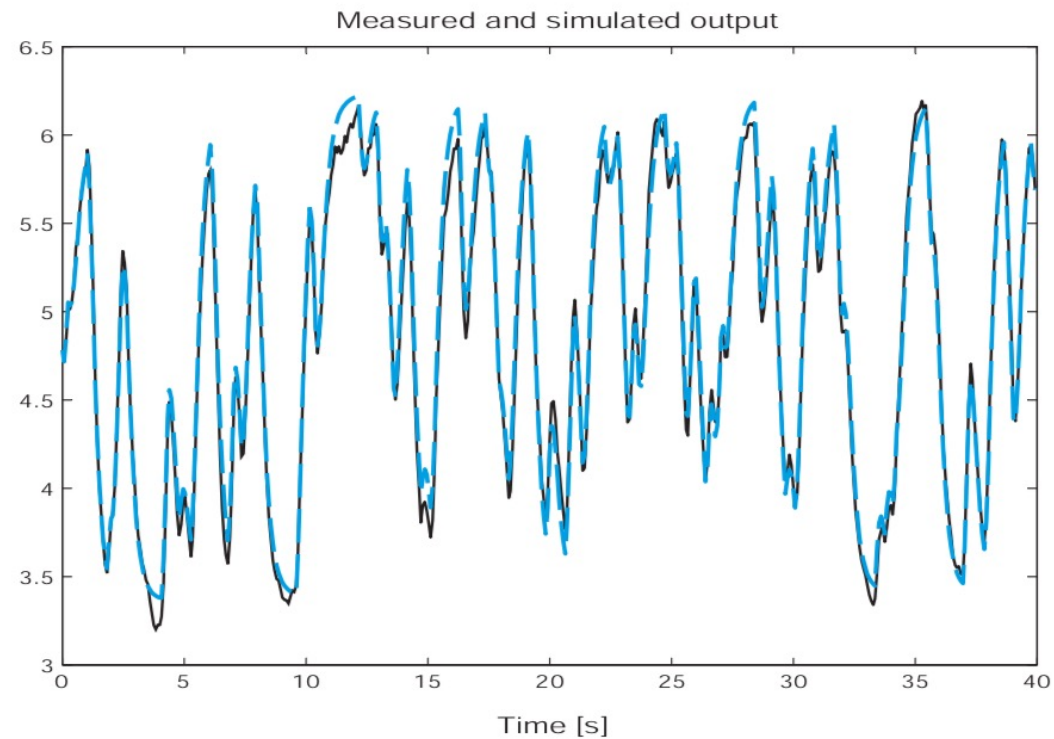


- The **real test** is to see how well the model can reproduce the **validation data**: cross-validation data test

## Comparison of model response to measured response with the validation data

Apply input signal in validation data set to estimated model

Compare simulated output with output stored in validation data set.

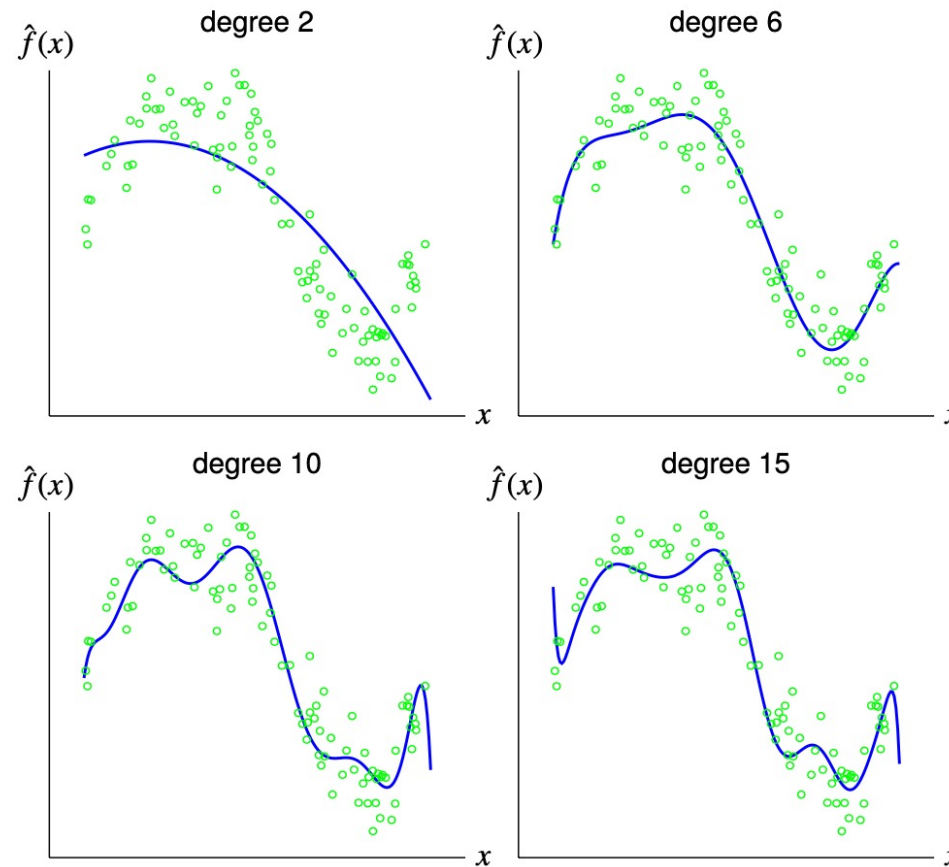


# Choice of the model order

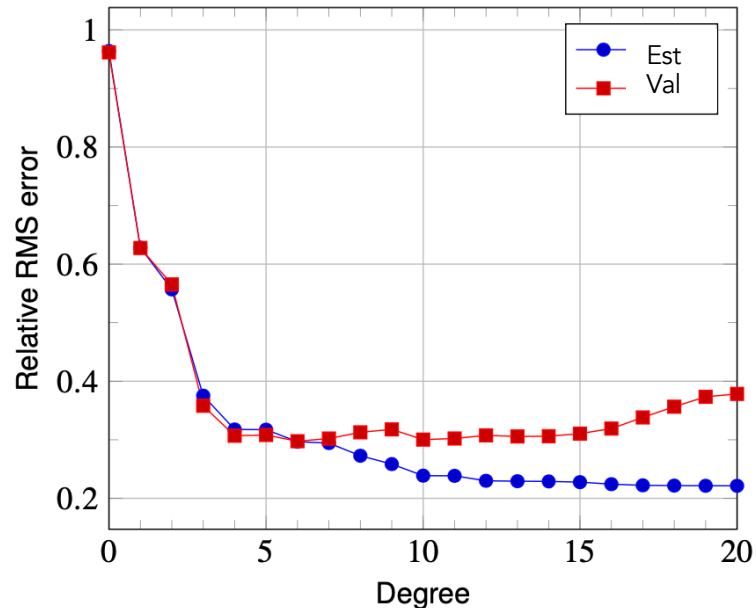
## Example: LS polynomial model fit

$$\hat{f}(x) = \theta_1 + \theta_2 x + \dots + \theta_p x^{p-1}$$

- Model fit using estimation data of 100 noisy points
- Plots below show simulation results on validation data of 100 points



## RMS error versus polynomial degree for both estimation and validation data



### Interpreting results:

- With a 6-th degree polynomial, the relative RMS test error for both estimation and validation data is around 0.3. It is a good sign, in terms of generalization ability, that the estimation and validation errors are similar
- RMS error plot suggest polynomial degree 4, 5, or 6 as reasonable choices

- Too few parameters: model fails to capture the function
- Too many parameters, the model captures the noise
- If **validation** RMS errors are larger than **estimation** RMS errors, model is over-fit
- Methods for avoiding overfit:
  - Keep the model simple
  - Use regularization

## Traditional criteria for model order selection

If fresh validation data is not available (=no cross-validation)

- A loss function  $J(n_p, Z^N)$  is formulated from two functions:
  - one term measuring the model fit based on the loss function
  - one term penalizing the model complexity

$$J(n_p, Z^N) = \log V(\hat{\theta}_{n_p}, Z^N) + \beta(n_p, Z^N)$$

- $\beta(n_p, Z^N)$  is a function which should increase with the model order but decrease to zero when  $N \rightarrow \infty$

## Traditional criteria for model order selection

- Usual approach: pick the model that minimizes
  - AIC (Akaike's Information Criterion)

$$AIC(n_p, Z^N) = \log V(\hat{\theta}_{n_p}, Z^N) + \frac{2n_p}{N}$$

- FPE (Final Prediction Error)

$$FPE(n_p, Z^N) = \frac{1 + \frac{n_p}{N}}{1 - \frac{n_p}{N}} V(\hat{\theta}_{n_p}, Z^N)$$

- YIC (Young's Information Criterion)

$$YIC = \log \left( \frac{\sigma_\varepsilon^2}{\sigma_y^2} \right) + \log \frac{1}{n_p} \sum_{j=1}^{n_p} \frac{\sigma_\varepsilon^2 \hat{p}_{jj}}{\hat{\theta}_j^2}$$

## Choosing among different model orders

- One approach is to fit multiple models to the same data  
*Which is the best model among these ?*
- Assuming the goal is to make good predictions on the validation data
  - Select the model order that has the **best YIC, AIC, FPE** with the **highest associated FIT/  $R_T^2$**  on the **validation data**

| np | m | n | nk | RT2  | YIC   | Niter | FPE  | AIC  |
|----|---|---|----|------|-------|-------|------|------|
| 5  | 2 | 3 | 0  | 0.83 | -8.33 | 10    | 2.35 | 0.85 |
| 6  | 2 | 4 | 0  | 0.92 | -8.19 | 5     | 1.13 | 0.12 |
| 7  | 1 | 5 | 0  | 0.53 | -7.51 | 10    | 6.90 | 1.93 |
| 4  | 1 | 3 | 0  | 0.51 | -3.72 | 10    | 8.40 | 2.12 |
| 5  | 1 | 4 | 0  | 0.03 | -0.89 | 10    | 13.9 | 2.63 |



- If several model candidates achieve similar performance, you should choose the simplest (lowest-order) one among these candidates



## Choice of the model order

### Take-home message

- Choosing the model order is difficult
  - Start with low-order candidate models, and so on. You can compare higher order models against these
  - Compare candidate models using validation data
  - Increasing the model order will always increase the FIT/  $R_T^2$  on the estimation data, but the important question is whether or not it substantially increases the FIT/  $R_T^2$  on the validation data sets
  - Increasing the model order can easily lead to over-fit. To avoid the over-fit:
    - keep the model simple (low-order)
    - use information criteria
    - use regularization

## Software available

Most of the theory covered in the course for continuous-time model identification is implemented in:

- the CONTinuous-Time System IDentification (**CONTSID**) toolbox for Matlab



- *A lot can be learned from the demos available*

# CONTinuous-Time System Identification

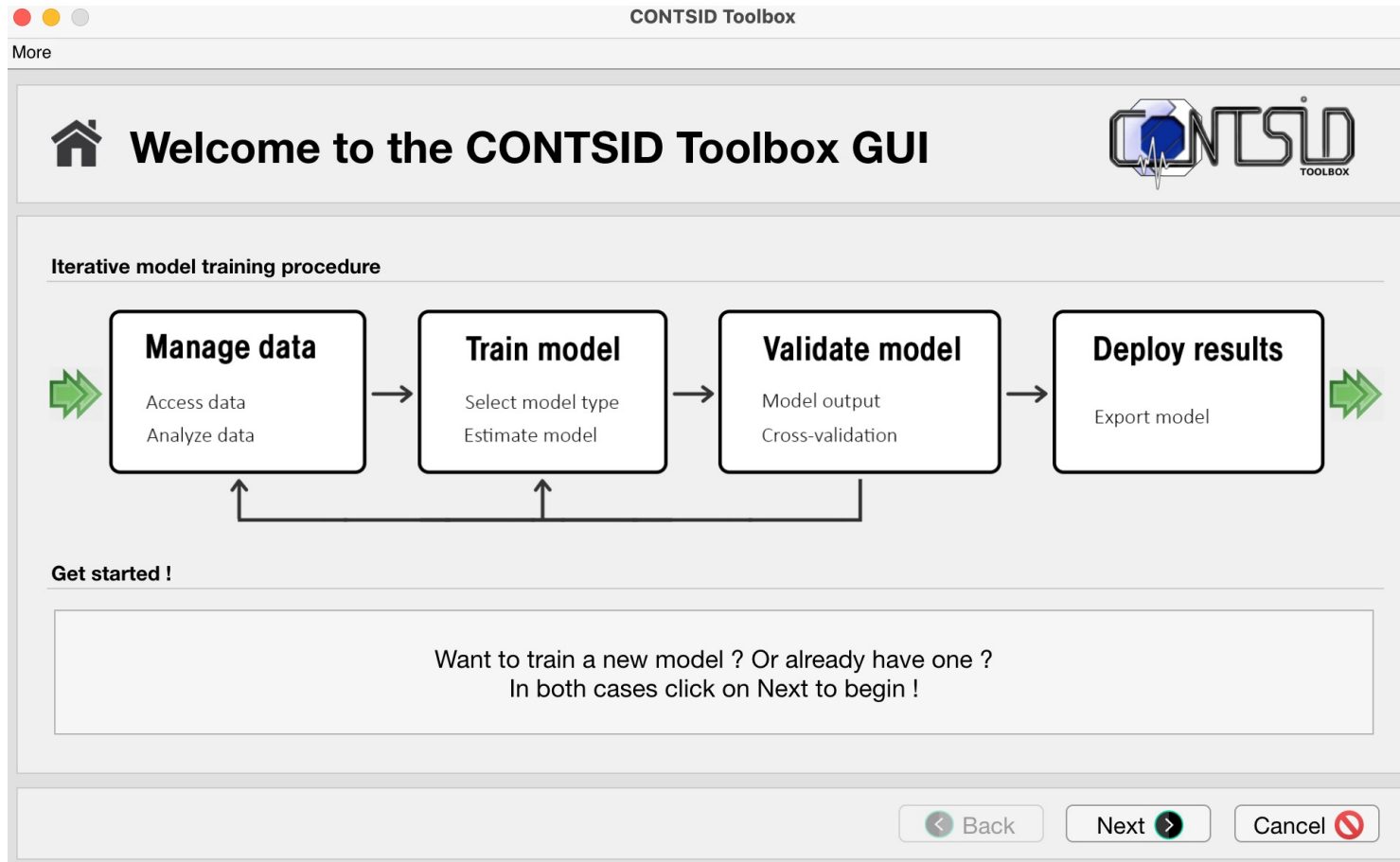
## Key features

- ✓ Supports *direct CT model identification* approaches
  - *Basic linear CT* models
    - Transfer function and state-space models
    - regularly and irregularly sampled data
    - Time-domain or frequency domain data
  - *More advanced black-box* models
    - *On-line, errors-in-variables* and *closed-loop* situations
    - Nonlinear systems: *block-structured, LPV* or *LTV* models
- May be seen as an add-on to the Matlab System Identification toolbox
  - Uses the same syntax, data and model objects
    - M=**procsrivc**(data,'P1')
    - M=**tfsrivc**(data,np,nz)
- P-coded version freely available from:  
[www.cran.univ-lorraine.fr/contsid](http://www.cran.univ-lorraine.fr/contsid)

## Main features of the latest version 7.5

- ✓ Core of the routines mainly based on **refined optimal IV**: *SRIVC*
  - CONTSID includes also a few PEM and subspace-based methods
- *SRIVC-based parameter estimation schemes for more advanced identification*
  - *Polynomial models with known delay* : *SRIVC*
  - *Simple process models with known/unknown delay*: *PROCSRIVC*
  - *Transfer function + known/unknown delay models*: *TFSRIVC*
  - *Transfer function + delay + noise models*: *TFRIVC*
  - *Time Varying Parameter models*: recursive *RSRIVC*
  - *Closed-loop identification*: *CLSRIVC*
  - *LPV models*: *LPVSRIVC*
  - *Hammerstein models*: *HSRIVC*, ...
- Includes a flexible *GUI* and many *demos* to illustrate its use and the recent developments

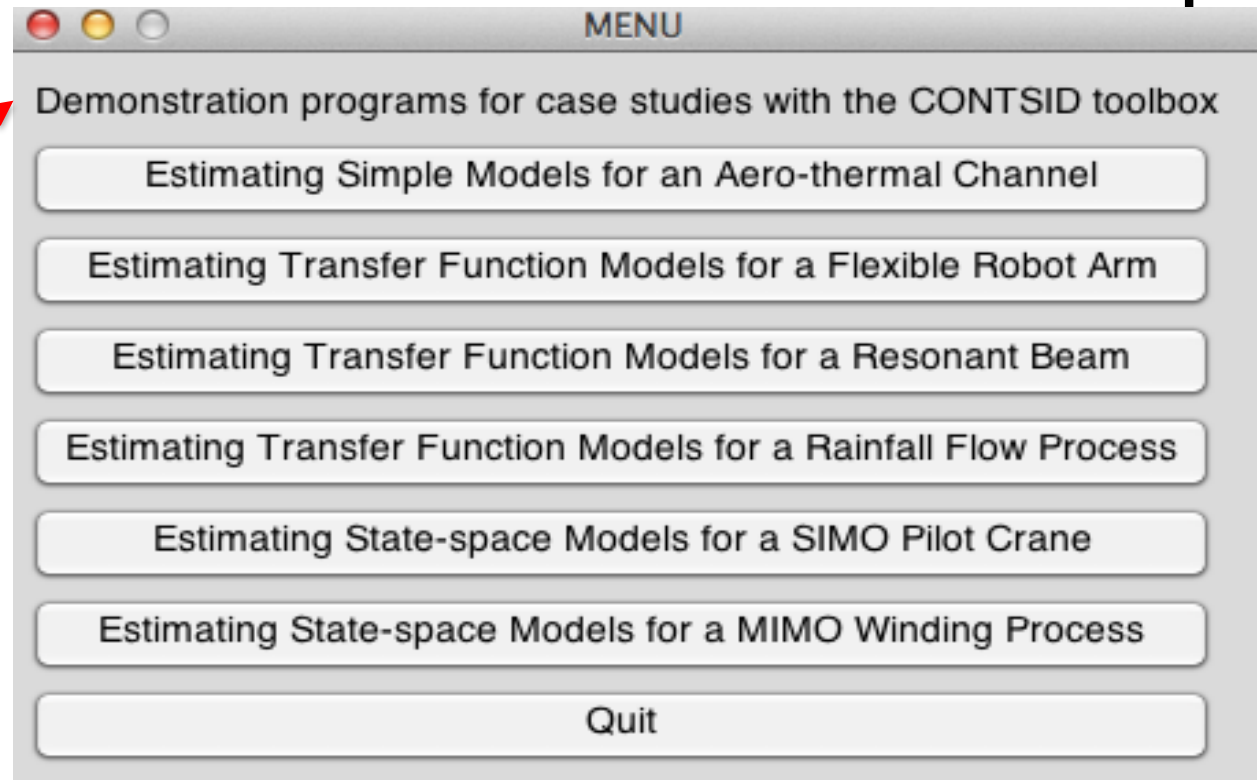
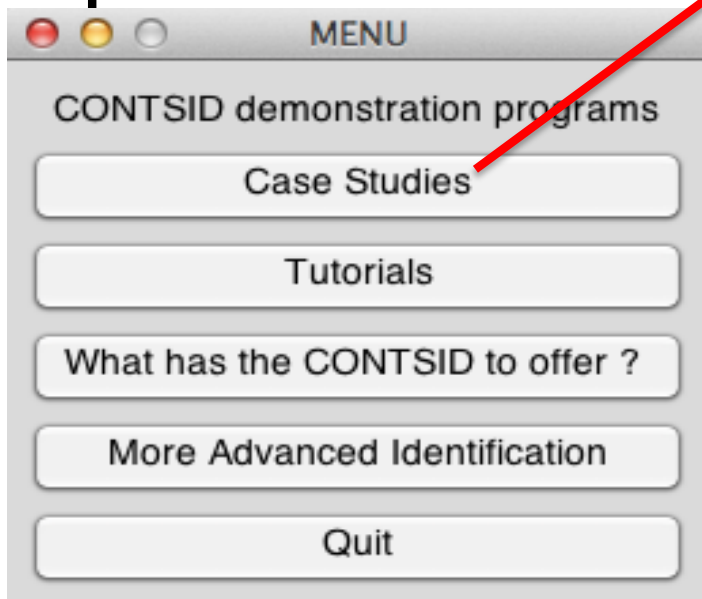
## CONTSID graphical user interface



- Allows the user to easily apply the iterative process of system identification

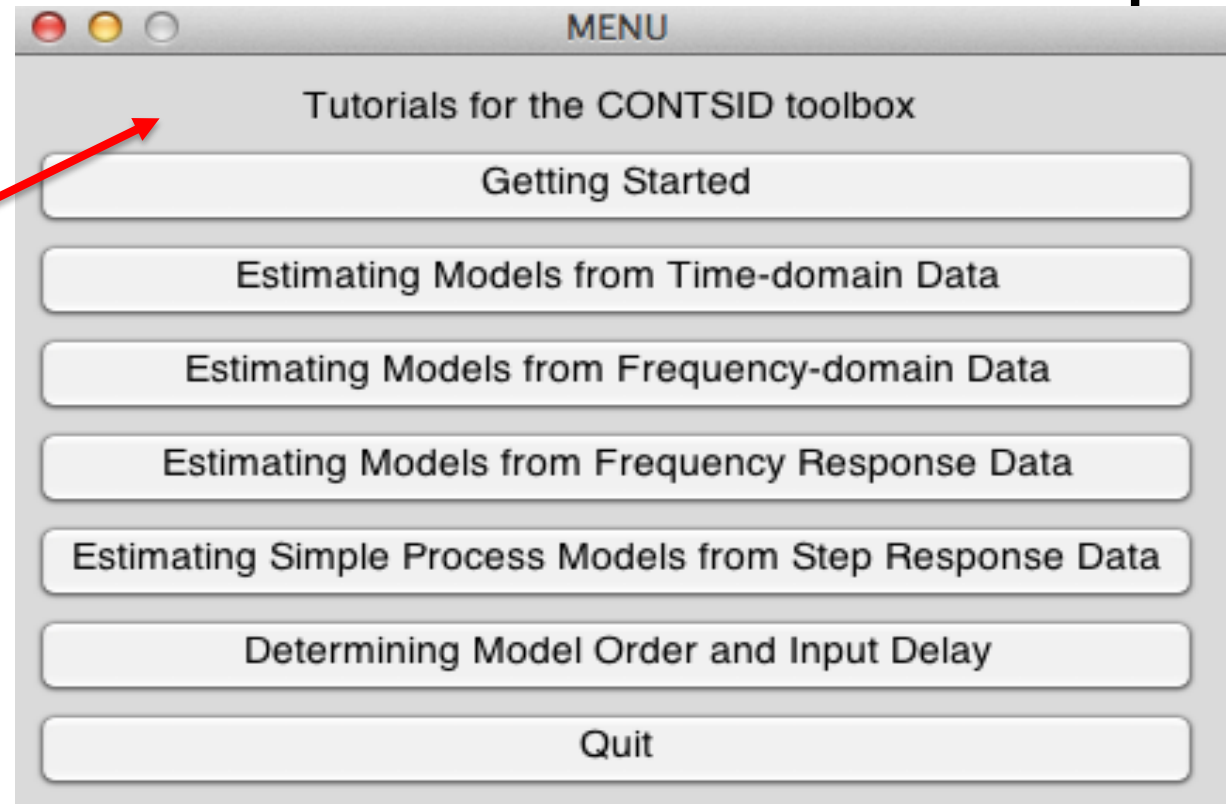
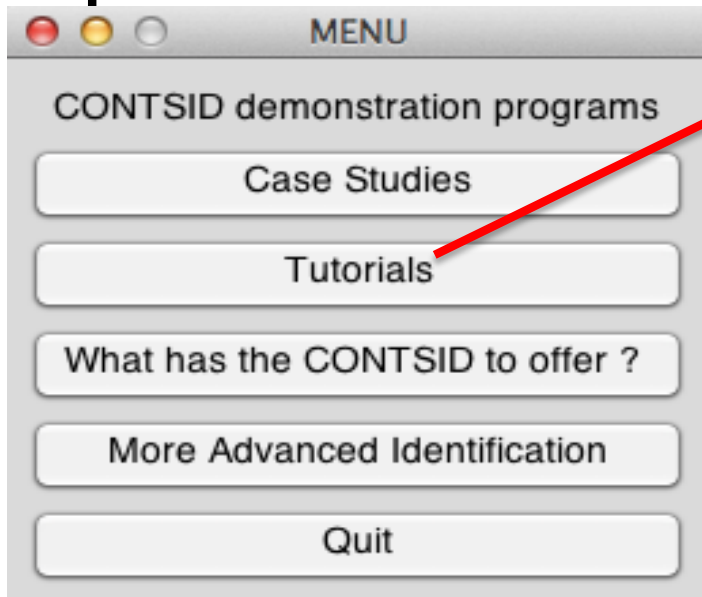
# CONTSID toolbox – Demonstration programs

>>contsid\_demo



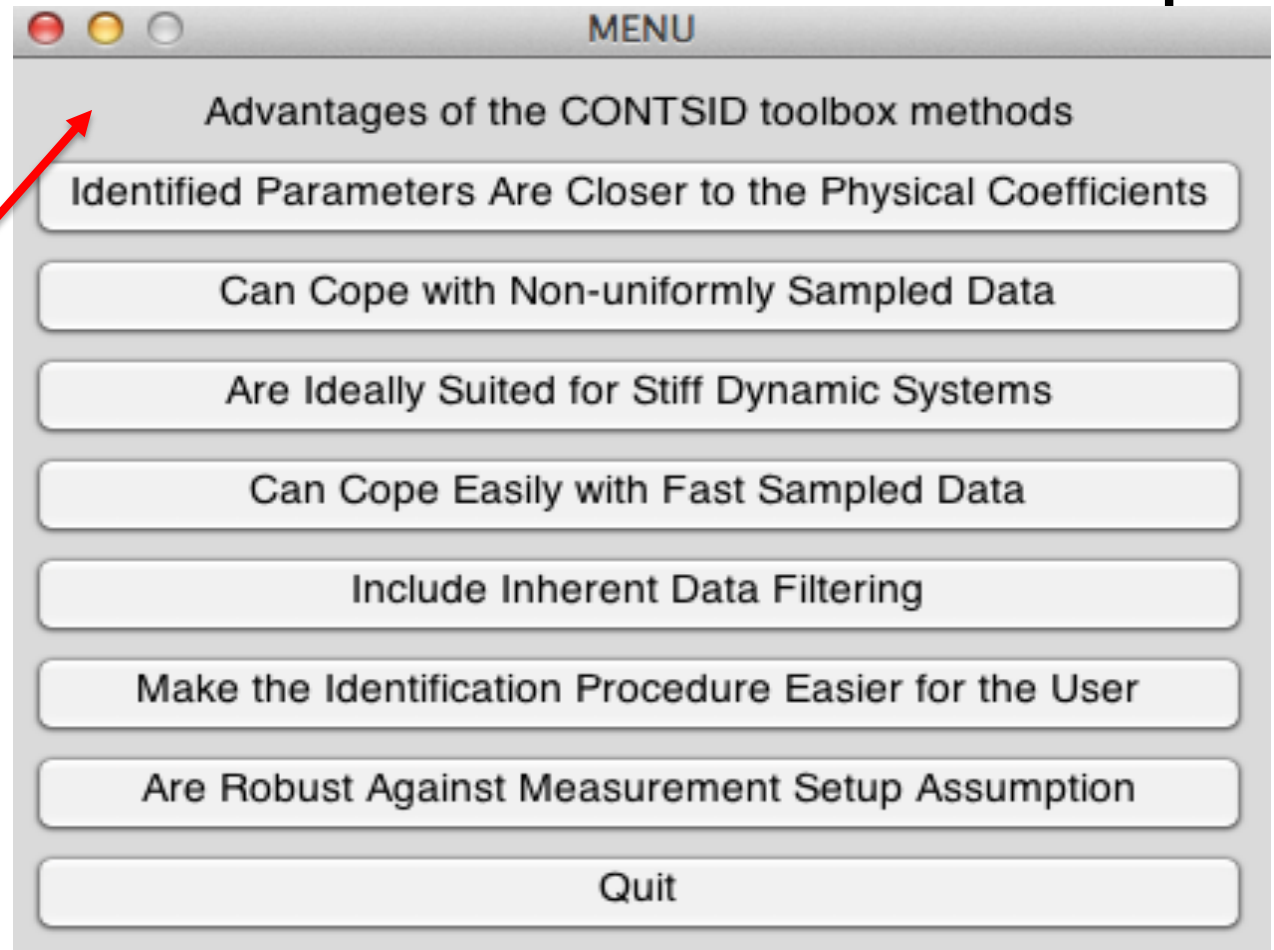
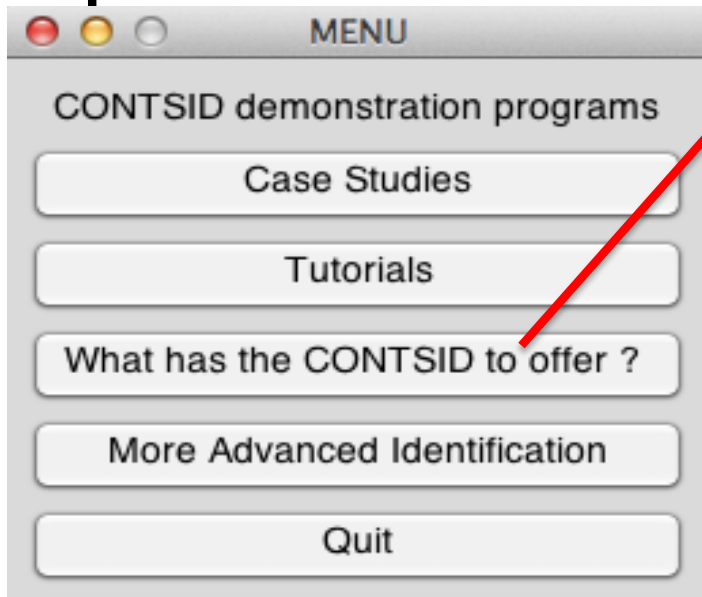
# CONTSID toolbox – Demonstration programs

>>contsid\_demo



# CONTSID toolbox – Demonstration programs

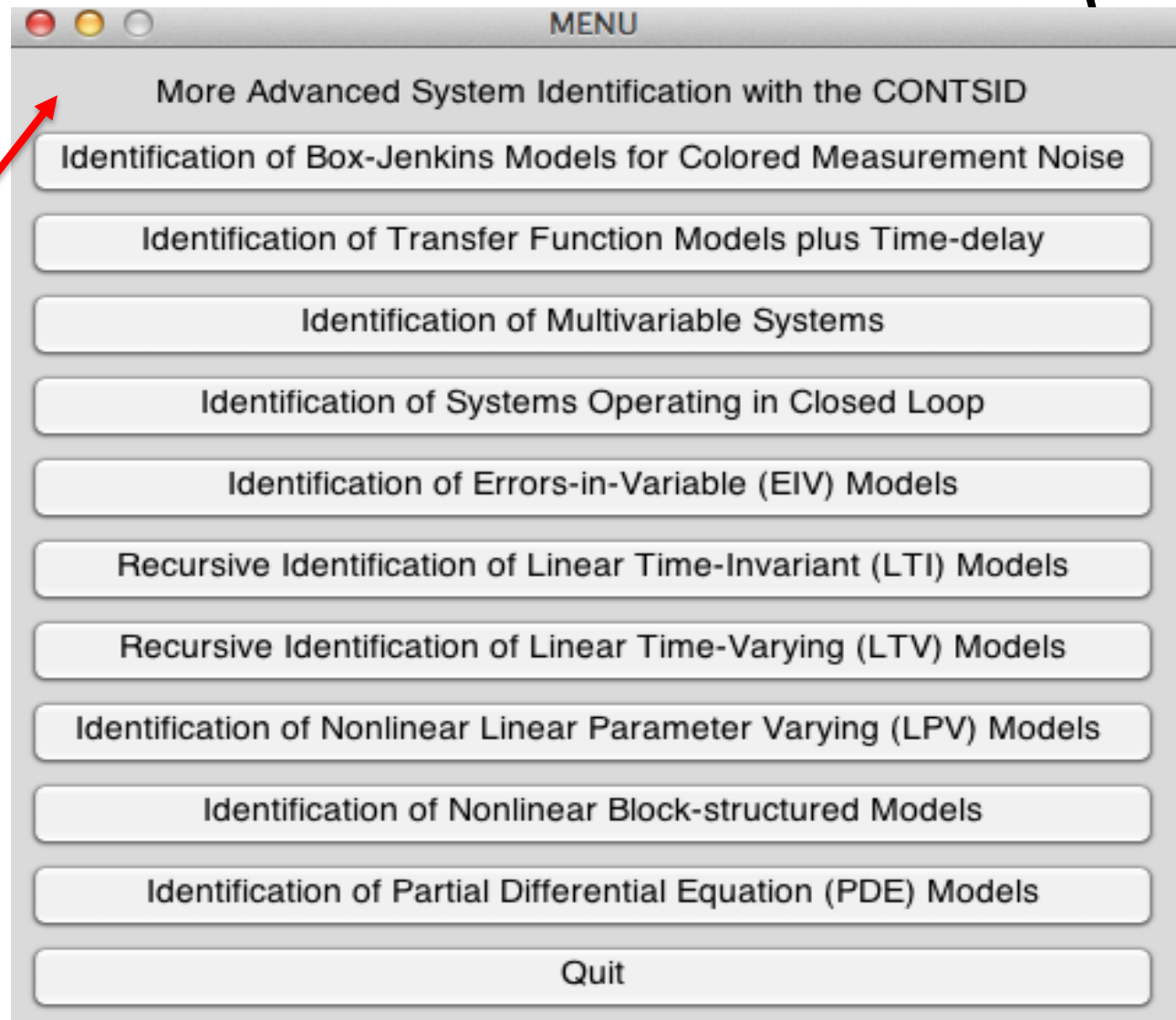
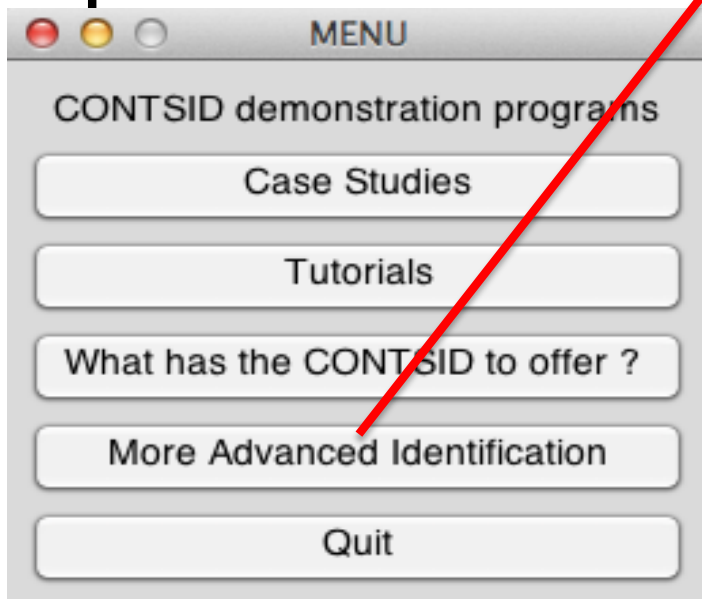
>>contsid\_demo





# CONTSID toolbox – Demonstration programs

>>contsid\_demo



## The simulated Rao-Garnier benchmark

See [contsid\\_tutorial1.m](#)

- 4th order simulated system  $p=d/dt$

$$G_o(p) = \frac{K(1-Tp)}{\left(\frac{p^2}{\omega_{n,1}^2} + \frac{2\zeta_1}{\omega_{n,1}}p + 1\right)\left(\frac{p^2}{\omega_{n,2}^2} + \frac{2\zeta_2}{\omega_{n,2}}p + 1\right)}$$

$$= \frac{-6400p + 1600}{p^4 + 5p^3 + 408p^2 + 416p + 1600}$$

$$K = 1$$

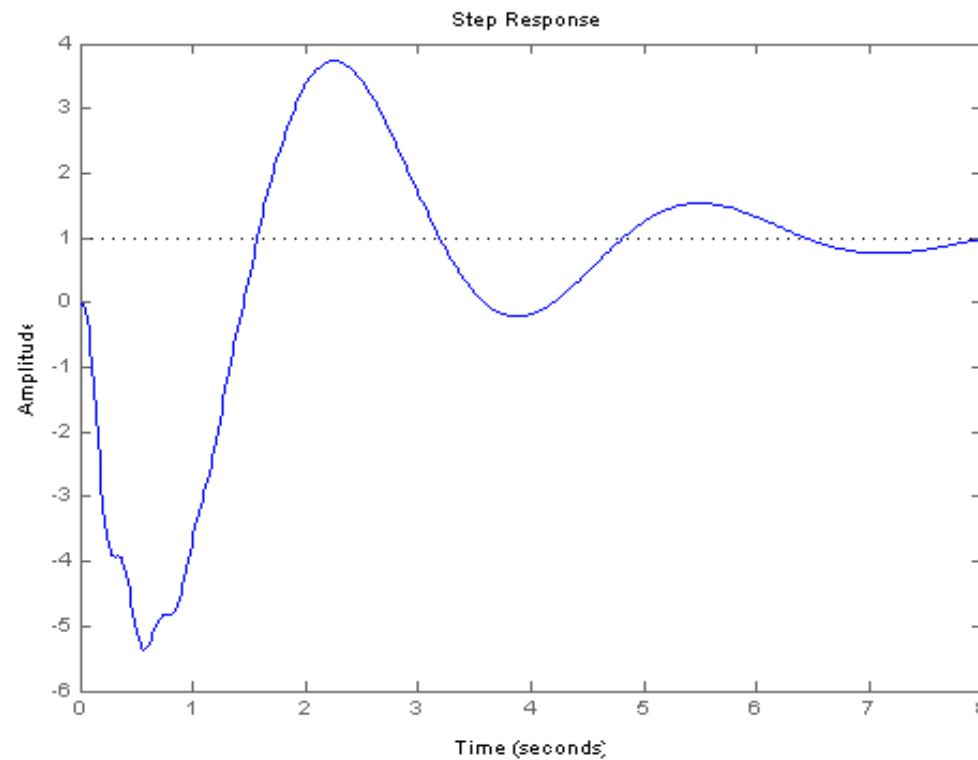
$$\omega_{n,1} = 2 \text{ rad / s}; \quad \omega_{n,2} = 20 \text{ rad / s}$$

$$\zeta_1 = 0.1; \quad \zeta_2 = 0.25$$

- ✓ 1 unstable zero
- ✓ 2 pseudo-oscillatory modes

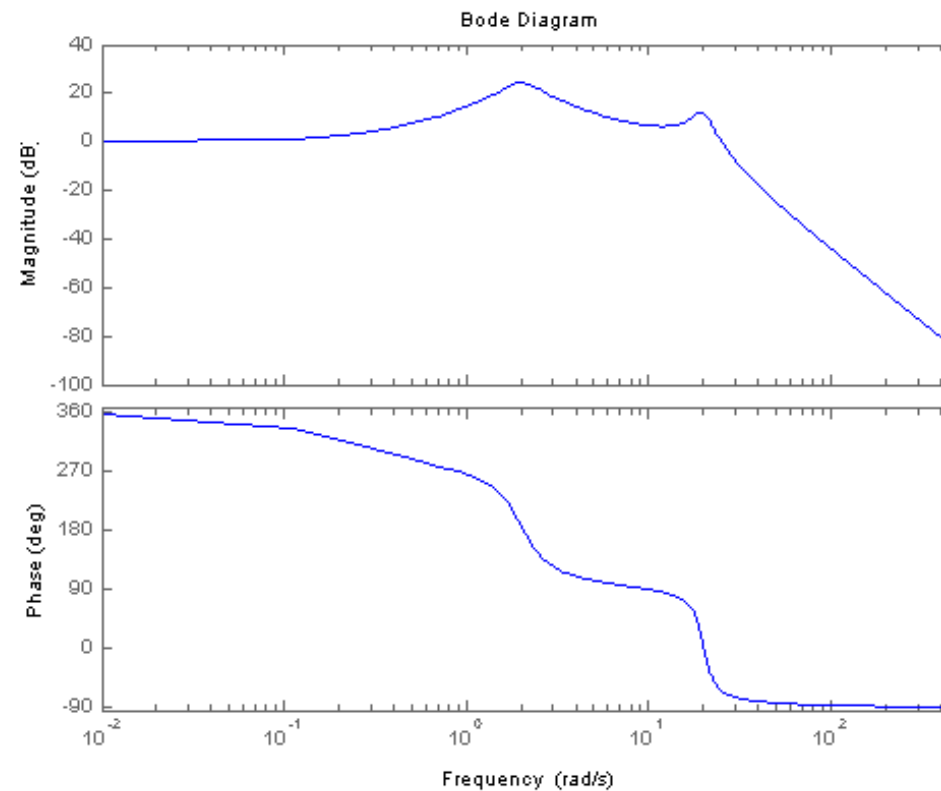
## The simulated *Rao-Garnier* benchmark

- The step response

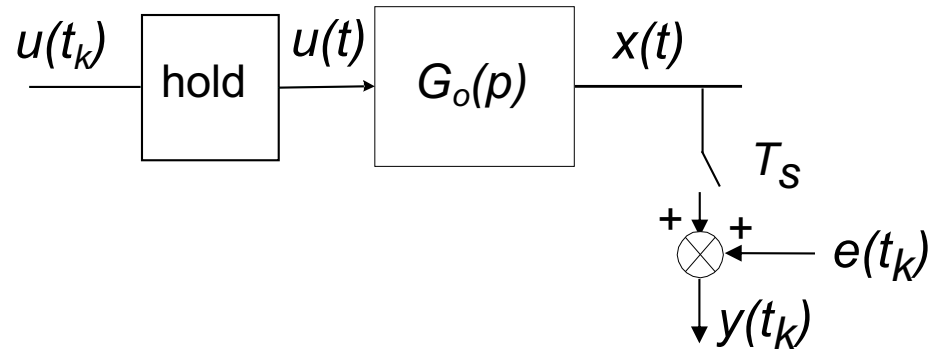


## The *Rao-Garnier* benchmark

- The Bode diagram

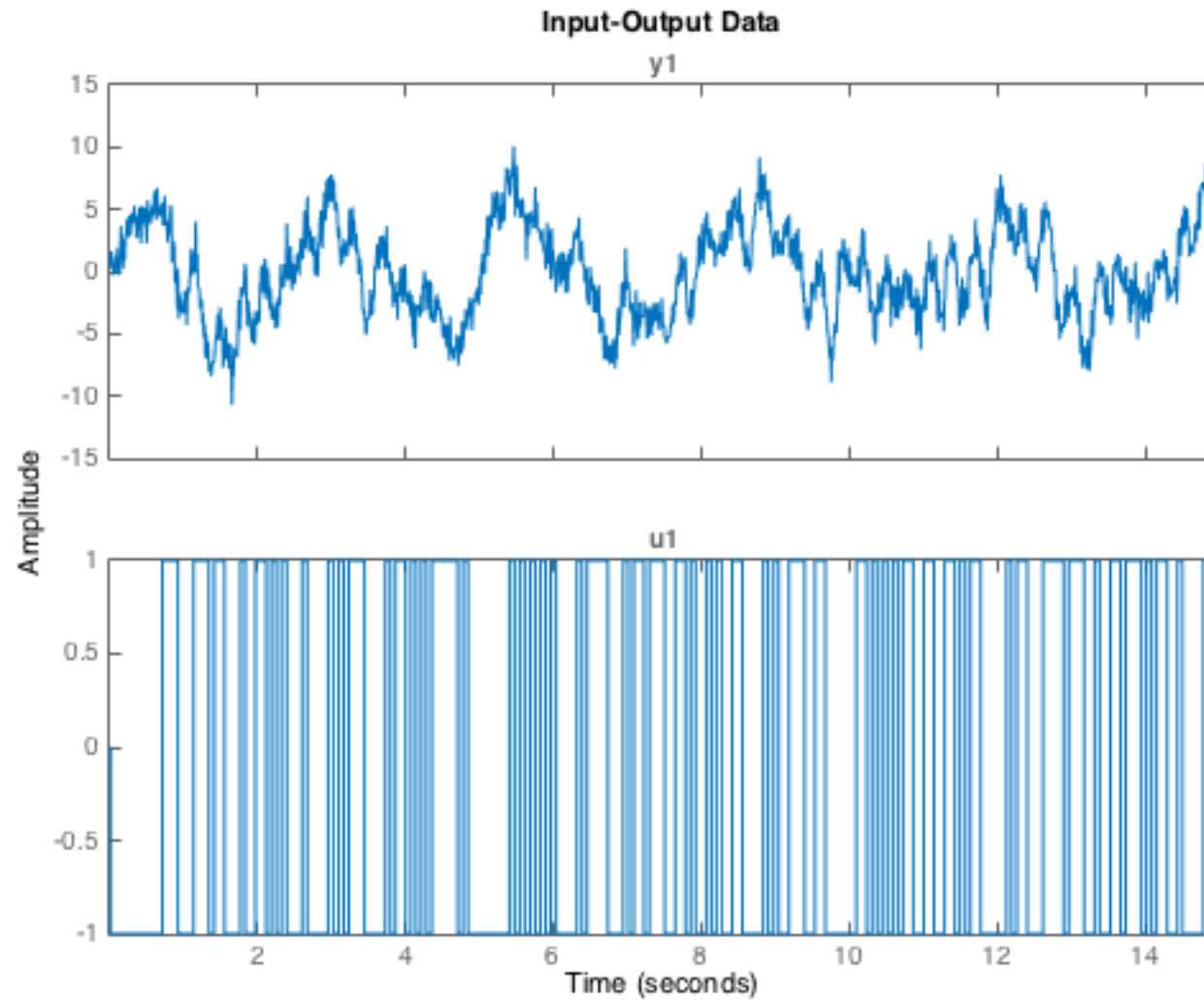


## Simulation setup



- $u(t_k)$ : PRBS (respects the ZOH assumption)
- $T_s = 10 \text{ ms}$ 
  - $f_s \approx 10$  times the system bandwidth, an often given rule
- $e(t_k)$  : DT white noise, SNR=10 dB

# I/O data – Rao-Garnier benchmark



## Model order determination

Different model structures in the range  $[m \ n \ nk] = [1 \ 3 \ 0]$  to  $[2 \ 5 \ 0]$  have been computed for the given data set

5 best models sorted according to YIC

$$G(s) = \frac{\sum_{i=0}^m b_i s^i}{\sum_{i=0}^n a_i s^i} e^{-n_k T} s$$

| np | m | n | nk | RT2  | YIC   | Niter | FPE  | AIC  |
|----|---|---|----|------|-------|-------|------|------|
| 5  | 2 | 3 | 0  | 0.83 | -8.33 | 10    | 2.35 | 0.85 |
| 6  | 2 | 4 | 0  | 0.92 | -8.19 | 5     | 1.13 | 0.12 |
| 7  | 1 | 5 | 0  | 0.53 | -7.51 | 10    | 6.90 | 1.93 |
| 4  | 1 | 3 | 0  | 0.51 | -3.72 | 10    | 8.40 | 2.12 |
| 5  | 1 | 4 | 0  | 0.03 | -0.89 | 10    | 13.9 | 2.63 |

The second model with  $[m \ n \ nk] = [2 \ 4 \ 0]$  seems to be quite clear cut  
It has the second most negative  $YIC = -8.19$ , with the highest  $R^2_T = 0.92$

See *tfsrvcstruc* and *selcstruc* in the *CONTSID* toolbox

# Estimated model parameters and their uncertainties via TFSRIVC

```
>> present(Mtfsrivic)
```

```
Mtfsrivic =
```

```
From input "u1" to output "y1":
```

```
          -6480 (+/- 131.8) s + 1880 (+/- 251.4)
```

```
-----  
s^4 + 5.382 (+/- 0.2094) s^3 + 407.6 (+/- 3.565) s^2 + 424.2 (+/- 11.53) s + 1566 (+/- 26.24)
```

```
Continuous-time identified transfer function.
```

```
Parameterization:
```

```
Number of poles: 4   Number of zeros: 1
```

```
Number of free coefficients: 6
```

```
Use "tfdata", "getpvec", "getcov" for parameters and their uncertainties.
```

```
Status:
```

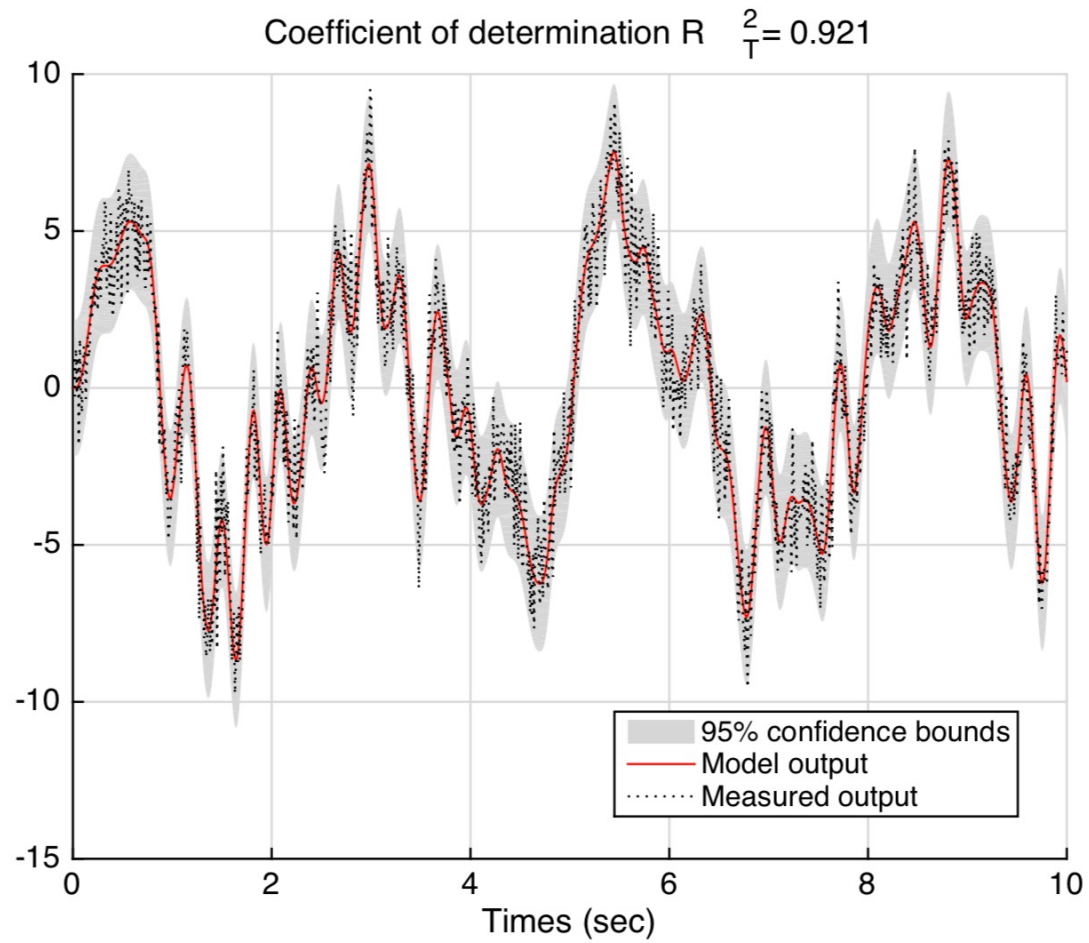
```
Estimated using Contsid TFSRIVC method on time domain data.
```

```
Fit to estimation data: 71.91
```

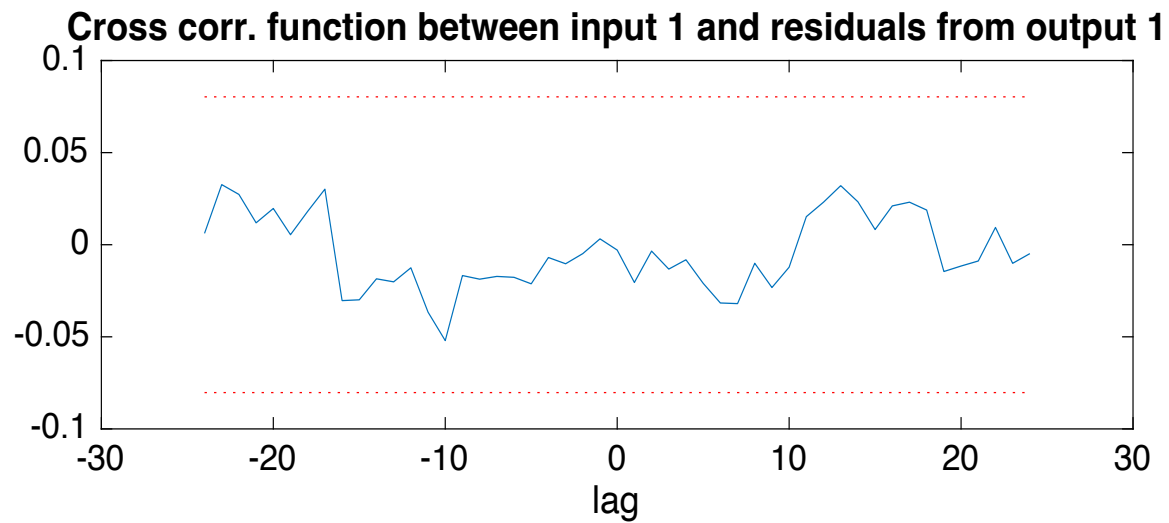
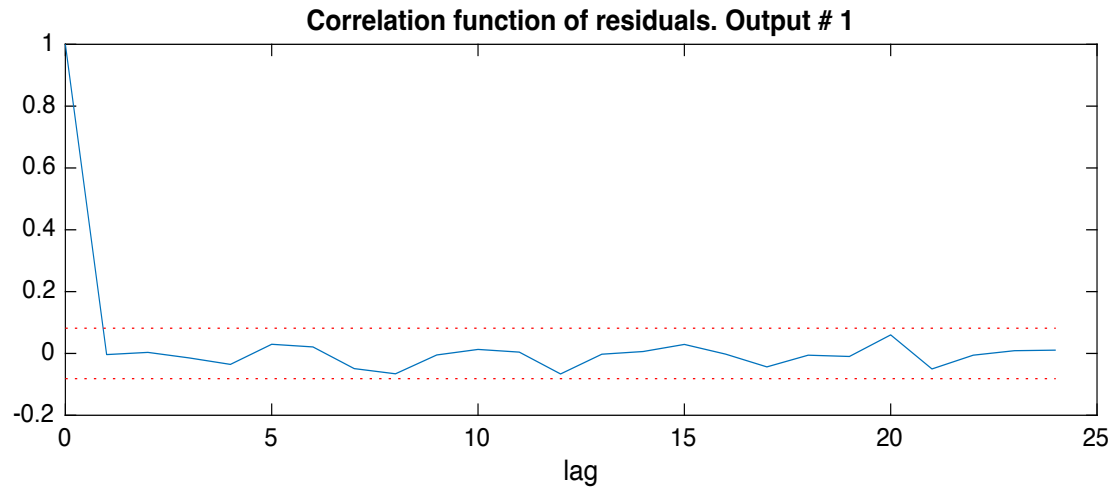
```
FPE: 1.140e+00, MSE 1.124e+00
```



# Comparison of model and measured output



# Statistical tests on residuals



## Data-driven linear model identification/learning

### Takehome message

- Several choices by the practitioners have to be made and often revised
  - Many of these choices have to be taken with the intended model use in mind and thus have a **subjective** flavour
  - The more **a priori knowledge from physics** you can exploit in the SYSID workflow, the better
  - **Interpretability** of the identified models in **meaningful physical** terms is **essential**
- Always keep in mind
  - Good models cannot be obtained from bad data !
  - All models are approximation of the real system !
  - Good models are simple !