MATLAB Exercises

#01 We can generate a set of data with a random error about some mean value with the command

```
>> num=100;
>> spr=0.1;
>> data=5.0+spr*sign(1.0-2.0*rand(1,num)).*rand(1,num)
```

What is the last line doing?

(a) Write a program which computes the mean and standard deviation of the data. Use FOR LOOPs.
(b) Eliminate the FOR LOOPs using the sum function or the mean function and the std function.

#02 The classic quadratic formula says that the two roots of the quadratic equation

$$ax^2 + bx + c = 0$$

are

$$x_1, x_2 = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Use this formula in MATLAB to compute both roots for

        a = 1 , b = -100000000 , c = 1

Compare your results with

```
>> roots([a,b,c])
```

What happens if you try to compute the roots by hand or with a calculator? You should find that the classic formula is good for computing one root, but not the other. So use it to compute one root accurately and then use the fact that

$$x_1 x_2 = \frac{c}{a}$$

to compute the other root.

#03 Use the MATLAB program in the notes to implement the 3-point Lagrange Interpolation procedure and solve the following exercise.

Suppose the number of DUI arrests as a function of weekly alcohol consumption in Cincinnati, Ohio are as follows:

	Alcohol (Gallons)	DUI Arrests
1	103000	237
2	245000	845
3	450000	1356

Determine the number of DUI arrests at a consumption of 325000 gallons.

#04 The critical size of a nuclear reactor is determined by a criticality equation. Suppose a simple version of the criticality equation is given by

$$\tan(0.1x) = 9.2e^{-x}$$

The solution that is physically meaningful is the smallest positive root satisfying 3 ≤ x ≤ 4. Determine this smallest positive root using the three methods(codes) described in the text.

#05 Numerically valuate the integrals:

(a) $I = \int_0^2 \sqrt{1+e^x}\,dx$

(b) $I = \dfrac{1}{\sqrt{\pi}} \int_{-4}^4 e^{-x^2}\,dx$

(c) $I = \int_0^{10} e^{-x/10} \sin x\,dx$

Use Simpson's rule and the Monte Carlo method

#06 A good way to get some estimate of the accuracy when using the Monte Carlo method is to do the integral a large number of times and then plot the values of the integrand and a line for the average value. The spread around the average is an estimate of the error. Add this feature to your program.

Do this plot for $N = 100,1000,10000,100000$. What is the accuracy of this method determined by?

#07 In optics, one learns that light "bends around" objects, i.e., exhibits diffraction effects. Perhaps the simplest case to study is the bending of light around a straight edge. In this case, we find that the intensity of the light varies as we move away from the edge according to

$$I = 0.5I_0 \left[\left(C(v) + 0.5 \right)^2 + \left(S(v) + 0.5 \right)^2 \right]$$

where I_0 is the intensity of the incident light, v is proportional to the distance moved and $C(v)$ and $S(v)$ are the Fresnel integrals

$$C(v) = \int_0^v \cos\left(\frac{\pi w^2}{2} \right) dw \quad , \quad S(v) = \int_0^v \sin\left(\frac{\pi w^2}{2} \right) dw$$

Numerically evaluate the Fresnel integrals, and thus evaluate I/I_0 as a function of v. Plot your results.

#08 Plot the trajectories in the complex plane of the eigenvalues of the matrix A with elements

$$a_{ij} = \frac{1}{i - j + t} \quad , \quad i,j = 1,2,3,4,5 \quad , \quad 0 < t < 1$$

#09 Run the program box and describe the time evolution of n_L (the number of particles on the left side of the box). Choose N = 8,16,64,400,800,3600. Estimate the time for the system to reach equilibrium from your plots. How does the time depend on N? What criterion do your use for equilibrium? Does n_L change when the system is in equilibrium?

For sufficiently large N, does the time dependence of n_L appear to be deterministic?

#10 Modify the program box so that averages are taken after equilibrium has been reached.

What is the maximum deviation of $n_L(t)$ from N/2 for N=64,400,800,3600? Run for a time long enough to yield meaningful results.

How do you know if they are? How do your results for the maximum deviation depend on N?

Compare the variance of n_L for the same values of N as above. How do the relative fluctuations, $\sigma/\langle n_L \rangle$, depend on N?

#11 A, B, C are three circles of unit radius with centers in the xy-plane at (1,2), (2.5,1.5), and (2,3) respectively. Devise a hit or miss Monte Carlo calculation to determine the size of the area that lies outside C but inside A and B, as well as inside the square centered on (2,2.5) that has sides of length 2 parallel to the coordinate axes.

#12 3-Dimensional Graphics Investigations

We have discussed some aspects of 3-dimensional graphics in earlier parts of the notes. Now we want to investigate more details such as function arguments and options.

The end result of your investigations and experimentation in this exercise should be a series of unique plots and images with labels, colors, etc .....

Line Plots

The command of interest here is **plot3.** An example is

```
>> t=linspace(0,10*pi,500);
>> plot3(sin(t),cos(t),t);
>> xlabel('sin(t)'),ylabel('cos(t)'),zlabel('t')
>> text(0,0,0,'Origin')
>> grid on
>> title('Helix')
>> box on
```

Investigate these items:
    File > Save As ...
    Edit > Figure Properties ...
    Edit > Axis Properties ...
    View > Plot Edit Toolbar ...
    Insert > .....
    Tools > Rotate3d
    Tools > Zoom ....

etc....

You can easily add more plots to the same figure using **hold on.**

Try it.

Remember to use **hold off** when finished.

Scalar Functions of Two Variables

Often we want to visualize a scalar function of two variables,
say, z = f(x,y), where each pair of values (x,y) generates a
value for z. A plot of z as a function of x and y is a surface
in three dimensions. MATLAB can plot this surface if the values
of z are stored in a matrix. x and y are independent variables,
z is a matrix of the dependent variable and the association of x
and y with z is

$$z(i,:) = f(x,y(i)) \quad \text{and} \quad z(:,j) = f(x(j),y)$$

that is, the ith row of z is associated with the ith element of y
and the jth column of z is associated with the jth element of x.

When z = f(x,y) can be expressed simply, it is convenient to use
array operations to compute all the values of z in a single
statement. To do so requires that we create matrices of all the
x- and y-values in the proper orientation. MATLAB provides the
function **meshgrid** to perform this step:

```
>> x = -3:3;  % choose x-axis values
>> y = 1:5;   % y-axis values
>> [X,Y] = meshgrid(x,y)
```

```
X = -3     -2     -1      0      1      2      3
    -3     -2     -1      0      1      2      3
    -3     -2     -1      0      1      2      3
    -3     -2     -1      0      1      2      3
    -3     -2     -1      0      1      2      3


Y =  1      1      1      1      1      1      1
     2      2      2      2      2      2      2
     3      3      3      3      3      3      3
     4      4      4      4      4      4      4
     5      5      5      5      5      5      5
```

As you can see meshgrid duplicated x for each of the five rows
in Y. Similarly, it duplicated y as a column for each of the
seven columns in X.

Given X and Y, if x = f(x,y) = $(x+y)^2$, then the matrix of data values defining the 3-D surface is given simply by

```
>> Z = (X+Y).^2
```

```
Z =   4     1     0     1     4     9    16
      1     0     1     4     9    16    25
      0     1     4     9    16    25    36
      1     4     9    16    25    36    49
      4     9    16    25    36    49    64
```

When a function cannot be expressed simply (a formula) you must use For Loops or While Loops to compute the elements of Z.

Mesh Plots

```
>> mesh(Z)
>> colormap(gray)
```

Look at the x and y axes. The labels are not correct. If you do not specify x and y values in the mesh command , then it just used index labels.

```
>> mesh(X,Y,Z)
```

puts in the correct labels.

Put in labels and a title.

Try other colormaps. A list of colormaps is:

autumn ---- varies smoothly from red, through orange, to yellow.
bone ------ is a grayscale colormap with a higher value for the
            blue component. This colormap is useful for adding
            an "electronic" look to grayscale images.
colorcube - contains as many regularly spaced colors in RGB
            colorspace as possible, while attempting to provide
            more steps of gray, pure red, pure green, and pure
            blue.
cool ------ consists of colors that are shades of cyan and
            magenta. It varies smoothly from cyan to magenta.
copper ---- varies smoothly from black to bright copper.
flag ------ consists of the colors red, white, blue, and black.
            This colormap completely changes color with each
            index increment.
gray ------ returns a linear grayscale colormap.

```
hot ------- varies smoothly from black through shades of red,
            orange, and yellow, to white.
hsv ------- varies the hue component of the hue-saturation-value
            color model. The colors begin with red, pass through
            yellow, green, cyan, blue, magenta, and return to
            red. The colormap is particularly appropriate for
            displaying periodic functions.
hsv(m) ---- is the same as hsv2rgb([h ones(m,2)]) where h is the
            linear ramp, h = (0:m-1)'/m.
jet ------- ranges from blue to red, and passes through the
            colors cyan, yellow, and orange. It is a variation
            of the hsv colormap. The jet colormap is associated
            with an astrophysical fluid jet simulation from the
            National Center for Supercomputer Applications. See
            the "Examples" section.
lines ----- produces a colormap of colors specified by the axes
            ColorOrder property and a shade of gray.
pink ------ contains pastel shades of pink. The pink colormap
            provides sepia tone colorization of grayscale
            photographs.
prism ----- repeats the six colors red, orange, yellow, green,
            blue, and violet.
spring ---- consists of colors that are shades of magenta and
            yellow.
summer ---- consists of colors that are shades of green and
            yellow.
white ----- is an all white monochrome colormap.
winter ---- consists of colors that are shades of blue and
            green.
```

You can also create your own colormaps.

```
function map = waves(m)
% save in file called waves.m
m=128;
R = zeros(m,3);
j=1:128;
r=0.5*(1-sin(2*pi*j/128));
b=0.5*(1+sin(2*pi*j/128));
g=0.2*(2+sin(2*pi*j/8));
R(:,1)=r';
R(:,2)=g';
R(:,3)=b';
map=R;
```

Try this example:

```
>> load flujet
>> image(X)     % in gray map chosen earlier
>> colormap(jet) % change map
```

Try some other colormaps including waves.

In the mesh plot note that grid is "on" by default and the areas
between the mesh lines are opaque rather than transparent (this
is called "hidden lines")

```
>> hidden off
>> hidden on
```

There are lots of interest graphics objects in MATLAB. Try

```
>> [X,Y,Z]=sphere(12);
>> mesh(X,Y,Z)
>> colormap(hsv)
>> hidden off
>> hidden on
```

Surface Plots

In this case the surface mesh is "shaded" in some manner. The
possibilities are

        faceted (default), flat, interpolated

Try

```
>> [X,Y,Z]=peaks(30);
>> surf(X,Y,Z)
>> colormap(hsv)
>> shading flat
>> shading interp
>> colormap waves
>> colormap hot
```

Experiment on your own.

Adding contours....

```
>> surfc(X,Y,Z)
```

Or adding lights.....

```
>> surfl(X,Y,Z)
>> shading interp
>> colormap(pink) % looks better with shades of one color
```

Try

Tools > Rotate3d   for FUN!

Then restore the default picture by

Tools > Reset View

You can change the view (viewpoint = where you are looking from)
by view(a,b) where

        a = azimuth angle with respect to the x=0 plane
        b = angle of elevation with respect to z=0 plane

The default is

    view(-37.5,30.0)

Try other viewpoints.

Other Useful Plots

Use the "peaks" data set and experiment with the commands

        contour, contour3, pcolor, contourf

For example

```
>> [X,Y,Z]=peaks;
>> contour(X,Y,Z,20);
>> colormap(hsv)
>> contour3(X,Y,Z,20);
>> pcolor(X,Y,Z);
>> shading interp
>> axis square
>> contourf(X,Y,Z,12);
>> C=contour(X,Y,Z,12);
>> clabel(C);
```

and so on....

Gradients and vector Fields

Experiment with these examples:

```
>> [X,Y,Z]=peaks(16);
>> [DX,DY]=gradient(Z,0.5,0.5);
>> contour(X,Y,Z,10);
>> colormap(hot)
>> hold on
>> quiver(X,Y,DX,DY);
>> hold off

[X,Y,Z]=peaks(20);
>> [Nx,Ny,Nz]=surfnorm(X,Y,Z);
>> surf(X,Y,Z);
>> hold on
>> quiver3(X,Y,Z,Nx,Ny,Nz);
>> hold off
```

#13 The carbon dioxide data for Mauna Loa in the manual is in the file global1.dat.

(a) Using linear regression find the rate of increase of CO_2 concentration in ppm per year. Besides the linear trend, the data shows an annual cycle of CO_2 concentration.

(b) Write a program that removes the annual cycle from the data (called filtering), that is, using spectral analysis, remove the cycle from the data.

(c) Plot the original and filtered data. What is the slope of the filtered data?

#14 Sunspots and El Ninos (example due to Clive Moler)

Read through and understand the sunspot example below and due a similar analyis on El Nino data (see exercise at end of sunspot example).

For centuries, people have noted that the face of the sun is not constant or uniform in appearance, but that darker regions appear at random locations on a cyclical basis. This activity is correlated with weather and and other economically significant terrestrial phenomena. In 1848, Rudolf Wolfer proposed a rule that combine the number and size of these sunspots into a single

index. Using archival records, astronomers have applied Wolfer's rule to determine sunspot activity back to the year 1700. Today the sunspot index is measured by many astronomers, and the worldwide distribution of data is coordinated by the Solar Influences Data Center at the Royal Observatory of Belgium.

The text file sunspot.dat has two columns of numbers. The first column is the years from 1700 to 1987 and the second column is the average Wolfer sunspot number for each year.

```
>> load sunspot.dat
>> t=sunspot(:,1);
>> wolfer=sunspot(:,2);
>> n = length(wolfer);
```

It turns out that there is a slight upward trend to the data. A least squares fit gives the trend line.

```
>> c=polyfit(t,wolfer,1);
>> trend=polyval(c,t);
>> plot(t,[wolfer,trend],'-',t,wolfer,'k.');
>> xlabel('year');
>> ylabel('Wolfer index');
>> title('Sunspot index with linear trend');
```

Sunspot index with linear trend
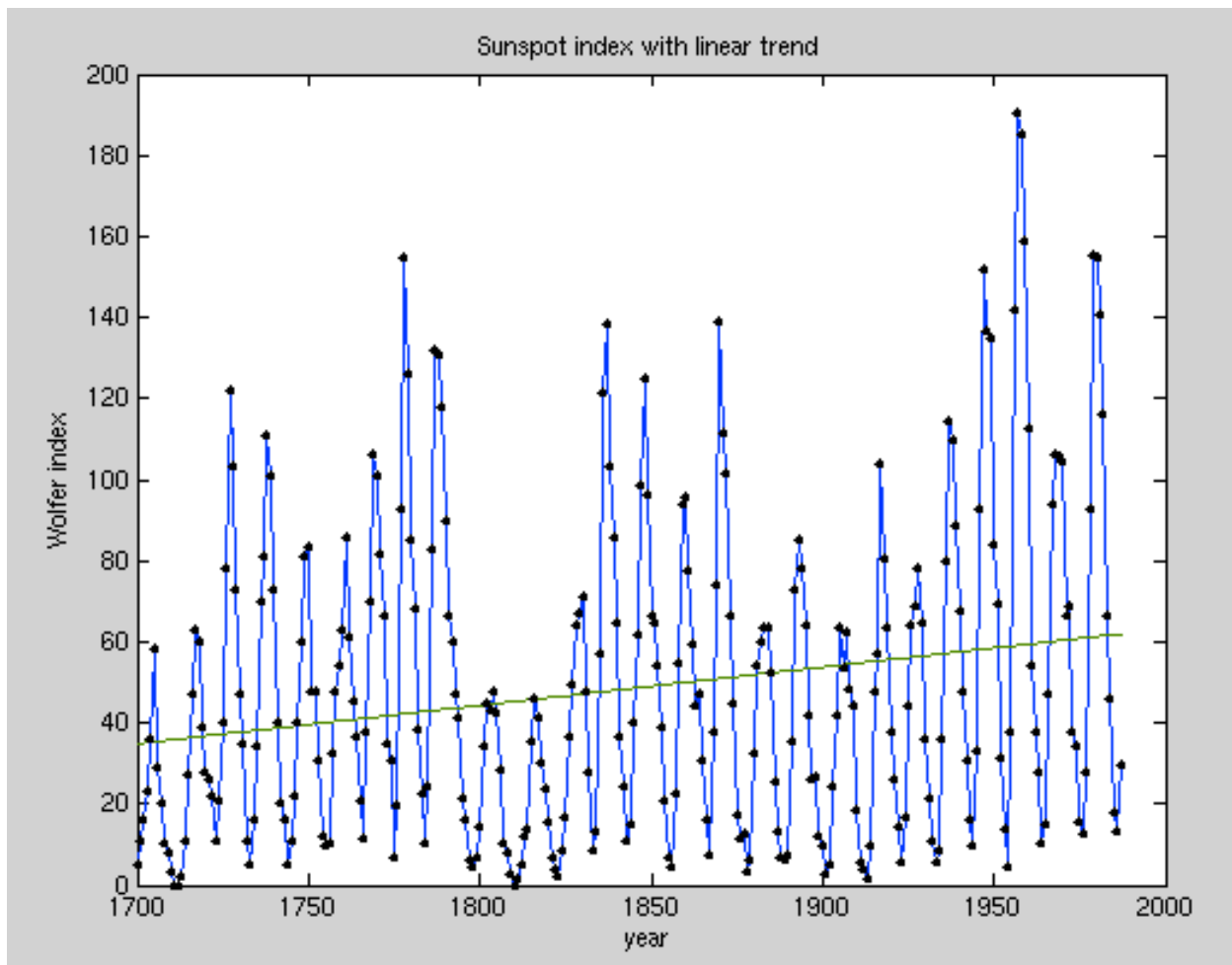
You can definitely see the cyclic nature of the sunspot phenomenon. The peaks and valleys are a little more than 10 years apart.

Now we subtract off the linear trend and take the FFT.

```
>> y=wolfer-trend;
>> Y=fft(y);
```

The vector $|Y|^2$ is the **power** of the signal. A plot of power versus frequency is called a periodogram. We will plot $|Y|$, rather than $|Y|^2$, because the scaling is not so exaggerated. The sample rate for these data is one observation per year, so the frequency f ahs units of cycles per year.

```
>> Fs = 1; % Sample rate
```

```
>> f = (0:n/2)*Fs/n;
>> pow = abs(Y(1:n/2+1));
>> pmax = 5000;
>> plot([f; f],[0*pow'; pow'],'c-',f,pow','r.', ...
      'linewidth',2,'markersize',16)
>> axis([0 .5 0 pmax]);
>> xlabel('cycles/year');
>> ylabel('power');
>> title('Periodogram');
```
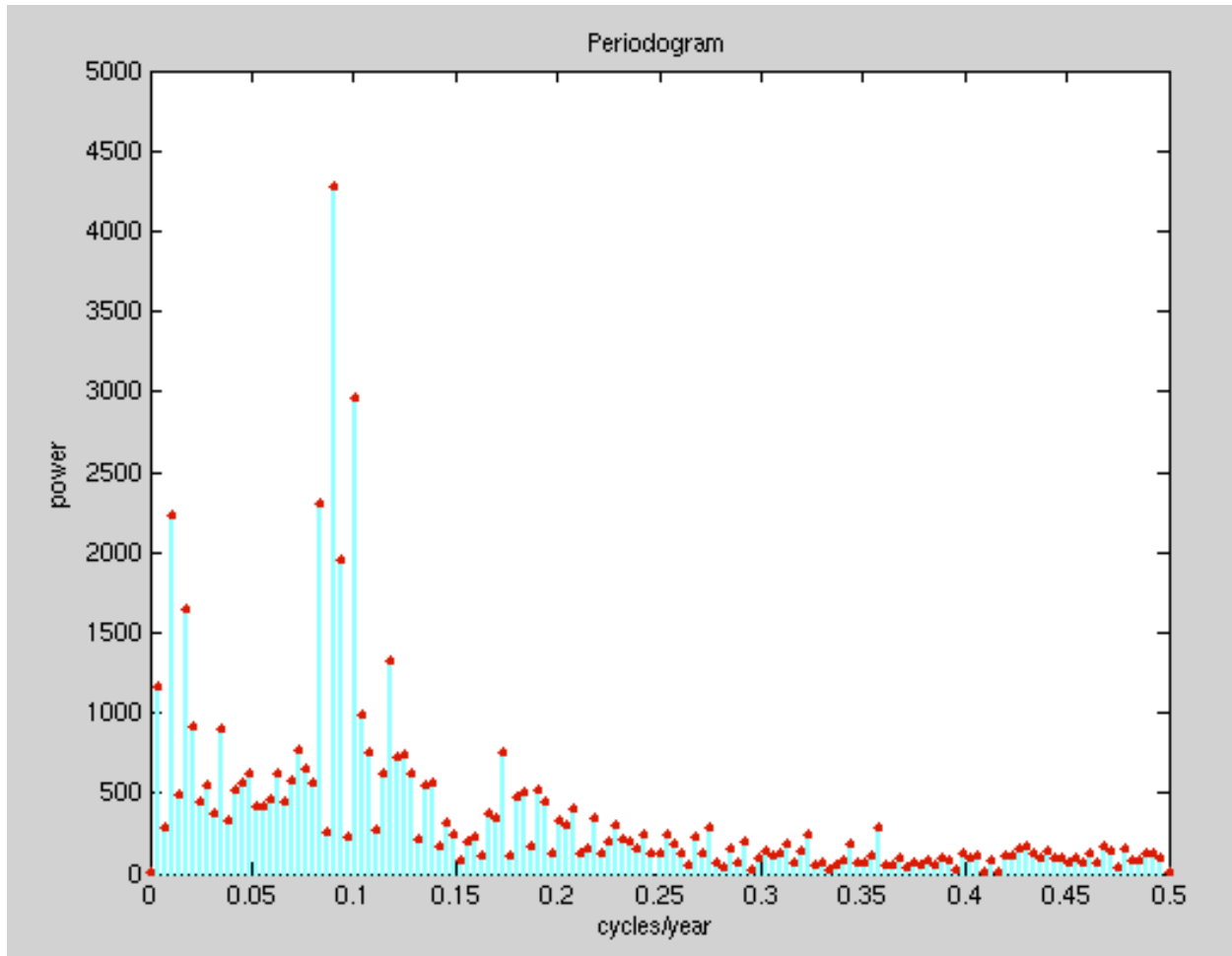

The maximum power occurs near frequency = 0.09 cycles/year. We would like to know the corresponding period in years/cycle. Let us zoom in on the plot and use the reciprocal of frequency to label the x-axis.
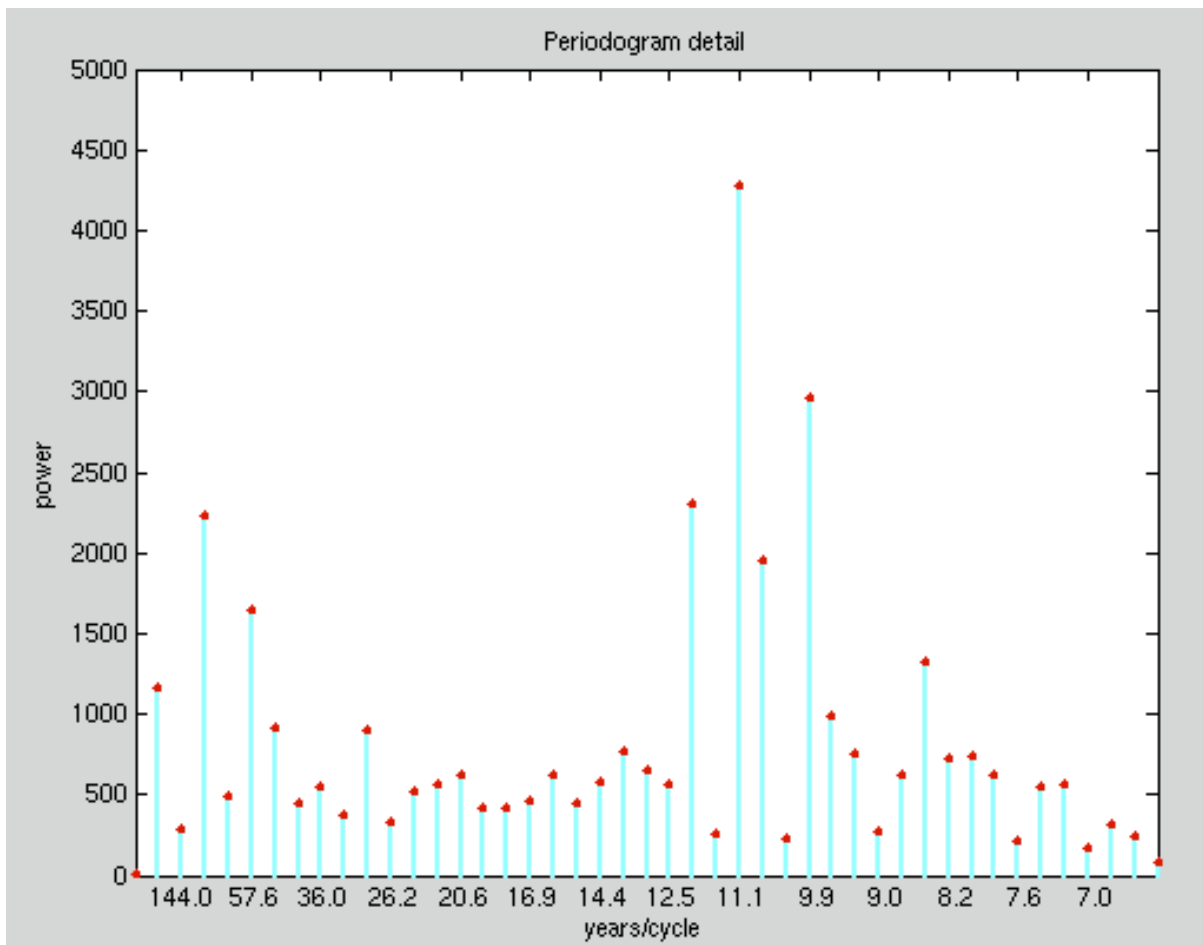
```
>> k=0:44;
>> f=k/n;
>> pow=pow(k+1);
>> plot([f; f],[0*pow'; pow'],'c-',f,pow','r.', ...
```

```
        'linewidth',2,'markersize',16);
>> axis([0 max(f) 0 pmax]);
>> k = 2:3:41;
>> f=k/n;
>> period = 1./f;
>> periods = sprintf('%5.1f|',period);
>> set(gca,'xtick',f);
>> set(gca,'xticklabel',periods);
>> xlabel('years/cycle');
>> ylabel('power');
>> title('Periodogram detail');
```


As expected, there is a very prominent cycle with a length of
about 11.1 years. This shows that over the last 300 years, the
period of the sunspot cycle has been slightly over 11 years.

El Nino

The climatological phenomenon el Nino results from changes in
atmospheric pressure in the southern Pacific ocean. The

"Southern Oscillation Index" is the difference in atmospheric pressure between easter Island and Darwin, Australia, measured at sea level at the same moment. The file elnino.dat contains the values of this index measured on a monthly basis over the 14-year period 1962 through 1975.

This exercise is to carry out an analysis similar to the sunspot example on the el Nino data. The unit of time is one month instead of one year. You should find a prominent cycle with a period of 12 months and a second, less prominent, cycle with a longer period. The second cycle shows up in about three of the Fourier coefficients, so it is hard to measure its length, but see if you can make an estimate.

```
>> load elnino.dat
>> plot(elnino)
>> n=length(elnino);
>> t=0:167;
>> elnino = elnino';
>> c=polyfit(t,elnino,1);
>> trend=polyval(c,t);
>> plot(t,[elnino;trend],'-',t,elnino,'k.');
>> y=elnino-trend;
>> Y=fft(y);
>> Fs=1/12;
>> f = (0:n/2)*Fs/n;
>> pow = abs(Y(1:n/2+1));
>>  plot([f; f],[0*pow; pow],'c-',f,pow,'r.', ...
       'linewidth',2,'markersize',16)
```

#15 **Logistic Population Growth** – The population curve P(t) for the USA is assumed to obey the differential equation for the logistic curve

$$\frac{dP}{dt} = aP - bP^2$$

Let t denote the year after 1900 and let the step size h = 10 years. Use the values a = 0.02 and b = 0.00004 as a model for the USA population. Use Euler-Cromer to solve this equation mumerically and compare your approximate answer with the actual population values below:

Year	Population(millions)
1900	76.1
1910	92.4
1920	106.5
1930	123.1
1940	132.6
1950	152.3
1960	180.7
1970	204.9
1980	226.5

Make a plot with your solution as a curve and the table data as superimposed squares. Make it a spectacular plot with all the frills.

#16 **Epidemic Model** – A mathematical model for epidemics is described as follows: Assume that there is a community of L members which contains P infected individuals and Q uninfected individuals. Let y(t) denote the number of infected individuals at time t . For a mild illness such as the common cold, everyone continues to be active, and the epidemic spreads from those who are infected to those who are uninfected. Since there are PQ possible contacts between these two groups, the rate of change of y(t) is proportional to PQ. Hence the problem can be stated as

$$\frac{dy}{dt} = ky(L - y)$$

Explain why.

Use L = 25,000, k = 0.00003, and h = 0.2 with initial condition y(0) = 250, and use the Runge-Kutta method to compute the approximate solution over the interval [0,12].

#17 A typical predator-prey simulation might use the coefficients

A = 2, B = 0.02, C = 0.0002, D=0.8

in the model equations in the manual.

Use the Runge-Kutta method to solve the differential equations
over the interval [0,5] using M=50 steps and h = 0.1 if
         (a) x(0) = 3000 rabbits and y(0) = 120 foxes
         (b) x(0) = 5000 rabbits and y(0) = 100 foxes

Do the curves generated by the model equations make sense?

#18 Another way of presenting the predator-prey equations is due
to Lotka and Volterra. The details of this model go as follows.
Here we are considering an ecosystem consisting rabbits that
have an infinite supply of food and foxes that prey on the
rabbits for their food. The system can be modeled by the
equations

$$\frac{dr}{dt} = 2r - \alpha rf \quad , \quad r(0) = r_0$$

$$\frac{df}{dt} = -f + \alpha rf \quad , \quad f(0) = f_0$$

where t is time, r(t) is the number of rabbits, f(t) is the
number of foxes and α is a positive constant. If
α = 0, the two populations do not interact, the rabbits do what
rabbits do best and the foxes die off from starvation. If α > 0,
the foxes encounter the rabbits with a probability that is
proportional to the product of their numbers. Such an encounter
results in a decrease in the number of rabbits and (for less
obvious reasons) and an increase in the number of foxes.

The solutions of this nonlinear system cannot be expressed in
nterms of other known functions; the equations must be solved
numerically. It turns out that the solutions are always
periodic, with a period that depends on the initial conditions.
In other words, for any r(0) and f(0) there is a value t = t_p
when both populations return to their priginal values.
Consequently, for all t,

$$r(t + t_p) = r(t) \quad , \quad f(t + t_p) = f(t)$$

(a) Compute the solution with $r_0 = 300, f_0 = 150, \alpha = 0.01$. You
    should find t_p is close to 5. Make two plots, one
    for r and one for f as functions of t and one phase
    plane plot of r versus f.
(b) Compute the solution with $r_0 = 15, f_0 = 22, \alpha = 0.01$. You
    should find t_p is close to 6.62. Make two plots, one

for r and one for f as functions of t and one phase
plane plot of r versus f.

(c) Compute and plot the solution with $r_0 = 15, f_0 = 198, \alpha = 0.01$.
Determine the period t_p in some way.

(d) The point $(r_0, f_0) = (1/\alpha, 2/\alpha)$ is a stable equilibrium
point. If the populations have these initial
values, then they do not change. If the initial
populations are close to these values, they do not
change very much. Let u(t) = r(t) - 1/α and
v(t) = f(t) - 2/α. The functions u(t) and v(t)
satisfy another nonlinear system of equations, but
if the uv terms are ignored, the system becomes
linear. What is this new linear system? What is the
period of its periodic solutions?

#19 Many modification of the standard predator-prey model have
been proposed to more accurately reflect what happens in nature.
For example, the number of rabbits can be prevented from growing
indefinitely by changing the equations as follows:

$$\frac{dr}{dt} = 2\left(1 - \frac{r}{R}\right)r - \alpha rf \quad , \quad r(0) = r_0$$

$$\frac{df}{dt} = -f + \alpha rf \quad , \quad f(0) = f_0$$

where t is time, r(t) is the number of rabbits, f(t) is the
number of foxes and α is a positive constant. Because α is
positive, dr/dt is negative whenever r≥R. Consequently, the
number of rabbits can never exceed R, For α = 0.01, compare the
behavior of the original model with the behavior of this
modified model with R = 400. In making this comparison solve the
equations with $r_0 = 300$ and $f_0 = 150$ over 50 units of time.

#20 Finally, let's do a nonlinear oscillator; one that exhibits
chaotic dynamics. There is no analytical solution - numerical
solution is the only way to go here. The equation of motion is:

$$\ddot{\theta} = a\cos(\omega_0 t) - \sin(\theta) - q\dot{\theta}$$

$$\theta(0) = \dot{\theta}(0) = 0$$

Some explanation of the terms on the right is necessary:

(1) The second term is what we'd write down for any
pendulum - it's the general case rather than the
small-angle limit

(2) The third is a damping term, with "damping strength" q.
(3) The first is a driving term, with driving amplitude a and frequency ω_0.

Some suggested parameter sets are below. Take h = 0.05 throughout.

q = 1/2 , ω_0 = 2/3

a=0.9 ; periodic
a=1.07 ; period doubling
a=1.15 ; chaos
a=1.35 ; periodic
a=1.45 ; period doubling
a=1.47 ; period doubling
a=1.50 ; chaos

Use Runge-Kutta to calculate the motion of the pendulum for several cycles.
Create a phase-space plot, where you plot $\dot\theta$ as a function of θ.
Restrict the θ motion to $-\pi \leq \theta \leq \pi$.

#21 **Poincare Plots**

Modify the last exercise so that we only plot one point per period of the driving force. The resulting plot is called a **Poincare** plot. It is able to analyze periodic motions and chaotic motions directly and quickly allow us to identify each type and determine period doubling points and strange attractors.

1D_Schrodinger Equation

The 1-dimensional Schrodinger equation takes the form

$$-\frac{\hbar^2}{2m}\frac{d^2\psi(x)}{dx^2}+V(x)\psi(x)=E\psi(x)$$

with boundary conditions

$$\psi(0),\frac{d\psi(0)}{dx}\quad \text{finite everywhere}$$

$$\psi(\infty)=0$$

$$\int_{-\infty}^{\infty}|\psi(\text{x})|^2\,dx \quad \text{finite}$$

We will consider the finite square-well potential functions in these notes, namely,

$$V(x) = \begin{cases} 0 & |x| > a \\ -V_0 & |x| < a \end{cases}$$

Since the bound state solutions will have E<0, we substitute E=-|E| and choose

$$\hbar = \sqrt{2} \quad , \quad m = a = 1 \quad , \quad |E| = \alpha V_0 \quad , \quad \frac{2mV_0 a^2}{\hbar^2} = V_0 = 15$$

to get the equations

$$\psi'' = 15\alpha\psi \qquad |x| > a$$
$$\psi'' = -15(1-\alpha)\psi \qquad |x| < a$$
$$\alpha < 1$$

Now both potentials satisfy the relation $V(x) = V(-x)$. This means that the solutions are of only two types:

Even parity:

$$\psi(x) = \psi(-x)$$
$$\psi(0) = \text{finite} = 1 \text{ (for simplicity)}$$
$$\frac{d\psi(0)}{dx} = 0$$

Odd parity:

$$\psi(x) = -\psi(-x)$$
$$\psi(0) = 0$$
$$\frac{d\psi(0)}{dx} = \text{finite} = 1 \text{ (for simplicity)}$$

The solution method (called **shooting method**) goes as follows:

[1] Choose ψ and ψ' at x = 0   (choose either odd/even solution)

[2] Pick an energy value

[3] With the chosen boundary conditions and energy value, break the equation up into two first-order equations and use the Runge-Kutta method to solve for the values of $\psi(x)$ for x > 0.
We let $\psi' = y$ and then $\psi'' = y'$ and we get as our equations

$$y' = (0.5x^2 - E)\psi = g(x,\psi,y,E)$$
$$\psi' = y = f(x,\psi,y,E)$$

The Runge-Kutta equations are then

$$x_{n+1} = x_n + h$$

$$\psi_{n+1} = \psi_n + \frac{h}{6}(f_1 + 2f_2 + 2f_3 + f_4)$$

$$y_{n+1} = y_n + \frac{h}{6}(g_1 + 2g_2 + 2g_3 + g_4)$$

where

$f_1 = f(x_n, \psi_n, y_n, E)$

$g_1 = g(x_n, \psi_n, y_n, E)$

$f_2 = f(x_n + h/2, \psi_n + hf_1/2, y_n + hg_1/2, E)$

$g_2 = g(x_n + h/2, \psi_n + hf_1/2, y_n + hg_1/2, E)$

$f_3 = f(x_n + h/2, \psi_n + hf_2/2, y_n + hg_2/2, E)$

$g_3 = g(x_n + h/2, \psi_n + hf_2/2, y_n + hg_2/2, E)$

$f_4 = f(x_n + h, \psi_n + hf_3, y_n + hg_3, E)$

$g_4 = g(x_n + h, \psi_n + hf_3, y_n + hg_3, E)$

[4] An allowed energy value (eigenvalue of the Hamiltonian)
occurs when the solution approaches zero for large values
of x.

[5] Our solution method homes in on that energy value for which
the value of x where the solution begins to diverge is a
maximum (within certain constraints).

SQErk.m (with **sq.m**) is a sample MATLAB programs for solving
these equations is for the finite **square-well.**

#22 Make the well deep enough so that there are several energy
levels. Run this program and generate the even eigenvalues.

#23 Run this program and generate the odd eigenvalues.

#24 Modify the code so that we plot the wave function in real
time at the end of each energy step.

Just Relaxing and using pcolor, contour, mesh and surf

#25 (modify laplsolve.m or laplrelax.m codes) Use an iterative
method to compute an approximate solution to Laplace's equation
where the boundary values are:

$u(x,0) = 20$ and $u(x,4) = 180$ for $0 < x < 4$

$u(0,y) = 80$ and $u(4,y) = 0$ for $0 < y < 4$

Present the result using pcolor, contour, mesh and surf.

#26 (modify laplsolve.m or laplrelax.m codes) Use an iterative
method to compute an approximate solution to Laplace's equation
where the boundary values are:

$$u(x,0) = x^4 \text{ and } u(x,1.5) = x^4\text{-}13.5x^2+5.0625 \quad \text{for} \quad 0<x<1.5$$

$$u(0,y) = y^4 \text{ and } u(4,y) = 5.0625\text{-}13.5y^2+y^4 \quad \text{for} \quad 0<y<1.5$$

Compare with the exact solution: $u(x,y) = x^4 - 6x^2y^2 + y^4$.

Present the result using pcolor, contour, mesh and surf.

#27 (modify laplsolve.m) Use an iterative method to compute an approximate solution to Poisson's equation where the boundary values are:

$$u(x,y) = x^4 - 6x^2y^2 + y^4 u(x,0) = x^3 \text{ and } u(x,1) = x^3 \quad \text{for} \quad 0<x<1$$

$$u(0,y) = 0 \text{ and } u(1,y) = 1 \quad \text{for} \quad 0<y<1$$

$$g(x,y)=y$$

Present the result using pcolor, contour, mesh and surf.

#28 (modify laplsolve.m) Use an iterative method to compute an approximate solution to Helmholtz's equation where the boundary values are:

$$u(x,y) = \cos(2x) + \sin(2y) \text{ for } 0<x<1 \text{ and } 0<y<1$$

$$g(x,y)=y \quad and \quad f(x,y)=4$$

Present the result using pcolor, contour, mesh and surf.

#29 **Filip Data Set**

One of the Statistical Reference Datasets (SRD) from NIST is the "Filip" data set. The data consists of several dozen observations of a variable y at different values of x. The task is to model y by a polynomial of degree 10 in x. The data set is controversial. NIST has given a set of "certified values" listed below. Not everyone agrees with these values! Some fitting packages will even say that the problem is too badly conditioned to solve! What happens when we apply a variety of MATLAB methods to this problem. The data is in the file **filip.dat.** The file description looks like:

```
NIST/ITL StRD
Dataset Name:  Filip (Filip.dat)

File Format:   ASCII
               Certified Values  (lines 31 to 55)
               Data               (lines 61 to 142)
Procedure:     Linear Least Squares Regression
Reference:     Filippelli, A., NIST.
Data:          1 Response Variable (y)
               1 Predictor Variable (x)
               82 Observations
               Higher Level of Difficulty
```

```
                 Observed Data
Model:           Polynomial Class
                 11 Parameters (B0,B1,...,B10)

                 y = B0 + B1*x + B2*(x**2) + ... + B9*(x**9) + B10*(x**10) + e

                 Certified Regression Statistics


                                          Standard Deviation
      Parameter         Estimate               of Estimate

         B0         -1467.48961422980      298.084530995537
         B1         -2772.17959193342      559.779865474950
         B2         -2316.37108160893      466.477572127796
         B3         -1127.97394098372      227.204274477751
         B4         -354.478233703349      71.6478660875927
         B5         -75.1242017393757      15.2897178747400
         B6         -10.8753180355343      2.23691159816033
         B7         -1.06221498588947      0.221624321934227
         B8         -0.670191154593408E-01 0.142363763154724E-01
         B9         -0.246781078275479E-02 0.535617408889821E-03
         B10        -0.402962525080404E-04 0.896632837373868E-05


      Residual
      Standard Deviation    0.334801051324544E-02


      R-Squared             0.996727416185620


              Certified Analysis of Variance Table


Source of Degrees of     Sums of               Mean
Variation  Freedom       Squares               Squares          F Statistic

Regression  10      0.242391619837339    0.242391619837339E-01 2162.43954511489
Residual    71      0.795851382172941E-03 0.112091743968020E-04
```

The actual data has one line for each data point (y,x). the x-values are not monotonically ordered but it will not be necessary to sort them. Let n be the number of data points and let p = 11 be the number of polynomial coefficients.

(a) Load the data into MATLAB, plot it with '.' as the line type and then invoke the **Basic Fitting** tool available under the **Tools** menu on the Figure window. Select the 10th-degree polynomial fit. You will be warned that the polynomial is badly conditioned, but ignore that for now.

How do the computed coefficients compare with the NIST values? How does the plotted fit compare with the graphic on the NIST web page?

http://www.itl.nist.gov/div898/strd/lls/data/Filip.shtml

The basic fitting tool also displays the norm of the residuals
||r||. Compare this with the NIST quantity "Residual Standard
Deviation" which is

$$\frac{\|r\|}{\sqrt{n-p}}$$

(b) Examine this data set more carefully by using six different
    methods to compute the polynomial fit. Think about the
    various warning messages you receive. You will have to
    figure out some of the commands needed.

    (1) Polyfit: Use polyfit(x,y,10)
    (2) Backslash: Use X\y where X is the n x p truncated
        Vandermonde matrix with elements
        $$X_{ij} = x_i^{p-j} \quad , \ i=1,2,...,n \quad , \ j=1,2,...,p$$
    (3) Pseudoinverse: Use pinv(X)*y
    (4) Normal equations: Use inv(X'*X)*X'*y
    (5) Centering: Let μ = mean(x), σ = std(x), t = (x-μ)/σ.
        Use polyfit(t,y,10).
    (6) The NIST "Certified Coefficients".

(c) What are the norms of the residuals for the fits computed by
    the six different methods?
(d) Which one of the six methods gives a poor fit? Maybe this is
    what all the complainers are using!
(e) Plot the five good fits. Use dots, '.', at the data points
    and curves obtained by evaluating the polynomials at a few
    hundred points over the range of x's. Only one of the plots
    is visually distinct from the other four? Which one?

MORAL: We must be careful even when using expensive and
        sophisticated packages.

#30 **Longley Data Set**

The Longley data set of labor statistics was one of the first
used to test the accuracy of least squares computations. It is
described below. The data is in the file longley.dat.

```
NIST/ITL StRD
Dataset Name:   Longley (Longley.dat)

File Format:    ASCII
                Certified Values  (lines 31 to 51)
                Data              (lines 61 to 76)
Procedure:      Linear Least Squares Regression
Reference:      Longley, J. W. (1967).
```

```
                    An Appraisal of Least Squares Programs for the
                    Electronic Computer from the Viewpoint of the User.
                    Journal of the American Statistical Association, 62, pp. 819-841.
Data:               1 Response Variable (y)
                    6 Predictor Variable (x)
                    16 Observations
                    Higher Level of Difficulty
                    Observed Data
Model:              Polynomial Class
                    7 Parameters (B0,B1,...,B7)

                    y = B0 + B1*x1 + B2*x2 + B3*x3 + B4*x4 + B5*x5 + B6*x6 + e

                    Certified Regression Statistics


                                              Standard Deviation
        Parameter           Estimate             of Estimate

           B0         -3482258.63459582        890420.383607373
           B1            15.0618722713733       84.9149257747669
           B2            -0.358191792925910E-01  0.334910077722432E-01
           B3            -2.02022980381683       0.488399681651699
           B4            -1.03322686717359       0.214274163161675
           B5            -0.511041056535807E-01  0.226073200069370
           B6          1829.15146461355        455.478499142212


        Residual
        Standard Deviation   304.854073561965

        R-Squared            0.995479004577296

                Certified Analysis of Variance Table

Source of  Degrees of     Sums of            Mean
Variation  Freedom        Squares            Squares            F Statistic

Regression    6      184172401.944494    30695400.3240823   330.285339234588
Residual      9         836424.055505915   92936.0061673238

Data:     y       x1      x2      x3      x4      x5      x6
```

The variables are:

```
    y   = Total Derived Employment
    x1  = GNP Implicit Price Deflater
    x2  = Gross National Product
    x3  = Unemployment
    x4  = Size of Armed Forces
    x5  = Noninstitutional Population Age 14 and Over
    x6  = Year
```

There are 16 observations of these 7 variables, gathered over the years 1947 to 1962.

Read the data using:

```
>> load longley.dat;
>> y = longley(:,1);
>> X = longley(:,2:7);   & data saved a a matrix X
```

The objective is to predict y by a linear combination of a
constant and six x's:

$$y \approx \beta_0 + \sum_{k=1}^{6} \beta_k x_k$$

(a) Use the MATLAB backslash operator to compute $\beta_0, \beta_1, \ldots, \beta_6$. This
    involves augmenting X with a column of all 1's,
    corresponding to the constant term. Why?

(b) Compare your $\beta_0, \beta_1, \ldots, \beta_6$ values with the "Certified" values.

(c) Use the **errorbar** command to plot y with error bars whose
    magnitude is the difference between y and the least squares
    fit.

(d) Use the **corrcoef** command to compute the correlation
    coefficients (what are they?) for X without the column of
    1's. Which variables are highly correlated?

(e) Normalize the vector y so that its mean is zero and its
    standard deviation is one using

```
>> y = y - mean(y)
>> y = y/std(y)
```

Do the same thing to the columns of X. Now plot all seven
normalized variables on the same axes. Include a legend.