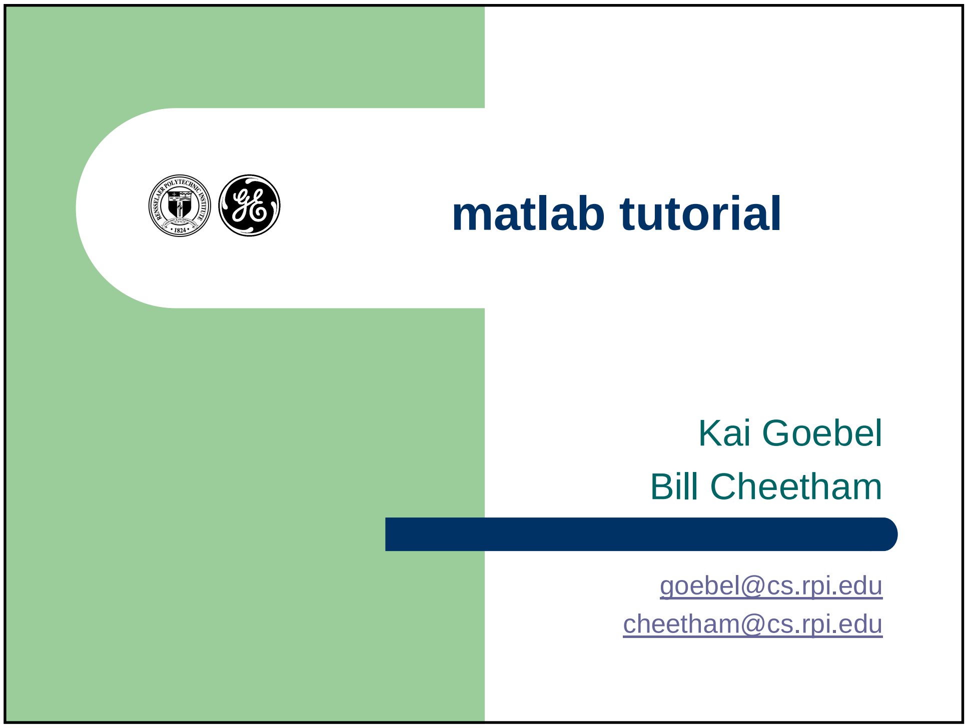
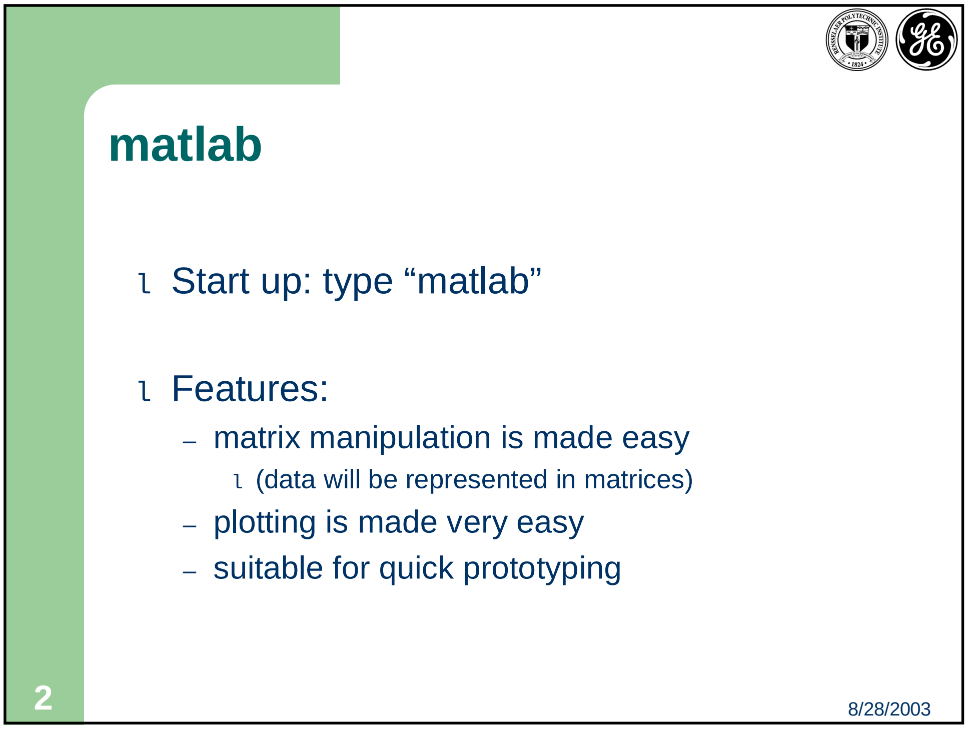


matlab tutorial

Kai Goebel
Bill Cheetham

goebel@cs.rpi.edu
cheetham@cs.rpi.edu



matlab

- Start up: type “matlab”
- Features:
  - matrix manipulation is made easy
    - (data will be represented in matrices)
  - plotting is made very easy
  - suitable for quick prototyping

2

8/28/2003

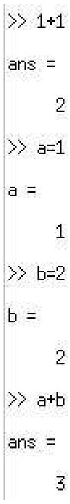
Matlab is installed on the RCS machines

- 1: log into an RCS account
- or 2: "attach" to the RCS AFS directory structure

  /usr/afsws/bin/klog <user-id-on-RCS>
- provide the appropriate RCS password.
- From that point, all software available on the RCS system should be available on the local machine.
- <u>Note</u>: this attachment needs only be performed once per login session (not for every command shell opened)

matlab tutorial

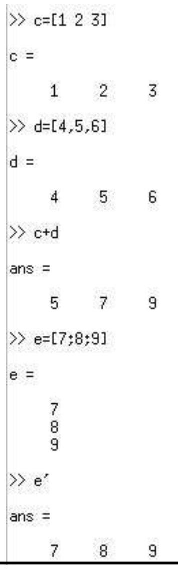
- - calculations
- - assignment of variables
- - manipulations of variables



```
>> 1+1

ans =

     2

>> a=1

a =

     1

>> b=2

b =

     2

>> a+b

ans =

     3
```

matlab tutorial (2)

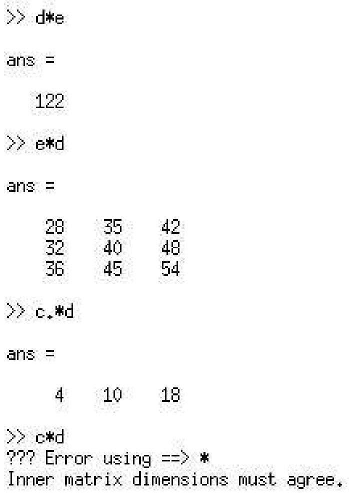
- vectors
- manipulation of vectors (addition)
- column vector
- transpose of vectors



```
>> c=[1 2 3]

c =

     1     2     3

>> d=[4,5,6]

d =

     4     5     6

>> c+d

ans =

     5     7     9

>> e=[7;8;9]

e =

     7
     8
     9

>> e'

ans =

     7     8     9
```

matlab (3)

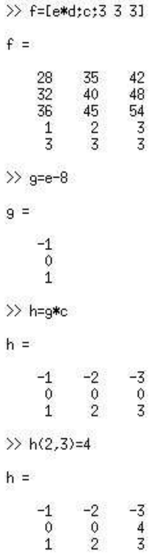
- vector multiplication
- dot product
- caveat:
  - check inner dimensions before operation



```
>> d*e

ans =

   122

>> e*d

ans =

    28    35    42
    32    40    48
    36    45    54

>> c.*d

ans =

     4    10    18

>> c*d
??? Error using ==> *
Inner matrix dimensions must agree.
```

matlab (4)

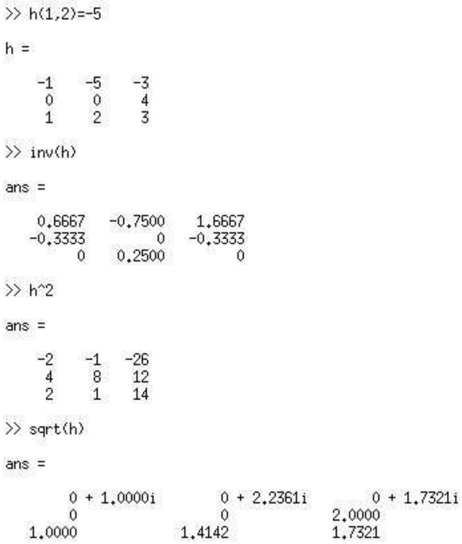
- creating matrices from existing ones
- subtracting a constant value from vectors
- accessing particular matrix elements



```
>> f=[e*d;c;3 3 3]

f =

    28    35    42
    32    40    48
    36    45    54
     1     2     3
     3     3     3

>> g=e-8

g =

    -1
     0
     1

>> h=g*c

h =

    -1    -2    -3
     0     0     0
     1     2     3

>> h(2,3)=4

h =

    -1    -2    -3
     0     0     4
     1     2     3
```

matlab (5)

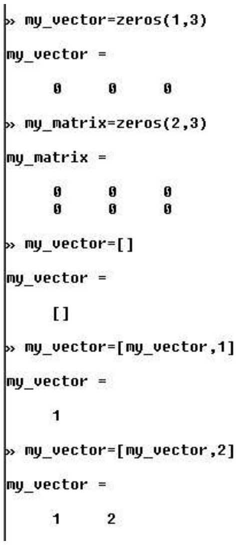
- inverting matrices
- matrix power of 2
- square root of a matrix



```
>> h(1,2)=-5

h =

    -1    -5    -3
     0     0     4
     1     2     3

>> inv(h)

ans =

    0.6667   -0.7500    1.6667
   -0.3333         0   -0.3333
         0    0.2500         0

>> h^2

ans =

    -2    -1   -26
     4     8    12
     2     1    14

>> sqrt(h)

ans =

        0 + 1.0000i        0 + 2.2361i        0 + 1.7321i
        0                  0             2.0000
   1.0000             1.4142             1.7321
```

matlab (6)

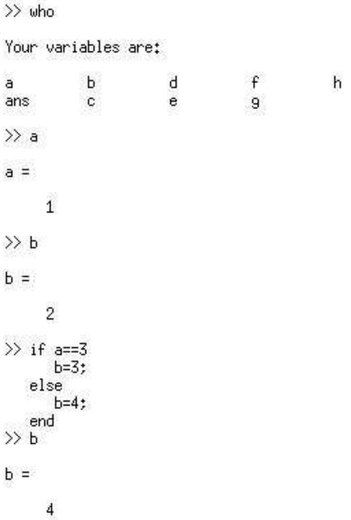
- Initializing vectors and matrices
  - known length
  - unknown length



```
>> my_vector=zeros(1,3)

my_vector =

     0     0     0

>> my_matrix=zeros(2,3)

my_matrix =

     0     0     0
     0     0     0

>> my_vector=[]

my_vector =

     []

>> my_vector=[my_vector,1]

my_vector =

     1

>> my_vector=[my_vector,2]

my_vector =

     1     2
```

matlab (7)

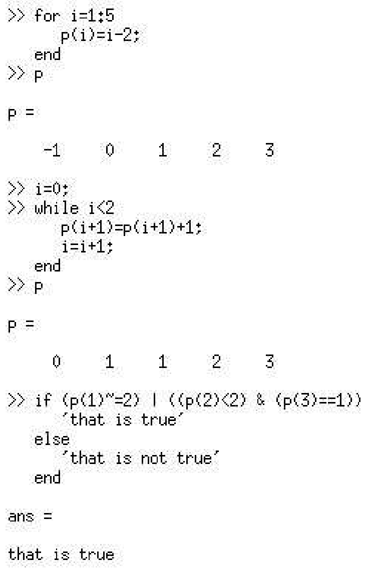
- checking your variables
- variables are stored until reassigned or when program is terminated
- IF THEN ELSE statements



```
>> who

Your variables are:

a         b         d         f         h
ans       c         e         g

>> a

a =

     1

>> b

b =

     2

>> if a==3
      b=3;
   else
      b=4;
   end
>> b

b =

     4
```

matlab (8)

- FOR loops
- WHILE loops
  (also note use of “<“)
- negation, OR, AND operators
  (also note prompt)

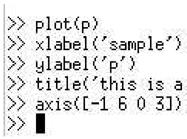


```
>> for i=1:5
      p(i)=i-2;
   end
>> p

p =

    -1     0     1     2     3

>> i=0;
>> while i<2
      p(i+1)=p(i+1)+1;
      i=i+1;
   end
>> p

p =

     0     1     1     2     3

>> if (p(1)~=2) | ((p(2)<2) & (p(3)==1))
      'that is true'
   else
      'that is not true'
   end

ans =

that is true
```

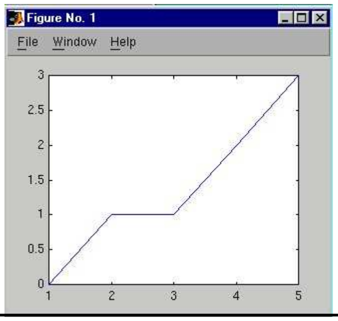
11

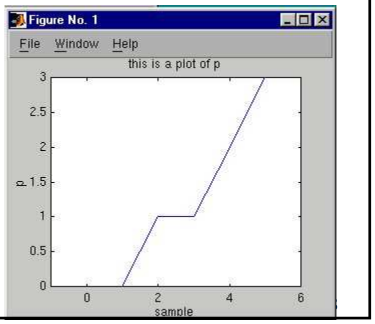
matlab (9)

- plotting data
- labeling axis
- adding title
- resizing data range



```
>> plot(p)
>> xlabel('sample')
>> ylabel('p')
>> title('this is a plot of p')
>> axis([-1 6 0 3])
>> 
```

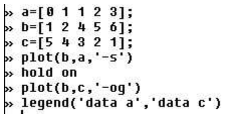




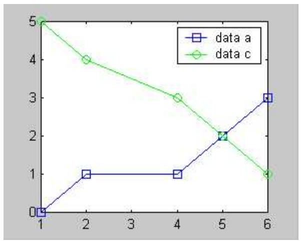
12

matlab (10)

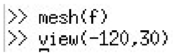
- Plotting one vector vs. the other
- …using different markers
- …and adding a legend



```
» a=[0 1 1 2 3];
» b=[1 2 4 5 6];
» c=[5 4 3 2 1];
» plot(b,a,'-s')
» hold on
» plot(b,c,'-og')
» legend('data a','data c')
```

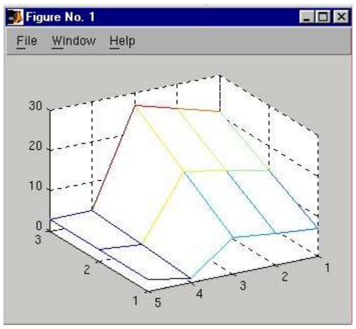


matlab (11)



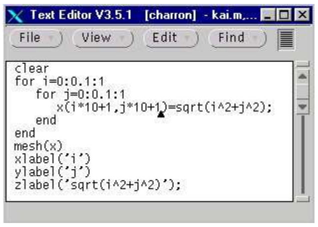
```
>> mesh(f)
>> view(-120,30)
```

- 3-D plots
- arrange viewing direction



matlab (12)

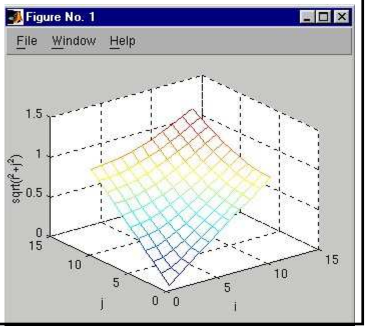
l save larger operations in text file



```
clear
for i=0:0.1:1
   for j=0:0.1:1
      x(i*10+1,j*10+1)=sqrt(i^2+j^2);
   end
end
mesh(x)
xlabel('i')
ylabel('j')
zlabel('sqrt(i^2+j^2)');
```

l invoke by typing filename without extension

```
>> kai
```

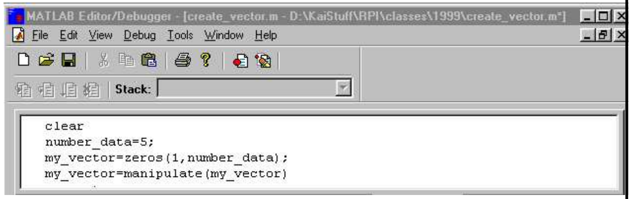


15

matlab (13)

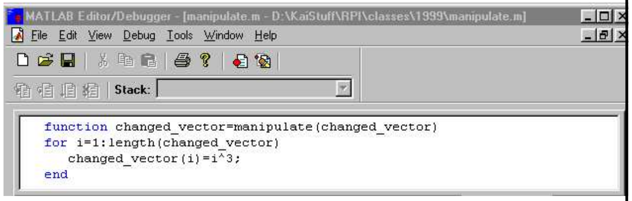
l using functions in different files

- for example: main function in create_vector.m

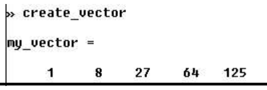

MATLAB Editor/Debugger - [create_vector.m - D:\KaiStuff\RPI\classes\1999\create_vector.m*]
File
Edit
View
Debug
Tools
Window
Help
Stack:

```
clear
number_data=5;
my_vector=zeros(1,number_data);
my_vector=manipulate(my_vector)
```

l subroutine in manipulate.m


MATLAB Editor/Debugger - [manipulate.m - D:\KaiStuff\RPI\classes\1999\manipulate.m]
File
Edit
View
Debug
Tools
Window
Help
Stack:

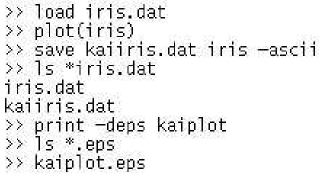
```
function changed_vector=manipulate(changed_vector)
for i=1:length(changed_vector)
   changed_vector(i)=i^3;
end
```



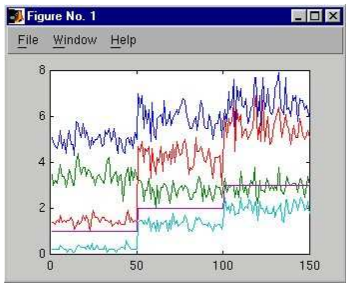
```
>> create_vector

my_vector =

     1     8    27    64   125
```

matlab (14)

- load and save data files
- use of system commands from within matlab
- print plots to file (or to printer)



```
>> load iris.dat
>> plot(iris)
>> save kaiiris.dat iris -ascii
>> ls *iris.dat
iris.dat
kaiiris.dat
>> print -deps kaiplot
>> ls *.eps
>> kaiplot.eps
```

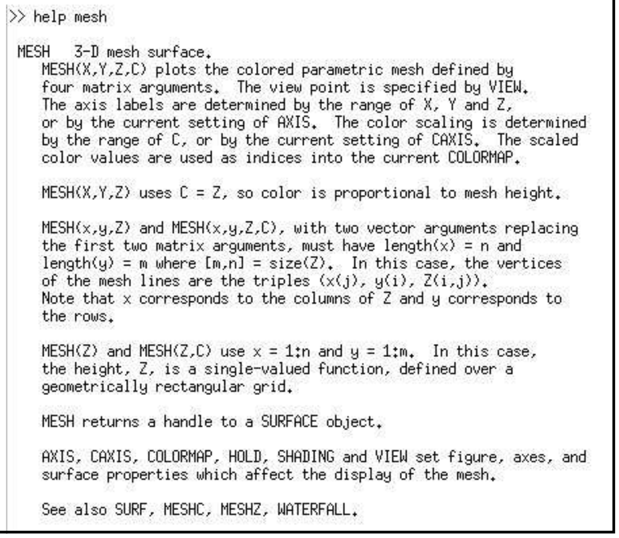


17

8/28/2003

matlab (15)

- HELP!

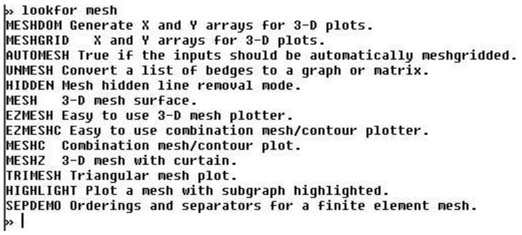


```
>> help mesh

 MESH   3-D mesh surface.
    MESH(X,Y,Z,C) plots the colored parametric mesh defined by
    four matrix arguments.  The view point is specified by VIEW.
    The axis labels are determined by the range of X, Y and Z,
    or by the current setting of AXIS.  The color scaling is determined
    by the range of C, or by the current setting of CAXIS.  The scaled
    color values are used as indices into the current COLORMAP.

    MESH(X,Y,Z) uses C = Z, so color is proportional to mesh height.

    MESH(x,y,Z) and MESH(x,y,Z,C), with two vector arguments replacing
    the first two matrix arguments, must have length(x) = n and
    length(y) = m where [m,n] = size(Z).  In this case, the vertices
    of the mesh lines are the triples (x(j), y(i), Z(i,j)).
    Note that x corresponds to the columns of Z and y corresponds to
    the rows.

    MESH(Z) and MESH(Z,C) use x = 1:n and y = 1:m.  In this case,
    the height, Z, is a single-valued function, defined over a
    geometrically rectangular grid.

    MESH returns a handle to a SURFACE object.

    AXIS, CAXIS, COLORMAP, HOLD, SHADING and VIEW set figure, axes, and
    surface properties which affect the display of the mesh.

    See also SURF, MESHC, MESHZ, WATERFALL.
```

18

matlab (16)

- Look for commands relating to a keyword



```
>> lookfor mesh
MESHDOM Generate X and Y arrays for 3-D plots.
MESHGRID   X and Y arrays for 3-D plots.
AUTOMESH True if the inputs should be automatically meshgridded.
UNMESH Convert a list of bedges to a graph or matrix.
HIDDEN Mesh hidden line removal mode.
MESH   3-D mesh surface.
EZMESH Easy to use 3-D mesh plotter.
EZMESHC Easy to use combination mesh/contour plotter.
MESHC  Combination mesh/contour plot.
MESHZ  3-D mesh with curtain.
TRIMESH Triangular mesh plot.
HIGHLIGHT Plot a mesh with subgraph highlighted.
SEPDEMO Orderings and separators for a finite element mesh.
>> |
```

- last slide