MATLAB Minicourse: Lecture 2

David S. Touretzky

March 2001

Scientific Functions

Trig:	sin, cos, tan, asin, acos, atan, sinh, cosh, tanh, asinh, acosh, atanh
Rounding:	floor, ceil, round, fix
Modular:	rem, mod
Expon.:	exp, log, log2, log10, sqrt
Primes:	factor, primes
Matrix:	det, inv, pinv, eig, svd, fft *and many more*
Polynomials:	roots, polyfit, polyval

Inf and NaN

3/0   returns   Inf

0/0   returns   NaN

3+Inf

Inf/Inf

-Inf

-NaN

Predicates

isreal(3)

isprime(17)

isnumeric([2 3 5])

isempty([ ])

isinf(Inf)

isnan(NaN)

islogical(1==1)

ischar('a')

isequal('foo','aardvark')

Return Values

Functions can return multiple values.

```
A = rand(5,3);

[r,c] = size(A)
```

Functions can return values or not, depending on whether the user is asking for values.

```
plot([1 2 3],[3 1 2])

h = plot([1 2 3],[3 1 2])

set(h,'LineStyle','--')

set(h,'LineWidth',8)
```

Variable # of Arguments

Some functions take a variable number of arguments.

```
peaks

peaks(50)
```

```
hist(randn(2000,1))

hist(randn(2000,1), 50)

b = hist(randn(2000,1), 50)

[b,c] = hist(randn(2000,1), 50)
```

nargin and nargout

Inside a function, **nargin** is the number of input arguments supplied with the call.

nargout is the number of output arguments requested with the call.

```
function [x,y,z] = test(p,q,r,s,t)
  % inputs are ignored

  if nargout >= 1
    x = 50;
    if nargout >= 2
      y = 'foo';
      if nargout >= 3
        z = 3:7;
      end
    end
  end
  whos
```

Name Spaces

Base workspace: variables created outside of any function exist in the base workspace.

Each function executes in a separate local workspace holding the arguments, return variables, and any local variables created by the function.

Functions cannot access variables of the base workspace.

Global workspace: variables declared global by a function are accessed in the global workspace. It's a good idea to also declare the variable global in the base workspace.

Global Variables

In the base workspace:

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

Inside a function:

```
function h = circ(x,y)

 % h = CIRC(x,y) draw circle at (x,y)

 global pts
 hh = plot(x+cos(pts),y+sin(pts));

 if nargout > 0, h = hh; end
```

Scripts Called by Functions

Scripts do not have their own workspaces.

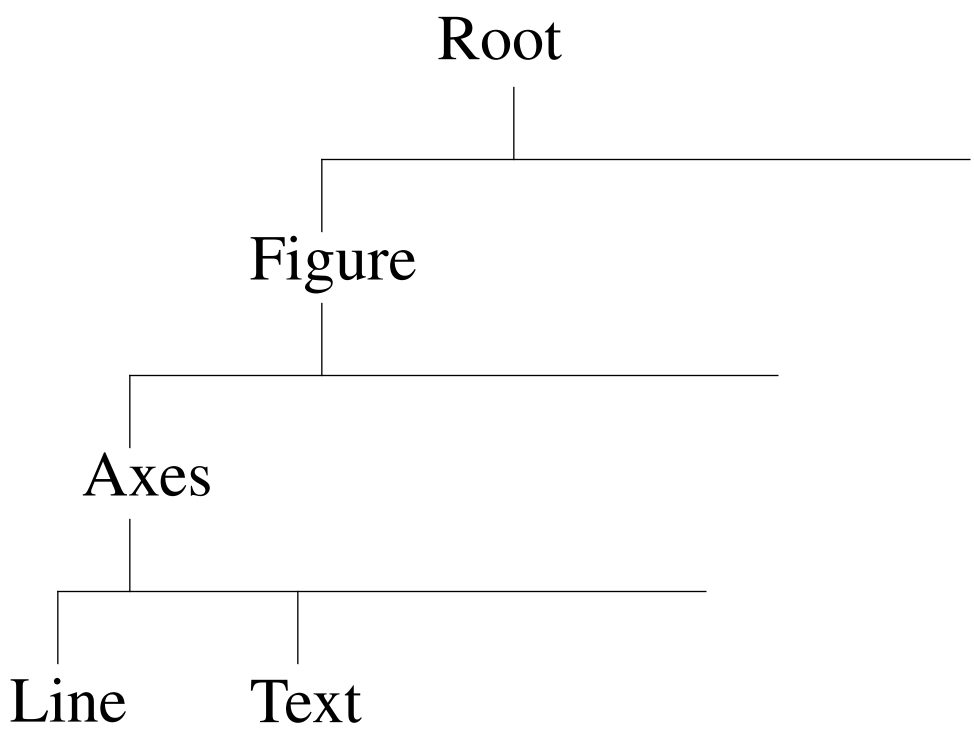
A script called from the keyboard executes in the base workspace.

A script called from within a function executes in the function's workspace.

Resetting Variables

clear x	removes variable x (and undoes global decl.)
clear all	clears everything
clear global	clears global declarations
whos global	show all global vars.

Handle Graphics




```
clf, plot(rand(5,3))

ax = get(gcf,'Children')
get(ax)

lines = get(ax,'Children')
get(lines(1))
```

Multiple Axes

```
clf

subplot(2,2,1)       % uses row-major order
plot(rand(5,5))

subplot(2,2,2)
bar3(rand(5,3))

subplot(2,2,3)
a = rand(15,1);
pie(a, a>0.7)

subplot(2,2,4)
pts = 0:pi/20:2*pi;
polar(pts,cos(2*pts))

set(gca,'Position',[0.32  0.1  0.4  0.4])
```

Exploring Graphic Objects

propedit(gcf)

Matlab Help Desk: click on
"Handle Graphics Property Browser"

set(gca,'Units')

3D Graphics

```
peaks

rotate3d  on



[x,y,z] = peaks;

surf(x,y,z,z)

surf(x,y,z,x)

surf(x,y,z,rand(length(x)))
```

Plotting in 3D

```
function helix

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

  x1 = cos(pts); y1 = sin(pts);
  x2 = cos(pts+pi); y2 = sin(pts+pi);
  z = pts/(2*pi);

  clf, whitebg(gcf,[0 0 0])
  hold on
  plot3(x1,y1,z,'y')
  plot3(x2,y2,z,'w')

  axis([-2 2 -2 2 0 2])
  view(95,9)
```

helix (cont.)

```
colors = 'rgbm';

for i = 4:4:length(pts)-4

  plot3([x1(i) x2(i)], [y1(i) y2(i)], z([i i]),  ...
      colors(ceil(rand(1)*length(colors))))

end
```

helix (cont.)

```
axis off
set(gcf,'Color','k')

set(gca,'CameraViewAngleMode','manual')

for az =  -180:5:80
  view(az,9)
  drawnow
end
```

Color Maps

```
clf
peaks

colorbar

m = colormap;
whos m

colormap(autumn)
brighten(0.5)

colormap(jet)

colormap(bone)

colormap(hot)
```

2D Data

```
[x,y] = meshgrid(-2:0.05:2);

z = sin(x).*cos(y);

contour(z,20)

imagesc(z)

imagesc(x(:),y(:),z)

surf(z)

surfc(z)
```

Surface Objects

```
sphere

[x,y,z]=sphere(30);

x(15,8)=NaN;

surf(x,y,z,y)

shading flat


surf(x,y,z,rand(31,31))

shading interp
```

Data From Files

Create a file temps.dat

```
38    50
42    53
33    57
45    56
44    46
41    40
```

load temps.dat

whos te*

plot(temps)

Polynomial Curve Fitting

```
load /afs/cs/usr/dst/expt1.dat
whos expt1

x = expt1(:, 1); y = expt1(:,2);

clf, hold on, plot(x,y,'o')

c = polyfit(x,y,3)
```

example polynomial representation:

c = [5 -1 4 3]

$$5x^3 - x^2 + 4x + 3$$

```
pts = min(x) : max(x);
plot(pts, polyval(c,pts), 'b')

helpwin polyfit

doc polyfit
```

Saving Variables

```
clear all

a = 'aardvark'

[x,y,z] = sphere(5);

save stuff.mat

whos -file stuff.mat


save junk.dat x y -ascii

type junk.dat
```

General OS Stuff

pwd

cd

dir

ls *.m

delete stuff.mat

!df

Debugging

Poor man's debugger:

Remove semicolons from assignments.

Add 'quoted strings' in appropriate places.

Add a call to **keyboard**. Use **dbcont** to return from keyboard input mode.

```
function y = buggy(vec)

 p = vec > 5
 'got this far'
 keyboard

 z = p*vec
 v = sin(z);
```

MATLAB Debugger

dbtype helix

dbstop helix 3

helix

dbstep

dbstep 5

whos

dbstep 30

dbquit

dbclear helix

help debug

Formatted Output

```
for i = 1:10
  fprintf('The square-root of %2d is %f\n', ...
      i, sqrt(i))
end
```

```
help fprintf
```

```
title(sprintf('f(x) over range %g to %g', ...
                  -3.5,  5.125))
```