Introduction to Matlab

Jingjing Huang

September 11, 2012

1 Basic Ideas

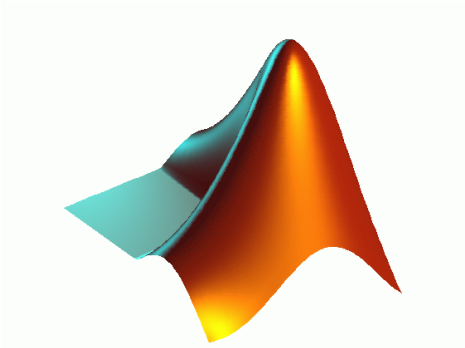
2 Matrix Operations

3 Control Flows

4 Define Functions

5 MATLAB Basic Graphics

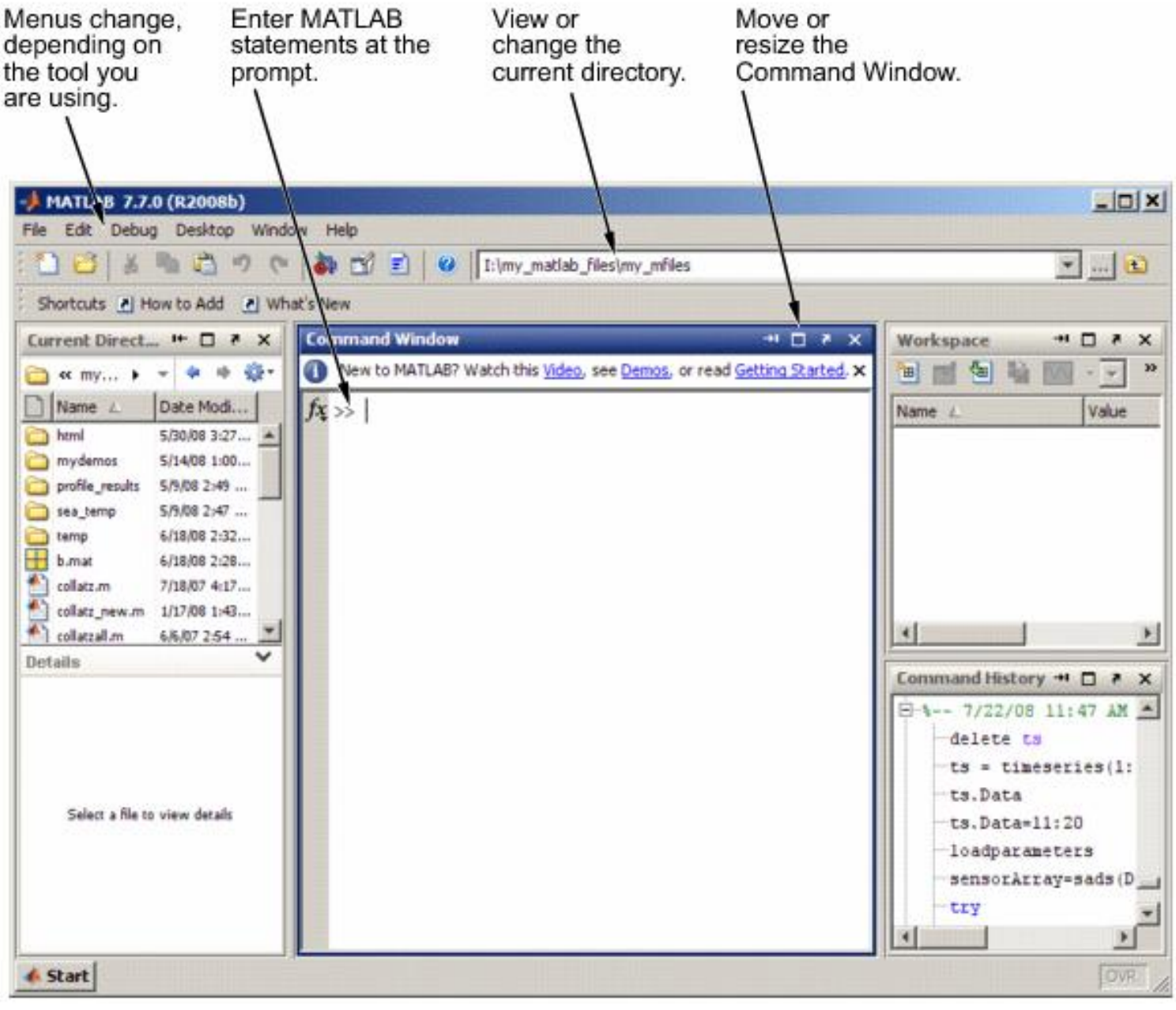
What is MATLAB ?



- MATLAB (MATrix LABoratory) is a numerical computing environment developed by MathWorks
- **Matrix manipulations**
- Plotting of functions and data
- Implementation of algorithms
- Creation of user interfaces
- Interfacing with programs written in other languages

Using MATLAB

- Install MATLAB in own computer
- Using MATLAB in department servers
  - mills, moore $\cdots$
  - `> ssh username@mills.mcmaster.ca -X`
  - `> matlab`


Menus change, depending on the tool you are using.
Enter MATLAB statements at the prompt.
View or change the current directory.
Move or resize the Command Window.
MATLAB 7.7.0 (R2008b)
File Edit Debug Desktop Window Help
I:\my_matlab_files\my_mfiles
Shortcuts How to Add What's New
Current Direct...
Name Date Modi...
html 5/30/08 3:27...
mydemos 5/14/08 1:00...
profile_results 5/9/08 2:49 ...
sea_temp 5/9/08 2:47 ...
temp 6/18/08 2:32...
b.mat 6/18/08 2:28...
collatz.m 7/18/07 4:17...
collatz_new.m 1/17/08 1:43...
collatzall.m 6/6/07 2:54 ...
Details
Select a file to view details
Command Window
New to MATLAB? Watch this Video, see Demos, or read Getting Started.
>>
Workspace
Name Value
Command History
%-- 7/22/08 11:47 AM
delete ts
ts = timeseries(1:
ts.Data
ts.Data=11:20
loadparameters
sensorArray=sads(D
try
Start
OVR

Implement Matrices Multiplication $A \times B$

$$\begin{bmatrix} 2 & 5 & 6 \\ 7 & 8 & 4 \end{bmatrix} \times \begin{bmatrix} 2 & 3 \\ 5 & -4 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} (2\times2 + 5\times5 + 6\times1) & (2\times3 + 5\times(-4) + 6\times0) \\ (7\times2 + 8\times5 + 4\times1) & (7\times3 + 8\times(-4) + 4\times0) \end{bmatrix}$$

$$= \begin{bmatrix} 35 & -14 \\ 58 & -11 \end{bmatrix}$$

- How ?

Define Matrix and Vectors

- Define a vector
  - >> a = [1 2 3 4 5]
  - >> a = [1,2,3,4,5]
  - >> a = 1:5
  - >> a = 5:-1:1

```
>> A = 1:5

A =

   1   2   3   4   5

>> A = 5:-1:1

A =

   5   4   3   2   1
```

Define Matrix and Vectors

- Define a matrix
  - `>> A = [1 2 3 4 5; 6 7 8 9 0]`
  - `>> A = [1:5; 6:10]`
  - `>> A = [5:-1:1; 6:10]`

```
>> A = [1:5; 6:10]

A =

   1   2   3   4   5
   6   7   8   9  10

>> A = [5:-1:1; 6:10]

A =

   5   4   3   2   1
   6   7   8   9  10
```

Matrix Operations

- Matrix operations
  - >> `A + B`
  - >> `A - B`
  - >> `A * B`
  - >> `A / B`
  - >> `transpose(A), A'`
  - >> `det(A), inv(A)`

Implement Matrices Multiplication $A \times B$

$$\begin{bmatrix} 2 & 5 & 6 \\ 7 & 8 & 4 \end{bmatrix} \times \begin{bmatrix} 2 & 3 \\ 5 & -4 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} (2\times2+5\times5+6\times1) & (2\times3+5\times(-4)+6\times0) \\ (7\times2+8\times5+4\times1) & (7\times3+8\times(-4)+4\times0) \end{bmatrix}$$

$$= \begin{bmatrix} 35 & -14 \\ 58 & -11 \end{bmatrix}$$

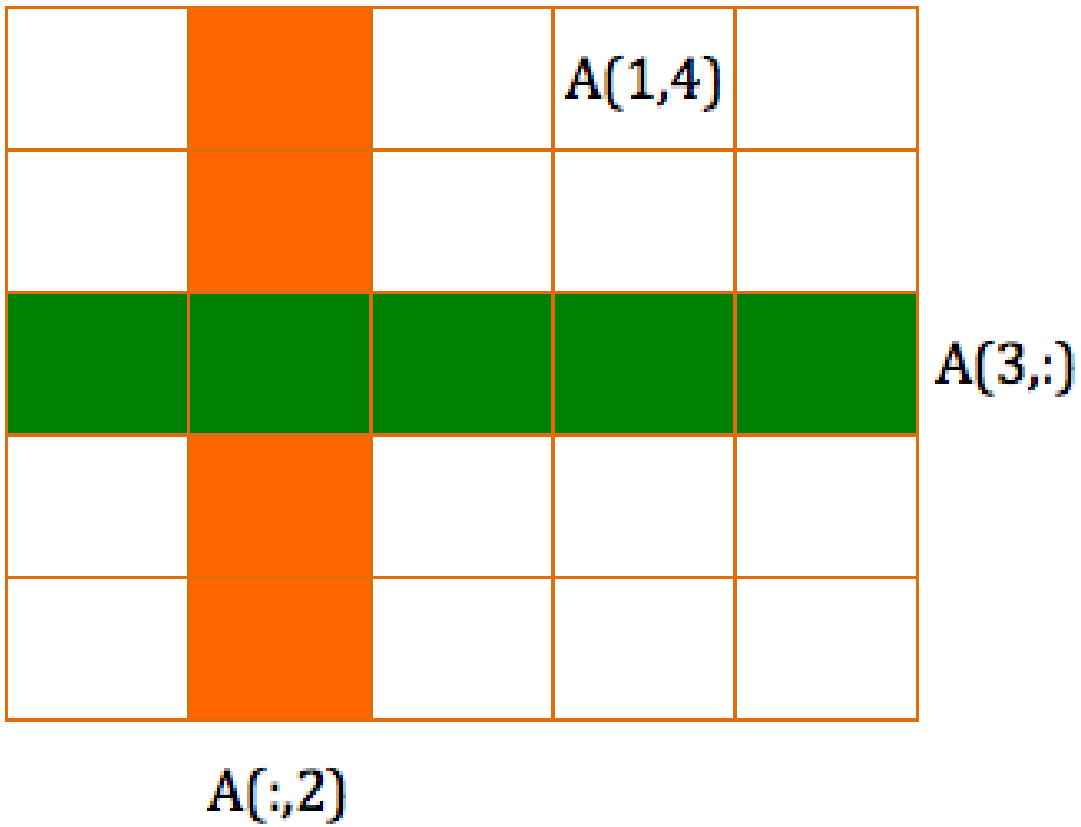
- How ?
- Method 1: >> `A * B`

Elementary Matrices

- `>> A = eye(5)` % Identity matrix
- `>> d = diag(A)` % Column Vector, Diagonals of A
- `>> A = diag(d)` % Diagonal matrix
- `>> A = ones(5)`
- `>> a = ones(1,5)` % Matrix of all ones
- `>> A = zeros(5)`
- `>> a = zeros(1,5)` % Matrix of all 0s
- `>> A = rand(5)`
- `>> a = rand(1,5)` % Matrix filled with uniformly distributed pseudorandom numbers between 0.0 and 1.0

Using Matrix Elements

- >> `A(1,4)`
- >> `A(:,2)`
- >> `A(3,:)`



Implement Matrices Multiplication $A \times B$

$$\begin{bmatrix} 2 & 5 & 6 \\ 7 & 8 & 4 \end{bmatrix} \times \begin{bmatrix} 2 & 3 \\ 5 & -4 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} (2\times2+5\times5+6\times1) & (2\times3+5\times(-4)+6\times0) \\ (7\times2+8\times5+4\times1) & (7\times3+8\times(-4)+4\times0) \end{bmatrix}$$

$$= \begin{bmatrix} 35 & -14 \\ 58 & -11 \end{bmatrix}$$

- Method 1: `A * B`
- Method 2: Using matrices elements
  - `C = zeros(2,2)`
  - `C(1,1)=A(1,1)*B(1,1)+A(1,2)*B(2,1)+A(1,3)*B(3,1)`
  - `C(1,2)= ···`
  - `C(2,1)= ···`
  - `C(2,2)= ···`

Implement Matrices Multiplication $A \times B$

- Method 1: `A * B`
- Method 2: Using matrices elements
  - `C = zeros(2,2)`
  - `C(1,1)=A(1,1)*B(1,1)+A(1,2)*B(2,1)+A(1,3)*B(3,1)`
  - `C(1,2)= ···`
  - `···`
- Method 3: Using vector operations
  - `C = zeros(2,2)`
  - `C(1,1) = A(1,:)*B(:,1)`
  - `C(1,2) = ···`
  - `···`

Using Control Flows

- `if` statement
  - `x = 1; y = 2;`
  - `if x == y`
  - `  disp('x == y');`
  - `elseif x == y + 1`
  - `  disp('x == y + 1');`
  - `elseif x == y - 1`
  - `  disp('x == y - 1');`
  - `else`
  - `  disp('No relation between x and y.');`

Using Control Flows

- `for` loop
  - `for index = values`
  - `  program statements;`
  - `  :`
  - `end`
- `for` loop index values
  - `for i = 1:10`
  - `for i = 10:-1:0  % Step by increments of -1`
  - `for i = [1,5,8,17]  % Defined set of index values`
  - `for i = eye(5)  % Successively set i to unit vectors`

Using Control Flows

- `for` loop
  - `for index = values`
  - `  program statements;`
  - `  :`
  - `end`
- `for` loop index values
  - `for i = 1:10`
  - `for i = 10:-1:0` % Step by increments of -1
  - `for i = [1,5,8,17]` % Defined set of index values
  - `for i = eye(5)` % Successively set i to unit vectors

Using Control Flows

- `while` loop
- `continue` statement
- `break` statement
- `switch... case...` statement
- `try... catch...` statement
- `return` statement

- Use `help` command for more information
- `>> help while`

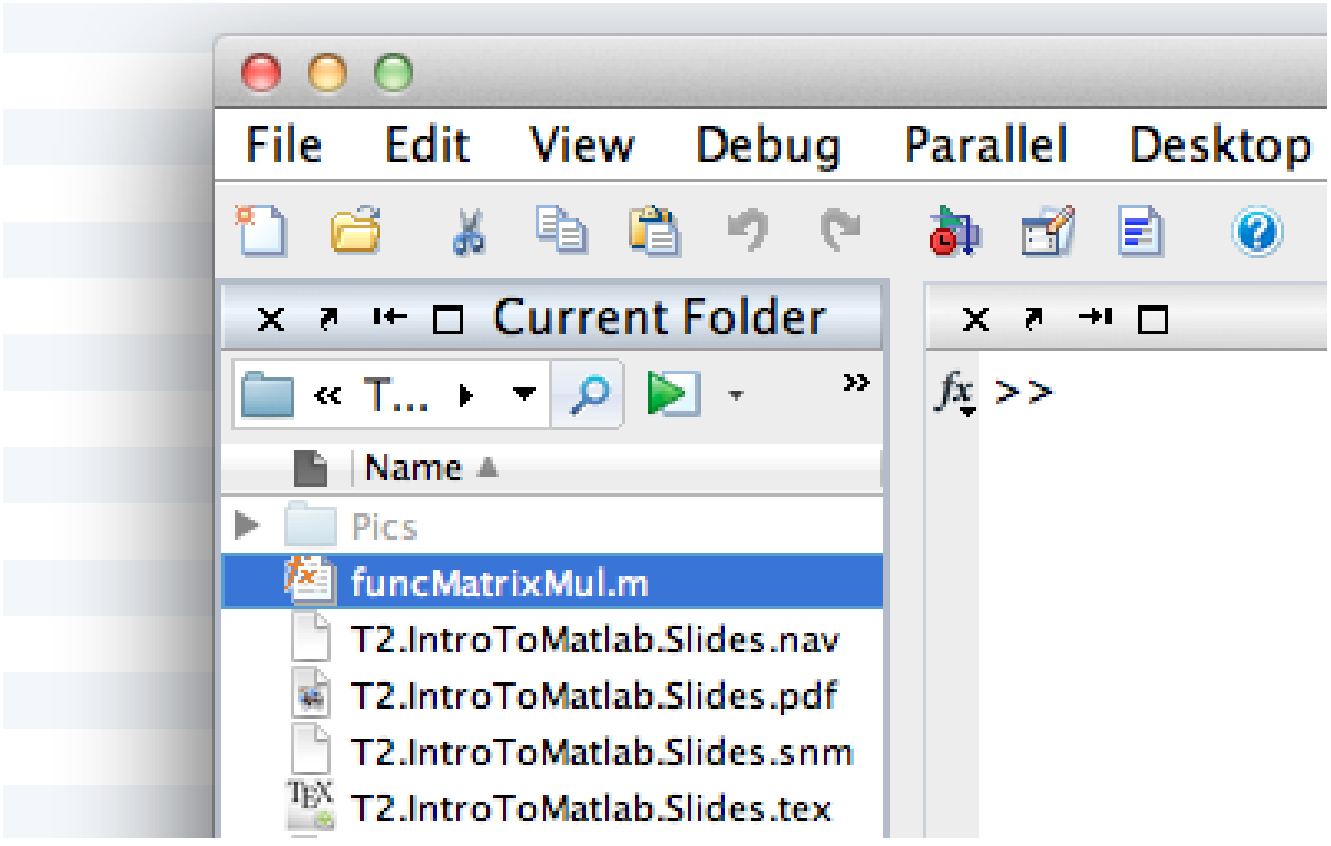
Implement Matrices Multiplication $A \times B$

- Method 1: `A * B`
- Method 2: Using matrices elements
  - `C(1,1)=A(1,1)*B(1,1)+A(1,2)*B(2,1)+A(1,3)*B(3,1)`
- Method 3: Using vector operations
  - `C(1,1) = A(1,:)*B(:,1)`
- Method 4: Using for loop
  - `C = zeros(2,2)`
  - `for i = 1:2`
  - `  for j = 1:2`
  - `    C(i,j) = A(i,:)*B(:,j);`
  - `  end`
  - `end`

Define a Matlab Function

```
1  function rMM = funcMatrixMul(p_A, p_B)
2  % Calculate the matrix multiplication of input matrix A, B
3      [pR, pACol] = size(p_A);
4      [pBRow, pC] = size(p_B);
5      rMM = zeros(pR, pC);
6      if  funcCheckMat(pACol, pBRow) == false
7        return;
8      end
9      for i = 1:pR
10       for j = 1:pC
11         rMM(i,j) = p_A(i,:) * p_B(:,j);
12       end
13     end % End of for loop.
14   function rCheck = funcCheckMat(mR, mC)
15     rCheck = false;
16     if mR == mC
17       rCheck = true;
18     else
19       disp('The two input matrices do not match.');
20     end
21   end % End of funcCheckMat(*,*)
```

Define a Matlab Function



- Save to CURRENT directory
- File name `funcMatrixMul.m`

Define a Matlab Function

```
>> C = funcMatrixMul(A, B)

C =

    0.9018    1.4199
    0.8452    1.1126

>> help funcMatrixMul
  Calculate the matrix multiplication of input matrix A, B

fx >>
```

- Can be invoked as other MATLAB functions
- Use `help` for more information

Define a Matlab Function

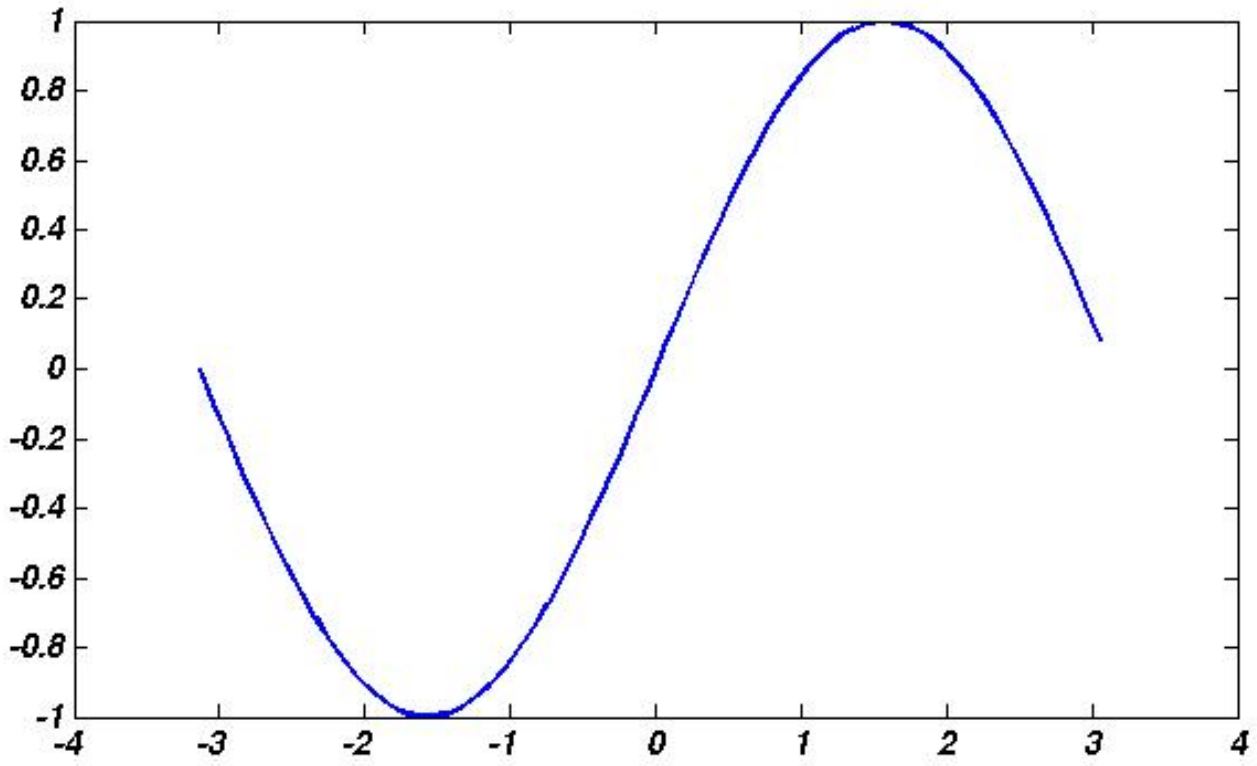
- M-files
- Plain text file
- Has same func. name can be called from outside of file
- Other functions are only visible inside the file
- Accessible in the current directory or MATLAB's search path
- Input / Output

Implement Matrices Multiplication $A \times B$

- Method 1: `A * B`
- Method 2: Using matrices elements
  - `C(1,1)=A(1,1)*B(1,1)+A(1,2)*B(2,1)+A(1,3)*B(3,1)`
- Method 3: Using vector operations
  - `C(1,1) = A(1,:)*B(:,1)`
- Method 4: Using for loop
  - `C = zeros(2,2)`
  - `for i = 1:2`
  - `  :`
  - `end`
- Method 5: Define a MATLAB fucntion

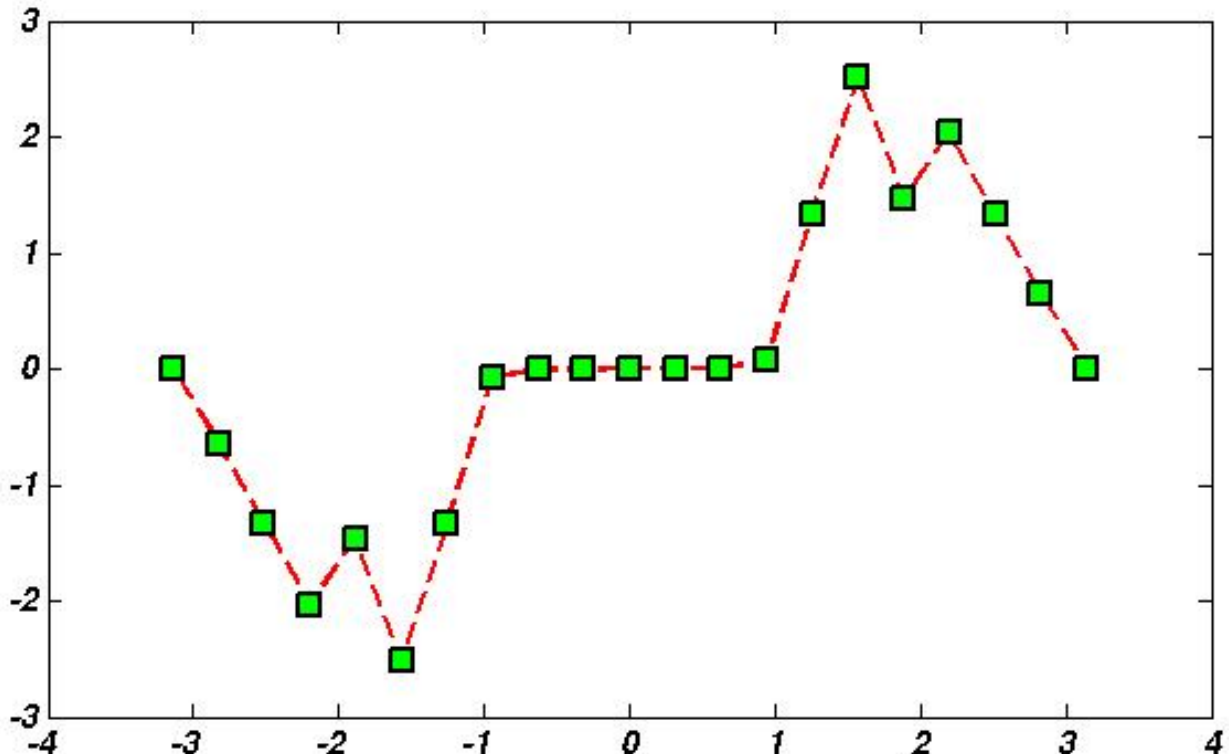
2-D line plot

- `x = -pi:.1:pi;`
- `y = sin(x);`
- `plot(x,y)`



2-D line plot

- `x = -pi:pi/10:pi;`
- `y = tan(sin(x)) - sin(tan(x));`
- `plot(x,y,'--rs','LineWidth',2,...`
  `'MarkerEdgeColor','k',...`
  `'MarkerFaceColor','g',... 'MarkerSize',10)`



Thanks!