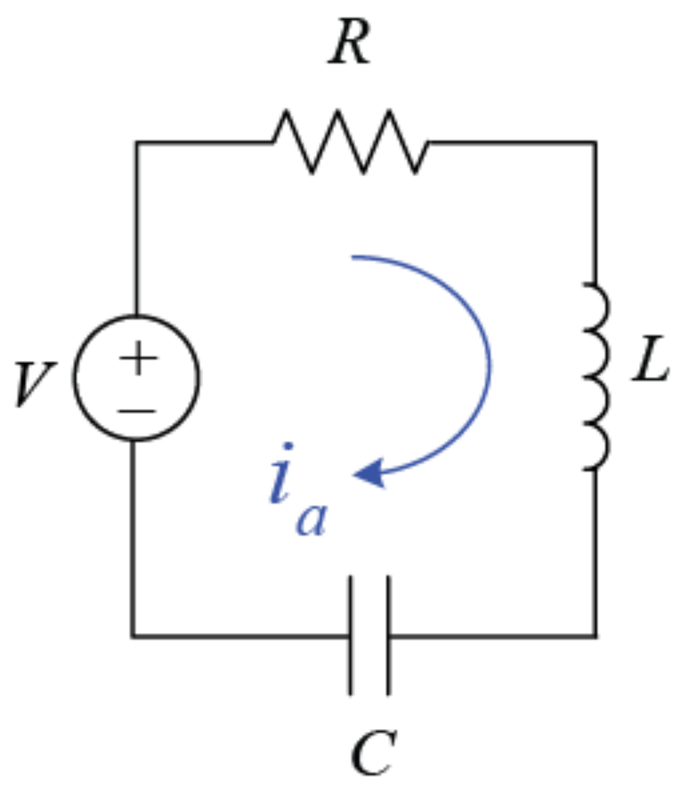
SYSTEM MODELING

- The first step in the control design process is to develop appropriate mathematical models of the system to be controlled. These models may be derived either from physical laws or experimental data. In this presentation, we introduce the transfer function representations

EXAMPLE: RLC CIRCUIT

- We will now consider a simple series combination of three passive electrical elements: a resistor, an inductor, and a capacitor, known as an **RLC Circuit.**



- Since this circuit is a single loop, each node only has one input and one output; therefore, application of KCL simply shows that the current is the same throughout the circuit at any given time, i(t).

EXAMPLE: RLC CIRCUIT

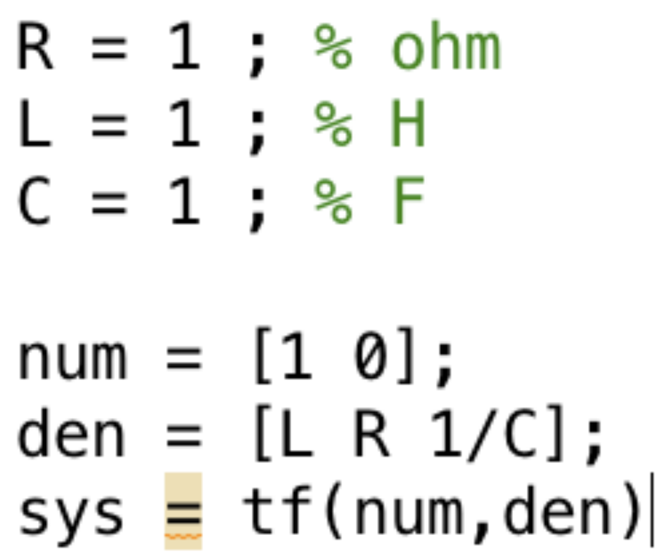
- Now applying KVL around the loop and using the sign conventions indicated in the diagram, we arrive at the following **governing equation**.

$$V(t) - Ri - L\frac{di}{dt} - \frac{1}{C}\int idt = 0$$

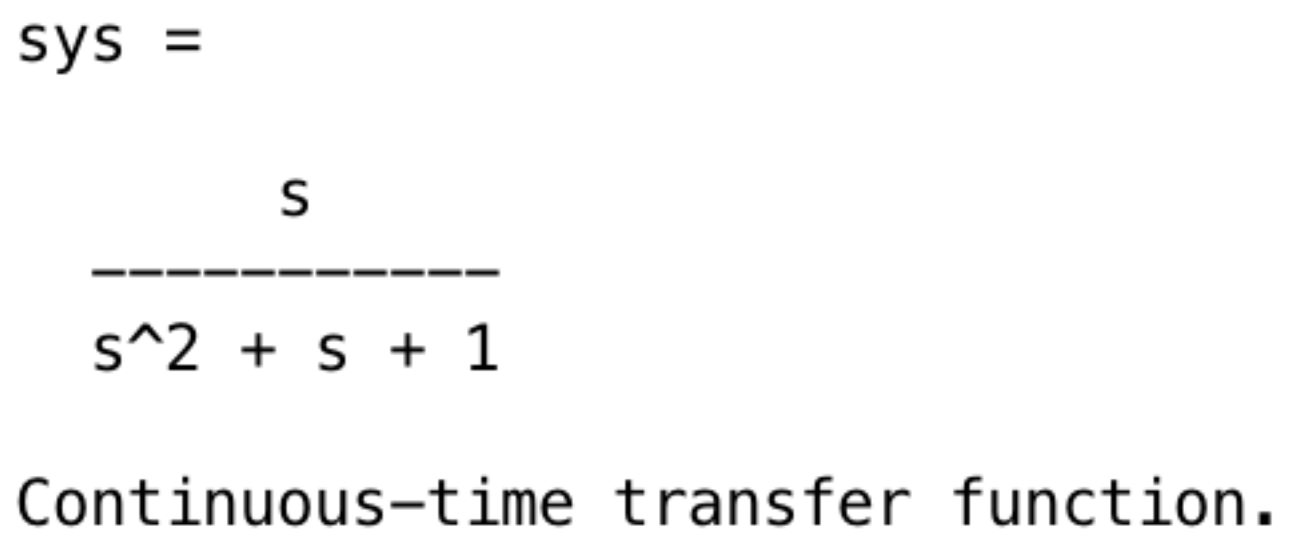
- The Laplace transform for this system assuming zero initial conditions is:

$$\frac{I(s)}{V(s)} = \frac{s}{Ls^2 + Rs + \frac{1}{C}}$$

EXAMPLE: RLC CIRCUIT

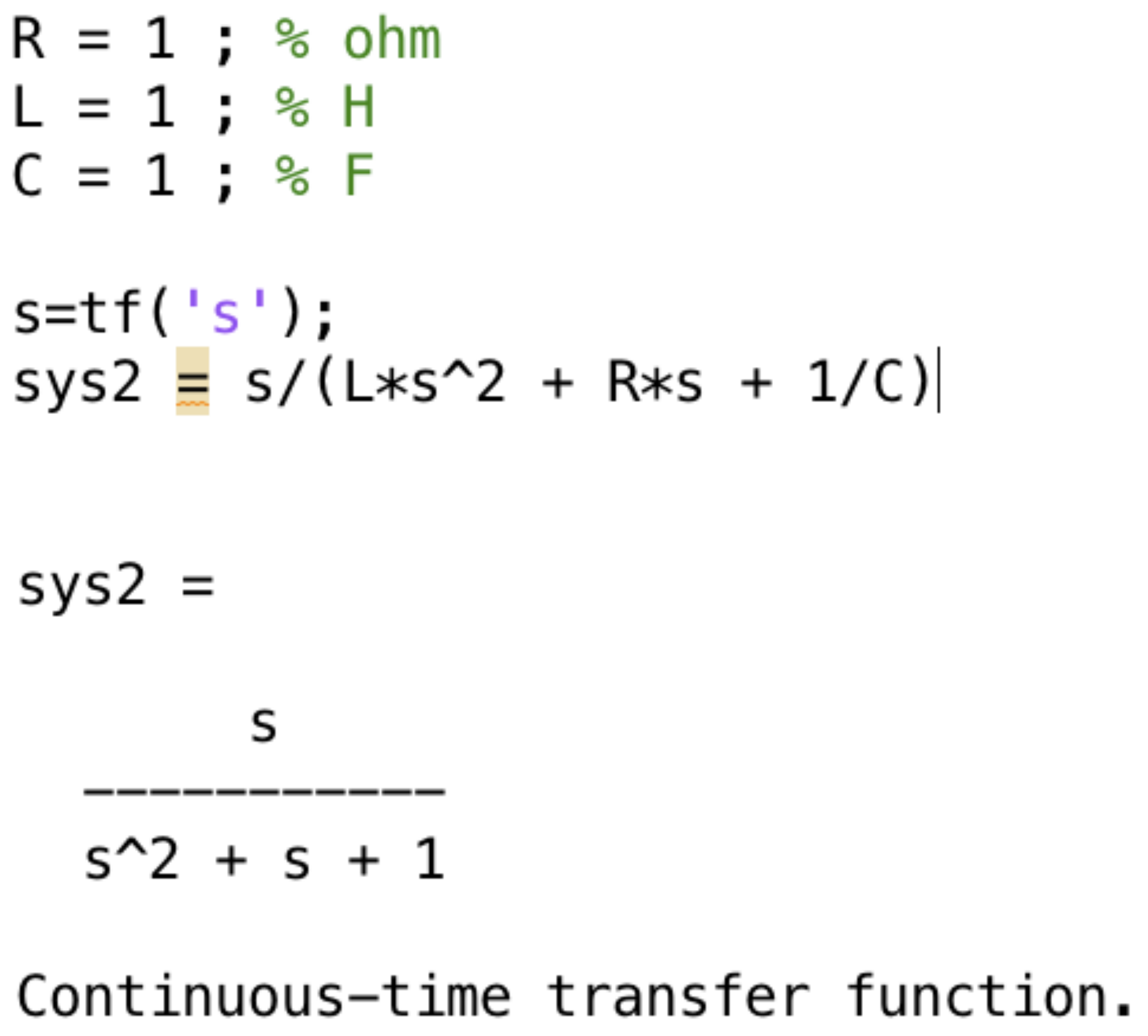


```
R = 1 ; % ohm
L = 1 ; % H
C = 1 ; % F

num = [1 0];
den = [L R 1/C];
sys = tf(num,den)
```



```
sys =

         s
  -----------
  s^2 + s + 1

Continuous-time transfer function.
```

EXAMPLE: RLC CIRCUIT

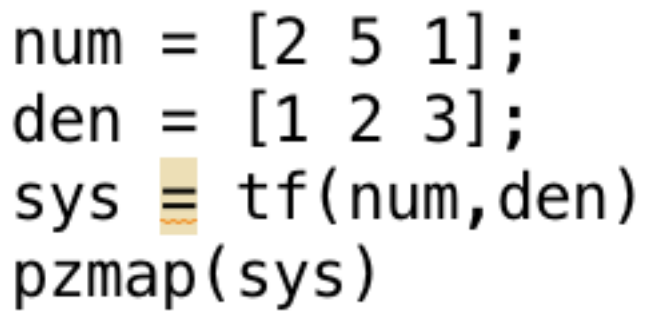


```
R = 1 ; % ohm
L = 1 ; % H
C = 1 ; % F

s=tf('s');
sys2 = s/(L*s^2 + R*s + 1/C)
```

```
sys2 =
 
        s
   -----------
   s^2 + s + 1
 
Continuous-time transfer function.
```

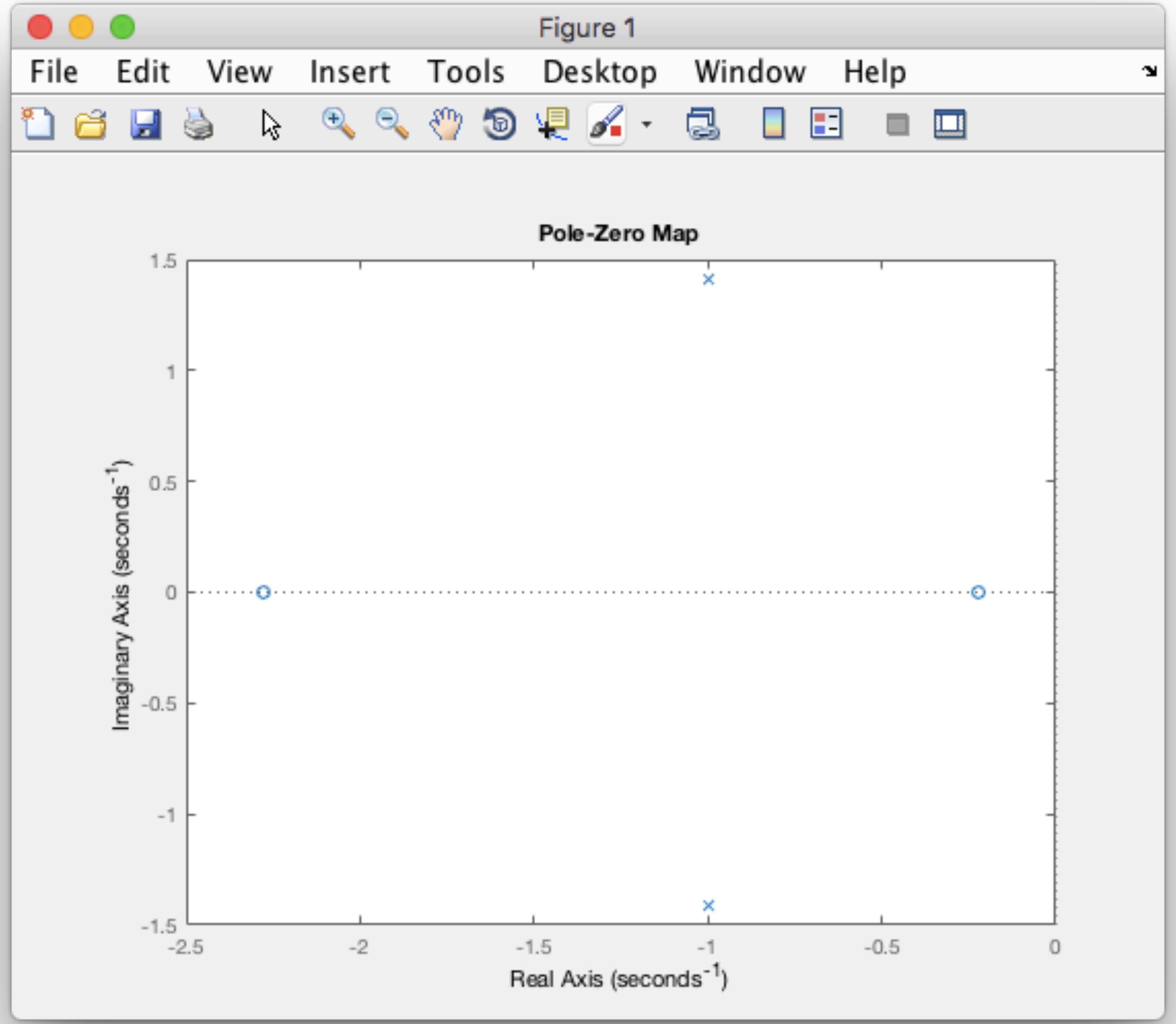
PLOTTING POLES AND ZEROS OF A SYSTEM

- **pzmap(sys)** : Compute pole-zero map of LTI models, plots the pole-zero map of the continuous or discrete-time LTI model sys.



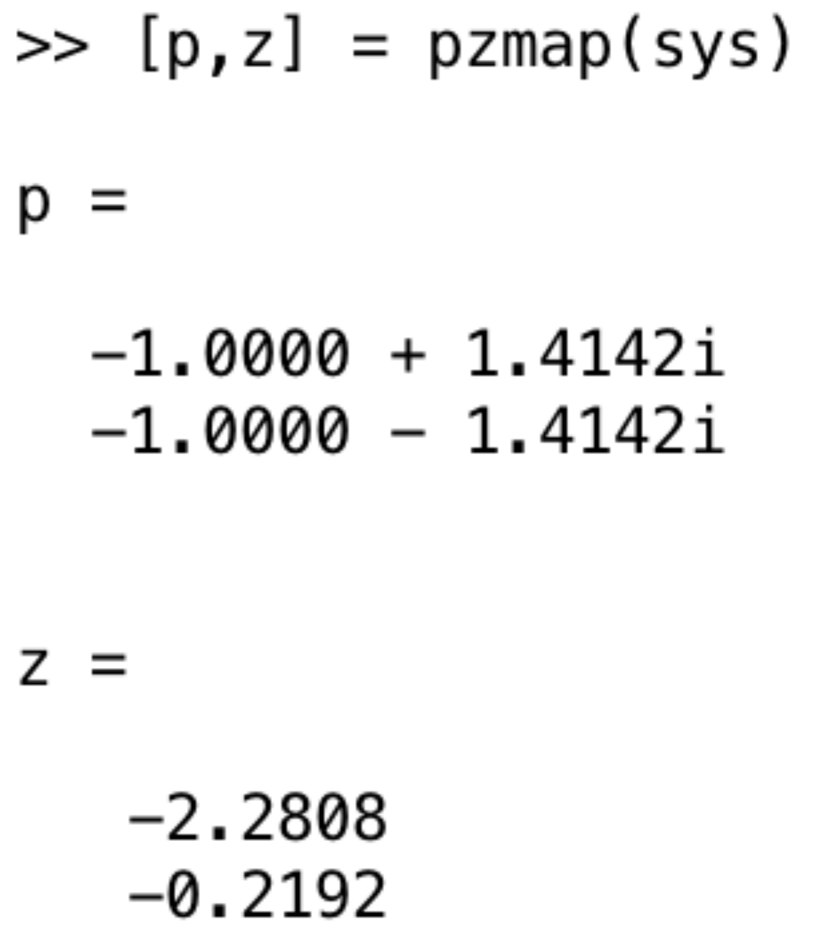
```
num = [2 5 1];
den = [1 2 3];
sys = tf(num,den)
pzmap(sys)
```

PLOTTING POLES AND ZEROS OF A SYSTEM



PLOTTING POLES AND ZEROS OF A SYSTEM

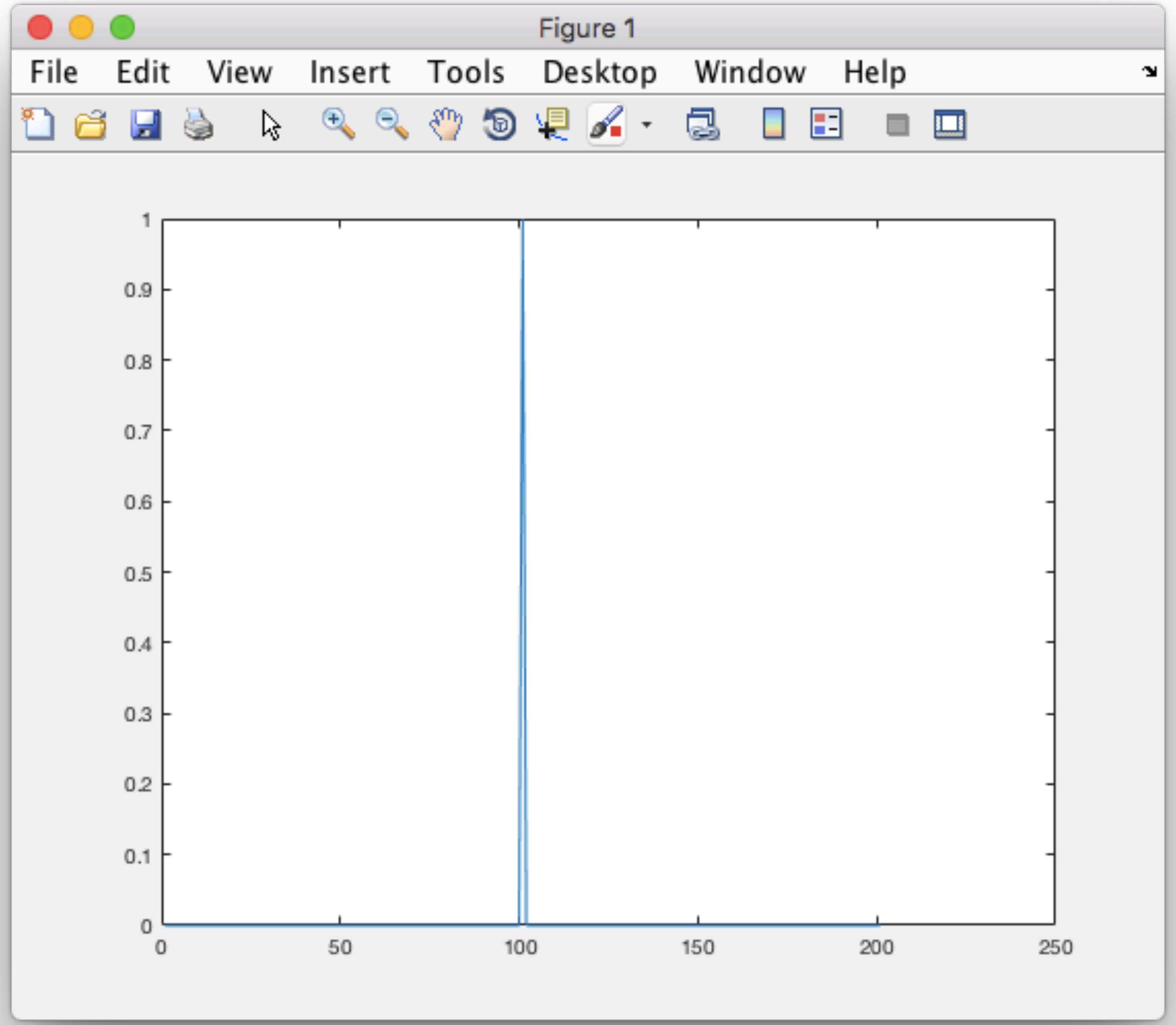
- **[p,z] = pzmap(sys)** : Returns the system poles and zeros in the column vectors p and z. No plot is drawn on the screen.



```
>> [p,z] = pzmap(sys)

p =

  -1.0000 + 1.4142i
  -1.0000 - 1.4142i


z =

   -2.2808
   -0.2192
```

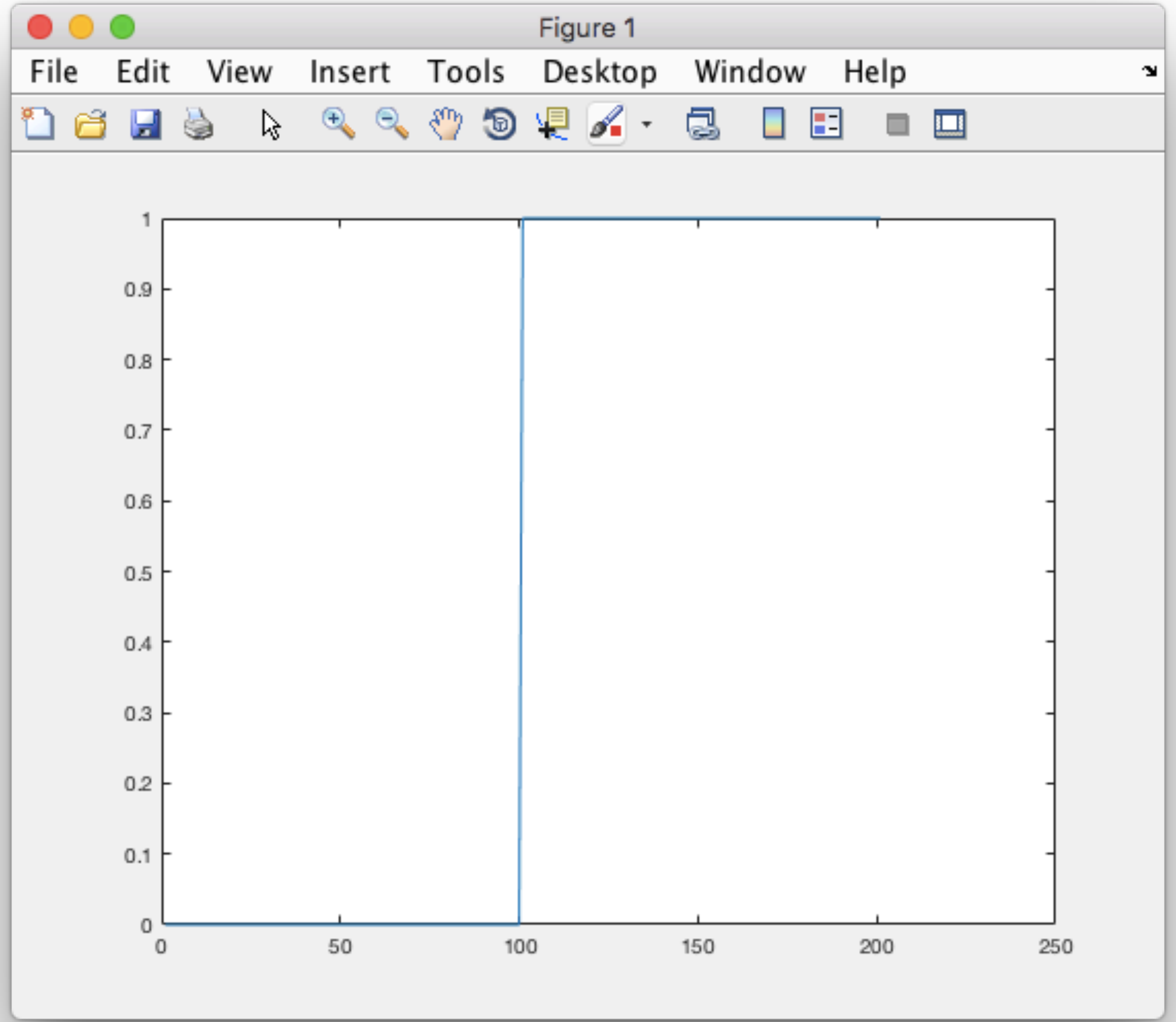
IMPULSE - STEP - RAMP

```
t = (-1:0.01:1)';

impulse = t==0;
figure
plot(impulse)
```



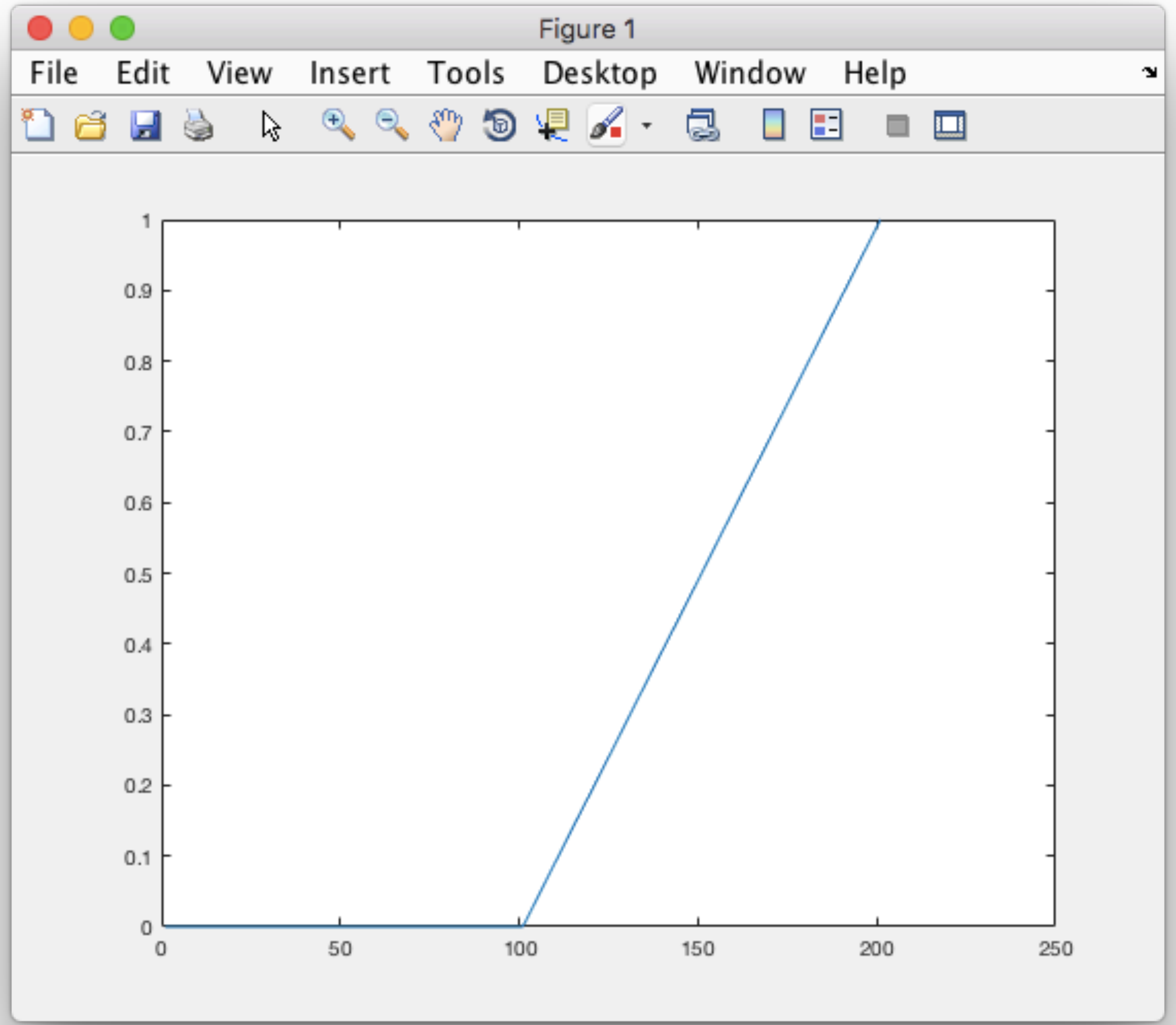
IMPULSE - STEP - RAMP

```
unitstep = t>=0;
figure
plot(unitstep)
```



IMPULSE - STEP - RAMP

```
ramp = t.*unitstep;
figure
plot(ramp)
```



IMPULSE - STEP - RAMP

To obtain an impulse response:

```
num = [2 5 1];
den = [1 2 3];
sys = tf(num,den);
figure;
impulse(sys)
```

Time-interval specification:

To contain the response of the system you can also specify the time interval to simulate the system to. For example:

```
figure;
t = 0:0.01:10;
figure;
impulse(sys,t)
```

IMPULSE - STEP - RAMP

To obtain a step response:

```
figure;
step(sys)
```

Again you can set the interval:

```
figure;
step(sys,t)
```

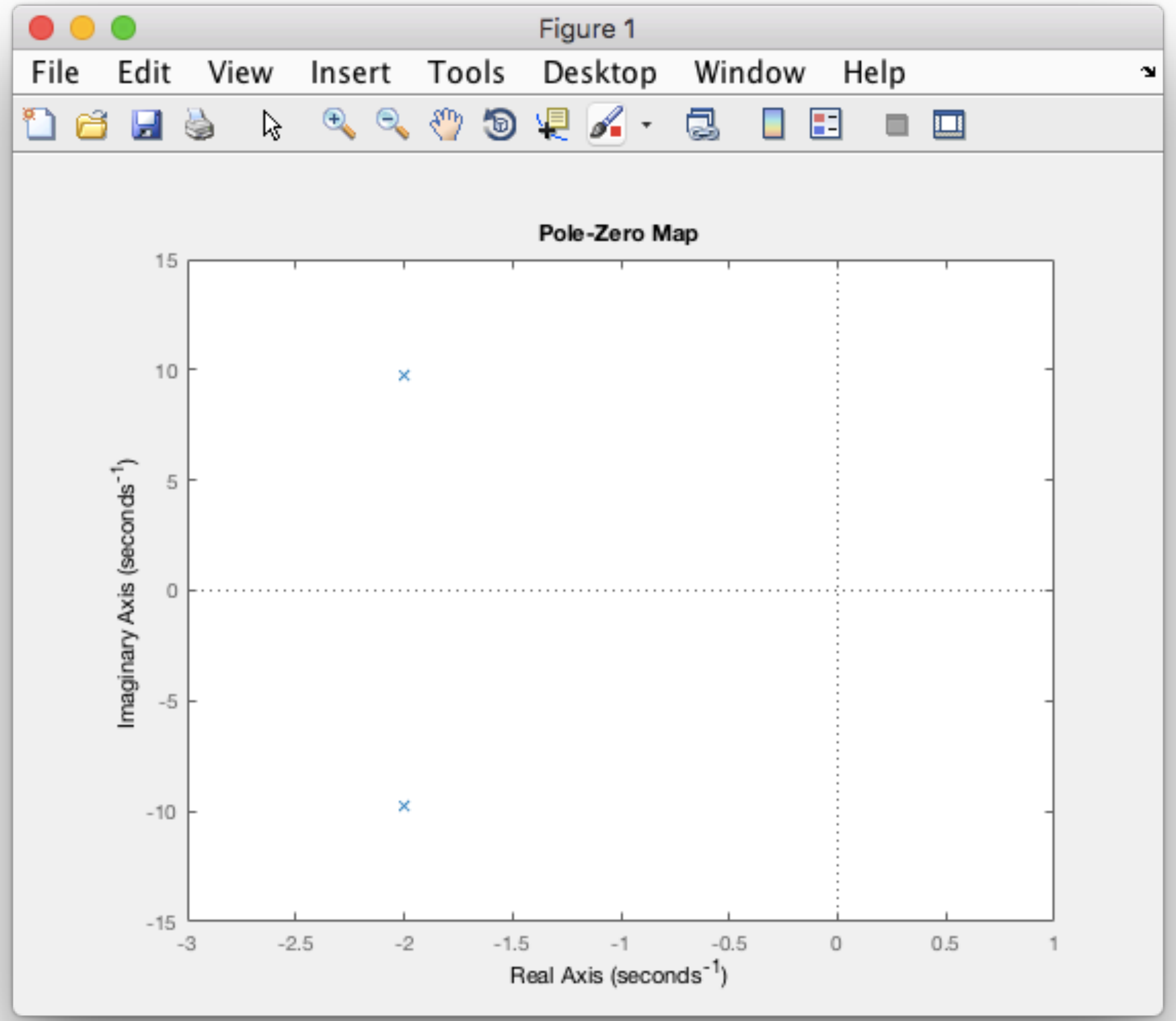
UNDERDAMPED SYSTEM

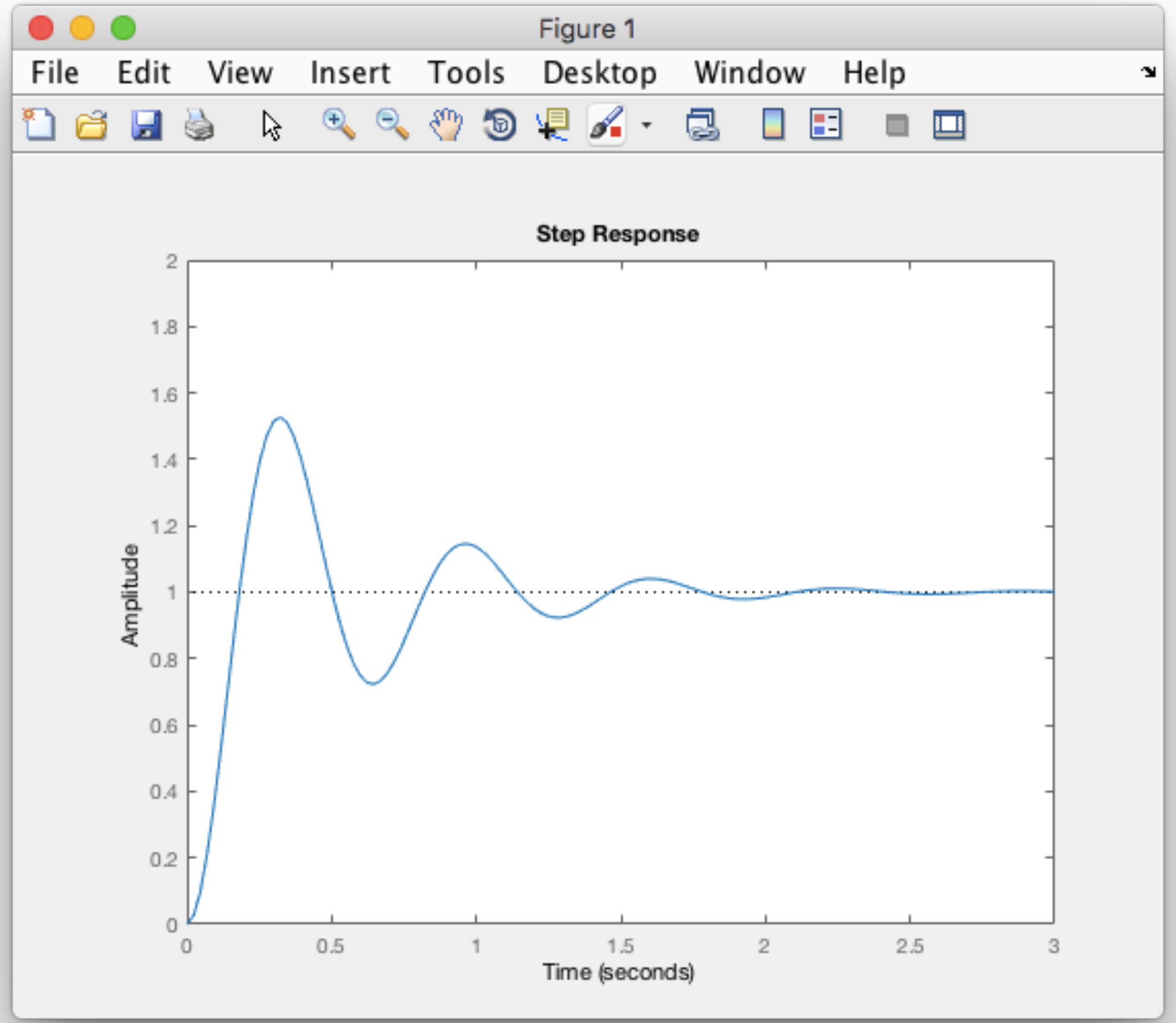
$$G(s) = \frac{1}{ms^2 + bs + k} = \frac{k_{dc}\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2}$$

- If $\zeta < 1$ then the system is **underdamped**. In this case, both poles are complex-valued with negative real parts; therefore, the system is stable but oscillates while approaching the steady-state value.

```
k_dc = 1;
w_n = 10;
zeta = 0.2;
s = tf('s');
G1 = k_dc*w_n^2/(s^2 + 2*zeta*w_n*s + w_n^2);

pzmap(G1)
axis([-3 1 -15 15])
figure
step(G1)
axis([0 3 0 2])
```

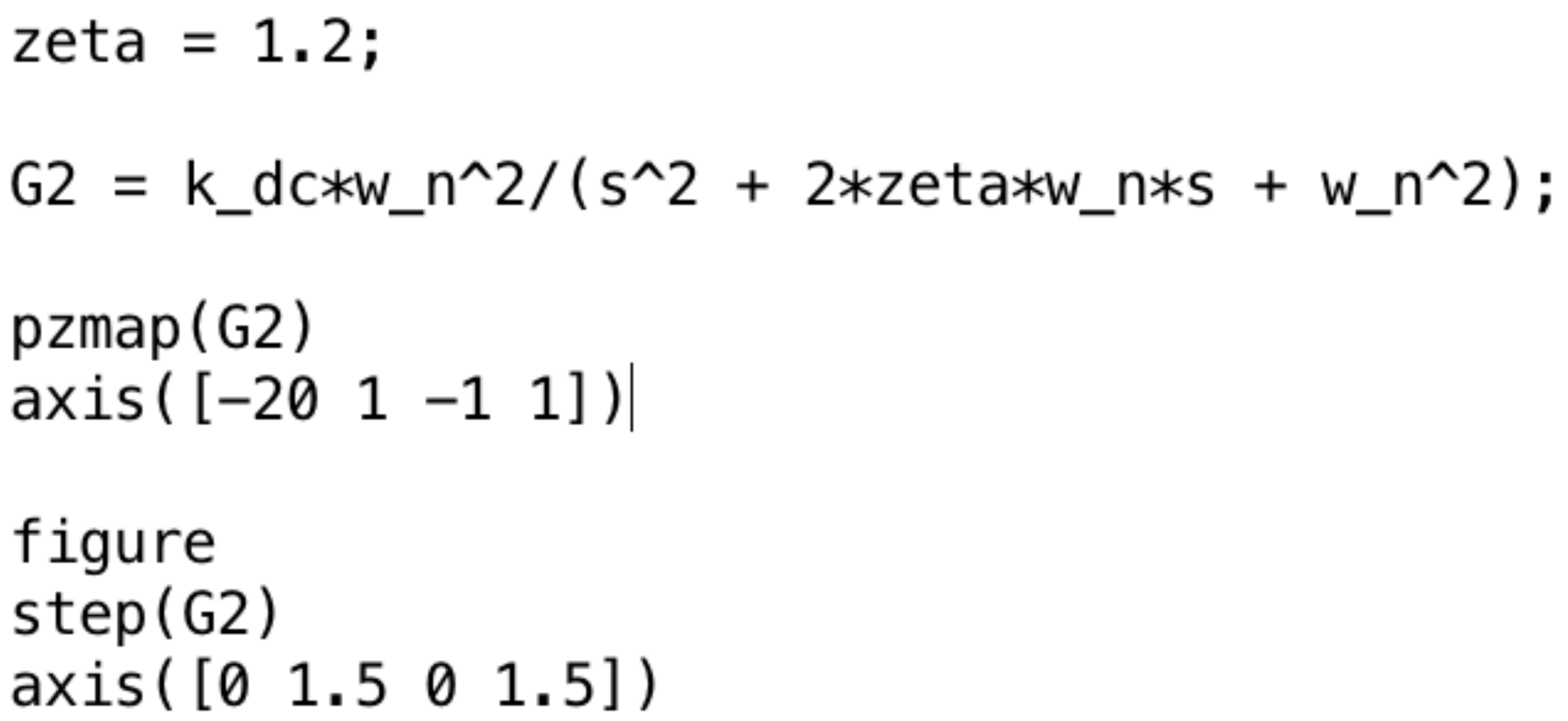
UNDERDAMPED SYSTEM





OVERDAMPED SYSTEM

- If $\zeta > 1$ then the system is **overdamped**. Both poles are real and negative; therefore, the system is stable and does not oscillate.



```
zeta = 1.2;

G2 = k_dc*w_n^2/(s^2 + 2*zeta*w_n*s + w_n^2);

pzmap(G2)
axis([-20 1 -1 1])

figure
step(G2)
axis([0 1.5 0 1.5])
```

For same system, change the value of the damping ratio to make the system firstly critically damped, secondly undamped, and re-plot the step response and pole-zero map.

homework#3.1

ANIMATION FOR THE BUS SUSPENSION EXAMPLE

- **Purpose :** The purpose of this Graphical User Interface (GUI) is to allow the user to view an animation of the Bus Suspension system with the step disturbance response plot. This allows the user to see the correlation between the plot and the system's physical response.

- **Running the GUI :** To run the GUI you will need 2 files. Copy each of them to the directory in which you are running MATLAB.

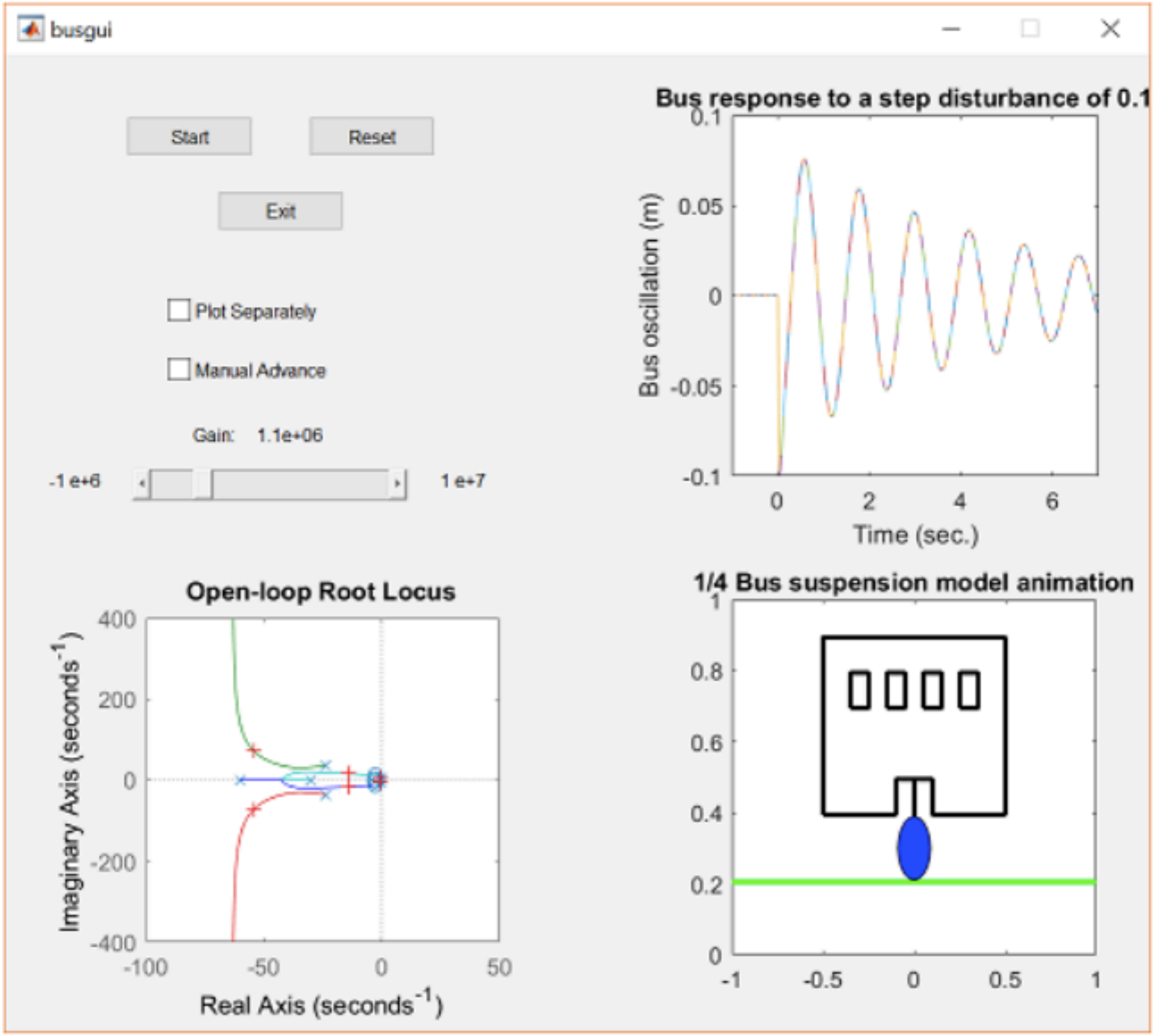
  busgui.fig - contains the graphical interface.

  busgui.m - contains the GUI callback function.

- Once the files are copied into your MATLAB directory, simply enter the following command

```
busgui
```

ANIMATION FOR THE BUS SUSPENSION EXAMPLE


busgui
Start
Reset
Exit
Plot Separately
Manual Advance
Gain: 1.1e+06
-1 e+6
1 e+7
Bus response to a step disturbance of 0.1
Bus oscillation (m)
Time (sec.)
Open-loop Root Locus
Imaginary Axis (seconds^-1)
Real Axis (seconds^-1)
1/4 Bus suspension model animation