Introduction to MATLAB

LAB # 01

Spring 2024

CSE-310L Control Systems Lab

Submitted by: **Ali Asghar**

Registration No.: **21PWCSE2059**

Class Section: **C**

"On my honor, as student of University of Engineering and Technology, I have neither given nor received unauthorized assistance on this academic work."

Submitted to:

Dr. Muniba Ashfaq

Date:

19th March 2024

Department of Computer Systems Engineering

University of Engineering and Technology, Peshawar

Objectives:

The objective of this lab is to learn about the following built-in MATLAB functions:

- roots
- poly
- polyval
- tf
- conv
- pzmap
- impulse
- step
- residue
- series
- parallel
- feedback

Introduction

This lab report summarizes various MATLAB functions used in control systems analysis and design. Each function plays a significant role in handling polynomials, transfer functions, and system responses.

MATLAB Functions

roots

The roots function finds the roots of a polynomial given its coefficients. It is useful for determining the stability of control systems.

For example, in the code below I have used the roots function to find roots of the polynomial $\mathbf{4x^2 - x - 3}$ which are **1** and **-0.75**

Editor - U:\ControlSystemLab1\MATLAB\RootsExample.m

RootsExample.m

```
C = [4, -1, -3]; %co-efficient of 4x^2 - x - 3
rts = roots(C);
disp(rts)
```

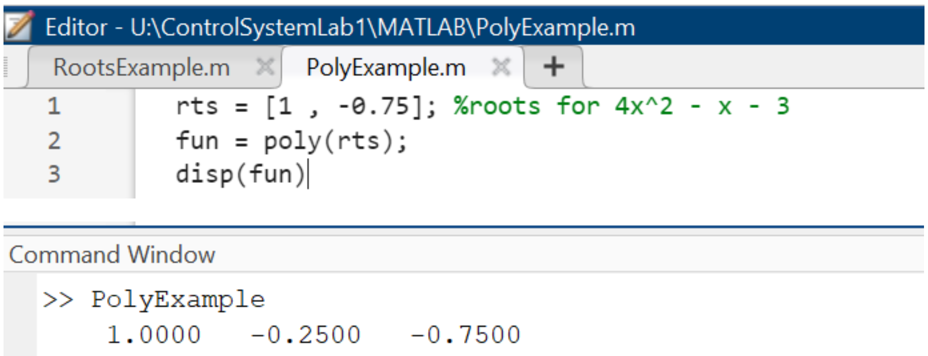
ommand Window

```
>> RootsExample
    1.0000
   -0.7500
```

poly

The poly function generates a polynomial with specified roots. This is useful for constructing system transfer functions from desired pole locations.

In below code, I have given the roots of above polynomial and generated the polynomial of my 1st example above. It can be seen in the output that the coefficients generated by poly functions are the same as 1st example polynomial's coefficients when multiplied by 4.

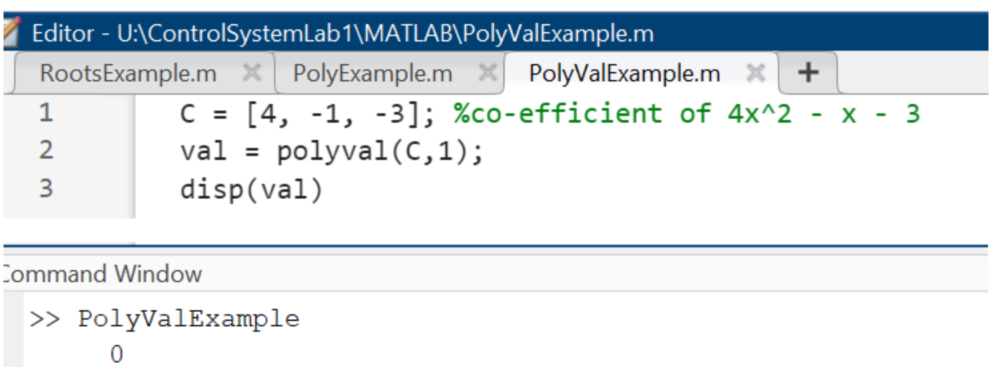


polyval

The polyval function evaluates a polynomial for a given set of values. It is useful for analyzing system outputs at specific input values.

In the code below, I have evaluated the polynomial $\boldsymbol{4x^2 - x - 3}$ by **1**.

This gives, P(1) = 4(1) – 1 – 3 => P(1) = 4 -4 => P(1) = 0



tf

The tf function creates a transfer function model from numerator and denominator coefficients. It is essential for representing dynamic systems.

Editor - U:\ControlSystemLab1\MATLAB\tfExample.m

RootsExample.m | PolyExample.m | PolyValExample.m | tfExample.m

```
C1 = [4, -1, -3]; %co-efficient of 4x^2 - x - 3
C2 = [2, 1, -2]; %co-efficient of 2x^2 + x - 2
trFun = tf(C1, C2);
disp(trFun)
```

Command Window

```
>> tfExample
  tf with properties:

      Numerator: {[4 -1 -3]}
    Denominator: {[2 1 -2]}
       Variable: 's'
        IODelay: [0]
     InputDelay: [0]
    OutputDelay: [0]
      InputName: {''}
      InputUnit: {''}
     InputGroup: [1×1 struct]
     OutputName: {''}
     OutputUnit: {''}
    OutputGroup: [1×1 struct]
          Notes: [0×1 string]
       UserData: []
```

conv

The conv function computes the convolution of two sequences. This is useful for determining the output of a system in response to a given input.

In below code, I have taken two vectors C1 and C2 for convolution. The length of output is length of C1 + length of C2 – 1. Result can be seen in the command screen.

Editor - U:\ControlSystemLab1\MATLAB\convExample.m

RootsExample.m | PolyExample.m | PolyValExample.m | tfExample.m | convExample.m

```
C1 = [4, -1, -3]; %co-efficient of 4x^2 - x - 3
C2 = [2, 1, -2]; %co-efficient of 2x^2 + x - 2
product = conv(C1, C2);
disp(product)
```

Command Window

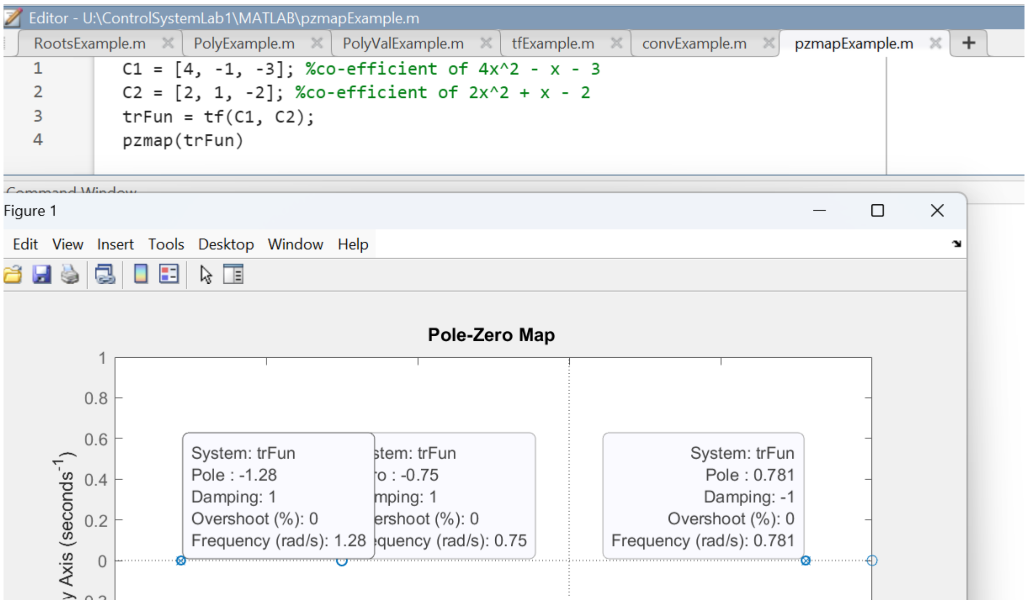
```
>> convExample
     8     2   -15    -1     6

>>
```

pzmap

The pzmap function generates a pole-zero map of a transfer function. This visualization helps in analyzing system stability and performance.

In below code, the system is $H(s) = \frac{4s^2 - s - 3}{2s^2 + s - 2}$

The poles are the roots of denominator while zeros are roots of numerator.



impulse

The impulse function computes the impulse response of a system. It is vital for understanding how systems respond to sudden inputs.

In below code, the system is $H(s) = \frac{s^2 - 6}{s^2 + s - 2}$

```
C1 = [1, 0, -6]; %co-efficient of x^2 - 6
C2 = [1, 1, -2]; %co-efficient of x^2 + x - 2
transferFunction = tf(C1, C2);
[ImpResp,t] = impulse(transferFunction);

% Plot the step response
figure;
plot(t, ImpResp);
title('Impulse Response of the Transfer Function');
xlabel('Time (seconds)');
ylabel('Amplitude');
grid on;
```

step

The step function computes the step response of a system. It helps evaluate how a system reacts to a step input over time.

In below code, the system is $H(s) = \frac{s^2-6}{s^2+s-2}$

```
C1 = [1, 0, -6]; %co-efficient of x^2 - 6
C2 = [1, 1, -2]; %co-efficient of x^2 + x - 2
transferFunction = tf(C1, C2);

% Compute the step response
[StepResp, t] = step(transferFunction);
% Plot the step response
figure;
plot(t, StepResp);
title('Step Response of the Transfer Function');
xlabel('Time (seconds)');
ylabel('Amplitude');
grid on;
```

residue

The residue function performs partial fraction decomposition. This is useful for simplifying transfer functions into summable terms.

```
C1 = [2, 0, -3]; %co-efficient of 2x^2 - 3
C2 = [2, 1, -2]; %co-efficient of 2x^2 + x - 2
[R, P, K] = residue(C1, C2);
```

series

The series function connects two transfer functions in series. This is important for cascading system responses.

```
Editor - U:\ControlSystemLab1\MATLAB\seriesExample.m
impulseExample.m  stepExample.m  tfExample.m  residueExample.m  seriesExample.m  parallelExample.m  cor
```

```matlab
C1 = [4, -1, -3]; %co-efficient of 4x^2 - x - 3
C2 = [2, 1, -2]; %co-efficient of 2x^2 + x - 2

C3 = [1, -3]; %co-efficient of x^2 - 3
C4 = [1, 1, 1]; %co-efficient of x^2 + x + 1

S1 = tf(C1, C2);
S2 = tf(C3,C4);

series(S1,S2)
```

```
ommand Window

ans =

      4 s^3 - 13 s^2 + 9
  ---------------------------
  2 s^4 + 3 s^3 + s^2 - s - 2
```

parallel

The parallel function combines two transfer functions in parallel. This is useful for systems with multiple pathways.

```
Editor - U:\ControlSystemLab1\MATLAB\parallelExample.m
impulseExample.m  stepExample.m  tfExample.m  residueExample.m  seriesExample.m  parallelExample.m
```

```matlab
C1 = [4, -1, -3]; %co-efficient of 4x^2 - x - 3
C2 = [2, 1, -2]; %co-efficient of 2x^2 + x - 2

C3 = [1, -3]; %co-efficient of x^2 - 3
C4 = [1, 1, 1]; %co-efficient of x^2 + x + 1

S1 = tf(C1, C2);
S2 = tf(C3,C4);
feedback(S1,S2)
```

```
ommand Window

ans =

     4 s^4 + 3 s^3 - 4 s - 3
  ------------------------------
  2 s^4 + 7 s^3 - 12 s^2 - s + 7
```

feedback

The feedback function computes the closed-loop transfer function of a system with feedback. This is essential for analyzing control system stability.

Editor - U:\ControlSystemLab1\MATLAB\feedbackExample.m

stepExample.m | tfExample.m | residueExample.m | seriesExample.m | parallelExample.m | convExample.m | feedbackExample.m

```matlab
C1 = [4, -1, -3]; %co-efficient of 4x^2 - x - 3
C2 = [2, 1, -2]; %co-efficient of 2x^2 + x - 2

C3 = [1, -3]; %co-efficient of x - 3
C4 = [1, 1, 1]; %co-efficient of x^2 + x + 1

S1 = tf(C1, C2);
S2 = tf(C3,C4);

feedback(S1,S2)
```

Command Window

```
ans =

     4 s^4 + 3 s^3 - 4 s - 3
  ------------------------------
  2 s^4 + 7 s^3 - 12 s^2 - s + 7
```