matlab

Linear_Control_Systems_14031

```
clc;                           %clear command window
clear all;                     %clear workspace
close all;                     %close all figures and plots
```

Complex Numbers

we can find the magnitude and angle of a complex number, Q using abs (Q) and angle (Q)

```
Z = 3 + 4j;              %define complex number
Z
```

```
Z = 3.0000 + 4.0000i
```

```
real(Z)             %Real part of complex number
```

```
ans = 3
```

```
imag(Z)             %imaginary part of complex number
```

```
ans = 4
```

```
angle(Z)            %Phase angle in radian
```

```
ans = 0.9273
```

```
conj(Z)             %Element-wise complex conjugate
```

```
ans = 3.0000 - 4.0000i
```

```
MagZ = abs(Z)           % Find magnitude of Z.
```

```
MagZ = 5
```

```
ThetaZ = (180/pi)*angle(Z)              % Find the angle of Z in degrees.
```

```
ThetaZ = 53.1301
```

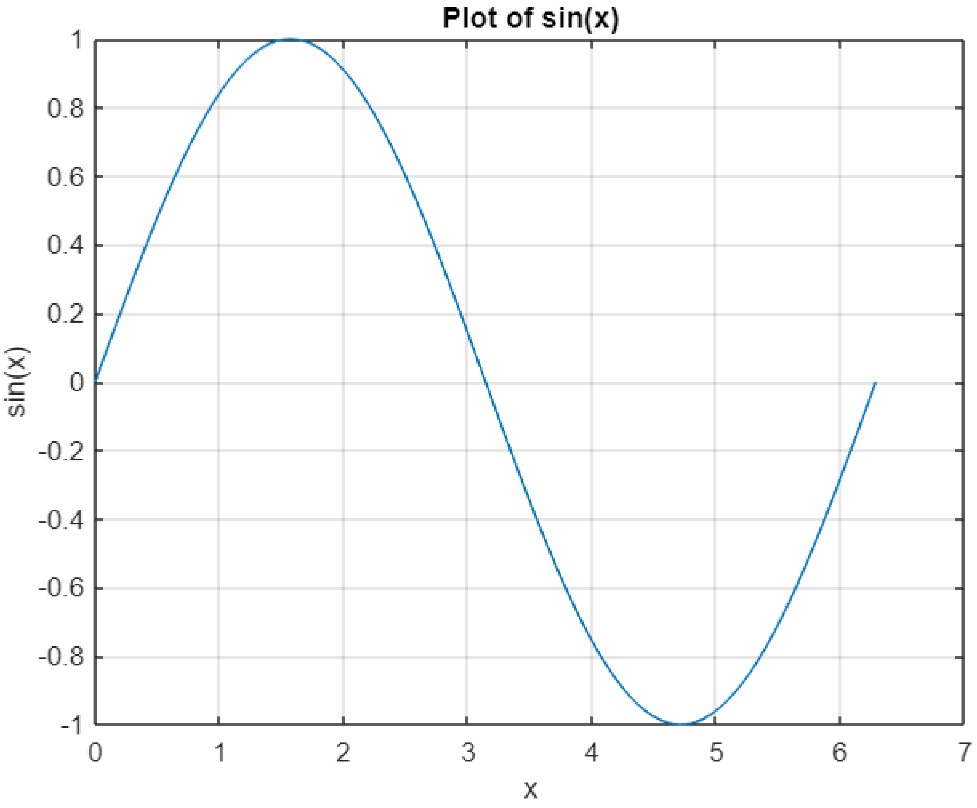
Elementary Functions

```
x = linspace(0, 2*pi, 100);
```

```
y = sin(x) % Sine of x (in radians)
```

```
y = 1×100
         0    0.0634    0.1266    0.1893    0.2511    0.3120    0.3717    0.4298 ⋯
```

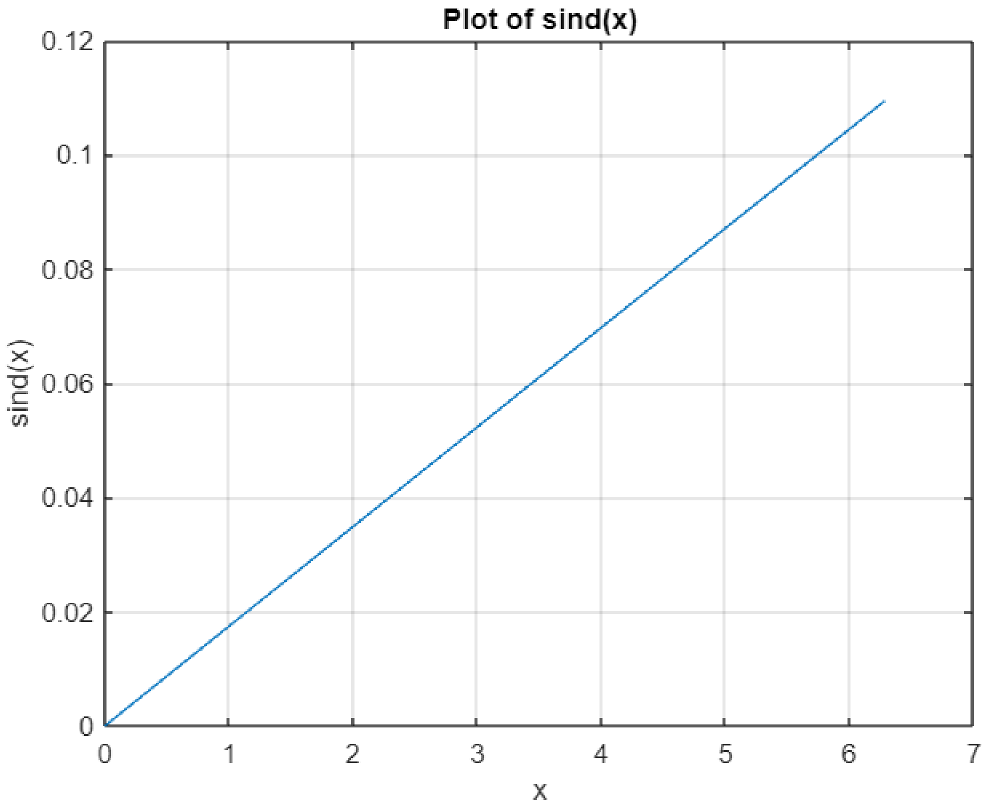
```
plot(x, y);
xlabel('x'); % Label for x-axis
ylabel('sin(x)'); % Label for y-axis
title('Plot of sin(x)'); % Title of the plot
grid on; % Turn on grid for better readability
```



```
y=sind(x)
```

```
y = 1×100
         0    0.0011    0.0022    0.0033    0.0044    0.0055    0.0066    0.0078 ⋯
```

```
plot(x, y);
xlabel('x'); % Label for x-axis
ylabel('sind(x)'); % Label for y-axis
title('Plot of sind(x)'); % Title of the plot
grid on;
```



```
sinh(x) %Analogous for the other trigonometric functions:cos, tan, csc, sec, and cot
```

```
ans = 1×100
         0    0.0635    0.1273    0.1916    0.2566    0.3227    0.3901    0.4590 ⋯
```

```
atan2(-2,-2)
```

```
ans = -2.3562
```

```
exp(1)
```

```
ans = 2.7183
```

logical

```
1>2
```

```
ans = logical
   0
```

```
1<2
```

```
ans = logical
   1
```

```
1==2
```

```
ans = logical
   0
```

```
1<=2
```

```
ans = logical
   1
```

```
1~=2
```

```
ans = logical
   1
```

```
s=true
```

```
s = logical
   1
```

```
g=false
```

```
g = logical
   0
```

```
s&g
```

```
ans = logical
   0
```

```
s|g
```

```
ans = logical
   1
```

```
~s
```

```
ans = logical
   0
```

Matrices and Arrays

Matrix and Vector Creation

```
A=[1 1 0 0;0 1 0 1;1 0 0 0;1 0 0 1]
```

```
A = 4×4
     1     1     0     0
     0     1     0     1
     1     0     0     0
     1     0     0     1
```

```
A(6)
```

```
ans = 1
```

```
F=A(2:4,2:4)
```

```
F = 3×3
     1     0     1
     0     0     0
     0     0     1
```

```
V = [1; 2; 3; 4]
```

```
V = 4×1
     1
     2
     3
     4
```

```
V(2)
```

```
ans = 2
```

```
length(A)
```

```
ans = 4
```

```
size(A)
```

```
ans = 1×2
     4     4
```

```
diag(A)
```

```
ans = 4×1
     1
     1
     0
     1
```

```
diag(V)
```

```
ans = 4×4
     1     0     0     0
     0     2     0     0
     0     0     3     0
     0     0     0     4
```

```
numel(A)
```

```
ans = 16
```

```
det(A)
```

```
ans = 0
```

```
rank(A)
```

```
ans = 3
```

```
trace(A)
```

```
ans = 3
```

```
zeros(3, 2)
```

```
ans = 3×2
     0     0
     0     0
     0     0
```

```
ones(3, 3)
```

```
ans = 3×3
     1     1     1
     1     1     1
     1     1     1
```

```
eye(3)
```

```
ans = 3×3
     1     0     0
     0     1     0
     0     0     1
```

Matrix Operations

```
C = A + A
```

```
C = 4×4
     2     2     0     0
     0     2     0     2
     2     0     0     0
     2     0     0     2
```

```
D = A * V
```

```
D = 4×1
     3
     6
     1
     5
```

```
AT = A'
```

```
AT = 4×4
     1     0     1     1
     1     1     0     0
     0     0     0     0
     0     1     0     1
```

```
A_inv = inv(A)          %computes the inverse of matrix
```

```
Warning: Matrix is singular to working precision.
A_inv = 4×4
   Inf   Inf   Inf   Inf
   Inf   Inf   Inf   Inf
   Inf   Inf   Inf   Inf
   Inf   Inf   Inf   Inf
```

```
d = det(A)              %Computes the determinant of matrix
```

```
d = 0
```

```
b=[2;3;4;5]
```

```
b = 4×1
     2
     3
     4
     5
```

```
x = A\b                 % Solves Ax = b for x
```

```
Warning: Matrix is singular to working precision.
x = 4×1
   NaN
   NaN
   Inf
     3
```

```
C = A .* b
```

```
C = 4×4
     2     2     0     0
     0     3     0     3
     4     0     0     0
     5     0     0     5
```

```
b=[2 3 4 5;3 4 5 6;4 5 6 7;5 6 7 8]
```

```
b = 4×4
     2     3     4     5
     3     4     5     6
     4     5     6     7
     5     6     7     8
```

```
C=A*b
```

```
C = 4×4
     5     7     9    11
     8    10    12    14
     2     3     4     5
     7     9    11    13
```

```
C = A .* b
```

```
C = 4×4
     2     3     0     0
     0     4     0     6
     4     0     0     0
     5     0     0     8
```

```
C = A ./ b
```

```
C = 4×4
    0.5000    0.3333         0         0
         0    0.2500         0    0.1667
    0.2500         0         0         0
    0.2000         0         0    0.1250
```

```
C = A.^2
```

```
C = 4×4
     1     1     0     0
     0     1     0     1
     1     0     0     0
     1     0     0     1
```

```
syms x y z
A = [x y; y z]
```

A =

$$\begin{pmatrix} x & y \\ y & z \end{pmatrix}$$

Vector Operations

```
v1 = [1 2 3];
v2 = [4 5 6];
dp = dot(v1, v2)                %dot product
```

```
dp = 32
```

```
cp = cross(v1, v2)              %cross product
```

```
cp = 1×3
    -3     6    -3
```

```
n = norm(v1)                % Euclidean norm (2-norm)
```

```
n = 3.7417
```

```
n1 = norm(v1, 1)                % 1-norm (sum of absolute values)
```

```
n1 = 6
```

```
nInf = norm(v1, Inf)                % Infinity norm (max value)
```

```
nInf = 3
```

```

J=[7 9 5; 6 1 9; 4 3 2]
```

```
J = 3×3
     7     9     5
     6     1     9
     4     3     2
```

```
K=[1 2 3; 4 5 6; 7 8 9]
```

```
K = 3×3
     1     2     3
     4     5     6
     7     8     9
```

```
L=[9 8 7; 6 5 4; 3 2 1]
```

```
L = 3×3
     9     8     7
     6     5     4
     3     2     1
```

```
cat(3,J,K,L)
```

```
ans = 
ans(:,:,1) =

     7     9     5
     6     1     9
     4     3     2


ans(:,:,2) =

     1     2     3
     4     5     6
     7     8     9


ans(:,:,3) =

     9     8     7
     6     5     4
     3     2     1
```

```
ch=poly(cp)
```

```
ch = 1×4
     1     0   -27   -54
```

```
roots(ch)
```

```
ans = 3×1
    6.0000
   -3.0000
   -3.0000
```

loops

```
%%if
G=0;
if G>0
    f=1
elseif G==0
    f=0
else
```

```
    f=-1
end
```

```
f = 0
```

```

%%for
x1=0;
for i=1:10
    x1=x1+1/i^2;
    i=i+1;
end
x1
```

```
x1 = 1.5498
```

```

%%while
G=0;
f=10;
i=1;
while i<=f

    G=G+i;
    if i==5
        break;
    end
    i=i+1;
end
G
```

```
G = 15
```

Helpful links and courses

we share some good links and courses for matlab. these can help you to be more familiar with matlab world and improves your coding skills. In this course, our team primarily focus on Simulink and MATLAB within the context of linear control systems. However, MATLAB offers a vast range of functions in various other domains, such as image and sound processing, signal processing, neural networks, and more. We aim to introduce you to some of these areas and provide support to help you learn and explore their educational applications.

some good youtube channels:

https://www.youtube.com/@AnselmGriffin

https://www.youtube.com/channel/UCFa6AP9Ts4EFZWA6p_TPJmA

persian courses:

https://faradars.org/courses/fvee96033-application-of-matlab-in-linear-control

websites:

https://undocumentedmatlab.com/

https://ww2.mathworks.cn/en/

https://matlabacademy.mathworks.com/details/matlab-onramp/gettingstarted

https://matlabacademy.mathworks.com/details/simulink-onramp/simulink

https://matlabacademy.mathworks.com/details/simulink-onramp/simulink

https://www.coursera.org/learn/matlab

https://matlabacademy.mathworks.com/details/control-design-onramp-with-simulink/controls

also our course GitHub page:

https://github.com/LinearControlSystems

https://github.com/LinearControlSystems/MATLAB-Simulink-Applications

This repository contains codes, examples, and educational files related to the Linear Control Systems course using MATLAB and Simulink.

thank you for your attention

Good luck