CNBC Matlab Mini-Course

David S. Touretzky
October 2023

Day 1: Essentials

What Is Matlab?

- Product of The Mathworks, Inc.

  http://www.mathworks.com

- Runs on Linux, Windows, and Macs.
- Student version just $99 (plus toolboxes).
- Latest release is Matlab R2023b.
- “Interactive” interface like BASIC, Python, Lisp, etc. Type in expressions and see the result.

What Is Matlab? (cont.)

- Full programming language.
- Strong on matrix manipulation and graphics.
- Optional toolboxes for statistics, image processing, signal processing, etc.
- Interfaces with C, Fortran, and Java.
- Can create stand-alone executable files.
  - HHsim, a Hodgkin-Huxley simulator developed by Dave Touretzky with help from Jon Johnson, is distributed as a stand-alone executable. (Source is also available.)

Why Should You Learn Matlab?

- Data analysis:
  - Much more versatile than a spreadsheet.
  - Extensive statistics toolbox.
  - SPM uses Matlab.
- Graphics:
  - Many ways to visualize your data – even animations!
  - Produce great figures for your papers.
- Modeling and simulation:
  - Best choice for neural net simulations.

Getting Started

- Log in to a workstation.
- On Linux:
  - Start a terminal
  - Type “matlab”

Variable Creation

```
a = 5

a = 6 ;

b = 'penguins love herring'

who

whos
```

single quote

Click on the **Workspace** tab for a graphical version of whos.

Matrix Creation

x = [1 2 3 ; 9 8 7]

zeros(3, 5)
zeros(5)
zeros(5, 1) column vector
zeros(1, 5) row vector

ones, rand, randn, eye

What does eye do?

Colon Creates Row Vectors

1 : 5

1 : 3 : 15

10 : -1 : 0

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

Size of a Matrix

whos pts

size(pts)

length(pts)

Subscripting

V = [10 20 30 40 50];

V(3) ← index from 1, not 0

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

M(6)

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(rows,cols) to preallocate large arrays instead of growing them dynamically.

Reshaping a Matrix

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

M'

M' '  or  (M')'
```


Exercise

- Create the following matrix using **only** the colon, reshape, and transpose operators.

```
 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] '

M = [M V]

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


Deleting Rows or Columns

```
M(: , 3) = [ ]

M(2, :) = [ ]

size([ ])
```


Command Line Editing

- Arrow keys work like you expect
- Basic Emacs commands also work:

Forward/back char	^F / ^B
Left/right word	alt-F / alt-B
Beginning/end of line	^A / ^E
Delete forward/back char	^D / backspace
Clear line	^U
Kill to end of line	^K
Undo	^_

- Environ. > Preferences > Keyboard > Shortcuts for a list, or to switch to Windows conventions.

Command Line History

- Scrolling through the command history:
  - Move to previous command ↑
  - Move to next command ↓
- Can also double click (or click and drag) on an item in the Command History window
- Command/function completion:
  cle<tab>
- Interrupt execution: ^C

Editing Files in Matlab

New > Script

Put **3+5** on the first line

Put **m = magic(5)** on the second line

Save the file as **foo.m** in the current directory.

Type **foo** in the Command Window

Basic Plotting

```
pts = 0 : pi/20 : 4*pi ;
plot(sin(pts))
plot(pts, sin(pts))

axis off / on
grid on / off
box off / on
whitebg(gcf, [0 0 0])
clf
clf reset
```


Plot Labeling

pl^P

xlabel('Angle \theta')

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')
```

Click and drag to position the legend.

Summary of Plot Options

- Colors: r,g,b,w c,m,y,k
- Symbols: . o x + * s(quare) d(iamond) etc.
- Line type: - (solid), -- (dashed), : (dotted), -. (dash-dot)

help plot

Printing

- On the **File** pulldown menu, select **Print**.
- Or type **^P** in the figure window.

- Printing to a file:

```
print -djpeg myfig.jpg
print -depsc -r300 myfig.ps
print -dtiff myfig.tiff
```

- To learn more: **help print**

Plotting With Error Bars

```
clf

y = sin(pts);

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

errorbar(pts, y, e)
```


Writing Your Own Functions

New > Function

```
function [ y ] = parabola( x )
% PARABOLA    Computes a quadratic.
% Y = parabola(X)     May be called with a vector.
y = x .^ 2;
```

Save as **parabola.m**

Try: parabola(5)
help parabola
clf, plot(parabola(-10 : 10),'r--s')

parabola ← Gives an error message. Why?

Multiple Figures

```
figure

bar3(abs(peaks(7)))

figure(5)

delete(2)
```

*Or type **^W** in a figure window to close it.*

Scripts vs. Functions

- **Scripts** take no input arguments and produce no return values.
- Scripts operate in the workspace of their caller.
- If called from the command line, scripts operate in the **base workspace**.
- If called from within a function, scripts operate in the function's **local workspace** and can see and modify its local variables.

Histograms

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

hist(dat)

hist(dat, 50)

b = hist(dat, 6)

bar(b)
```


Scripts vs. Functions

- **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.
- Local variables disappear when the function returns.

Logical Operations

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

Can't use != as in Java or C

Logical values:
0 means "*false*"
1 (or any non-zero number) means "*true*"

a = (3 >= 1 : 5) What are the type and size of a?

Boolean Subscripting

```
V = [1 2 3 4 5];
V(logical([1 0 1 1 0]))

V( V >= 3 )
V( V >= 3) = 0

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


The IF Statement

```
if x >= 3
  y = x;
else
  y = x + 3;
  hadHelp = true;
end
```

Differences from C/C++/Java:

- No ( ) parens around the condition expression.
- No { } braces around the then/else clauses.
- Requires **end** keyword.

Short form – use commas or semicolons:
if x>3, y=x; **else** y=x+3; hadHelp=true; **end**

Control Structure: FOR Loops

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


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

(you can use ^C to terminate the loop)

Control Structure: WHILE Loops

How quickly can a random accumulator reach 5?

```
accum = 0; steps = 0;
while accum < 5
  steps = steps + 1;
  accum = accum + rand(1);
end
steps, accum
```


Element-Wise Arithmetic

Element-wise operators: + − .* ./ .^

Dot means "element-wise"

```
M = rand(5,3)
M + 100
M .* 5    same as    M * 5
M .* M    not same as    M * M
M ./ M
M .^ 2
```

Matrix Arithmetic

```
m1 = rand(5,3)
m2 = rand(3, 5)

m1 * m2      (5×3) * (3×5) → (5×5)
m2 * m1      (3×5) * (5×3) → (3×3)
m1 * m1      Error!  Shapes don't fit.
m1 / m2      Error!  Shapes don't fit.
m1' / m2

pinv(m1)     (5×3) → (3×5)
```


Exercise: Data Plotting Script

```
x = 0 : pi/20 : 5*pi ;
y = sin(x) + x/3 + randn(1,length(x))/4;
z = smooth(y,20)' ;
clf, hold on
plot(x, y, 'bo--')
plot(x, z, 'm', 'LineWidth', 3)
```

Save as mydata.m and run it several times.

Exercise (cont.)

Now add these additional lines:

```
maxL = [1, z(2:end) > z(1:end-1)] ;
maxR = [z(1:end-1) > z(2:end), 1];
localMax = maxL & maxR; % true if point is local maximum
px = x(localMax); px(2,:)=0; px(3,:)=NaN;
pz = z(localMax); pz(2,:)=z(localMax); pz(3,:)=NaN;
plot(px, pz, 'r')
```

For homework: figure out how it works.

Reduction Operators

```
M = rand(5, 3)

sum(M)
sum(M, 2)      sum along 2nd dimension

sum, prod, min, max, mean, var

min(min(M))
min( M(:) )
```


Expanding with REPMAT

- REPMAT is often used to expand a vector to fit the shape of a matrix.
- Example: adjusting a dataset to have zero mean.

```
M = rand(5, 3)
avgs = mean(M)
Mavgs = repmat(avgs, 5, 1)
Mzero = M – Mavgs
sum(Mzero)
```


Exercise

- Suppose we want the *rows* of M to sum to zero, instead of the columns.
- How would you do this, *without* using the transpose operator?

Matlab Documentation

help cos

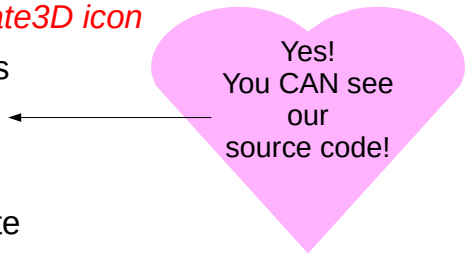
doc cos

clf, peaks

click on rotate3D icon

which peaks

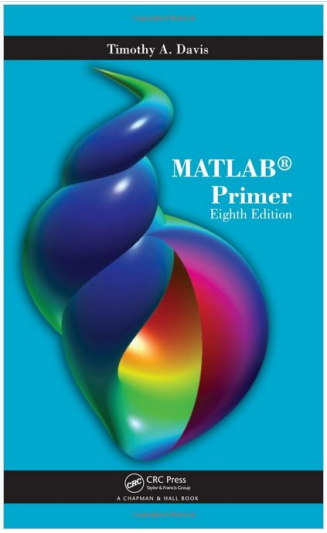
edit peaks



lookfor rotate

Browsing Online Documentation

- Press F1 to bring up the Documentation Browser
- In the documentation browser:
  > Statistics and Machine Learning Toolbox
  > Probability Distributions
  > Continuous Distributions
  > Beta Distribution
  > (Concepts) Beta Distribution



MATLAB Primer, 8th ed.

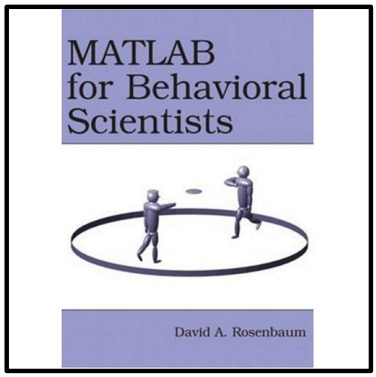
Timothy A. Davis

CRC Press

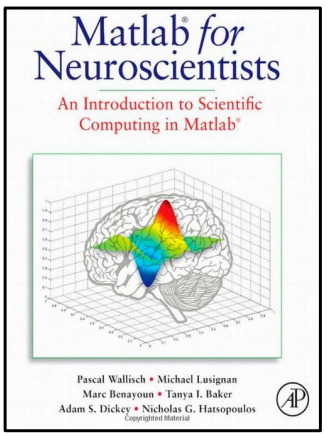
$26 paperback

$18 ebook

Handy pocket reference.



Introductory Text



Examines a variety of neuroscience applications, with examples.

Ways To Learn Matlab

- Three more days of this mini-course.
- Tutorial videos at mathworks.com
- Built-in demos:
  doc demo
- Browse the online documentation
- Dozens of books:
  Amazon.com reports 7,900 search results!
- Matlab Central: user community site
  http://www.mathworks.com/matlabcentral
- Questions to support@mathworks.com