GALGOTIAS UNIVERSITY

NAAC GRADE A+ Accredited University

Plot No.2, Sector 17-A, Yamuna Expressway,
Greater Noida, Gautam Buddha Nagar, U.P., India

School of Computer Science and Engineering

Lab Report

B. Tech- II Sem

Winter: 2023-24

Engineering Mathematics-II(Lab)

(C1UC222B)

Submitted by:

Name:.........................

Batch:...… … … … … ..

Adm No:… … … … … .

Submitted to:

……………………..

Scanned with OKEN Scanner

S.No.	List of Experiments
1.	
2.	
3.	
4.	
5.	
6.	
7.	
8.	
9.	
10.	
11.	
12.	

Scanned with OKEN Scanner

S. No.	Experiment	Date (Performed)	Signature (with date)
1.			
2.			
3.			
4.			
5.			
6.			
7.			
8.			
9.			
10.			
11.			
12.			

Scanned with OKEN Scanner

Experiment :- 1 Vectors in 2D and 3D

Exercises

To write a Scilab code of the following problems:

Q1. (i) Create a row vector with 3 elements.

```
--> M=[1 4 2]
 M  =

   1.   4.   2.
```

(ii) Create a column vector with 4 elements.

```
--> U=[1;7;0;8]
 U  =

   1.
   7.
   0.
   8.
```

2. By taking the first term a=1 and the last term b=10 create a vector:

(i) By taking the spacing between two consecutive terms d=2,

```
--> S=[1:2:10]
 S  =

   1.   3.   5.   7.   9.
```

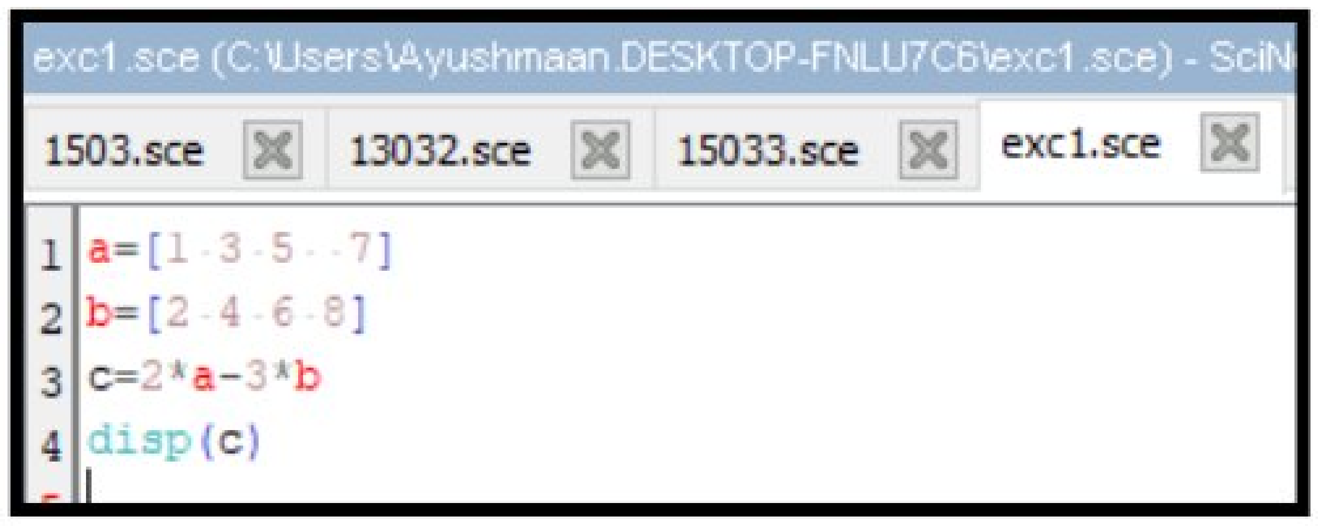
(ii) By taking the number of terms n= 12

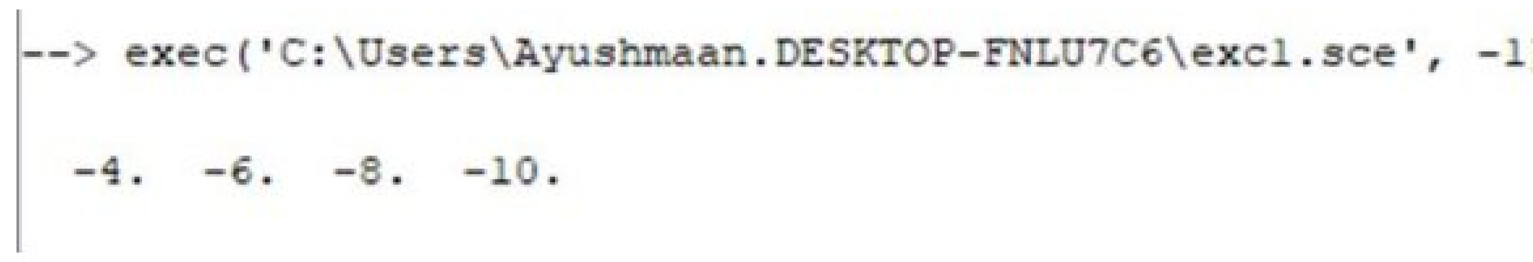
```
--> K=linspace(1,10,12)
 K  =

         column 1 to 9

   1.   1.8181818   2.6363636   3.4545455   4.2727273   5.0909091   5.9090909   6.7272727   7.5454545

         column 10 to 12

   8.3636364   9.1818182   10.
```

Scanned with OKEN Scanner

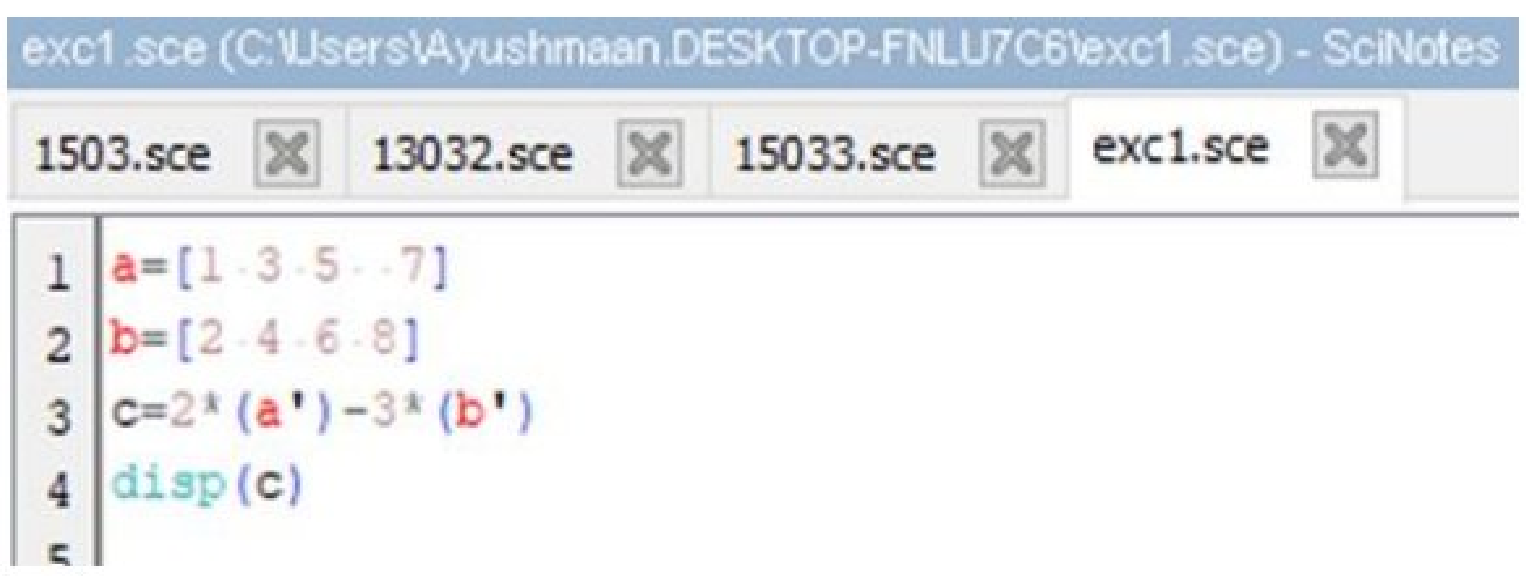
(iii) Create two-row vectors a and b such that the following operations are defined and hence find :

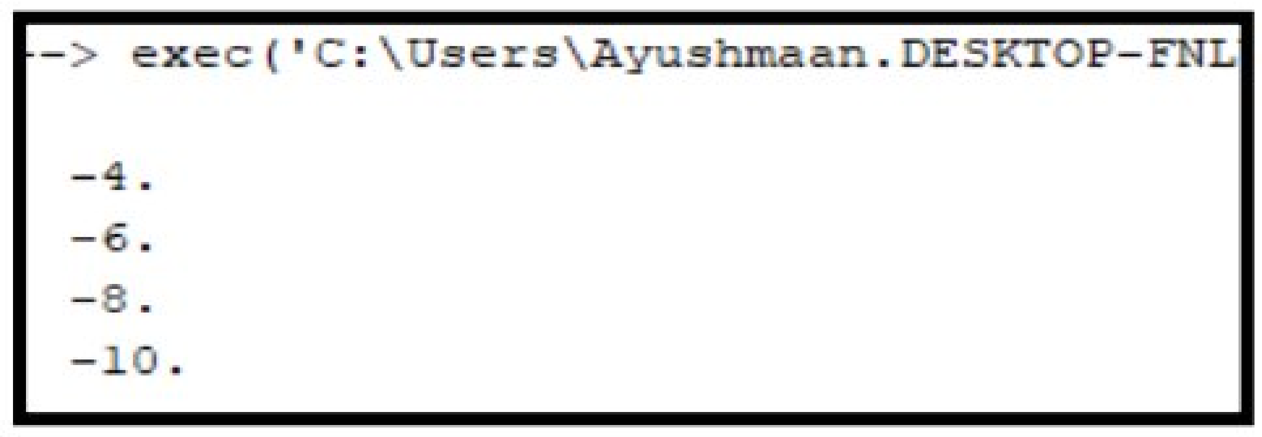
(i) 2a-3b





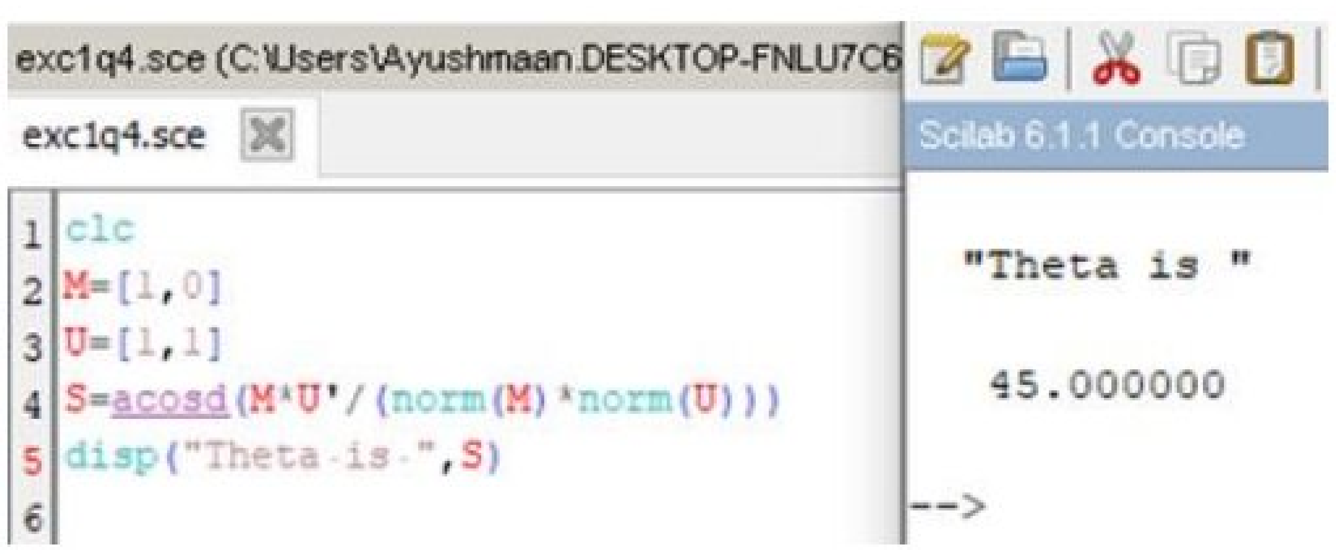
(ii) 2(transpose a) - 3(transpose b)



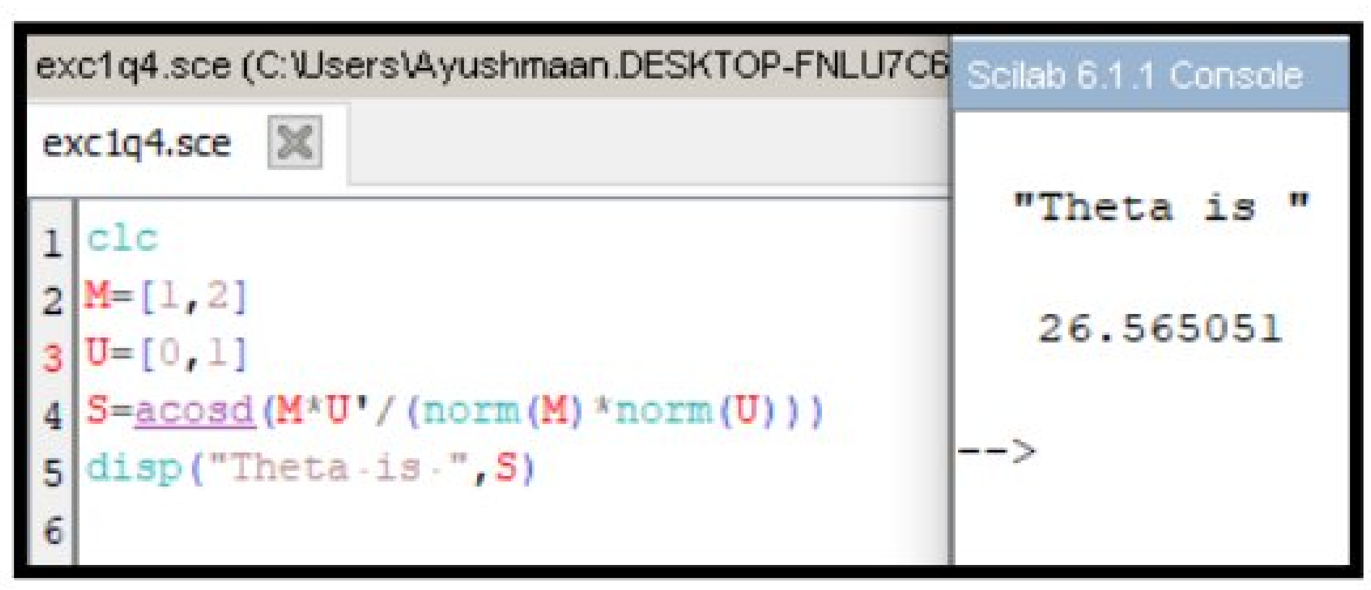


4. Find the angle between the following pair of vectors

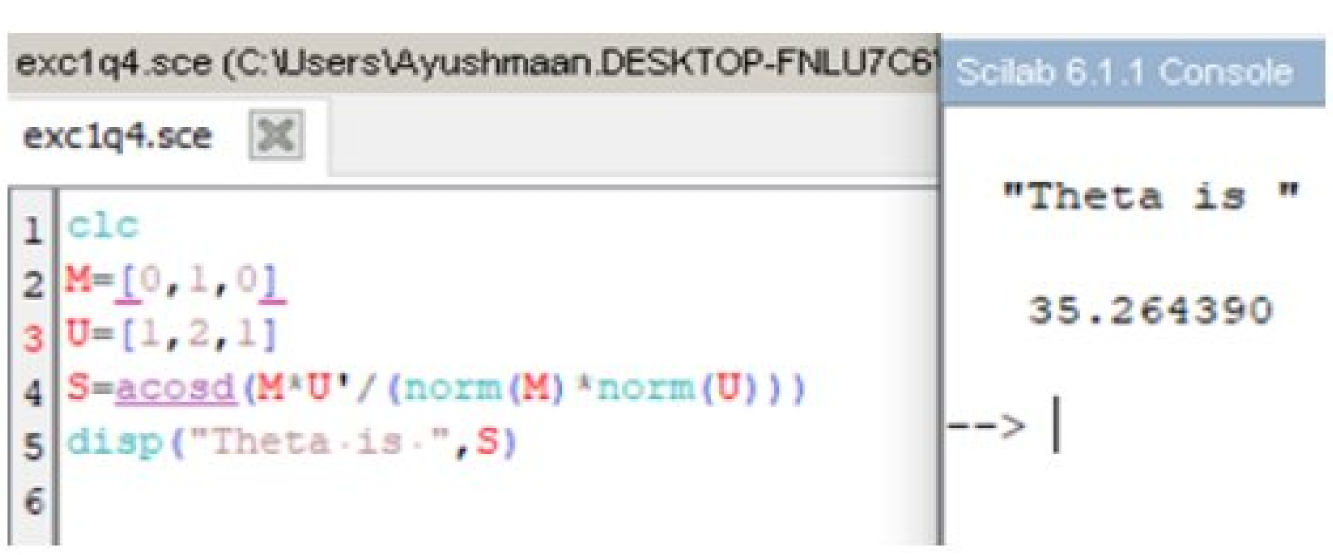
(i) (0,0),(1,1)



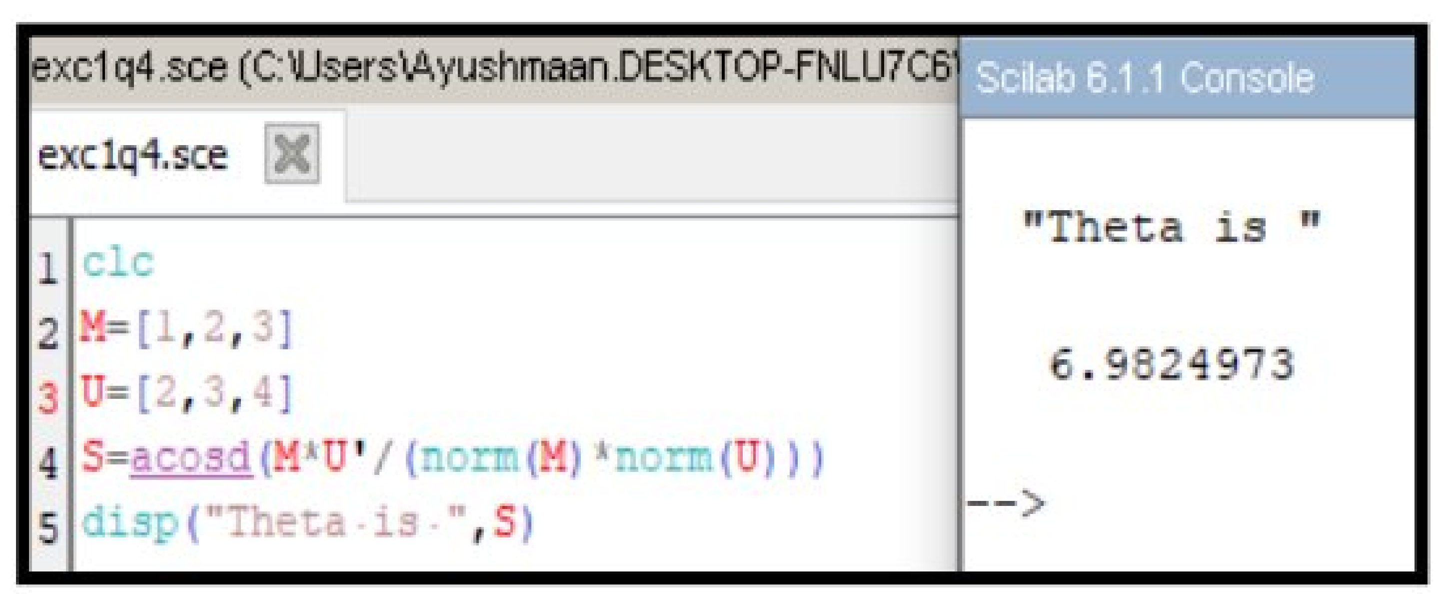
(ii) (1,2), (0,1)



(iii) (0,1,0), (1,2,1)



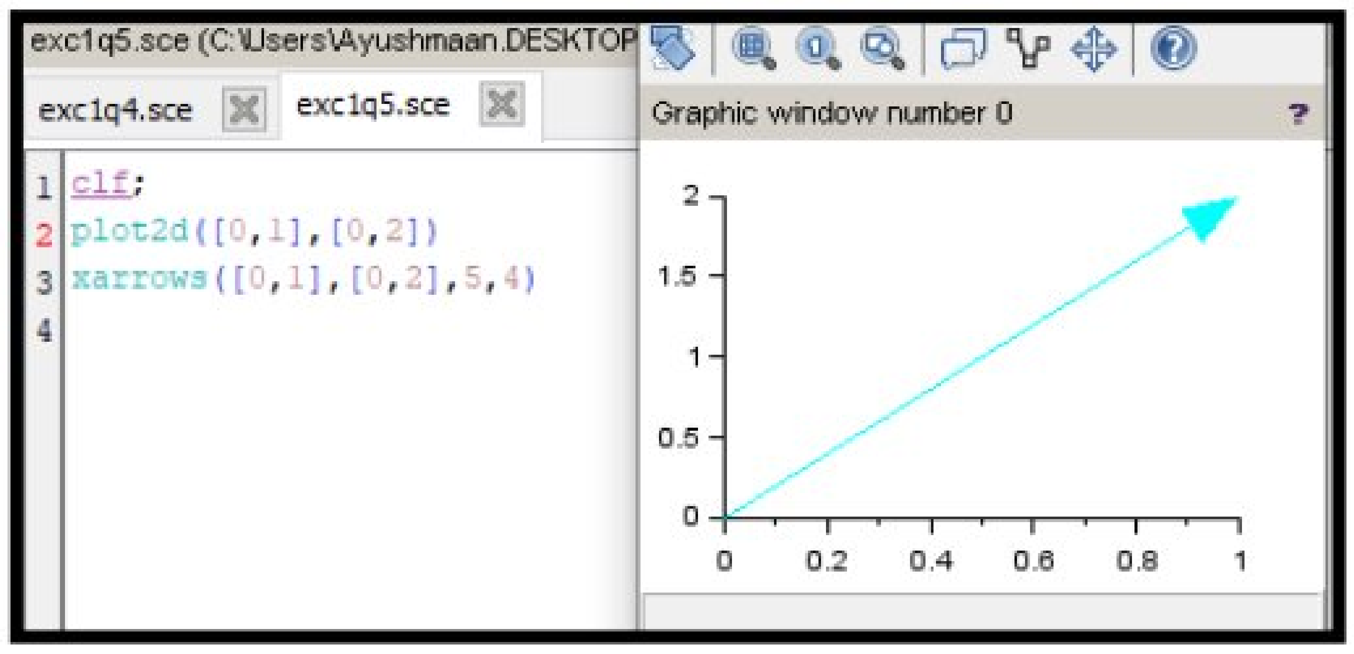
(iv) (1,2,3), (2,3,4)



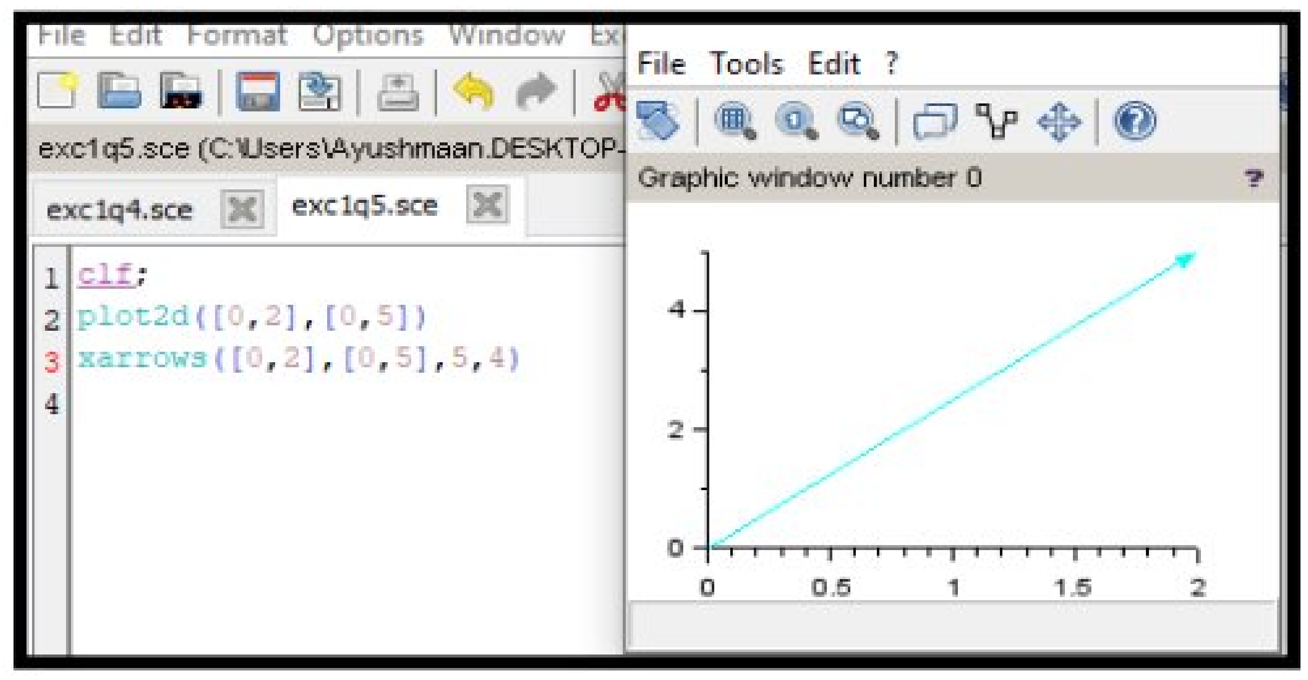
Scanned with OKEN Scanner

5. Draw the arrows for the following vectors:

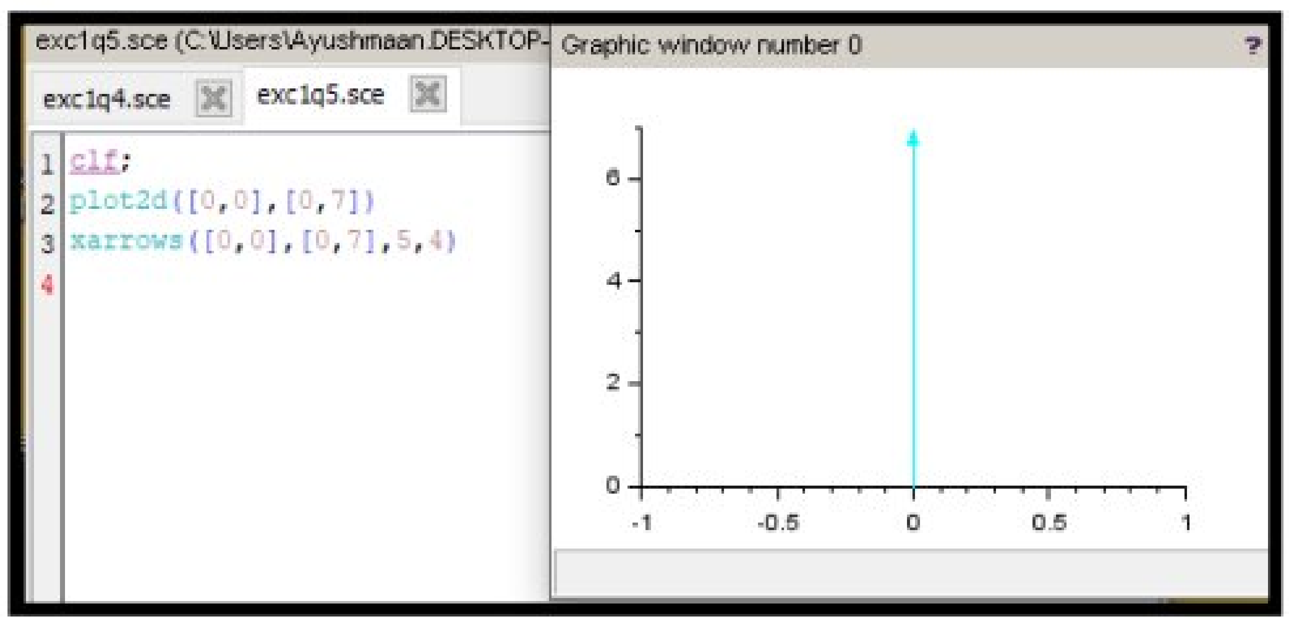
(i) (1,2)



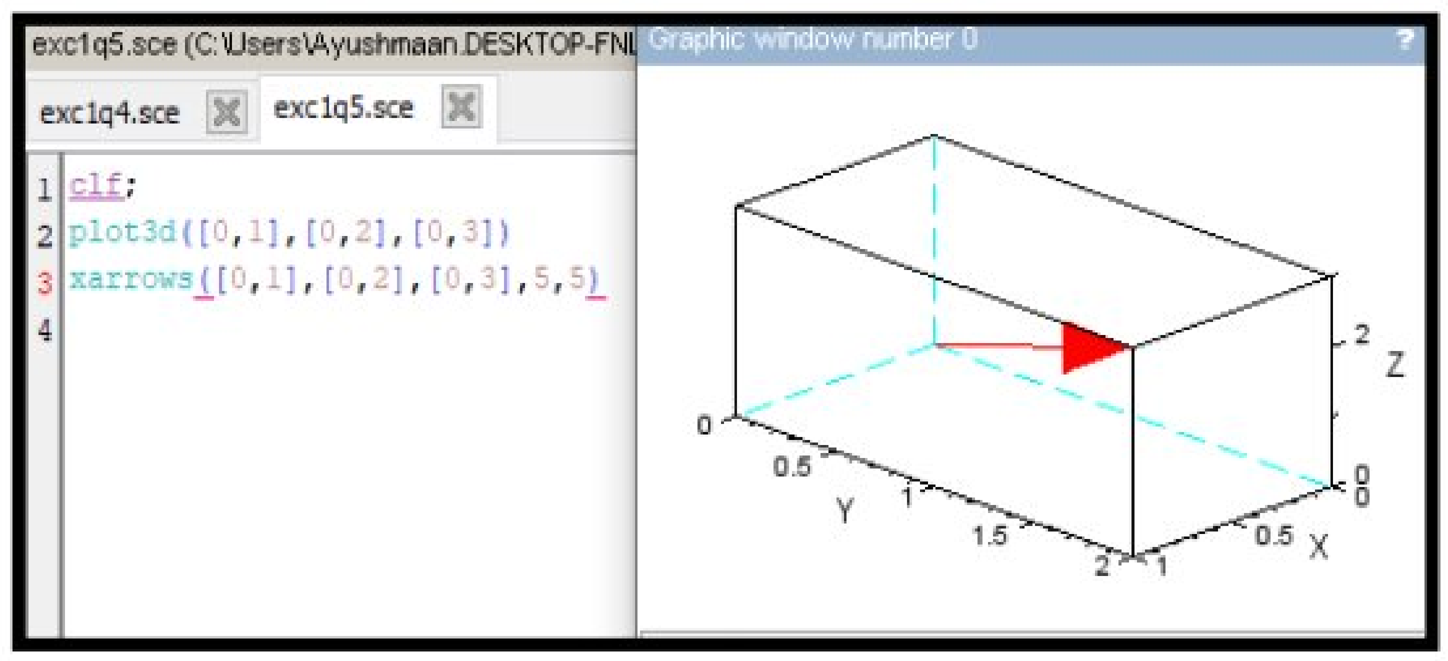
(ii) (2,5)



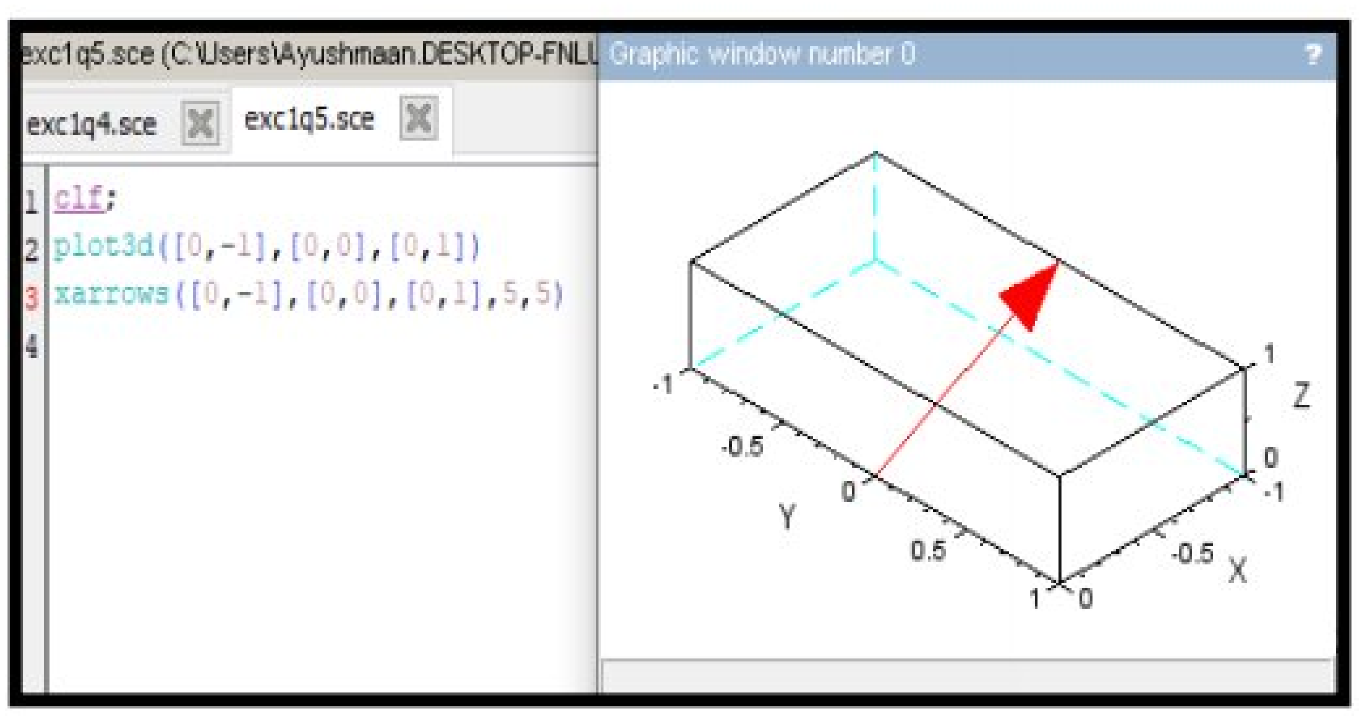
(iii) (0,7)



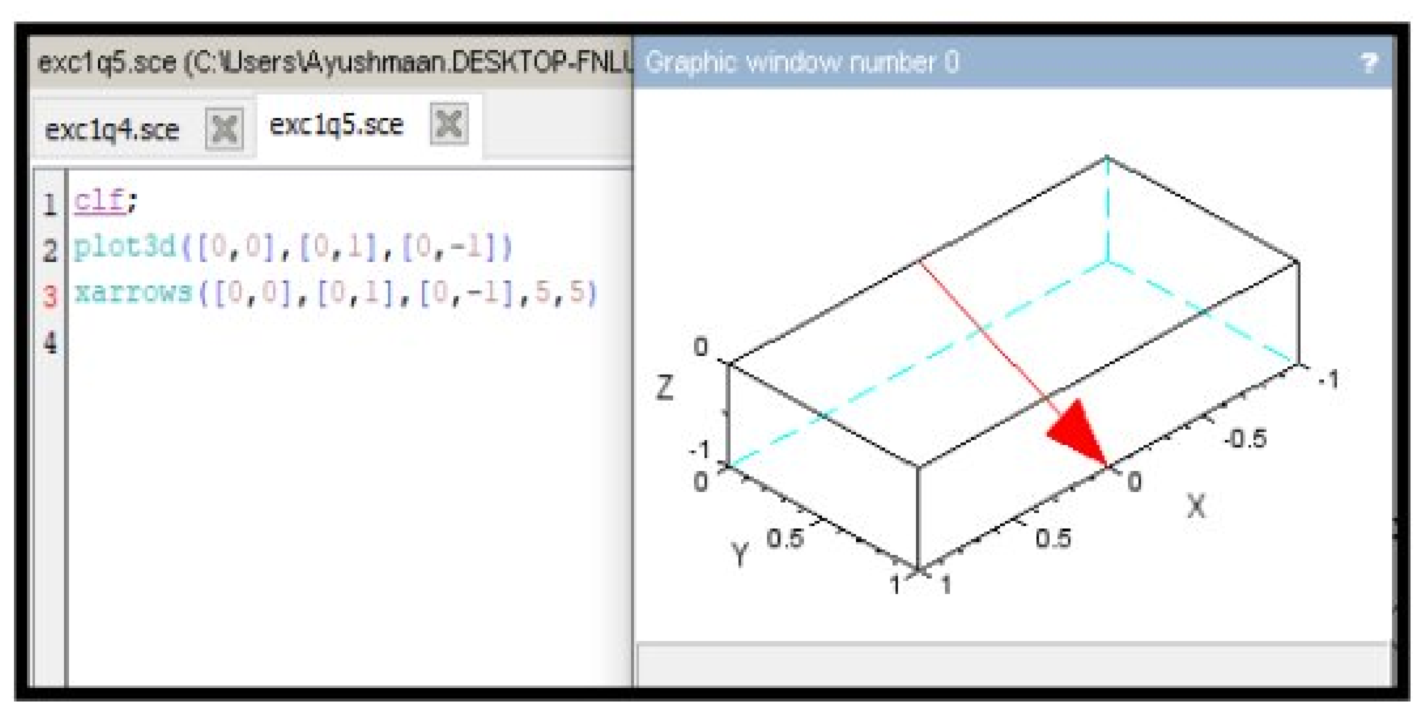
(iv) (1,2,3)



(v) (-1,0,1)



(vi) (0,1,-1)



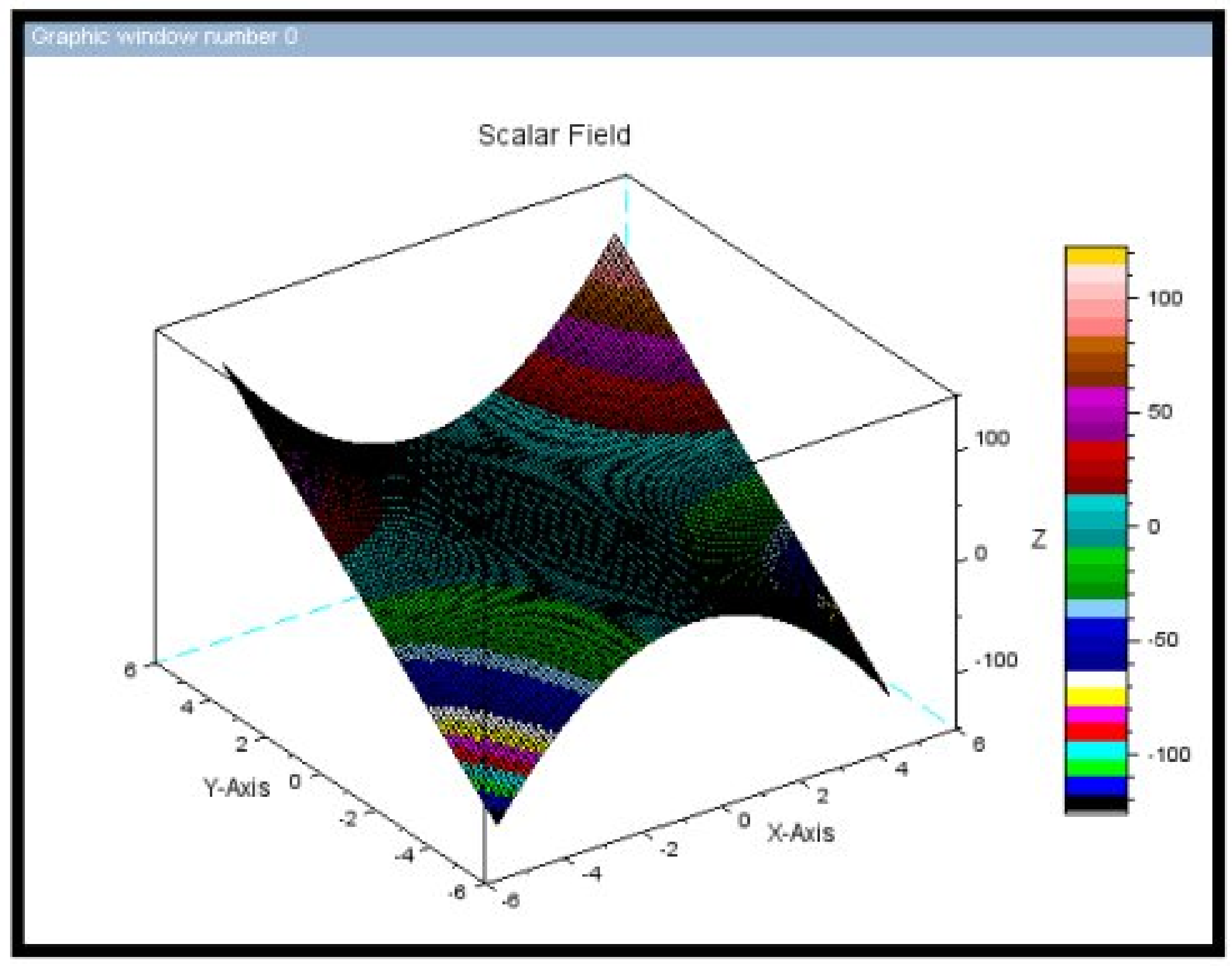
Scanned with OKEN Scanner

Experiment :- 2

Exercises:

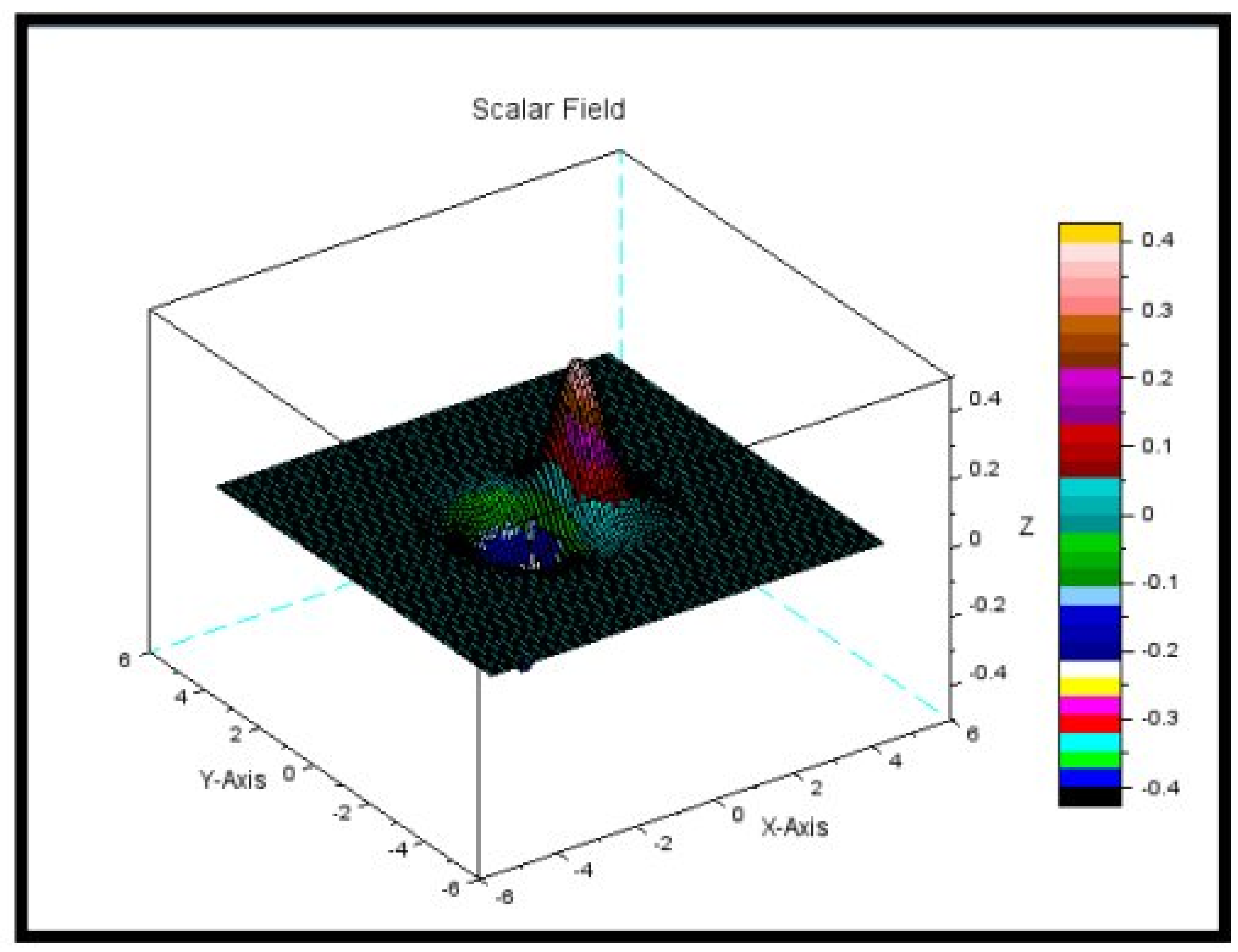
1) Plotting of a Scalar Field $f(x,y)=x^{2y}$

```
clc; clear; clf;
function z=scalarfield(x, y)
   z=(x.^2).*y
endfunction
x=linspace(-5,5,100)
y=linspace(-5,5,100)
[X,Y]=meshgrid(x,y)
z=scalarfield(X,Y)
surf(x,y,z)
xtitle('Scalar Field','X-Axis','Y-Axis')
colorbar
```



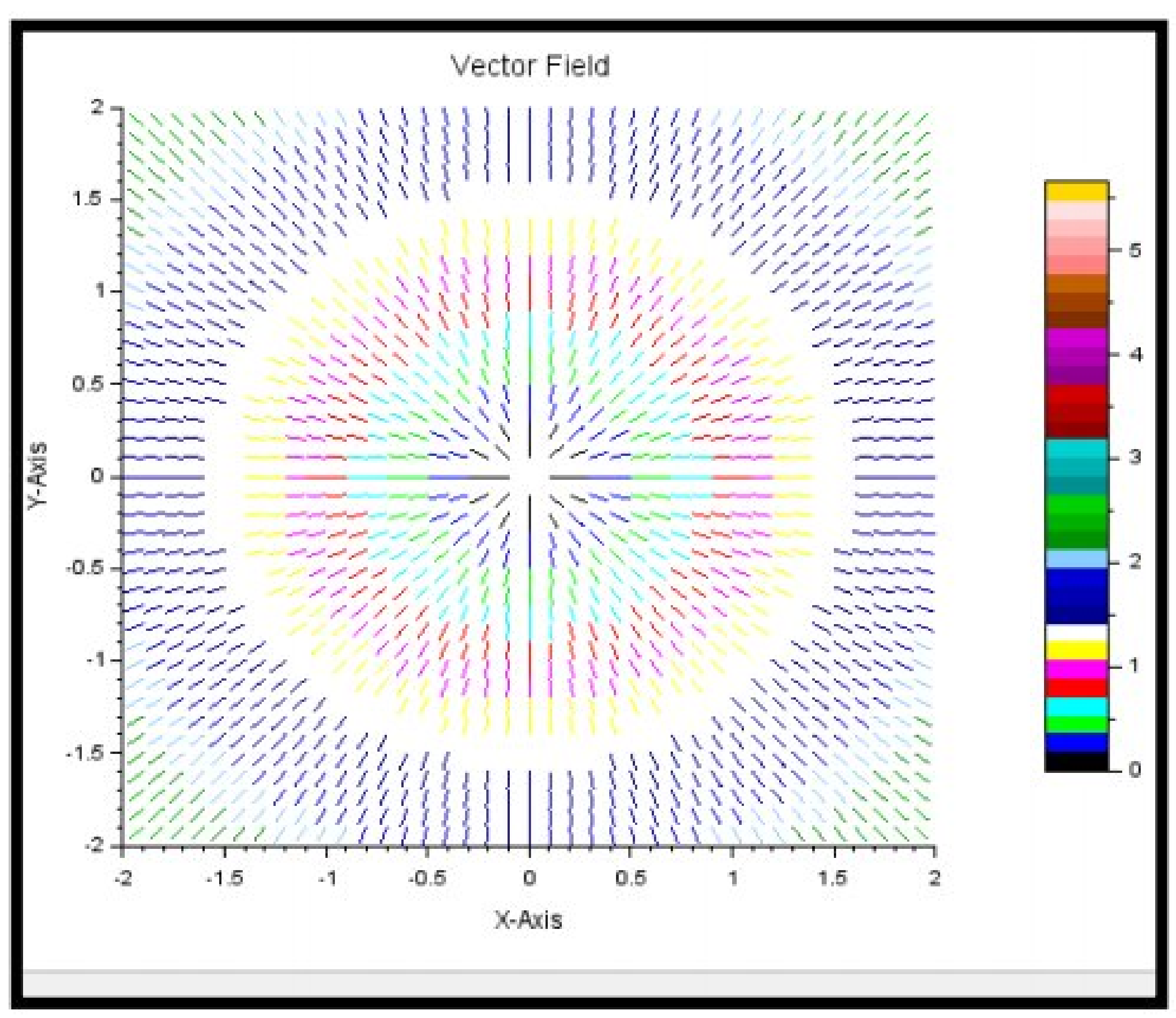
2) Plotting of a Scalar Field $f(x,y)= x*e^{-x^2+y^2}$

```
clc; clear; clf;

function z=scalarfield(x, y)
   z=x.*exp(-x.^2-y.^2)
endfunction
x=linspace(-5,5,100)
y=linspace(-5,5,100)
[X,Y]=meshgrid(x,y)
z=scalarfield(X,Y)
surf(x,y,z)
xtitle('Scalar Field','X-Axis','Y-Axis')
colorbar
```



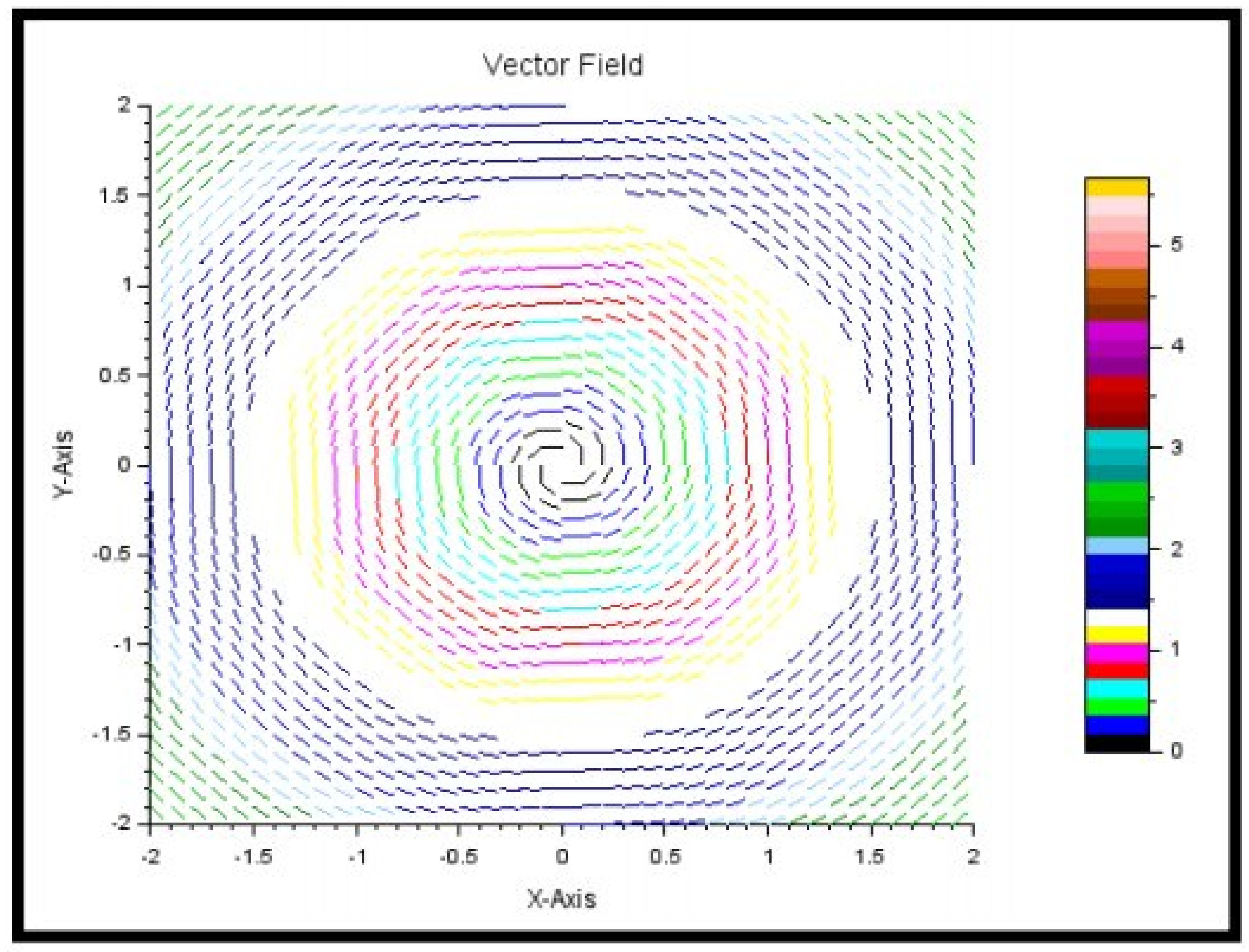
3) Plotting of vector field f(x,y)= (x,y)

```
clc; clear; clf;
function [Zx, Zy]=vfield(x, y)
   Zx=x
   Zy=y
endfunction
x=-4:.1:4
y=-4:.1:4
[X,Y]=ndgrid(x,y)
[Zx,Zy]=vfield(X,Y)
champ(x,y,Zx,Zy,0.2,rect=[-2,-2,2,2])
gce.colored="on"
xtitle('Vector Field','X-Axis','Y-Axis')
colorbar
```



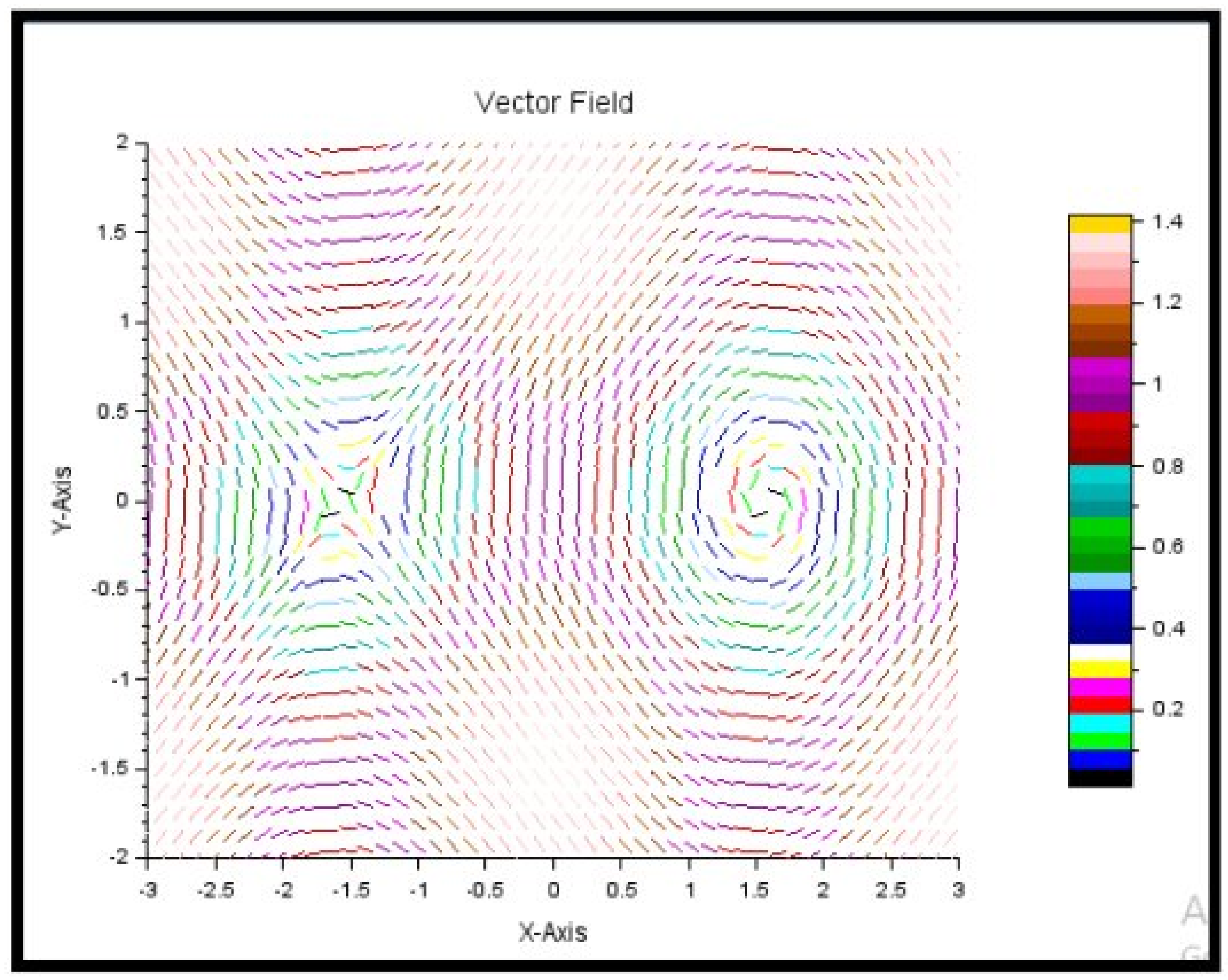
Scanned with OKEN Scanner

4) Plotting of vector field f(x,y)= (-y,x)




```
clc; clear; clf;
function [Zx, Zy]=vfield(x, y)
   Zx=-y
   Zy=x
endfunction
x=-4:.1:4
y=-4:.1:4
[X,Y]=ndgrid(x,y)
[Zx,Zy]=vfield(X,Y)
champ(x,y,Zx,Zy,0.2,rect=[-2,-2,2,2])
gce.colored="on"
xtitle('Vector Field','X-Axis','Y-Axis')
colorbar
```

5) Plotting of vector field f(x,y)= (Sin y,Cos x)

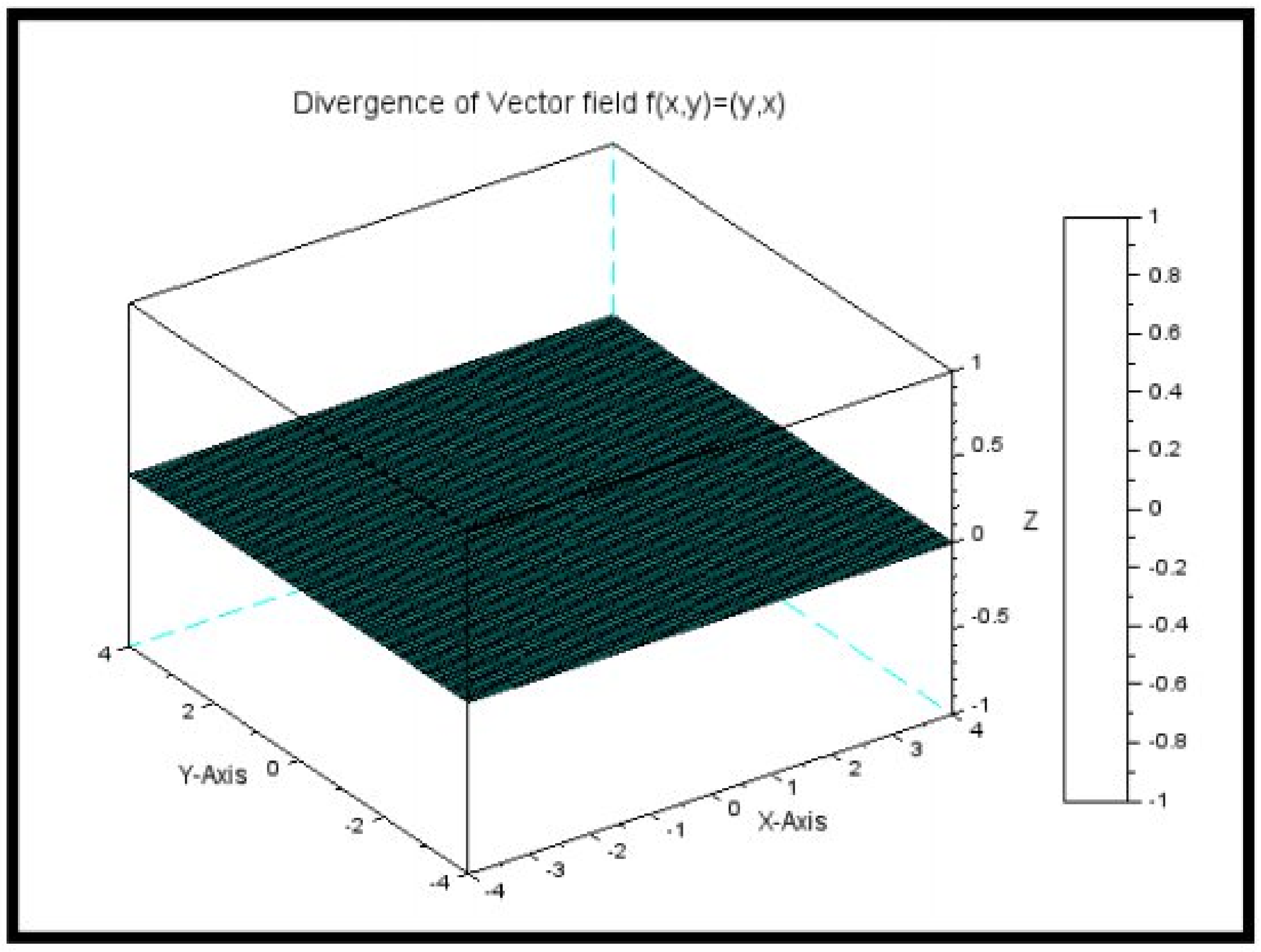


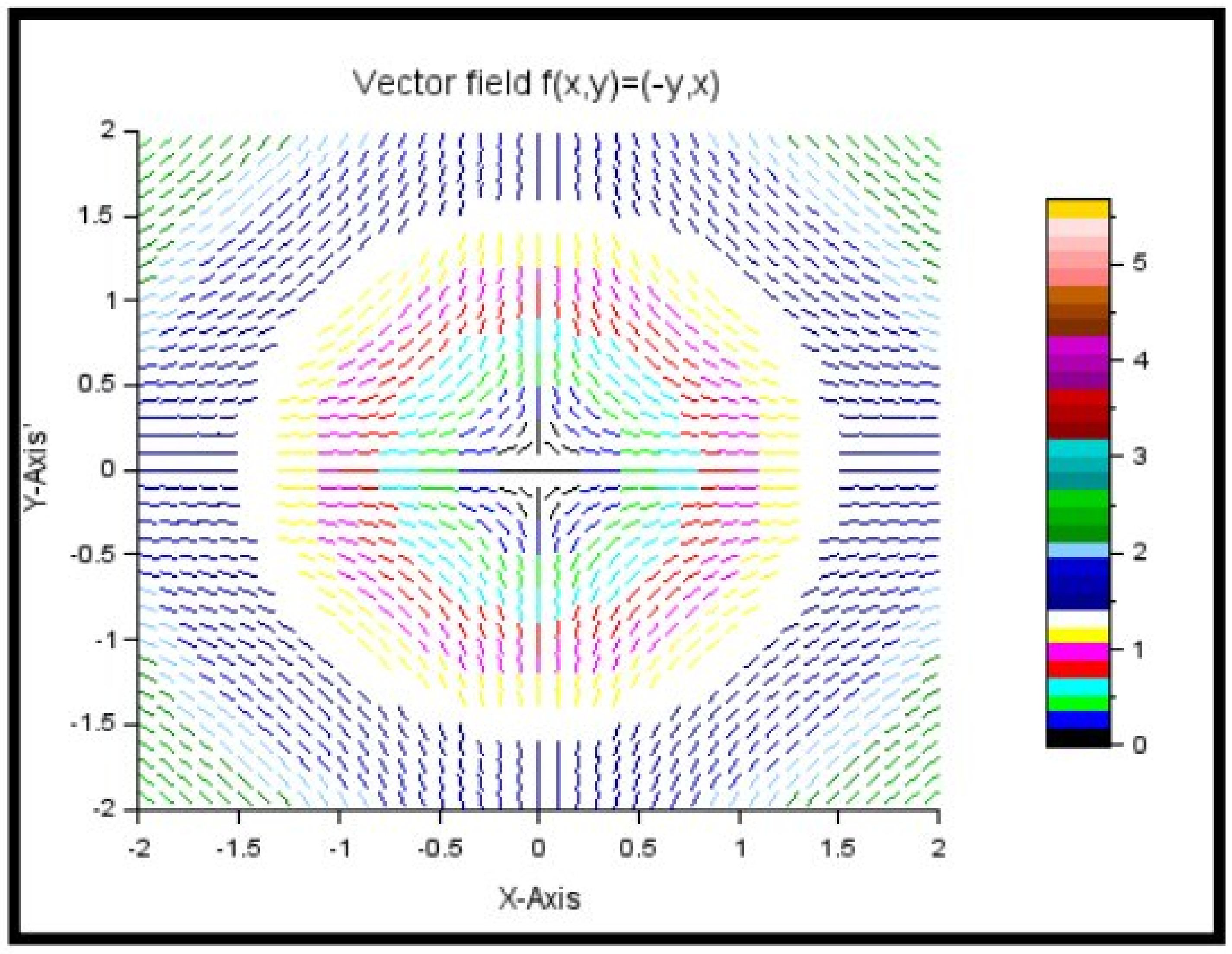

```
clc; clear; clf;
function [Zx, Zy]=vfield(x, y)
   Zx=sin(y)
   Zy=cos(x)
endfunction
x=linspace(-1,1,100)*2*%pi
y=linspace(-1,1,100)*2*%pi
[X,Y]=ndgrid(x,y)
[Zx,Zy]=vfield(X,Y)
champ(x,y,Zx,Zy,0.2,rect=[-3,-2,3,2])
gce.colored="on"
xtitle('Vector Field','X-Axis','Y-Axis')
colorbar
```

6) Divergence of vector field f(x,y)=(-y,x)

```
clc; clear; clf;
function [Zx, Zy, Div]=vfield(x, y)
Zx=-y; Zy=x; Div=0*x
endfunction
x=-4:.1:4; y=-4:.1:4; [X,Y]=meshgrid(x,y)
[Zx,Zy,Div]=vfield(X,Y)
surf(x,y,Div)
xtitle('Divergence of Vector field f(x,y)=(y,x)','X-Axis','Y-Axis'); colorbar;
scf();
champ(x,y,Zx,Zy,0.2,rect=[-2,-2,2,2])
gce().colored="on"
xtitle('Vector field f(x,y)=(-y,x)','X-Axis','Y-Axis”);
colorbar
```

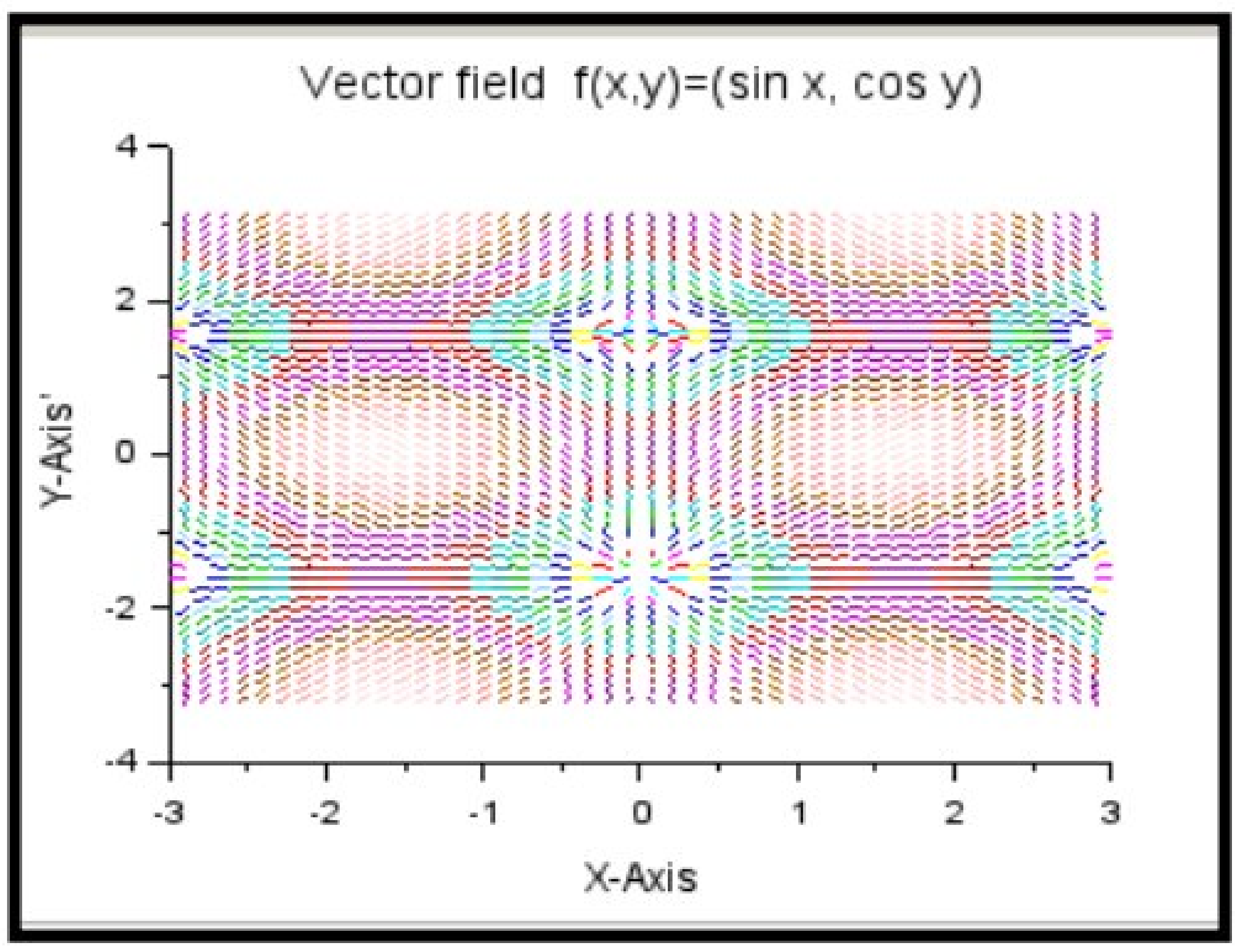
Scanned with OKEN Scanner

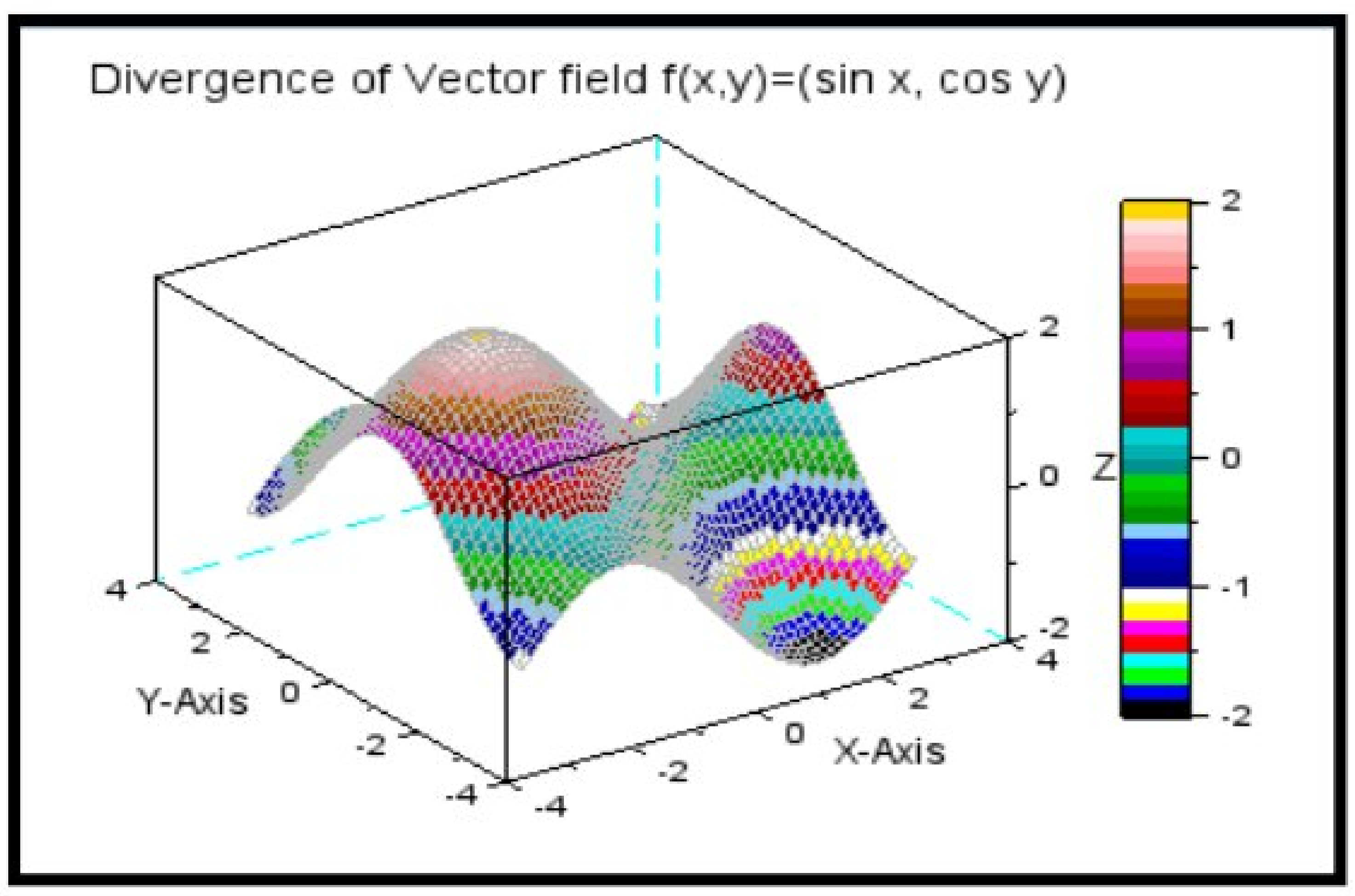




7) Divergence of vector field f(x,y)=(sin x , cos y)

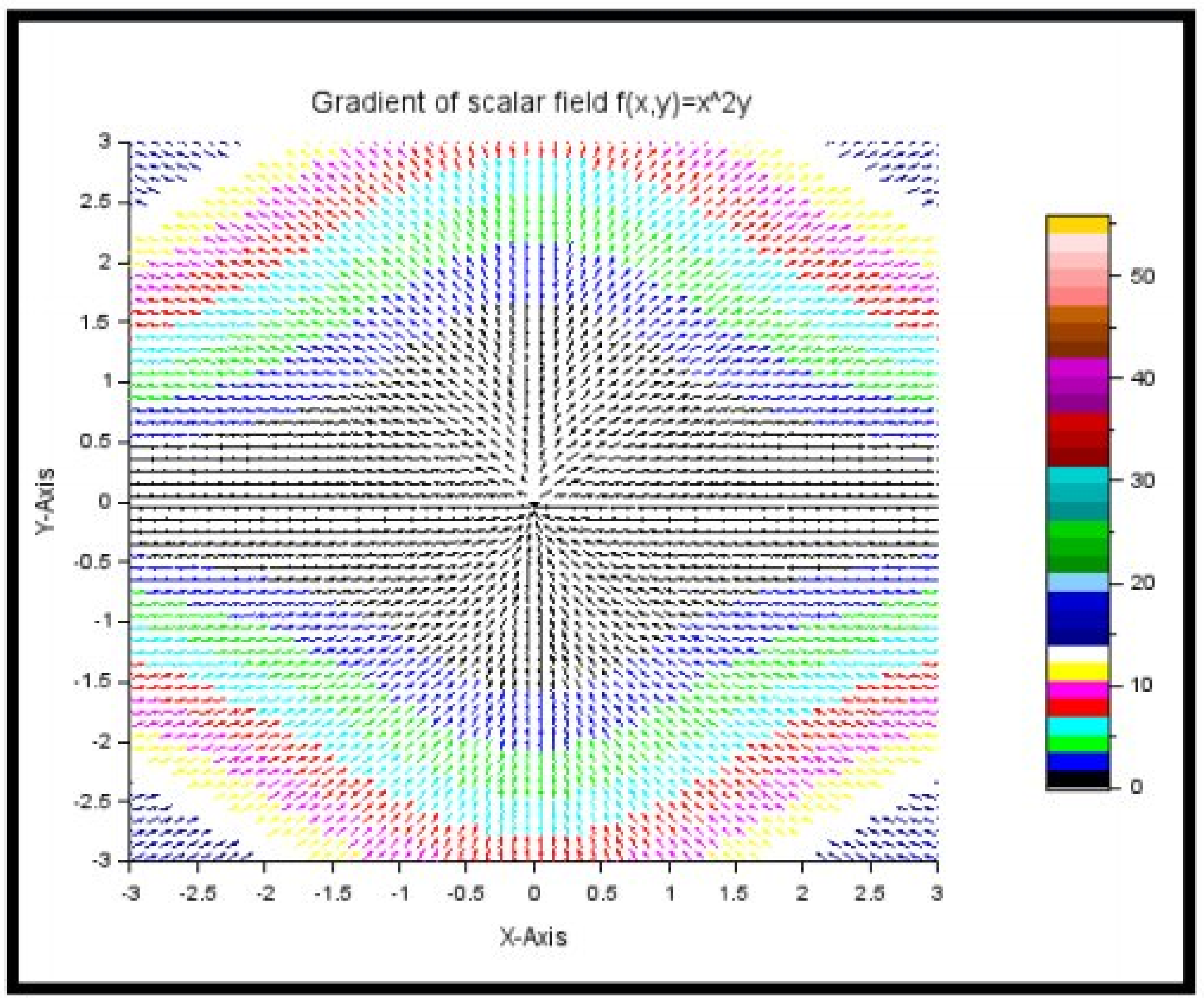
```
clc; clear; clf;
function [Zx, Zy, Div]=vfield(x, y)
Zx=sin (x); Zy=cos (y); Div=cos(x)- sin(y)
endfunction
x=linspace(-1,1,50)*%pi; y=linspace(-1,1,50)*%pi
[X,Y]=ndgrid(x,y)
[Zx,Zy,Div]=vfield(X,Y)
surf(x,y,Div)
xtitle('Divergence of Vector field f(x,y)=(sin x, cos y)','X-Axis','Y-Axis'); colorbar; scf;
champ(x,y,Zx,Zy,0.2,rect=[-3,-%pi,2.5,%pi]); gce().colored="on";
 xtitle('Vector field  f(x,y)=(sin x, cos y)','X-Axis','Y-Axis'');
```

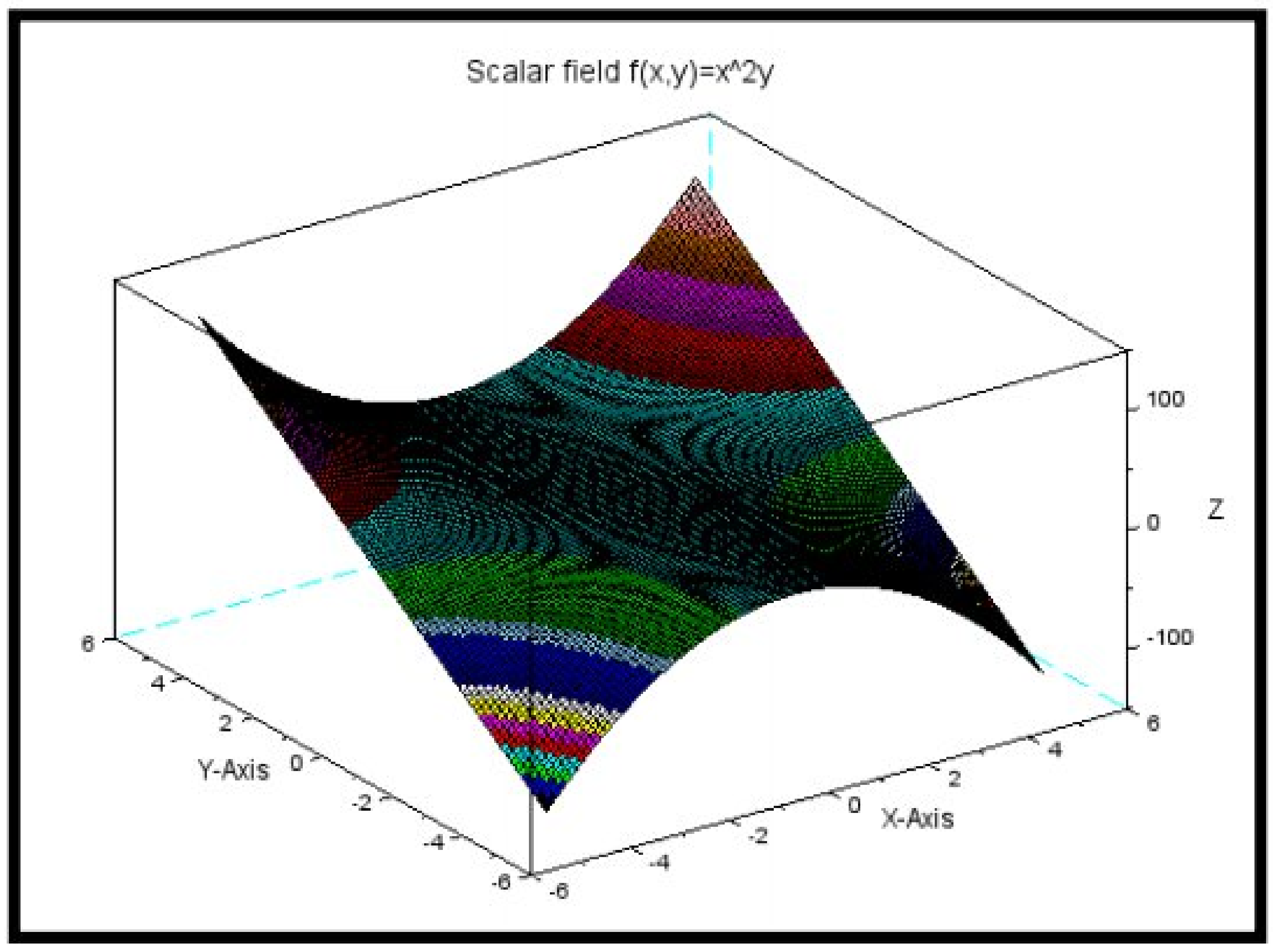




8) The gradient of scalar field $f(x,y)=x^2y$

```
clc; clear; clf;
function [z, DZx, DZy]=scalarfield(x, y)
   z = x.^2 .* y;
   DZx = 2 * x .* y;
   DZy = x.^2;
endfunction
x = linspace(-5, 5, 100);
y = linspace(-5, 5, 100);
[X, Y] = meshgrid(x, y);
[z, DZx, DZy] = scalarfield(X, Y);
surf(x, y, z);
xtitle('Scalar field f(x,y)=x^2y', 'X-Axis', 'Y-Axis');  scf();
champ(x, y, DZx, DZy, 0.5, rect=[-3, -3, 3, 3]);
gce().colored = "on";
xtitle('Gradient of scalar field f(x,y)=x^2y', 'X-Axis', 'Y-Axis');
colorbar
```





Experiment :- 3 Linear dependence and Independence

Exercises:

Determine LI of the following vectors:

1) (0,1), (1,0)

```
clc
clear
v1=[0,1]
v2=[1,0]
n=input('Enter the number of vector: ')
A=[v1;v2]
if rank (A)== n then
    disp('L.I.')
else
    disp('L.D.')
end
```

```
Enter the number of vector: 2

  "L.I."

-->
```

2) (2,4), (1,2)

```
clc
clear
v1=[2,4]
v2=[1,2]
n=input('Enter the number of vector: ')
A=[v1;v2]
if rank (A)== n then
    disp('L.I.')
else
    disp('L.D.')
end
```

```
Enter the number of vector: 1

  "L.I."

-->
```

Scanned with OKEN Scanner

3) (1,1), (1,3), (2,5)

```
clc
clear
v1=[1,1]
v2=[1,3]
v3=[2,5]
n=input('Enter the number of vector: ')
A=[v1;v2;v3]
if rank (A)== n then
    disp('L.I.')
else
    disp('L.D.')
end
```

```
Enter the number of vector: 2

  "L.I."

-->
```

4) (1,2,3), (1,2,4)

```
clc
clear
v1=[1,2,3]
v2=[1,2,4]
n=input('Enter the number of vector: ')
A=[v1;v2]
if rank (A)== n then
    disp('L.I.')
else
    disp('L.D.')
end
```

```
Enter the number of vector: 2

  "L.I."

-->
```

5) (1,1,0), (1,0,1), (0,1,1)

```
clc
clear
v1=[1,1,0]
v2=[1,0,1]
v3=[0,1,1]
n=input('Enter the number of vector: ')
A=[v1;v2;v3]
if rank (A)== n then
    disp('L.I.')
else
    disp('L.D.')
end
```

```
Enter the number of vector: 3

  "L.I."

-->
```

6) (2,2,1), (1,-1,1), (1,0,1)

```
clc
clear
v1=[2,2,1]
v2=[1,-1,1]
v3=[1,0,1]
n=input('Enter the number of vector: ')
A=[v1;v2;v3]
if rank (A)== n then
    disp('L.I.')
else
    disp('L.D.')
end
```

```
Enter the number of vector: 3

  "L.I."

-->
```

Scanned with OKEN Scanner

7)(1,2,3,1), (2,1,-1,1), (4,5,5,3), (5,4,1,3)

```
clc
clear
v1=[1,2,3,1]
v2=[2,1,-1,1]
v3=[4,5,5,3]
v4=[5,4,1,3]
n=input('Enter the number of vector: ')
A=[v1;v2;v3;v4]
if rank (A)== n then
    disp('L.I.')
else
    disp('L.D.')
end
```

```
Enter the number of vector: 2

  "L.I."

-->
```

8) (1,2,3), (1,2,4), (0, 0, 0)

```
clc
clear
v1=[1,2,3]
v2=[1,2,4]
v3=[0,0,0]
n=input('Enter the number of vector: ')
A=[v1;v2;v3]
if rank (A)== n then
    disp('L.I.')
else
    disp('L.D.')
end
```

```
Enter the number of vector: 2

  "L.I."

-->
```

Scanned with OKEN Scanner

Experiment :- 4 Matrix representation of a linear Transformation

To find Matrix representation of any linear transformation T and use it to determine the inverse of T if the inverse exists.

Let T: $\boldsymbol{R}3$ to $\boldsymbol{R}3$ be a linear transformation with basis B={(1, 1,1), (1,1,0), (1,0,0)} and B' ={(1, 0,1), (-1,2,1), (2,1,1)} respectively and defined by T(a,b,c)= (3a+b, a+b, a-b).

(i) Find matrix [T;B,B']
(ii) Determine, if T is invertible

```
// Define basis vectors in B
b1 = [1; 1; 1];
b2 = [1; 1; 0];
b3 = [1; 0; 0];
// Define basis vectors in B'
bp1 = [1; 0; 1];
bp2 = [-1; 2; 1];
bp3 = [2; 1; 1];
// Define the linear transformation function
function [y] = T(x)
  y = [3*x(1) + x(2); x(1) + x(2); x(1) - x(2)];
endfunction
// Apply T to each basis vector in B
T_b1 = T(b1);
T_b2 = T(b2);
T_b3 = T(b3);
// Create the transformed basis in B' (optional, not used directly)
transformed_basis_B = [T_b1 T_b2 T_b3];
// Create the matrix representation of B' (optional, not used directly)
B_prime = [bp1 bp2 bp3];
// Calculate the coordinates of T(v1), T(v2), and T(v3) with respect to B'
C1 = inv(B_prime) * T(b1);
C2 = inv(B_prime) * T(b2);
C3 = inv(B_prime) * T(b3);
// Combine the coordinates into a single matrix
C = [C1 C2 C3];
// Display the coordinates of T(v1), T(v2), and T(v3) with respect to B'
disp(C);// Check for invertibility (assuming T_BBp is square)

// Check for invertibility
det_C = det(C);
if (det_C~=0)
  disp("T is invertible.");
else
  disp("T is not invertible.");
end
```

```
-2.5   -2.5   -0.25
-0.5   -0.5   -0.25
 3.     3.     1.5

"T is invertible."
```

Scanned with OKEN Scanner

Find the matrix representation of linear map T given below with respect to the user input basis of domain and codomain.

(a) T(x,y,x) = (x+y,y+z,z+x)

```
clc;
clear;
function T = linear(v)
    T = [v(1) + v(2); v(2) + v(3); v(3) + v(1)];
endfunction
v1 = input("Enter v1 of Basis of domain vector space: ");
v2 = input("Enter v2 of Basis of domain vector space: ");
v3 = input("Enter v3 of Basis of domain vector space: ");

w1 = input("Enter w1 of Basis of co-domain vector space: ");
w2 = input("Enter w2 of Basis of co-domain vector space: ");
w3 = input("Enter w3 of Basis of co-domain vector space: ");
Tv1 = linear(v1);
Tv2 = linear(v2);
Tv3 = linear(v3);

B = [w1, w2, w3]';

c1 = B \ Tv1;
c2 = B \ Tv2;
c3 = B \ Tv3;
C = [c1, c2, c3];

disp("Matrix of Linear map with respect to (A,B) is given by the matrix:");
disp(C);
```

```
Enter v1 of Basis of domain vector space: [1;0;0]

Enter v2 of Basis of domain vector space: [0;1;0]

Enter v3 of Basis of domain vector space: [0;0;1]

Enter w1 of Basis of co-domain vector space: [1;1;0]

Enter w2 of Basis of co-domain vector space: [1;0;1]

Enter w3 of Basis of co-domain vector space: [0;1;1]


  "Matrix of Linear map with respect to (A,B) is given by the matrix:"

   0.   1.   0.
   1.   0.   0.
   0.   0.   1.
```

Scanned with OKEN Scanner

(b) T(x,y,z) = (x-y,y-z,z-x)

```
clc;
clear;
function T = linear_a(v)
    T = [v(1) - v(2); v(2) - v(3); v(3) - v(1)];
endfunction


v1 = [1; 0; 0];
v2 = [0; 1; 0];
v3 = [0; 0; 1];

w1 = [1; 1; 0];
w2 = [1; 0; 1];
w3 = [0; 1; 1];

Tv1 = linear_a(v1);
Tv2 = linear_a(v2);
Tv3 = linear_a(v3);

B = [w1, w2, w3]';

c1 = B \ Tv1;
c2 = B \ Tv2;
c3 = B \ Tv3;

C = [c1, c2, c3];

disp("Matrix of Linear map (a) with respect to (A,B) is given by the matrix:");
disp(C);
```

```
"Matrix of Linear map (a) with respect to (A,B) is given by the matrix:"

  1.   0.  -1.
  0.  -1.   1.
 -1.   1.   0.
```

(c) (c) T(x,y) = (x+2y,y-2x)

```
clc;
clear;

function T = linear_c(v)
    T = [v(1) + 2*v(2); v(2) - 2*v(1)];
endfunction

v1 = input("Enter v1 of Basis of domain vector space: ");
v2 = input("Enter v2 of Basis of domain vector space: ");


w1 = input("Enter w1 of Basis of co-domain vector space: ");
w2 = input("Enter w2 of Basis of co-domain vector space: ");

Tv1 = linear_c(v1);
Tv2 = linear_c(v2);

B = [w1, w2]';

c1 = B \ Tv1;
c2 = B \ Tv2;
C = [c1, c2];

disp("Matrix of Linear map (c) with respect to (A,B) is given by the matrix:");
disp(C);
```

Scanned with OKEN Scanner

```
Enter v1 of Basis of domain vector space: [1;0]

Enter v2 of Basis of domain vector space: [0;1]

Enter w1 of Basis of co-domain vector space: [1;1]

Enter w2 of Basis of co-domain vector space: [1;-1]


  "Matrix of Linear map (c) with respect to (A,B) is given by the matrix:"

  -0.5   1.5
   1.5   0.5
```

(d) $T(x,y) = (-y,x)$

```
clc;
clear;
function T = linear_d(v)
    T = [-v(2); v(1)];
endfunction
v1 = input("Enter v1 of Basis of domain vector space: ");
v2 = input("Enter v2 of Basis of domain vector space: ");
w1 = input("Enter w1 of Basis of co-domain vector space: ");
w2 = input("Enter w2 of Basis of co-domain vector space: ");
Tv1 = linear_d(v1);
Tv2 = linear_d(v2);
B = [w1, w2]';
c1 = B \ Tv1;
c2 = B \ Tv2;
C = [c1, c2];
disp("Matrix of Linear map (d) with respect to (A,B) is given by the matrix:");
disp(C);
```

```
Enter v1 of Basis of domain vector space: [1;0]

Enter v2 of Basis of domain vector space: [0;1]

Enter w1 of Basis of co-domain vector space: [1;0]

Enter w2 of Basis of co-domain vector space: [0;1]


  "Matrix of Linear map (d) with respect to (A,B) is given by the matrix:"

   0.  -1.
   1.   0.
```

Scanned with OKEN Scanner

Experiment :- 5 Gram Schmidt Orthogonalization Process

Find an orthogonal set corresponding to the given sets

(a) B={(1,1),(1,0)}

```
clc;
clear;
n = input("Enter number of vectors: ");
for i = 1:n
    v(:,i) = input("Enter vector: ");
end

disp("Input vectors:");
disp(v);

w(:,1) = v(:,1);

for i = 2:n
    sum = 0;
    for j = 1:i-1
        sum = sum + ((w(:,j)' * v(:,i)) / (w(:,j)' * w(:,j))) * w(:,j);
    end
    w(:,i) = v(:,i) - sum;
end

disp("Orthogonal vectors:");
disp(w);
```

```
Enter number of vectors: 2

Enter vector: [1,1]

Enter vector: [1,0]


  "Input vectors:"

   1.   1.
   1.   0.

  "Orthogonal vectors:"

   1.   0.5
   1.  -0.5

-->
```

Scanned with OKEN Scanner

(b) {(1,2),(1,-1)}

```
clc;
clear;
n = input("Enter number of vectors: ");
for i = 1:n
    v(:,i) = input("Enter vector: ");
end

disp("Input vectors:");
disp(v);

w(:,1) = v(:,1);

for i = 2:n
    sum = 0;
    for j = 1:i-1
        sum = sum + ((w(:,j)' * v(:,i)) / (w(:,j)' * w(:,j))) * w(:,j);
    end
    w(:,i) = v(:,i) - sum;
end

disp("Orthogonal vectors:");
disp(w);
```

```
Enter number of vectors: 2

Enter vector: [1,2]

Enter vector: [1,-1]


  "Input vectors:"

   1.   1.
   2.  -1.

  "Orthogonal vectors:"

   1.   1.2
   2.  -0.6
```

(c) {(1,1,1),(1,0,1)}

```
clc;
clear;
n = input("Enter number of vectors: ");
for i = 1:n
    v(:,i) = input("Enter vector: ");
end

disp("Input vectors:");
disp(v);

w(:,1) = v(:,1);

for i = 2:n
    sum = 0;
    for j = 1:i-1
        sum = sum + ((w(:,j)' * v(:,i)) / (w(:,j)' * w(:,j))) * w(:,j);
    end
    w(:,i) = v(:,i) - sum;
end

disp("Orthogonal vectors:");
disp(w);
```

```
Enter number of vectors: 2

Enter vector: [1,1,1]

Enter vector: [1,0,1]


  "Input vectors:"

   1.   1.
   1.   0.
   1.   1.

  "Orthogonal vectors:"

   1.   0.3333333
   1.  -0.6666667
   1.   0.3333333
```

(d) {(1,1,2),(1,0,1),(1,3,2)}

Scanned with OKEN Scanner

```
clc;
clear;
n = input("Enter number of vectors: ");
for i = 1:n
    v(:,i) = input("Enter vector: ");
end

disp("Input vectors:");
disp(v);

w(:,1) = v(:,1);

for i = 2:n
    sum = 0;
    for j = 1:i-1
        sum = sum + ((w(:,j)' * v(:,i)) / (w(:,j)' * w(:,j))) * w(:,j);
    end
    w(:,i) = v(:,i) - sum;
end

disp("Orthogonal vectors:");
disp(w);
```

```
Enter number of vectors: 3

Enter vector: [1 1 2]

Enter vector: [1 0 1]

Enter vector: [1 3 2]


  "Input vectors:"

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

  "Orthogonal vectors:"

   1.   0.5   0.6666667
   1.  -0.5   0.6666667
   2.   0.   -0.6666667
```

(e) {(1,2,3),(2,0,3),(1,4,2)}

```
clc;
clear;
n = input("Enter number of vectors: ");
for i = 1:n
    v(:,i) = input("Enter vector: ");
end

disp("Input vectors:");
disp(v);

w(:,1) = v(:,1);

for i = 2:n
    sum = 0;
    for j = 1:i-1
        sum = sum + ((w(:,j)' * v(:,i)) / (w(:,j)' * w(:,j))) * w(:,j);
    end
    w(:,i) = v(:,i) - sum;
end

disp("Orthogonal vectors:");
disp(w);
```

```
Enter number of vectors: 3

Enter vector: [1 2 3]

Enter vector: [2 0 3]

Enter vector: [1 4 2]


  "Input vectors:"

   1.   2.   1.
   2.   0.   4.
   3.   3.   2.

  "Orthogonal vectors:"

   1.   1.2142857   0.9836066
   2.  -1.5714286   0.4918033
   3.   0.6428571  -0.6557377
```

(f) {(1,2,3,4),(2,3,4,5),(3,4,5,6)}

```
clc;
clear;
n = input("Enter number of vectors: ");
for i = 1:n
    v(:,i) = input("Enter vector: ");
end

disp("Input vectors:");
disp(v);

w(:,1) = v(:,1);

for i = 2:n
    sum = 0;
    for j = 1:i-1
        sum = sum + ((w(:,j)' * v(:,i)) / (w(:,j)' * w(:,j))) * w(:,j);
    end
    w(:,i) = v(:,i) - sum;
end

disp("Orthogonal vectors:");
disp(w);
```

```
Enter number of vectors: 3

Enter vector: [1 2 3 4]

Enter vector: [2 3 4 5]

Enter vector: [3 4 5 6]


  "Input vectors:"

   1.   2.   3.
   2.   3.   4.
   3.   4.   5.
   4.   5.   6.

  "Orthogonal vectors:"

   1.   0.6666667  -2.220D-15
   2.   0.3333333  -1.776D-15
   3.   0.          0.
   4.  -0.3333333   0.
```

Scanned with OKEN Scanner