# *Identification of linear black-box models*

------

*Practical aspects*

Hugues GARNIER

hugues.garnier@univ-lorraine.fr

## Model identification of linear time-invariant systems

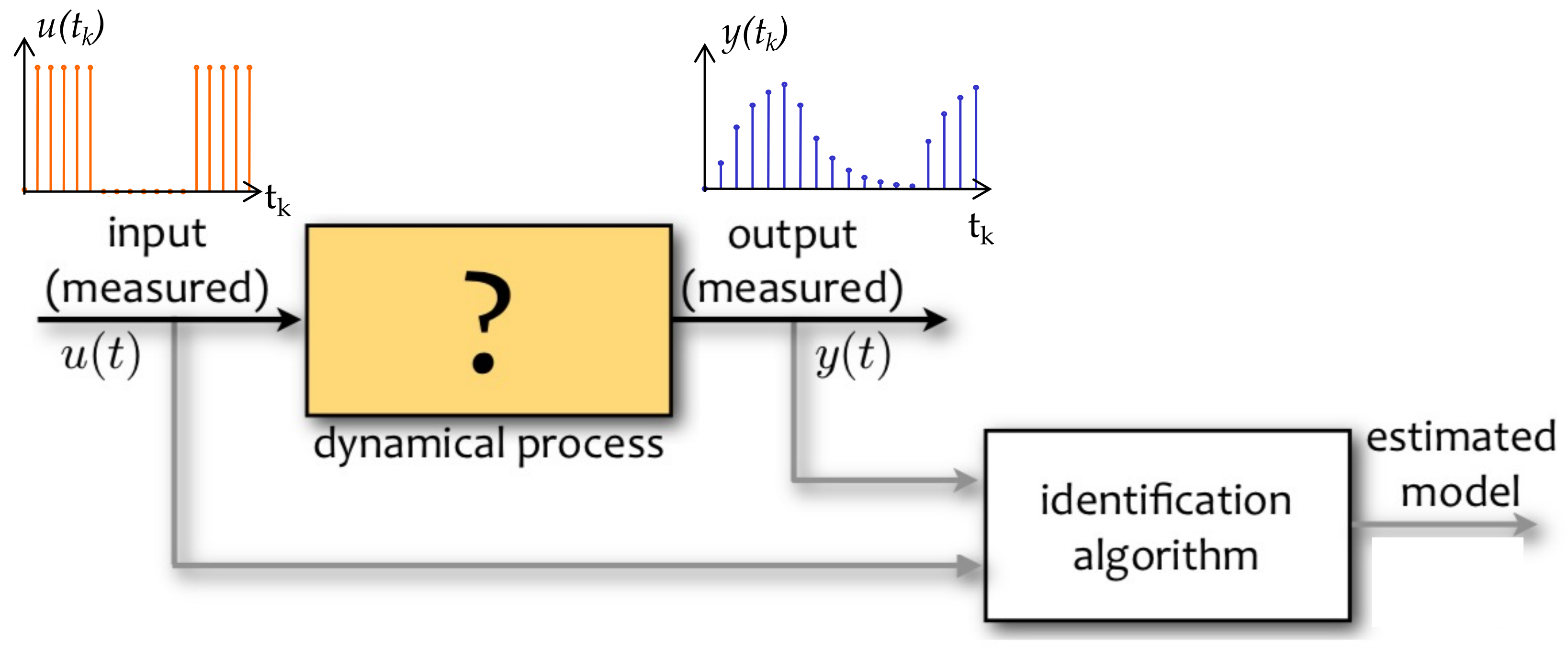
- For ***control design and analysis***, Linear Time-Invariant (LTI) models have been hugely important, mainly motivated by
  - their simplicity
  - their performance and robustness properties are well understood
- Classical SYSID methods of linear models
  - share these properties in many regards
  - have a relatively low computational complexity
  - have strong systems-theoretical background, with well developed concepts such as identifiability, input design, informative data selection

➢ Always try FIRST data-driven methods for identifying
*linear time-invariant models*

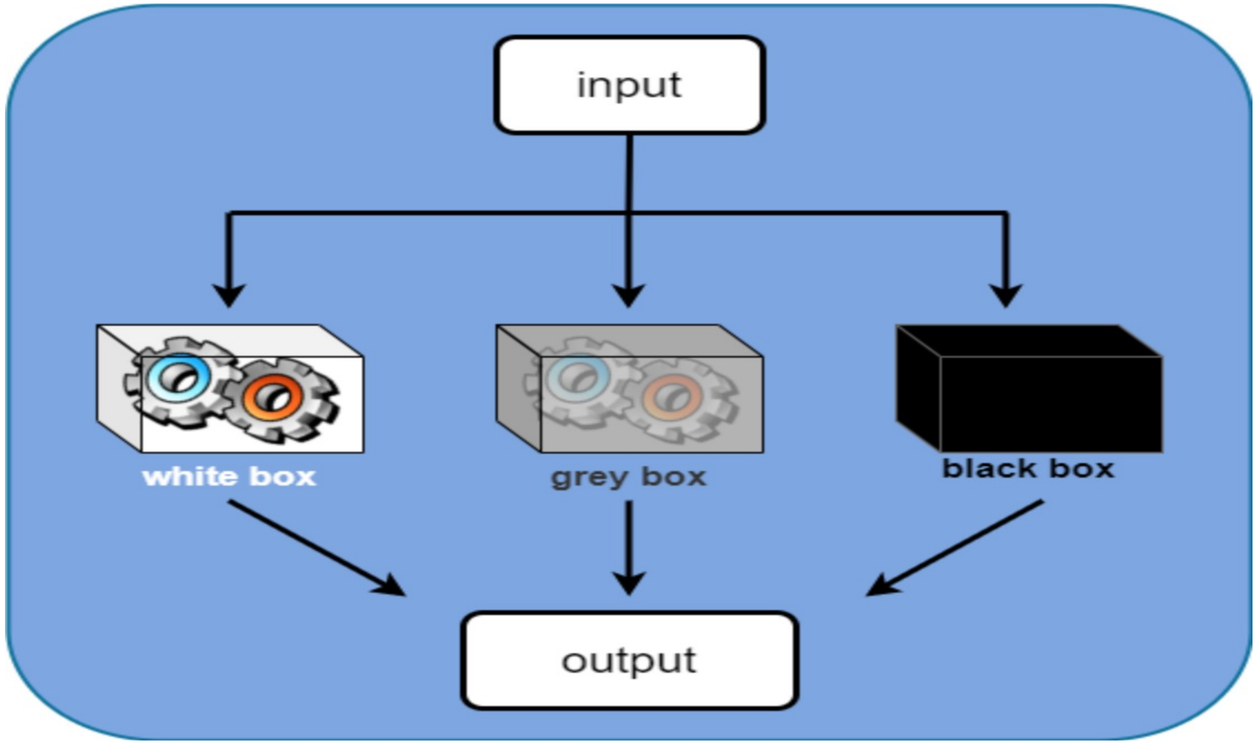
➢ If the fit is not good, try to estimate nonlinear or time-varying models

## 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



## A palette of color for the models (structure + parameters)



**White-box model**

- Structure built from Physics
- Parameters a priori known

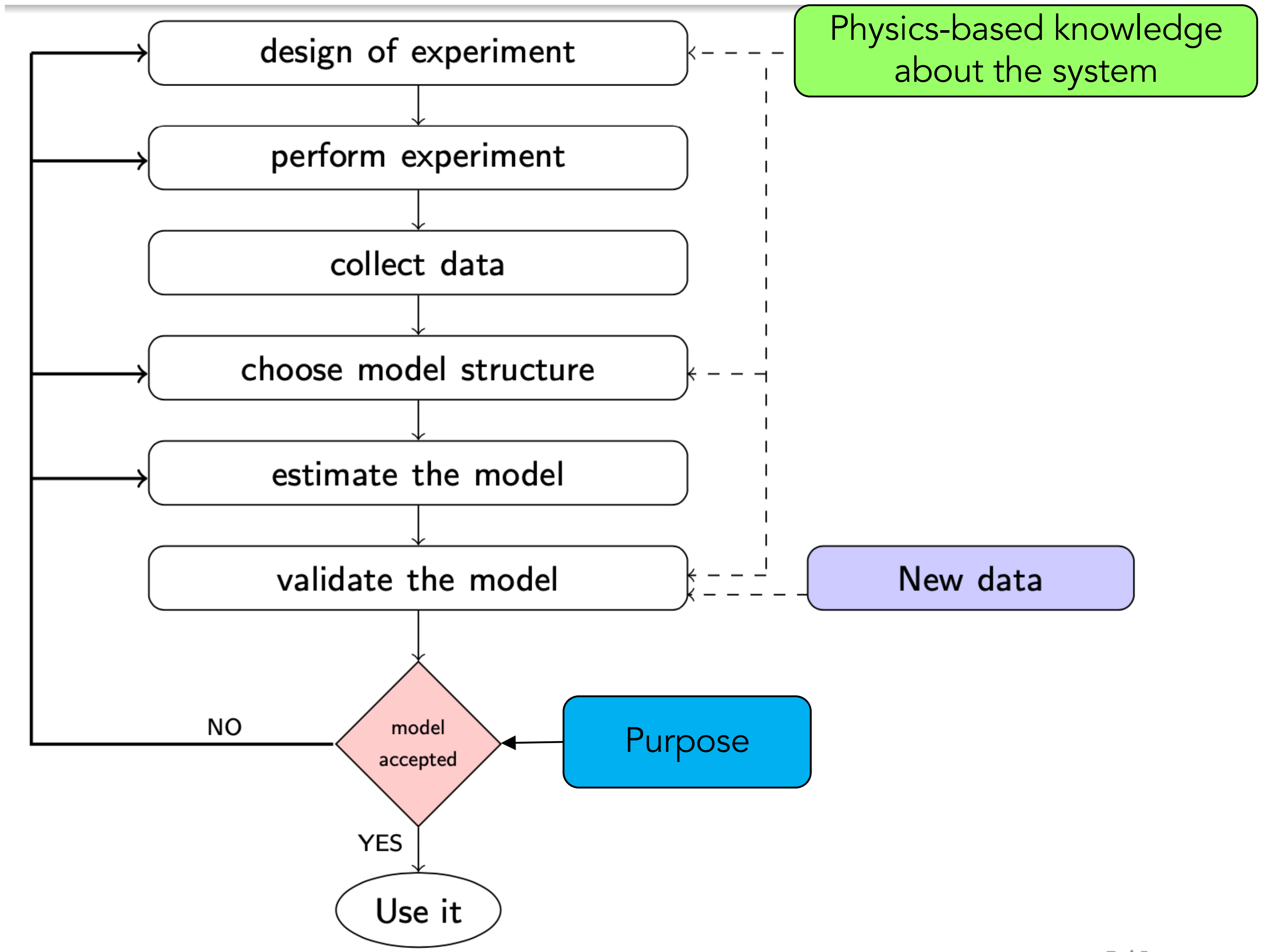
**Grey-box model**

- Structure built from Physics
- Some parameters are estimated from data

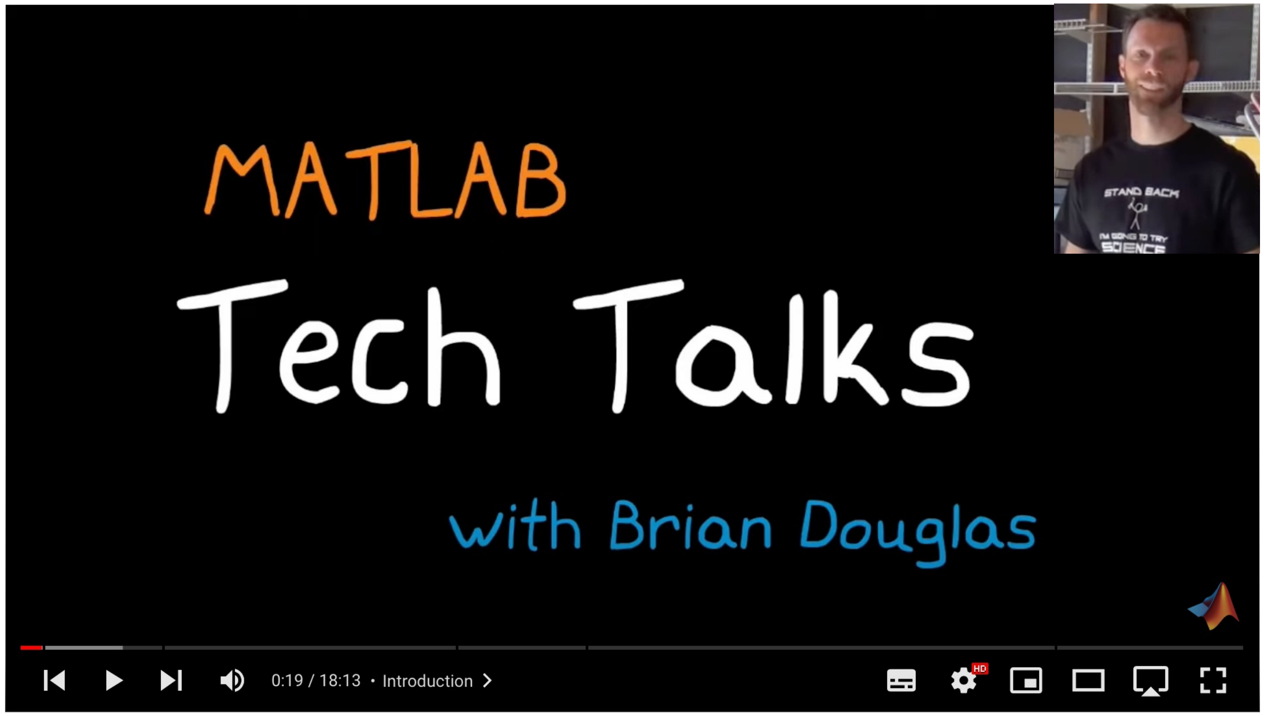
**Black-box model**

- Structure built mainly from data
- Parameters estimated from data

## The iterative system identification workflow



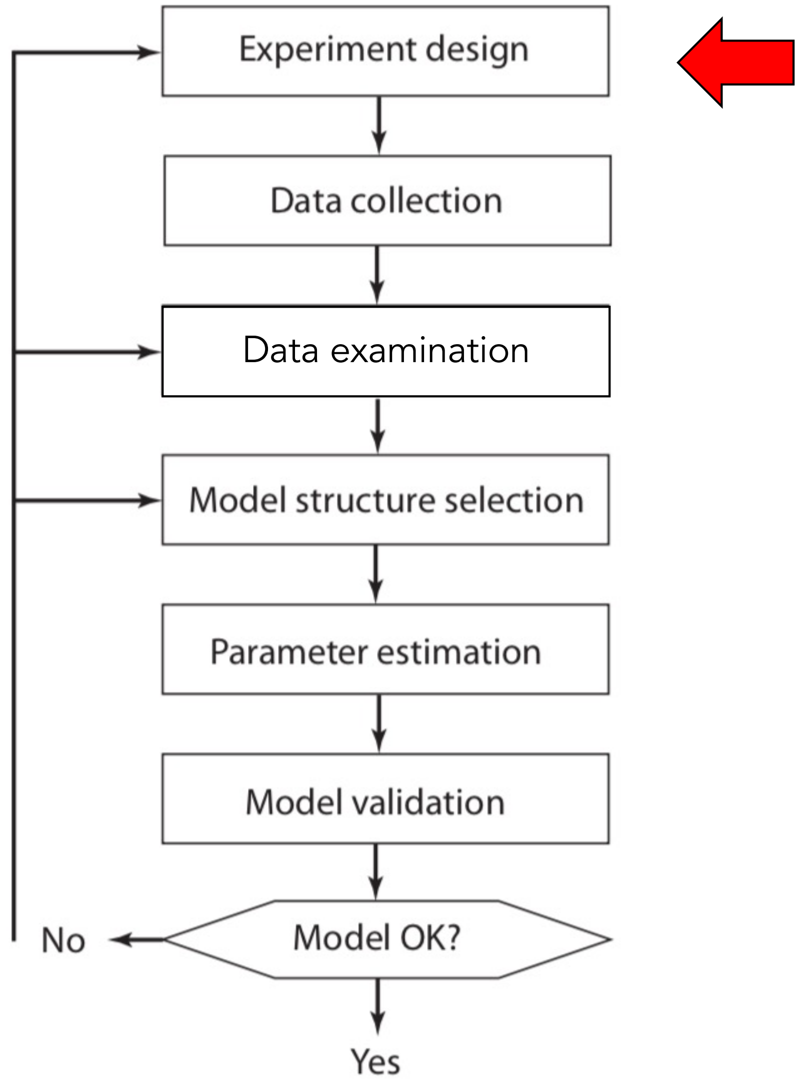
## Linear System Identification – A introduction from Brian



**Linear System Identification | System Identification, Part 2** 16 mn

*https://www.youtube.com/watch?v=qC_C04SEV1E*

# Data-driven system identification
## An iterative procedure



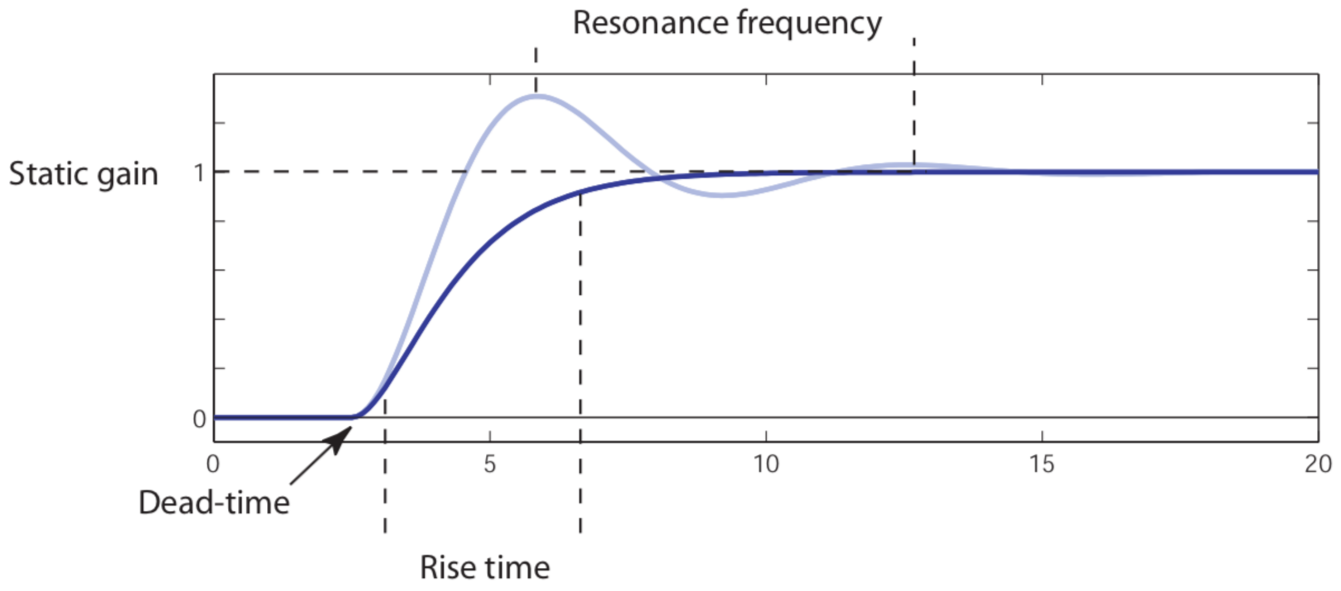
## Experiment design for data collection

- 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

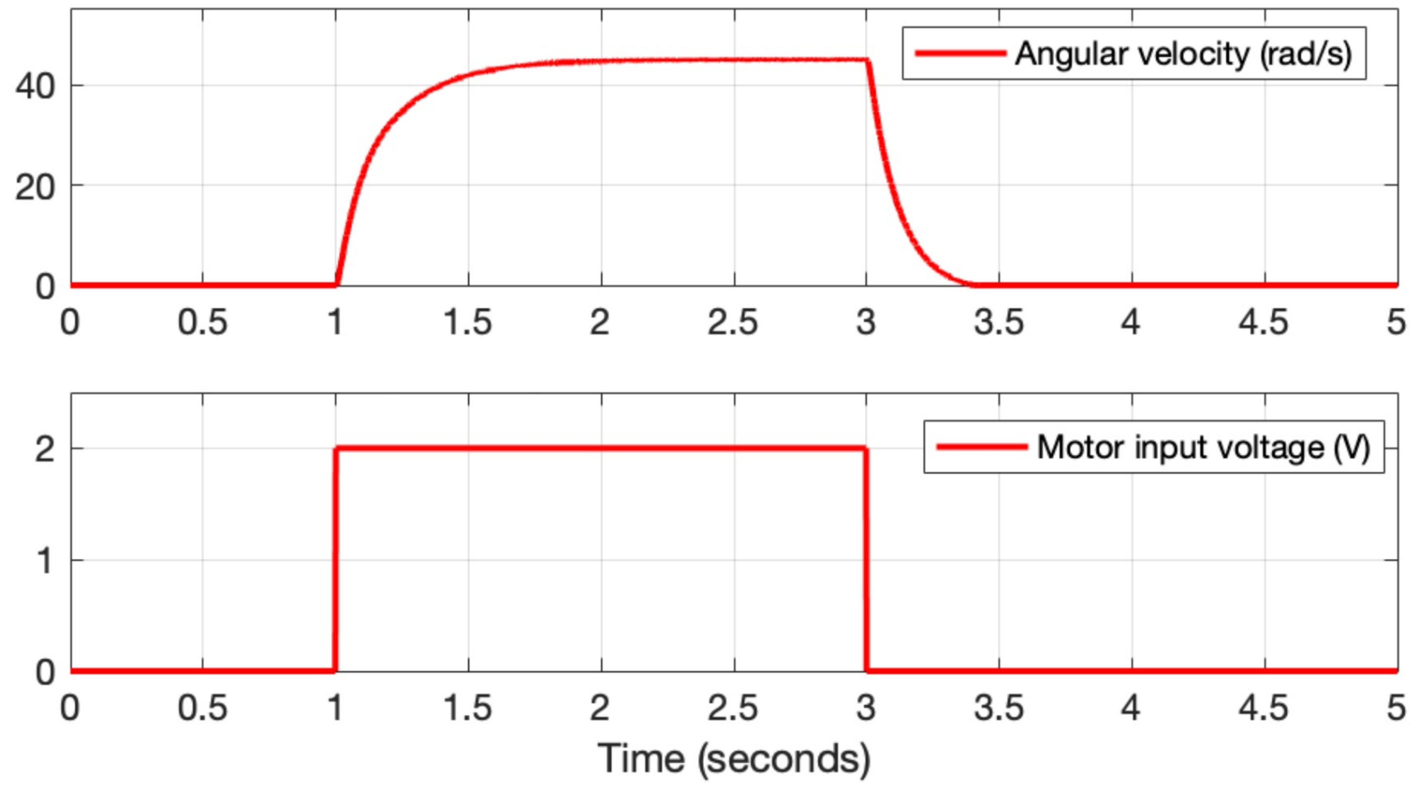


Useful for obtaining qualitative information about system

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

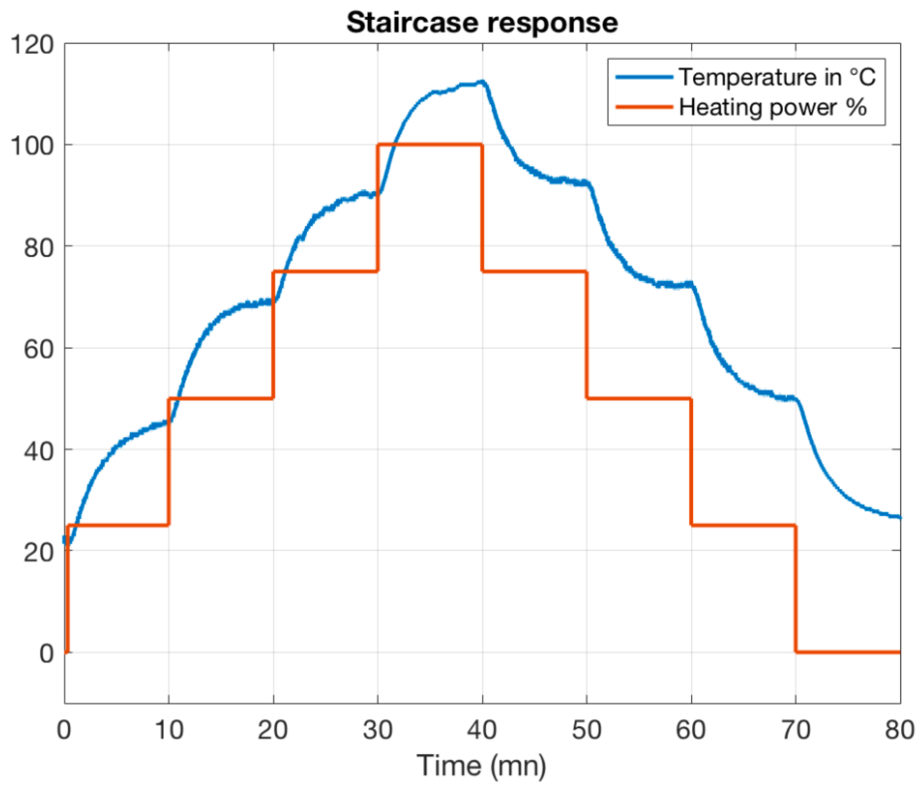
## Tests for verifying linearity

- A linear system has the same response to periodic square wave input
- Test square wave response around a given operating point



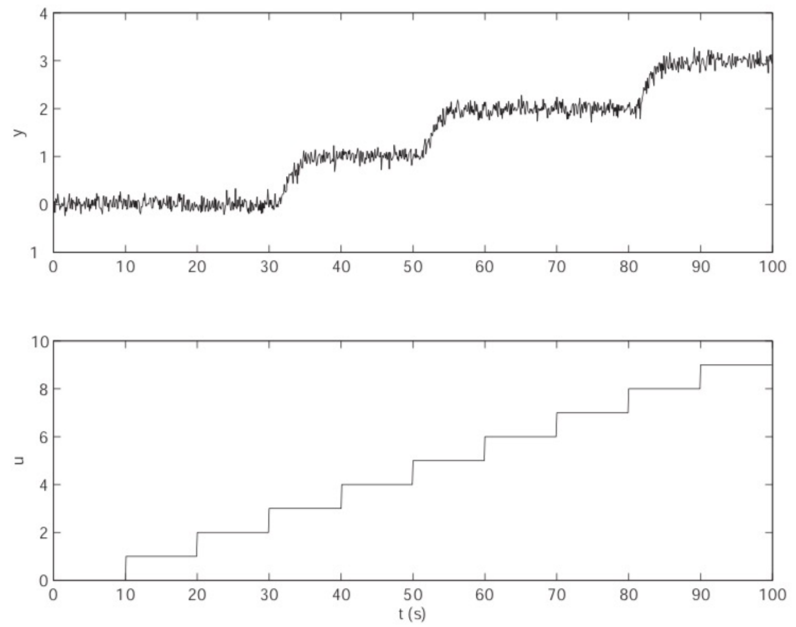
## Tests for verifying linearity

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



## Tests for detecting dry friction

Friction can be detected by using small step increases in input



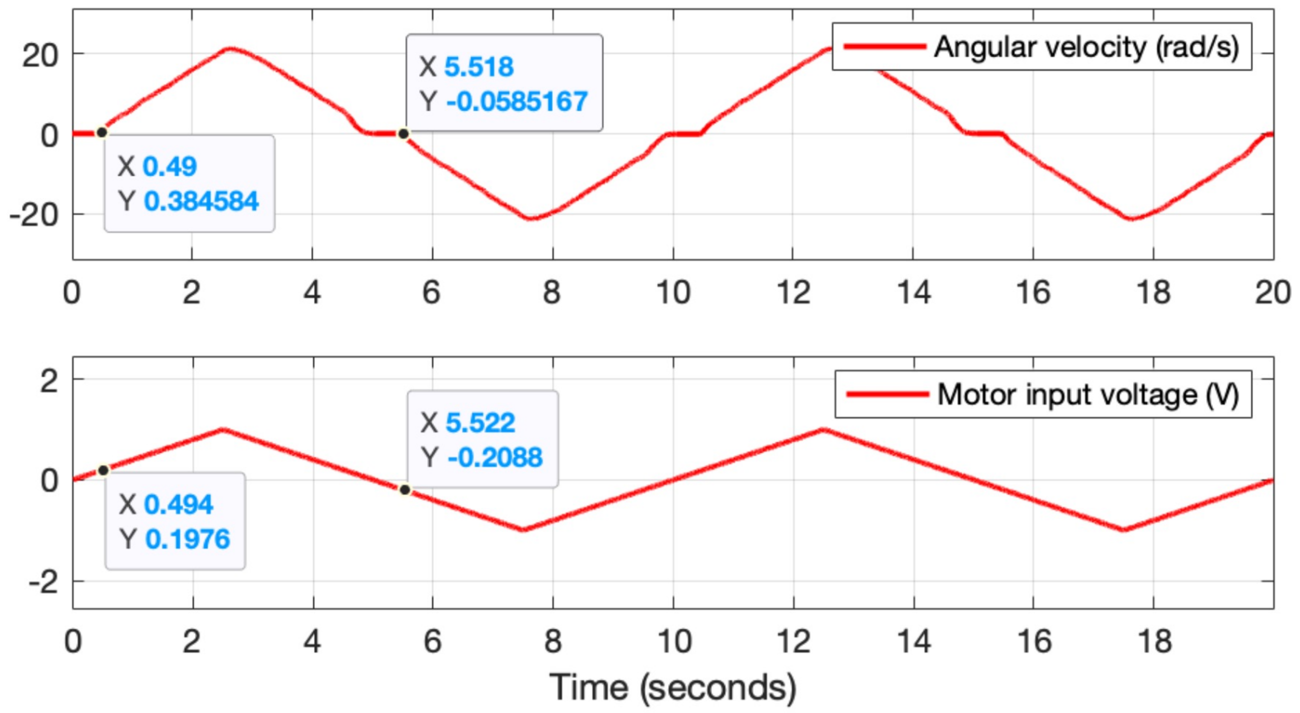
Input moves every two or three steps.

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
- Increase the amplitude of the steps to cancel out the friction effects

## Tests for detecting dry friction

- A linear system has the same response to periodic triangular wave input
- Test triangular wave response around a given operating point



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

- Use a nonlinear dead zone model to better capture the friction effects

## Choice of inputs for informative data

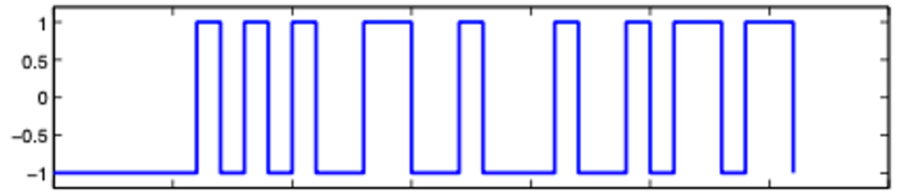
- Needs to be sufficiently "rich."
  - Input signal is needed to excite the system
  - The experiment should be carried out under conditions that are similar to those under which the model is going to be used
- Amplitude
  - Trade-off is needed
    - Large amplitude gives good signal-to-noise ratio, 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)

UNIVERSITÉ DE LORRAINE

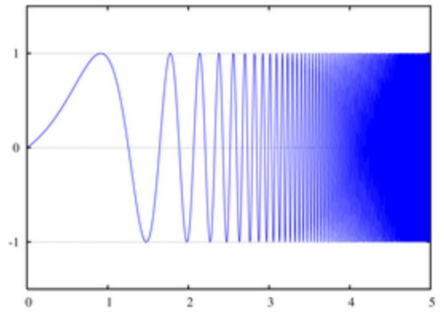
POLYTECH NANCY

## Common choice of inputs for system identification

- Square wave *(or basic step)* input *(not ideal)*
  - ➢ is often OK but can be insufficient to get accurate results for complex systems
- Pseudo-random Binary Sequence (PRBS)



- Chirp signal



H. Garnier

## Input design in Matlab

- In the Matlab SYSID toolbox, `idinput` function can generate various types of input
  - ➢ `u = idinput(N, type, band, levels)`

    where

    - N: number of data points

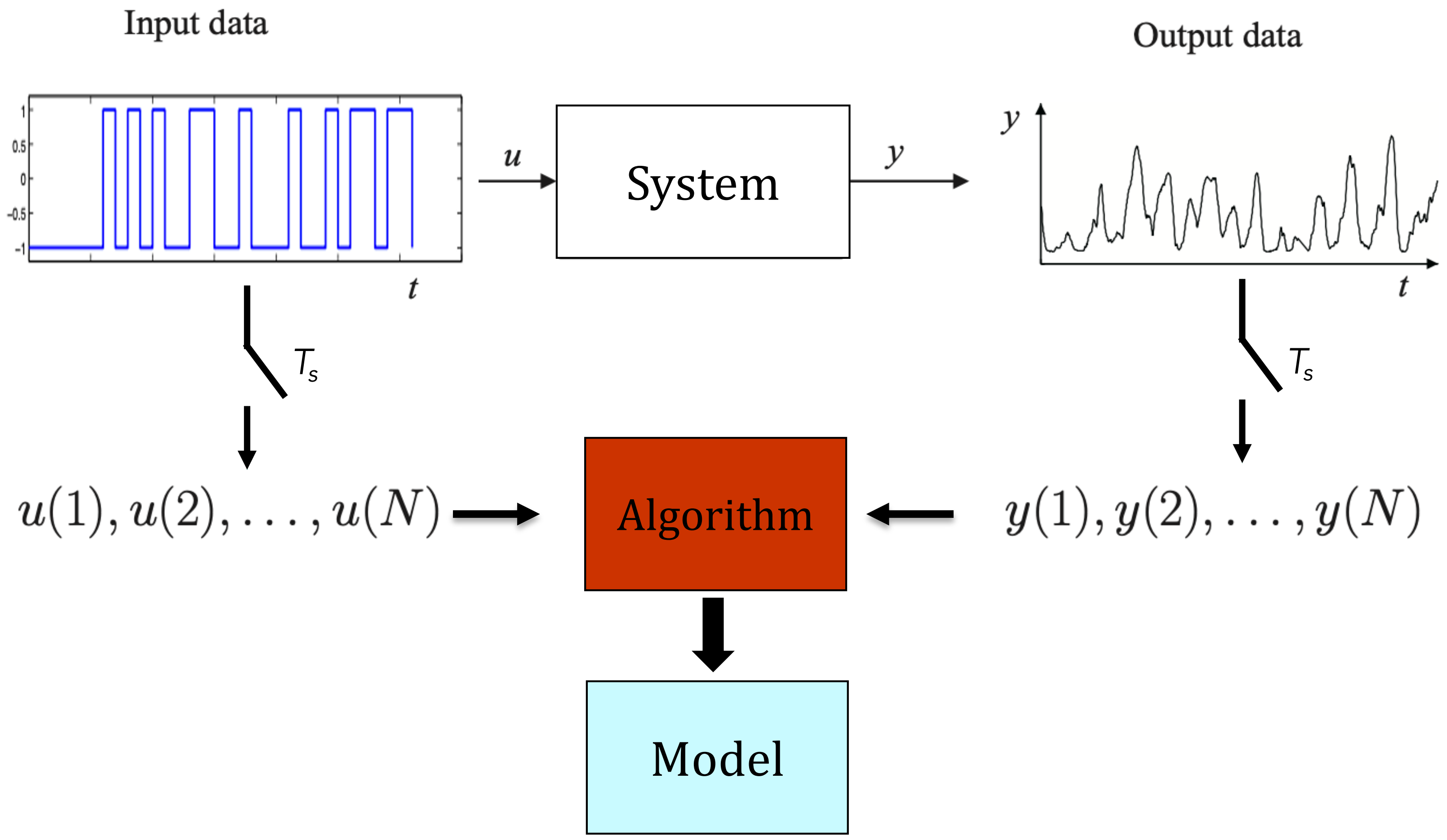
    - type:

      - RGS: generates a Random, Gaussian Signal
      - RBS: generates a Random, Binary Signal
      - PRBS: generates a Pseudo-random Binary Signal (PRBS)
      - SINE: generates a sum-of-sinusoid signal

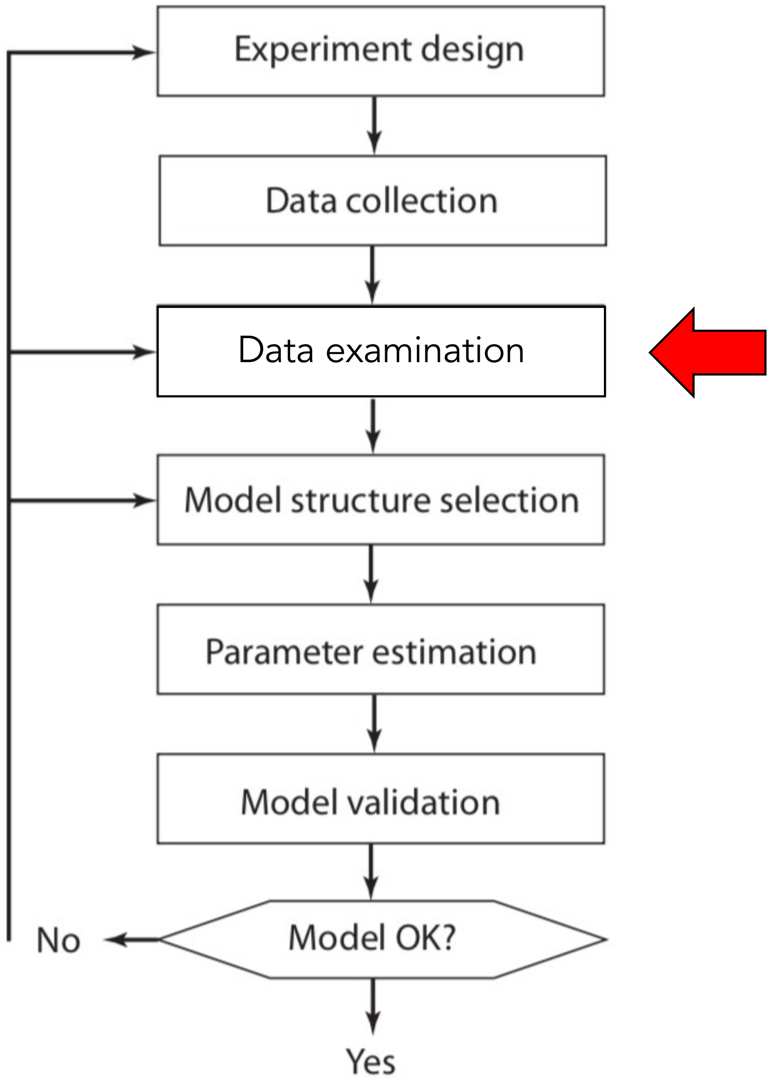
- In CONTSID toolbox, `prbs` function can generate a maximum-length PRBS input
  - ➢ `u = prbs(n,p,levels)`
    - n: order (number of stages) of the shift register. n must be between 6 and 18
    - p: coefficient such that the prbs signal remains constant over intervals of length p

$$n\, pT_s > t_{settling\_time}$$

# Data collection



# Data-driven system identification
An iterative procedure



## Examination of the collected data

ALWAYS plot FIRST the input/output data !

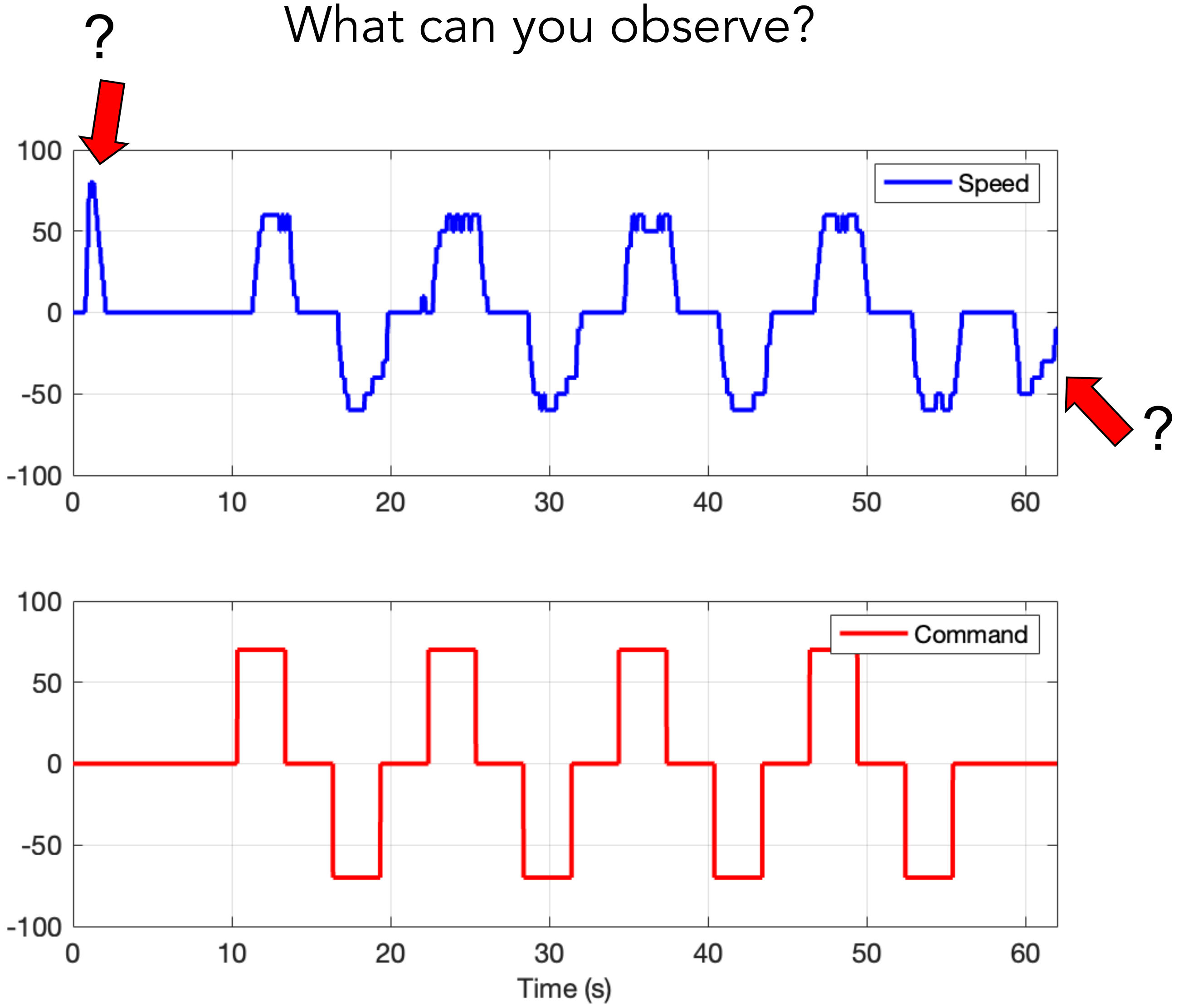
Examine carefully the measured data and identify possible problems

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

Select meaningful data segments for model estimation and validation ONLY

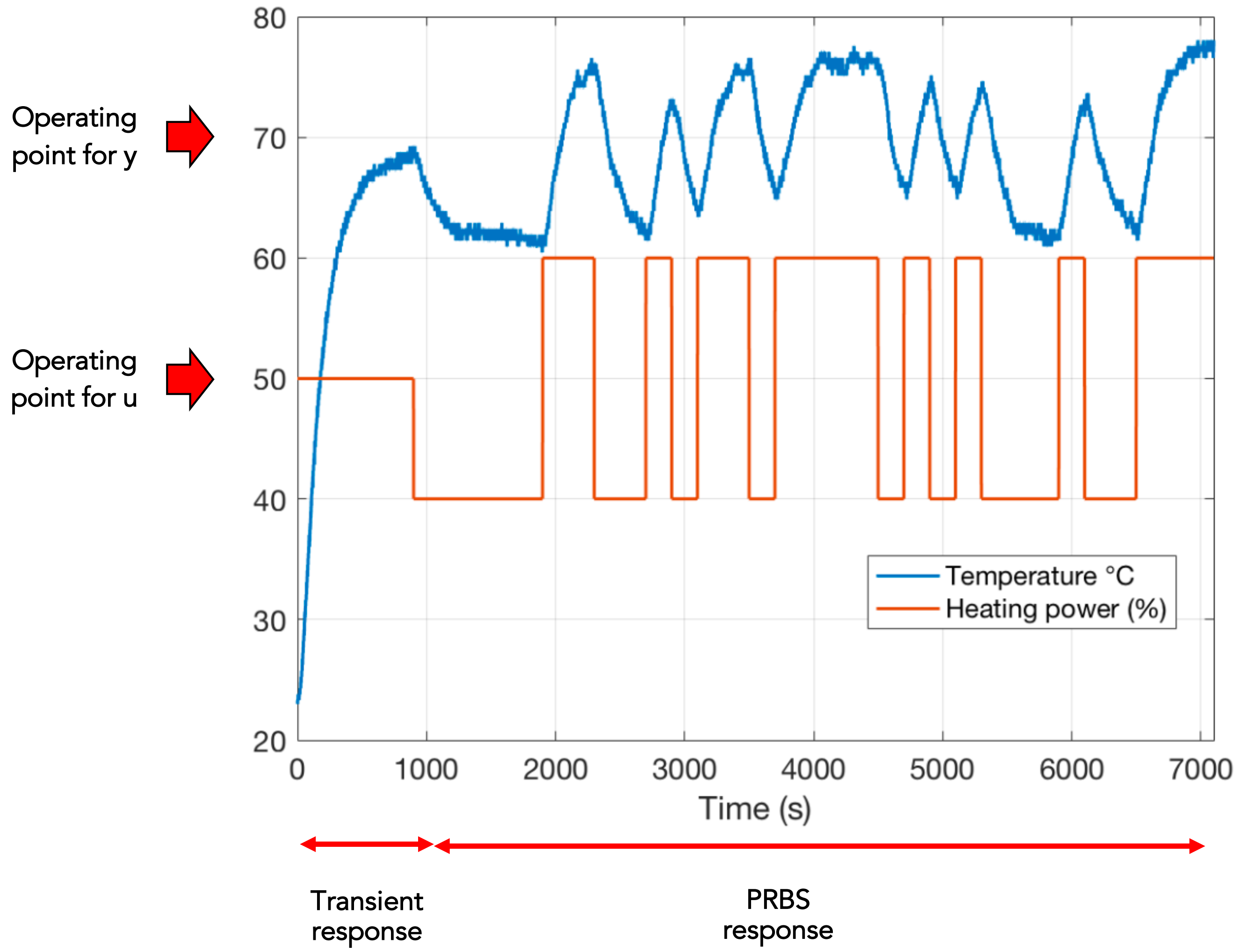
*If, from visual inspection, you cannot explain the output response you observe from the input, do not expect the identification algorithm will do !*

## Examination of collected data
## What can you observe?



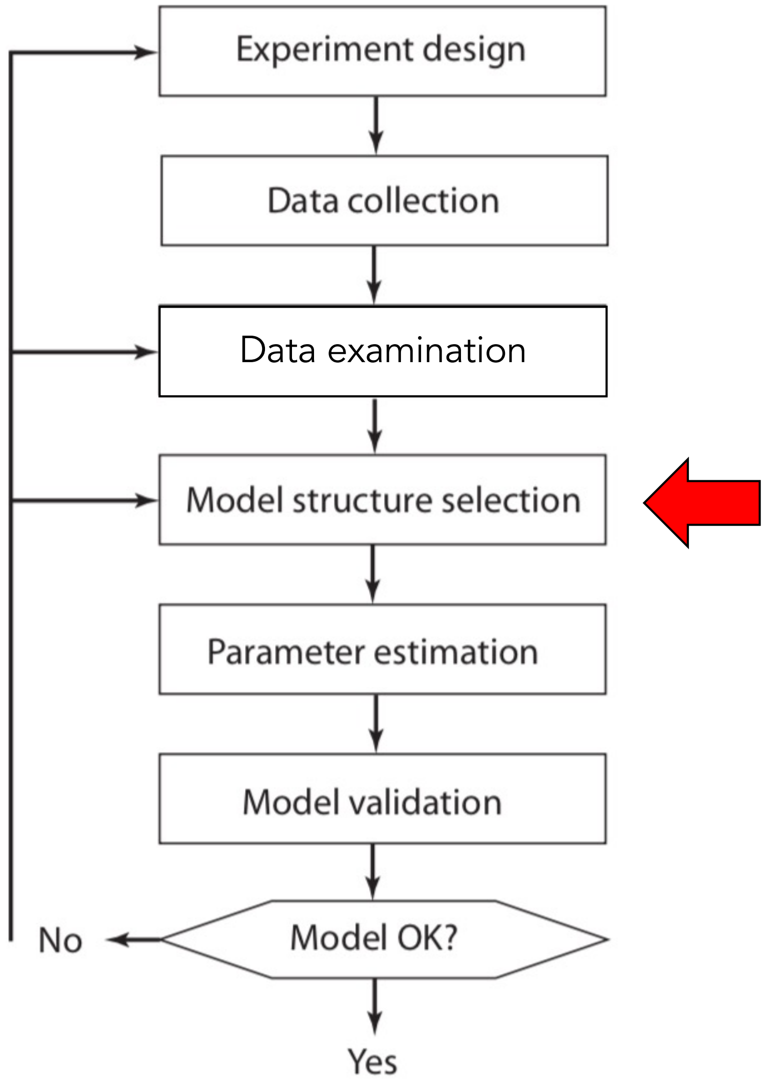
Discard the first and last 5s from your data

## Examination of the collected data
## What do you observe?



Discard the transient response and then mean values from your data

## 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

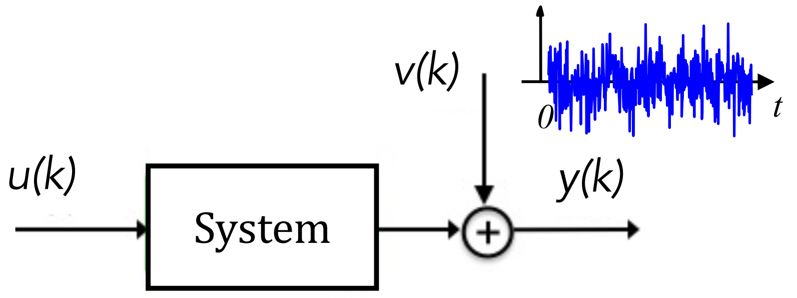
- Common examples of linear model structures are:

| | | |
|---|---|---|
| ➢ 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 | $G(s) = \frac{\left(b_0 + b_1 s + b_2 s^2 + \ldots\right)}{\left(1 + f_1 s + f_2 s^2 + \ldots\right)} e^{-T_d s}$ | 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)

## Models for measurement noise

So far: only deterministic models



| | | |
|---|---|---|
| $u(k)$ | $=$ | input |
| $v(k)$ | $=$ | measurement noise |
| $y(k)$ | $=$ | output |

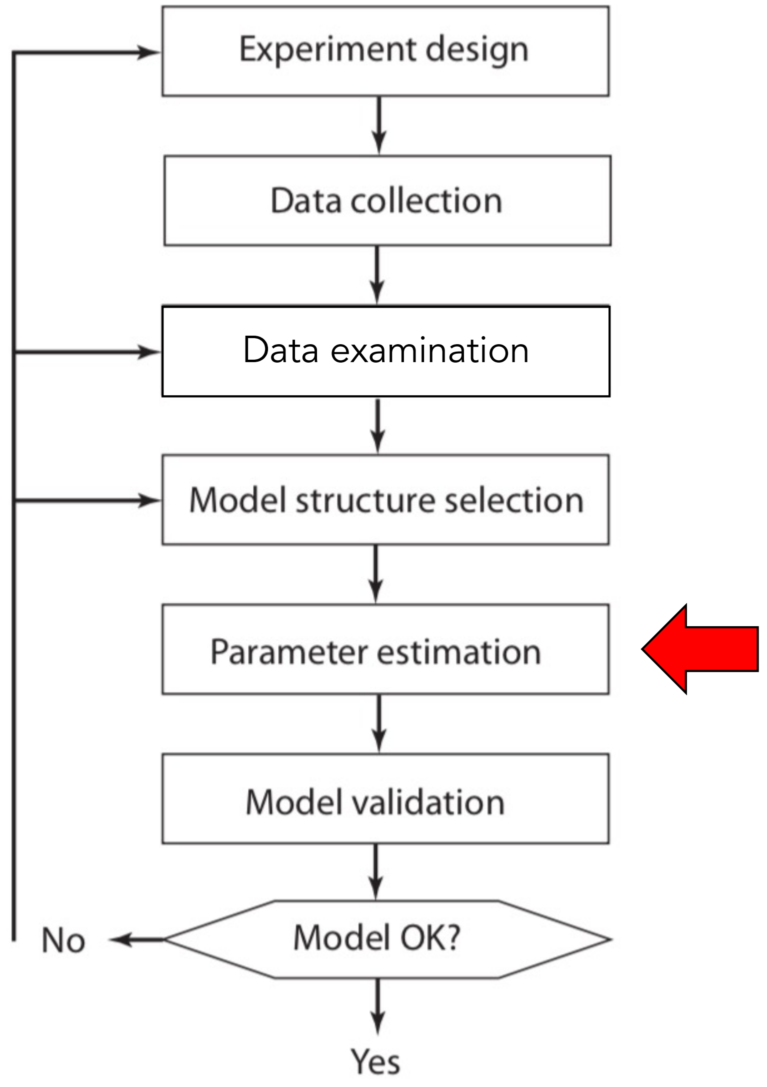
- Measurement noise modelled as a stochastic discrete-time signal
- Stochastic models:
  - means, covariances
  - spectra (energy or power)

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

$e(k)$ 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:

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

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

**Estimate:**

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

## Instrumental variable (IV) methods

System:

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

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

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

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

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

✓ Amongst the different IV versions, one is particularly recommended:

- SRIVC: **S**imple **R**efined **I**nstrumental **V**ariable algorithm for **COE** 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:**

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

## Subspace methods

- Use linear algebra to estimate state-space models
  - ➢ Well adapted to multivariable-input multivariable-output (MIMO) systems
  - ➢ More to come next year

UNIVERSITÉ DE LORRAINE

POLYTECH NANCY

## Main estimation routines in the SID and CONTSID toolboxes

**Matlab SID toolbox** **CONTSID toolbox**

**Commands for Offline Estimation**

| Model Type | Estimation Commands | |
|---|---|---|
| Transfer function models | `tfest` | **`tfsrivc/tfrivc`** |
| Process models (low-order transfer functions expressed in time-constant form) | `procest` | **`procsrivc`** |
| 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>`polyest` (for all models) | **`lssvf (CARX)`**<br>**`rivc (CBJ)`**<br>**`srivc (COE)`**<br>**`coe (COE)`** |
| State-space models | `n4sid`<br>`ssest`<br>`ssregest` | **`sidgpmf`** |

- The majority of these estimation algorithms are iterative
- As these routines can estimate different models quickly, you should try as many different structures as possible to see which one produces the best results
- Let us investigate the case of a simulated system known as the Rao-Garnier benchmark

## The simulated *Rao-Garnier* benchmark

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

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

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

✓ *1 "unstable" zero*

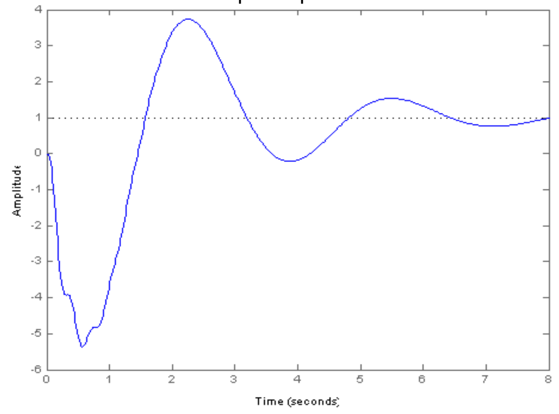
✓ *2 pseudo-oscillatory modes*

$K = 1$

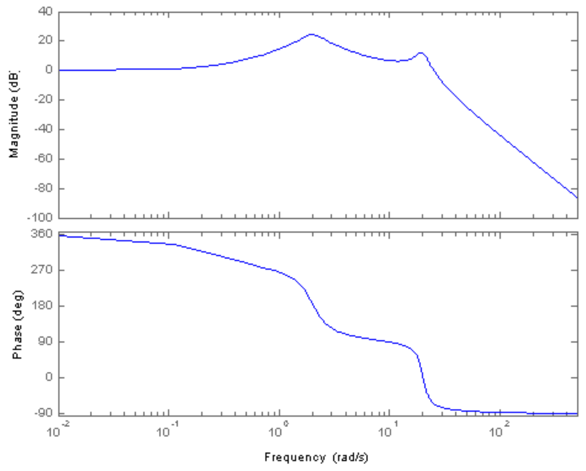
$\omega_{n,1} = 2\ rad/s;\quad \omega_{n,2} = 20\ rad/s$

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

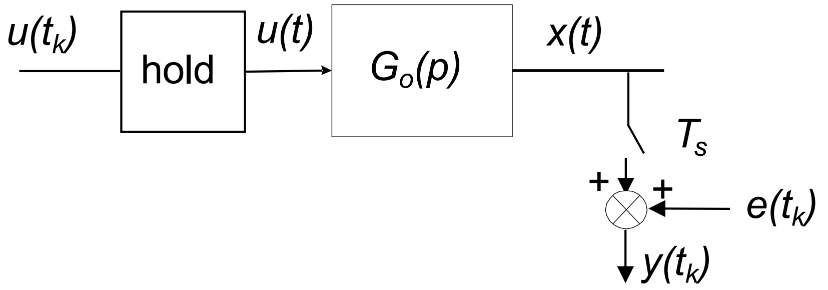
Step response



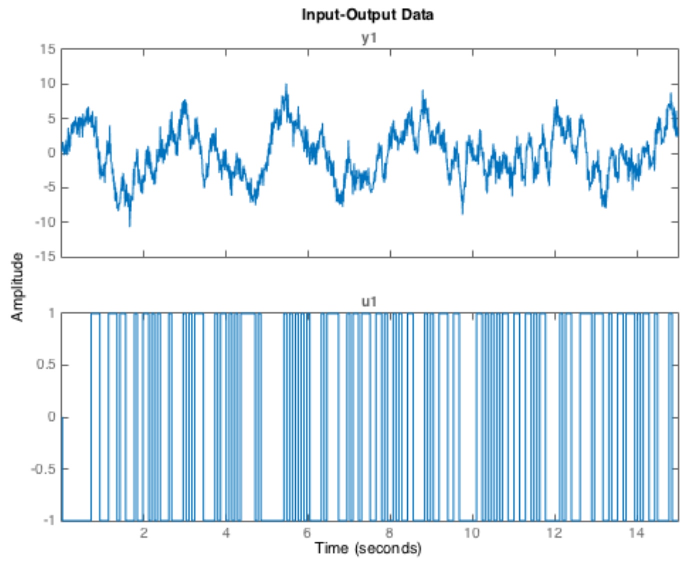
Bode diagram



## Simulation setup



- $u(t_k)$: PRBS *(respects the ZOH assumption)*
- *$T_s$=10 ms (quite fast sampling scenario)*
  - *$f_s \approx$ 10 times the system bandwidth*
    *an often given rule for DT identification*
- *$e(t_k)$* : DT white noise, SNR=10 dB
- Model order of 4 is assumed known



H. Garnier

## Estimation results - Rao-Garnier benchmark (Garnier EJC 2015)

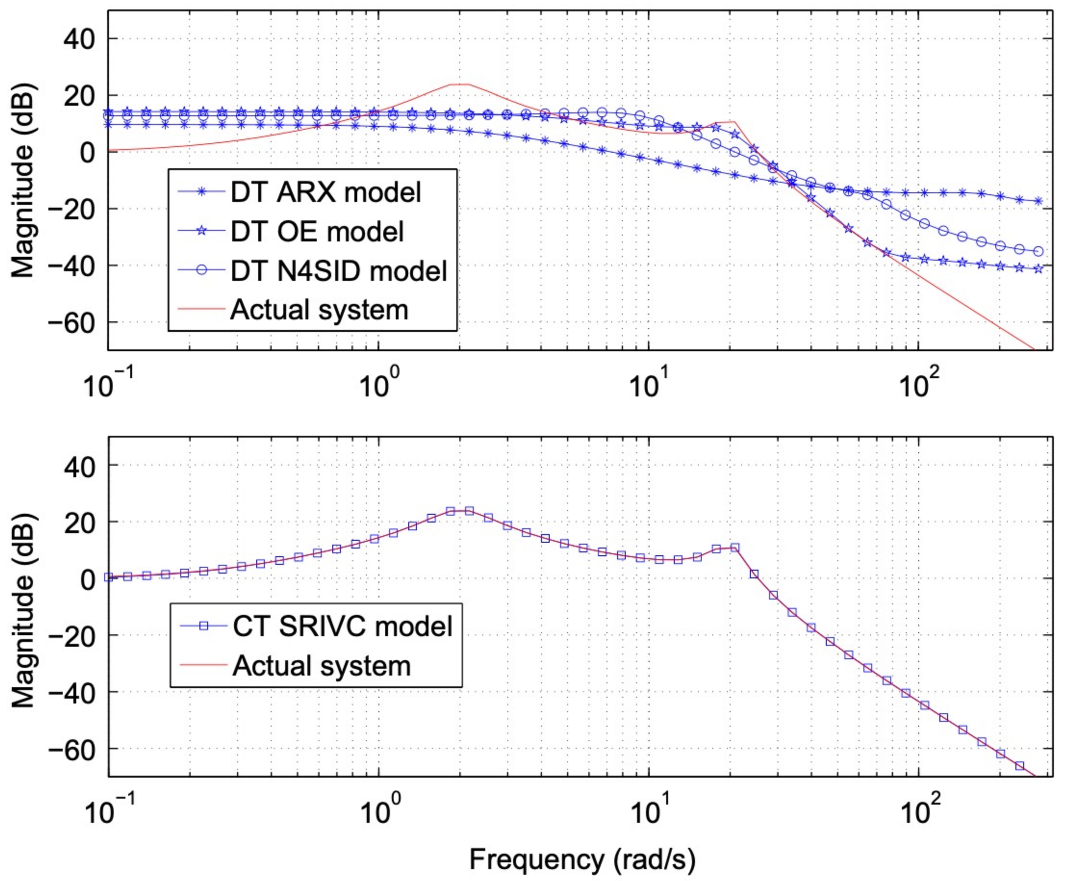


**Fig. 1.** Actual Rao–Garnier benchmark magnitude Bode plot compared with those of the CT SRIVC (bottom figure) and traditional DT ARX, OE and N4SID (top figure) routines (the IV4 routine was also tested but, for some unclear reasons, delivers systematically an error message with these data). The estimated CT SRIVC model can hardly be distinguished from the true system Bode plot (bottom figure) while the 3 DT methods cannot capture the dynamics of the benchmark test.

*DT methods suffer from numerical issues when the data are fast sampled and do not work well*

*CT methods from CONTSID are free of these difficulties*

## Discrete-time (DT) versus continuous-time (CT) model estimation methods

- This example illustrates some of the difficulties that may appear in DT model estimation methods
  - sensitivity to the initialization of the iterative search (via ARX/IV4 or N4SID)
  - numerical issues in the case of fast sampling
  - non inherent data prefiltering

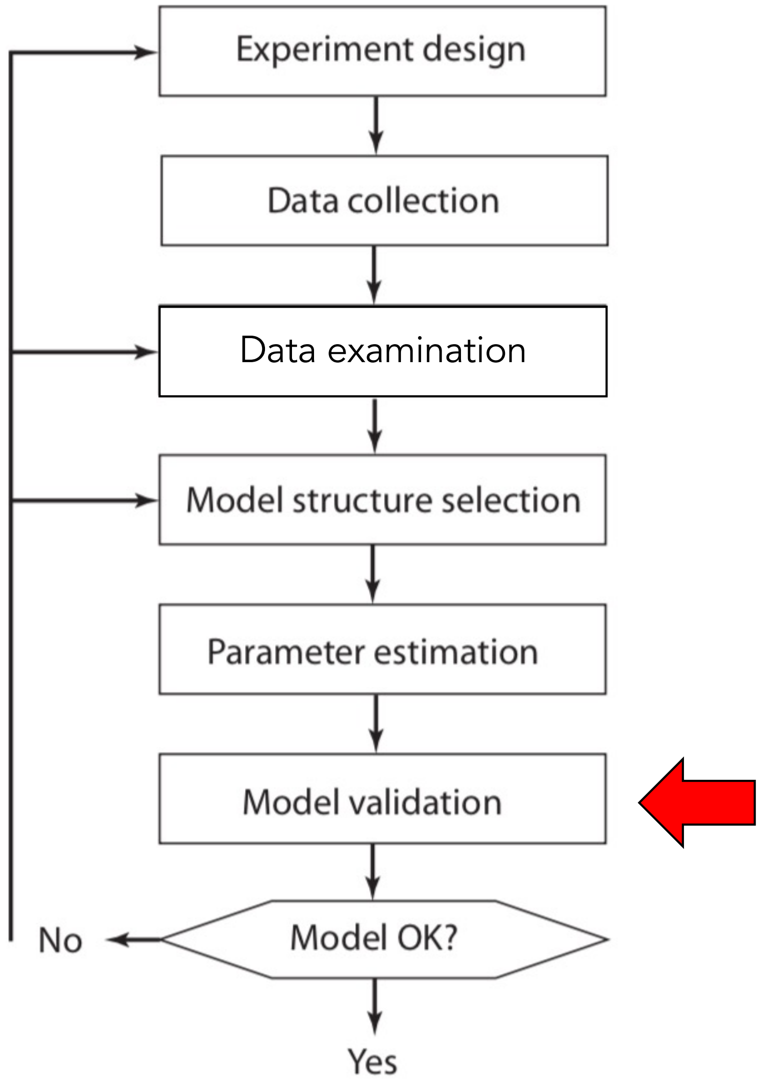
  These difficulties require extra care from the practitioners in DT modelling

- The CT model estimation methods are free from these difficulties
  - they make the application of the SYSID procedure much easier
  - they are recommended as a first choice

- Use preferably the following CT model estimation routines

| In the SID toolbox | In the CONTSID toolbox |
|---|---|
| **`procest`** | **`procsrivc`** |
| **`tfest`** | **`tfsrivc`** |
| **`ssest`** | **`srivc`** |

## Data-driven system identification
## An iterative procedure

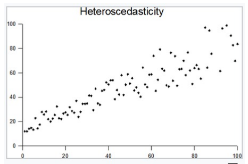


## Recommended workflow of model validation

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

1. 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, time-varying effects, noise heteroskedasticity, …



Plot with random data showing heteroscedasticity: The variance of the *y*-values of the dots increase with increasing values of *x*.

2. Interpret the main features of the identified model in a physical sense
   - Steady-state gain, time-constants, time-delay, damping coefficient, natural frequencies

## Estimation data versus validation data

Split your data:

Estimation data

data use for parameter estimation

$\Downarrow$

compute $\hat{\theta}_N$

Validation data

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

use them to evaluate the quality of the fit

$\Downarrow$

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

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

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

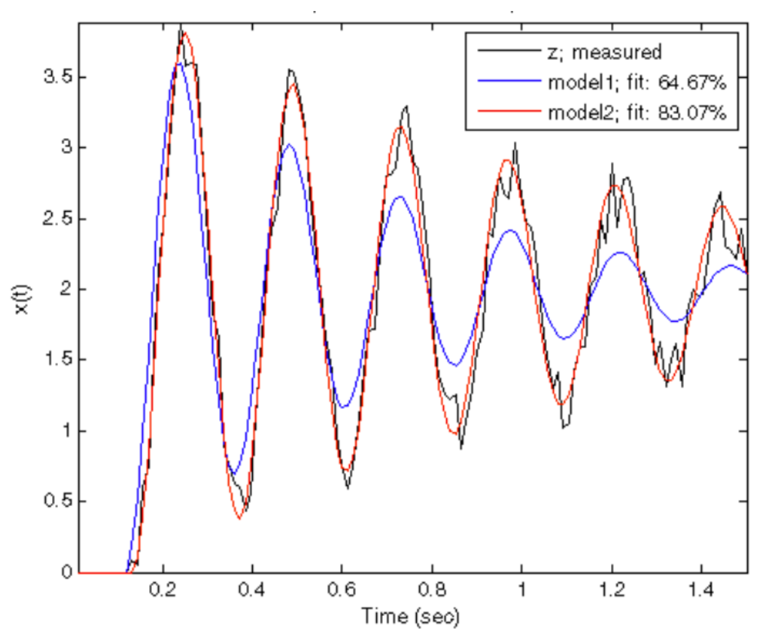
- the coefficient of determination

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

  - ➢ 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



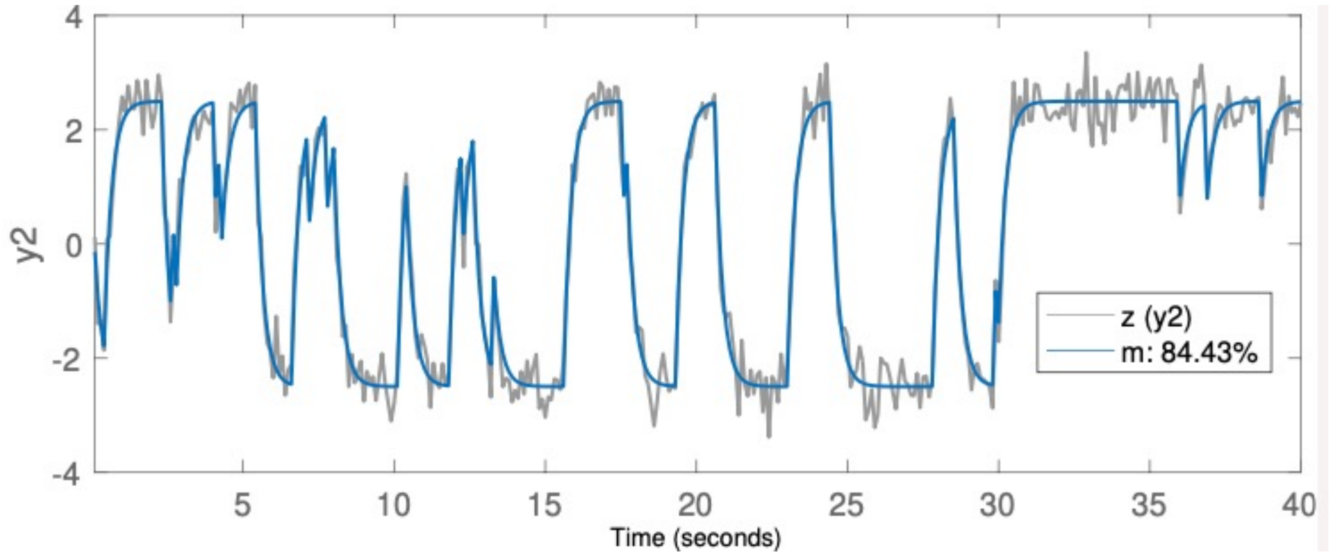
➢ model2 above is better than model1 because model2 better fits the data (83% vs. 65%)

UNIVERSITÉ DE LORRAINE

POLYTECH NANCY

## 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

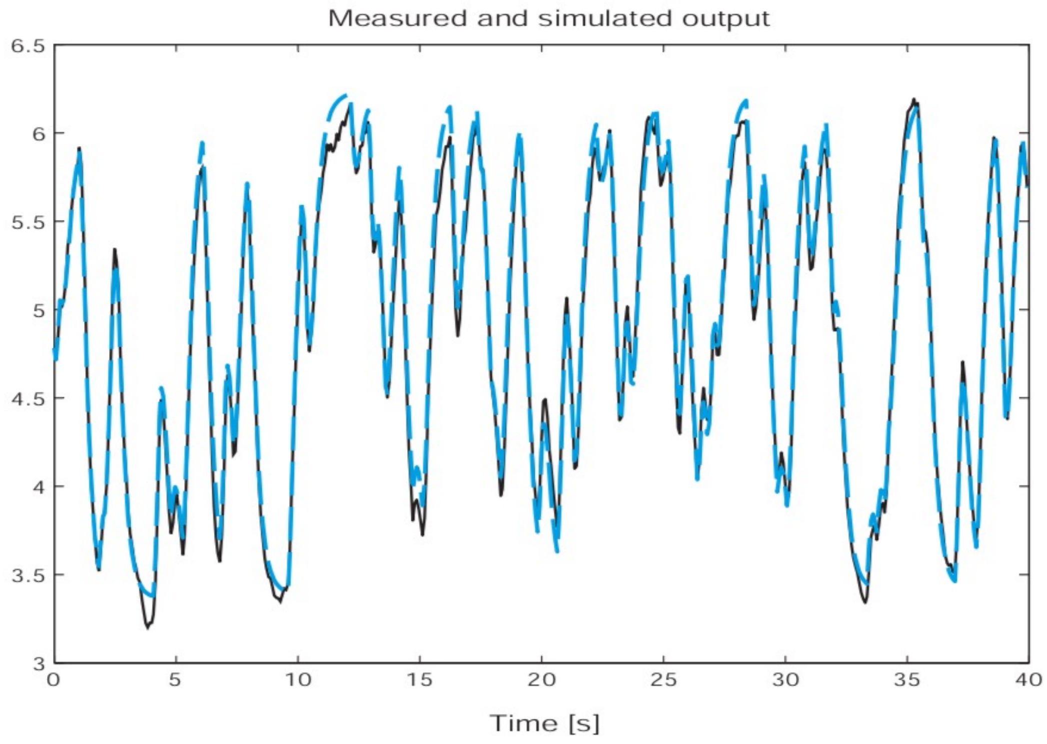


- The real test is to see how well the model can reproduce the validation data: cross-validation data test

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

Apply input signal in validation data set to estimated model

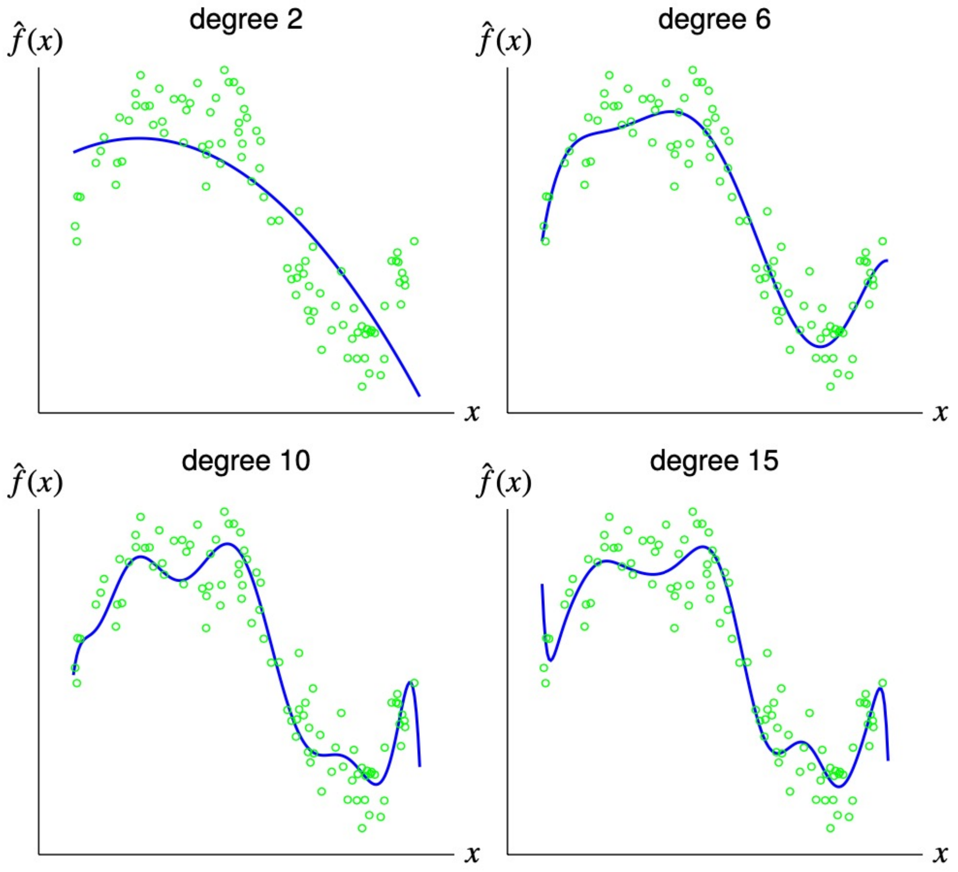
Compare simulated output with output stored in validation data set.



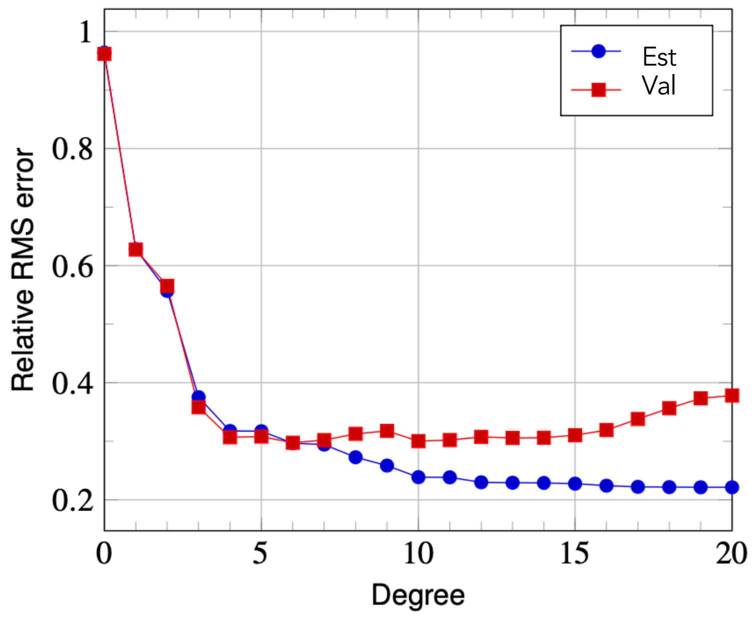
## Choice of the model order
## Example: LS polynomial model fit

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

- 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 $N \longrightarrow \infty$*

## 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 |

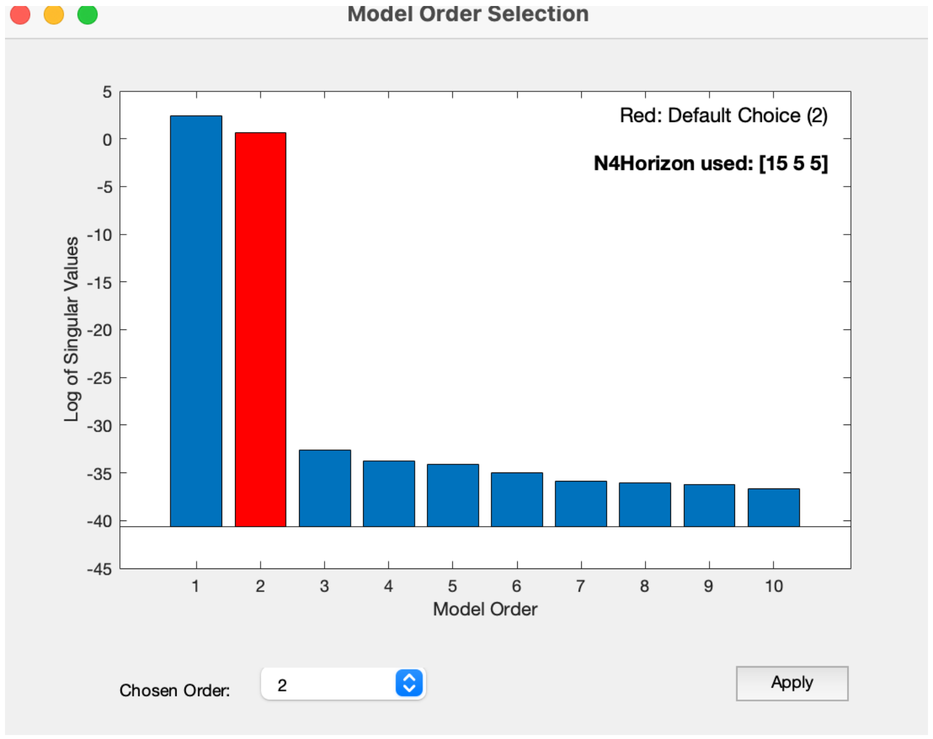
  - If several model candidates achieve similar performance, you should choose the simplest (lowest-order) one among these candidates

## Model order selection from subspace state-space model estimation

A pragmatic and interesting way to choose the model order is to use the susbpace-based estimation method n4sid directly, as an alternative to ssest

The algorithm automatically estimates a discrete-time state-space model of the best order in the 1:10 range with the estimation data

```
>>M = n4sid(data)
```



## Choice of the model order
## Take-home message

- Choosing the model order is most of the time 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

## Model assumption verification via residual statistical tests

- When you choose a model structure (ARX, ARMAX, OE, BJ, …), you make the implicit assumption that the input/output has been generated by the chosen model (assumption about the noise model in particular)
- With this assumption, residual tests coming from probability/statistics can be used

If we fit the parameters of the model

$$y[t] = G(q;\theta)u[t] + H(q;\theta)e[t]$$

to data, the *residuals*

$$\varepsilon[t] = H(q;\theta)^{-1}\left\{y[t] - G(q;\theta)u[t]\right\}$$

: explains mismatch between model and observed data.

If the model is correct, the residuals should be

- white, and
- uncorrelated with $u$

## Autocovariance of the residuals

Residuals

$$\epsilon(t, \hat{\theta}_N) = y(t) - \hat{y}(t|\hat{\theta}_N)$$

- Ideally these should be independent (if noise model is well-estimated)
- Estimating Autocovariance of the residuals

$$\hat{R}_\epsilon(\tau) = \frac{1}{N}\sum_{t=1}^{N} \epsilon(t+\tau)\epsilon(t)$$

- Plot $\hat{R}_\epsilon$
- independent residuals: $\hat{R}_\epsilon(\tau)$ is a $\delta$ function

- Large components indicate unmodelled dynamics for the noise model

## Cross-covariance between the residuals and the input

Residuals:

$$\epsilon(t, \hat{\theta}_N) = y(t) - \hat{y}(t|\hat{\theta}_N)$$

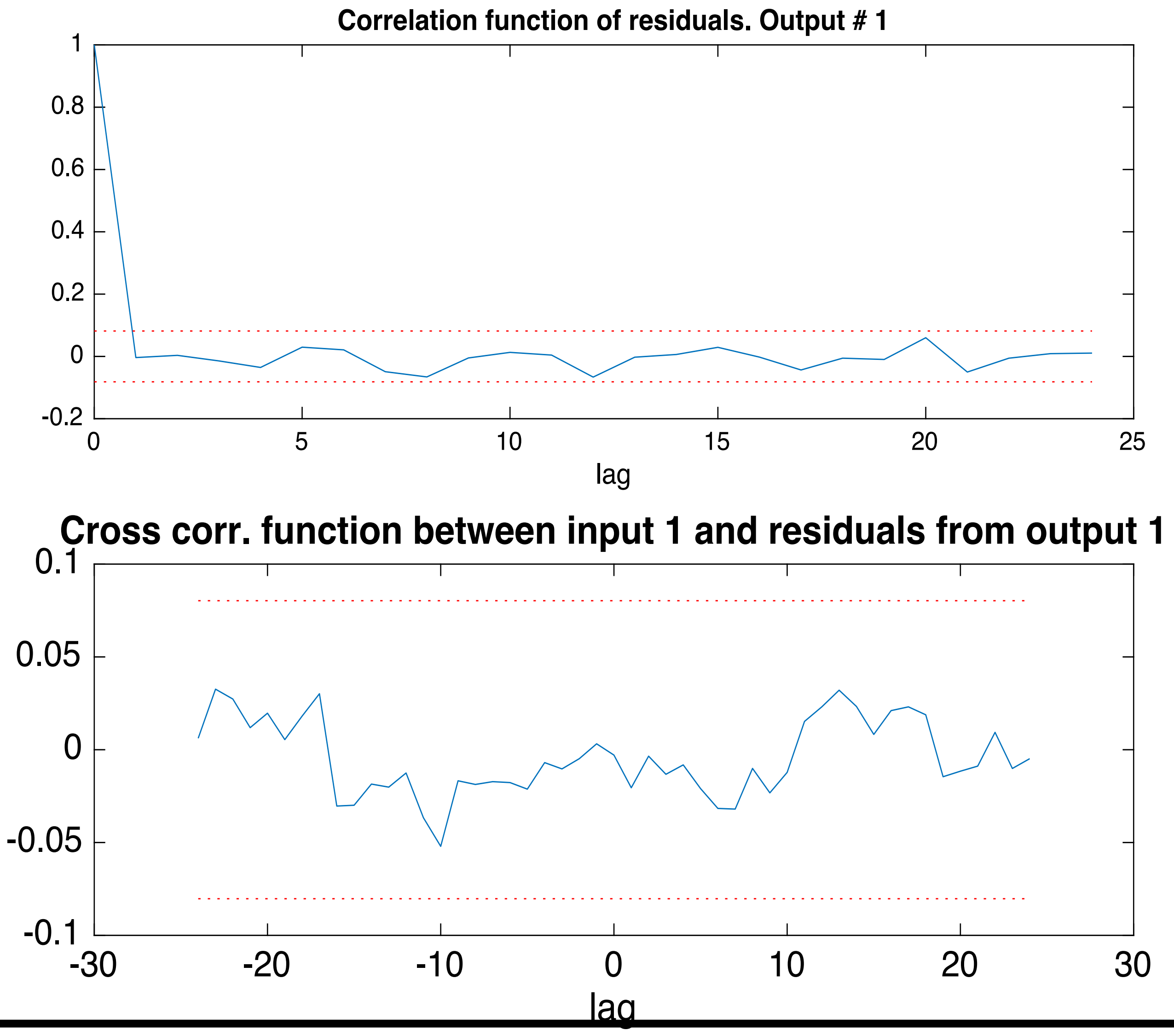
- Ideally $\epsilon$ should be independent of $u$
- Estimating Cross-covariance between $\epsilon$ and $u$

$$\hat{R}_{\epsilon u}(\tau) = \frac{1}{N} \sum_{t=1}^{N} \epsilon(t+\tau)u(t)$$

- Plot $\hat{R}_{\epsilon u}$
- Correlation for negative $\tau$: perhaps there is a feedback ($u(t)$ "depends" on $\epsilon(t-\tau)$)

- Large components indicate unmodelled dynamics for the plant model

## Statistical tests on the residuals



## Data-driven linear System identification
## 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 !