Matlab software tools for model identification and data analysis

Mara Tanelli

Email: mara.tanelli@polimi.it

Thanks to *Prof. Marcello Farina* for sharing his slides with me!

Outline

- **Data generation and analysis**
  - Definition and generation of time series and dynamic systems
  - Analysis of systems
  - Analysis of time series (sampled covariance and spectral analysis)

- **Optimal predictors**

- **Identification**
  - Parameter identification
  - Structure selection
  - Model validation

- **Exercises**

– **White noise**

```
>> N=100;  % number of samples
>> lambda=1; % standard deviation
>> mu=1;   % expected value
```

- **I method**

```
>> u=mu+lambda*randn(N,1);
```

- **II method**

```
>> u=wgn(M,N,P); % zero-mean signal
M: number of signals
P: signal power (dBW) (10 log10(lambda^2))
```

Definition of dynamic linear discrete-time systems: external representation (transfer function – *LTI format*)

```
>> system = tf(num,den,Ts)
```

- num: numerator
- den: denominator
- Ts: sampling time (1 default for discrete-time systems, if Ts omitted, the system is continuous-time)

```
>> system=tf([1 1],[1 -0.8],1)
Transfer function:
            z + 1
           -------
           z - 0.8
```

- Step response of a LTI system

```
>> T=[0:Ts:Tfinal];
>> [y,T]=step(system,Tfinal); % step response
>> step(system,Tfinal)        % plot
```

- Response of a LTI system to a generic input signal

```
>> T=[0:Ts:Tfinal];
>> u1=sin(0.1*T);              % e.g., sinusoidal input
>> u2=1+randn(T,1);            % e.g., white gaussian noise
>> [y,T]=lsim(system,u1+u2,T);
```

The SYSTEM IDENTIFICATION TOOLBOX uses different formats for models (IDPOLY *format*) and signals (IDDATA *format*)

Definition of IDDATA *objects (i.e., inputs&outputs* of a system in a single object*)*:

```
>> data=iddata(y,u,Ts);
u: input data
y: output data
Ts: sampling time
```

- Definition of *IDPOLY objects*:

```
>> model_idpoly = idpoly(A,B,C,D,F,NoiseVariance,Ts)
A,B,C,D,F are vectors (i.e., containing coefficients of the polynomials in
  increasing order of the exponent in the form z^-1)
Model_idpoly:
A(q) y(t) = [B(q)/F(q)] u(t-nk) + [C(q)/D(q)] e(t)
(If ARMAX: F=[], D=[])
The delay nk is defined in polynomial B(q).

>> conversion from LTI format:
Model_idpoly = idpoly(model_LTI)
model_LTI is a LTI model (defined, e.g., using tf, zpk, ss)
```

- Response of an IDPOLY *system* to a generic input signal (vector u)

```
>> y=sim(Model_idpoly,u); -> It does not simulate the noise
                            component, but just the deterministic one
```

Analysis of systems (both for LTI and IDPOLY *formats*):

- Zeros and poles:

```
>> [Z,P,K]=zpkdata(model)
Z =   [-1] % positions of the zeros
P = [0.8000] % positions of the poles
K = 1 transfer constant (different from the gain!)
```

- Bode diagrams:

```
>> [A,phi] = bode(model,W)
W: values of the angular frequency ‘w’ where to compute ‘A(w)’ and ‘phi(w)’
A:   amplitude (A(w))
Phi: phase (phi(w))
```

Analysis of time series

- `m=mean(X)` % Average or mean value.
- `Y=detrend(X,'constant')` % Remove constant trend from data sets.
- `Y=detrend(X,'linear')` % Remove linear trend from data sets.

– Covariance function

```
>> gamma=covf(y,10) % tau=0,1,…,10-1
>> plot([0:9],gamma)
```

Warning: `covf` computes the correlation function!

TO OBTAIN THE COVARIANCE FUNCTION THE SIGNAL MUST HAVE ZERO MEAN

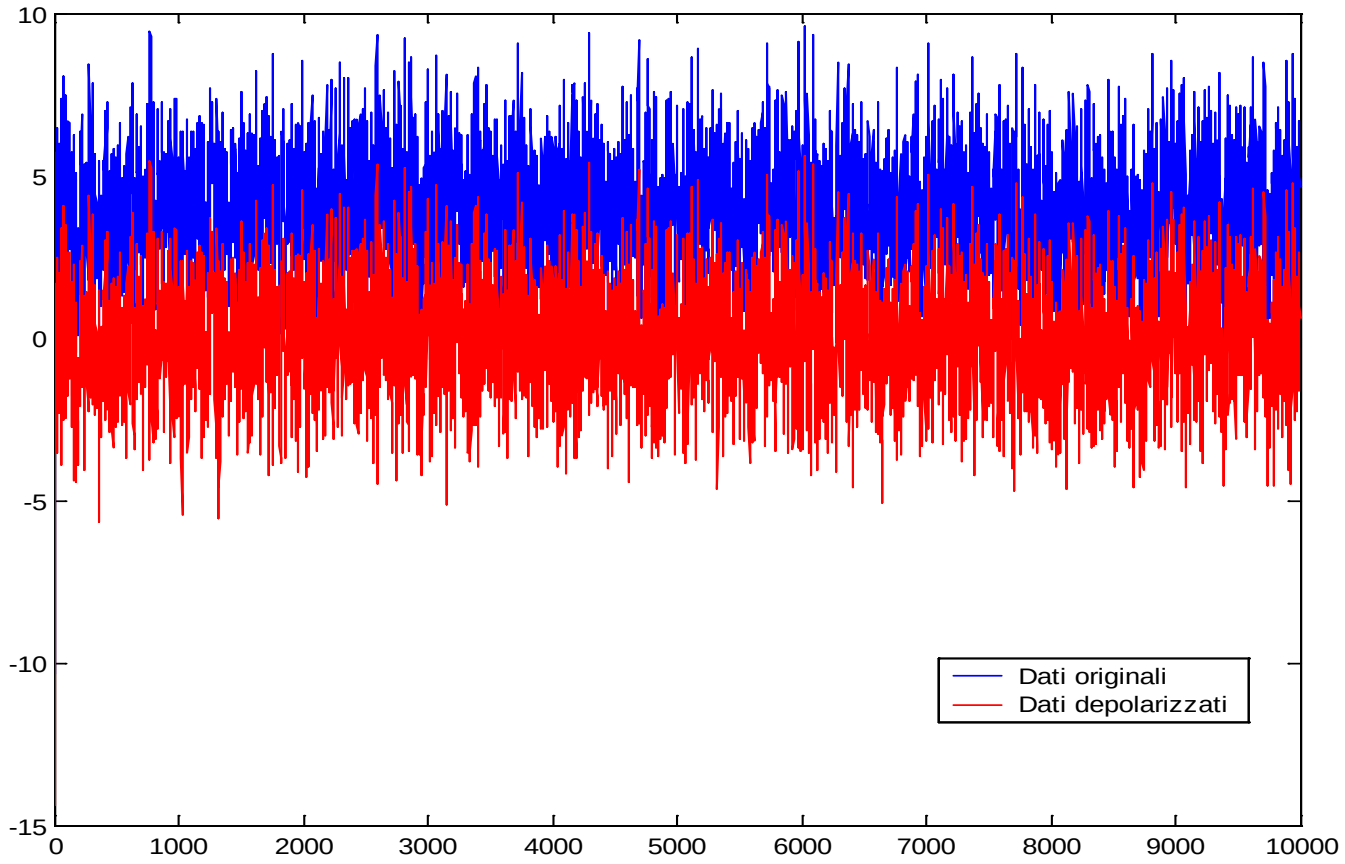
– Spectrum –periodogram method

```
>> periodogram(y);     % plots the periodogram
>> [Pxx,w]=periodogram(y);
>> plot(w/pi,10*log10(Pxx))
```

Warning: no regularization technique is applied: the estimator is not consistent!

Other methods are also available (e.g., fast fourier transform, see «help spectrum»)

Constant Trend removal

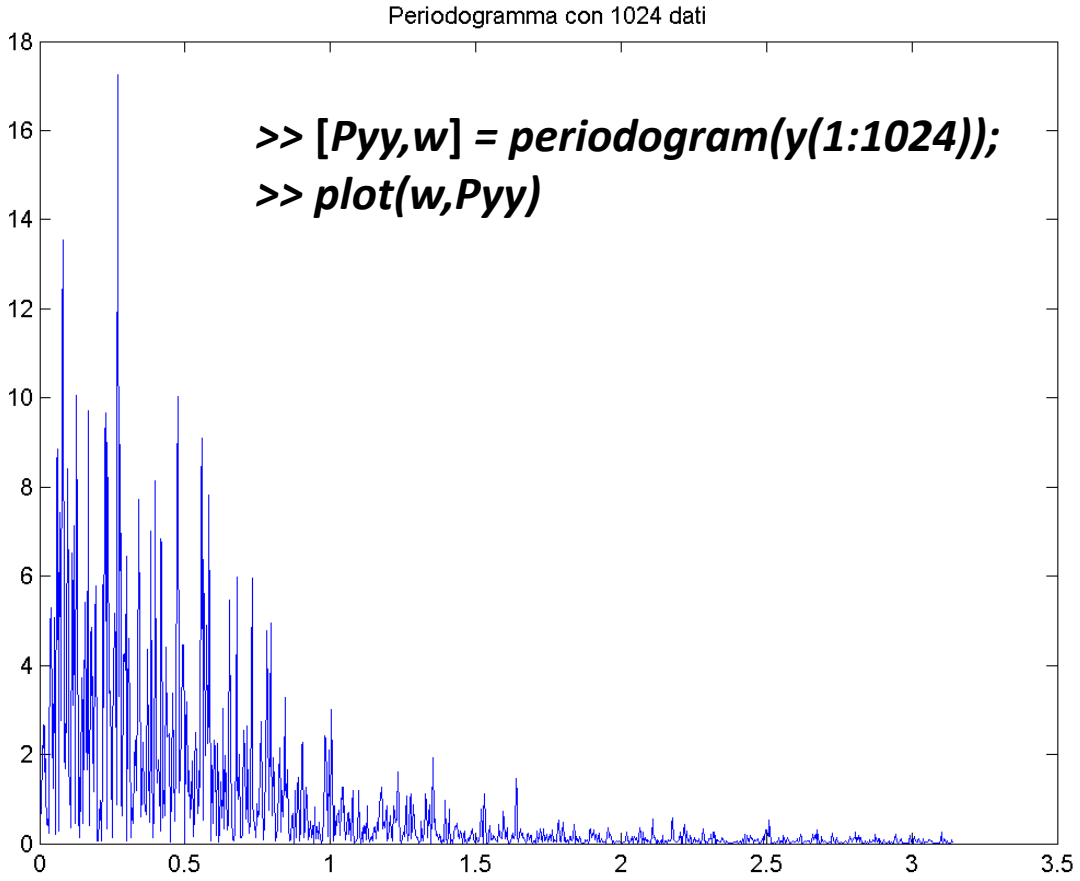


Periodogram

Plot of periodogram based on 1024 data points of a SSP.

The PSD is represented as a function of the angular frequency from 0 to π [rad/sample].

Uses FFT (good to use data in powers of 2)



Regularization of the spectral estimate

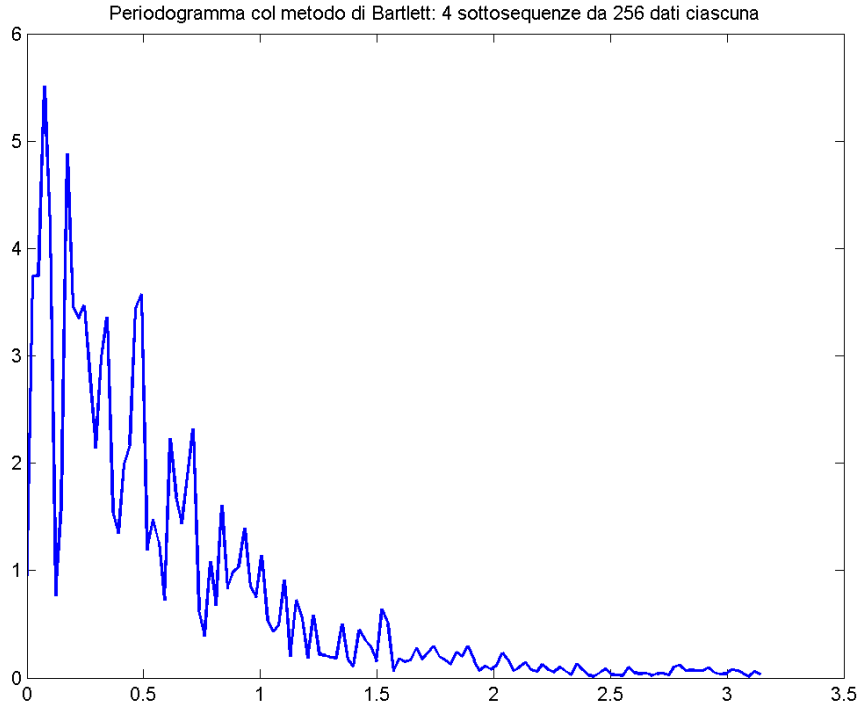
One of the most used methods to regularize the spectrum is the Bartlett approach, which splits data into sub-sequences, computes the periodogram over each sequence and then averages results with a weighted average using a “Bartlett window”

>> [Pyy,w]= pwelch(y(1:1024),bartlett(256),0);

where

bartlett(256) says that we are using a barlett window (among the many possible choices), with 256 samples

The last parameter indicated the overlap in the sequences: 0 means no overlap is used (the sequences are separated)



Predictors

Given the system

$$y(t) = \frac{B(z)}{A(z)}u(t-\tau) + \frac{C(z)}{A(z)}e(t), e \sim WN(0, \sigma^2)$$

The optimal k-steps predictor is given by

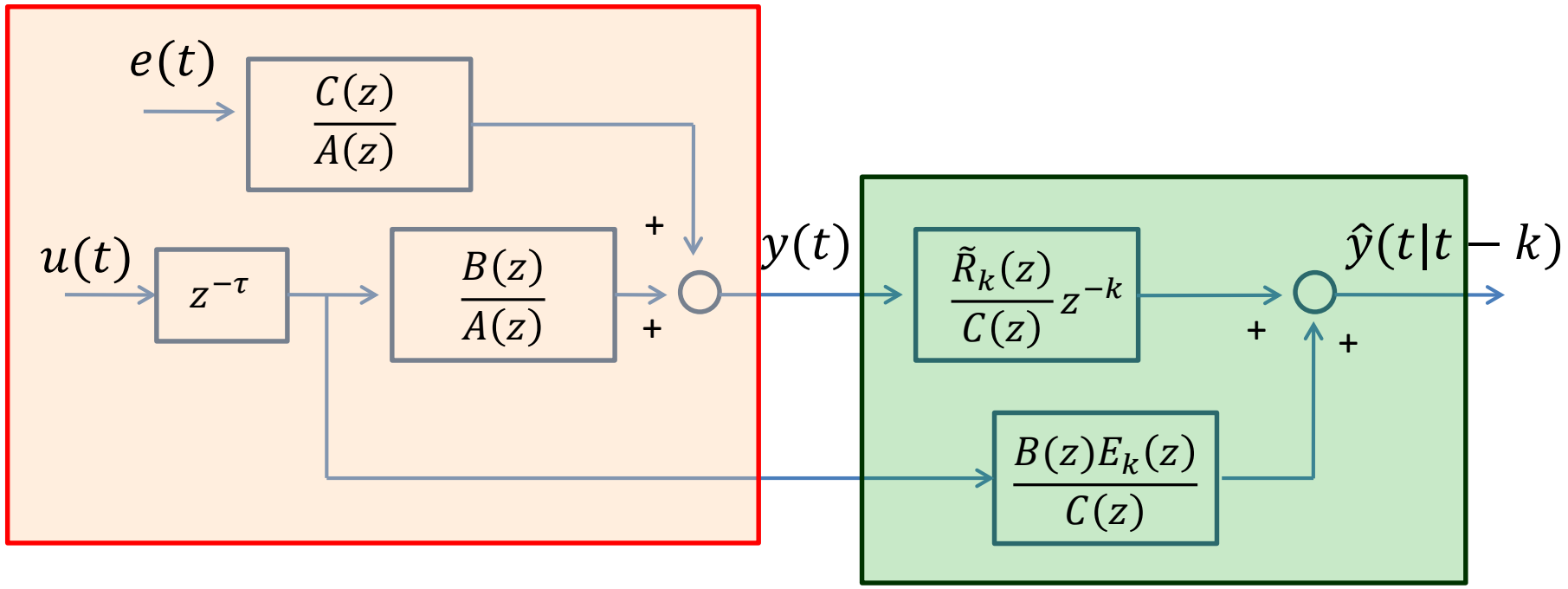
$$\hat{y}(t|t-k) = \frac{B(z)E_k(z)}{C(z)}u(t-\tau) + \frac{R_k(z)}{C(z)}y(t)$$

$$= \frac{B(z)E_k(z)}{C(z)}u(t-\tau) + \frac{\tilde{R}_k(z)}{C(z)}y(t-k)$$

where $E_k(z)$ e $R_k(z)$ are the result and the remainder, respectively, of the long division, and

$$R_k(z) = \tilde{R}_k(z)z^{-k}$$

Predictors


$e(t)$
$\frac{C(z)}{A(z)}$
$u(t)$
$z^{-\tau}$
$\frac{B(z)}{A(z)}$
+
+
$y(t)$
$\frac{\tilde{R}_k(z)}{C(z)} z^{-k}$
$\frac{B(z)E_k(z)}{C(z)}$
+
+
$\hat{y}(t|t-k)$

Predictors

```
>> [ypred, x0_est, pred_model] = predict(model, data, k)
INPUTS:
model   idpoly object
data        iddata object
k       prediction steps (integer number)
OUTPUT
ypred   y_hat(t+1/t), predicted output, iddata object
x0_est  estimate of the initial condition (for state_space model)
pred_model  prediction model (discrete-time state-space model):
```

$$\begin{cases} \hat{x}(t+1) &= A\hat{x}(t) + B\begin{bmatrix} y(t) \\ u(t) \end{bmatrix} + Ke(t) \\ \hat{y}(t) &= C\hat{x}(t) + D\begin{bmatrix} y(t) \\ u(t) \end{bmatrix} + e(t) \end{cases}$$

Predictors

```
>> pred_model_tf = tf(pred_model) % to obtain the transfer functions
```

From input "y1" to output "y-hat": $\frac{R(z)}{C(z)}$ It includes the prediction delay

From input "u1" to output "y1“: $\frac{B(z)E(z)}{C(z)}z^{-\tau}$ It includes the control delay

Predictors

- **EXAMPLE 1 (ARMAX model):**

```
>> Model_idpoly=idpoly([1 0.3],[0 1 0.2],[1
   0.5],[],[],1,1)
Discrete-time IDPOLY model: A(q)y(t) = B(q)u(t) +
   C(q)e(t)

A(q) = 1 + 0.3 q^-1
B(q) = 1 + 0.2 q^-1
C(q) = 1 + 0.5 q^-1
tau  = 1        % control delay
```

```
>> [yd,xoe,modpred]=predict(Model_idpoly,data,1);
>> tf(modpred)
From input "y1" to output "y1":
    0.2 z^-1
  ------------
  1 + 0.5 z^-1

From input "u1" to output "y1":
  z^-1 + 0.2 z^-2
  ---------------
   1 + 0.5 z^-1
```

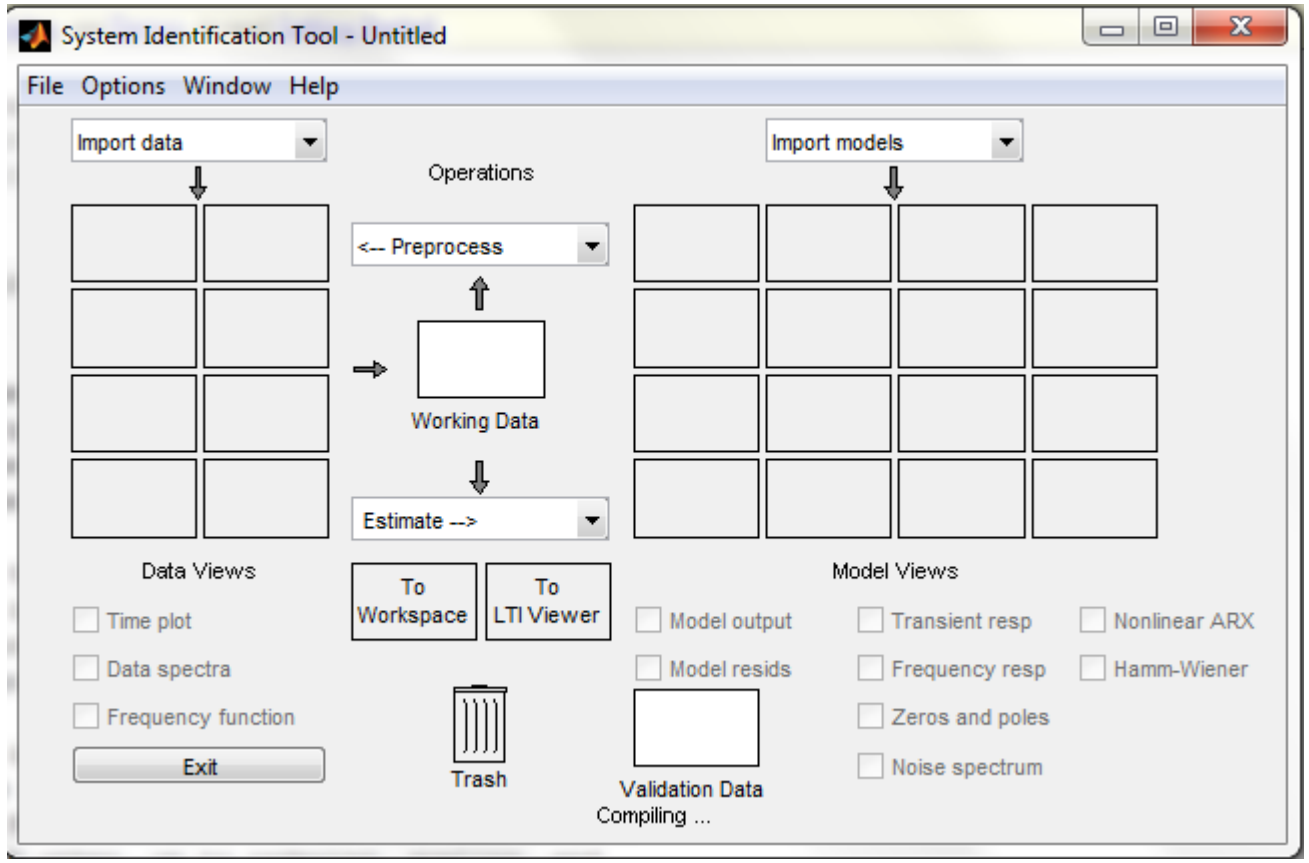
Predictors

- EXAMPLE 2 (ARX model):

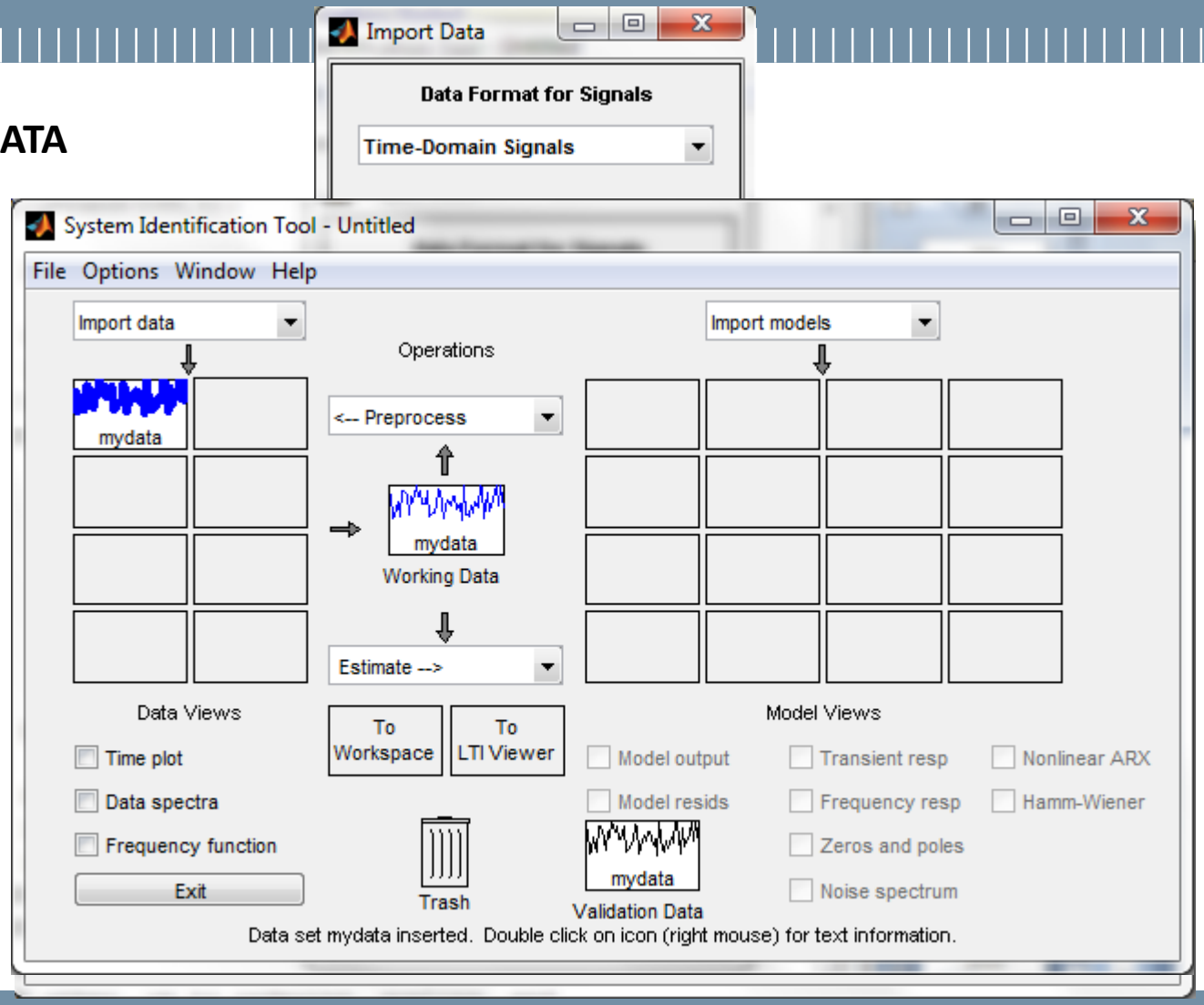
```
>> Model_idpoly=idpoly([1 0.3],[0 1 0.2],[1],[],[],1,1)
Discrete-time IDPOLY model: A(q)y(t) = B(q)u(t) + C(q)e(t)

A(q) = 1 + 0.3 q^-1
B(q) = 1 + 0.2 q^-1
C(q) = 1
tau  = 1        % control delay

>> [yd,xoe,modpred]=predict(Model_idpoly,data,1);
>> tf(modpred)

From input "y1" to output "y1":
  -0.3 z^-1
From input "u1" to output "y1":
  z^-1 + 0.2 z^-2
```

Identification

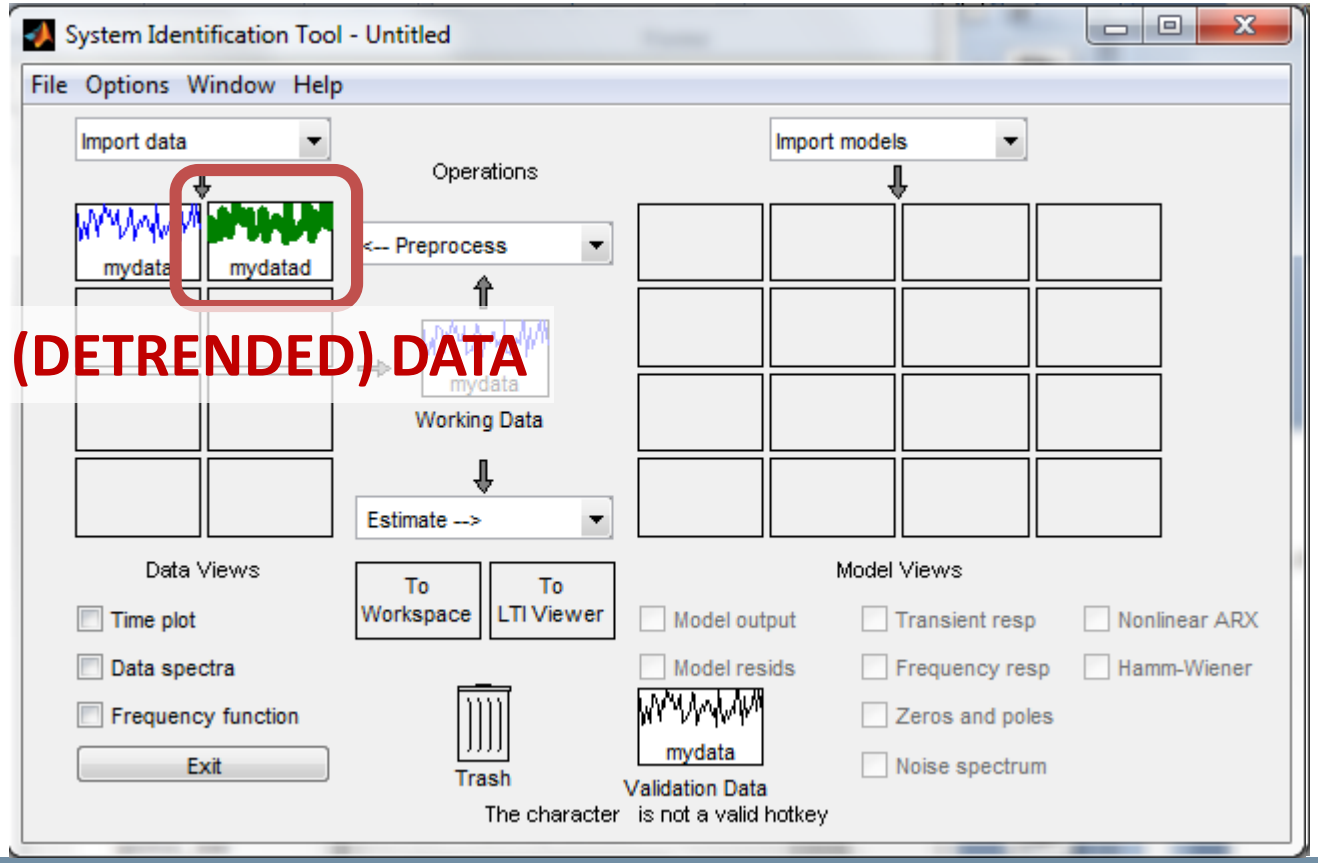
```
>> systemIdentification % start the MATLAB identification toolbox GUI
```



1.IMPORT DATA

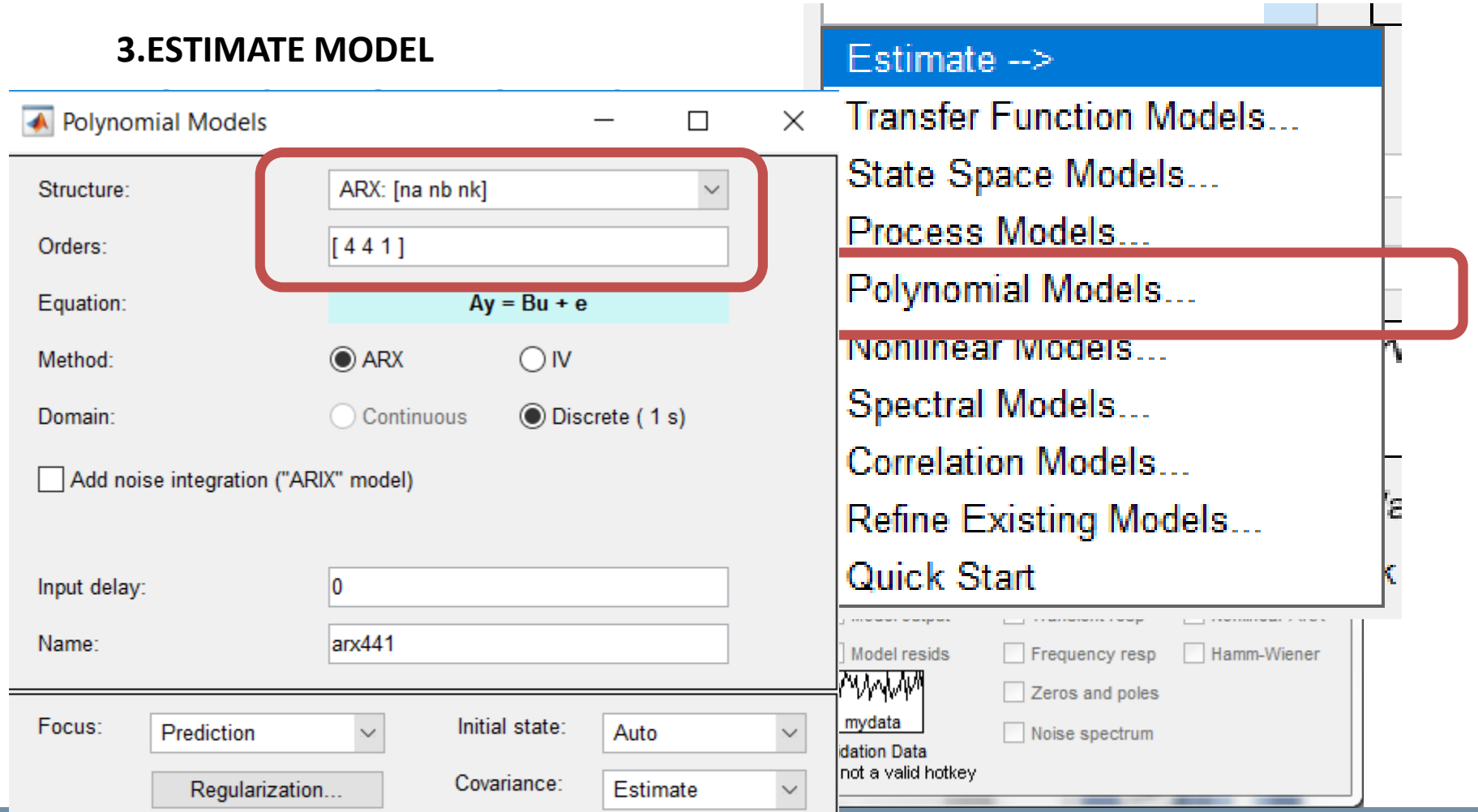


2.PREPROCESS DATA



NEW (DETRENDED) DATA

3.ESTIMATE MODEL



3.1. MODEL STRUCTURES

- **ARX(na,nb,k):** $A(z)y(t) = B(z)u(t) + e(t)$, where

  - $A(z)=1+ a\_1\ z^{-1}+ \dots + a\_na\ z^{-na}$
  - $B(z)=b\_1\ z^{-k} + \dots + b\_nb\ z^{-(k+nb-1)}$

  $y(t)=-a\_1\ y(t-1)-\dots-a\_na\ y(t-na) + b\_1\ u(t-k)+\dots+b\_nb\ u(t-n\_b-k+1)+e(t)$

- **ARMAX(na,nb,nc,k):** $A(z)y(t) = B(z)u(t) + C(z)e(t)$, where

  - $A(z)=1+ a\_1\ z^{-1}+ \dots + a\_na\ z^{-na}$
  - $B(z)=b\_1\ z^{-k} + \dots + b\_nb\ z^{-(k+nb-1)}$
  - $C(z)=1+ c\_1\ z^{-1}+ \dots + c\_nc\ z^{-nc}$

  $y(t)=-a\_1\ y(t-1)-\dots-a\_na\ y(t-na) + b\_1\ u(t-k)+\dots+b\_nb\ u(t-n\_b-k+1)$
  $+e(t)+c\_1\ e(t-1)+ \dots + c\_nc\ e(t-nc)$

Identification – other model structures (not seen in our course)

3.1. MODEL STRUCTURES

- **OE(nb,nf,k)**: $y(t) = B(z)/F(z)u(t) + e(t)$, where

  - $B(z)=b\_1\ z^{-k} + \dots + b\_nb\ z^{-(k+nb-1)}$
  - $F(z)=1+ f\_1\ z^{-1}+ \dots + f\_nf\ z^{-nf}$

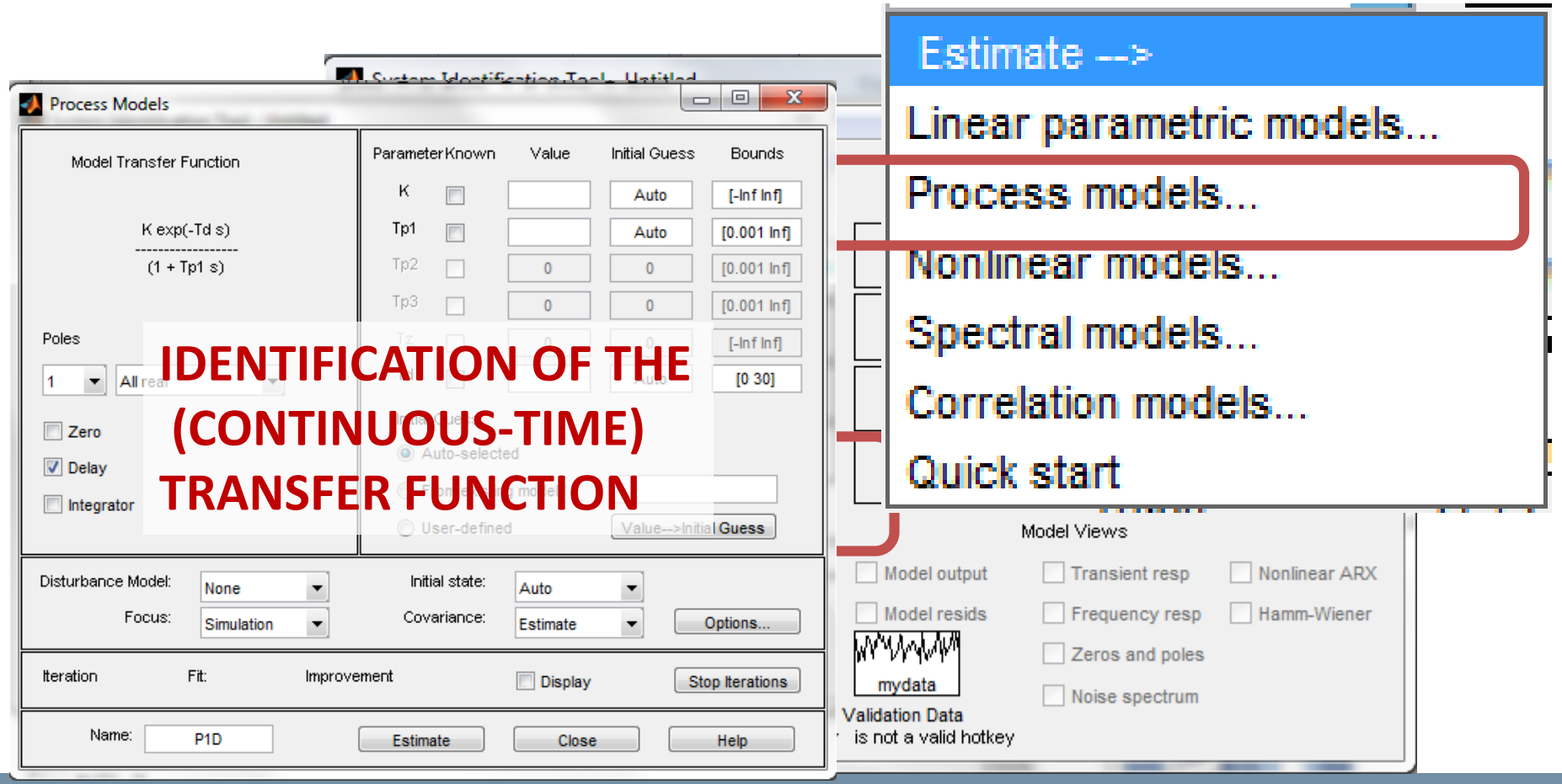
$$y(t)=-f\_1\ y(t-1)-\dots-f\_na\ y(t-nf) + b\_1\ u(t-k)+\dots+b\_nb\ u(t-n\_b-k+1) + e(t)+f\_1\ e(t-1)+ \dots + f\_nf\ e(t-nf)$$

- **BOX-JENKINS(nb,nc,nd,nf,k):** $y(t) = B(z)/F(z)u(t) + C(z)/D(z)e(t)$

- **STATE-SPACE (n: order) [PEM or subspace identification]**

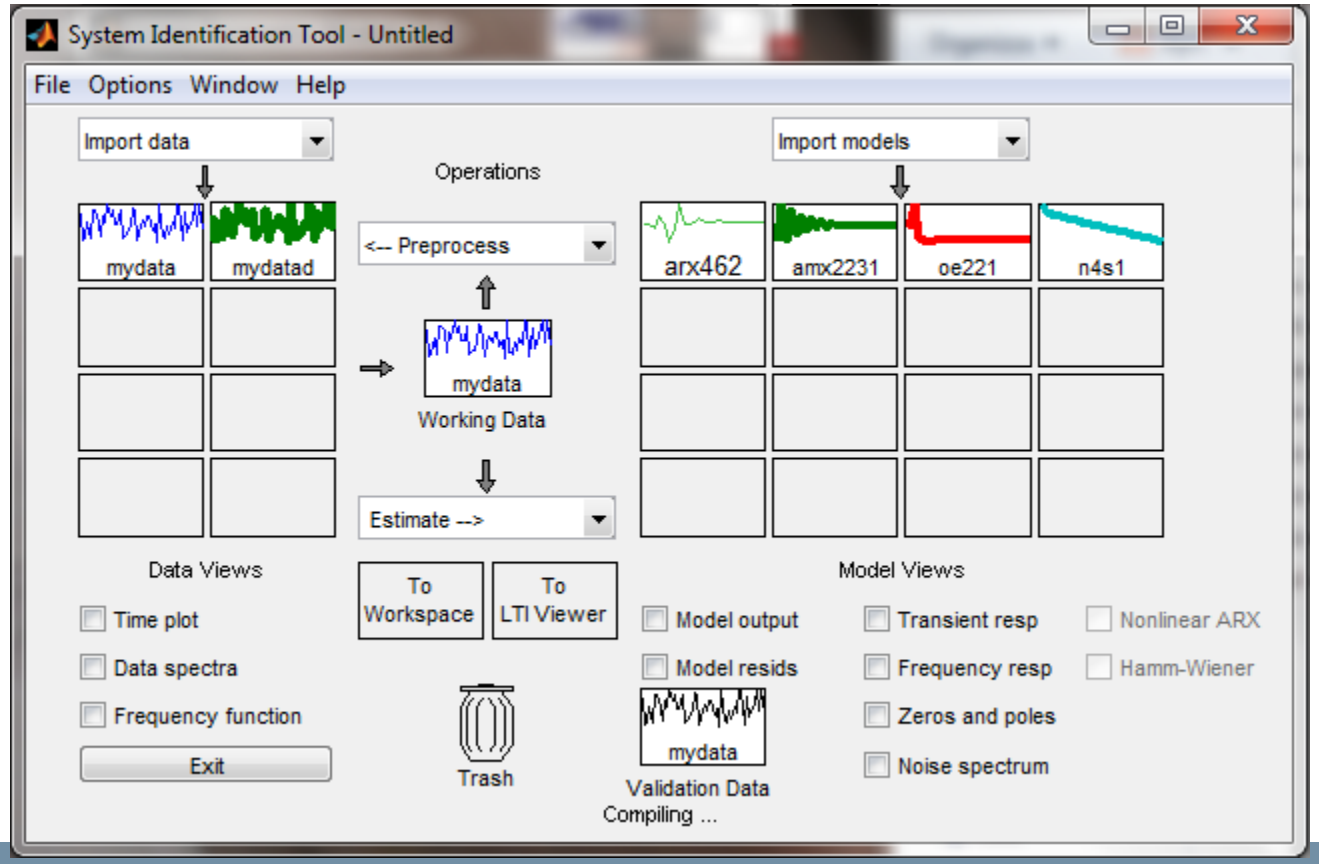
$$x(t+1)=Ax(t)+Bu(t)+Ke(t)$$
$$y(t)=Cx(t)+Du(t)+e(t)$$

3.ESTIMATE MODEL



IDENTIFICATION OF THE (CONTINUOUS-TIME) TRANSFER FUNCTION

4.Compare and validate models



Parametric estimation

For MA e ARMA(X) models, we use

>> model = armax(data,'na',na,'nb',nb,'nc',nc,'nk',nk)

dove:

- *data* iddata object with data points;
- *na* order of autoregressive part
- *Nb* order of eXogenous part
- *nc* order of MA part
- *nk* intrinsic delay
- *model* is the output, an idpoly object with all coefficients plus info on the uncertainty related to the results

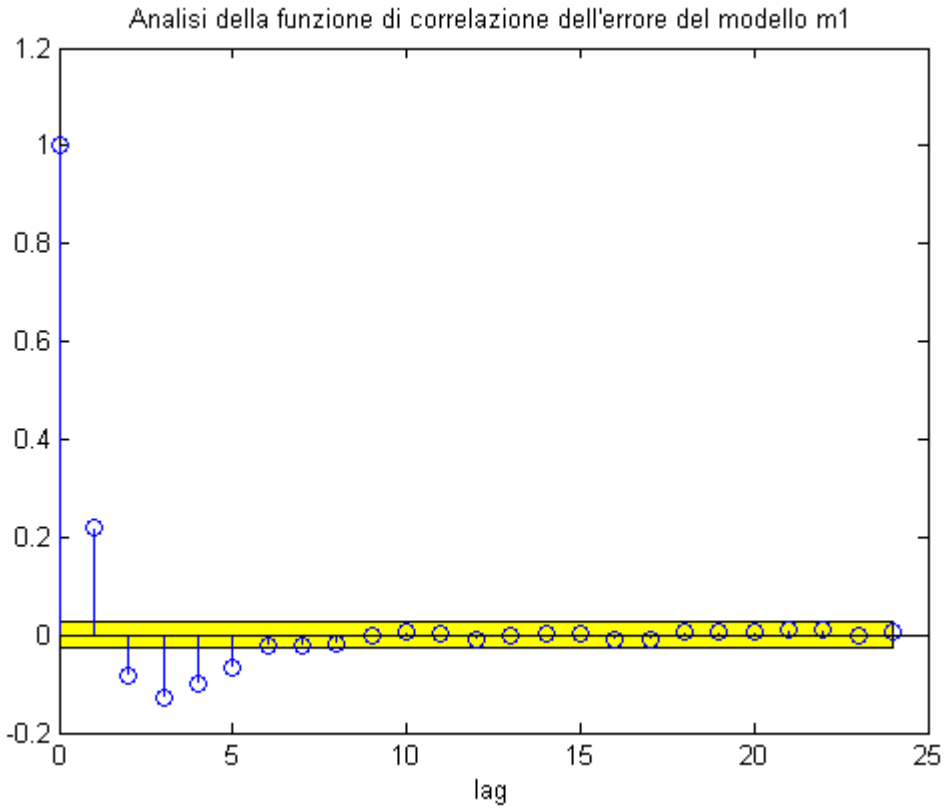
Residual analysis

– ARDERSON TEST FOR THE RESIDUAL (PREDICTION ERROR) → whiteness test for the residual

```
>> e=resid(model_idpoly,data)

inputs  model: IDPOLY format
        data:  IDDATA format (input-output data)
outputs e:     IDDATA ([prediction error,input])

If the outputs are not specified, a plot is shown:
- Empirical correlation function (empirical - sampled) of the prediction error
- 99% confidence region
  -> if more than 99% of the samples of the normalized correlation function
  lie in the confidence region: positive results of the whiteness Test!
```

Residual analysis

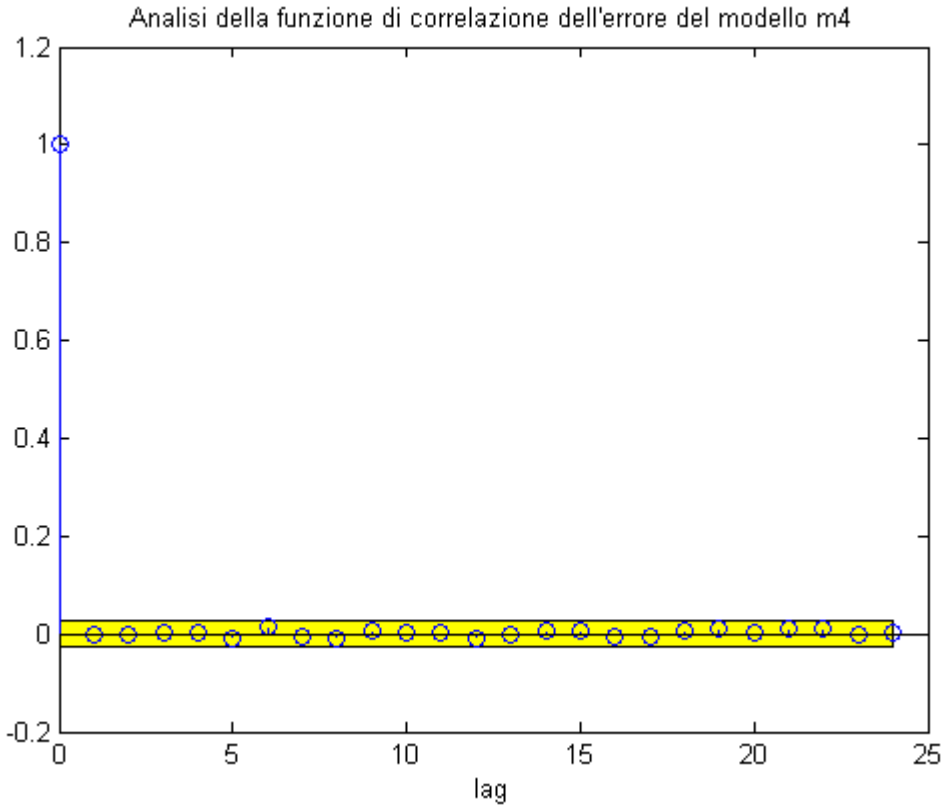
Non-white prediction error, the data generation mechanism does not belong to the model class!



Residual analysis

The residual is white: only the first sample of the autocorrelation is non-zero!

The data generation mechanism belongs to the model class!



Choice of the optimal order

To choose the optimal model complexity, we use cross-correlation approaches.

Two criteria that allow doing it are the Final Prediction Error (FPE) e the Akaike Information Criterion (AIC). The functions:

>> aic(model)

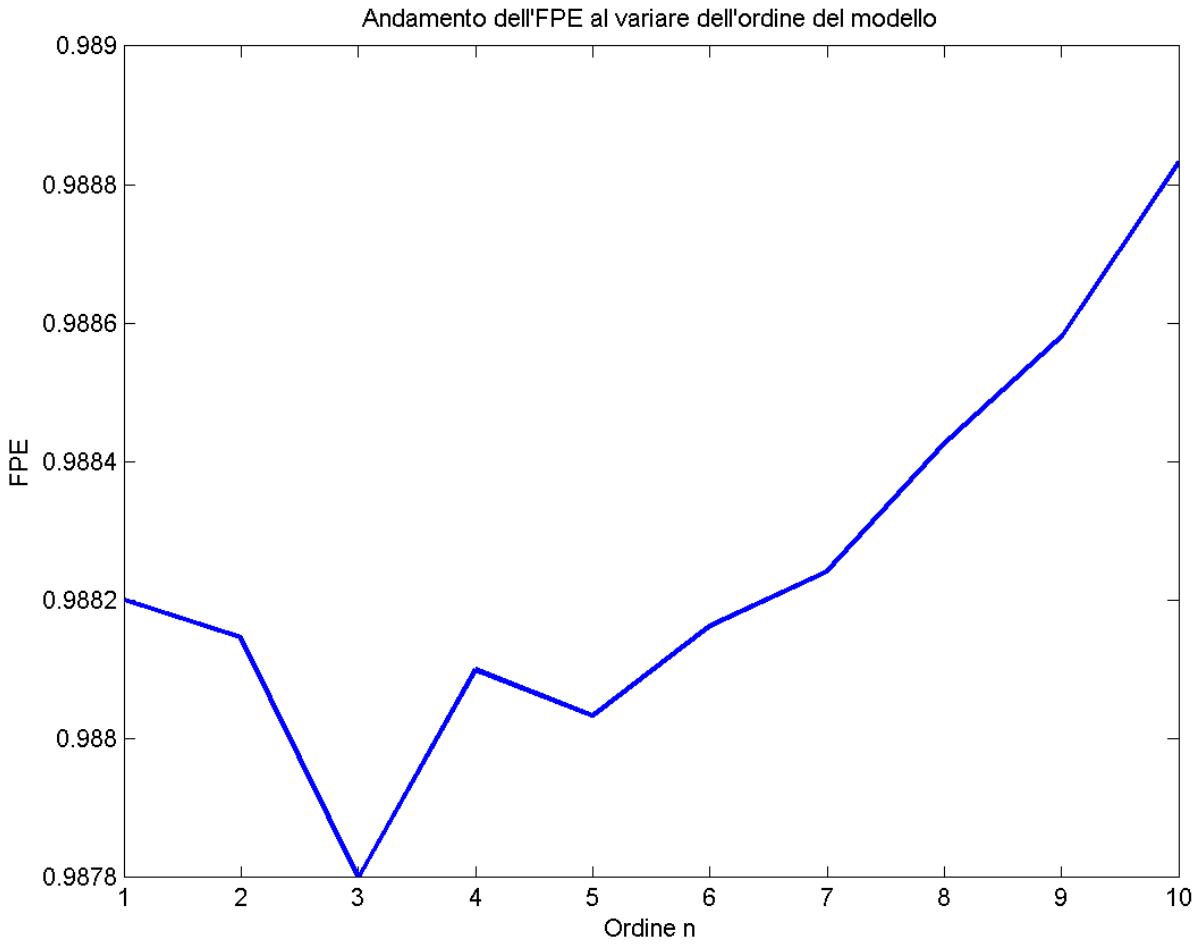
>>fpe(model)

allow to compute these cost functions and find the optimal order as that minimizing it

Choice of the optimal order

FPE as the model order varies $n=1,\ldots,10$ for an AR(n).

The optimal order is $n=3$.



Identification

For further details, see

http://it.mathworks.com/products/sysid/

Other related Matlab functions (2 Identification toolbox)

Parametric model estimation

- `ar` - AR-models of signals using various approaches.
- `armax` - Prediction error estimate of an ARMAX model.
- `arx` - LS-estimate of ARX-models.

Model structure selection

```
– aic             - Compute Akaike's information criterion.
– fpe             - Compute final prediction error criterion.
– arxstruc        - Loss functions for families of ARX-models.
– selstruc        - Select model structures according to
          various criteria.
– idss/setstruc   - Set the structure matrices for idss
          objects.
– struc           - Generate typical structure matrices for
          ARXSTRUC.
```

For more details please see:

```
>> help ident
```

Exercises

Linear systems

1. Consider the discrete-time linear system, characterized by the following transfer function

$$W(z) = \frac{1+z^{-1}}{1-0.8z^{-1}}$$

1.1 Plot the step response of the system. What are its features, in the light of the properties (stability, gain, etc.) of the system?

1.2 Plot the response of the system to two different sinusoidal inputs, i.e., $u(t)=\sin(0.1t)$ and $u(t)=\sin(t)$. Analyze the properties of the two output signals in the light of the frequency response theorem.

Identification of time series

Consider the MA(1) stochastic process *y*(*t*), generated from the dynamic system

$$y(t) = e(t) + 0.5e(t-1)$$

with input *e*(*t*), i.e., white Gaussian noise with mean $\mu = 0$ and variance $\lambda^2 = 1$.

Define a realization of *y*(*t*) consisting of *N*=1000 samples.

1. Assume that the data generation system is unknown and the model class, selected for identification purposes, consists of a AR(1) process, i.e.,

$$y(t) = \vartheta y(t-1) + \xi(t)$$

where $\xi(t)$ is a zero mean white noise process.

1.1 Compute, using the «ident» GUI, the optimal estimate $\hat{\vartheta}_N$ of ϑ.

Exercises

1.2 Define the evolution of the prediction error $\varepsilon(t) = y(t) - \hat{y}(t|t-1)$. Estimate its mean and variance.

1.3. Assuming that the data generation system is known, compute analytically the value of ϑ that minimizes the cost function $\bar{J}(\vartheta) = E[\varepsilon(t)^2]$, and the corresponding value of $\bar{J}(\vartheta)$.

Model Identification

Consider the ARMAX system y(t)=0.1 y(t-1)+2u(t-1)-0.8 u(t-2)+e(t)+0.3e(t-1)
Where e(t) is a white noise with zero mean and unitary variance.

1. Generate a 10000-samples realization of the output signal y(t), when the input u(t) is a persistently exciting signal (e.g., a white noise), and where y(0)=0;

2. Identify, using the identification toolbox GUI:
    1. an ARX(1,1,1) model
    2. an ARX(1,2,1) model
    3. an ARMAX(1,2,2,1) model
    4. an ARMAX(1,1,2,1) model
3. Compare and analyze the results (possibly add validation data to validate the results).