Department of Electrical Engineering
University of Arkansas

A Tutorial on Matlab

Ch. 3 Programming in Matlab

Dr. Jingxian Wu

wuj@uark.edu

OUTLINE

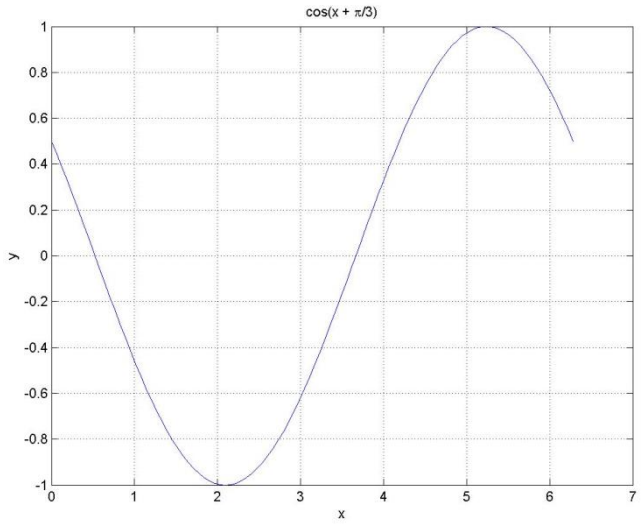
- Plotting
- M-file Scripts
- Functions
- Control Flows
- Exercises

PLOTTING

- **Simple plotting**

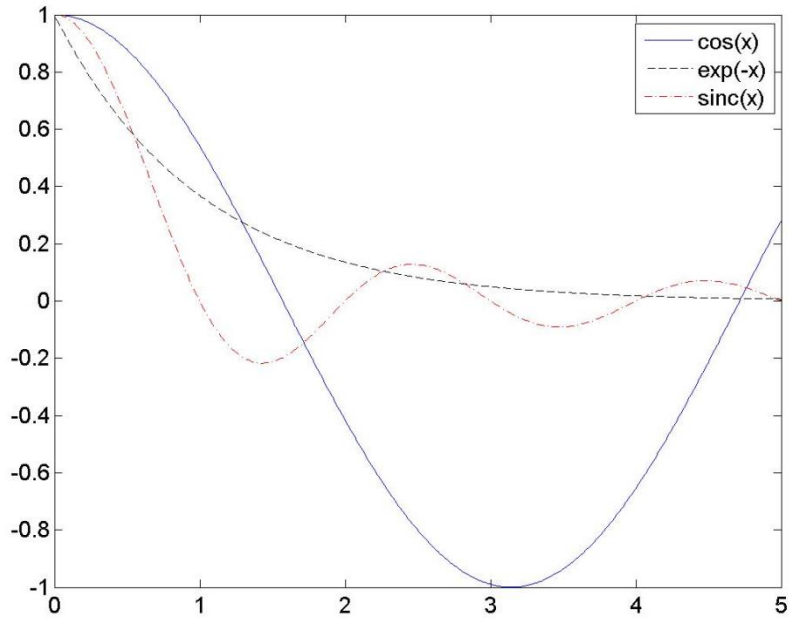
```
>> x = [1 2 3 4 5];
>> y = [3 -1 2 -3 -4]; % vector x and y must be of the same length
>> plot(x, y)
```

```
>> x = linspace(0, 2*pi, 100);
>> y = cos(x + pi/3);
>> plot(x, y)
>> xlabel('x');   % add a label to the x-axis
>> ylabel('y');  % add a label to the y-axis
>> title('cos(x + \pi/3)');  % add a title to the figure
>> grid on;   % turn on the grid
>> print –djpeg fig.jpg; % save the figure to a file.
```



PLOTTING

- **Multiple curves in one figure**
  - Use the “hold on” command




```
>> x = [0:0.01:5];
>> y1 = cos(x);
>> y2 = exp(-x);
>> y3 = sinc(x);
>> plot(x, y1);
>> hold on;
>> plot(x, y2, 'k--'); % black, dashed line
>> plot(x, y3, 'r-.'); % red, dash-dot line
>> legend('cos(x)', 'exp(-x)', 'sinc(x)'); % adding legends
>> set(gca, 'fontsize', 14);  % change the fontsize
```

Use “help plot” to find out more line styles and colors

PLOTTING

- **Rescale a figure**
  - Use the axis command

```
>> x = [-20:0.01:20];
>> plot(x, sinc(x))
>> axis([-5, 5 -0.3 1]); % x range: [-0.5, 0.5], y-range: [-0.3, 1]
```

- **Close one or more figures**
  - figure(1); % make figure 1 the current figure
  - close; % close the current figure
  - close all; % close all figures

OUTLINE

- Plotting
- **M-file Scripts**
- Functions
- Control Flows
- Exercises

M-FILE SCRIPTS

- **Script file:**
  - We can write a sequence of Matlab commands in an external file with extension *.m
    - The file is usually called an m-file
  - Execute the m-file file will execute all commands in the file

M-FILE SCRIPTS

- **Create a script file**
  - 1. In Matlab, click on File → New → Script, to load the default editor
  - 2. In the editor, type the commands:

```
x = [0:0.01:10];
y1 = cos(x);
y2 = exp(-x);
plot(x, y1);
hold on;
plot(x, y1+y2, 'r--');
```

  - 3. Save the file. In the editor window, click on File → Save → test.m
  - 4. Execute the file. In the editor window, click on Debug → Run test.m

M-FILE SCRIPTS

- **Execute the file**
  - Switch to the command window
  - At the command prompt, type

```
>> test          % note that .m is not included in the command
```

  - Important: the m-file to be executed must be in the current working directory

```
>> pwd           % display the current working directory
ans =
c:\skydrive\teaching\ELEG3124\Matlab
>> ls *.m        % list all m-files in the current working directory
test.m
```

OUTLINE

- Plotting
- M-file Scripts
- **Functions**
- Control Flows
- Exercises

FUNCTIONS

- **Function**
  - A function can be defined and saved in a separate m-file
  - Example

```
function y = average(x)
% function y = average(x)
% compute the average of a vector x, and return the value to y

N_element = length(x);
y = sum(x)/N_element;
```

  - Must start with the keyword: function
  - Save to an m-file: in the editor window, File → Save → average.m.
    - The name of the m-file must be the function name.
  - The comments after the function title will be displayed when you type "help average" in the command line

FUNCTIONS

- **Call the function**
  - In the command window

```
>> x = 1:10;
>> y = average(x)

>> z = sqrt(x);
>> average(z)
```

FUNCTION

- **Function with multiple inputs and/or outputs**

```
function [addition, difference] = total_diff(x,
y)
% function [total, difference] = total_diff(x, y)
% find the sum and difference between two vectors

addition = x + y;
difference = x - y;
```

OUTLINE

- Plotting
- M-file Scripts
- Functions
- **Control Flows**
- Exercises

CONTROL FLOWS

- **Matlab control flows:**
  - 1. if … elseif … else … end
  - 2. for … end
  - 3. while … end

CONTROL FLOWS

- **if … end**

```
x = 10;
y = sqrt(x)-x/3;
if y < 0
    'y is less than 0'
    y = y + 1;
end
```

CONTROL FLOWS

- **if … else … end**

```
x = 10;
y = sqrt(x)-x/3+1;
if y < 0
    'y is less than 0'
    y = y + 1;
else
    'y is greater than or equal to 0'
    y = y - 1;
end
```

CONTROL FLOWS

- **if … elseif … else … end**

```
y = 0;
if y < 0
    'y is less than 0'
    y = y + 1;
elseif y == 0
    'y is 0'
else
    'y is greater than 0'
    y = y - 1;
end
```

CONTROL FLOWS

- **for … end**

```
for mm = 1:2:10
    y(mm) = mm^2;
end
```

```
% or equivalently:
% this is more efficient
y = [1:2:10].^2;
```

CONTROL FLOWS

- **Double loops**

```
A = [1 3 2; 4 -1 0];
[n_row, n_col] = size(A);

for mm = 1:n_row
    row_avg(mm) = mean(A(mm, :));
    for nn = 1:n_col
        B(mm, nn) = A(mm, nn).^2;
    end
end
```

```
% Alternatively and more efficiently
row_avg = mean(A, 2); % calculate the row average
B = A.^2;
```

CONTROL FLOWS

- **while … end**

```
x = 1;

while x <= 20
    x = 3*x+2;
end
```

OUTLINE

- Plotting
- M-file Scripts
- Functions
- Control Flows
- **Exercises**

EXERCISES

- 1. Plot the real part, imaginary part, and amplitude of the following function in the same figure. Use xlabel, ylable, legend, grid, and different color and line styles for different curves. Use a font size of 14

$s(t) = e^{-t}e^{j2\pi t}$, $0 \le t \le 10$

- 2. rescale the above curve for x between 0 and 5, and y between -2 and 2

- 3. Write a function to calculate the standard deviation of a vector, where x_i is the i-th element of the vector, $\bar{x}$ is the average of the vector

$$y = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

  - Use the above function, evaluate the standard deviation of the following vector

    x = [0, 3, 1, 5, -2, -4, 3, 2, -5]

EXERCISES

- 4. Write a Matlab function to generate the first n Fibonacci number, where n is the function input. The Fibonacci number is defined as

$$F_1 = 1$$
$$F_2 = 1$$
$$F_n = F_{n-1} + F_{n-2}$$

  - Generate the first 10 Fibonacci number
- 5. Write a function, which takes a vector x as input. The output y is a vector of the same size, defined as follows

$$y_n = \begin{cases} \sqrt{x_n}, & x_n > 0 \\ 0, & x_n == 0 \\ -x_n^2, & x_n < 0 \end{cases}$$

  - Test your function with x = -3:3
- 6. Consider a sequnce $x_1 = 0, x_n = 2x_{n-1}^{3/2} + 1$

Use while loop to find out what is the smallest n such that $x_n > 100$