

# MATLAB Mini-Tutorial

Tobias M. Schneider<sup>1</sup>

Harvard SEAS

AM105b  
Jan, 2011

---

<sup>1</sup>based on notes by Shreyas Mandre, Brown

# MATLAB - Getting started

## Access

- Available through:  
`http://downloads.fas.harvard.edu/download`
- Requires HUID and PIN,
- Link posted on the supplements page of the course website.

## Installation

- Just double-click on the file you download and follow instructions.

## What you see when you open MATLAB...

- Command Window
- Current Directory
- Command History

## A fun way to learn MATLAB

- Demos at  
`http://www.mathworks.com/products/matlab/demos.html`

## Basic operations

### A grand calculator

- Compute basic arithmetic expressions,
- Operators: +, -, \*, / and ^
- Variables: e.g. to calculate the hypotenuse of a triangle with sides 5 and 10,

```
>> x = 5
```

```
x =
```

```
5
```

```
>> y = 10;
```

```
>> r = sqrt(x ^ 2 + y ^ 2) % prints r = 11.1803
```

- Includes trigonometric, exponential, hyperbolic, inverse, etc. functions `sin(x)`, `cos(x)`, `tan(x)`, `sinh(x)`, `cosh(x)`, `tanh(x)`, `exp(x)`, `log(x)`, ...
- Note, semicolon ";" suppresses output of the line.

# Everything is a complex matrix

## Elementary matrix manipulation

- Creating a row vector:  
`r = [1 2 3];` or  
`r = [1, 2, 3];`
- ... a column vector:  
`c = [1; 2; 3];`
- .. a matrix:  
`A = [1 2 3; 4 5 6;];`
- Accessing elements:  
`A(2,1)`  
`A(2,1) = 7;`
- Transpose: `B = A'`
- The `:` operator  
`1:10`  
`0:5:50`

## Linear algebra

- Element-wise operations: `.*`, `./`, `.^`
- Matrix operators: `+`, `-`, `*`, `/`, `^`
- Solving equations (the magical `\` operator):

$$x + y + z = 4,$$

$$2x + 3y + z = 7,$$

$$y + z = 3.$$

$$\Leftrightarrow Ax = b \quad \text{with}$$

$$A = [1 \ 1 \ 1; 2 \ 3 \ 1; 0 \ 1 \ 1];$$

$$b = [4 \ 7 \ 3];$$

Solve for x:

$$x = A \setminus b;$$

## Documentation: No need to remember everything!

- Online documentation:

`http://www.mathworks.com/help/techdoc/index.html`

or

Search “MATLAB help” on Google.

- MATLAB help:

Help - Product Help

or

`>> doc`

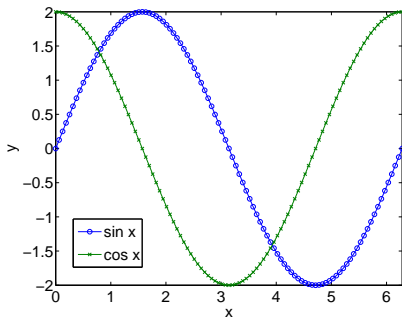
Information about a specific command:

`>> help command`

## Plotting

The plot command

```
>> x = linspace(0, 2*pi, 101);  
>> y = 2*sin(x);  
>> plot(x, y, 'o-');  
>> y2 = 2*cos(x);  
>> hold on;  
>> plot(x, y2, 'x-');  
>> xlabel('x');  
>> ylabel('y');  
>> axis([0 2*pi -2 2]);  
>> legend('sin x', 'cos x');
```



# Scripts: Saving commands

## Scripting

- Save all commands in sequential order in the file 'funkyname.m'
- Running the command  
`>> funkyname`  
will run the commands in the file.

## Notes

- Script names are case-sensitive.
- Do not choose filenames to be the name of a variable.
- File must be saved in the 'current directory'.
- Add comments:  
`% This line is ignored but explains what is happening.`

## MATLAB functions

- A function relates “arguments” to “return values”.  
e.g. `>> y = sin(x)`  
Here `x` = argument of `sin`  
and `y` = return value of `sin`.
- Function files:

Celsius to Fahrenheit (`c2f.m`)

```
function fahrenheit = c2f(celsius)
fahrenheit = 32 + 9*celsius/5;
```

Hypotenuse of a triangle (`myhypot.m`)

```
function [h, t] = myhypot(a, b)
h = sqrt(a.^2 + b.^2);
t = acos(a/h);
```



## Programs: Control structures

### Loops

```
for x=1:10
    disp(x)
end
```

### If statement

```
for x=1:10
    if (x*x==9)
        disp(x)
    end
end
```

### Notes:

- No semicolon after `for`, `if`, `end`,....
- Other constructs: while loops, if elseif else,....

## Function handles – using ode45

### Problem definition

Solve ordinary differential equation

$$y'(x) = y, \quad y(0) = 1.$$

### Matlab code

Function `myode.m`:

```
function dy = myode(x, y)
    dy = zeros(1,1);
    dy = y;
```

Script `solveode.m`:

```
xspan = [0 5];
y0 = 1;
[x, y] = ode45(@myode, xspan, y0);
plot(x, y, 'ro', x, exp(x));
```

