MATLAB Minicourse: Lecture 1

David S. Touretzky

March 2001

What Is Matlab?

- Product of The Mathworks, Inc.

  http://www.mathworks.com

- Runs on Unix, LINUX, Windows, and Macs.
- Strong on matrix manipulation and graphics.
- Full programming language.
- Interfaces with C.
- Latest release is version 6.0.

Variable Creation

```
a = 5

a = 5;

b = 'penguins love herring'

who

whos
```

Matrix Creation

x = [1 2 3; 9 8 7]

zeros(3,5)
zeros(5)
zeros(5,1)

ones, rand, randn, eye

The colon operator creates row vectors:

    1:5
    1:3:15
    10:-1:0

    pts = 0 : pi/20 : 2*pi;

Size of a Matrix

whos pts

size(pts)

length(pts)

Subscripting

```
V = [10 20 30 40 50];

V(3)

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

```
M =
      1   2   3
      4   5   6
      7   8   9
```

```
M(2,2)

M(2)     % access in column-major order
```

Matrix Slices

V(2:4)

V(2:end)

M(1:2,2:3)

M(:)

M(:,:)

Expanding a Matrix

```
a = [1 2 3]

a = [a 4]

a(7) = 5

a(end+1) = 6

b = [a; a.^2]
```

Efficiency tip: Use ZEROS to preallocate space instead of expanding dynamically.

Reshaping a Matrix

M = reshape(1:15, 5, 3)

M'

M''

Exercise: how can you recreate the following matrix using just the operators above?

1	2	3
4	5	6
7	8	9
10	11	12
13	14	15

Adding Rows vs. Columns

```
M = [ 1 2 ;  3 4]

M = [M ; 5 6]



V = [10 20 30]'

whos M V

M = [M V]



M = [M [99 98 97]']
```

Deleting Rows or Columns

M(:,3) = [ ]

M(2,:) = [ ]

size([ ])

Command Line Editing

Basic editing:	
Forward/back char	^F / ^B
Left/right word	^L / ^R
Beginning/end of line	^A / ^E
Delete forward/back char	^D / ^H
Clear line	^U
Kill to end of line	^K
Toggle insert mode	^T
Interrupt execution:	^C
History:	
Next/previous line	^N / ^P
Keyed history:	
foo^P	

help cedit

Command/function completion:

cle<tab><tab>

Editing in Emacs

Current buffer is foo.m

Put “3+5” in the file (without the quotes.)

On a new line, put “magic(5)” in the file.

Save the file.

Type **foo** to Matlab

Basic Plotting

```
pts = 0 : pi/20 : 4*pi;

plot(sin(pts))

plot(pts, sin(pts))

whitebg(gcf, [ 0 0 0 ])

grid on/off

box on/off

axis on/off

clf
```

Plot Labeling

```
xlabel('Input value')

ylabel('y = sin(\theta)')

title('The Sine Function')
```

Multiple Plots

```
clf

hold on

plot(pts, sin(pts))

plot(pts, cos(pts), 'm')

plot(pts, cos(pts), 'go')

legend('sin','cos','pts')
```

Use the mouse to position the legend.

Summary of Plot Options

Color:

r,g,b,w	red, green, blue, white
c,m,y,k	cyan, magenta, yellow black

Symbol:

. o x	dot, circle, cross,
+ *	plus, star
s d	square, diamond

etc. (there are more)

Line type:

-	solid
--	dashed
..	dotted
-.	dash-dot

help plot

Printing

On the **File** pulldown menu, select **Print**.

Or type **^P** in the figure window.

print -depsc -r300 myfig.ps

print -dtiff myfig.tiff

print -djpg myfig.jpg

help print

Plotting with Error Bars

```
clf

y = sin(pts);

e = rand(1,length(y)) * 0.4;

errorbar(pts,y,e)
```

Multiple Figures

figure

figure(5)

delete(2)

Or type **^W** in the figure window.

Histograms

```
dat = randn(10000,1);

hist(dat)

hist(dat,50)

b = hist(dat,6)

bar(b)
```

Writing Functions

In Emacs, create the file parabola.m

```
function y = parabola(x)

% y = PARABOLA(x)
%    computes a quadratic

  y = x .^ 2;
```

parabola(5)

clf, plot(parabola(-10:10))

help parabola

Scripts vs. Functions

Scripts take no input arguments and return no values.

Scripts operate in the workspace of their caller (i.e., the "base" workspace if called from the command line.)

Functions can take zero or more arguments and return zero or more values.

Functions operate in their own local workspace.

Variables created inside a function are local to that function; they disappear when the function returns.

Logical Operations

Operators: == ~= < > <= >=

Binary values are true/false

The IF statement:

```
if  x  > 3
   y = x;
else
   y = x - 3;
end
```

Short form (all on one line):

```
if x>3, y=x; else y=x-3; end
```

Boolean Subscripting

```
V = [1 2 3 4 5];

V(logical([1 0 1 1 0]))


a = V >= 3;

whos a


V(V>=3)

V(V>=3) = 0


S = 'banana cabana'

S(S == 'a') = [ ]
```

Control Structure

```
for i = 1:5
   [i  i^2]
end
```

```
clf, hold on
for x = pts
   plot(x,cos(x),'gd')
   pause(1)
end
```

```
x = 0; i = 0;
while x < 5
    i = i + 1;
    x = x + rand(1);
end
i, x
```

Matrix Arithmetic

Element-wise operators: + - .* ./ .^

M = rand(5,3)

M + 100

M * 5

M .* M

M ./ M

M .^ 2

Matrix Multiplication

m1 = rand(5,3)

m2 = rand(3, 5)

m1 * m2

m2 * m1

m1 * m1 *-- error! --*

m1'/m2

pinv(m1)

Exercise: Rotation in 2D

```
function rot(theta)
 % ROT(theta) prints a rotated sine wave

 rads = theta/360*2*pi;

 rotmat = [cos(rads) sin(rads); ...
           -sin(rads) cos(rads)];

 pts = 0:pi/20:4*pi;

 data = rotmat * [pts/(4*pi); cos(pts)];

 plot(data(1,:), data(2,:))
 axis([-1.5 1.5 -1.5 1.5])
```

Rotation Exercise (cont.)

Test your function:

```
rot(30)
```

Now try this:

```
for i = 0 : 10 : 90
   rot(i), pause
end
```

Hit the space key to continue from the pause.

Reduction Operators

```
M = rand(5,3)

sum(M)

sum(M,2)

sum, prod, min, max, mean, var

min(min(M))

min(M(:))
```

Expanding with REPMAT

The REPMAT function is often used to expand a vector to fit the shape of a matrix.

Example: adjusting a dataset to have zero means.

```
M = rand(5,3)

avg = mean(M);

ma = repmat(avg,5,1);

mz = M - ma

sum(mz)
```

Exercise

Suppose we want the rows of M to sum to zero, instead of the columns.

How would you do this, without using transpose?

MATLAB Documentation

help cos

doc cos

help

which peaks

lookfor rotate

The Matlab Helpdesk:
    helpdesk

Hint: Starting Matlab

Matlab version 6 includes an elaborate "desktop" feature built in Java.

The desktop is slow and buggy.

To disable the entire Java Virtual Machine:

```
% matlab -nojvm
```

To disable just the desktop:

```
% matlab -nodesktop
```

In case of OpenGL lossage, do:

```
opengl neverselect
```

Can put this in your startup.m file.