



# *Aktorik und Sensorik mit intelligenten Materialsystemen 4*

## *Computer Lecture: Numerical Linearization of a SMA-Spring Actuator in Matlab*

Gianluca Rizzello

[gianluca.rizzello@mmsl.uni-saarland.de](mailto:gianluca.rizzello@mmsl.uni-saarland.de)

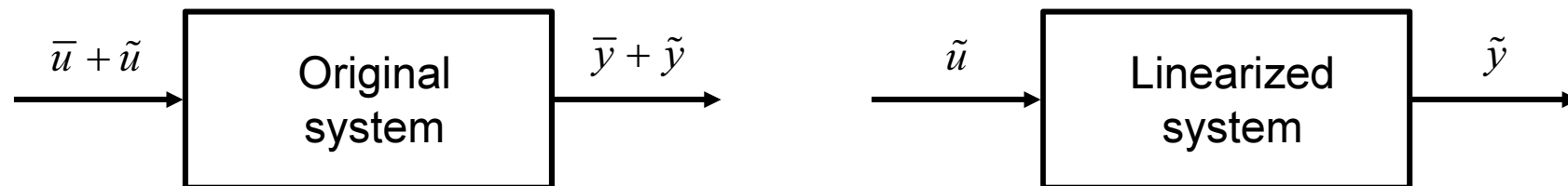
Saarland University



- GOALS:
  - Obtain a linearized model describing the response of a SMA-spring actuator around an equilibrium state via a numerical identification method
  - Identification of Joule heating – displacement linearized dynamics
  - Identification of environmental temperature – displacement linearized dynamics
  - Identification of load force – displacement linearized dynamics



- Numerical linearization: obtain a linearized model describing relationship between input deviations and output deviations from an equilibrium point



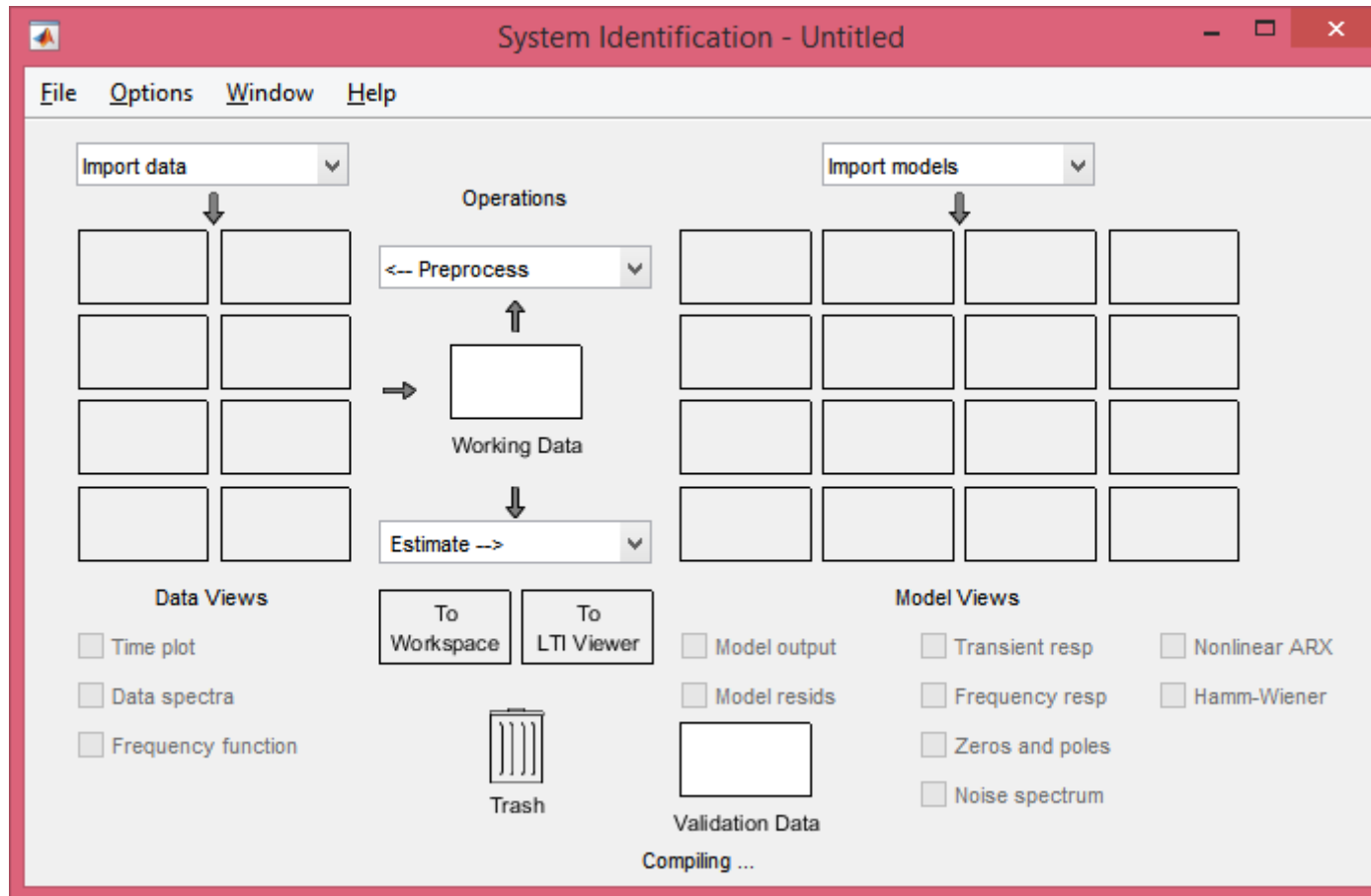
- This operation can be performed in Matlab via System Identification Toolbox
- To open the System Identification Toolbox GUI, type the following command in Matlab:

`systemIdentification`



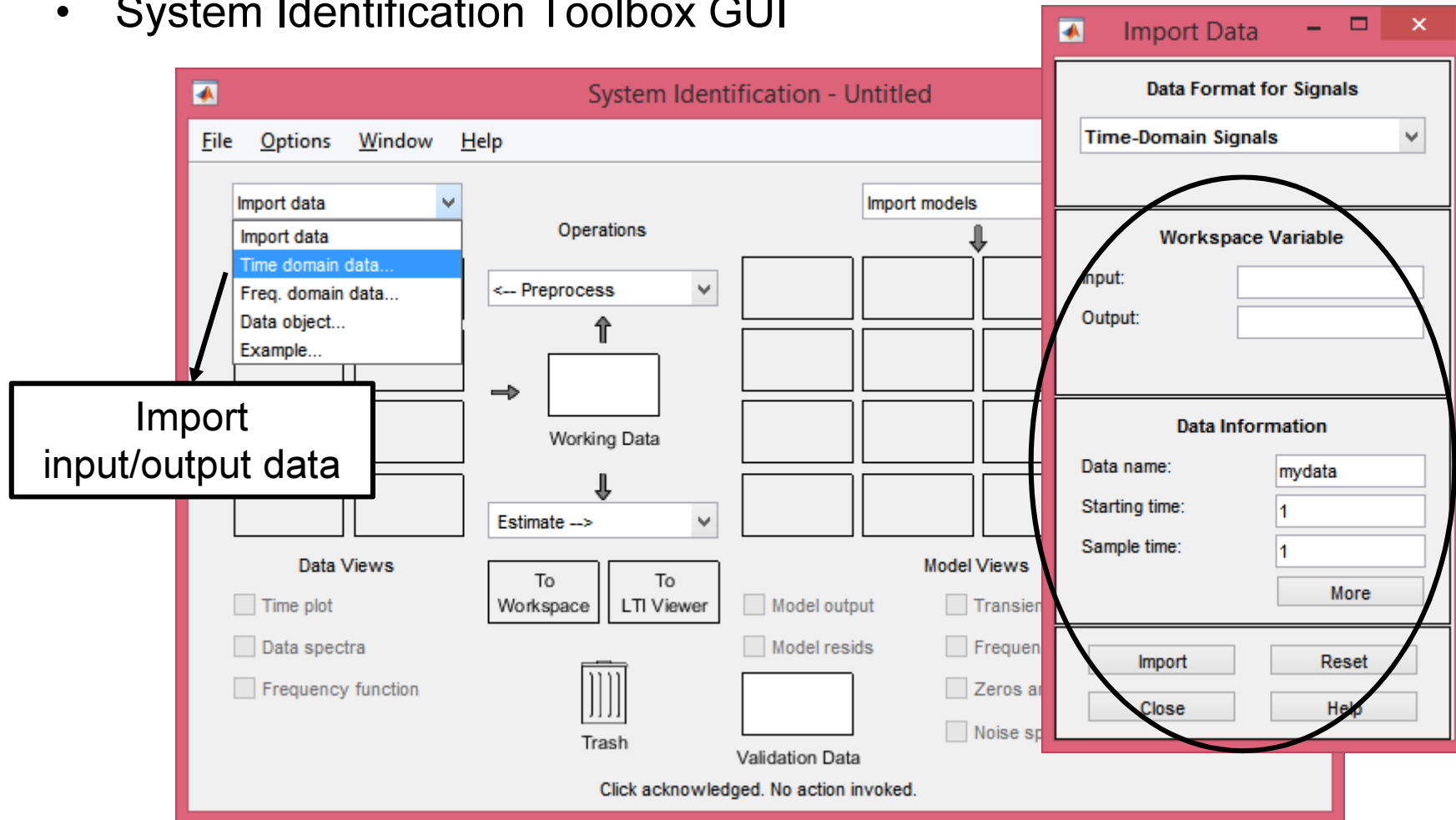
# System Identification Toolbox

- System Identification Toolbox GUI





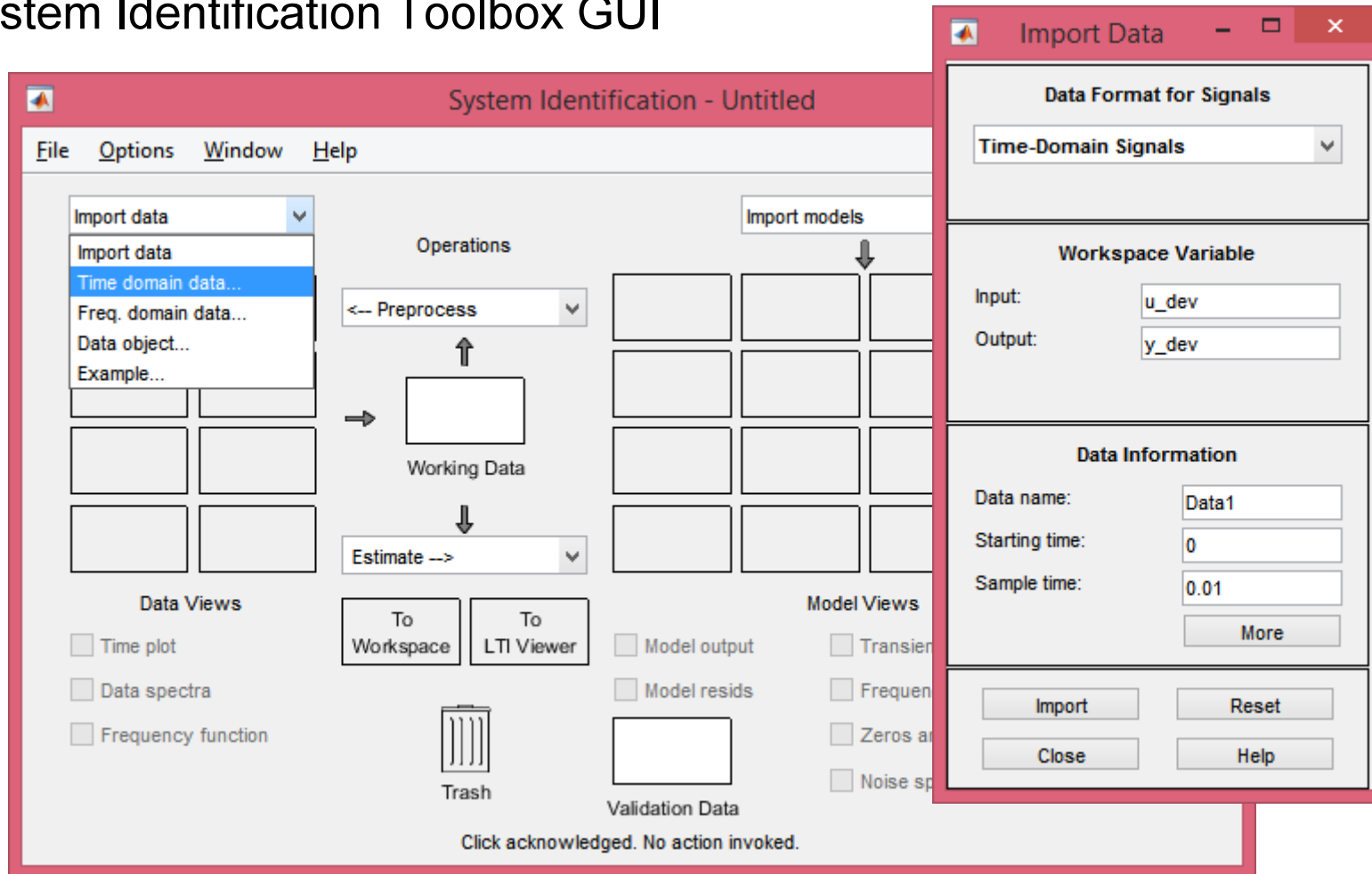
- System Identification Toolbox GUI



Specify here all the required information, then select 'Import'



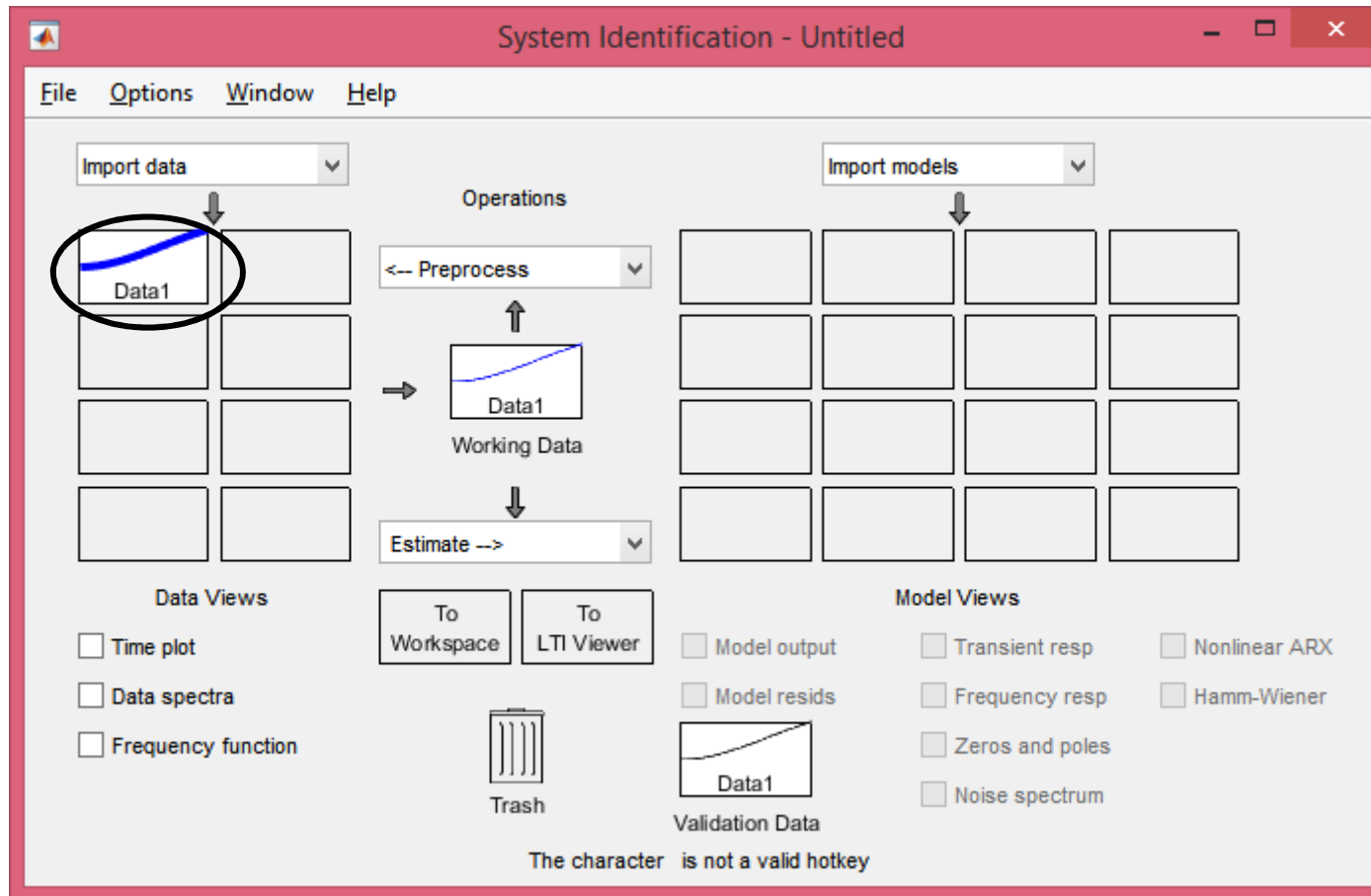
- System Identification Toolbox GUI



Input and output must be column vectors with same size,  
sampled with uniform sampling time



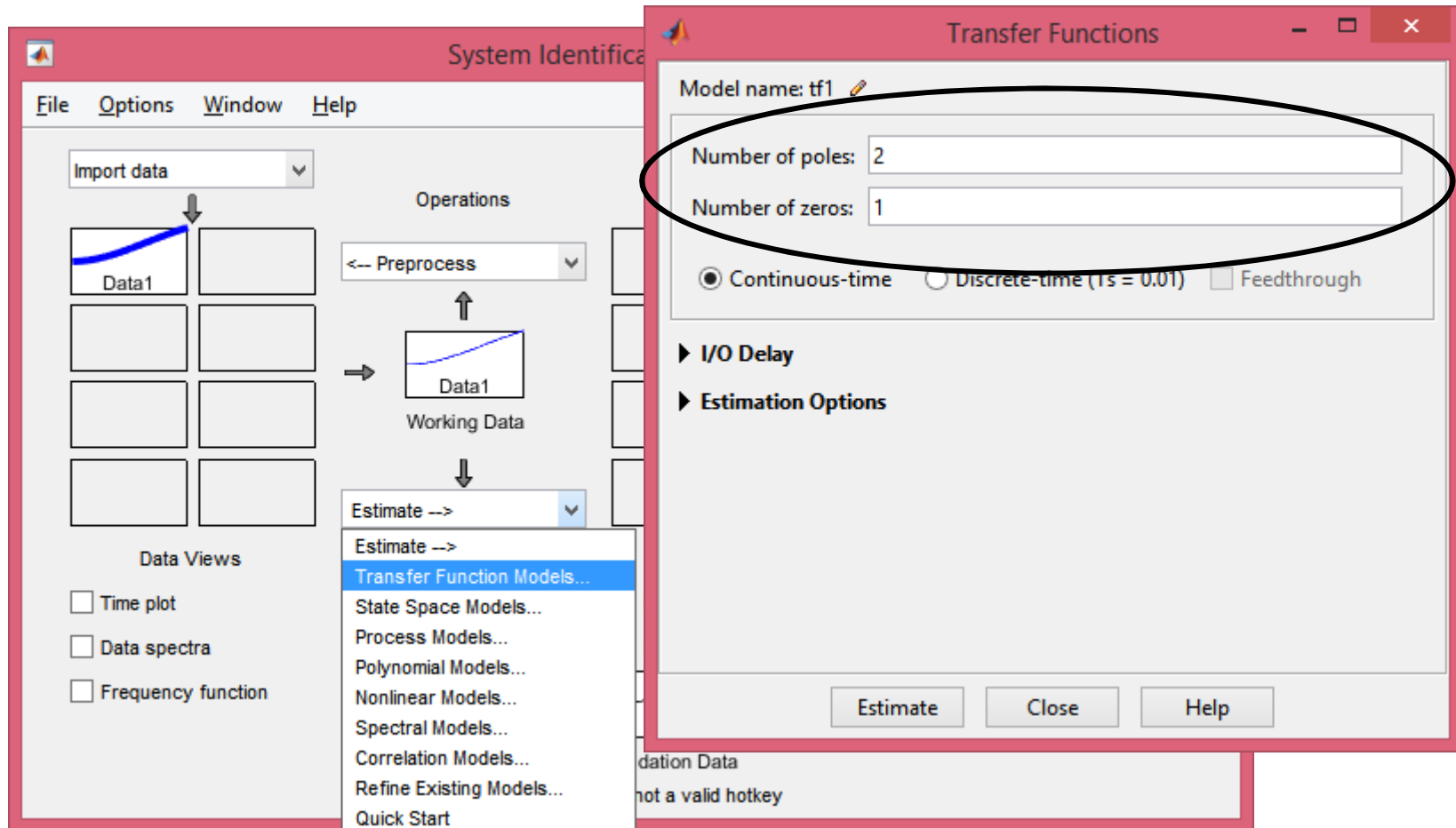
- System Identification Toolbox GUI



Data have now been imported



- System Identification Toolbox GUI



From the new window, choose number of zeros and poles for the optimal model, then select 'Estimate'





- System Identification Toolbox GUI

The screenshot displays two windows from the System Identification Toolbox GUI. The 'Plant Identification Progress' window on the left shows the results of a transfer function identification process. It includes a table of estimation progress and a final result section. The 'Transfer Functions' window on the right shows the estimated transfer function model.

**Plant Identification Progress**

Transfer Function Identification  
Estimation data: Time domain data Data1  
Data has 1 outputs, 1 inputs and 1001 samples.  
Number of poles: 2, Number of zeros: 0  
Initialization Method: "iv"

Estimation Progress

| Iteration | Cost        | Norm of step | First-order optimality | Improvement (%) Expected | Achieved |
|-----------|-------------|--------------|------------------------|--------------------------|----------|
| 0         | 3.44104e-05 | -            | 303                    | 6.86e+05                 | -        |
| 1         | 2.93244e-05 | 2.66         | 2.49e+03               | 6.86e+05                 | 14.8     |
| 2         | 2.82216e-05 | 1.03         | 614                    | 1.58e+05                 | 3.76     |
| 3         | 2.81543e-05 | 0.266        | 19.6                   | 1.16e+04                 | 0.238    |
| 4         | 2.81488e-05 | 0.0845       | 2.27                   | 953                      | 0.0192   |
| 5         | 2.81484e-05 | 0.0231       | 0.445                  | 79.7                     | 0.0016   |

Result  
Termination condition: Near (local) minimum, (norm(g) < tol).  
Number of iterations: 9, Number of function evaluations: 19  
Status: Estimated using TFEEST with Focus = "simulation"  
Fit to estimation data: 83.65%, FPE: 2.8431e-05

**Transfer Functions**

f1

Poles: 2  
Zeros: 0

Continuous-time ☐ Discrete-time (Ts = 0.01) ☐ Feedthrough

Options

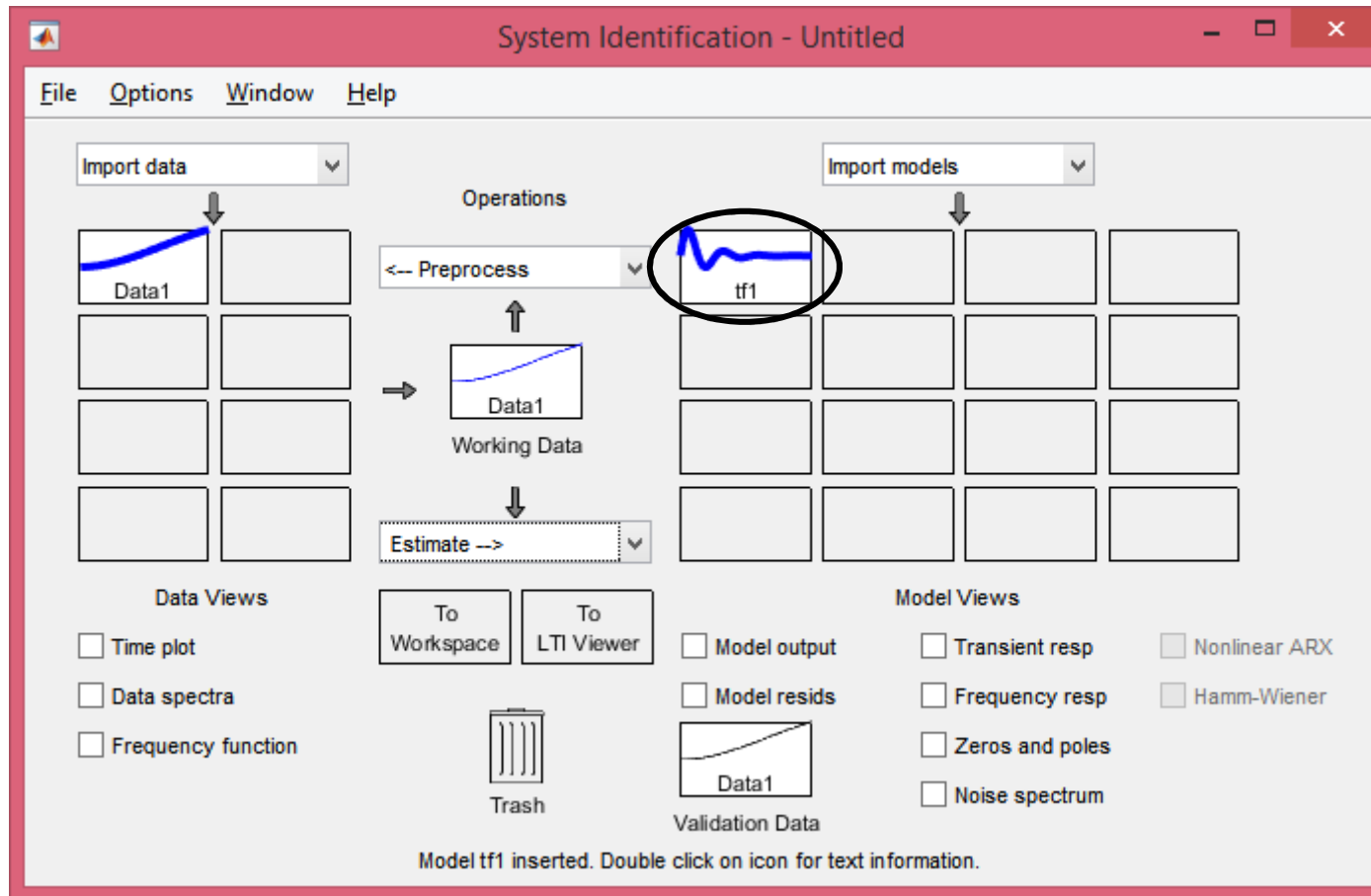
Estimate Close Help

for text information.

FIT of the resulting model



- System Identification Toolbox GUI



The resulting model is saved here



- System Identification Toolbox GUI

The screenshot displays the System Identification Toolbox GUI. The main window has a menu bar (File, Options, Window, Help) and a toolbar. The 'Import data' dropdown is set to 'Data1'. The 'Operations' section shows a workflow: 'Data1' (with a plot) is processed by 'Preprocess' and then 'Estimate -->'. The 'Data Views' section has checkboxes for 'Time plot', 'Data spectra', and 'Frequency function'. The 'To Workspace' and 'To LTI Viewer' buttons are visible. A 'Trash' icon is at the bottom. The 'Data/model Info: tf1' dialog box is open, showing the model name 'tf1' and color '[0,0,1]'. The transfer function is displayed as 
$$4.336 \frac{s^2 + 2.466 s + 22.02}{s^2 + 2.466 s + 22.02}$$
 (Note: the image shows a simplified representation of the transfer function). The 'Diary and Notes' section contains the following MATLAB code: 

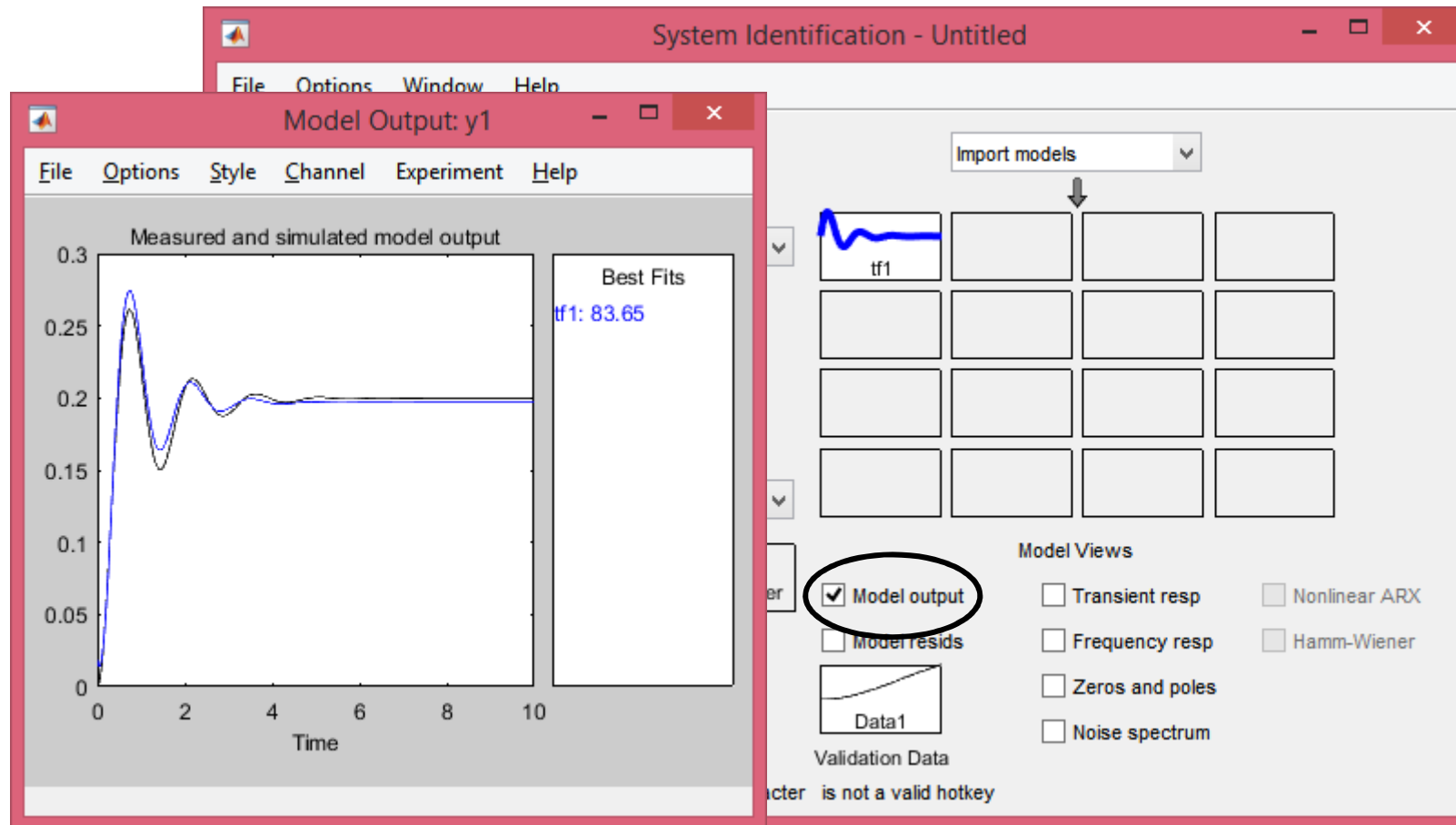
```
% Import Data1
% Transfer function estimation
Options = tfestOptions;
Options.Display = 'on';
```

 The 'Show in LTI Viewer' button is highlighted. At the bottom of the dialog are 'Present', 'Export', 'Close', and 'Help' buttons.

Right-click to show the identified transfer function model, which can be eventually exported in Matlab as a transfer function object



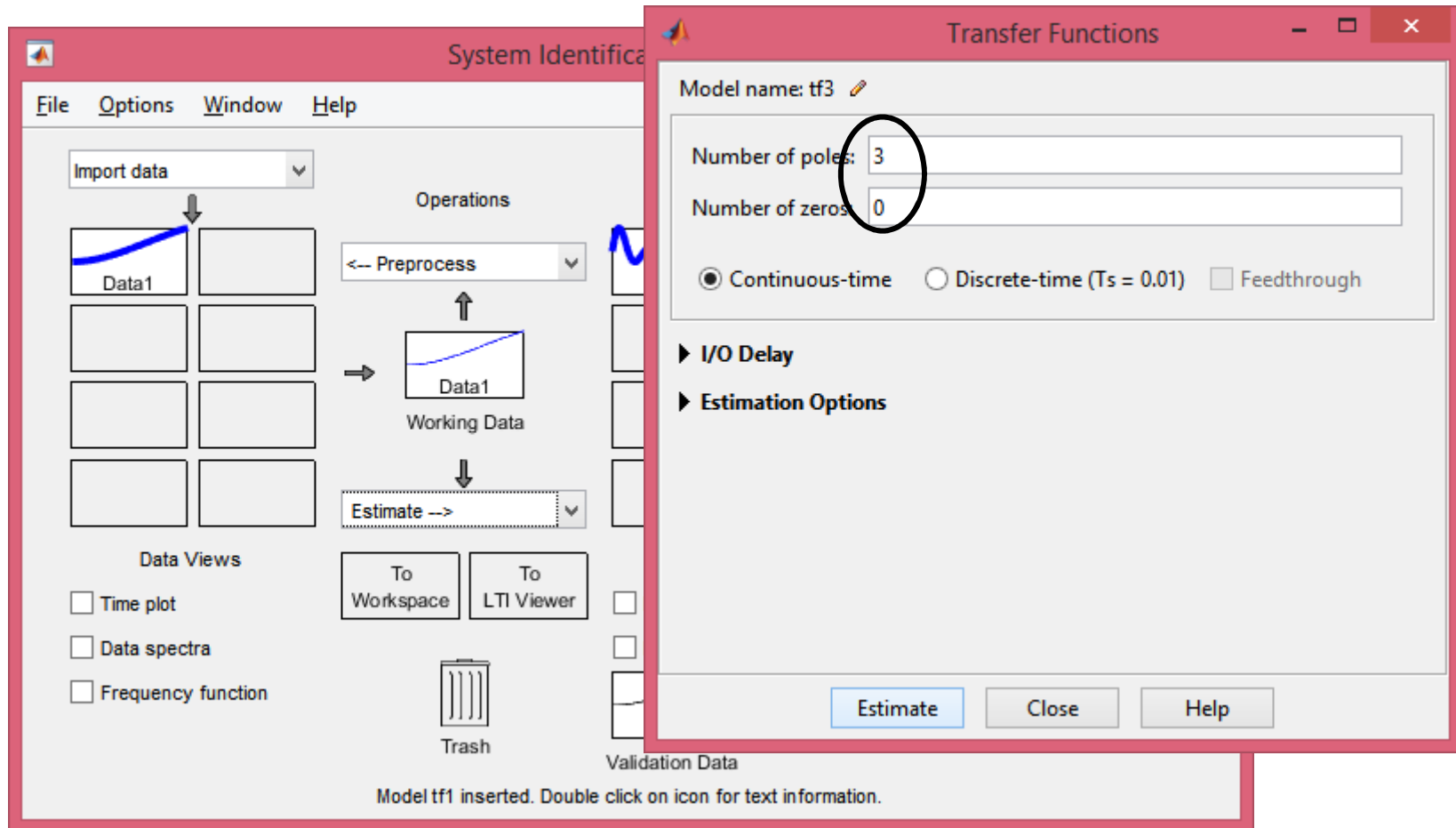
- System Identification Toolbox GUI



To show time-domain response, select 'Model output'



- System Identification Toolbox GUI



To improve the accuracy, we can increase the complexity of the model, e.g., by increasing the number of zeros and poles



- System Identification Toolbox GUI

The screenshot displays two windows from the System Identification Toolbox GUI. The 'Plant Identification Progress' window shows the results of a transfer function identification. The 'Transfer Functions' window shows the configuration for the identified model.

**Plant Identification Progress**

Transfer Function Identification  
Estimation data: Time domain data Data1  
Data has 1 outputs, 1 inputs and 1001 samples.  
Number of poles: 3, Number of zeros: 0  
Initialization Method: "iv"

**Estimation Progress**

| Iteration | Cost Function | Relative Error | Number of Poles | Number of Zeros |
|-----------|---------------|----------------|-----------------|-----------------|
| 14        | 1.92053e-05   | 7.63           | 1.81e+05        | 4.76e+06        |
| 15        | 1.7342e-05    | 4.83           | 1.82e+05        | 5.05e+06        |
| 16        | 1.73133e-05   | 6.81           | 1.84e+05        | 5.66e+06        |
| 17        | 1.28682e-05   | 3.29           | 2.05e+05        | 5.75e+06        |
| 18        | 8.8393e-06    | 4.23           | 2.47e+05        | 7.76e+06        |
| 19        | 1.82716e-06   | 3.26           | 5.42e+05        | 1.13e+07        |

**Result**

Termination condition: Maximum number of iterations reached.  
Number of iterations: 20, Number of function evaluations: 106

Status: Estimated using TFEST with Focus = "simulation"  
Fit to estimation data: **99.98%**, FPE: 3.08923e-11

**Transfer Functions**

Model name: tf3

Number of poles: 3  
Number of zeros: 0

☒ Continuous-time ☐ Discrete-time (Ts = 0.01) ☐ Feedthrough

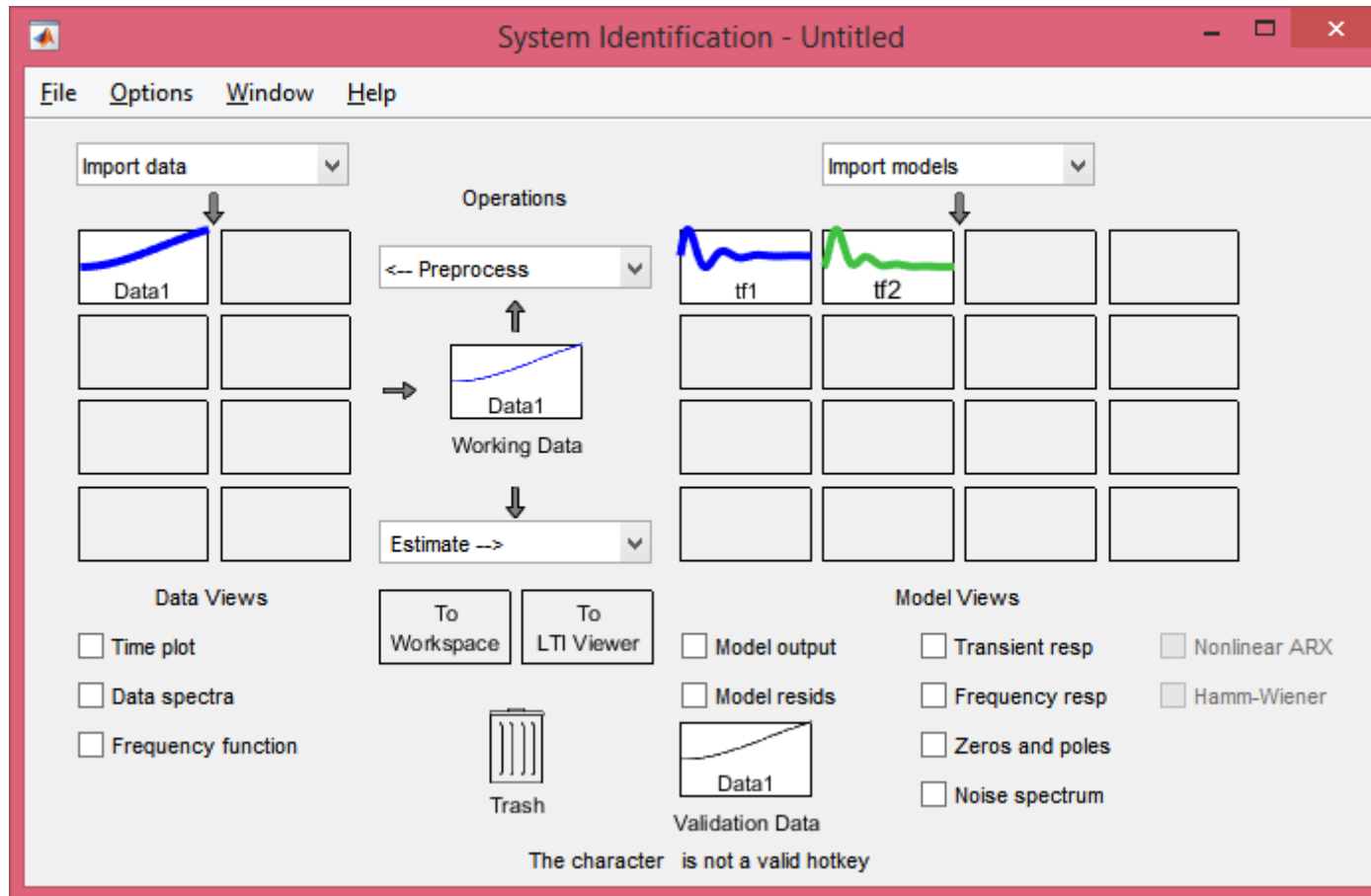
**I/O Delay**  
**Estimation Options**

Estimate Close Help

In this particular case, we get an almost perfect FIT



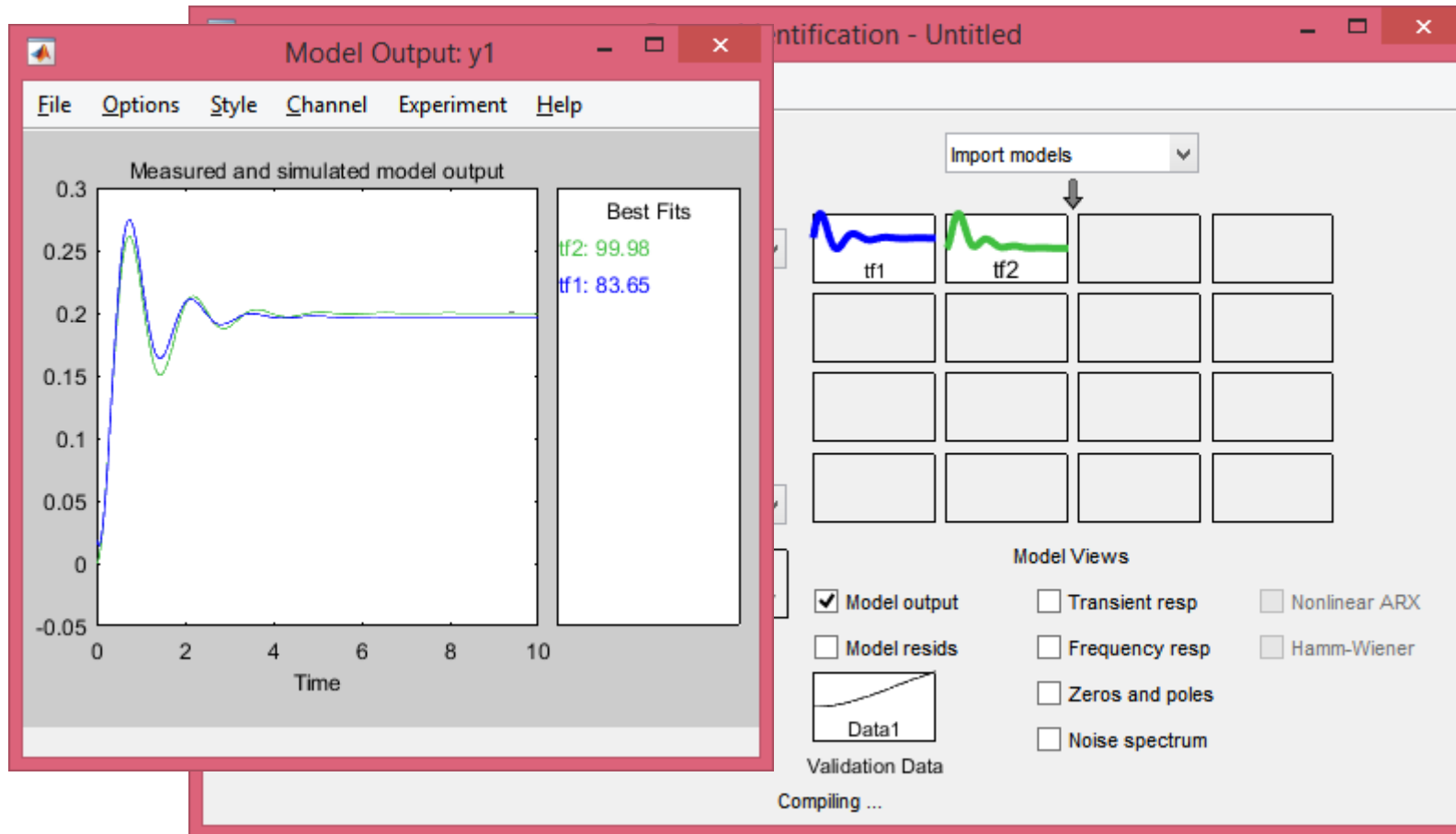
- System Identification Toolbox GUI



In this particular case, we get an almost perfect FIT



- System Identification Toolbox GUI

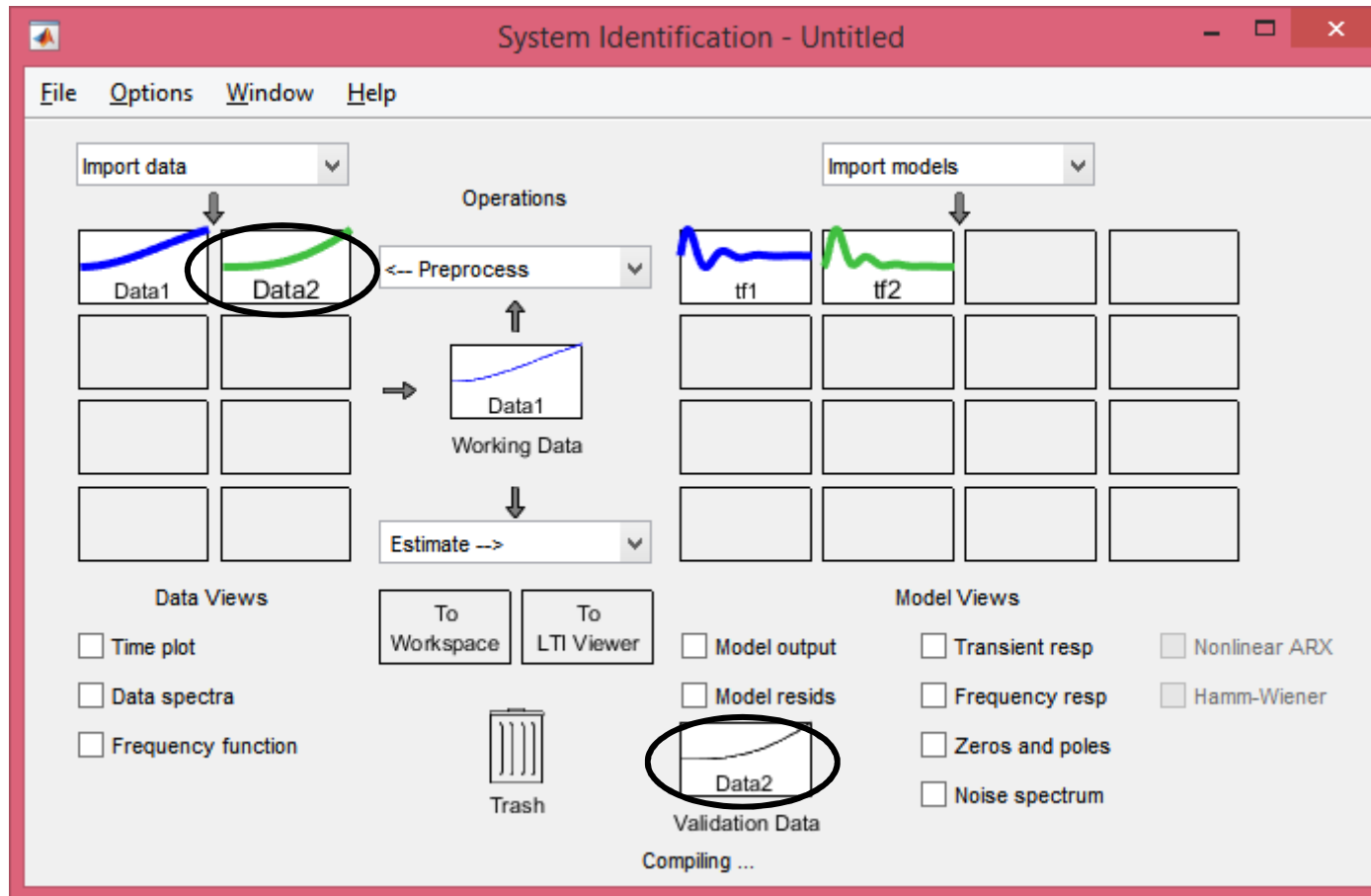


In this particular case, we get an almost perfect FIT





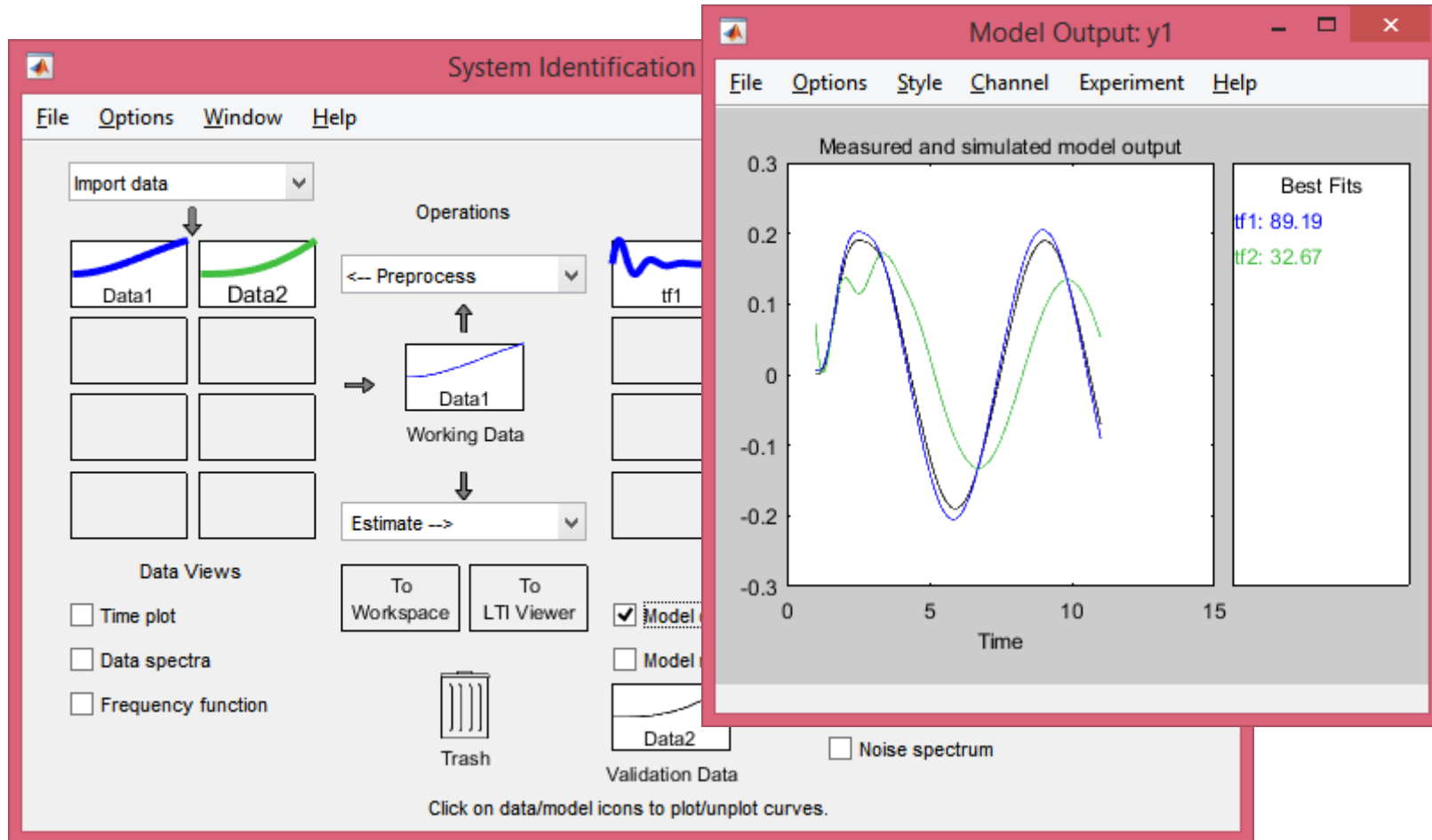
- System Identification Toolbox GUI



New data can be eventually imported and used to perform validation of the model



- System Identification Toolbox GUI



New data can be eventually imported and used to perform validation of the model



- To represent a transfer function object in Matlab, one can use the command

$$G = \text{tf}(\text{num}, \text{den})$$

where `num` and `den` are arrays containing coefficients of numerator and denominator polynomials, respectively, while `G` is the transfer function object

- Example 1: to represent the following transfer function

$$G(s) = \frac{2s + 1}{s^3 + 4s^2 + 4s + 2}$$

we use the following command

$$G = \text{tf}([2 \ 1], [1 \ 4 \ 4 \ 2])$$

- Example 2: to represent products between factors, such as in

$$G(s) = \frac{2(s+3)}{(s^2 + 2s + 2)(s+5)}$$

one can use the command `conv`

$$G = \text{tf}(2*[1 \ 3], \text{conv}([1 \ 2 \ 2], [1 \ 5]))$$



- Useful commands on transfer functions:
  - `step(G)` – shows step response of  $G$
  - `impz(G)` – shows impulse response of  $G$
  - `lsim(G,u,t)` – computes response of  $G$  to input  $u$  with respect to time  $t$
  - `pzmap(G)` – shows poles  $\times$  and zeros  $\circ$  of  $G$  on the complex plane
  - `bode(G)` – shows Bode plots of  $G$
  - `nyquist(G)` – shows Nyquist plot of  $G$
  - `zpk(G)` – factors zeros and poles of  $G$
  - $G_3 = G_1 + G_2$  – computes  $G_3$  as the sum of  $G_1$  and  $G_2$ , i.e., parallel connection
  - $G_3 = G_1 * G_2$  – computes  $G_3$  as the product of  $G_1$  and  $G_2$ , i.e., series connection
  - $T = G / (1 + G)$  – computes  $T$  as the negative feedback connection of  $G$
- All of these functions are also valid for state-space objects as well
- Transfer function objects can be also converted to state-space and vice-versa
  - $S = \text{tf2ss}(G)$  – converts transfer function  $G$  into state-space object  $S$
  - $G = \text{ss2tf}(S)$  – converts state-space object  $S$  into transfer function  $G$



# Assignment

- Given the SMA-spring model, consider the equilibrium state corresponding to the following constant inputs:
  - $J = 0.125 \text{ W}$
  - $T_E = 293 \text{ K}$
  - $F_L = 0 \text{ N}$
- 1) By using the Simulink model, bring the system to the desired equilibrium point.
- 2) Apply a very small variations of  $J$ , and measure the corresponding small variation of output displacement. By using the System Identification Toolbox GUI, identify a suitable transfer function model which relates the two deviations from equilibrium.  
HINT: for generating the small input deviation you can use different signals, including steps, bipolar square waves, or sinewaves.
- 3) Repeat the previous step by progressively increasing the amount of variation of  $J$  from the equilibrium, and compare the resulting transfer functions obtained.
- 4) Repeat the previous identification by only producing variations on  $T_E$ .
- 5) Repeat the previous identification by only producing variations on  $F_L$ .