INTRODUCTION TO MATLAB

MATLAB is an **interactive program** for numeric computation and data visualization. Fundamentally, MATLAB is built upon a foundation of sophisticated matrix software for analyzing linear systems of equations. The tools springing from these numerical foundations have proven to be extraordinarily versatile and capable in their ability to solve problems in applied math, physics, chemistry, engineering, finance - almost any application area that deals with complex numerical calculations. Run some demos available from the MATLAB screen to see examples of MATLAB at work.

MATLAB works with **scalars, vectors, and matrices**. A scalar is really just a 1-by-1 matrix, and a vector is a long, thin matrix given as either a row or a column. In this sense, **everything that MATLAB operates on is a matrix**.

Simple examples:

Create a simple vector with 9 elements called 'a':

```
>>a= [1 2 3 4 6 4 3 4 5]
a=
     1  2  3  4  5  6  4  3  4  5
```

Add 2 to each element of our vector 'a' and store the result in vector 'b':

```
>>b=a+2
b=
     3  4  5  6  7  8  6  5  6  7
```

To plot vector 'b':

```
>>plot (b)
```

To plot it with an '*':

```
>>plot(b, '*')
```

To add a grid:

```
>>grid
```

To make a bar graph:

```
>>bar(b)
```

To create a matrix:

```
>>A=[1 2 0; 2 5 -1; 4 10 -1]
A=
     1  2  0
     2  5 -1
     4 10 -1
```

To create a 2x2 sub-matrix B:

```
>>B=A(1:2, 1:2)
B =
     1  2
     2  5
```

To get the center element of matrix A:

```
>>t=A(2,2)
t =
     5
```

To get the second row of the matrix A:

```
>>B=A(2,:)
B =
     2   5  -1
```

To get the second column of the matrix A:

```
>>B=A(:,2)
B =
     2
     5
    10
```

To transpose a matrix:

```
>>B=A'
```

```
     1  2   4
     2  5 10
     0 -1  -1
```
To get the value of a variable 'A':

>>A

A=
```
     1  2  0
     2  5 -1
     4 10 -1
```
To get a listing of the variables:

>>whos

An example of "reshaping" the matrix:

>>a=[1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24];

>>b=reshape(a,3,8)

b =

1	4	7	10	13	16	19	22
2	5	8	11	14	17	20	23
3	6	9	12	15	18	21	24

>>b=reshape(a,8,3)

b =

1	9	17
2	10	18
3	11	19
4	12	20
5	13	21
6	14	22
7	15	23
8	16	24

>>b=reshape(a,8,3)'

b =

1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24

MATLAB is extendable!

MATLAB is both an environment and a programming language, and one of its great strengths is the fact that the **MATLAB language allows you to build your own reusable tools**.

MATLAB programs, or M-files, are ASCII files denoted by the filename suffix '**.m**' and come in two basic forms: **scripts and functions**. When the name of a script M-file is typed at the command line, MATLAB simply executes the commands found in the file. This can be used to automate long sequences of commands. A function M-file is like a script, except that it allows argument passing and local variables. Many of the MATLAB commands available at the command line are actually function M-files that can be open for your inspection.

In M-files, the familiar **if, while,** and **for** statements of other programming languages can be used for control flow. MATLAB is also equipped with a **debugger**.

Finally, MATLAB contains an extensive collection of **graphical interface tools** (pushbuttons, radio-buttons, edit boxes, sliders, etc.) that you can use to build user interfaces.

TOOLBOXES

Toolboxes are specialized collections of M-files (MATLAB language programs) built specifically for solving particular classes of problems.

Toolboxes are more than just collections of useful functions, though. They represent the efforts of some of the world's top researchers in fields such as controls, signal processing, system identification, neural networks, optimization, control systems, robust control, image processing, statistics, nonlinear control, etc.

IMAGE PROCESSING TOOLBOX

The Image Processing Toolbox provides a powerful and flexible environment for image processing and analysis. Its comprehensive library of functions is an open system that may be easily tailored to fit specific application requirements.

>>help images

```
Image Processing Toolbox.
Version 2.0 Beta  15-Nov-1996

Release notes
  Readme     - Version 2.0 Beta release notes.

Image types and type conversions.
  dither     - Floyd-Steinberg image dithering.
  gray2ind   - Convert gray scale intensity image to indexed image.
  im2bw      - Convert image to black and white by thresholding.
  ind2gray   - Convert indexed image to gray intensity image.
  ind2rgb    - Convert indexed image to an RGB image.
  isbw       - True for black and white images.
  isgray     - True for intensity images.
  isind      - True for indexed images.
  mat2gray   - Convert matrix to (gray) intensity image.
  rgb2gray   - Convert RGB values to gray.
  rgb2ind    - Convert RGB image to indexed image.

Image file I/O.
  imread     - Read an image file.
  imwrite    - Write an image file.

Image display.
  colorbar   - Display color bar (MATLAB Toolbox).
  getimage   - Get image data from axes.
  image      - Display indexed image (MATLAB Toolbox).
  imagesc    - Scale data and display as image (MATLAB Toolbox).
  immovie    - Make a movie of an image deck.
  imshow     - Display all types of image data.
  imzoom     - Zoom in and out an image or 2-D plot.
  montage    - Display an image deck as a rectangular montage.
  subimage   - Display multiple images.
  truesize   - Resize figure so that image is actual size.
  warp       - Warp an image onto a surface.

Geometric operations.
  griddata   - Data gridding and surface fitting (MATLAB Toolbox).
  imcrop     - Crop image.
  imresize   - Resize image.
  imrotate   - Rotate image.
  interp2    - Two-dimensional data interpolation (MATLAB Toolbox).
```

Pixel values and statistics.
  contour   - Contour (level curves) plot (MATLAB Toolbox).
  corr2   - Two-dimensional correlation coefficient.
  imhist   - Image histogram.
  impixel   - Color of a pixel.
  improfile - Intensity profile.
  mean2   - Mean of a matrix.
  std2   - Two-dimensional standard deviation.

Image analysis.
  edge   - Edge extraction.
  qtdecomp   - Quadtree decomposition.
  qtgetblk   - Get block values according to a quadtree decomposition.
  qtsetblk   - Assign block values according to a quadtree decomposition.

Image enhancement.
  grayslice - Density (intensity) slicing.
  histeq   - Histogram equalization.
  imadjust   - Adjust and stretch image intensity.
  imnoise   - Image noise.
  wiener2   - Adaptive 2-D Wiener filtering.

Linear filtering.
  conv2   - Two-dimensional convolution (MATLAB Toolbox).
  convmtx2   - Two-dimensional convolution matrix.
  convn   - N-dimensional convolution (MATLAB Toolbox).
  filter2   - Two-dimensional filtering (MATLAB Toolbox).
  fspecial   - Special 2-D filters.

Linear 2-D filter design.
  freqspace - Frequency response spacing (MATLAB Toolbox).
  freqz2   - Two dimensional frequency response.
  fsamp2   - 2-D FIR filter design via frequency sampling.
  ftrans2   - 2-D FIR filter design via frequency transformation.
  fwind1   - 2-D FIR filter design using 1-D windows.
  fwind2   - 2-D FIR filter design using 2-D windows.

Image transforms.
  dct2   - Two-dimensional discrete cosine transform.
  dctmtx   - Discrete cosine transform matrix.
  fft2   - Two-dimensional fast Fourier transform (MATLAB Toolbox).
  fftn   - N-dimensional fast Fourier transform (MATLAB Toolbox).
  fftshift   - Move zeroth lag (DC component) to center (MATLAB Toolbox).
  idct2   - Two-dimensional inverse discrete cosine transform.
  ifft2   - Two-dimensional inverse FFT (MATLAB Toolbox).
  ifftn   - N-dimensional inverse fast Fourier transform (MATLAB Toolbox).
  radon   - Radon transform.

Neighborhood and block processing.
  bestblk   - Best block size for block processing.
  blkproc   - Process an image in blocks.
  col2im   - Rearrange distinct or sliding column blocks to form image.
  colfilt   - Local non-linear filtering as columns.
  im2col   - Rearrange distinct or sliding blocks into columns.
  medfilt2   - Two-dimensional median filtering.
  nlfilter   - Local non-linear filtering.
  ordfilt2   - 2-D order-statistic filtering.

Region-based processing.
  mfilter2   - Masked filter.

```
  roicolor   - Define region of interest (ROI) by color.
  roifill    - Smoothly interpolate within a specified region.
  roipoly    - Define polygonal region of interest (ROI).

 Binary image operations.
  applylut   - Binary image lookup table operation.
  bwarea     - Area of objects in binary image.
  bweuler    - Euler number.
  bwfill     - Binary image flood fill.
  bwlabel    - Label connected components in a binary image.
  bwmorph    - Morphological operators.
  bwperim    - Perimeter of objects in a binary image.
  bwselect   - Select objects in a binary image.
  dilate     - Dilate (thicken) a binary image.
  erode      - Erode (thin) a binary image.
  makelut    - Generate LUT for use in APPLYLUT.

 Colormap manipulation.
  brighten   - Brighten or darken a colormap (MATLAB Toolbox).
  cmgamdef   - Default gamma correction table.
  cmgamma    - Gamma correct colormap.
  cmpermute  - Permute colormap positions.
  cmunique   - Find unique colormap colors and corresponding image.
  colormap   - Set or get the color look-up table (MATLAB Toolbox).
  imapprox   - Approximate indexed image by an image with fewer colors.
  rgbplot    - Plot RGB colormap components (MATLAB Toolbox).

 Colorspace conversions.
  hsv2rgb    - Convert HSV values to RGB values (MATLAB Toolbox).
  ntsc2rgb   - Convert NTSC values to RGB values.
  rgb2hsv    - Convert RGB values to HSV values (MATLAB Toolbox).
  rgb2ntsc   - Convert RGB values to NTSC values.
  rgb2ycbcr  - Convert RGB values to YCbCr values.
  ycbcr2rgb  - Convert YCbCr values to RGB values.
```

MATLAB EXAMPLES

```
>>demo                                    %  MATLAB demos
```

```
>>y=readimage('brain.dat',[128,128],'uint16');  % read image
>>imshow(y)                                    % display image
>>colormap('gray')                             % display image using a gray scale
```

Do you see an image? The command *imshow* needs the minimum and maximum of the image data.

```
>>mn=min(min(y))
>>mx=max(max(y))
>>imshow(y, [mn, mx])
```

```
>>figure                                  % create a new figure window
>>figure(2)                                    % create figure window no. 2
```

```
>>pts=ginput(1)                           % x and y coordinates (mouse)
        pts =
                75.7821   58.3462
>>pts=ginput(4)                           % four x and y coordinates (mouse)
        pts =
                83.9872   36.6026
                50.3462   39.0641
```

```
        65.1154    87.8846
        104.0897    88.7051
```

```
>>t=size(y,1)                     % image size (number of rows)
>>r=size(y,2)                     % image size (number of columns)
>>[t,r]=size(y)                   % get the size of the image
```

```
>>clg                             % clear the image window
```

```
>>hist(y(:))                      % histogram with 10 bins
>>hist(y(:),256)                  % histogram with 256 bins
>>[a,b]=hist(y(:),256);           % histogram saved in an array
```

```
>>plot(a,b,'+')                   % linear 2D plot
>>plot(b,a,'+')                   % another linear 2D plot
>>title('plot')                   % title

>>xlabel('pixel intensity')       % label
>>ylabel('No. of pixels')         % label

>>axis([0,34000,0,4000])          % change display range
>>axis([0,25000,0,2000])          % change display range
```

FUNCTION readimage.m

```
function im=readimage(filename, dim, type)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  READIMAGE
%  Read an image file with a given name of the file, image dimensions and image type.
%
%
%          IM = READIMAGE(FILENAME, DIM,TYPE)
%
%    where:
%
%          FILENAME     filename of image file to be read
%          DIM          2x1 vector specifying dimensions of image as [ROWS COLS]
%          TYPE         image type: (schar, uchar, uint16, int16, float, etc.)
%                            schar - signed character 8-bit,
%                            uchar - unsigned character 8-bit,
%                            uint16 - unsigned integer 16-bit,
%                            int16 - signed integer 16-bit,
%                            float - floating point 32-bit
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
error(nargchk(2,3,nargin));

if nargin<3,
     type='uchar';                %default type is 'uchar'
end

fid=fopen(filename);              % open a file
x=fread(fid,type);                % read binary data from a specified file and write to a matrix x
fclose(fid);                      % close a file

im=reshape(x,dim(2),dim(1))';     % change size; returns the dim(2)-by-dim(1) matrix im,
%                                 whose elements are taken columnwise from matrix x
```