Introduction to MATLAB

Violeta Ivanova, Ph.D.
Educational Technology Consultant
MIT Academic Computing

violeta@mit.edu
http://web.mit.edu/violeta/www/IAP2006

IST Information Services and Technology

Topics

- ✓ MATLAB Interface and Basics
- ✓ Linear Algebra and Calculus
- ✓ Graphics
- ➢ Programming
- ■ MATLAB Practice
- ■ Math On the Web (optional)

Class Materials

- On laptops download from:

  http://web.mit.edu/acmath/matlab/IntroMATLAB
- On Athena copy from locker acmath

```
athena% add acmath
athena% cp
   /mit/acmath/matlab/IntroMATLAB/Programming.tar .
```

Help in MATLAB

- Command line help

  `>> help` *`command`*

  e.g. `help nargin`

  `>> lookfor` *`keyword`*

  e.g. `lookfor arguments`

- Desktop menu
  - Help->Help MATLAB

MATLAB Help Browser

- MATLAB
  - + Data Types
  - + Basic Program Components
    - + Variables
    - + Operators
      - + Arithmetic Operators
      - + Relational Operators
      - + Logical Operators
    - + Program Control Statements
  - + M-File Programming
    - + M-File Scripts and Functions
  - + Functions - Categorical List

MATLAB Programming Basics

Variables and Operators

Program Control Statements

Script and Function M-Files

Information Services and Technology

Computer Languages

- Machine language (lowest level)

```
1 0 1 0 1 0 1 1 1 0 0 0 1 1 0 1
```

- Higher level languages
  - Programming languages: Java, C++
  - Scripting languages: Perl
  - Markup languages: HTML, mathML
  - etc.

What is MATLAB?

- Computational Software

  From The MathWorks: www.mathworks.com
- MATrix LABoratory
- Algorithm Development Environment

  … with some built-in abilities of a high-level programming and scripting language.

Built-In Functions

- Workspace

```
>> clear
>> who, whos
```

- Search path

```
>> path
>> addpath
```

- File operations

```
>> ls, dir
>> cd
>> copyfile
>> pwd
>> mkdir
```

Built-In Functions (continued)

- File input / output

```
>> load
>> open
>> uigetfile, uiimport
>> save, saveas
```

- Arrays and matrices

```
>> disp
>> format
>> zeros, ones
>> isempty
>> isnumeric
>> size, length
```

Built-In Functions (continued)

- Special characters

  ```
  []  ()  {}  ;  %  :  =  .  …  @
  ```

- Arithmetic operations

  ```
  +  -  /  \  ^  .\  ./  .*  .^
  ```

- Relational operations

  ```
  <  >  <=  >=  ==  ~=
  ```

- Logical operations

  ```
  |  &  ||  &&  true  false
  ```

- Built-in function variables

  ```
  nargin
  ```

Built-In Functions (continued)

- Built-in constants

```
pi  i  j  Inf
```

- Variables in memory

```
>> global
>> persistent
```

- Characters and strings

```
>> strcmp
>> lower
```

- Type conversion

```
>> num2str
>> str2num
```

Built-In Functions (continued)

- Programming flow control
  - `if, elseif, else`
  - `switch, case`
  - `for`
  - `while`
  - `end`
  - `return`
- Error handling
  - `>> error`
  - `>> warning`
- And many others
  - `>> round`

Variable Types

- Local (default)
  - Every function has its own local variables.
  - Scripts share local variables with functions they call and with the base workspace.
- Global

  ```
  global speedoflight
  ```

  - Functions, scripts, and the base workspace share global variables.
- Persistent

  ```
  persistent R, C
  ```

  - Can be declared and used only in functions.

Data Types

- Numeric

```
>> x = 5; y = 5.34; z = 0.23e+3
```

  - Default: double-precision floating point
  - Can be converted to integers, etc.
  - Numeric manipulation

```
>> y = 5.3456; x = round(y)
>> format long
```

  - Complex numbers

```
>> x = 5 + 6i
```

Data Types (continued)

- Characters and strings

```
>> a = '5'
>> b = 'Hello'
```

  - String conversions

```
>> x = 5;        a = num2str(x)
>> a = '5';      x = str2num(a)
```

  - String manipulations

```
>> isempty(b)
>> strcmp(b, 'hi there')
>> abc = lower('ABC')
```

Data Types (continued)

- Keywords

  `if, switch, for, end, global, for, …`

  DO NOT USE AS VARIABLE NAMES!
- Special Values

  `pi, i, j, …`
- Structures

  `person.name = 'Jane'; person.age = 20`
- Cell Arrays

  `person = {'Jane' 'female'; 20  1996}`

Operators

- Arithmetic: `x+y;` `A*B;` `X.*Y;` etc.
- Logical
  - Element-wise AND: `a & b`
  - Element-wise OR: `a | b`
  - “Short cuts”: `||` and `&&`
- Relational

  `a == 5; a >= b; b ~= 6;`
- Operator precedence

  `() {} []` -> Arithmetic -> Relational -> Logical

Program Flow Control: for

```
x = [1 : 0.01 : 10]; a = 60; b = 30;
N = length(x);
y = zeros(1, N);
for n = 1 : N
  y(n) = a - b*cos(pi/3 + x(n)*pi/6)
end
P = plot (x, y, 'ro')
```

Program Flow Control: `if`

```
if strcmp(planet, 'Earth')
        R = 6376; g0 = 9.814;
elseif strcmp(planet, 'Mars')
        R = 3396; g0 = 3.688;
else
        R  = input('Enter R: ');
        g0 = input('Enter g0: ');
end
```

Program Flow Control: switch

```
switch units
   case ‘metric’
        R = 6376; g0 = 9.814;
   case ‘English’
        R = 3963; g0 = 32.2;
   otherwise
        error(‘Unknown units.’)
end
```

File Input / Output

- Commands `load` and `save`

```
data = load('datain.txt', '-ascii')
save('dataout.txt', 'A', '-ascii')
```

- Open browser for input with `uigetfile`

```
[filename, pathname] = uigetfile( …
    {'*.txt', 'Get Text Files'}, …
     'Pick a file')
filename = [pathname filename]
data = load (filename, '-ascii');
```

Command Window I/O

- Get input from Command Window

```
num = input('What altitude: ')
str = input('Which planet: ', 's')
```

- Display output in Command Window
  - Strings

```
disp('Velocity is 500.')
error('Unknown units.')
```

  - If there are numbers to display:

```
message = ['Velocity is: ' str2num(V)]
disp(message)
```

M-File Programming

- Script M-Files
  - Automate a series of steps.
  - Share workspace with other scripts and the command line interface.
- Function M-Files
  - Extend the MATLAB language.
  - Can accept input arguments and return output arguments.
  - Store variables in internal workspace.

Function M-Files

Example: orbitalvelocity.m

```
function V = orbitalvelocity(R, g0, H)
% H1 line: ORBITALVELOCITY computes V.
% Help text: this text appears when
% you type “help orbitalvelocity”.

% Comment: function body is below
V = sqrt( g0 * R^2 / (R + H);
return
```

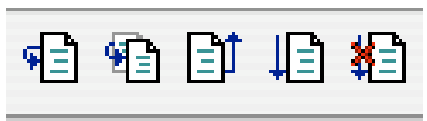
A MATLAB Program

- Always has one script M-File
- Uses built-in functions as well as new functions defined in function M-files
- Created in MATLAB Editor / Debugger

```
>> edit program.m
```

  - Debugging mode


```
k >>
```



MATLAB Programming “Extras”

MATLAB Compiler

Interface to other languages

MAT and MEX Files

Help Browser on “Extras”

- MATLAB
  - + External Interfaces
    - + Importing and Exporting Data
      - + Using MAT Files
    - + Calling C and Fortran Programs
    - + Creating C Language MEX-Files
    - + Calling Java from MATLAB
      - + Bringing Java classes and methods
- MATLAB Compiler

Programming “Extras” Advice

- MATLAB Compiler:
  - Theoretically: you can compile a program and use it outside of MATLAB
  - Practically:
    - It is platform dependent
    - It is C-compiler dependent
    - There are license issues involved
    - On Athena (Linux), read `readme.athena`
- Interface to other languages
  - You need to know C or Java first …
- Attend advanced MATLAB programming training

Programming Exercises

- Exercise One: `modelfitplot.m`
  - Program flow control: `for` loops
  - Using `zeros` and `length` functions
- Exercise Two: `plotprogram.m`
  - User defined functions
  - Function and script M-files
  - File input with `uigetfile`
- Exercise Three: `velocityprogram.m`
  - Program flow control: `if` and `switch` statements
  - Control Window input and output