Outline

- Basics of Matlab
- Control Structures
- Scripts and Functions
- Basic Plotting Functions
- Graphical User Interface
- Help
- Read a truecolor image
- Read a colormapped image
- Colormapped vs. Truecolor
- Image Types
Basics of Matlab

• MATLAB stands for *Matrix Laboratory*.
• Matlab had many functions and toolboxes to help in various applications
• It allows you to solve many technical computing problems, especially those with matrix and vector formulas, in a fraction of the time it would take to write a program in a scalar non-interactive languages such as C.

Basics of Matlab

• **The Language**
  – The MATLAB language is a high-level matrix/array language with control flow statements, functions, data structures, input/output, and object-oriented programming features.

• **Graphics**
  – MATLAB has extensive facilities for displaying vectors and matrices as graphs, as well as editing and printing these graphs. It also includes functions that allow you to customize the appearance of graphics as well as build complete graphical user interfaces on your MATLAB applications.

• **External Interfaces**
  – The external interfaces library allows you to write C programs that interact with MATLAB.
Basics of Matlab

- Command-based environment
- $A(i,j)$ denotes the element located at $i$'th row and $j$'th column
- Matrices are defined using brackets ‘[’ and ‘]’.
- Rows are separated by semicolon ‘;’.
- Matlab has various toolboxes containing ready-to-use functions for various tasks.

Basics of Matlab

- Matlab application window:
Basics of Matlab

- The prompt consists of two right arrows: `>>`
- Just type your command and press Enter.
- Matlab has all elementary functions.

```matlab
>> sqrt(2)
ans =  
1.4142
>> sin(pi/10)
ans =  
0.3827
>> log(10)
ans =  
2.3026
>> log(10^2)
ans =  
2.0000
```

Basics of Matlab

- Create variables directly, and use them in other functions.
- All variables are created with double precision unless specified.

```matlab
>> a=5*(7/2)
   a =  
   27.9375
>> b=sin(pi/9)-cos(pi/9)
   b =  
   -0.6977
>> log(a^2)/log(5)
   ans =  
   7
>> >> atan(1/b)
   ans =  
   -1.0321
```
Basics of Matlab

Colon ‘:’ Operator

• MATLAB’s most powerful operator!
  • 1:5 means 1 2 3 4 5
  • 1:3:10 means 1 4 7 10
  • 100:-10:50 means 100 90 80 70 60 50
  • A(:, 3) returns the third column of A
  • A(3, :) returns the third row of A
  • A(1:2, 1:3) returns the top two rows and first three columns

Basics of Matlab

Generating Matrices

• zeros(M,N)
• ones(M,N)
• eye(N)
• rand(M,N) [uniformly-distributed]
• randn(M,N) [normally-distributed]
• magic(N) [sums along rows, columns and diagonals are the same]
• How can you generate a matrix of all 5’s?
• How can you generate a matrix whose elements are between 2 and 5?
Basics of Matlab

Matrix Concatenation

- \( A = [B \ C] \) concatenates \( B \) and \( C \) left-to-right
- \( A = [B; C] \) concatenates \( B \) and \( C \) top-to-bottom

Basics of Matlab

Deleting Rows or Columns

- \( A(2,:) = [\] \) deletes the second row of \( A \)
- \( A(:,3) = [\] \) deletes the third column of \( A \)
Basics of Matlab

Obtaining Matrix Properties

- \( \min(A) \) finds the minimum of each columns
- \( \min(\min(A)) \) finds the minimum element of \( A \)
- \( \max(\max(A)) \) finds the max. element of \( A \)
- \( \text{sum}(\text{sum}(A)) \) returns the summation
- \( \text{size}(A) \) returns the row and column counts
- \( \text{length}(D) \) returns the length of one-dimensional array \( D \)
- \( \text{ndims}(C) \) returns the dimension of \( C \)
- ‘\text{whos B}’ shows the matrix properties
Basics of Matlab

Relational & Logical Operators

- `<`
- `<=`
- `>`
- `>=`
- `==` Equal to
- `~=` Not equal to

& AND

| OR

~ NOT

Basics of Matlab

• Dealing with matrices:

\[
A = \begin{bmatrix}
4 & -2 & -4 & 7 \\
1 & 5 & 3 & 2 \\
6 & -8 & -5 & -6 \\
-7 & 3 & 0 & 1
\end{bmatrix}
\]

>> size(a)
ans =
  4  4

>> a = [4 -2 -4 7; 1 5 3 2; 6 -8 -5 -6; -7 3 0 1]

>> a(2,3)
ans =
 -3

>> a(2:4,3)
ans =
 -3
 -5
  0

>> a(3,:)
ans =
  6
 -8
 -5
 -6

>> a(:,2)
ans =
 -2
  5
 -8
  3

*; * indicates block of data
Basics of Matlab

- Dealing with matrices:
- We can directly add, subtract, multiply, invert matrices.

```
>> b=[2 4 -7 -4; 5 6 3 -2; 1 -8 -5 -3; 0 -6 7 -1] 
>> 2*a-3*b
ans =
  2 -16  13  26
 -13  -8 -15  10
   9   8  5  -3
 -14  24 -21   1
```

```
>> inv(a)
ans =
 -0.0125  0.0552  -0.0231  -0.1619
 -0.0851  0.1456  -0.0352  -0.0466
 -0.0406  0.1060  -0.1039  -0.1274
  0.1082  -0.0805  -0.0862  0.0064
```

```
>> a'
ans =
  4   1   6  -7
 -2   5  -8   3
 -4  -3  -5   0
  7   2  -6   1
```

Control Structures

- Conditional Control
  - if, else, elseif
  - switch, case
- Loop Control
  - for, while, continue, break
- Error Control
  - try, catch
- Program Termination
  - return
Control Structures

• If Statement Syntax

```matlab
if (Condition_1)
    Matlab Commands
elseif (Condition_2)
    Matlab Commands
elseif (Condition_3)
    Matlab Commands
else
    Matlab Commands
end
```

**Examples**

```matlab
if ((a>3) & (b==5))
    Matlab Commands;
end
```

```matlab
if (a<3)
    Matlab Commands;
elseif (b~=5)
    Matlab Commands;
end
```

```matlab
if (a<3)
    Matlab Commands;
else
    Matlab Commands;
end
```

Control Structures

• For loop syntax

```matlab
for i=Index_Array
    Matlab Commands
end
```

**Examples**

```matlab
for i=1:100
    Matlab Commands;
end
```

```matlab
for j=1:3:200
    Matlab Commands;
end
```

```matlab
for m=13:-0.2:-21
    Matlab Commands;
end
```

```matlab
for k=[0.1 0.3 -13 12 7 -9.3]
    Matlab Commands;
end
```
Control Structures

• While Loop Syntax

  while (condition)
     Matlab Commands
  end

Example

while ((a>3) & (b==5))
    Matlab Commands;
end

Scripts and Functions

• There are two kinds of M-files:

  - **Scripts**, which do not accept input arguments or return output arguments. They operate on data in the workspace. Any variables that they create remain in the workspace, to be used in subsequent computations.

  - **Functions**, which can accept input arguments and return output arguments. Internal variables are local to the function.
Scripts and Functions

- Functions are m-files which can be executed by specifying some inputs and supply some desired outputs.
- The code telling the Matlab that an m-file is actually a function is:

  function out1=functionname(in1)
  function out1=functionname(in1,in2,in3)
  function [out1,out2]=functionname(in1,in2)

- You should write this command at the beginning of the m-file and you should save the m-file with a file name same as the function name.

---

Scripts and Functions

- Examples
  - Write a function: out=squarer (A, ind)
    - Which takes the square of the input matrix if the input indicator is equal to 1
    - And takes the element by element square of the input matrix if the input indicator is equal to 2
Another function which takes an input array and returns the sum and product of its elements as outputs.

The function sumprod(.) can be called from command window or an m-file as:

Example:

```matlab
function h = falling(t)
    global GRAVITY
    h = 1/2*GRAVITY*t.^2;
```

Global Variables

If you want more than one function to share a single copy of a variable, simply declare the variable as global in all the functions. The global declaration must occur before the variable is actually used in a function.
Basic Plotting Functions

MATLAB provides a variety of techniques to display data graphically.

Interactive tools enable you to manipulate graphs to achieve results that reveal the most information about your data.

You can also edit and print graphs for presentations, or export graphs to standard graphics formats for presentation in Web browsers or other media.

Basic Plotting Functions

The plot function has different forms, depending on the input arguments.

If y is a vector, plot(y) produces a piecewise graph of the elements of (y) versus the index of the elements of (y).

If you specify two vectors as arguments, plot(x,y) produces a graph of y versus x.

You can also label the axes and add a title, using the 'xlabel', 'ylabel', and 'title' functions.

Example:

```matlab
x = 0:2*pi;
y = sin(x);
plot(x,y)
xlabel('x = 0:2\pi')
ylabel('Sine of x')
title('Plot of the Sine Function','FontSize',12)
```
Basic Plotting Functions

- **Plotting Multiple Data Sets in One Graph**
  - Multiple x-y pair arguments create multiple graphs with a single call to `plot`.
  
  **For example:**
  ```
  x = 0:pi/100:2*pi;
  y = sin(x);
  y2 = sin(x-.25);
  y3 = sin(x-.5);
  plot(x,y,x,y2,x,y3)
  ```
Basic Plotting Functions

• **Specifying Line Styles and Colors**
  It is possible to specify color, line styles, and markers (such as plus signs or circles) when you plot your data using the plot command:
  \[ \text{plot}(x,y,'color\_style\_marker') \]

  For example: \[ \text{plot}(x,y,'r:+') \]
  plots a red-dotted line and places plus sign markers at each data point.

• **Graphing Imaginary and Complex Data**
  When the arguments to plot are complex, the imaginary part is ignored except when you use a single complex argument.
  For example: \[ \text{plot}(Z) \]
  which is equivalent to: \[ \text{plot}(\text{real}(Z),\text{imag}(Z)) \]

• **Adding Plots to an Existing Graph**
  When you type: \[ \text{hold on} \]

  MATLAB does not replace the existing graph when you issue another plotting command; it adds the new data to the current graph, rescaling the axes if necessary.
Basic Plotting Functions

- **Figure Windows**
  Graphing functions automatically open a new figure window if there are no figure windows already on the screen.

- To make a figure window the current figure, type
  \[
  \text{figure(n)}
  \]
  where \( n \) is the number in the figure title bar. The results of subsequent graphics commands are displayed in this window.

Basic Plotting Functions

- **Displaying Multiple Plots in One Figure**
  \[
  \text{subplot(m,n,p)}
  \]
  This splits the figure window into an \( m \)-by-\( n \) matrix of small subplots and selects the \( p \)th subplot for the current plot.

- **Example:**
  \[
  t = 0:pi/10:2*pi;
  [X,Y,Z] = cylinder(4*cos(t));
  subplot(2,2,1); mesh(X)
  subplot(2,2,2); mesh(Y)
  subplot(2,2,3); mesh(Z)
  subplot(2,2,4); mesh(X,Y,Z)
  \]
Basic Plotting Functions

- **Setting Axis Limits & Grids**
  
  The `axis` command lets you to specify your own limits:
  
  ```
  axis([xmin xmax ymin ymax])
  ```

  You can use the `axis` command to make the axes visible or invisible:
  
  ```
  axis on / axis off
  ```

  The `grid` command toggles grid lines on and off:
  
  ```
  grid on / grid off
  ```

Graphical User Interface

- **GUIDE**, the MATLAB **G**raphical **U**ser **I**nterface **D**evelopment **E**nvironment, provides a set of tools for creating graphical user interfaces (GUIs). These tools greatly simplify the process of designing and building GUIs.
Help

- “%” is the neglect sign for Matlab (equivalent of “//” in C). Anything after it on the same line is neglected by Matlab compiler.
- Sometimes slowing down the execution is done deliberately for observation purposes. You can use the command “pause” for this purpose:

  ```matlab
  pause %wait until any key
  pause(3) %wait 3 seconds
  ```

Help

- You can always use help of Matlab by typing

  ```matlab
  >> help
  >> help command_name
  >> help toolbox_name
  ```
Read a Truecolor Image

Coordinate Convention

![Coordinate Convention Diagram]

**FIGURE 2.1** Coordinate conventions used (a) in many image processing books, and (b) in the Image Processing Toolbox.

Read a Truecolor Image

![MATLAB Code Example]

To get started, select MATLAB Help or Comment from the menu bar.

```matlab
>> I = imread('Les_Bipolarnoise.jpg');
>> imshow(I)
>> axis on
>> xlim([600, 1200])
```

```matlab
>> figure
```

![MATLAB Output]

```matlab
>> axes
```

![MATLAB Output]

Read a Truecolor Image

```
>> I = imread('Les_BingBong.jpg');
>> class(I)
ans =
uint8
>> size(I)
ans =
600 1200 3
>> figure
>> imagesc(I)
>> title('Les BingBong: The Bing-Bong Bloggers')
>> xlabel('Photo: Bart Nagel, 2006, www.bartnagel.com')
```

Read a Truecolor Image

```
>> I = imread;
>> class(I)
ans =
uint8
>> size(I)
ans =
600
>> figure
>> imagesc(I)
>> title('Les BingBong: The Bing-Bong Bloggers')
>> xlabel('Photo: Bart Nagel, 2006, www.bartnagel.com')
```
Read a Truecolor Image

First, select a region using the magnifier.

Cut out a region from the image.

From this close-up we can estimate the coordinates of the region:

rows: about 125 to 425
cols: about 700 to 1050
Read a Truecolor Image

To get start
>> J = imread;
>> class(J)
ans =
uint8
>> size(J)
ans =
800       600
>> figure
>> image(J)
>> title("Image")
>> xlabel("RGB")
>> ylabel("truecolor")
>> J = I(:,:,4);
>> figure
>> image(J)
>> truecolor;

Bring it to the front using the figure command.

Now close the other image.
Read a Truecolor Image

```
>> I = imread('Les_Buissonnieres.jpg','jpg');
>> class(I)
ans =
uint8
>> size(I)
ans =
600 1200 3
>> figure
>> Irgb = I
>> title('Les Buissonnieres: The Going-Going Bloggers')
>> imshow(Irgb)
>> J = I(125:425,700:1050,:);  
>> figure
>> imshow(Irgb(2))
>> title('Les Buissonnieres: The Going-Going Bloggers')
>> imshow(Irgb)
>> close
```

then type 'close'
at the prompt.

Read a Colormapped Image

```
>> cd 'D:\classes\EECE233\Fall 2006\graphics\matlab intro'
>> [L,map] = imread('Liu_Woodring - PlusMinus.gif','gif');
>> figure
>> image(I)
>> set(gca)
>> axis image
>> title('Liu_Woodring - PlusMinus.gif')
>> xlabel('Liu_Woodring - PlusMinus.gif')
>> close
```
Read a Colormapped Image

To get started, select MATLAB Help or Demos from the Help menu.

```matlab
>> cd 'D:\class\ECSE535\Fall 2006\graphics\matlab intro'
>> [I,cmap] = imread('Jim Woodring - PlusMinus.gif','gif');
>> figure
>> image(I)
>> class(I)
ans =
uint8
>> size(I)
ans =
383 533
>> colormap(cmap)
>> title('Plus Minus');
>> xlabel('Jim Woodring [http://www.jimwoodring.com]
>> trueview
```

Read a Colormapped Image

```matlab
>> cd 'D:\class\ECSE535\Fall 2006\graphics\matlab intro'
>> [I,cmap] = imread('Jim Woodring - PlusMinus.gif','gif');
>> figure
>> image(I)
>> class(I)
ans =
uint8
>> size(I)
ans =
383 533
>> colormap(cmap)
>> title('Plus Minus');
>> xlabel('Jim Woodring [http://www.jimwoodring.com]
>> trueview
```
Read a Colormapped Image

```matlab
To get started,
>> cd 'D:\classes1'
>> [I,map] = imread
>> figure
>> image(I)
>> colormap(I)
>> size(I)
>> ans =
323 523
>> colorbar
>> title('Plus Minus1');
>> xlabel('Jim Woodring http://www.jimwoodring.com')
>> truecolor
>> T = imread('Jim Woodring - PlusMinus1.jpg',',jpg');
>> figure
>> image(T)
>> truecolor
>>
```

Intensity values are integers between 0 and 255.

```matlab
image class: uint8
image type: truecolor
```

Colormapped vs. Truecolor

```matlab
image class: uint8
image type: truecolor
```

```
T(231,326,:)
227
222
96
```

Intensity values are integers between 0 and 255.
Colormapped vs. Truecolor

Intensity values are numbers between 0 and 1.

Intensity values are integers between 0 and 1.

Number at pixel location is an index into a colormap.
Colormapped vs. Truecolor

Example truecolor and colormapped images

24-bit truecolor   8-bit colormapped to 24 bits

Colormapped vs. Truecolor

Example truecolor and colormapped images

24-bit truecolor   8-bit colormapped to 24 bits
Colormapped vs. Truecolor

How to convert a colormapped image to true color

\[ M \text{ is a 512x512x1, 8-bit image. It has 262,144 pixels. Each pixel has a value between 0 & 255.} \]

\[ \text{cmap is the colormap that is stored in 'button_mapped.bmp' along with image. cmap is a 256x3 type-double matrix, each row of which lists a color in terms of its R, G, & B intensities, which are given as fractions between 0 and 1.} \]

\[
\begin{align*}
\text{>> } & [M,\text{cmap}] = \text{imread('button\_mapped.gif','gif')}; \\
\text{>> } & T = \text{uint8} \left( \text{reshape} \left( \text{cmap} \left( M+1, : \right), \left[ \text{size}(M) \ 3 \right] \right) \cdot 255 \right);
\end{align*}
\]

The 262,144 x 3 matrix of intensity values is reshaped into a 512x512x3 image of type double. The values are scaled to lie between 0 & 255 then converted to type uint8.

Colormapped vs. Truecolor

How to Make Colormaps

\[
\begin{align*}
\text{>> } & \text{ramp} = \left( 0:255 \right)'/255; \\
\text{>> } & \text{rkm} = \text{[ramp ramp ramp]}; \\
\text{>> } & \text{rcm} = \text{[ramp zeros(256,2)]}; \\
\text{>> } & \text{gcm} = \text{[zeros(256,1) ramp zeros(256,1)]}; \\
\text{>> } & \text{rcm} = \text{[zeros(256,2) ramp]};
\end{align*}
\]

This code, 0:255, generates a 1 row by 256 element vector of class double that contains numbers 0 through 255 inclusive.

This, \((0:255)^\prime\), has the same contents and class but is a 256 row by 1 column vector. The apostrophe (') is the matrix transpose operator.
Colormapped vs. Truecolor

R, G, & B bands of a truecolor image displayed with grayscale colormaps.

R, G, & B bands of a truecolor image displayed with tinted colormaps.
Colormapped vs. Truecolor

Saving Images as Files

```matlab
>> % truecolor as .bmp
>> imwrite(I,'image_name.bmp','bmp');
>>
>> % truecolor as .jpg (default quality = 75)
>> imwrite(I,'image_name.jpg','jpg');
>>
>> % truecolor as .jpg (quality = 100)
>> imwrite(I,'image_name.jpg','jpg','Quality',100);
>>
>> % colormapped as .bmp
>> imwrite(I,cmap,'image_name.bmp','bmp');
>>
>> % colormapped as .gif
>> imwrite(I,cmap,'image_name.gif','gif');
>>
Assuming that
'I' contains the image of
the correct class,
that
'cmap' is a colormap,
and that
'image_name' is the
file-name that you
want.
```

Image Types

Image Formats

- TIFF: Tagged Image File Format
- JPEG: Joint Photographic Experts Group
- GIF: Graphics Interchange Format
- BMP: Windows Bitmap
- PNG: Portable Network Graphics
- XWD: X Window Dump
Image Types

- Image Processing Toolbox of Matlab supports four types of images:
  - Intensity images
  - Binary images
  - Indexed images
  - RGB images

Intensity Images

- If the elements of an intensity image are of class uint8, or class uint16, then they have integer values in the range [0, 255] and [0, 65535], respectively.
- If the image is of class double, the values are floating-point numbers and take the values in the range [0, 1].
Image Types

Binary Images

- A binary image is a logical array of 0s and 1s.
- A numeric array is converted to binary using the function ‘logical’.

Image Types

Converting between image classes

- `im2uint8` : Convert image to 8-bit unsigned integers.
- `im2uint16` : Convert image to 16-bit unsigned integers.
- `mat2gray` : Convert matrix to intensity image
- `im2double` : Convert image to double precision
- `im2bw` : Convert image to binary image by thresholding.
Exercises

- How can you flip an image vertically?
- How can you flip an image horizontally?
- How can you crop an image?
- How can you subsample an image (i.e. make smaller)?

Summary

- Basics of Matlab
- Control Structures
- Scripts and Functions
- Basic Plotting Functions
- Graphical User Interface
- Help
- Read a truecolor image
- Read a colormapped image
- Colormapped vs. Truecolor
- Image Types
References

- http://www.mathworks.com
- "Lecture Notes" by V. Adams and S.B. Ul Haq
- "Lecture Notes" by İ.Y. Özbek
- "Lecture Notes" by Frank (Qingzhong) Liu, University of New Mexico Tech.
- "Lecture Notes" by Richard Alan Peters, Vanderbilt University.