

Introduction

- What is MATLAB?
 - MATLAB (MATrix LABoratory) is an interactive program for scientific and engineering **numeric** calculation. Applications include:
 - matrix manipulation
 - finding the roots of polynomials
 - digital signal processing
 - x-y and polar plotting
 - 3-dimensional graphics
 - Combines:
 - The mathematics of linear algebra
 - C++ programming environment
 - The UNIX command shell
 - High-level functions
 - This combination makes MATLAB ideal for signal processing applications.

Getting Started

- There are a couple of different ways to start MATLAB on Athena:
 - From *Dash*
 - *Numerical/Math//Analysis and Plotting//MATLAB*
 - To use software designed for 6.003
Courseware//EECS//6.003//6.003MATLAB
 - From *Athena prompt*
 - `athena% add matlab`
`athena% matlab`
 - `athena% add 6.003`
`athena% ~6.003/startup`
- Getting Help
 - MATLAB on Athena
<http://web.mit.edu/matlab/www/>
 - Mathworks website
<http://www.mathworks.com/access/helpdesk/help/techdoc/matlab.shtml>
 - 6.003 MATLAB pico-course
 - Run *picocrse.m* script in ~6.003/picocouse directory

MATLAB Navigation

- Getting help from within MATLAB

```
>> help <functionname>
```

Shows help document for a give function

```
>> lookfor <keyword>
```

Searches all the help documents for a given keyword

```
>> demo
```
- Navigation within MATLAB is done using regular UNIX commands

```
>> cd (change directory)
>> pwd (show the path)
>> ls (list contents of directory)
>> !<unix command> (access the UNIX shell)
```
- Useful MATLAB Commands

```
>> path <directory>
>> what (lists MATLAB specific files)
>> info (gives information about toolboxes)
```

Variables

Real Scalars

```
>> x = 5
x = 5
```

Complex Scalars

```
>> x = 5+10j %5+10i works, but not 5+10*j
x =
  5.0000 +10.0000i
```

Row Vector (1 x 3)

```
>> x = [1 2 3]
x =
     1     2     3
```

Column Vector (3 x 1)

```
>> x = [1;2;3]; %";" suppresses output
>> x
x =
     1
     2
     3
```

Matrix (3 x 3)

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

Note: Variable Names are case sensitive

Matrices & Vectors

Generating Vectors (Useful for time axis)

```
>> x = [0:0.2:1] %0 to 1 in incr. Of 0.2
x =
    0    0.20    0.40    0.60    0.80    1.00
0 to 1 in increments of 0.2

>> x = linspace(0,1,6)
x =
    0    0.20    0.40    0.60    0.80    1.00
6 points from 0 to 1 on a linear scale

>> x = logspace(0,1,6)
x =
    1.0000    1.5849    2.5119    3.9811
    6.3096    10.0000
6 points from 100 to 101 on a log scale
```

Accessing Matrix Elements

```
>> A = [ 1 2 3; 4 5 6; 7 8 9];
>> x = A(1,3) %A(<row>,<column>)
x =
    3

>> y = A(2,:)
y =
    4    5    6

>> z = A(1:2,1:3)
z =
    1    2    3
    4    5    6
```

Matrix Operations

Complex Number Operations

```
>> x = 3+4j
>> abs(x) %Absolute value.
x = 5
>> angle(x) %Phase angle.
x = 0.9273
>> conj(x) %Complex conjugate.
x = 3-4j
>> imag(x) %Complex imaginary part.
x = 4
>> real(x) %Complex real part.
x = 3
```

Other Matrix operations

Math Functions

```
sin(x), cos(x), tan(x), atan(x), exp(x),
log(x), log10(x), sqrt(x)
```

Operators

```
Usual operators +,-,*,^,
>> M = A' %Conjugate transpose of matrix
>> M = A.' %Unconjugated transpose
>> y = A\b %left division is soln to A*y=b
>> y = b/A %right division is soln to y*A=b
```

Element-by-Element Operators (.*,.^,./)

```
>> A = [ 1 2; 3 4]
A =
    1    2
    3    4

>> B = A.*A
B =
    7    10
   15    22

>> C = A.^A
C =
    1    4
    9   16
```

Programming in MATLAB

• M-Files

- Sets of MATLAB commands can be executed via scripts
- Scripts are written into files with extensions *.m
Ex. filename.m
- These scripts are executed in MATLAB by entering the name of .m file
>> filename

• Functions

- Commonly performed operations can be written into functions
- In a file named functionname.m
Function [output]=functionname(input)
Command 1
Command 2

• Flow Control operations and operators

- Similar to C can use for, while, if, do statements with &, |, ~ operators
- Also have <, <=, >, >=, ==, ~= operators available for programming

Plotting and Output

• Simple plotting commands

```
>> plot(t,y) %Plot Continuous Function
>> stem(y) %Plot Discrete samples
>> loglog(f,Y) %Log on x and y
>> semilogx(f,Y) %Log x and linear y
>> xlabel('time')
>> ylabel('Voltage')
>> title('Voltage vs. Time')
>> Axis([xmin xmax ymin ymax])
>> Figure(n) %Makes figure n current
>> subplot(r,c,n)
%Creates r x c sub-figures, and n is
used to reference the sub-figures
```

• Printing

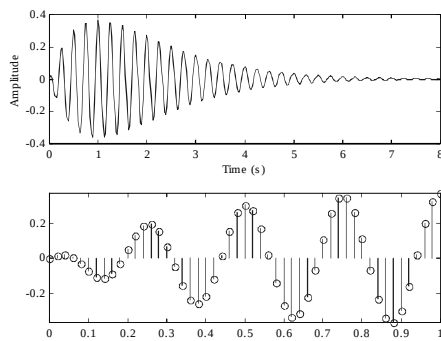
```
>> print -dps filename.ps
%output current figure to ps file
>> print -Pprintername
%outputs to printer
```

• Saving variables

```
>> save filename.mat <variable names>
% Saves variables in binary form
>> load filename
% Loads binary data or ascii matrix data
```

Plotting Example

```
>> t = linspace(0,8, 401);
>> x = t.*exp(-t).*cos(2*pi*4*t);
>> figure(1)
>> subplot(2,1,1)
>> plot(t,x);
>> xlabel('Time (s)');
>> ylabel('Amplitude')
>> subplot(2,1,2)
>> stem(t,x);
>> axis([0 1 min(x) max(x)])
```



Other Examples

- Polynomials factorization

Find the roots of the following expression

$$13x^3 + 25x^2 + 3x + 4$$

```
>> C = [13 25 3 4];
>> r = roots(C)
r =
-1.8872
-0.0179 + 0.4034i
-0.0179 - 0.4034i
```

- Partial Fractions

$$\frac{5s+3}{s^3+3s^2-4}$$

```
>> [R,P,K] = Residue([5,3],[1 3 0 -4])
```

```
R =
-0.8889
2.3333
0.8889
```

```
P =
-2.0000
-2.0000
1.0000
```

```
K =
[]
```

$$\frac{-\frac{8}{9}}{s+2} + \frac{\frac{7}{3}}{(s+2)^2} + \frac{\frac{8}{9}}{s-1}$$