1. Introduction to MATLAB

Simple Initialization

```
A = 5

B = 2

C = A + B

C

Out [1] : 7
```

Array Operations

MATLAB is ideally suited for working with Array and Matrices.

```
% Initialize array of Ones

A = Ones (1,10)      1x10 array of ones.
          Rows  columns

A =

    1  1  1  1  1  1  1  1  1  1

% Initialize matrix of Ones

A = Ones (5, 5)

A =

    1  1  1  1  1
    1  1  1  1  1
    1  1  1  1  1
    1  1  1  1  1
    1  1  1  1  1

% Initialize array of Zeros

A = Zeros (2,2)

A =  0  0
     0  0
```

Array Creation

% Initialize a general array

[1, 2, 3, 4]

ans =

1 2 3 4

% Initialize a matrix, (ie) 2D array

[1, 2 ; 3, 4]

ans =

1 2

3 4

Uniform spaced Array

% Generate array of integers 1 to 10

A = [1 : 10]

A =

1 2 3 4 5 6 7 8 9 10

% Can also specify step-size

A = [1 : 2 : 10]

A =

1 3 5 7 9

Array addition

A = [1 : 10]

B = ones (1, 10)

A + 2 * B

ans =

3 4 5 6 7 8 9 10 11 12

A	→	1	2	3	4	5	6	7	8	9	10
B	→	1	1	1	1	1	1	1	1	1	1
2*B	→	2	2	2	2	2	2	2	2	2	2
A+2*B	→	3	4	5	6	7	8	9	10	11	12

Array multiplication

% Element wise multiplication using .*

A = [1 : 10]

B = 2 * Ones (1, 10)

A .* B

ans =

2 4 6 8 10 12 14 16 18 20

A →	1	2	3	4	5	6	7	8	9	10
B →	2	2	2	2	2	2	2	2	2	2
A .* B →	2	4	6	8	10	12	14	16	18	20

Array division (Element wise division using ./)

A = [1 : 10]

B = 2 * Ones (1, 10)

A ./ B

ans =

0.5 1 1.5 2 2.5 3 3.5 4 4.5 5

A →	1	2	3	4	5	6	7	8	9	10
B →	2	2	2	2	2	2	2	2	2	2
A ./B →	0.5	1	1.5	2	2.5	3	3.5	4	4.5	5

Exponentiation

For exponentiation, use ^ and .^

A = [1 : 10]

A .^ 2

ans =

1 4 9 16 25 36 49 64 81 100

A →	1	2	3	4	5	6	7	8	9	10
A.^2 →	1^2	2^2	3^2	4^2	5^2	6^2	7^2	8^2	9^2	10^2
	1	4	9	16	25	36	49	64	81	100

B = [1 2 ; 3 4]

B.^2 % Element wise exponentiation

ans =

1 4
9 16

B ^2 % Matrix exponentiation (A * A)

ans =

7 10
15 22

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} = \begin{bmatrix} 7 & 10 \\ 15 & 22 \end{bmatrix}$$

Logical operators

```
A = [1 : 10]
A > 5
ans =
      0  0  0  0  0  1  1  1  1  1

A ~= 5
ans =
      1  1  1  1  0  1  1  1  1  1
```

General commands (Self explanatory)

```
abs()       log2()      real()
exp()       log10()     imag()
sqrt()      max()       sum()
```

Random numbers

```
% Generate samples of Gaussian RV
% Mean = 0 , Variance = 1
randn()
ans =
      1.8339
```

Gaussian RV

```
% Generate block of samples of Gaussian RV
% with arbitrary mean and variance
% mean + sqrt(variance) * randn(1, blocklength)
2 + sqrt(5) * randn(1, 10)
ans =  3.2023    6.1007   -3.0509    3.9279    2.7128
      -0.9241    1.0305    2.7661   10.0015    8.1926
```

% Generate matrix of samples of Gaussian RV

% mean + sqrt (variance) * randn (m, n)

2 + sqrt (5) * randn (2, 2)

ans =

3.5015	3.6038
-0.7000	5.6453

Generate 10 iid samples White Gaussian Noise of power 3 dB.

Mean = 0 (loz. of White Noise).

Variance = SNR = 2

Solution:

0 + sqrt(2) * randn (1, 10).

$10 \log_{10} SNR = SNR\_dB$

$10 \log_{10} SNR = 3$

$\log_{10} SNR = 0.3$

$SNR = 10^{0.3}$

$\boxed{SNR \approx 2}$

Random integers

To generate array and matrix of random integers, use Syntaxes below:

% randi ([0, a], m, n)

% Generates m x n array of random integers in [0, a]

$\text{randi}(\underbrace{[0,5]}_{\text{Random integer value b/w 0 and 5}},\ \overbrace{2,2}^{2\times 2\ \text{Matrix}})$

ans =

3	1
0	4

BPSK symbols

To generate BPSK symbols ±1, use syntax below.

```
2 * randi([0,1]) - 1

ans =

   -1
```

To generate an array

```
2 * randi([0,1], 2, 2) - 1

ans =

    1   -1
    1    1
```

Generate a 2×2 array of QPSK symbols of average power of 3 dB.

```
(2 * randi([0,1], 2, 2) - 1)

+ 1i (2 * randi([0,1], 2, 2) - 1)
```

$$10 \log_{10} SNR = SNR\_dB$$

$$10 \log_{10} SNR = 3$$

$$\log_{10} SNR = 0.3$$

$$SNR = 10^{0.3}$$

$$\boxed{SNR \approx 2}$$

Linear Algebra

```
A = [1, 2; 3, 4]
% inverse of matrix A
inv(A)

ans =

   -2      1
   1.5   -0.5
```

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$$

$$A^{-1} = \frac{1}{|A|} \operatorname{adj}(A)$$

$$= \frac{1}{(1)(4)-(2)(3)} \begin{bmatrix} 4 & -2 \\ -3 & 1 \end{bmatrix}$$

$$= -\frac{1}{2} \begin{bmatrix} 4 & -2 \\ -3 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} -2 & 1 \\ 1.5 & -0.5 \end{bmatrix}$$

Matrix Multiplication

A = [1, 2 ; 3, 4]

B = [1, 2 ; 3, 4]

A * B

ans =

7 10

15 22

$$A * B = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} = \begin{bmatrix} 7 & 10 \\ 15 & 22 \end{bmatrix}$$

Pseudo-inverse (When Matrix is not square)

For pseudo-inverse, use pinv

Recall, $\text{pinv}(A) = (A^H A)^{-1} A^H$

A = [1, 2 ; 3, 4 ; 5, 6]

pinv (A)

ans =

-1.3333 -0.3333 0.6667

1.0833 0.3333 -0.4167

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix}$$

$$A^H A = \begin{bmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \end{bmatrix} \begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix} = \begin{bmatrix} 35 & 44 \\ 44 & 56 \end{bmatrix}$$

$$(A^H A)^{-1} = \frac{1}{(35)(56) - (44)(44)} \begin{bmatrix} 56 & -44 \\ -44 & 35 \end{bmatrix} = \frac{1}{24} \begin{bmatrix} 56 & -44 \\ -44 & 35 \end{bmatrix}$$

$$(A^H A)^{-1} A^H = \frac{1}{24} \begin{bmatrix} 56 & -44 \\ -44 & 35 \end{bmatrix} \begin{bmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \end{bmatrix} = \frac{1}{24} \begin{bmatrix} -32 & -8 & 16 \\ 26 & 8 & -10 \end{bmatrix}$$

Plotting

x = [1 : 10]

plot (x, x.^2, 'b-s')

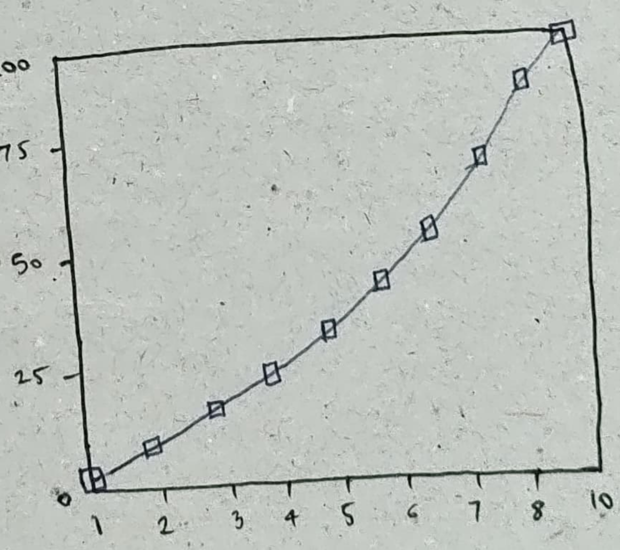
blue color

solid line

Squares □

x-axis

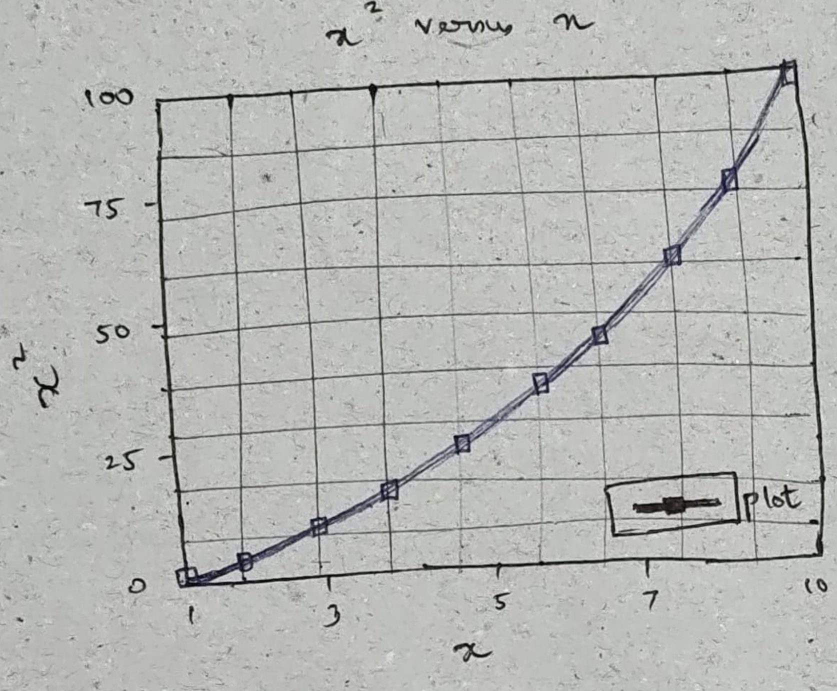
y-axis



MATLAB Plot

Plot with grid, title, legend, x-axis label, y-axis label

```
x = [1 : 10]
plot (x, x.^2, 'b-s', 'LineWidth', 3.0);
grid on;
title ('x^2 versus x')
legend ('plot', 'Location', 'South East')
xlabel ('x')
ylabel ('x^2')
```



Plot the PDF of a Standard Gaussian RV in the range [-5, 5].

We have,

Gaussian PDF, $f_X(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$

Standard Gaussian distribution (Mean = 0, variance = 1)

$\Rightarrow f_X(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}$

```
x = -5 : 0.2 : 5 ;
f_x = (1 / sqrt (2*pi)) * exp (-x.^2 / 2) ;
plot (x, f_x, 'b-s', 'Linewidth', 1.0) ;
grid on ;
title ('Gaussian pdf')
legend ('Gaussian pdf', 'Location', 'South East')
x label ('x')
y label ('f_x')
```