Control Lab
www.a-lab.ee

TALLINN UNIVERSITY OF TECHNOLOGY

ISS0023 Intelligent Control Systems

Laboratory Work: Application of Fractional-order Calculus to Identification and Control of a Process

Aleksei Tepljakov

December 2017

IT Akadeemia
toetab Skype™

Laboratory work overview

In this laboratory work it will be shown how fractional-order calculus can be applied to control problems using a set of CACSD tools for MATLAB. You will learn how to—

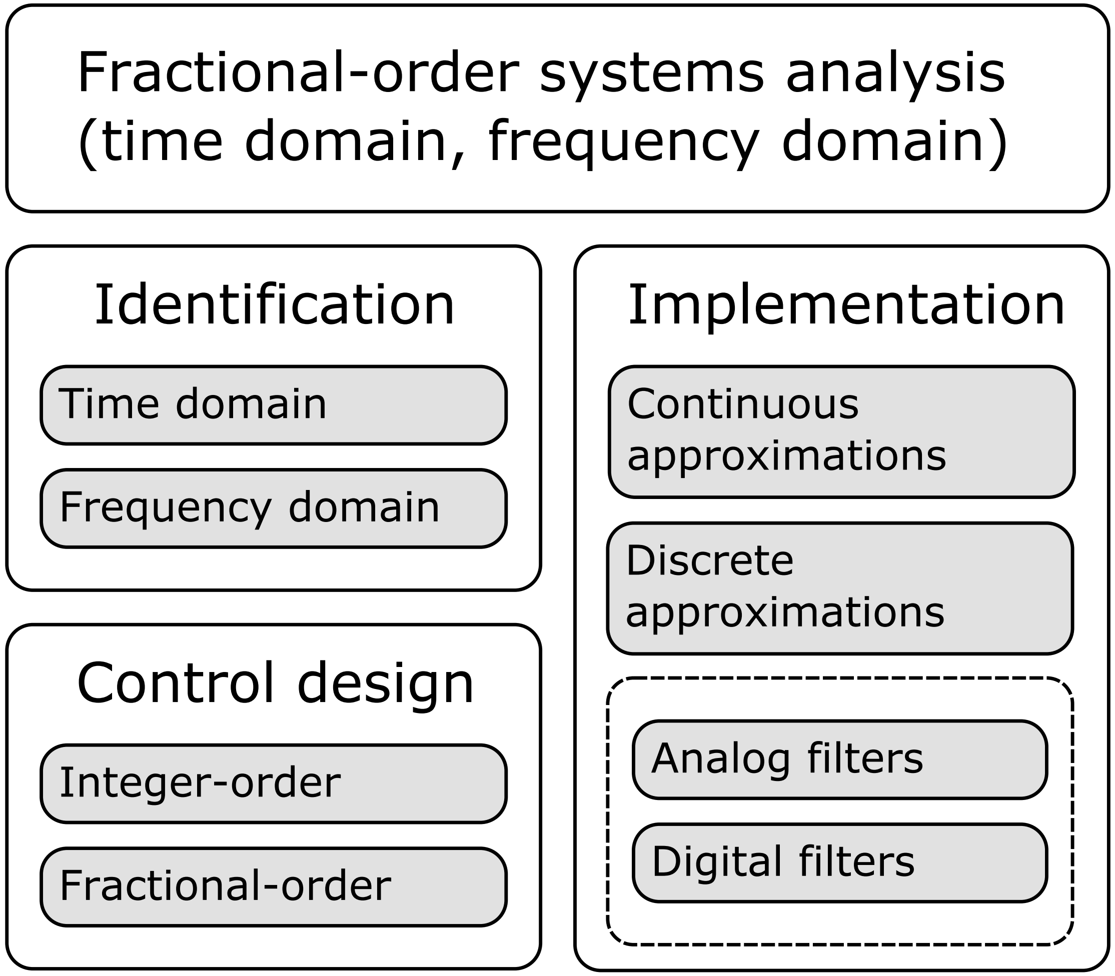
- Install the FOMCON toolbox and use the tools it provides;
- Obtain and validate a fractional-order dynamic model of a process from an experimental data set using time-domain identification;
- Use the obtained model to design a fractional-order PID controller under some performance specifications;
- Evaluate the performance of the designed controller with the original plant in Simulink.

FOMCON project: Fractional-order Modeling and Control

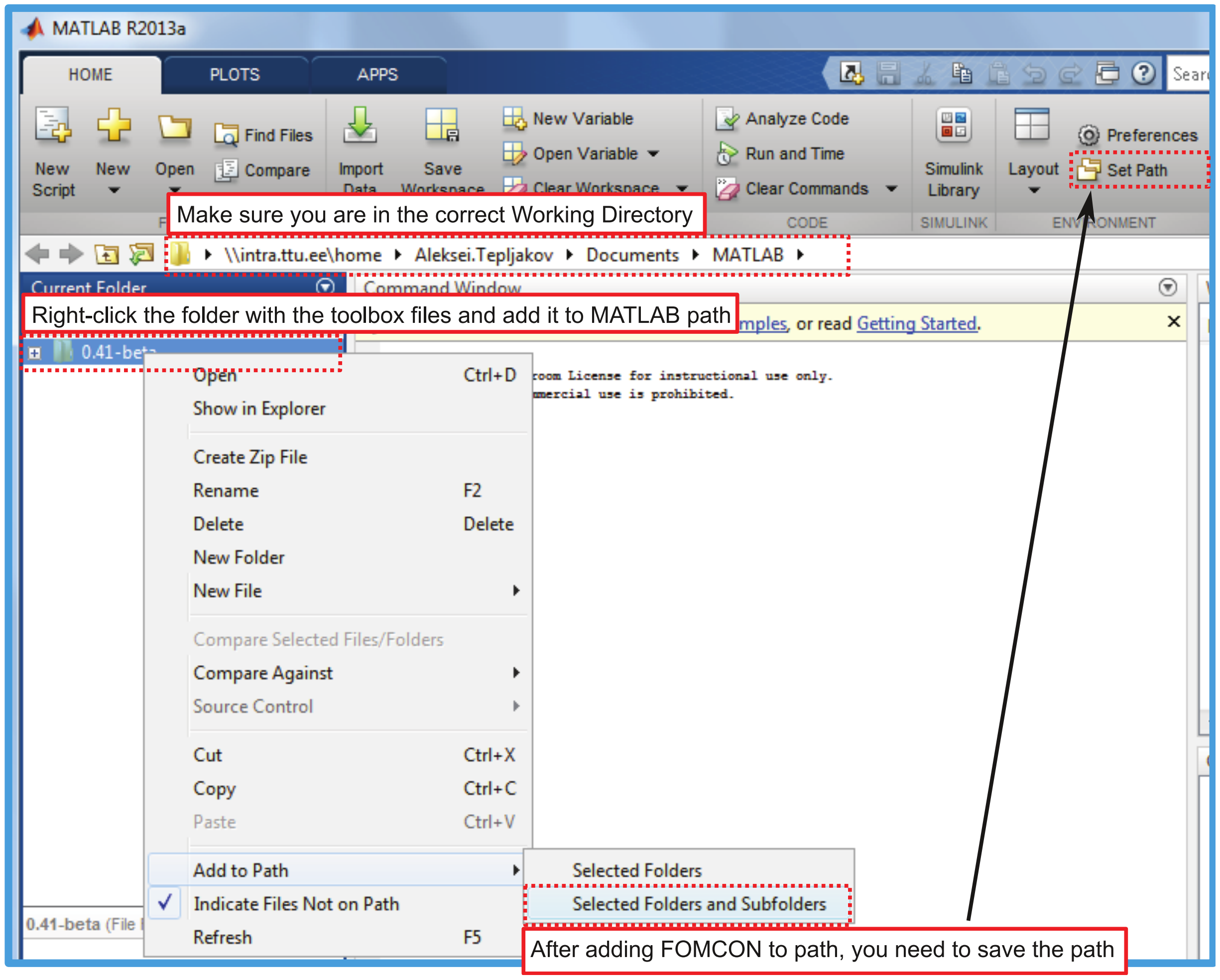
FOMCON

- Official website: `http://fomcon.net/`
- Toolbox for MATLAB available, development via GitHub: `https://github.com/AlekseiTepljakov/fomcon-matlab`
- Recently: Added initial support for studying FO MIMO systems.

FOMCON toolbox structure



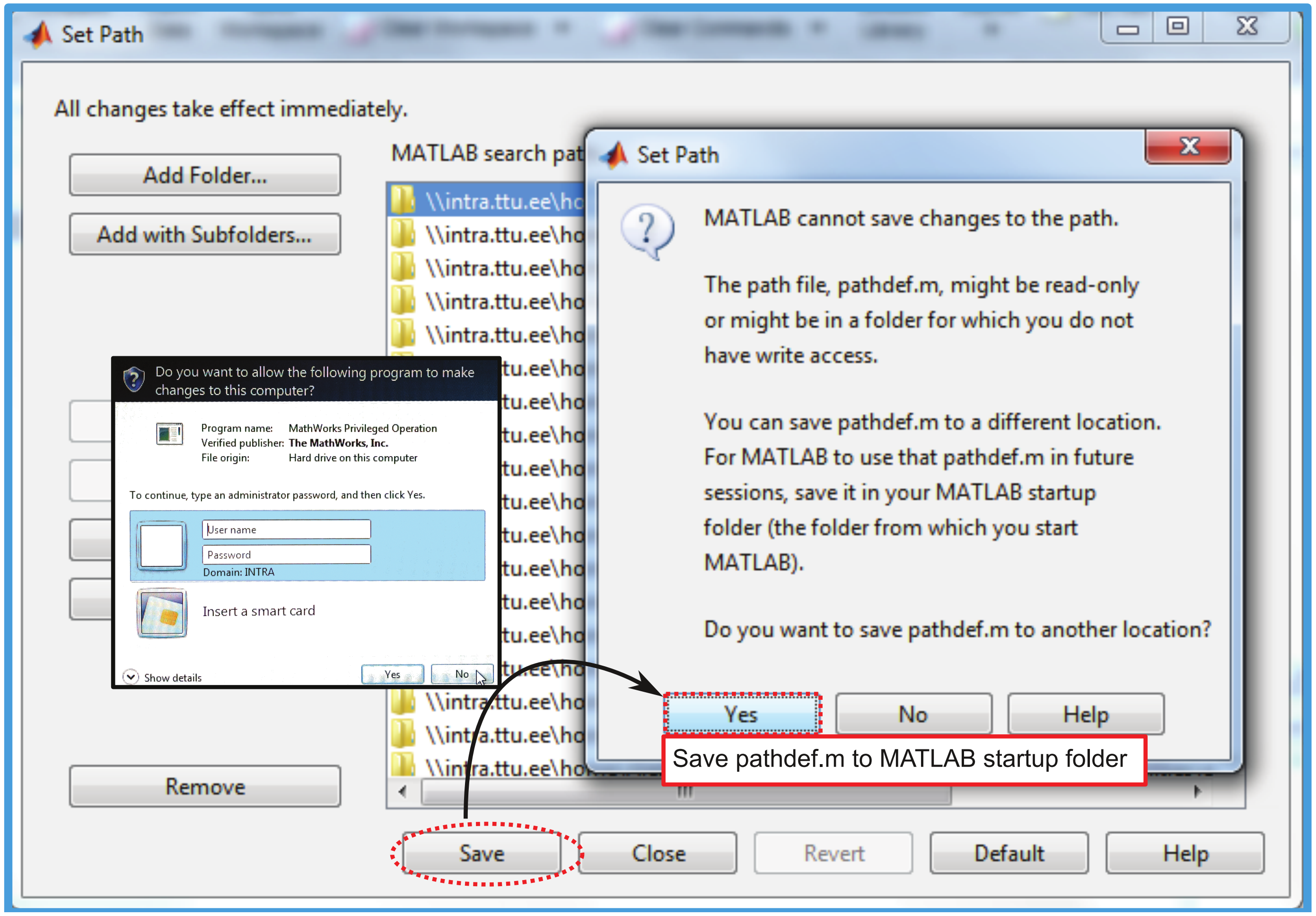
FOMCON toolbox: Installation (Download from `http://a-lab.ee/edu/iss0023`)


Make sure you are in the correct Working Directory

Right-click the folder with the toolbox files and add it to MATLAB path

After adding FOMCON to path, you need to save the path

FOMCON toolbox: Installation (continued)



MATLAB cannot save changes to the path.

The path file, pathdef.m, might be read-only or might be in a folder for which you do not have write access.

You can save pathdef.m to a different location. For MATLAB to use that pathdef.m in future sessions, save it in your MATLAB startup folder (the folder from which you start MATLAB).

Do you want to save pathdef.m to another location?

Do you want to allow the following program to make changes to this computer?

Program name: MathWorks Privileged Operation
Verified publisher: **The MathWorks, Inc.**
File origin: Hard drive on this computer

To continue, type an administrator password, and then click Yes.

Fractional-order models

A linear, fractional-order continuous-time dynamic system can be expressed by a fractional differential equation of the following form

$$\begin{aligned} a_n \mathscr{D}^{\alpha_n} y(t) + a_{n-1} \mathscr{D}^{\alpha_{n-1}} y(t) + \cdots + a_0 \mathscr{D}^{\alpha_0} y(t) &= \\ b_m \mathscr{D}^{\beta_m} u(t) + b_{m-1} \mathscr{D}^{\beta_{m-1}} u(t) + \cdots + b_0 \mathscr{D}^{\beta_0} u(t), \end{aligned} \tag{1}$$

where $a_k,\ b_k \in \mathbb{R}$. The system is said to be of *commensurate-order* if in (1) all the orders of derivation are integer multiples of a base order γ such that $\alpha_k,\ \beta_k = k\gamma$, $\gamma \in \mathbb{R}^+$. The system can then be expressed as

$$\sum_{k=0}^{n} a_k \mathscr{D}^{k\gamma} y(t) = \sum_{k=0}^{m} b_k \mathscr{D}^{k\gamma} u(t). \tag{2}$$

Fractional-order transfer functions

Applying the Laplace transform to (1) with zero initial conditions the input-output representation of the fractional-order system can be obtained in the form of a transfer function:

$$G(s) = \frac{Y(s)}{U(s)} = \frac{b_m s^{\beta_m} + b_{m-1} s^{\beta_{m-1}} + \cdots + b_0 s^{\beta_0}}{a_n s^{\alpha_n} + a_{n-1} s^{\alpha_{n-1}} + \cdots + a_0 s^{\alpha_0}}. \tag{3}$$

In the case of a system with commensurate order γ we have

$$G(s) = \frac{\sum\limits_{k=0}^{m} b_k \left(s^{\gamma}\right)^k}{\sum\limits_{k=0}^{n} a_k \left(s^{\gamma}\right)^k}. \tag{4}$$

FOMCON toolbox: FOTF Viewer

FOTF systems
in MATLAB workspace

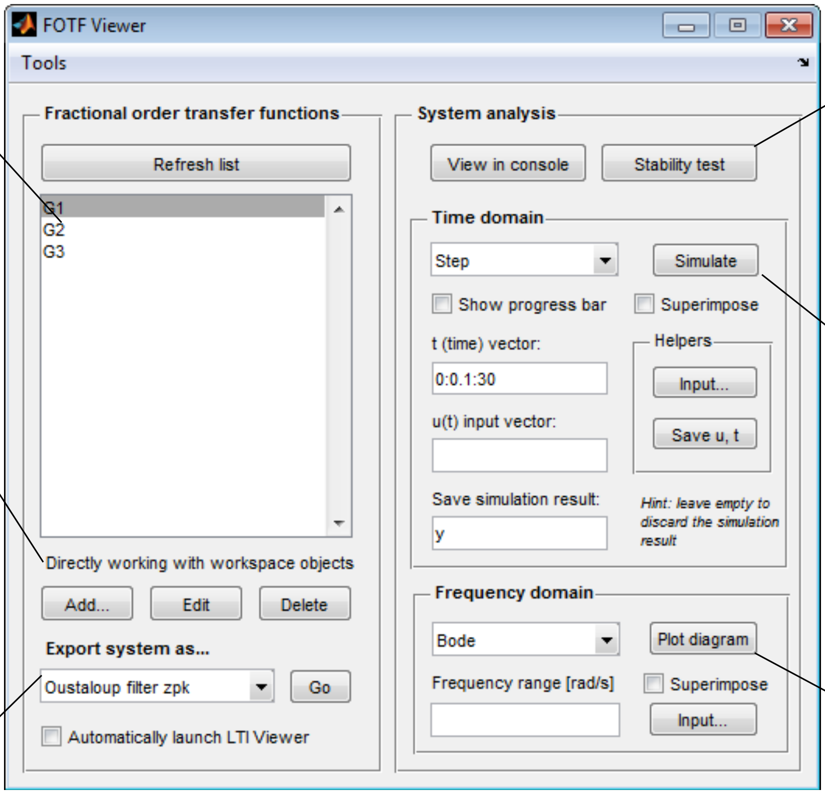
Add, Edit or Delete
FOTF systems

Export FOTF systems
to other formats

Stability test

Time-domain
analysis

Frequency-domain
analysis



Exercise: Analyze a fractional-order dynamic system

Using the *FOTF Viewer* tool, create a system

$$G(s) = \frac{5.7}{s^{0.95} - 2s^{0.5} + 3}$$

and—

- Check whether this system is stable.
- Obtain the step response of the system in the time ranges $t_1 = [0, 5]$ and $t_2 = [0, 50]$ with a time step of $\Delta t = 0.01$ in both cases. What is the static gain of this system?
- Obtain a Bode plot of this system. Right-click on it, choose "Characteristics$\rightarrow$Minimum stability margins", click the dot on the phase plot. What does the phase margin value mean?

Pseudoorder of a fractional system

By convention, we shall call the number of terms ν in the fractional pole polynomial $P(s)$, excluding the free term, the pseudoorder of the system

$$G(s) = Z(s)/P(s),$$

So for, for instance, the pseudoorder of the system

$$G(s) = \frac{s+1}{s - 2s^{0.5} + 5}$$

is $\nu = 2$. In this case, the pseudoorder coincides with the order of the pseudorational function of the commensurate-order system

$$H(\lambda) = \frac{\lambda^2 + 1}{\lambda^2 - 2\lambda + 5}.$$

The Control Problem to be Solved

You are given the task of controlling the output $y(t)$ of a certain system. The following is known:

- The system has an input control range of $u = [-1, 1]$ and an output range of $y = [0, 2\pi]$;
- There may be a deadzone in the control and/or an input-output delay;
- Two sets of collected experimental data are available to you.

Your goal is to identify a model describing this dynamic system and design a fractional-order controller based on this model. It is then required to evaluate the performance of the controller with the original system (this will be done in Simulink). Control system design specifications include:

- In the time domain: Settling time $\tau_s \leqslant 20$ s for $y_r = [0, 1]$ and overshoot $\vartheta \leqslant 5\%$, error band $y_e = \pm 5\%$ of the set point y_r;
- In the frequency domain: Gain margin $G_m \geqslant 15$ dB, phase margin $\varphi \geqslant 60^\circ$ and in addition, phase response at critical frequency ω_c must be as flat as possible to ensure robustness to gain variations.

Necessary lab materials: `http://a-lab.ee/edu/iss0023/#tab=Materials`

Time-domain identification: Output error minimization

Given the transfer function model

$$G(s) = \frac{b_m s^{\beta_m} + b_{m-1} s^{\beta_{m-1}} + \cdots + b_0 s^{\beta_0}}{a_n s^{\alpha_n} + a_{n-1} s^{\alpha_{n-1}} + \cdots + a_0 s^{\alpha_0}}$$

we search for a parameter set $\theta = [\ a_p \quad \alpha_p \quad b_z \quad \beta_z\ ]$, such that

$$\begin{array}{rcl} a_p & = & [\ a_n \quad a_{n-1} \quad \cdots \quad a_0\ ],\ \alpha_p = [\ \alpha_n \quad \alpha_{n-1} \quad \cdots \quad \alpha_0\ ], \\ b_z & = & [\ b_m \quad b_{m-1} \quad \cdots \quad b_0\ ],\ \beta_z = [\ \beta_n \quad \beta_{n-1} \quad \cdots \quad \beta_0\ ], \end{array}$$

by employing numerical optimization with an objective function given by an output error norm $\left\|e\left(t\right)\right\|_2^2$, where $e(t) = y(t) - \tilde{y}(t)$ is obtained by taking the difference of the original model output $y(t)$ and simulated model output $\tilde{y}(t)$.

Time-domain identification: Process models

Consider the following generalizations of conventional process models used in industrial control design.

(FO)FOPDT	$G(s) = \frac{K}{1+Ts}\mathrm{e}^{-Ls}$	$G(s) = \frac{K}{1+Ts^{\alpha}}\mathrm{e}^{-Ls}$
(FO)IPDT	$G(s) = \frac{K}{s}\mathrm{e}^{-Ls}$	$G(s) = \frac{K}{s^{\alpha}}\mathrm{e}^{-Ls}$
(FO)FOIPDT	$G(s) = \frac{K}{s(1+Ts)}\mathrm{e}^{-Ls}$	$G(s) = \frac{K}{s(1+Ts^{\alpha})}\mathrm{e}^{-Ls}$

Therefore, due to additional parameters K (gain) and L (delay) we may update the identified parameter set discussed previously to

$$\theta = [\ K \quad L \quad a_p \quad \alpha_p \quad b_z \quad \beta_z\ ].$$

Time-domain identification: Experimental data and model validation

The task of model identification has a very particular goal: to arrive at the best possible description of the system based on a set of measurements arising from external stimuli applied to the system (that is, control signals). So the process of identification has to comprise at least the following steps:

1. Plan the identification experiment with the real plant: Choose suitable input signal ranges, forms, amplitudes, as well as the duration or frequency of each control signal. If a nonlinear system is to be identified by a linear model, choose a particular working point and try to avoid nonlinearities (e.g., saturation). Remember: We want to gather as much useful information as possible.

   (a) Example input signals: Step, ramp, sine wave.

2. Choose a suitable initial guess model. In case of linear, time-invariant, fractional-order systems this means selecting the structure of the model, its pseudoorder and commensurate order (if applicable).
3. Carry out the identification process and validate the resulting model using a different experimental data set. If results are unsatisfactory, repeat Step 2, and if that fails, go back to Step 1.

Task: Loading the data sets and launching the identification tool

First, load the experimental data sets into MATLAB. Make sure you are in the working directory containing the lab files including `idsets.mat` and use the command

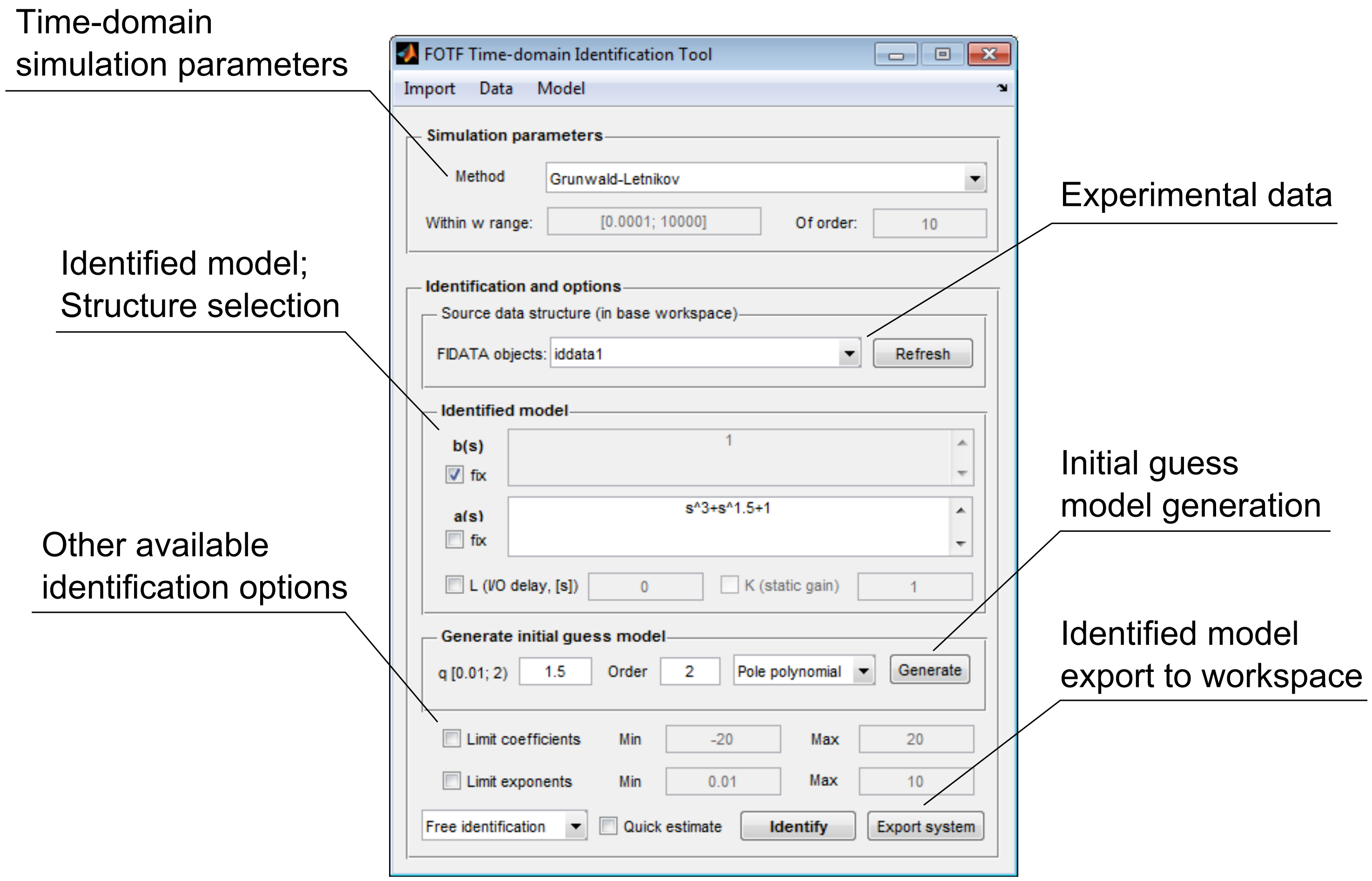
```
>> load idsets
```

You should see two new variables in the workspace named `proc_id_v1` and `proc_id_v2`. These are your experimental data sets. Now type

```
>> fotfid
```

The graphical user interface for the identification tool should launch. You can also start it from the menu bar of the *FOTF Viewer* application ("Tools→Identification→Time-domain...").

FOMCON toolbox: Time-domain identification


Time-domain simulation parameters
Identified model; Structure selection
Other available identification options
Experimental data
Initial guess model generation
Identified model export to workspace
FOTF Time-domain Identification Tool
Import
Data
Model
Refresh
Generate
Identify
Export system

Simulation parameters

Method: Grunwald-Letnikov

Within w range: [0.0001; 10000] Of order: 10

Identification and options

Source data structure (in base workspace)

FIDATA objects: iddata1

Identified model

b(s) [✓ fix]: 1

a(s) [ ] fix: s^3+s^1.5+1

[ ] L (I/O delay, [s]) 0 [ ] K (static gain) 1

Generate initial guess model

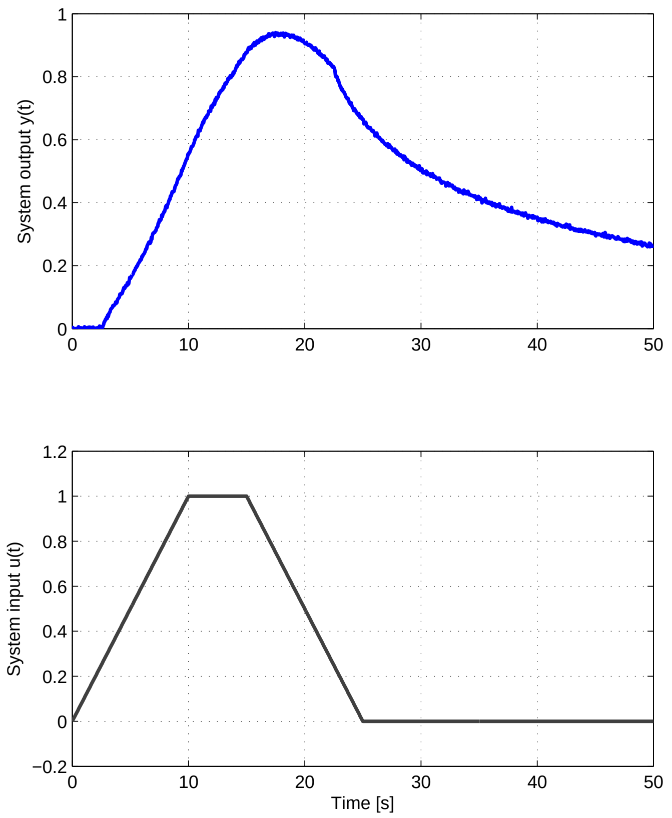
q [0.01; 2) 1.5 Order 2 Pole polynomial

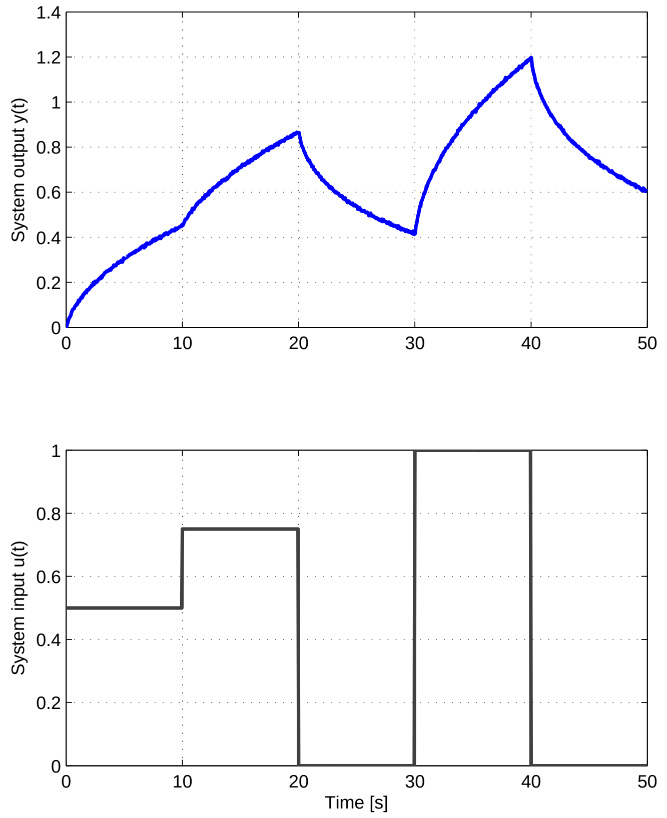
[ ] Limit coefficients Min -20 Max 20

[ ] Limit exponents Min 0.01 Max 10

Free identification [ ] Quick estimate

Task: The experimental data sets





`proc_id_v1` `proc_id_v2`

Task: Try multiple identification approaches

- To identify an integer-order conventional model, make sure that your initial guess model in the textboxes has only integer exponents, e.g.

$$b(s) = 1, \quad a(s) = s + 1,$$

  unlock both $b(s)$ and $a(s)$ and then choose the “Fix exponents” method from the drop-down list in the lower left corner of the identification tool.
- In order to identify a fractional-order process model, unlock both $b(s)$ and $a(s)$, select “Free identification” from the lower left corner drop-down list, and enable the static gain (K) parameter estimation.
- The suggested identification procedure is as follows: use the set `proc_id_v1` for identification, and `proc_id_v2` for validation.
- Can you improve the result of identification by transforming the `proc_id_v1` data set? Having selected it in *FIDATA objects* list choose “Data→Plot” from the menu bar. Notice the initial response $y(t)$ during $t = [0, 2.5]$. Can it be neglected for the purposes of the identification? Choose “Data→Trim” in the plot window, set $T_1 = 2.5$ and enter the name `proc_id_v1_t` for the new data set. Refresh the data set list, choose the new data set and repeat the identification process.

Task: Exporting the identified model to MATLAB workspace

Once you are satisfied with the result of the identification, click on the “Export system” button in the lower right corner of the identification tool GUI. Name the model `Gp`. In order to use this model later, you could save it to a MATLAB .mat file by typing in the command line

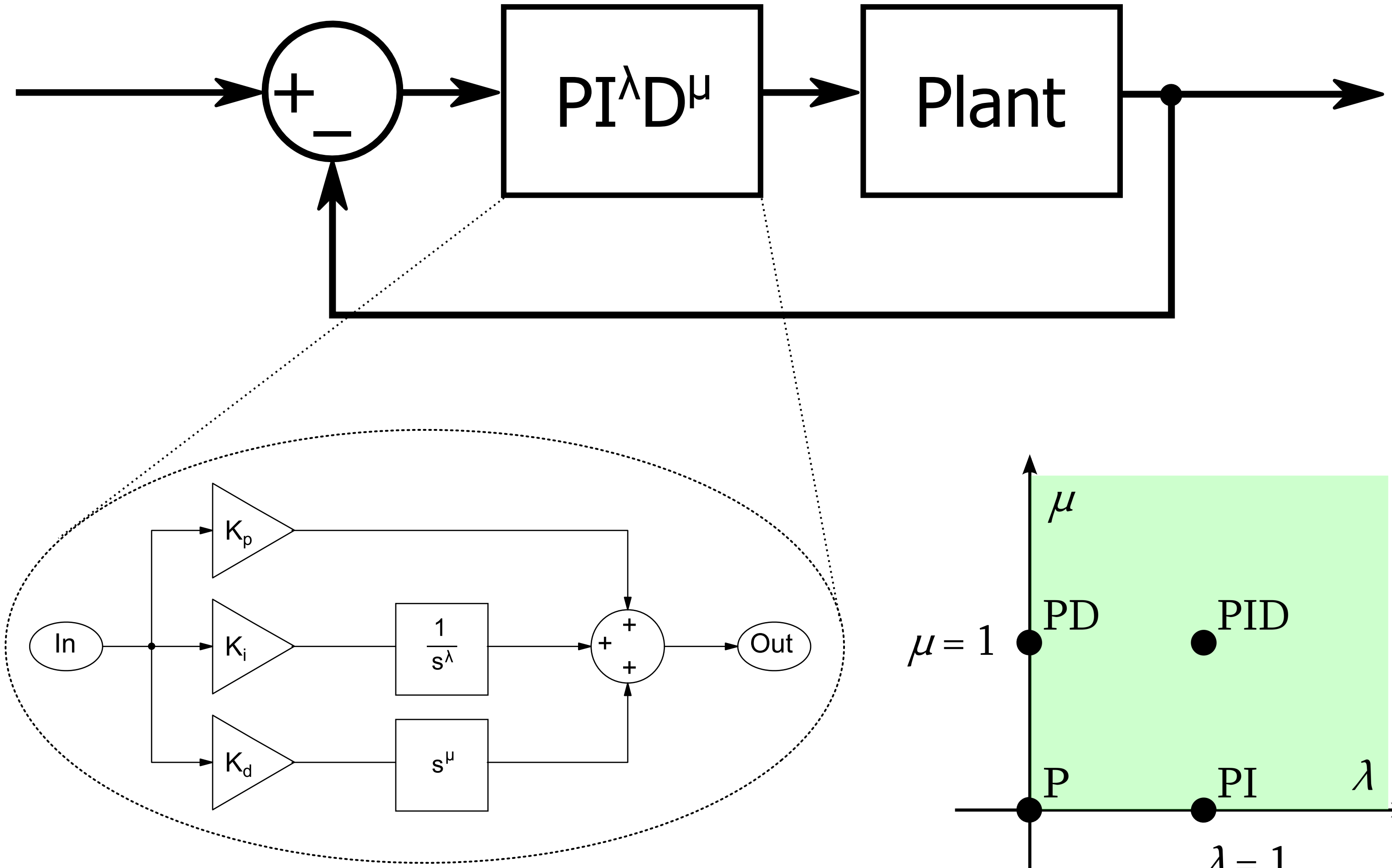
```
>> save mymodel.mat Gp
```

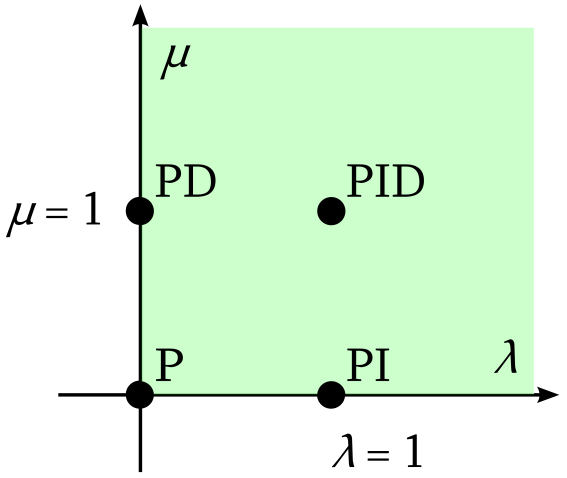
where `mymodel.mat` is the name of the file that you are saving the model to. To load this file later, type

```
>> load mymodel.mat
```

In what follows, we will work with this model to design a fractional PID controller for the plant.

Fractional-order Control: PI$^\lambda$D$^\mu$ control loop





Optimization based PI$^\lambda$D$^\mu$ tuning: Cost function

In case of a linear model we use time-domain simulation of a typical negative unity feedback loop

$$G_{cs}(s) = \frac{C(s)G(s)}{1 + C(s)G(s)}. \tag{5}$$

For the cost function we consider performance indicies:

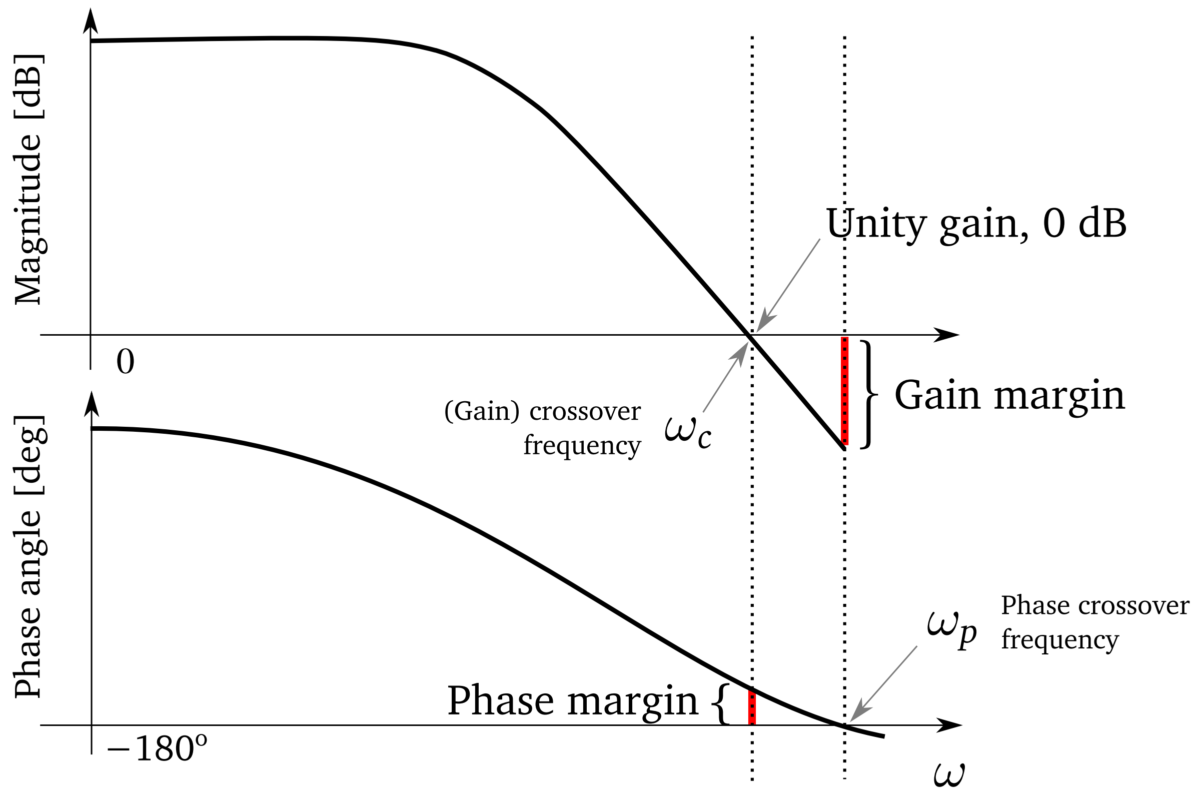
- integral square error $ISE = \int_0^\tau e^2(t)\mathrm{d}t$,
- integral absolute error $IAE = \int_0^\tau \left|e(t)\right| \mathrm{d}t$,
- integral time-square error $ITSE = \int_0^\tau te^2(t)\mathrm{d}t$,
- integral time-absolute error $ITAE = \int_0^\tau t\left|e(t)\right| \mathrm{d}t$.

Optimization based PI$^\lambda$D$^\mu$ tuning: Constraints

The design specifications include:

- Gain margin G_m and phase margin φ_m specifications;
- Complementary sensitivity function $T(j\omega)$ constraint, providing A dB of noise attenuation for frequencies $\omega > \omega_t$ rad/s;
- Sensitivity function $S(j\omega)$ constraint for output disturbance rejection, providing a sensitivity function of B dB for frequencies $\omega < \omega_s$ rad/s;
- Robustness to plant gain variations: a flat phase of the system is desired within a region of the system critical frequency ω_{cg};
- For practical reasons, a constraint on the control effort $u(t)$ may also be set.

Gain and phase margin specifications



Task: Getting to know the FOPID Optimization tool

You can launch the fractional-order PID optimization tool from MATLAB command line by typing

```
>> fpid_optim
```

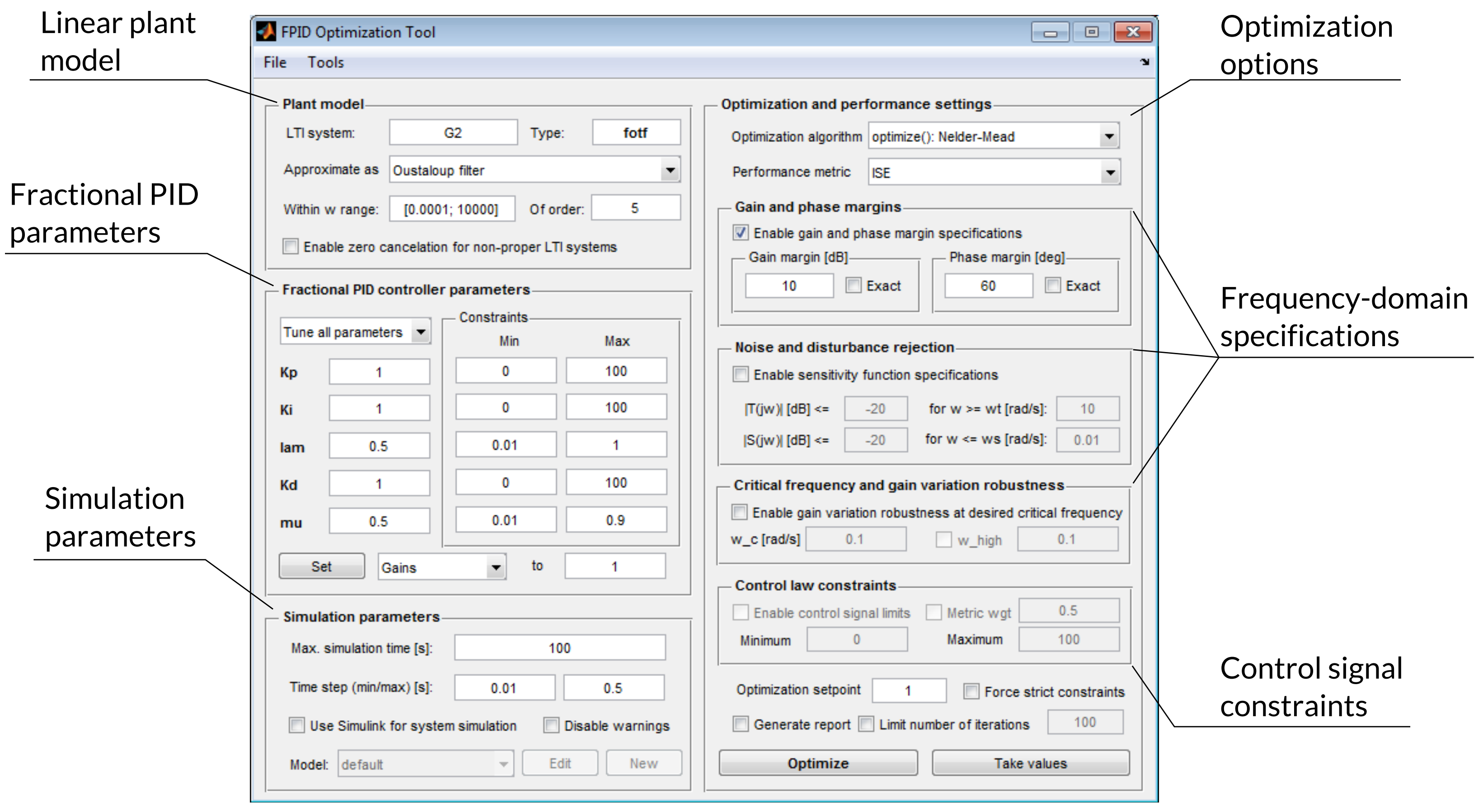
It can also be started from the menu bar of the *Fractional PID Design Tool* application ("Tuning→Optimize"), which, in turn, can be either launched from the *FOTF Viewer* ("Tools→Fractional PID Design") or by typing

```
>> fpid
```

in the command line.

NB! Make sure at this point that you are in the correct working directory. All laboratory work files should be present here.

FOMCON toolbox: Optimization based PI$^\lambda$D$^\mu$ tuning



Plant model

LTI system: G2 Type: fotf

Approximate as: Oustaloup filter

Within w range: [0.0001; 10000] Of order: 5

☐ Enable zero cancelation for non-proper LTI systems

Fractional PID controller parameters

Tune all parameters

		Constraints Min	Constraints Max
Kp	1	0	100
Ki	1	0	100
lam	0.5	0.01	1
Kd	1	0	100
mu	0.5	0.01	0.9

Set Gains to 1

Simulation parameters

Max. simulation time [s]: 100

Time step (min/max) [s]: 0.01 0.5

☐ Use Simulink for system simulation ☐ Disable warnings

Model: default Edit New

Optimization and performance settings

Optimization algorithm: optimize(): Nelder-Mead

Performance metric: ISE

Gain and phase margins

☑ Enable gain and phase margin specifications

Gain margin [dB]: 10 ☐ Exact Phase margin [deg]: 60 ☐ Exact

Noise and disturbance rejection

☐ Enable sensitivity function specifications

|T(jw)| [dB] <= -20 for w >= wt [rad/s]: 10

|S(jw)| [dB] <= -20 for w <= ws [rad/s]: 0.01

Critical frequency and gain variation robustness

☐ Enable gain variation robustness at desired critical frequency

w_c [rad/s] 0.1 ☐ w_high 0.1

Control law constraints

☐ Enable control signal limits ☐ Metric wgt 0.5

Minimum 0 Maximum 100

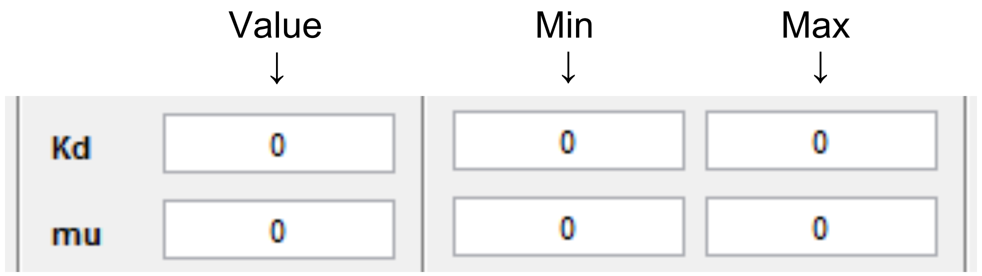
Optimization setpoint 1 ☐ Force strict constraints

☐ Generate report ☐ Limit number of iterations 100

Optimize Take values

Task: Obtain a PI$^\lambda$, PD$^\mu$, or PI$^\lambda$D$^\mu$ controller for the model

- Learn how to use the "File→Save/Load configuration" feature. This allows to save and recall all optimization options (including FOPID parameters) at once. One can think about this as saving/loading an optimization project. Among the items in the laboratory work archive there is a configuration file `initial_control.mat` in which basic initial optimization settings are stored. Load it up it by means of the "Load configuration" feature.
- To remove a component from optimization—be it I$^\lambda$ or D$^\mu$— set for that parameter such constraints, that $Value = Max = Min$. The component will still be used as part of the final controller.
- To remove a particular component of the fractional-order controller completely, set these constraints for both the component gain and order to zero, i.e. $Value = Max = Min = 0$. E.g., here is how to completely remove D^μ:

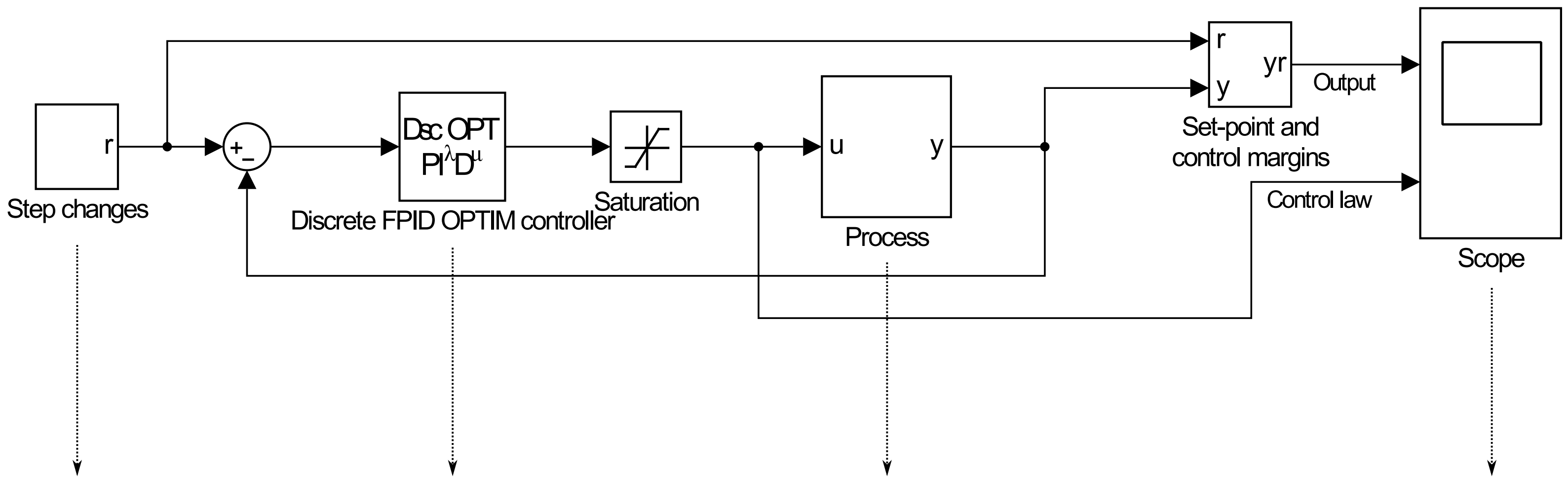


	Value	Min	Max
Kd	0	0	0
mu	0	0	0

Task: Obtain a PI$^\lambda$, PD$^\mu$, or PI$^\lambda$D$^\mu$ controller for the model (continued)

- First obtain a controller without regard to the "robustness to gain variations" specification. However, make sure that gain and phase margin specifications are enabled and set properly ($G_m \geqslant 15$ dB, $\varphi_m \geqslant 60°$).
- Once a suitable parameter set is found, see the open-loop Bode diagram generated automatically at the end of optimization (or if you have already closed the report window check the "Simulate only" checkbox at the bottom right and run Simulation without optimizing anything). Right-click anywhere in the diagram and choose "Characteristics→Minimum stability margins". Click the dot on the phase plot. Take note of the frequency ω_c you find there.
- Now enable the robustness to gain variations specification ("Enable critical frequency specification"), and uncheck the ω_{high} checkbox. Enter the ω_c value from the previous step into the textbox. Proceed with the optimization process.
- If now you get too high an overshoot, remember, that it is possible to control it using the phase margin φ_m specification.
- Once you have obtained a good result, save the corresponding configuration into a file `test_control.mat` in your working directory. Now open the model `lab_object_control.mdl` and run it in Simulink. Take note of the results, and if they are unsatisfactory, repeat the tuning process.

Controller tuning: Process control evaluation model in Simulink



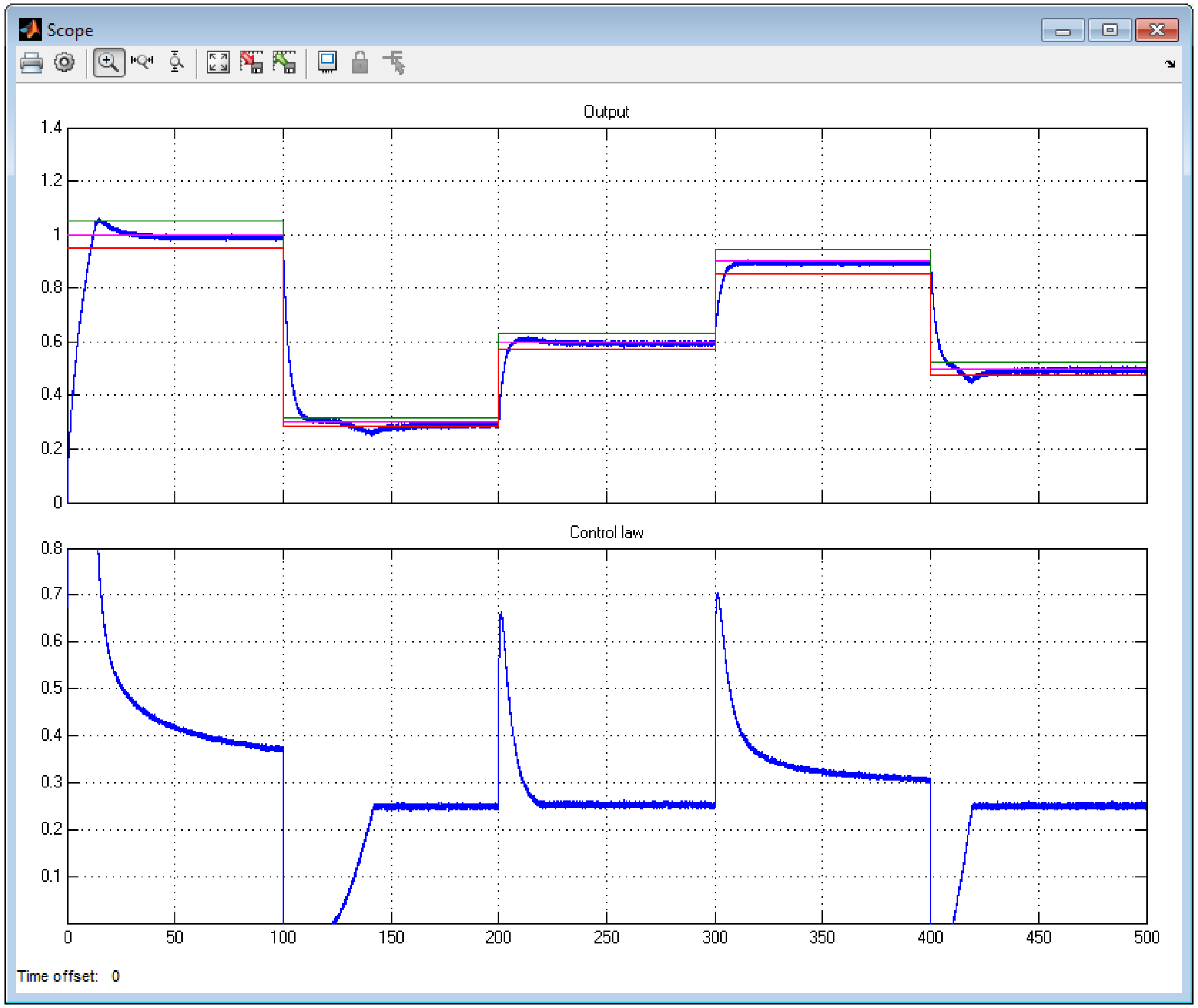
Sets process reference values

Loads the controller from "test_control.mat" configuration file

The simulated fractional process to be controlled

Observed process output $y(t)$, reference signal $y_r(t)$, output error band and control law $u(t)$

Controller tuning: Exemplary result


Scope
Output
Control law
Time offset: 0