MATLAB tutorial for physicists

0011 0010 1010 1101 0001 0100 1011

Year 2009

Robert Scholten

Outline

- Why MATLAB?
- MATLAB through example:
  1. Signal processing: vectors, 2d plotting
  2. Image processing: arrays and images, Mandelbrot(?)
  3. Programming

Why MATLAB?

- Efficient
  - Very fast to learn
  - Calculate and visualise
- Powerful
  - Everything from ’hello world’ to GUI Widgets
- Ubiquitous
  - “Embedded” in many engineering (and science) books
- Free
  - University staff site license
- Valuable
  - Very broadly used in engineering and science

WEB resources

Mathworks homepage

www.mathworks.com

Mathworks support: documentation, links to tutorials and how-to guides, examples, etc etc

www.mathworks.com/support

Help!

- Online help: help

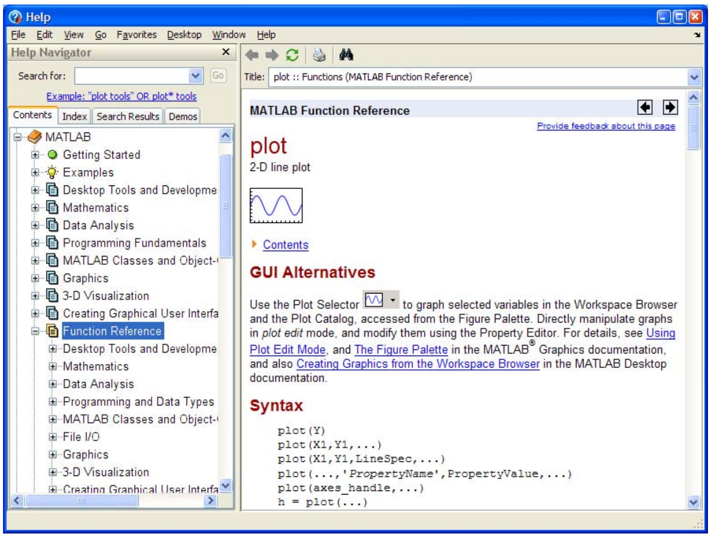
  Specific commands: `help plot` for help on `plot`

- Online documentation: doc

  Specific commands: `doc plot` for `plot` documentation

- Special help

```
help elfun
help specfun
help elmat
help ops
help general
```


Help
File Edit View Go Favorites Desktop Window Help
Help Navigator
Search for:
Go
Example: "plot tools" OR plot* tools
Contents
Index
Search Results
Demos
MATLAB
Getting Started
Examples
Desktop Tools and Developme
Mathematics
Data Analysis
Programming Fundamentals
MATLAB Classes and Object-
Graphics
3-D Visualization
Creating Graphical User Interfa
Function Reference
Desktop Tools and Developme
Mathematics
Data Analysis
Programming and Data Types
MATLAB Classes and Object-
File I/O
Graphics
3-D Visualization
Creating Graphical User Interfa

Title: plot :: Functions (MATLAB Function Reference)

MATLAB Function Reference

Provide feedback about this page

plot

2-D line plot

Contents

GUI Alternatives

Use the Plot Selector to graph selected variables in the Workspace Browser and the Plot Catalog, accessed from the Figure Palette. Directly manipulate graphs in *plot edit* mode, and modify them using the Property Editor. For details, see Using Plot Edit Mode, and The Figure Palette in the MATLAB® Graphics documentation, and also Creating Graphics from the Workspace Browser in the MATLAB Desktop documentation.

Syntax

```
    plot(Y)
    plot(X1,Y1,...)
    plot(X1,Y1,LineSpec,...)
    plot(...,'PropertyName',PropertyValue,...)
    plot(axes_handle,...)
    h = plot(...)
```

Executive commands

- Starting MATLAB

```
 scholten@astro01:/$ matlab          development environment
 scholten@astro01:/$ matlab -nojvm             classic style
```

- Keeping a diary

```
>> diary mysession
>> % do stuff here
>> diary off
```

… (can then edit `mysession` with text editor)

- Saving current session, to continue later

```
>> save thissession
```

… (leave Matlab)

… (restart Matlab)

```
>> load thissession    % recall (and re-do) stuff from diary
```

Sample MATLAB session

```
scholten@astro01:/$ matlab -nojvm
                              < M A T L A B >
                  Copyright 1984-2007 The MathWorks, Inc.
                         Version 7.5.0.338 (R2007b)
                               August 9, 2007

  To get started, type one of these: helpwin, helpdesk, or demo.
  For product information, visit www.mathworks.com.

>> a=5
a =
    5

>> b=[2 3 4]
b =
    2   3   4

>> a,b
a =
    5

b =
    2   3   4
>> c=a*b
C =
    10   15   20
>> exit

scholten@astro01:/$
```

Creates a vector

Creates another vector

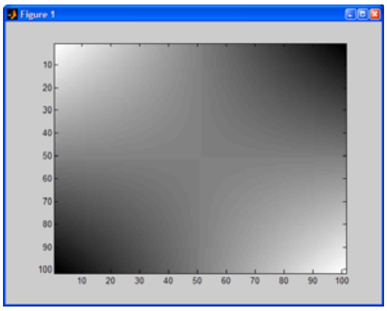
You do it…

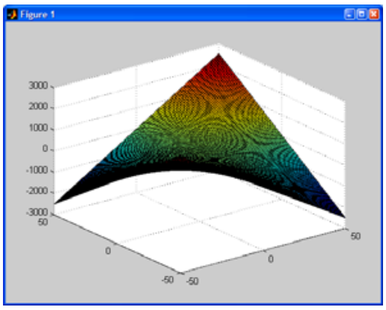
• Create an array	`x=[0 1 2 3 4 5 6 7 8 9];`
• Calculate a function of that	`sinx=sin(x/5.0);`
• Show what’s there	`sinx`
• Manipulate a bit	`'x * sinx is',x'*sinx`
• Plot that	`plot(x,sinx)`
• Get some help on plot	`help plot`

Vectors, arrays, matrices

- Create an array `x=zeros(10)`
- Or pre-indexed `x=-50:50`
- 2d `xy=ones(2,50)`
  `xy=[x x]`
- Matrix multiply `xx=x' * x`
- Show what's there `colormap(gray); imagesc(xx);`
  `colormap(jet);surf(x,x,xx);`

```
>> x=[x 51 52 53]     ; %add a few elements to x
>> x(3),x(3:9)        ; %array subscripts, ranges
```





Main Level Programs

In your favourite editor (emacs, nano) or in MATLAB itself, make a file called

`fred.m`:

```
a=6;
b=[1 2 3];
a,b
c=a*b;
c
```

In MATLAB:

```
>> fred
C =
       6       12       18
```

Sample MATLAB program: chaos, logistic map

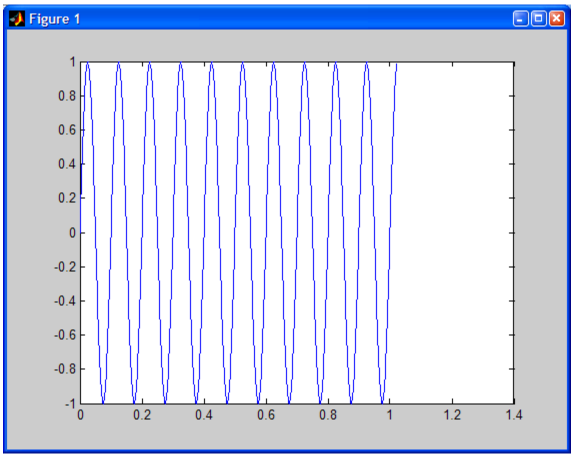
```
pin=input('Enter initial population ')
rin=input('      rate ')
numit=input('      number of iterations ')
p=pin
r=rin

for i=1:numit
  pnp= p+r*p*(1.0-p)
  p=pnp
end
```

Problem 1: Signal processing

```
% fabricate some simple waveform
delt=0.001; % time resolution
npts=1024;  % number of time data points (should be power of 2)
t=0:(npts-1);
t=t*delt;
f1=10*(2*pi); % frequency of waves
s1=sin(f1*t);
plot(t,s1)
```

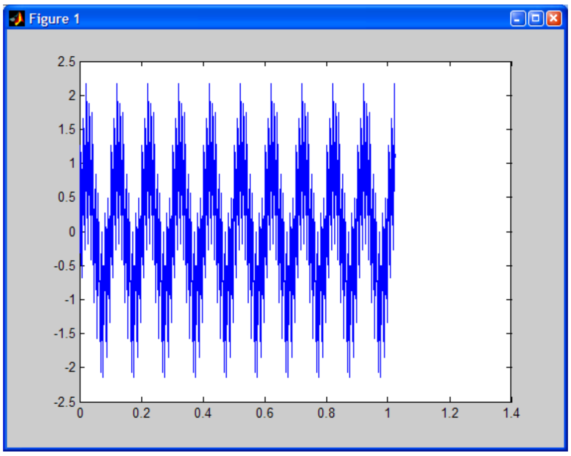
Creates a vector:
`[0,1,…,npts-1]*delt`

Creates another vector:
`[sin(0),sin(f1*delt),…]`



Signal processing

```
% add some more components
f2=100*(2*pi); s2=0.5*sin(f2*t+0.1*pi);
f3=350*(2*pi); s3=0.8*sin(f3*t-0.2*pi);
s=s1+s2+s3;
plot(t,s)
```

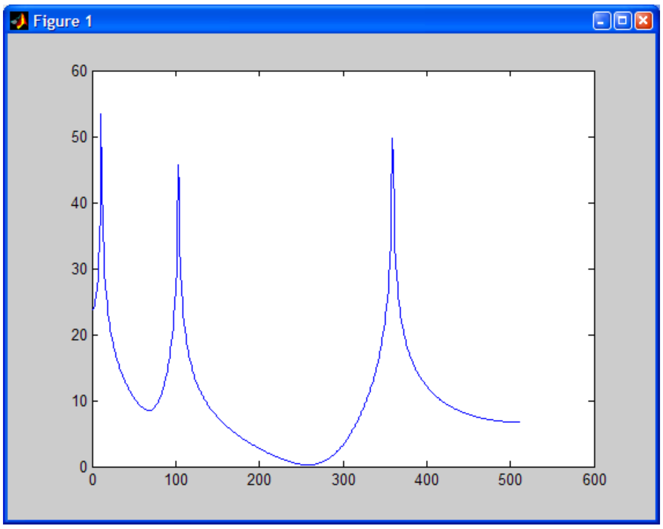


Export (sound file)

```
; save to sound file
rate=1/delt;
wavwrite(s/2,rate,16,'tute.wav')
```

Signal processing (FFT)

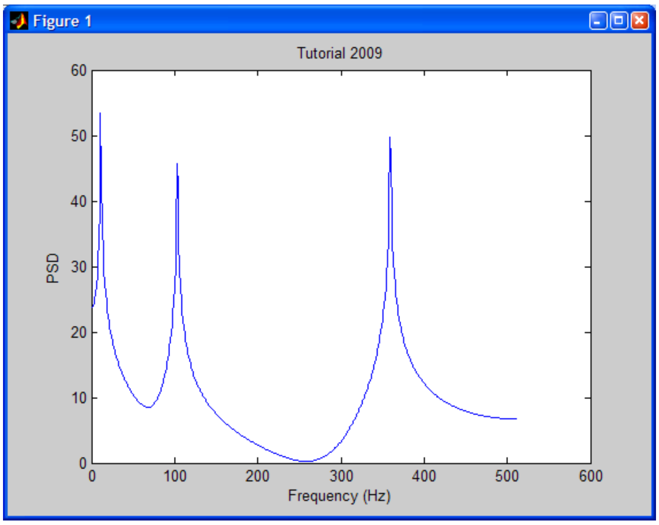
```
; fourier analyse
spec=fft(s);
delf=1.0/delt;
f=(1:npts/2)* delf;
mag=abs(spec(1:npts/2));
plot(f/1000.0,10*log10(mag.^2));
```



Signal processing…

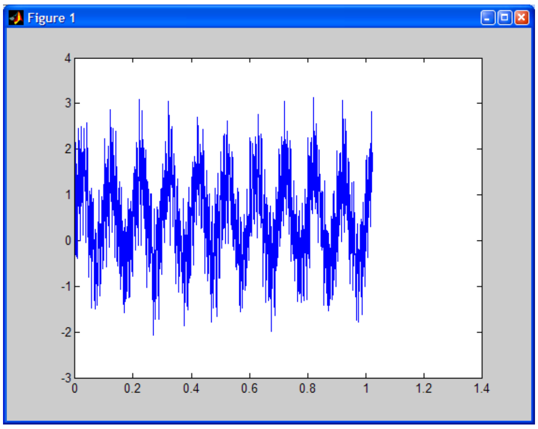
```
; make plot a little prettier
xlabel('Frequency (Hz)')
ylabel('PSD')
title('Tutorial 2009')
```

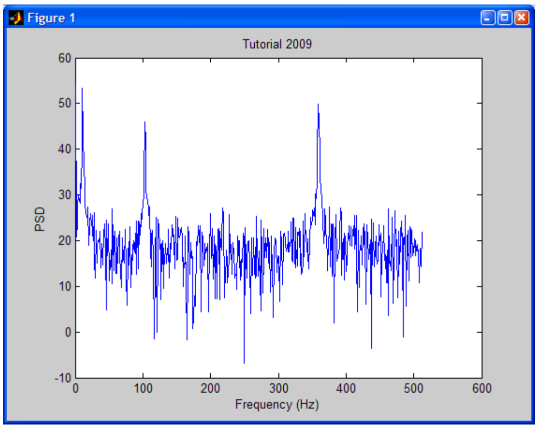
NOTE: use **`plottools`** command if on a graphics terminal



Signal processing…

```
; add some noise
noise=rand(1,npts);
sn=s+noise;
plot(t,sn);
specn=fft(sn);

mag=abs(specn(1:npts/2));
plot(f/1000.0,10*log10(mag.^2));
xlabel('Frequency (Hz)'); ylabel('PSD'); title('Tutorial 2009')
```

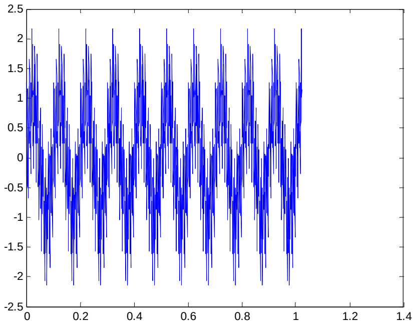


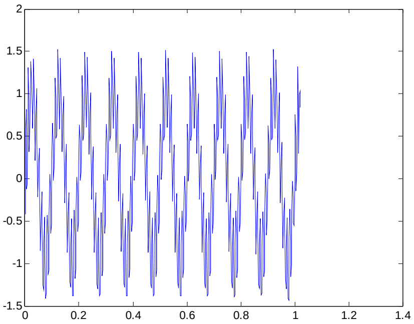


Frequency domain filtering

```
% frequency domain filtering, 20dB suppression around f0
f0=350*delf; fsig=25*delf;
filter=ones(1,npts/2) %using only half spectrum - throw away phase
fpts=find(abs(f-f0) < fsig)
filter(fpts)=filter(fpts)*1e-4
filtspec=spec(1:npts/2).*filter
sigfiltered=ifft(filtspec)
plot(t,s1+s2)
plot(t(1:npts/2)*2,sigfiltered)
```

help find



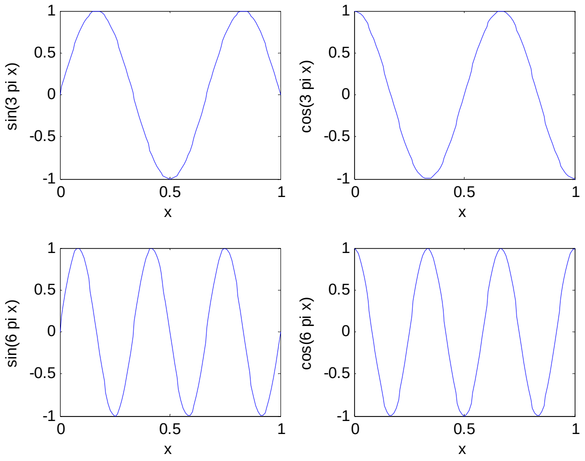


Multi-plots (several curves on one set of axes)

```
plot(s,t)
title('Multi-plot')
xlabel('x-axis'), ylabel('y-axis')
grid

% now add a curve
hold on
plot(x,y)
hold off

print –deps fig1      % print to .eps file

% hold off does not erase; can do that with
clf    % clear figure
```

Multiple plots (several separate sub-plots)

```
% split graphics window into m x n array of sub-windows
% subplot(mnX) where X is the window to select
% 1 = top left, 2 = next to right, etc.
x=linspace(0,1,101);
y=sin(3*pi*x);
subplot(221), plot(x,y)
  xlabel('x'),ylabel('sin(3 pi x)')
subplot(222), plot(x,cos(3*pi*x))
 xlabel('x'),ylabel('cos(3 pi x)')
subplot(223), plot(x,sin(6*pi*x))
 xlabel('x'),ylabel('sin(6 pi x)')
subplot(224), plot(x,cos(6*pi*x))
 xlabel('x'),ylabel('cos(6 pi x)')
```



Programming: functions

In MATLAB, functions are kept in separate .m files

Create a file such as area.m with code such as:

```
    function [A] = area(a,b,c)
    % compute the area of a triangle with sides of length a,b,c
    % output A=area
    % Usage: Area=area(2,3,4);
    S=(a+b+c)/2;
    A=sqrt(s*(s-a)*(s-b)*(s-c));
    %%% end of area %%%
```

Note that “s” is a local variable and cannot be accessed outside the function.

The help command will produce the leading comments from the file:

```
>> help area
    compute the area of a triangle with sides of length a,b,c
    output A=area
    Usage: Area=area(2,3,4);
>> Area=area(10,15,20)
Area =
   72.6184
```

Programming: while and for loops

```
S = 1; n = 1;
while S + (n+1)^2 < 100
  n = n+1; S = S + n^2;
end
>> [n, S]
Ans =
     6    91
```

```
N=10;
for R = 1:N
  for C = 1:N
    A(R,C) = 1/(R+C-1);
  end
end
```

Programming: if … else… elseif… end

```
if a >= c
  b = sqrt(a^2 - c^2)
  elseif a^c > c^a
    b = c^a/a^c
  else
    b = a^c/c^a
  end
b =
   0.2347
```

`==`	equal
`~=`	not equal
`<`	less than
`<=`	less than or equal to
`>`	greater than
`>=`	greater than or equal to
`~`	not

Reading and writing data files

```
fid=fopen('table.dat','r'); % open a file for reading
a=fscanf(fid,'%3d%4d');      % read pairs of numbers
fclose(fid);                 % close the file

fprintf(fid,'format',variables);  % write to file
```

The format is compatible with C. An example is:

```
'%6.2f  %12.8f\n'
```

for two numbers, of widths 6 and 12 characters, 2 and 8 decimal places.
Use **doc** for more info.