Octave Tutorial

I. STARTING OCTAVE AND BASIC COMMANDS

1. Open terminal and type octave.
2. An octave prompt will appear.
3. Calculator

```
octave:1> 3+2
ans = 5
octave:2> sin(pi/4)
ans = 0.70711
octave:3> exp(-0.5^2)
ans = 0.77880
```

4. Variables

```
octave:4> a=4
a = 4
octave:5> b=5
b = 5
octave:6> c=a*b
c = 20
octave:7> c=c-4
c = 16
```

II. ARRAYS AND MATRICES

1. Simple arrays

```
  octave:8> v=[1 5 4.6 0.1]
v =

  1.00000  5.00000  4.60000  0.10000
```

2. Interval arrays

```
octave:9> v=1:10
v =

   1   2   3   4   5   6   7   8   9  10

octave:10> v=10:-1:1
v =

  10   9   8   7   6   5   4   3   2   1

octave:11> v=0:0.1:1
v =

 Columns 1 through 8:

  0.00000  0.10000  0.20000  0.30000  0.40000  0.50000  0.60000  0.70000
```

```
 Columns 9 through 11:

  0.80000  0.90000  1.00000
```

3. Transpose

```
octave:12> v=[1 2 3]
v =

  1  2  3

octave:13> v'
ans =

  1
  2
  3
octave:14> v=[i pi e]
v =

  0.00000 + 1.00000i  3.14159 + 0.00000i  2.71828 + 0.00000i

octave:15> v'
ans =

  0.00000 - 1.00000i
  3.14159 - 0.00000i
  2.71828 - 0.00000i
```

4. Accessing elements of an array

```
octave:21> v=[0.1 0.3 0.4 -0.1 -0.2]
v =

   0.10000   0.30000   0.40000  -0.10000  -0.20000

octave:22> v(1)
ans = 0.10000
octave:23> a=v(1)
a = 0.10000
octave:24> v1=v(2:4)
v1 =

   0.30000   0.40000  -0.10000
```

5. Simple matrix

```
octave:16> M=[3.2 -1;1 i]
M =

   3.20000 + 0.00000i  -1.00000 + 0.00000i
   1.00000 + 0.00000i   0.00000 + 1.00000i
octave:17> M=[1 2
> 3 4]
M =
```

```
  1  2
  3  4
```

6. Matrix from smaller matrices

```
octave:18> m1=[1 2;
> 3 4]
m1 =

  1  2
  3  4

octave:19> m2=[1 0;
> 0 1]
m2 =

  1  0
  0  1

octave:20> M=[m1 m2;m2 m1]
M =

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

7. Special matrices

```
octave:25> M=zeros(3)
M =

  0  0  0
  0  0  0
  0  0  0

octave:26> M=ones(3)
M =

  1  1  1
  1  1  1
  1  1  1

octave:27> M=rand(3)
M =

  0.16338  0.88114  0.29410
  0.57624  0.51402  0.63339
  0.24837  0.13677  0.84621

octave:28> M=randn(3)
M =

   0.0053675  -1.3425698  -1.4632978
   0.8983108   0.6619930   0.8550171
  -1.1904200  -0.3376529  -0.2272521

octave:29> v=[1 2 3];M=diag(v)
```

```
M =

  1  0  0
  0  2  0
  0  0  3

octave:30> v=[1 2 3];M=diag(v,1)
M =

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

octave:31> v=[1 2 3];M=diag(v,-1)
M =

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

octave:32> M=rand(3);v=diag(M)
v =

  0.61578
  0.93705
  0.32860

octave:33> v=[1 2 3];M=repmat(v,2,3)
M =

  1  2  3  1  2  3  1  2  3
  1  2  3  1  2  3  1  2  3

octave:34> v=[1 2 3];M=repmat(v,3,2)
M =

  1  2  3  1  2  3
  1  2  3  1  2  3
  1  2  3  1  2  3
```

8. Accessing elements of a matrix

```
octave:25> M=zeros(3)
M =

  0  0  0
  0  0  0
  0  0  0

octave:26> M=ones(3)
M =

  1  1  1
  1  1  1
  1  1  1
```

```
octave:27> M=rand(3)
M =

  0.16338  0.88114  0.29410
  0.57624  0.51402  0.63339
  0.24837  0.13677  0.84621

octave:28> M=randn(3)
M =

   0.0053675  -1.3425698  -1.4632978
   0.8983108   0.6619930   0.8550171
  -1.1904200  -0.3376529  -0.2272521

octave:29> v=[1 2 3];M=diag(v)
M =

  1  0  0
  0  2  0
  0  0  3

octave:30> v=[1 2 3];M=diag(v,1)
M =

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

octave:31> v=[1 2 3];M=diag(v,-1)
M =

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

octave:32> M=rand(3);v=diag(M)
v =

  0.61578
  0.93705
  0.32860

octave:33> v=[1 2 3];M=repmat(v,2,3)
M =

  1  2  3  1  2  3  1  2  3
  1  2  3  1  2  3  1  2  3

octave:34> v=[1 2 3];M=repmat(v,3,2)
M =

  1  2  3  1  2  3
  1  2  3  1  2  3
  1  2  3  1  2  3
```

III. CONTROL STRUCTURES

1. The *for* loop

```
octave:41> for n=1:3
> m=2^n
> endfor
m = 2
m = 4
m = 8
octave:42> v=1:2:6;for n=v
> m=2^n
> endfor
m = 2
m = 8
m = 32
```

2. The *if* structure

```
octave:43> m=[1:10];
octave:44> for n=1:length(m)
> if m(n)<5
> printf("Number %f\n",m(n));
> endif
> endfor
Number 1.000000
Number 2.000000
Number 3.000000
Number 4.000000
```

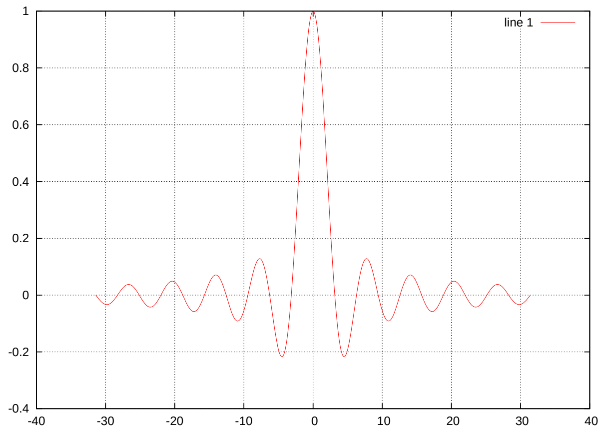
IV. PLOTTING

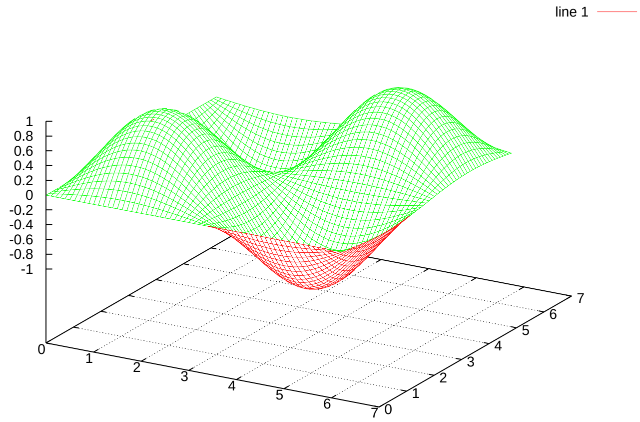
1. Two-dimensional (see figure)

```
octave:49> y=sin(theta)./theta;
octave:50> plot(theta,y)
octave:51> grid on
octave:52> print("sinc.eps","-depsc")
```

2. Three-dimensional (see figure)

```
octave:61> x=0:0.1:1;y=x;
octave:62> [xx,yy]=meshdom(x,y);
octave:63> mesh(xx,yy,sin(xx).*sin(yy))
```





V. USER-DEFINED FUNCTIONS AND SCRIPTS

1. A function that calculates the Taylor expansion of $\exp(x)$ for a given x and order N. At the same time it draws a progressive plot of the expansion at each order against the real function.

```
hande@p439a:~/teaching/phys741/octave-tutorial$ cat taylor.m
## A function that calculates the Taylor expansion of $exp(x)$ for a
## given $x$ and order $N$. At the same time it draws a progressive plot of
## the expansion at each order against the real function.

function f=taylor(x,N)

  f=ones(size(x));

  for n=1:N
    f+=(1/prod([1:n]))*x.^n;
    plot(x,f,'r-;Expansion;',x,exp(x),'b-;Real function;');
    pause
  endfor
endfunction
```

2. Script that creates a two-dimensional crystal

```
## Lengths of lattice vectors
a=1.5;  b=2;
## Angle between lattice vectors
theta=pi/5;

## Lattice vectors
a1=a*[1 0];
a2=b*[cos(theta) sin(theta)];

## Basis atoms
u1=0.3; u2=0.4;
basis=[0 0
       a1*u1+a2*u2];
Nbasis=size(basis,1);

## Atoms
atoms=[];
```

```
for n1=0:N1-1
  for n2=0:N2-1
    atoms=[atoms;
           basis+repmat(n1*a1+n2*a2,Nbasis,1)];
  endfor
endfor

plot(atoms(:,1),atoms(:,2),'b*');
```