Modelling and control summaries

The University Of Sheffield.

by Anthony Rossiter

USE OF MATLAB 5 – step responses

OVERVIEW: These notes gives a very narrow view of MATLAB and how to do a limited number of things. In general students need to become effective independent learners of MATLAB.

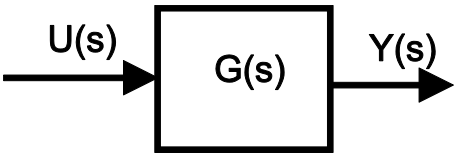
ASSUMPTIONS

1. Readers are familiar with how to enter a transfer function object into MATLAB.
2. Readers understand that this object could represent either a signal or a transfer function depending upon how it is used.
3. This note details how to determine numerical answers rather than algebraic ones so there is no formal discussion of inverse Laplace methods.

SCENARIO

For the block diagram given here we wish to determine the output signal y(t) assuming ZERO initial conditions. The input signal U(s) can be either:

1. A unit impulse [So U(s)=1]
2. A unit step. [So U(s)=1/s]

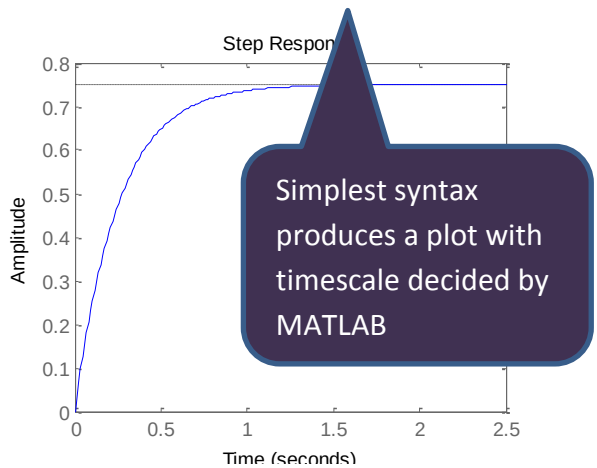


REMARKS

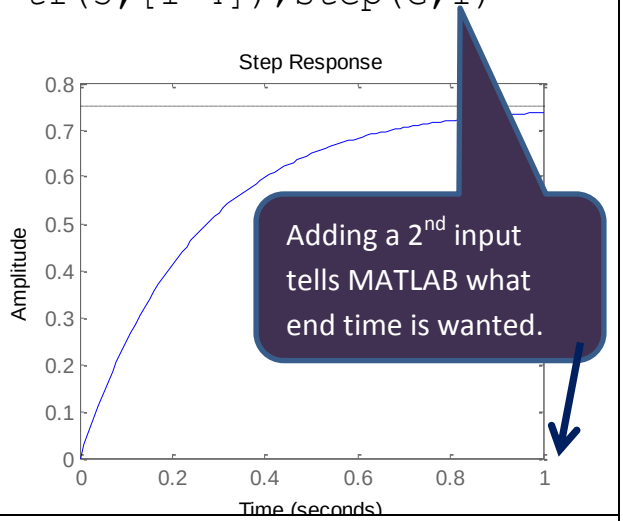
- **The impulse function does a direct inverse Laplace on G(s) and thus is in effect equivalent to doing the inverse Laplace of a signal represented by G(s).**
- **The step function does $L^{-1}[G(s)/s]$ so is equivalent to the output response of a transfer function G(s) subject to a unit step input – usually denoted the step response.**

BASIC USES OF THE COMMAND STEP

```
G=tf(3,[1 4]);step(G)
```




```
G=tf(3,[1 4]);step(G,1)
```



When no output arguments are included in the call statement, the MATLAB **step** function produces a plot in the active axis. The end time can be controlled by the USER if desirable.

Collecting output values from step

If the USER gives output arguments, then no plot will be given and rather data will be produced in the workspace. The USER must use the plot command directly with this data if they want a plot. A number of different examples are given below.

y = step(G);	Matlab selects end time and spacing. Output values in y.
[y,t] = step(G);	Matlab selects end time and spacing. Output values in y and times in t.
y = step(G,t);	USER sets the end time but Matlab selects spacing.
[y,tt] = step(G,t);	Matlab selects spacing, USER sets end time. Outputs in y and times in tt.
tspace =linspace(a,b,c);	USER defines a set of **increasing** and **equi-spaced** time points
y = step(G,tspace);	Output values with correspond to times in tspace.
[y,tt] = step(G,tspace);	as above but tt=tspace so in fact redundant.

READERS should try some of these commands for themselves to explore the functionality. You can use command 'whos' to look at the size of the outputs.

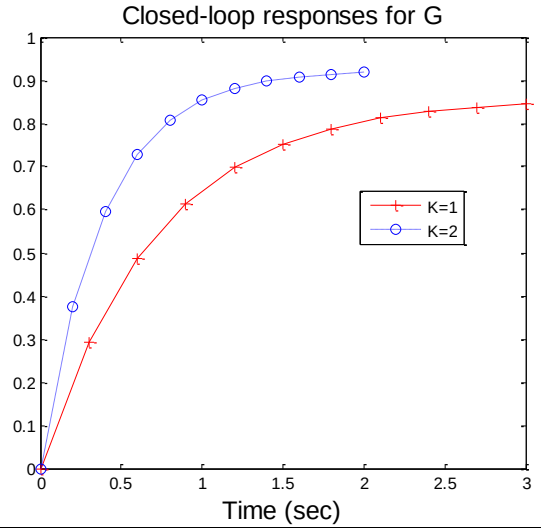
PLOTTING AND OVERLAYING STEP RESPONSES

Although step without output arguments will overlay responses from several systems, often it is better to export values to the workspace and use plot directly as this allows better control of the eventual plot. An example is given here where a short script file has been written to contain the key commands (hence this can be saved and re-used).

1. Note use of different line styles for different choices of feedback gain.
2. Note increase in font sizes for the labels.
3. Note precise control of the desired legend.
4. Note different end times and time spacings for each step response plot.

SCRIPT FILE

```
G=tf(1.2,[1 0.2]);
Gc1 = feedback(G*1,1);
Gc2 = feedback(G*2,1);
[y1,t1] = step(Gc1,0:.3:3);
[y2,t2]=step(Gc2,0:.2:2);
plot(t1,y1,'r-+',t2,y2,'bo:');
legend('K=1','K=2');
title('Closed-loop responses for
G','fontsize',15)
xlabel('Time sec)','fontsize',15)
```



impulse.m

This file uses the same syntax as step and thus instructions are not given.