

Using Matlab's Control System Toolbox to study continuous time linear system

The Control System Toolbox in Matlab supports continuous-time and discrete-time linear models of the following types:

- Transfer functions
- State-space models

Here we will discuss transfer function modeling of continuous-time function and leave the rest to future chapters.

Creating Models

The matlab function `tf` creates *transfer functions*.

`tf` needs two MATLAB-vectors, one containing the coefficients of the *numerator* polynomial - taken in descending orders - and one for the *denominator* polynomial of the transfer function.

As an example we will create the transfer function

$$H_0(s) = \frac{K_0 s}{T_0 s + 1}$$

```
>> K0=2; T0=4;
>> num0=[K0 0];
>> den0=[T0 1];
>> H0=tf(num0,den0);
```

To display H_0 , we execute

```
>> H0
```

Transfer function:

```
  2 s
-----
 4 s + 1
```

An LTI-model can contain a *time-delay*, as in the following model:

$$H_d(s) = \frac{K_0 s}{T_0 s + 1} e^{-T_{\text{delay}} s}$$

This system is created by (except from the time-delay term, H_d is equal to H_0 defined above):

```
>> Tdelay = 1;
>> Hd = tf(num0,den0, 'InputDelay', Tdelay)
```

Transfer function:

$$\exp(-1*s) * \frac{2 s}{4 s + 1}$$

Suppose we have created a transfer function, and we want to *retrieve detailed information* about the model. The function `tfdata` retrieves the information:

```
>> [numHd,denHd] = tfdata(Hd, 'v')
numHd =
     2     0
denHd =
     4     1
```

'v' indicates that we are dealing with 1-d input and output.

Input delay can be retrieved by the `get` command:

```
>> get(Hd, 'InputDelay')
ans =
     1
```

Try the following command:

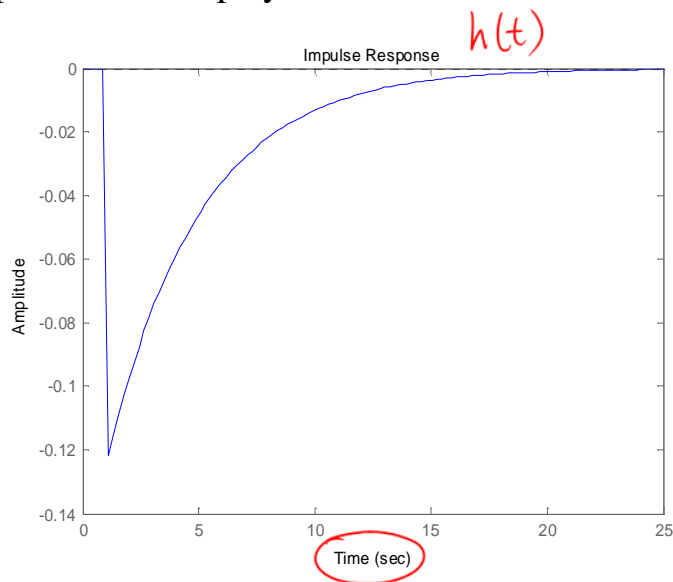
```
>> get(Hd)
```

Time Response

To show the impulse response of a system, we use the command `impulse`:

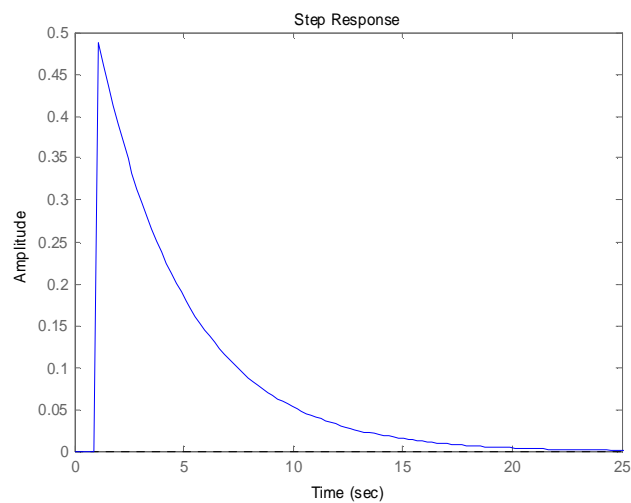
```
>> impulse(Hd)
```

The following plot will be displayed:



We can also get the step response:

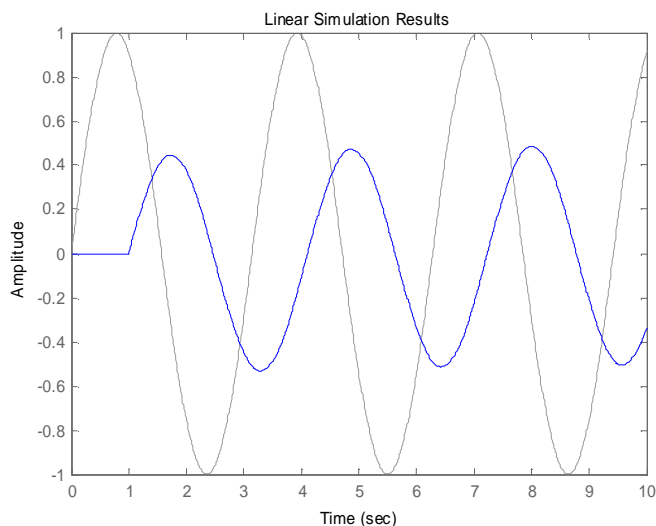
```
>> step(Hd)
```



It should be noted that in order to actually plot the time response, Matlab always converts the continuous-time output signal into discrete-time signal by sampling at the “appropriate” sampling frequency.

We can actually simulate the system for any input by using the `lsim` command, which plots both the input (gray color) and the output (blue):

```
>> t = 0:0.01:10;      % time range
>> x = sin(2*t);      % input
>> lsim(Hd,x,t)
```



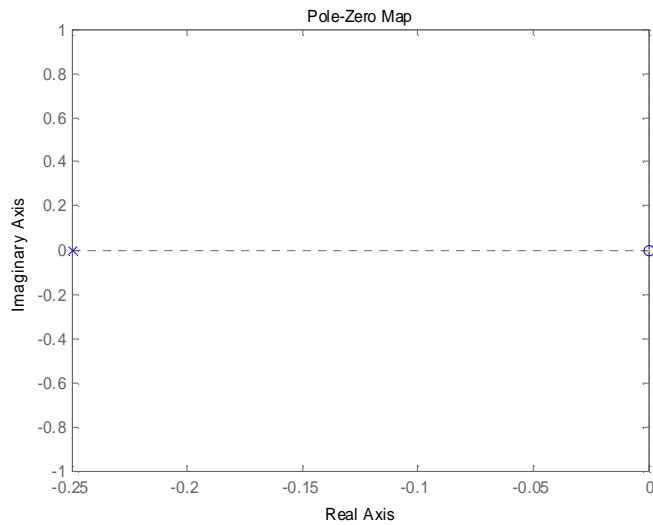
Poles and Zeros

Poles and Zeros can be retrieved using the command `pole` and `zero`. Their locations are conveniently displayed by the command `pzmap`.

```
>> pole(Hd)
ans =
    -0.2500
```

```
>> zero(Hd)
ans =
     0
```

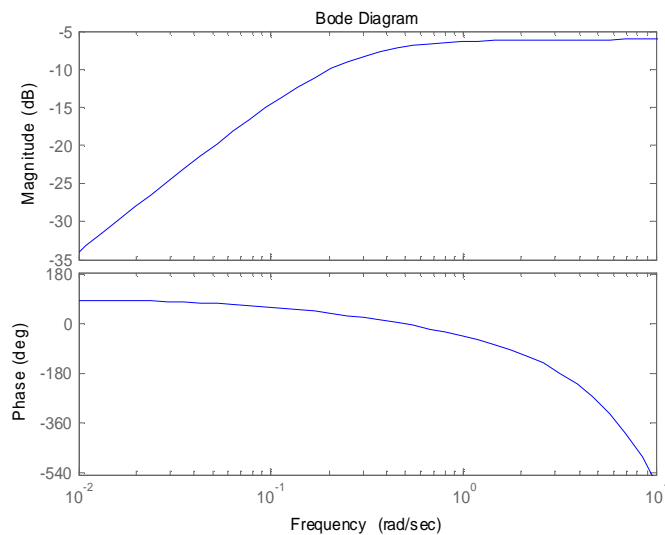
```
>> pzmap(Hd)
```



Frequency Response

Frequency response of a system is the Fourier Transform of the impulse response. It is an important analysis and design tool which will be the subject of our next lecture. To get a preview, we can use the command `bode` to plot the magnitude and phase of the frequency response:

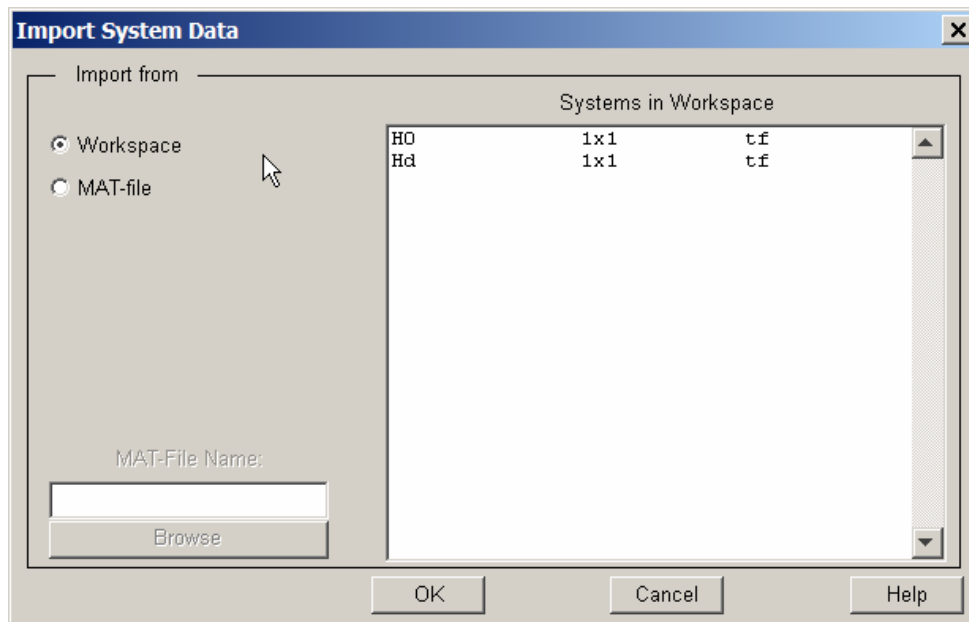
```
>> bode(Hd)
```



Finally, to make things really convenient for you, Matlab has wrapped all the above commands (plus a lot more) into the function `ltiview`:

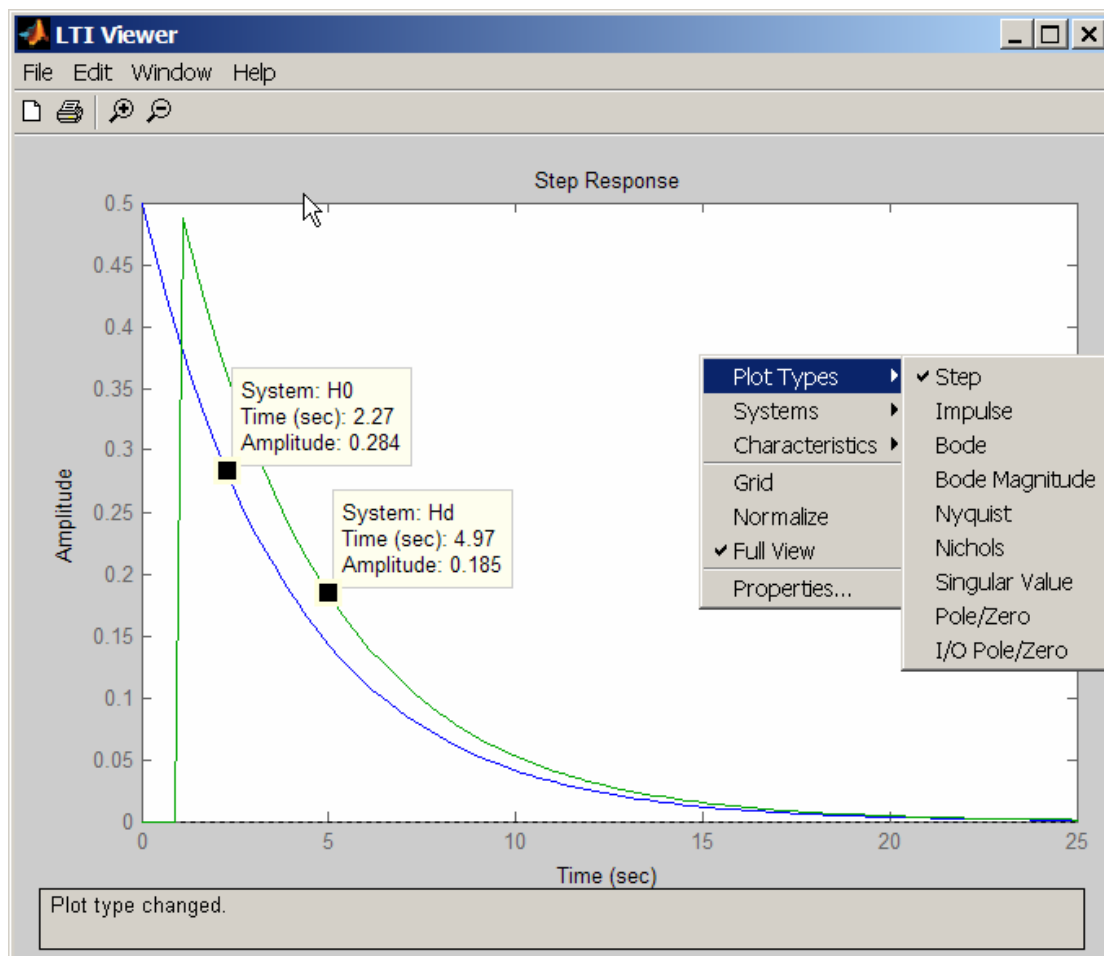
```
>> ltiview
```

A plot window will pop up. Click Import under the File menu and the following window will open:



This window allows use to select all the LTI systems we have defined so far for analysis. You can click to select any one of them (or CTRL-click to select multiple).

The nicest thing about `ltiview` is that you can show many different kinds of plots by clicking the right mouse button and select the one you want. Also, you can click on any point of the curve with the left mouse button to get the precise numerical value as well as the identification of the corresponding system:



Using `ltiview`, get the impulse response, step response, bode plot, and the pole zero plots of the following transfer functions:

a. $H(s) = \frac{s+1}{s^2+2}$ b. $H(s) = s + \frac{1}{s+1}$ c. $H(s) = \frac{s}{s-1}$ d. $H(s) = 1$