

**COURSE Tool  
LIST OF COMMANDS**

**Systems available:**

system	initialization command (m-file)	Simulink model (*.mdl)
Simple Pendulum	pend	pendm
Double Pendulum	dpend	dpendm
Ball & beam	ball	ballm
Helicopter Azimuth	heli	helim

**Level 1 macros:**

active	Choose and activate a system.	<i>interactive</i>	
control	Choose interactively and run a design procedure	<i>interactive</i>	
simul	Choose interactively and open a Simulink scheme related to the active system.		<i>interactive</i>
setic	Modify initial conditions.		
ed	View and edit m-files related to the active model.		

**Level 2 macros:**

*Activating models (the following macros are called by the ACTIVE command – see above):*

pend	Activate the simple pendulum system.
dpend	Activate the double pendulum system.
ball	Activate the ball & beam system.
heli	Activate the helicopter azimuth system.

*Control Utilities (the following macros are called by the CONTROL command – see above):*

pp	Compute a pole-placement controller to the active model.
db	Compute a deadbeat controller to the active model.
lqg	Compute an LQG sampled to the active model.

*Simulation Utilities (the following macros are called by the SIMUL command – see above):*

model	Show the Simulink model of the active system.
regul	Show the Simulink closed-loop scheme (CL initial conditions response) related to the active system.
track	Show the Simulink scheme for reference tracking, related to the active system.
dist	Show the Simulink scheme for disturbance rejection, related to the active system.
noise	Show the Simulink scheme for noise rejection, related to the active system.

**Level 3 macros:**

*Control Utilities:*

pendpp	Compute a pole-placement controller.
penddb	Compute a deadbeat controller.
pendlqg	Compute an LQG sampled controller.

Accordingly:	dpendpp	ballpp	helipp
	dpenddb	balldb	helidb
	dpendlqg	balllqg	helilqg

*Simulation Utilities:*

pendm	Simulink model of the simple pendulum system (MDL file).
pendi	Simulink closed-loop scheme (CL initial conditions response) related to the simple pendulum system (MDL file).
pendt	Simulink scheme for reference tracking, related to the simple pendulum system (MDL file).
pendd	Simulink scheme for disturbance rejection, related to the simple pendulum system (MDL file).
pendn	Simulink scheme for noise rejection, related to the simple pendulum system (MDL file).

Accordingly:	dpendm	ballm	helim	
	dpendi	balli	helii	
	dpendt	ballt	helit	
	dpendd	balld	helid	
	dpendn	balln	helin	(All MDL files.)

## COURSE Tool USAGE

### 1. Activating a Model.

This step can be done either by calling the **active** command and selecting a model, or directly by running any of **pend**, **dpend**, **ball** or **heli** macros.

This step results in a set of variables that appears in the MATLAB workspace. Among them, also the polynomials **b,a** and **bz, az** can be found which are the numerator and denominator of the linearized, and linearized and sampled model respectively (**Ts** is used as the sampling period. If it does not exist in the MATLAB workspace, **Ts = 0.1s** is assigned).

To view or edit the activating command (eg. changing the parameters), call the **ed** macro.

In the sequel, let us suppose the simple pendulum has been activated by the **pend** command

### 2. Open-loop Simulation (optional)

Open the Simulink model of the active system (simple pendulum in our case). This can be done either

- by calling **simul** and selecting "1" (valid for any system active)
- by calling **model** (valid for any system active)
- by calling **pendm** (**dpendm, ballm** and **helim** for the other systems)

### 3. Controller Design

A controller to the given plant can be computed either "manually" by any method you may wish to use, or using the scripts for pole-placement, deadbeat and LQG control that are a part of this tool.

#### a) *Designing a controller yourself:*

As a (linear) model of the system, the polynomials **a,b** and **az,bz** are available that define the linearized model  $P(s) = b(s)/a(s)$ , and linearized sampled model  $Pz(z) = bz(z)/az(z)$  respectively. Linearization around the state of interest is performed by the activating command.

In order to keep the simulation tools functional, save the controller RMF numerator in the variable **q** and denominator in **p** (in case of the two-degree-of-freedom structure, **q/p** defines the feedback and **r/p** the feedforward part of the controller).

#### b) *Calling an existing routine:*

- call **control** and select the design procedure (valid for any system active)
- call **pp, db** or **lqq** to design a pole-placement, deadbeat or LQG controller (valid for any system active)
- call **pendpp, penddb** or **pendlqq** to design a pole-placement, deadbeat or LQG controller (**dpendpp, ballpp, helipp**, etc., for the other systems)

In any case, the controller RMF numerator and denominator are assigned in the **q** and **p** variables respectively (or **p,q,r** in the TDOF case)

To view or change the parameters of the built-in scripts, use the **ed** command (**ed pp, ed db, ed lqq**).

### 4. Simulation

The tool contains Simulink schemes of various control structures.

They can be achieved:

- by calling the **simul** command and selecting appropriate scheme
- by calling **model, regul, track, dist, noise** (see related helps)
- by calling **pendm, pendi, pendt, pendd, pendn** (**dpendm, ballm, helim**, etc., for the other systems)

Of course, any other structure can be added. Just be aware that the controller RMF numerator and denominator are stored in the **q** and **p** (or **q,r,p**) variables in the MATLAB workspace if you have employed any of the built-in design procedures in step 3.