An introduction of Matlab for scientific computations

Dept. of Applied Math. NCTU

Chin-Tien Wu

05/27/2006

Matrices and vectors

Matrix and vector construction:

```
A = [1 2 3; 4 5 6; 7 8 0];  b=[1;2;-1];c=[1:2:10];
A(1,2);A(3,3); b(2); d=c(2:4);e=c([1,3,5]);
A(1:2,2:3);A([1,3],[1,3]);
Tril(A);Tril(A,-1);Tril(A,-2);
Triu(A);Triu(A,-1);Triu(A,-2);
Diag(A,0);Diag(A,1);Diag(A,-1);
n=5;m=10;
A=zeros(n,m);
A=ones(n,m);
A=eye(n);
A=diag(c,0);B=diag(b,2);A+B;
A=rand(n);A=randn(n,m);
A=magic(n);
A=sparse(d,e,b);
```

Matrix and vector operations:

+ addition
-subtraction
*** multiplication**
^ power
' transpose
\ left division
/ right division

A\b is the solution of A*x=b
d/A is the solution of x*A=d
A=[$a_{i,j}$];A'=[$a_{j,i}$];

A^2=A*A; A.^2=[$a_{i,j}^2$]

[1,2,3,4].*[1,2,3,4]

[1,2,3,4].^2

If & switch control and relations

```
if A>B
   ‘greater’
elseif A<B
   ‘less’
elseif A==B
   ‘equal’
else
   error(‘A and B are different data’)
end
```

```
switch sign(A-B)
   case 0
        ‘A=B’
   case 1
       ‘A>B’
   case -1
       ‘A<B’
   otherwise
   error(‘A and B are different data type’)
end
```

Relations: The relational operators in MATLAB are

- **< less than, > greater than**
- **<= less than or equal,**
- **>= greater than or equal**
- **== equal, ~= not equal.**
- **& and, | or, ~ not**

Built-in functions and numbers:

- **pi=3.141592…..**
- **eps=2^{-52}**
- **realmin=2^{-1022}**
- **realmax=(2-eps)2^{1023}**
- **Inf = infinity**
- **Nan = not a number (0/0, inf-inf)**

For and while loops

```
x = [];
for i = n:-2:1
    x = [x,i^2 ];
end
```

Continue: jump to next iteration
(i.e. skip the lines between the “continue” and the “end” lines)
Break: exit the for or while loop.

```
for i = 1:m
     for j = n:-1:1
          H(i, j) = 1/(i+j-1);
     end
end
```

```
a=10000;n = 0;z=1
while z < a
    n = n + 1;
    if mod(n,2)==1
              continue
    end
    z=2^n-1;
    if ~isprime(z)
        break
    end
end
```

```
n = 0;
while 2^n < a
    n = n + 1;
end
```

Matlab Functions

```
function [mean, stdev ]= stat(x)

% STAT Mean and standard deviation
% For a vector x, stat(x) returns the
% mean and standard deviation of x.
% For a matrix x, stat(x) returns two
%row vectors  containing,
%respectively, the mean and standard
%deviation of each column.

[m n] = size(x);
if m == 1
   m = n;  % handle case of a row vector
end
mean = sum(x)/m;
stdev = sqrt(sum(x.^ 2)/m - mean.^2);
```

```
function a = gcd(a,b)

%GCD Greatest common divisor
% gcd(a,b) is the greatest common
%divisor of  the integers a and b, not
%both zero.

a = round(abs(a)); b = round(abs(b));
if a == 0 & b == 0
     error('The gcd is not defined
when… both numbers are zero')
else
     while b ~= 0
          r = rem(a,b);
          a = b; b = r;
     end
end
```

Some useful Routines

- **Root finding: fzero(@myfunc, x0)**
- **Optimization: fminsearch(@myfunc, x0)**
- **Optimization under constraints:**
  fmincon(@myfun,x0,A,b,Aeq,beq,lb,ub)
  (Minimize myfun with constraints: A*x<=b, Aeq*x=beq and lb<=x<=ub)

```
y=@(x) 1./((x-.3).^2+.01)+…
          1./((x-.9).^2+.04)-6
x=0:0.002:1;
plot(x,y(x),'b-')
P=fminsearch(y,.5)
Q=fzero(y,.5)
hold
plot(P, y(P), 'r*')
```

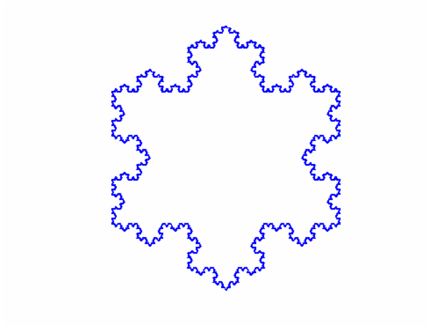
```
x0=rand([1 2]); %x0=[0.5,0.5];
A=[2 1]; b=[4];
Aeq=[]; beq=[];
lb=[0 0]'; ub=[3 3]';
[x,fval]=fmincon(@myfun,x0,…
                    A,b,Aeq,beq,lb,ub)

function f=myfun1(x)
    f=-(x(1).^2+x(2).^2-4.*x(1).*x(2)+1);
end
```

Function recursive calls

(Koch Snow flower)

```
P=[0 0; 0.5 3^0.5/2; 1 0;0 0];
N=size(P,1)-1;
figure;hold
for i=1:N;
    koch(P(i,:),P(i+1,:),3);
end
```



```
function koch(p,q,n)
if (n==0) plot([p(1);q(1)], [p(2);q(2)], 'LineWidth',4,'Color','red');
    hold on;
else
    c = q-p; c = [-c(2); c(1)];
    c = (p+q)/2 + c/sqrt(12);
    a = (2*p+q)/3;  b = (p+2*q)/3;
    koch(p, a, n-1);
    koch(a, c, n-1);
    koch(c, b, n-1);
    koch(b, q, n-1);
end
```

2D and 3D plots

2D

- ezplot('x^2+y^2-4')
- ezplot('sin(x)/(1+x^2)‘,[0,5])
- ezpolar('sin(t)/t' , [-6*pi,6*pi])
- fplot('[cos(x), 1-x^2/2, 1-x^2/2+x^4/24]', [-pi,pi])
- x=-pi:.1:pi; y=-pi:.2:pi; Y=[sin(x)', sin(2*x)', sin(4*x)']; plot(x,Y)

3D

- ezplot3('cos(2*pi*t)','sin(2*pi*t)','t',[0,4])
- ezsurf('x*exp(-x^2 - y^2)')
- ezsurf('s*cos(t)', 's*sin(t)', 't',[.4,1],[0,6*pi])
- ezsurfc('x*y*exp(-(x^2+y^2))' ,20)
- [X,Y]=meshgrid(x,y); Z=X.*Y.*exp(-(X.^2+Y.^2));[Zx,Zy]=gradient(Z,.1,.2); mesh(x,y,Z); contour(x,y,Z,20); quiver(X,Y,Zx,Zy);

Simple I-O for files

- **Variables saving: save variables filename**
- **Variables loading: load filename**
- **File openning:**

```
fid= fopen(filename, permission)
     permission='w','r' or 'a' …, etc.
     %% 'w' meaning the file is writable    %%
     %% 'r'  meaning the file is read only  %%
     %% 'a' meaning the file is appended %%
```

- **File closing: fclose(fid)**
- **File reading:**

```
[A,count]=fscanf(fid,format,size)
format='%s','%d' or '%f', …, etc.
s: string, d: integer, f: floating point
```

- **File writing:**

```
fprintf(fid,format,A)
```

An example of file I-O

```
fid = fopen('magic5.bin','w')
fprintf(fid,'%5d %5d %5d %5d%5d\n',magic(5))
fclose(fid)
fid=fopen('magic5.bin','r')
B=fscanf(fid,'%5d',[5,5])
B=B+randn(5);
fclose(fid)
fid = fopen('magic5.bin','w')
fprintf(fid,'%6.2f %6.2f %6.2f %6.2f %6.2f\n',B)
fclose(fid)
fid=fopen('magic5.bin','r')
clear B
[B,count]=fscanf(fid,'%f',[5,5])
fclose(fid)
```

Simple I-O for Audio

Y=WAVRECORD(N,FS,CH,DTYPE)

records N audio samples at FS Hertz from CH number of input channels.
Fs=8000, 11025, 22050, and 44100 Hz.
CH=1 (mono) or 2 (stereo)
DTYPE= 'double‘, 'single' , 'int16' or 'uint8'

WAVWRITE(Y,FS,NBITS,FILENAME)

NBITS=16, 16, 16 or 8

WAVPLAY(Y,FS)

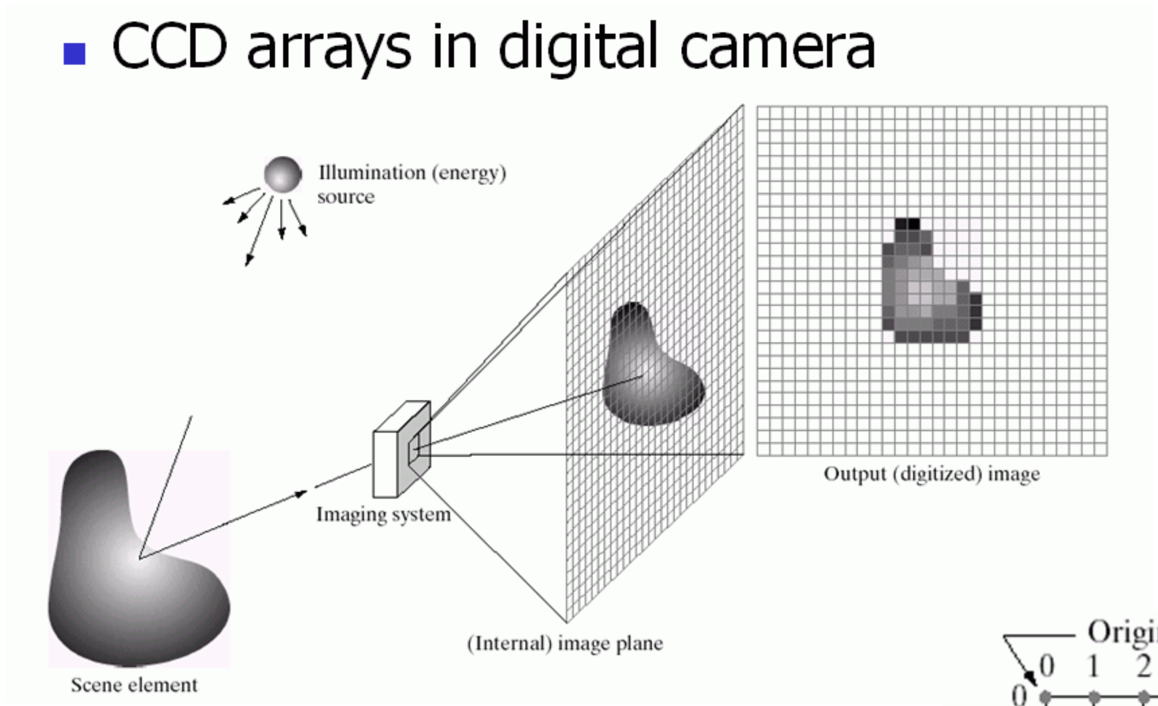
Y=WAVREAD(FILENAME)

An example of audio I-O

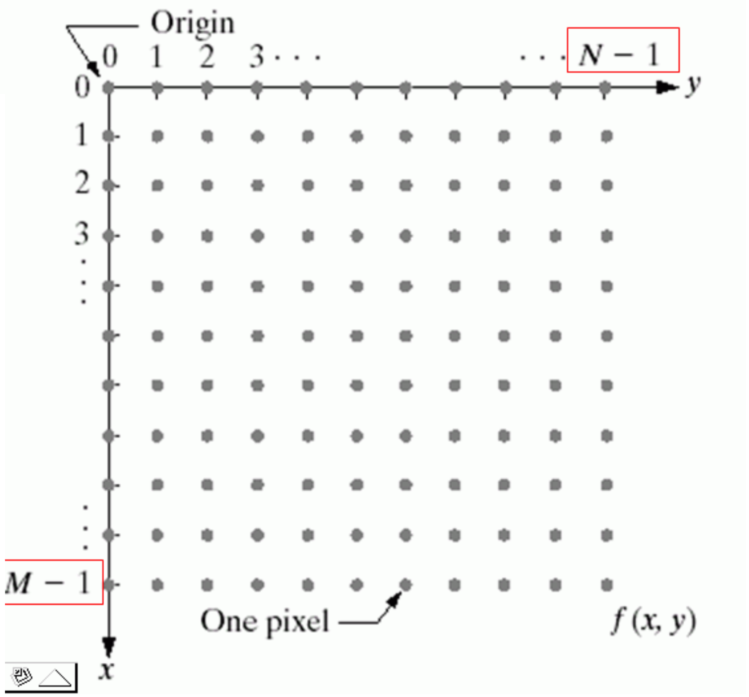
Example: Record and play back 5 seconds of 16-bit audio sampled at 11.025 kHz.

```
Fs = 8000;
y  = wavrecord(5*Fs, Fs, ‘single');
wavplay(y, Fs);
wavwrite(y,Fs,16,’howareyou’)
yy=wavread(‘howareyou’)
wavplay(yy,Fs)
```

- CCD arrays in digital camera



Physical representation of an image



Mathematical representation of an image

Simple I-O for image

[A,map] = IMREAD(FILENAME,FORMAT)

IMWRITE(A,FILENAME,FORMAT)

FORMAT=‘jpg’,’tiff’,’bmp’,…etc.

IMAGE(A) or IMSHOW(A) : displays matrix A as an image.

M=GETFRAME : Get movie frame

M = IM2FRAME(A,MAP)

MOVIE(M,N): plays the movie in array M N-times

MOVIE2AVI(M,FILENAME) : Create AVI movie from MATLAB movie

An example of image I-O

```
eskimo=imread(‘Eskimo.jpg');
eskg=rgb2gray(eskimo);
subplot(2,2,1); imshow(eskg)
imwrite(eskg,’eskimo-gray.jpg’,’jpg’)
eskimog=imread(‘eskimo-gray.jpg');
subplot(2,2,2); imag(eskimog)
h = fspecial('motion'); % h = fspecial('motion',50,45);
eskimog_mo= imfilter(eskimog,h);
subplot(2,2,3); Imshow(eskimo_mo)
eskimog_eq=histeq(eskimog)
subplot(2,2,4);imshow(eskimog_eq)
```

fspecial(TYPE): special effect for image

TYPE:	'average'	averaging filter
	'disk'	circular averaging filter
	'gaussian'	Gaussian lowpass filter
	'laplacian'	2-D Laplacian filter
	'unsharp'	unsharp contrast enhancement filter

Some more applications (I)

Fourier transform:

transfer data from time domain to frequency domain

$$f\left(x\right) \approx \sum_{k=-n/2}^{n/2} c_k e^{ik\pi x},$$

$$\text{where } e^{ik\pi x} = \cos\left(k\pi x\right) + i\sin\left(k\pi x\right)$$

FFT: given n data points $f(x_1)\ldots f(x_n)$, find the associated c_k, k=1..n

IFFT: input c_k, k=1..n, and recover f.

%%% Example (1) FFT in signal processing %%%

```
t = 0:.001:.25;                                   % total data points=250,
x = sin(2*pi*50*t) + sin(2*pi*120*t);             % frequency=1/0.001
z= 2*randn(size(t));
y = x +z;
figure
subplot(3,1,1);plot(x(1:50)); title(‘original signal')
subplot(3,1,2);plot(z(1:50));title('Noisy signal');
subplot(3,1,3);plot(y(1:50));title(‘mixed signal');
Y = fft(y,256);Yi=ifft(Y);                         % 256 frequency modes
figure;plot(y,’bx-’);hold;plot(Yi,’ro-’);          % to represent data
PY= Y.*conj(Y)/256;                                % amplitudes of frequency modes
f = 1000/256*(0:127);
figure
plot(f,PY(1:128));title('Power spectral density');xlabel('Frequency (Hz)')
plot(f(1:50),PY(1:50));title('Power spectral density');xlabel('Frequency (Hz)')
X1=zeros(size(Y));X2=X1;
lf_index=find(PY>0.5*max(PY));hf_index=find(PY<=.5*max(PY));
X1(lf_index)=Y(lf_index);X2(hf_index)=Y(hf_index);
Y1=ifft(X1,256);Y2=ifft(X2,256);
figure
plot(x(1:50),’bx-’);hold; plot(Y1(1:50),’r-o’);
```

```
%%% Example(2) FFT in audio processing%%%

Fs = 8000;num_sec=3;            %Fs: sampling frequency
                                %num_sec: length of the input audio
y=wavread('howareyou_nos');
wavplay(y,Fs)
fftnum=num_sec*Fs;
Y=fft(y);
PY=Y.*conj(Y)/(fftnum);
f=Fs/(fftnum)*(0:0.5*fftnum-1);
figure
subplot(2,1,1);plot(y)
subplot(2,1,2);plot(f,PY(1:0.5*fftnum))
lf_index=find(PY(1:fftnum)<5); hf_index=find(PY(1:fftnum)>=5);
X1=zeros(size(Y));X1(lf_index)=Y(lf_index);
X2=zeros(size(Y));X2(hf_index)=Y(hf_index);
Y1=ifft(X1, fftnum);Y2=ifft(X2, fftnum);
wavplay(Y1);
wavplay(Y2);
figure;plot(Y1+Y2);
```

```
%%% Example (4) Image processing %%%

load optdeblur P
 [m,n] = size(P); mn = m*n;
 imshow(P);
 title(sprintf('%i x %i (%i pixels) ',m,m,mn));
blur = 5;  mindex = 1:mn;  nindex = 1:mn;
for i = 1:blur
  mindex=[mindex i+1:mn 1:mn-i];
  nindex=[nindex 1:mn-i i+1:mn];
end
D = sparse(mindex,nindex,1/(2*blur+1)); x=reshape(P,mn,1);
G = D*x;
imshow(reshape(G,m,n))
% min( | D*P(:) - G(:) |^2 + 0.0004*| L*P(:) |^2 ) with 0<=P<=1
L = sparse( [1:mn,2:mn,1:mn-1],  [1:mn,1:mn-1,2:mn], ...
   [4*ones(1,mn) -1*ones(1,2*(mn-1))]  );
A = [D ; 0.02*L]; b = [ G(:) ; zeros(mn,1) ];
options = optimset('LargeScale', 'on','Display', 'off' );
x = lsqlin(A, b, [], [], [], [], zeros(mn,1), ones(mn,1), [], options);
imshow(reshape(x,m,n))
subplot(1,2,1);imshow(reshape(G,m,n));
subplot(1,2,2);imshow(reshape(x,m,n));
```

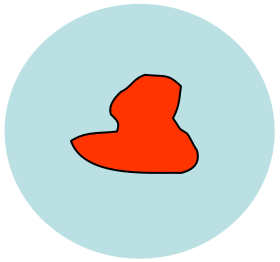
Some more applications (II)

Solving ordinary differential equations (ODE):

Example (5): a simple model for the spread of a disease

$$\frac{du}{dt} = ku(t)\big(u(t)-1\big),\ t \geq 0$$

$$u(t) = \frac{red}{blue} \text{ at time } t$$



Red: population of infected animal
Blue: population of healthy animal

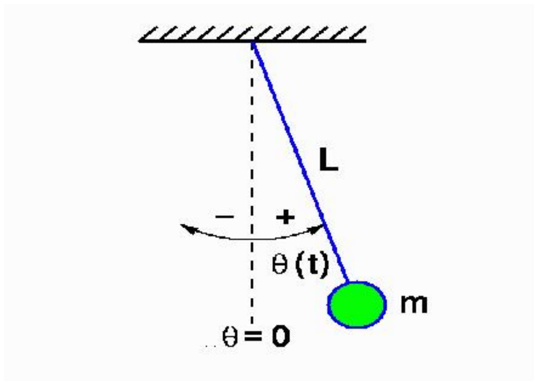
Example (6): Pendulum motion

$$\frac{d^2}{dt^2}\theta(t) + \sin\big(\theta(t)\big) = 0.$$

$$\Downarrow$$

Let $\omega(t) = \frac{d}{dt}\theta(t),$

$$\frac{d}{dt}\begin{bmatrix}\theta(t) \\ \omega(t)\end{bmatrix} = \begin{bmatrix}\omega(t) \\ -\sin\big(\theta(t)\big)\end{bmatrix}$$



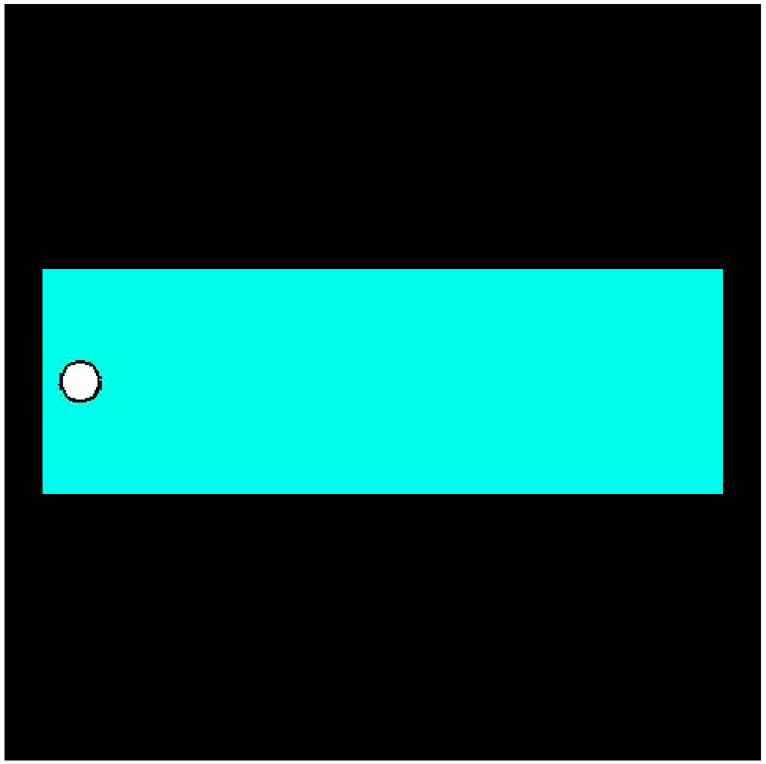
Example (5) Solving ODE

```
[ t, u ] = ode45( 'disease', [0,100] , 1/1000  );
figure; plot( t,u,'o');
options = odeset('abstol',1.e-9,'reltol', 1.e-9) ;
[tt,uu] = ode45('disease', [0:.05:100], 1/1000,options);
figure; plot( t,u,'x', tt,uu,'-' )
[t,Y]  = ode45( 'pendulum', [0:.25:20], [pi/4;0] );
% Solve ODE and plot
u  =  Y(:,1)  ;
figure; plot(  t, u,'-' )
mov=plot_pendulum(u,64);
movie(mov)
```

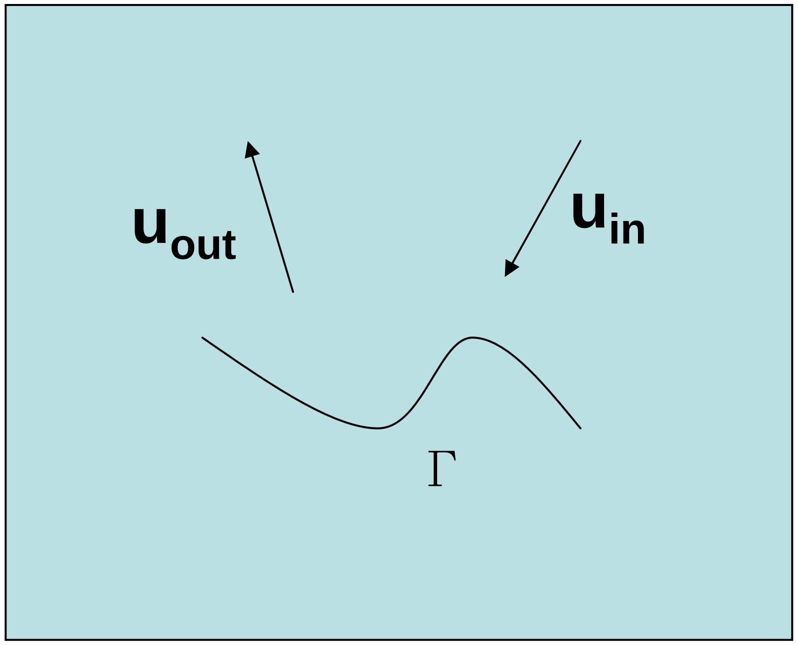
You can do much more!

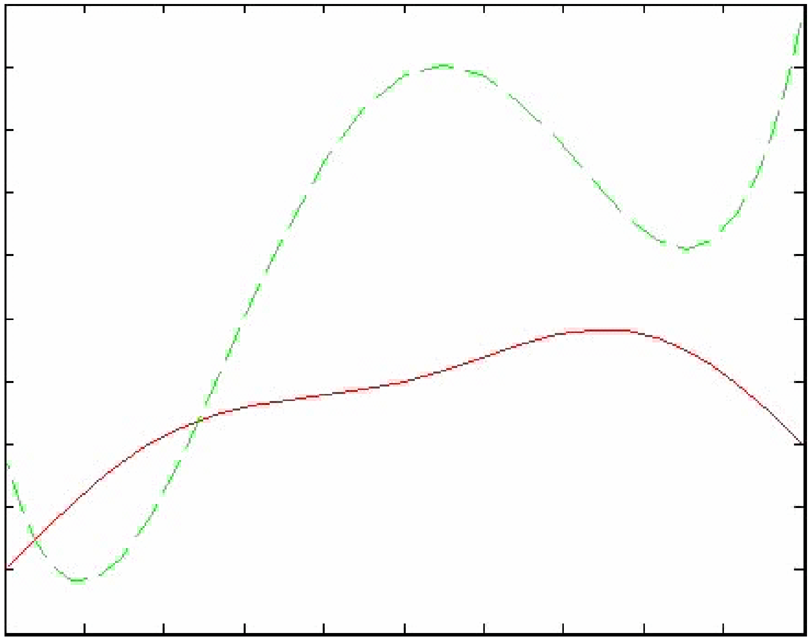
- Solving partial differential equations (fluid, structure and electronic-magnetic wave simulations)
- Solving integral equations (medical images, and radar and sonar detection,…,.etc)
- Many toolboxes available for a wide range of applications.

Karman Vortex street simulation



Crack detection



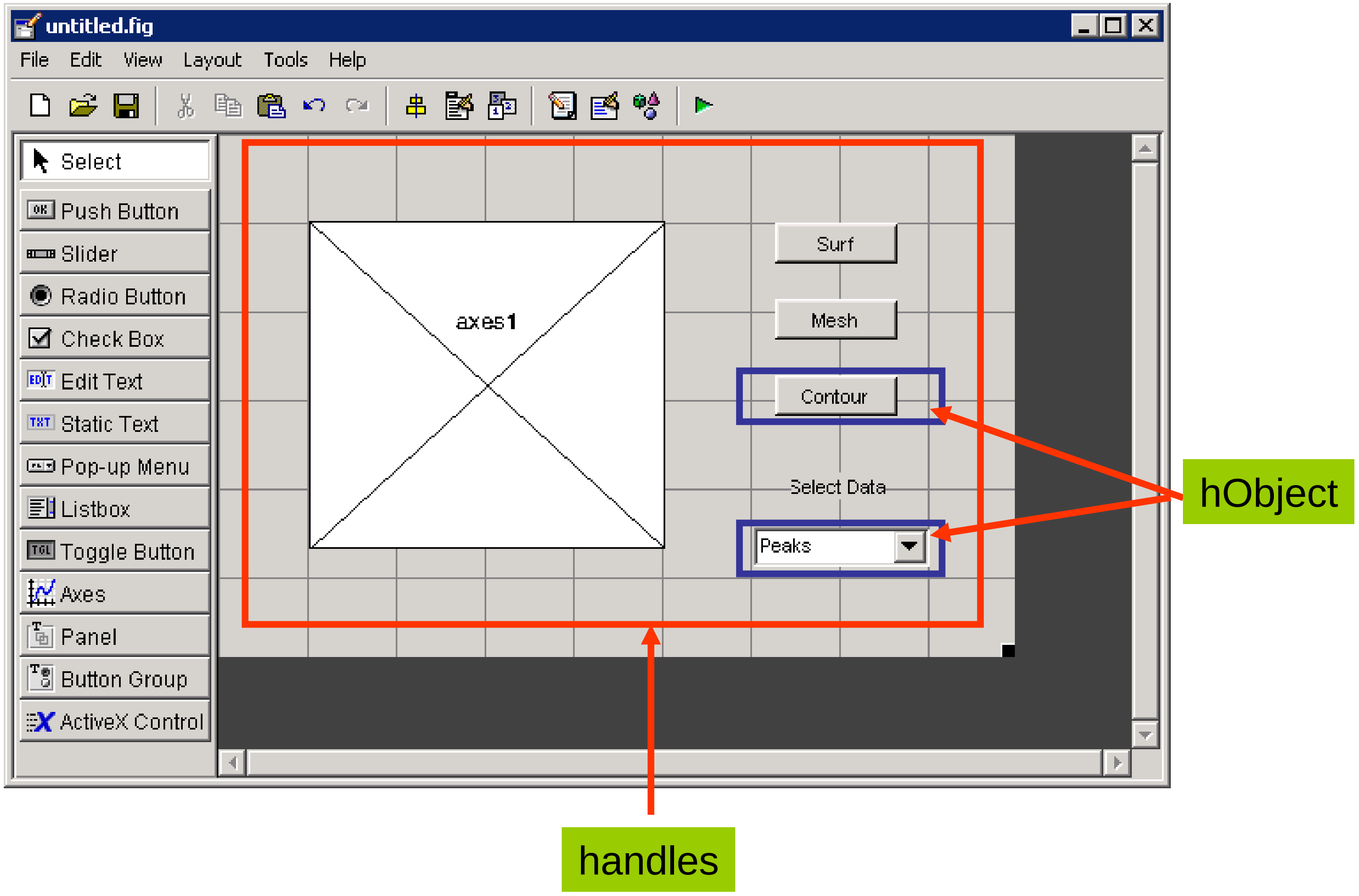


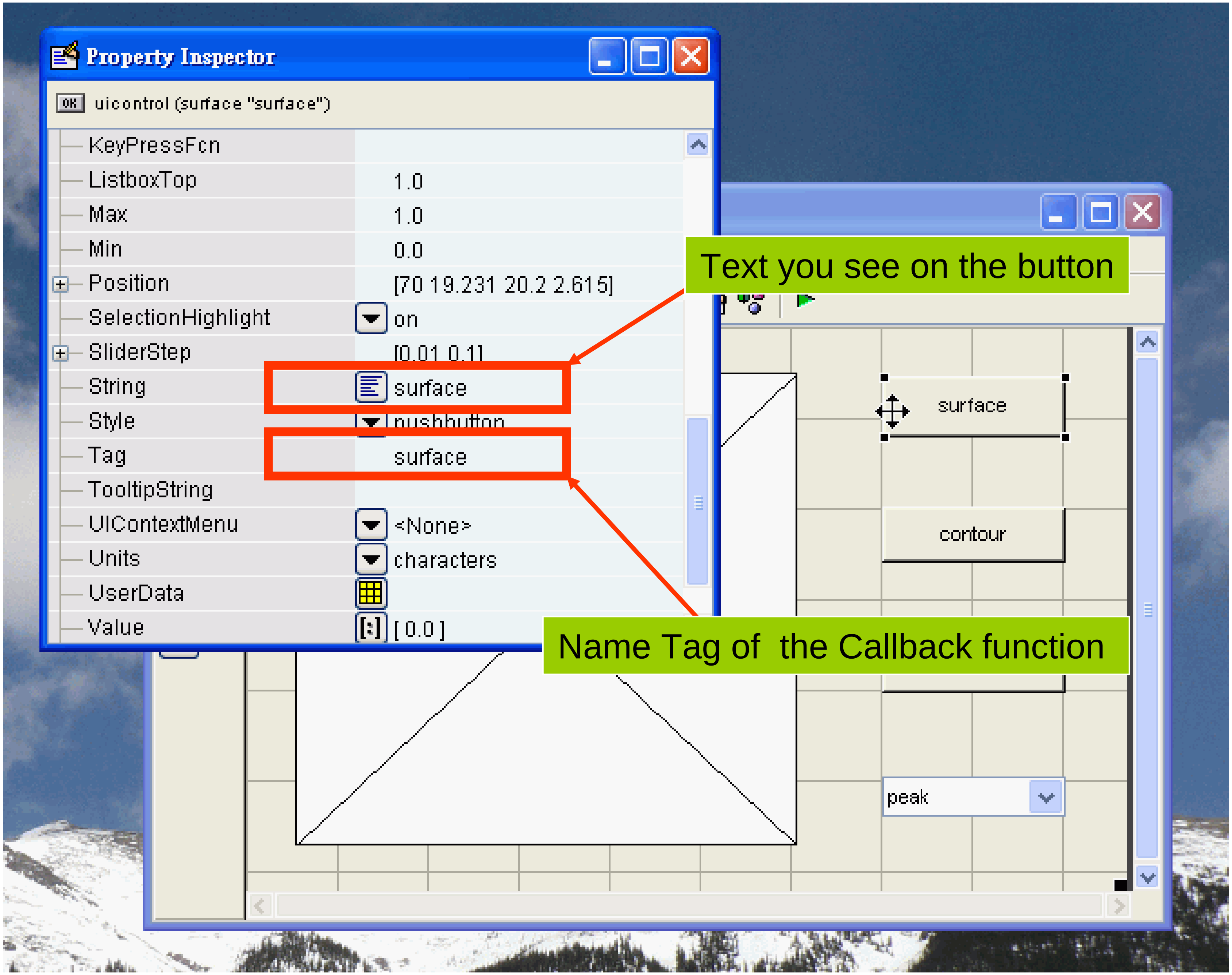
Graphic User Interface (GUI)

- Enter guide in the matlab command window
- Opening a New GUI in the Layout Editor
  - Setting the GUI Figure Size
  - Adding the Components
  - Aligning the Components
  - Adding Text to the Components
  - Save the GUI figure to “my_gui”

In GUI, many buttons, menu lists, …and the mouse etc, all together can do a lot of things. Each element mentioned above is called an “object”, each object needs to “respond” to the mouse or keyboard action from user. Information has to be sent through related objects for executing user’s command.

- A Gui is created:
  Opening_Fnc and Output_Fnc are created
- Information is carried by **handles**
- Each element (**hObject**) is created:
  - **Callback** is created
  - **Create_Fnc** is created if necessary
- When mouse click on the element, the associated callback is executed.
- use **guidata(hObject, handles)** to save the changes of information.

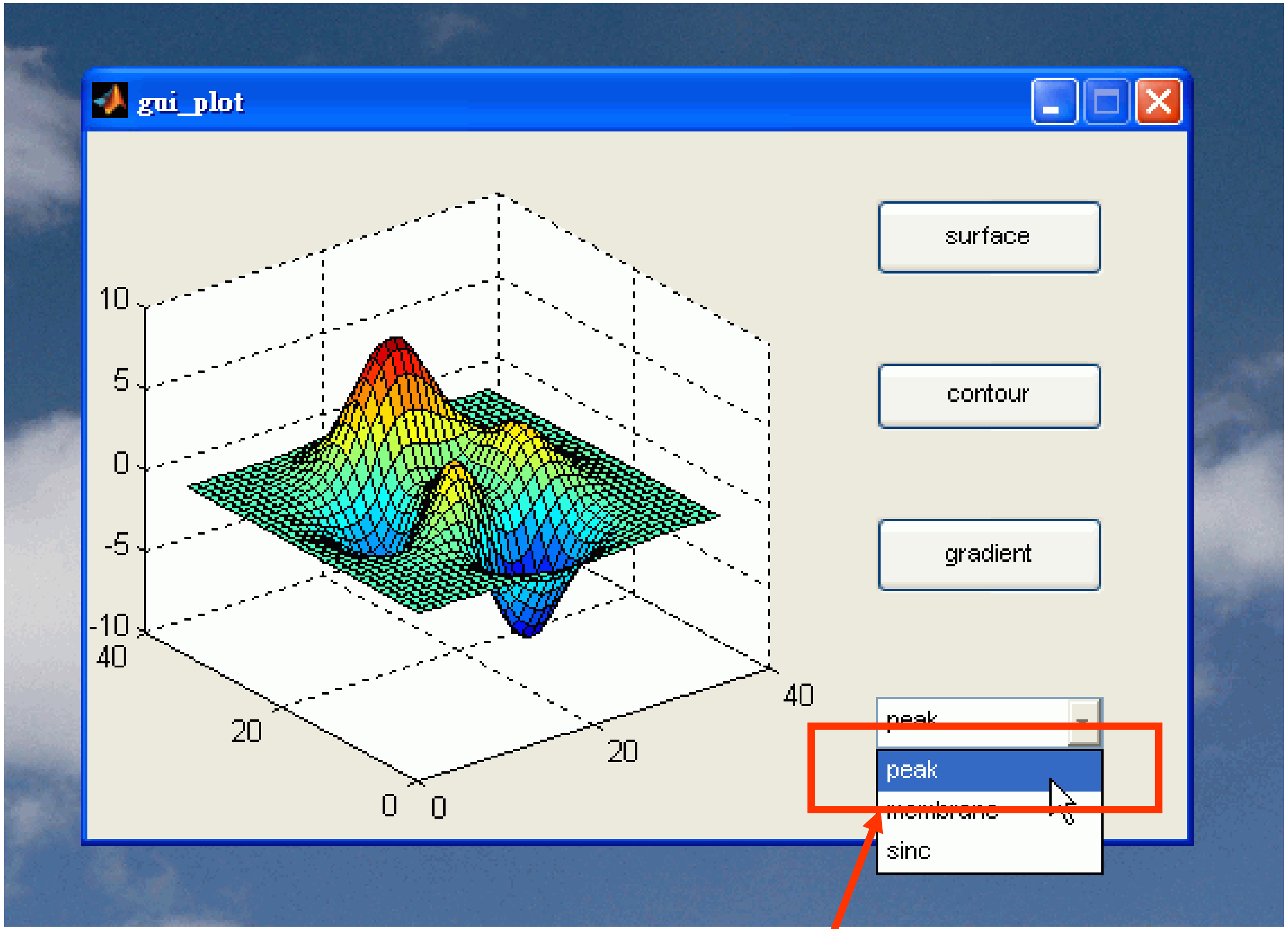

untitled.fig
File
Edit
View
Layout
Tools
Help
Select
Push Button
Slider
Radio Button
Check Box
Edit Text
Static Text
Pop-up Menu
Listbox
Toggle Button
Axes
Panel
Button Group
ActiveX Control
axes1
Surf
Mesh
Contour
Select Data
Peaks
hObject
handles



Property Inspector

uicontrol (surface "surface")

KeyPressFcn	
ListboxTop	1.0
Max	1.0
Min	0.0
Position	[70 19.231 20.2 2.615]
SelectionHighlight	on
SliderStep	[0.01 0.1]
String	surface
Style	pushbutton
Tag	surface
TooltipString	
UIContextMenu	<None>
Units	characters
UserData	
Value	[0.0]



Current_data

Add

```
[x,y] = meshgrid(-8:.5:8);
r = sqrt(x.^2+y.^2) + eps;
sinc = sin(r)./r;
handles.peaks=peaks(35);
handles.membrane=membrane;
handles.sinc = sinc;
% Set the current data value.
handles.current_data = handles.peaks;
surf(handles.current_data)
```

To my_gui_Openingfcn

Add

```
str = get(hObject, 'String');
val = get(hObject,'Value');
switch str{val};
case 'Peaks' % User selects peaks
handles.current_data = handles.peaks;
case 'Membrane' % User selects membrane
handles.current_data = handles.membrane;
case 'Sinc' % User selects sinc
handles.current_data = handles.sinc;
end
% Save the handles structure.
guidata(hObject,handles)
```

to Popupmenu_Callback

Add

```
mesh(handles.current_data);
```

to surface_Callback

Add

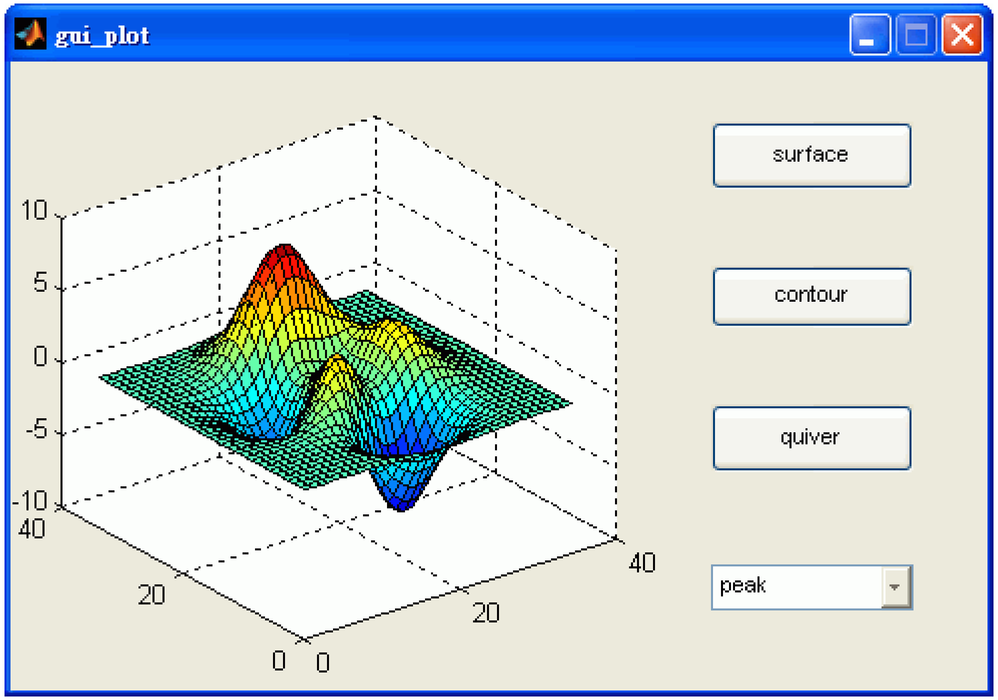
```
contour(handles.current_data);
```

to contour_Callback

Add

```
z=handles.current_data;
[zx,zy]=gradient(z);
quiver(zx,zy);
```

to quiver_Callback



Some References

- **Matlab basic:**

  http://umath.nuk.edu.tw/~scnet/CourseModule/KernelTool01/Math_software/download/WangWeichung_MatlabIntro_6up.pdf

  http://cedesign.me.ncu.edu.tw/swengap/class/notes/mainframe.htm

  http://libai.math.ncu.edu.tw/bcc16/B/matlab/index.shtml

- **Audio process:**

  http://neural.cs.nthu.edu.tw/jang/books/audioSignalProcessing/

- **Image process:**

  http://staffweb.ncnu.edu.tw/jcliu/course/dip2006.html

- **Matlab Guide:**

  http://neural.cs.nthu.edu.tw/jang/books/matlabProgramming4beginner/cdrom/matlabProgramming4beginner/slide/08-GUIDE.ppt#1