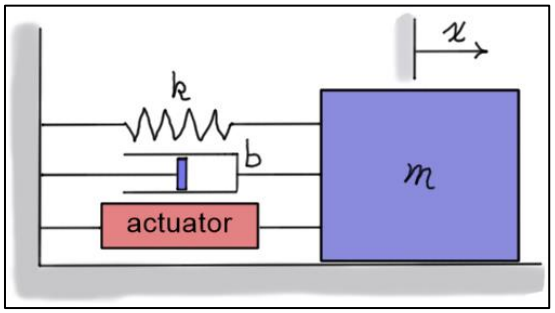
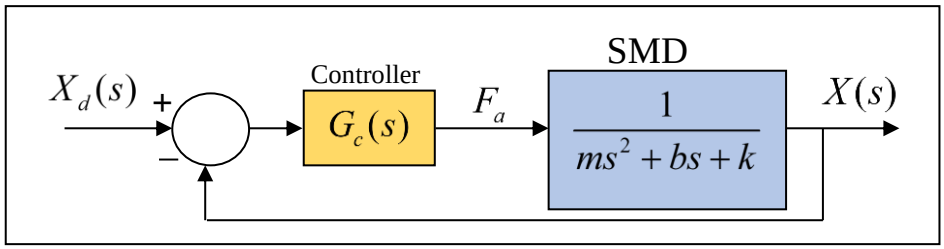
Introductory Control Systems
Using MATLAB to Study Closed-Loop Step Responses

As an example of how to use MATLAB to perform a ***unit step response***, consider again ***position control*** of a spring-mass-damper (SMD) system using ***proportional control***. The figure shows a ***mass*** m with a ***force actuator***, a ***spring*** of stiffness k, and a ***damper*** with coefficient b. The variable x measures the position of the mass.



Assuming ***ideal actuator*** and ***sensor*** responses, the closed-loop ***position control*** of the SMD can be described using the following ***block diagram***. Here, X_d represents the ***desired position***, X represents the ***actual position***, and $G_c(s)$ represents the ***transfer function*** of the ***controller***.



Using ***block diagram reduction***, the transfer function for proportional control with $G_c(s)=K$ is

$$\boxed{\frac{X}{X_d}(s)=\frac{K}{ms^2+bs+(k+K)}}$$

To study the response of this system for ***various*** system and control parameters using MATLAB, consider the MATLAB script shown in the box below. In the first block of code, the physical parameters of the system are defined. Using these parameters, the characteristic equation of the SMD can be written as follows.

$$\boxed{s^2+8.8s+40=(s+4.4)^2+\left(40-4.4^2\right)=(s+4.4)^2+20.64\triangleq(s+a)^2+\omega^2}$$

From the Laplace transform tables (see item 16), this is an ***under-damped system***, and the ***frequency*** of its free response is

$$\boxed{\omega=\sqrt{20.64}\approx 4.543\ \text{(rad/s)}\approx 0.723\ \text{(cycles/sec)}}$$

The ***characteristic equation*** of the ***closed-loop system*** has the same coefficients as the open-loop system for the first two terms, but not for the third. For $K>0$, the ***third coefficient*** in the

characteristic equation is ***increased***. So, for small gains K (relative to the spring stiffness k), the closed-loop system will oscillate ***below*** 1 cycle/sec, and as the ***gain*** is ***increased*** its ***frequency*** of oscillation will ***increase***.

MATLAB Script

```
% Example: Proportional control of SMD position
%
% Define physical parameters of the SMD
   m = 1;
   b = 8.8;
   k = 40;

% Define a time vector from 0 to 2 seconds in steps of 0.01 seconds
   t = [0:0.01:2];

% Define figure 1, clear it, and specify "hold on" to put all plots on the
% same graph
   figure(1); clf; hold on;

% Define the closed-loop system transfer function for K = 100 and plot the
% step response (in blue) over the time vector
   K = 100;
   num = [K];
   den = [m, b, (k+K)];
   sys = tf(num,den)
   step(sys,'b',t);

% Define the closed-loop system transfer function for K = 500 and plot the
% step response
   K=500;
   num = [K];
   den = [m, b, (k+K)];
   sys = tf(num,den)
   step(sys,'g',t)

% Define the closed-loop system transfer function for K = 2000 and plot the
% step response
   K=2000;
   num = [K];
   den = [m, b, (k+K)];
   sys = tf(num,den)
   step(sys,'r',t)

% Turn the grid feature "on" and place a "title", "y-axis label", and
% legend on the graph
   grid on;
   title('Unit Step Position Command Response');
   ylabel('Position (ft)');
   legend('K = 100','K = 500','K = 2000','Location','southeast')
```

It is shown in later notes that the ***settling time*** of the system can be ***estimated*** as follows.

$$T_s \approx \frac{4}{a} = \frac{4}{4.4} \approx 0.909 \text{ (sec)}$$

Note that this value is the ***same*** for both the open-loop and closed-loop systems.

These values can be used to define a ***time vector*** (`t`) for the response. The time vector in the script is defined to span from 0 to 2 seconds with an increment of 0.01 seconds. If the ***range*** is too small, the plots will ***terminate prematurely***, and if the ***increment*** is ***too large***, the plots will ***not*** be ***smooth***. If the time vector is not specified, MATLAB will choose these values, and they may or may not provide a proper view of the actual response.

The next section of the script (`figure(1); clf; hold on;`) opens figure window 1, clears it (in preparation for the plot (or plots) that follow), and turns on the "`hold`" feature. Setting "`hold on`" allows multiple plots to be placed on the same graph. In this example, three plots are shown on the same graph. The plots show the system responses for gains of 100, 500, and 2000.

The next section defines the ***transfer function*** and ***plots*** the ***step response*** for $K = 100$. The ***numerator*** (`num`) and ***denominator*** (`den`) of the ***transfer functions*** are defined using ***row vectors*** that contain the coefficients of their respective polynomials. The function "`tf(num,den)`" uses those numerator and denominator polynomials to create the transfer function of the system (called "`sys`" in the script). Finally, the "`step(sys,'b',t)`" function uses the transfer function ("`sys`") and the time vector ("`t`") to calculate and plot the step response. The second argument of the function defines the color of the plotted response. For $K = 100$ the plot will be blue (`'b'`).

The next two sections of the script ***calculate*** and ***plot*** the step responses for $K = 500$ and $K = 2000$. Their plots are green and red, respectively. The last section of the script places a ***grid*** (`grid on`), title, y-axis label, and plot legend on the graph. See MATLAB figure below.