Index

Experiment no. 1

Q1. Plot the vector in 2D: $\overrightarrow{A} = 3\hat{a}x + 2\hat{a}y$.

Ans:- $A = 3\hat{i} + 2\hat{j}$

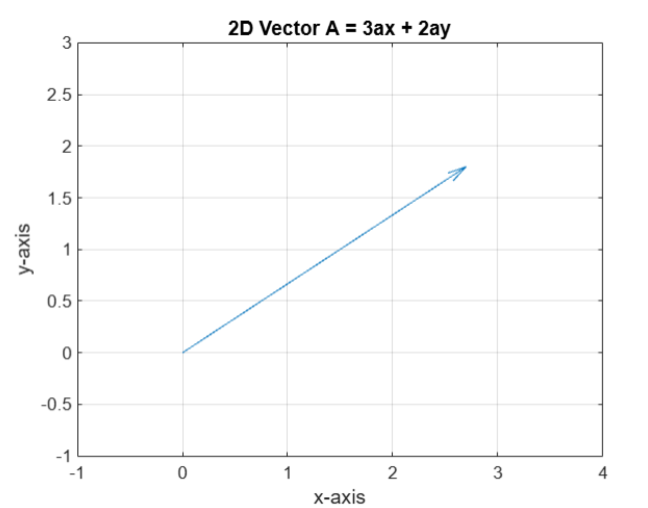
$|A| = \sqrt{3^2+2^2} = \sqrt{13}$

$\theta = \tan^{-1}\left(\frac{2}{3}\right)$ $\theta = 33.69°$

Code:

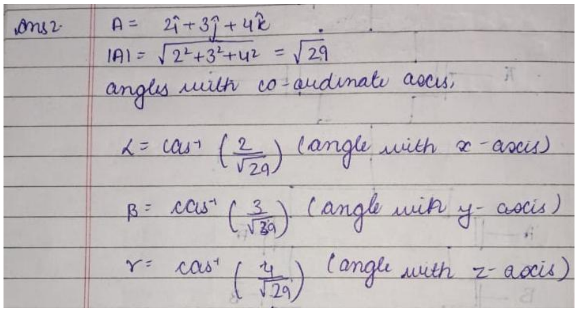
```
% Define the vector components
x_component = 3;
y_component = 2;
% Define the starting point of the vector (we'll start at the origin)
x_start = 0;
y_start = 0;
% Plot the vector
quiver (x_start, y_start, x_component, y_component);
% Set axis limits to visualize the vector better
xlim([-1 4]);
ylim([-1 3]);
% Add labels and title
xlabel('x-axis');
ylabel('y-axis');
title ('2D Vector A = 3ax + 2ay');
% Grid on for better visualization
grid on;
```

Output:



Q2: To find the magnitude and angles and plot vector in 3D: $\vec{A} = 2\hat{a}x + 3\hat{a}y + 4\hat{a}z$.

Code:



ans2 $A = 2\hat{i} + 3\hat{j} + 4\hat{k}$

$|A| = \sqrt{2^2+3^2+4^2} = \sqrt{29}$

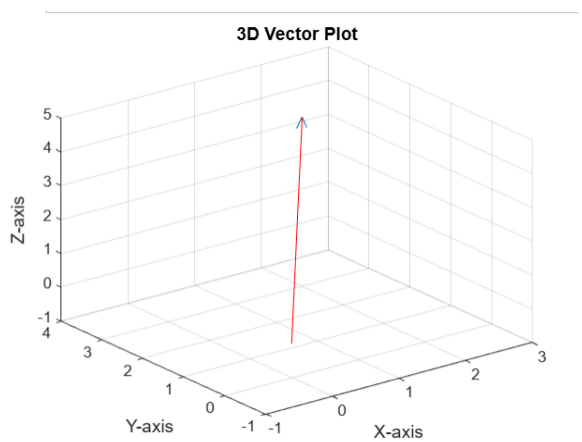
angles with co-ordinate axis;

$\alpha = \cos^{-1}\left(\frac{2}{\sqrt{29}}\right)$ (angle with x-axis)

$\beta = \cos^{-1}\left(\frac{3}{\sqrt{29}}\right)$ (angle with y-axis)

$\gamma = \cos^{-1}\left(\frac{4}{\sqrt{29}}\right)$ (angle with z-axis)

```
% Define the components of the vector
Ax = 2; % x-component
Ay = 3; % y-component
Az = 4; % z-component
% Calculate the magnitude of the vector
magnitude = sqrt (Ax^2 + Ay^2 + Az^2);
% Calculate the angles with respect to the axes
theta_x = acosd(Ax / magnitude); % Angle with x-axis
theta_y = acosd(Ay / magnitude); % Angle with y-axis
theta_z = acosd(Az / magnitude); % Angle with z-axis
% Display the results
fprintf('Magnitude of the vector: %.2f\n', magnitude);
fprintf('Angle with x-axis: %.2f degrees\n', theta_x);
fprintf('Angle with y-axis: %.2f degrees\n', theta_y);
fprintf('Angle with z-axis: %.2f degrees\n', theta_z);
% Plot the vector in 3D
figure; % Create a new figure window
quiver3(0, 0, 0, Ax, ay, Az, 'AutoScale', 'off'); % Plot the vector
hold on; % Hold the plot for additional plotting
plot3([0, Ax], [0, Ay], [0, Az], 'r'); % Plot the vector as a line for better visualization
hold off; % Release the plot hold
% Set axis limits for better visualization
axis ([-1 3 -1 4 -1 5]);
grid on; % Turn on the grid
xlabel('X-axis');
ylabel('Y-axis');
zlabel('Z-axis');
title ('3D Vector Plot');
```



Output:

```
Magnitude of the vector: 5.39
Angle with x-axis: 68.20 degrees
Angle with y-axis: 56.15 degrees
Angle with z-axis: 42.03 degrees
```

Q3: Find Dot Product and Cross Product of the given two vectors also find the angle between them. A=2ax+3ay+4az & B=ax+2ay+4az

Code:

Ans3- $A = 2\hat{i} + 3\hat{j} + 4\hat{k}$

$B = \hat{i} + 2\hat{j} + 4k$

dot product $(A \cdot B) = 2 + 6 + 16 = 24$

Crossproduct $(A \times B)$

$$\begin{vmatrix} i & j & k \\ 2 & 3 & 4 \\ 1 & 2 & 4 \end{vmatrix} = \hat{i}(12-8) - \hat{j}\,(8-4) + \hat{k}\,(4-3) = 4\hat{i} - 4\hat{j} + k$$

Angle $\cos\theta = \frac{A \cdot B}{|A||B|}$ $\quad |A| = \sqrt{29}$, $|B| = \sqrt{21}$

$$\cos\theta = \frac{24}{\sqrt{29}\sqrt{21}}$$

```
% Define the components of the vectors
A = [2, 3, 4];
B = [1, 2, 4];
% Calculate the dot product
dot_product = dot (A, B);
% Calculate the cross product
cross_product = cross (A, B);
% Calculate the magnitudes of the vectors
magnitude_A = norm(A);
magnitude_B = norm(B);
% Calculate the angle between the vectors in degrees
angle = acosd(dot_product / (magnitude_A * magnitude_B));
% Display the results
disp(['Dot Product: ', num2str(dot_product)]);
disp(['Cross Product: [', num2str(cross_product), ']']);
disp(['Angle between vectors: ', num2str(angle), ' degrees']);
```

Output:

```
>> gradient_verification
Dot Product: 24
Cross Product: [4 -4  1]
Angle between vectors: 13.4609 degrees
```

Q4: Find the sine angle between two vectors. A= 2ax+3ay+4az & B= ax+2ay+4az.

Code:

Ans4 using cross product

$\sin\theta = \dfrac{|A \times B|}{|A||B|}$ $\quad |A \times B| = \sqrt{4^2+(-4)^2+1^2} = \sqrt{33}$

$\sin\theta = \dfrac{\sqrt{33}}{\sqrt{29}\sqrt{21}}$

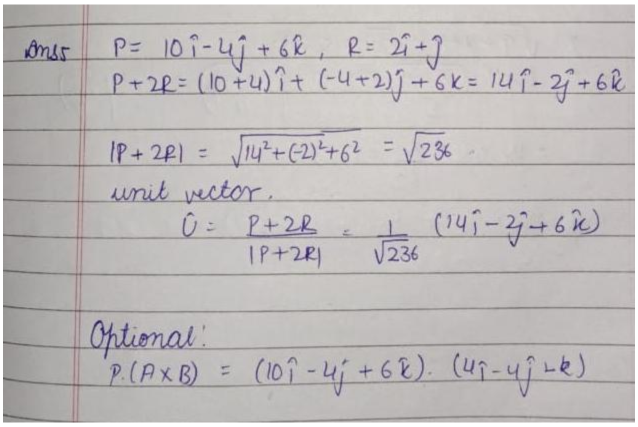
```
% Define the components of the vectors
A = [2, 3, 4];
B = [1, 2, 4];
% Calculate the dot product
dot_product = dot (A, B);
% Calculate the magnitudes of the vectors
magnitude_A = norm(A);
magnitude_B = norm(B);
% Calculate the angle between the vectors
in radians
angle_rad = acos(dot_product /
(magnitude_A * magnitude_B));
% Calculate the sine of the angle
sine_angle = sin(angle_rad);
% OR sine_angle = sqrt (1 - (dot_product /
(magnitude_A * magnitude_B)) ^2)
% Display the result
disp(['Sine of the angle between vectors: ',
num2str(sine_angle)]);
```

Output:

```
Sine of the angle between vectors: 0.23278
```

Q5: Find the unit vector along P+2R and also find the magnitude and angle between the unit vector and the coordinates axes with P=10ax-4ay+6az & R= 2ax+ay.

Code:



Ans $P = 10\hat{i} - 4\hat{j} + 6\hat{k}$, $R = 2\hat{i} + \hat{j}$

$P+2R = (10+4)\hat{i} + (-4+2)\hat{j} + 6k = 14\hat{i} - 2\hat{j} + 6\hat{k}$

$|P+2R| = \sqrt{14^2+(-2)^2+6^2} = \sqrt{236}$

unit vector.

$$\hat{U} = \frac{P+2R}{|P+2R|} = \frac{1}{\sqrt{236}}(14\hat{i} - 2\hat{j} + 6\hat{k})$$

Optional:

$P.(A\times B) = (10\hat{i} - 4\hat{j} + 6\hat{k}).(4\hat{i} - 4\hat{j} - k)$

```
% Define the components of the vectors P and R
P = [10, -4, 6];
R = [2, 1, 0];
% Calculate the vector P + 2R
resultant_vector = P + 2 * R;
% Calculate the magnitude of the resultant vector
magnitude_resultant = norm(resultant_vector);
% Calculate the unit vector along P + 2R
unit_vector = resultant_vector / magnitude_resultant;
% Calculate the angles with respect to the coordinate axes
theta_x = acosd(unit vector(1));
theta_y = acosd (unit_vector (2));
theta_z = acos0d(unit_vector (3));
% Display the results
disp(['Unit vector along P + 2R: [', num2str(unit_vector), ']']);
disp(['Magnitude of the resultant vector: ', num2str(magnitude_resultant)]);
disp(['Angle with x-axis: ', num2str(theta_x), ' degrees']);
disp(['Angle with y-axis: ', num2str(theta_y), ' degrees']);
disp(['Angle with z-axis: ', num2str(theta_z), ' degrees']);
```

Output:

```
Unit vector along P + 2R: [0.91132    -0.13019     0.39057]
Magnitude of the resultant vector: 15.3623
Angle with x-axis: 24.3113 degrees
Angle with y-axis: 97.4805 degrees
Angle with z-axis: 67.0102 degrees
```

Q6: Find the scalar triplet product: P. (AxB) & vector triple product: Px(AxB).
Where P=10ax-4ay+6az & A= 2ax+3ay+4az & B= ax+2ay+4az.

Code:

```
% Define the vectors
P = [10 -4 6];
A = [2 3 4];
B = [1 2 4];
% Calculate the scalar triple product
scalar_triple_product = dot (P, cross (A, B));
% Calculate the vector triple product
vector_triple_product = cross (P, cross (A, B));
% Display the results
disp(['Scalar triple product: ', num2str(scalar_triple_product)]);
disp(['Vector triple product: ', num2str(vector_triple_product)]);
```

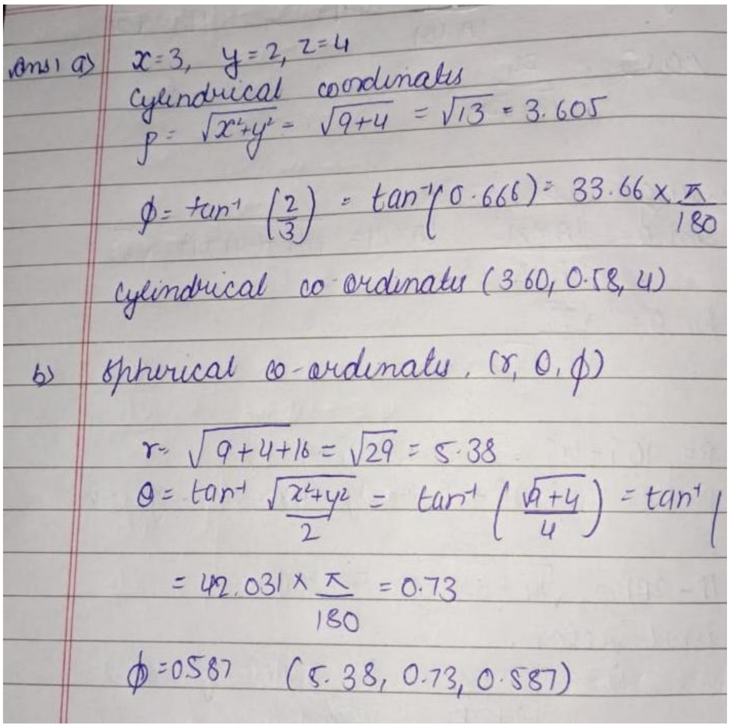
Output:

```
Scalar triple product: 62
Vector triple product: 20  14 -24
```

Experiment no. 2

Que1: Transform point (3,2,4) from cartesian to: i) Cylindrical ii) Spherical coordinate (r,θ,Φ)
Using point transformation and also write the Matlab program for these point to point Transformation. Do similar exercise for (ρ, Φ,z)⍰(x,y,z) and (r, θ, Φ)⍰(x,y,z) and (ρ, Φ,z)⍰(r, θ,Φ) And (r, θ,Φ)⍰(ρ, Φ,z).

Code:



Ans) a) $x=3,\ y=2,\ z=4$

Cylindrical coordinates

$\rho = \sqrt{x^2+y^2} = \sqrt{9+4} = \sqrt{13} = 3.605$

$\phi = \tan^{-1}\left(\frac{2}{3}\right) = \tan^{-1}(0.666) = 33.66 \times \frac{\pi}{180}$

Cylindrical co ordinates (3.60, 0.58, 4)

b) Spherical co-ordinates, (r, θ, ϕ)

$r = \sqrt{9+4+16} = \sqrt{29} = 5.38$

$\theta = \tan^{-1} \frac{\sqrt{x^2+y^2}}{z} = \tan^{-1}\left(\frac{\sqrt{9+4}}{4}\right) = \tan^{-1}($

$= 42.031 \times \frac{\pi}{180} = 0.73$

$\phi = 0.587$ (5.38, 0.73, 0.587)

```
% Define the Cartesian coordinates
x = 3;
y = 2;
z = 4;
% Cartesian to Cylindrical coordinates
rho = sqrt (x^2 + y^2);
phi = atan2(y, x);
z_cyl = z;
% Cartesian to Spherical coordinates
r = sqrt (x^2 + y^2 + z^2);
theta = acos(z / r);
phi_sph = atan2(y, x);
% Display the results
disp(['Cylindrical coordinates (ρ, Φ, z): (', num2str(rho), ', ', num2str(phi), ', ', num2str(z_cyl), ')']);
disp(['Spherical coordinates (r, θ, Φ): (', num2str(r), ', ', num2str(theta), ', ', num2str(phi_sph), ')']);
```

Output:

```
Cylindrical coordinates (ρ, Φ, z): (3.6056, 0.588, 4)
Spherical coordinates (r, θ, Φ): (5.3852, 0.73358, 0.588)
```

Code:

```
% Define the Cylindrical coordinates
rho = 3;
phi = pi/4; % Example value
z_cyl = 4;
% Cylindrical to Cartesian coordinates
x_cyl = rho * cos(phi);
y_cyl = rho * sin(phi);
z_cart = z_cyl;
% Display the results
disp(['Cartesian coordinates (x, y, z): (', num2str(x_cyl), ', ', num2str(y_cyl), ', ', num2str(z_cart), ')']);
```

Output:

```
Cartesian coordinates (x, y, z): (2.1213, 2.1213, 4)
```

Code:

```
% Define the Spherical coordinates
r = 5;
theta = pi/3; % Example value
phi_sph = pi/4; % Example value
% Spherical to Cartesian coordinates
x_sph = r * sin(theta) * cos(phi_sph);
y_sph = r * sin(theta) * sin(phi_sph);
z_sph = r * cos(theta);
% Display the results
disp(['Cartesian coordinates (x, y, z): (', num2str(x_sph), ', ', num2str(y_sph), ', ', num2str(z_sph), ')']);
```

Output:

```
Cartesian coordinates (x, y, z): (3.0619, 3.0619, 2.5)
```

Code:

```
% Define the Cylindrical coordinates
rho = 3;
phi = pi/4; % Example value
z_cyl = 4;
% Cylindrical to Spherical coordinates
r_sph = sqrt (rho^2 + z_cyl^2);
theta_sph = atan2(rho, z_cyl);
phi_sph = phi;
% Display the results
disp(['Spherical coordinates (r, θ, Φ): (', num2str(r_sph), ', ', num2str(theta_sph), ', ',
num2str(phi_sph), ')']);
```

Output:

```
Spherical coordinates (r, θ, Φ): (5, 0.6435, 0.7854)
```

Code:

```
% Define the Spherical coordinates
r = 5;
theta = pi/3; % Example value
phi_sph = pi/4; % Example value
% Spherical to Cylindrical coordinates
rho_cyl = r * sin(theta);
phi_cyl = phi_sph;
z_cyl = r * cos(theta);
% Display the results
disp(['Cylindrical coordinates (ρ, Φ, z): (', num2str(rho_cyl), ', ', num2str(phi_cyl), ', ',
num2str(z_cyl), ')']);
```

Output:

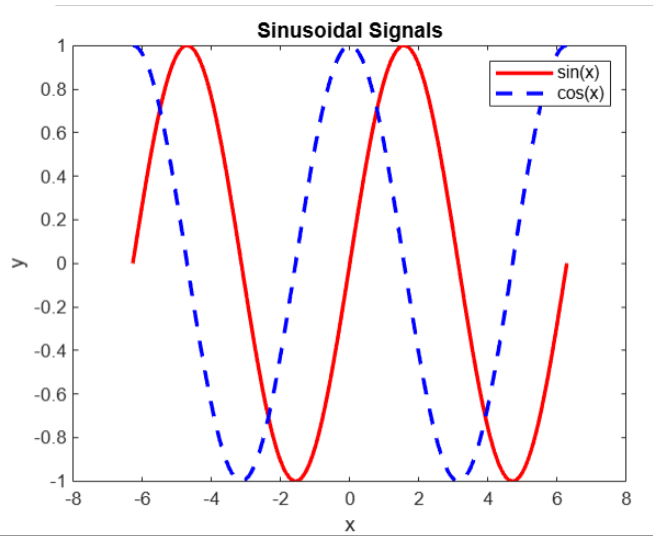
```
Cylindrical coordinates (ρ, Φ, z): (4.3301, 0.7854, 2.5)
```

Q2: Write a matlab program to plot sinx and cosx on a single figure window using HOLD command. Let sinx be red coloured and solid, and cosx be blue and dotted line pattern. Also assign a title (Title) (sinusoidal signal to the graph. Use linespace, legend, and gtext and set.

Code:

```
% Generate x values from -2*pi to 2*pi with 1000 points
x = linspace(-2*pi, 2*pi, 1000);
% Calculate sin(x) and cos(x)
y1 = sin(x);
y2 = cos(x);
% Plot sin(x) in red and solid line
plot (x, y1, 'r-', 'LineWidth', 2);
hold on; % Hold the plot
% Plot cos(x) in blue and dotted line
plot (x, y2, 'b--', 'LineWidth', 2);
% Add title and labels
title ('Sinusoidal Signals');
xlabel('x');
ylabel('y');
% Add legend
legend('sin(x)', 'cos(x)');
% Add text annotation
gtext('This is a sinusoidal signal plot');
% Set figure properties
set (gcf, 'Color', 'w'); % Set figure background color to white
set (gca, 'FontSize', 12); % Set font size for axes labels and title
```

Output:



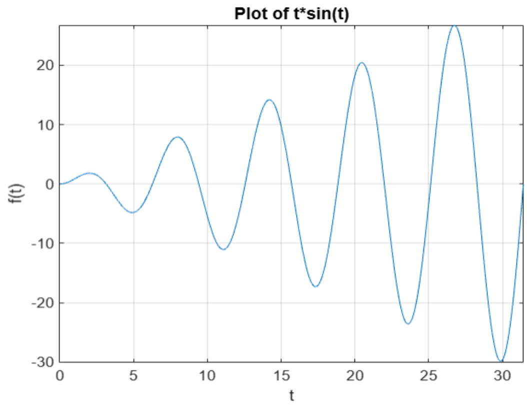
Experiment no. 3

Q1: Plot f(t)=tsin(t), 0<=t<=10pi using FPLOT command.

Code:

```
% Define the function
f = @(t) t.*sin(t);
% Plot the function using fplot
fplot(f, [0, 10*pi]);
% Add labels and title
xlabel('t');
ylabel('f(t)');
title ('Plot of t*sin(t)');
% Set grid on
grid on;
```

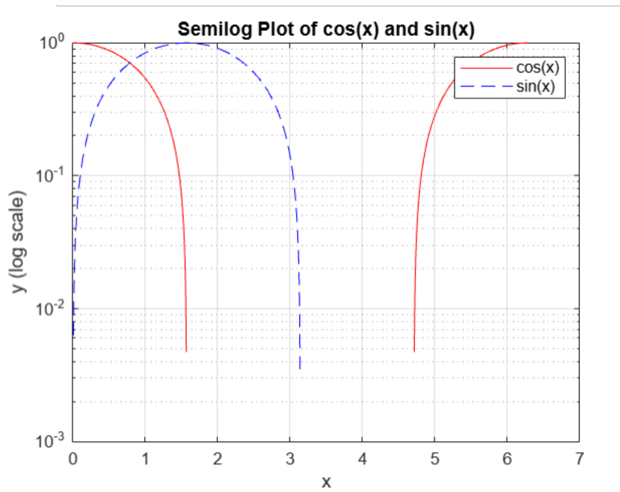
Output:



Q2: Plot cosx and sinx on a semilog scale.

Code:

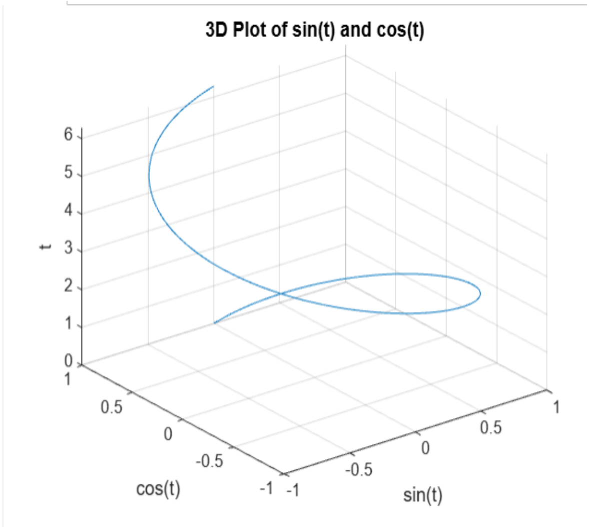
```
% Generate x values
x = linspace(0, 2*pi, 1000);
% Calculate cos(x) and sin(x)
y1 = cos(x);
y2 = sin(x);
% Create a semilog plot
semilogy(x, y1, 'r-', x, y2, 'b--');
% Add labels and title
xlabel('x');
ylabel('y (log scale)');
title('Semilog Plot of cos(x) and sin(x)');
% Add legend
legend('cos(x)', 'sin(x)');
% Set grid on
grid on;
```



Que3: Plot sin(t), cos(t) with t= 0 to2pi using plot 3 command.

Code:

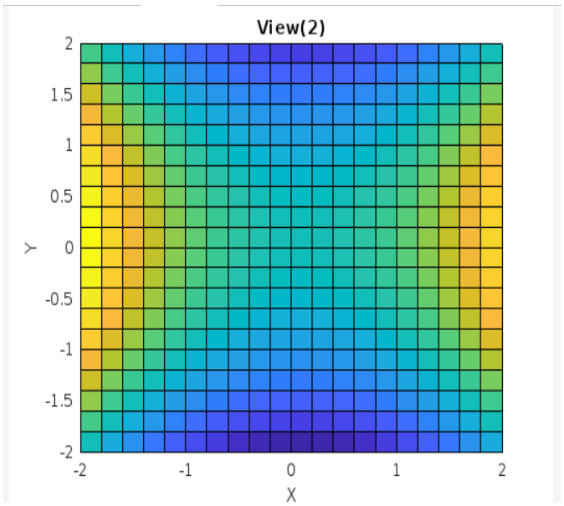
```
% Generate t values from 0 to 2*pi
t = linspace(0, 2*pi, 100);
% Calculate sin(t) and cos(t)
x = sin(t);
y = cos(t);
z = t;
% Plot the 3D curve
plot3(x, y, z);
% Add labels and title
xlabel('sin(t)');
ylabel('cos(t)');
zlabel('t');
title ('3D Plot of sin(t) and cos(t)');
% Set grid on
grid on;
```



Que4: Plot a 3D function and use VIEW (2) command and title them 3D view and view (2) respectively.

Code:

```
% Create a 3D function
[X, Y] = meshgrid(-2:0.2:2);
Z = X.^2 - Y.^2;
% Plot the 3D function
figure;
surf (X, Y, Z);
title ('3D View');
xlabel('X');
ylabel('Y');
zlabel('Z');
% View the plot from a 2D perspective
figure;
surf (X, Y, Z);
view (2);
title ('View (2)');
xlabel('X');
ylabel('Y');
zlabel('Z');
```

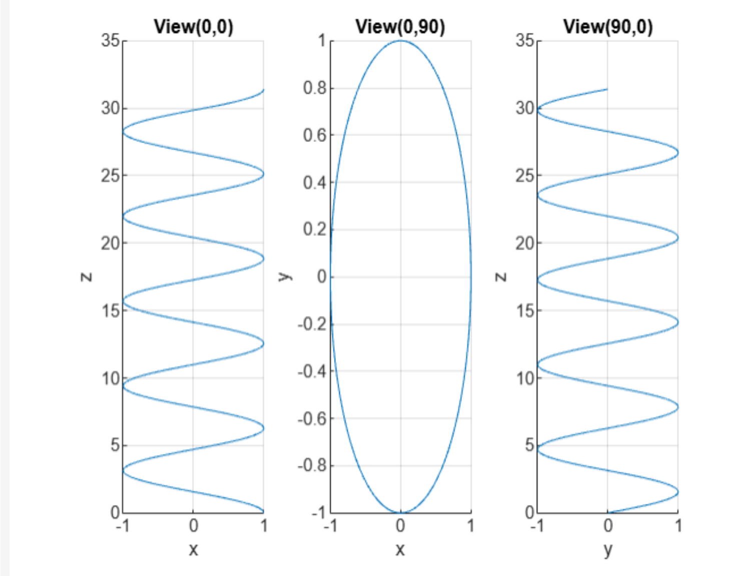


Que5: Write a Matlab Program to plot a function using Plot 3 command and view command. For plot 3 (x,y,z), x=cos(t), y=sin(t) and z=t and for view(0,0),(0,9,0) and (9,0,0) and title them accordingly.

Code:

```
% Generate t values
t = linspace(0, 10*pi, 1000);
% Calculate x, y, and z values
x = cos(t);
y = sin(t);
z = t;
% Plot the 3D curve
figure;
plot3(x, y, z);
title ('3D Plot of the Curve');
xlabel('x');
ylabel('y');
zlabel('z');
grid on;
% View the plot from different angles
figure;
subplot (1, 3, 1);
plot3(x, y, z);
view (0, 0);
title ('View (0,0)');
xlabel('x');
ylabel('y');
zlabel('z');
grid on;
subplot (1, 3, 2);
plot3(x, y, z);
view (0, 90);
title ('View (0,90)');
xlabel('x');
ylabel('y');
zlabel('z');
grid on;
subplot (1, 3, 3);
plot3(x, y, z);
view (90, 0);
title('View (90,0)');
xlabel('x');
ylabel('y');
zlabel('z');
grid on;
```

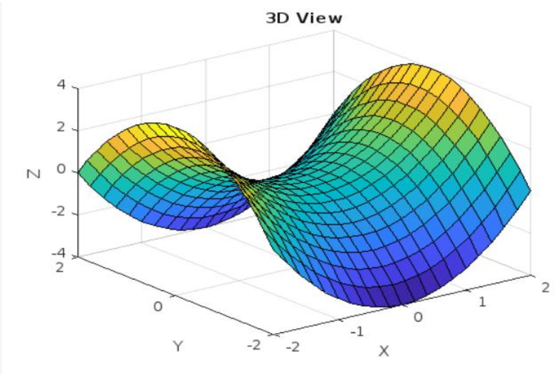
Output:



Que6: Write a matlab Program to plot a 3D function using view (3) command.

Code: Output:

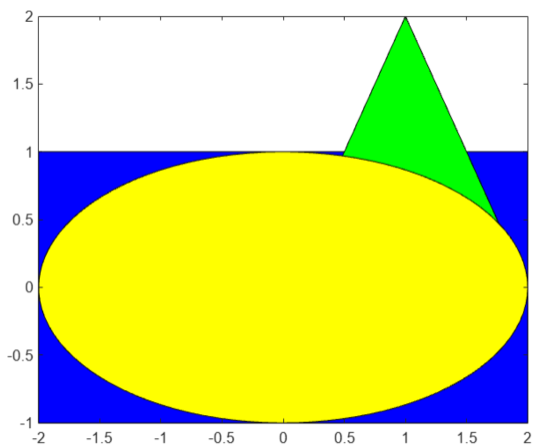
```
% Create a 3D function
[X, Y] = mesh grid (-2:0.2:2);
Z = X.^2 - Y.^2;
% Plot the 3D function
figure;
surf (X, Y, Z);
title ('3D View');
xlabel('X');
ylabel('Y');
zlabel('Z');
view (3);
```



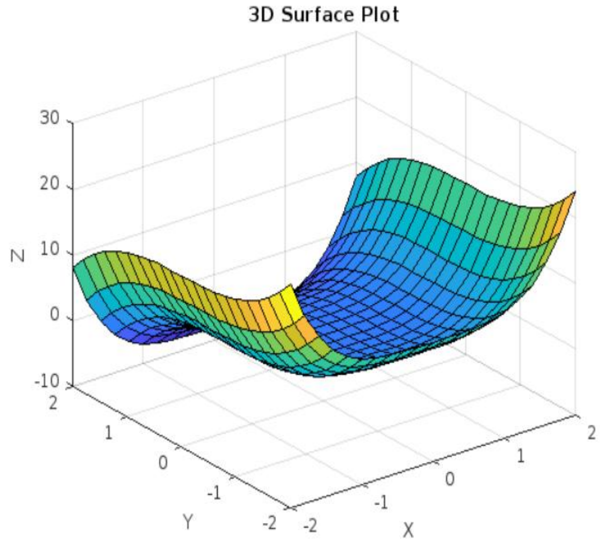
Q7: Write a matlab program to color different shapes and pattern on a 2D and 3D image using fill, meshgrid, contour, surf, command to draw different 3D shapes and 3D images.

Code: Output:

```
% Create a figure 2D figure;
% Plot a filled circle
t = linspace(0, 2*pi, 100);
x = cos(t);
y = sin(t);
fill (x, y, 'r');
hold on;
% Plot a filled rectangle
x = [-2 2 2 -2];
y = [-1 -1 1 1];
fill (x, y, 'b');
% Plot a filled triangle
x = [0 2 1];
y = [0 0 2];
fill (x, y, 'g');
% Plot a filled ellipse
a = 2;
b = 1;
t = linspace(0, 2*pi, 100);
x = a*cos(t);
y = b*sin(t);
fill (x, y, 'y');
hold off;
% Create a 3D meshgrid
[X, Y] = meshgrid(-2:0.2:2);
% Create a 3D function
Z = X.^4 - Y.^3;
```



```
% Plot a 3D surface
figure;
surf (X, Y, Z);
title ('3D Surface Plot');
xlabel('X');
ylabel('Y');
zlabel('Z');
% Plot a 3D contour plot
figure;
contour (X, Y, Z);
title ('3D Contour Plot');
xlabel('X');
ylabel('Y');
% Plot a 3D mesh plot
figure;
mesh (X, Y, Z);
title ('3D Mesh Plot');
xlabel('X');
ylabel('Y');
zlabel('Z');
```

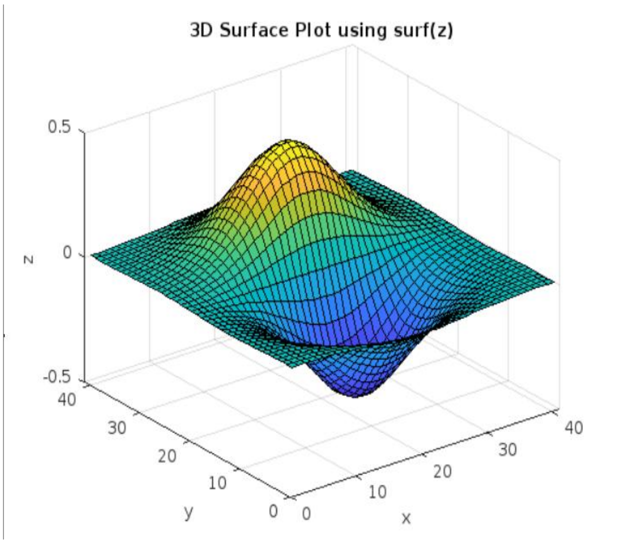


Q8: Write a matlab program to plot the 3D image for y=x exp (-x^2 – y^2) using meshgrid and surf (x,y,z) and surf(z) command.

Code:

```
% Create a 3D meshgrid
[x,y] = meshgrid(-2:0.1:2);
% Calculate the function values
z = y.* exp (-x.^2 - y.^2);
% Plot the 3D surface using surf(x,y,z)
figure;
surf(x,y,z);
title ('3D Surface Plot using surf(x,y,z)');
xlabel('x');
ylabel('y');
zlabel('z');
% Plot the 3D surface using surf(z)
figure;
surf(z);
title ('3D Surface Plot using surf(z)');
xlabel('x');
ylabel('y');
zlabel('z');
```

Output:



Experiment no. 4

Q1: Write a matlab program to find the gradient of the following scalar field. And verify analytically also.

a) T=x exp(-x^2-y^2) at (x,y)=(1,2)
b) V=-e^(-z) Sin2xCoshy
c) U= ρ^ (2) zcos2 Φ
d) T1=x^ (2)y +xyz
e) V1= ρzsin Φ + x^ (2) cos^ (2) Φ + ρ^2

Code:

Ans) - a) $T = x e^{-x^2-y^2}$

$$\frac{\partial T}{\partial x} = e^{-x^2-y^2} - 2x^2 e^{-x^2-y^2}$$

$$\frac{\partial T}{\partial y} = -2xy e^{-x^2-y^2} \quad \text{at } (x,y) = (1,2)$$

$$\frac{\partial T}{\partial x} = -e^{-5} \qquad \frac{\partial T}{\partial y} = -4e^{-5}$$

$$\nabla T = -e^{-5} a_x - 4e^{-5} a_y$$

b) $V = -e^{-z} \sin(2x) \cos y$

$$\frac{\partial V}{\partial x} = -e^{-z} \cos(2x) \cos y \qquad \frac{\partial V}{\partial y} = -e^{-z} \sin(2x) \sin y$$

$$\frac{\partial V}{\partial z} = e^{-z} \sin(2x) \cos y$$

$$\nabla V = -2e^{-z} \cos(2x) \cdot \cos y \, a_x$$
$$= -e^{-z} \sin(2x) \sin y \, a_y + e^{-z} \sin(2x) \cos y$$

d) $U = \rho^2 \cdot z \cos(2\phi)$ $\quad \nabla U = \left(\frac{\partial U}{\partial \rho}, \frac{1}{\rho} \frac{\partial U}{\partial \phi}, \frac{\partial U}{\partial z} \right)$

$$\frac{\partial U}{\partial \rho} = 2\rho z \cos(2\phi) \qquad \frac{\partial U}{\partial z} = \rho^2 \cos(2\phi)$$

$$\frac{\partial U}{\partial \phi} = -2\rho^2 z \sin(2\phi)$$

$$\nabla U = 2\rho z \cos(2\phi) a_\rho - 2\rho z \sin(2\phi) a_\phi + \rho^2 \cos(2\phi) a_\phi$$

d) $T_1 = x^2y + xy^2$

$\frac{\partial T_1}{\partial x} = 2xy + y^2$

$\frac{\partial T_1}{y} = x^2 + x^2$

$\frac{\partial T_1}{\partial z} = xy$

$\nabla T_1 = (2xy + yz)a_x + (x^2 + xz)a_y + xy\,a_z$

e) $V_1 = \rho z \sin\phi + x^2 \cos^2\phi + \rho^2$

$\frac{\partial V_1}{\partial \rho} = z\sin\phi + 2\rho$

$\frac{\partial V_1}{\partial \phi} = \rho z \cos\phi - 2x^2 \cos\phi \sin\phi$

$\frac{\partial V_1}{\partial z} = \rho \sin\phi$

$\nabla V_1 = (z\sin\phi + 2\rho)a_\rho + (\rho z \cos\phi - 2x^2 \cos\phi \sin\phi)a_\phi + \rho \sin\phi\, a_z$

```
% a) T = x*exp(-x^2-y^2) at (1,2)
syms x y;
T = x*exp(-x^2-y^2);
grad_T = gradient(T);
grad_T_numerical = gradient (subs (T, [x,y], [1,2]));
% b) V = -e^(-z) *sin(2*x) *cosh(y)
syms x y z;
V = -exp(-z) *sin(2*x) *cosh(y);
grad_V = gradient(V);
% c) U = rho^2*z*cos(2*phi)
syms rho phi z;
U = rho^2*z*cos(2*phi);
grad_U = gradient (U, [rho, phi, z]);
% d) T1 = x^2*y + x*y*z
syms x y z;
T1 = x^2*y + x*y*z;
grad_T1 = gradient(T1);
% e) V1 = rho*z*sin(phi) + x^2*cos^2(phi) + rho^2
syms rho phi z;
V1 = rho*z*sin(phi) + x^2*cos^2(phi) + rho^2;
grad_V1 = gradient (V1, [rho, phi, z]);
% Display the results
disp('Gradient of T at (1,2):');
disp(grad_T_numerical)
disp('Gradient of V:');
disp(grad_V);
disp('Gradient of U:');
disp(grad_U);
disp('Gradient of T1:');
disp(grad_T1);
disp('Gradient of V1:');
disp(grad_V1);
```

Output:

```
Gradient of T at (1,2):
Gradient of V:
-2*cos(2*x)*exp(-z)*cosh(y)
  -sin(2*x)*exp(-z)*sinh(y)
   sin(2*x)*exp(-z)*cosh(y)

Gradient of U:
   2*rho*z*cos(2*phi)
-2*rho^2*z*sin(2*phi)
     rho^2*cos(2*phi)

Gradient of T1:
2*x*y + y*z
  x^2 + z*x
        x*y

Gradient of V1:
                             2*rho + z*sin(phi)
- 2*cos(phi)*sin(phi)*x^2 + rho*z*cos(phi)
                                  rho*sin(phi)
```

Q2: Given W=x^2 + y^2 + xyz , compute $\vec{\nabla} W$ and the directional derivative dW/dl in the direction 3ax + 4ay + 12az at (2,-1,0). Write a matlab program for finding $\vec{\nabla} W$ and directional directive.

Code:

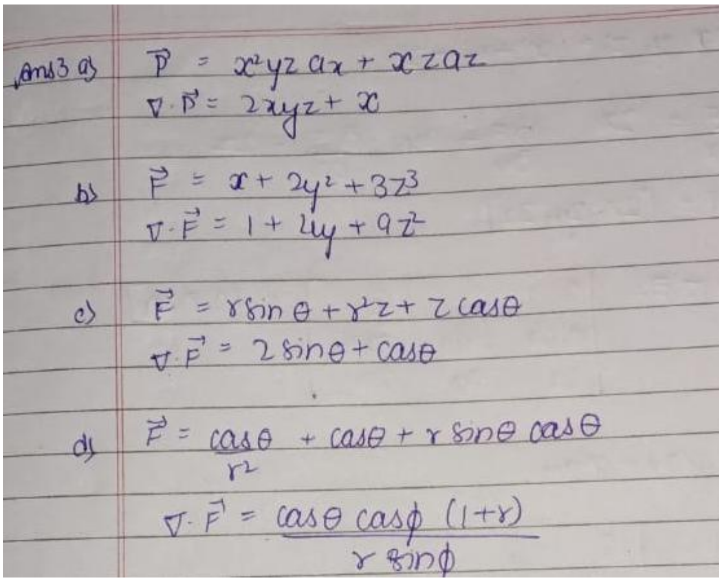
```
% Define the scalar field W
syms x y z;
W = x^2 + y^2 + x*y*z;
% Calculate the gradient of W
grad_W = gradient(W);
% Evaluate the gradient at the point (2, -1,0)
point = [2 -1 0];
grad_W_at_point = subs (grad_W, [x, y, z], point);
% Define the direction vector l
l = [3 4 12];
% Normalize the direction vector
l_normalized = l / norm(l);
% Calculate the directional derivative
directional_derivative = dot (grad_W_at_point, l_normalized);
% Display the results
disp('Gradient of W at (2,-1,0):');
disp(grad_W_at_point);
disp('Directional derivative of W at (2,-1,0) in the direction of l:');
disp(directional_derivative);
```

Output:

```
Gradient of W at (2,-1,0):
 4
-2
-2

Directional derivative of W at (2,-1,0) in the direction of l:
-20/13
```

Q3: Determine the divergence of the vector fields.
(a)$\boldsymbol{P} = \boldsymbol{yz}\ \boldsymbol{ax} + \boldsymbol{xz}\ \boldsymbol{az}$
(b)$\boldsymbol{R} = \boldsymbol{x}\ \boldsymbol{ax} + \boldsymbol{2y^2}\ \boldsymbol{ay} + \boldsymbol{3z^3}$ az
(c)$\boldsymbol{Q} =$ ρ$\boldsymbol{sin\phi}\ \boldsymbol{a}$ρ $+ \boldsymbol{rho^2z}\ \boldsymbol{a\phi} + \boldsymbol{zcos\phi}\ \boldsymbol{az}$
(d)$\boldsymbol{T} = \boldsymbol{1/r^2cos}\ \boldsymbol{theata}\ \boldsymbol{ar} + \boldsymbol{rsin}\ \boldsymbol{theata}\ \boldsymbol{cos\phi az} + \boldsymbol{cos}\ \boldsymbol{theata}\ \boldsymbol{a\phi}$
Write the matlab program for same.

Code:


ans3 a) $\vec{P} = x^2yz\,a_x + xz\,a_z$
$\nabla\cdot\vec{P} = 2xyz + x$
b) $\vec{F} = x + 2y^2 + 3z^3$
$\nabla\cdot\vec{F} = 1 + 4y + 9z^2$
c) $\vec{F} = r\sin\theta + r^2z + z\cos\theta$
$\nabla\cdot\vec{F} = 2\sin\theta + \cos\theta$
d) $\vec{F} = \frac{\cos\theta}{r^2} + \cos\theta + r\sin\theta\cos\theta$
$\nabla\cdot\vec{F} = \frac{\cos\theta\cos\phi\,(1+r)}{r\sin\phi}$

```
%3_a
function div = compute_divergence(Fx, Fy, Fz)
   % Fx, Fy, Fz: Components of the vector field
   % Calculate partial derivatives
   syms x y z;
   div_x = diff (Fx, x);
   div_y = diff (Fy, y);
   div_z = diff (Fz, z);
   % Compute the divergence
   div = div_x + div_y + div_z;
end
% Example vector field components
Fx = ((x^2) *y*z);
Fy = 0;
Fz = x*z;
% Calculate the divergence
div_result = compute_divergence(Fx, Fy, Fz);
% Display the result
disp('Divergence of the vector field:');
disp(div_result);
%3_b
% Example vector field components
Fx = (x);
Fy = (2*y^2);
Fz = (3*z^3);
% Calculate the divergence
```

```
div_result = compute_divergence(Fx, Fy, Fz);
% Display the result
disp('Divergence of the vector field:');
disp(div_result);
%3_c
function div_cylindrical = compute_divergence_cylindrical(F_r, F_theta, F_z, r)
   % F_r, F_theta, F_z: Components of the vector field
   % r: Radial distance
   % Calculate partial derivatives
   syms r theta z;
   div_r = (1/r) * diff (r*F_r, r);
   div_theta = (1/r) * diff (F_theta, theta);
   div_z = diff (F_z, z);
   % Compute the total divergence
   div_cylindrical = div_r + div_theta + div_z;
end
% Example vector field components (replace with your own functions)
 syms r theta z;
F_r = (r*sin(theta));
F_theta = ((r^2) *z);
F_z = (z*cos(theta));
% Parameters
r = sym('r');
theta = sym('theta');
z = sym('z');
% Calculate the divergence in cylindrical coordinates
div_result = compute_divergence_cylindrical(F_r, F_theta, F_z, r);
% Display the result
disp('Divergence in cylindrical coordinates:');
disp(div_result);
%3_d
function div_spherical = compute_divergence_spherical(F_r, F_phi, F_theta, r, phi)
   % F_r, F_phi, F_theta: Components of the vector field
   % r: Radial distance
   % phi: Polar angle (in radians)
   % Calculate partial derivatives
   syms r phi theta;
   div_r = (1/r^2) * diff (r^2*F_r, r);
   div_phi = (1/(r*sin(phi))) * diff(F_phi*sin(phi), phi);
   div_theta = (1/(r*sin(phi))) * diff (F_theta, theta);
   % Compute the total divergence
   div_spherical = div_r + div_phi + div_theta;
end
 syms r phi theta;
% Example vector field components (replace with your own functions)
F_r = (1/(r^2) *cos(theta));
F_phi = cos(theta);
F_theta = (r*sin(theta)*cos(phi));
% Parameters
r = sym('r');
phi = sym('phi');
```

```
theta = sym('theta');
% Calculate the divergence in spherical coordinates
div_result = compute_divergence_spherical(F_r, F_phi, F_theta, r, phi);
% Display the result
disp('Divergence in spherical coordinates:');
disp(div_result);
```

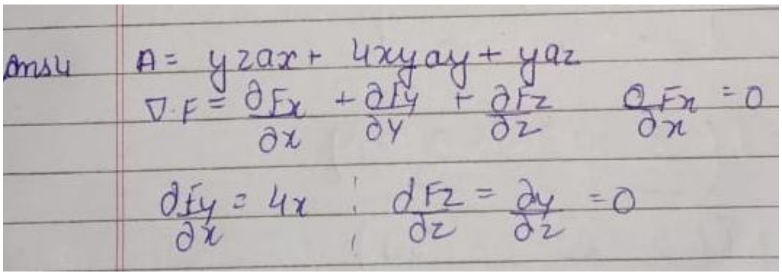
Output:

```
Divergence of the vector field:
x + 2*x*y*z

Divergence of the vector field:
9*z^2 + 4*y + 1

Divergence in cylindrical coordinates:
cos(theta) + 2*sin(theta)

Divergence in spherical coordinates:
(cos(phi)*cos(theta))/sin(phi) + (cos(phi)*cos(theta))/(r*sin(phi))
```

Q4: Write the matlab program determining the divergence of vector field

$\boldsymbol{A} = \boldsymbol{yz}\ \boldsymbol{az} + \ \boldsymbol{4xy}\ \boldsymbol{ay} + \boldsymbol{y}\ \boldsymbol{az}$ at (1,-2,3)

Code:



Ans4 $A = yz\,ax + 4xy\,ay + y\,az$

$\nabla \cdot F = \frac{\partial F_x}{\partial x} + \frac{\partial F_y}{\partial y} + \frac{\partial F_z}{\partial z}$ $\quad \frac{\partial F_x}{\partial x} = 0$

$\frac{\partial F_y}{\partial x} = 4x$; $\frac{d F_z}{\partial z} = \frac{\partial y}{\partial z} = 0$

```
% Define symbolic variables
syms x y z
% Define the components of the vector field F
F_x = y*z;
F_y = 4*x*y;
F_z = y;
% Define the vector field
F = [F_x, F_y, F_z];
% Calculate the divergence of the vector field
div_F = divergence (F, [x, y, z]);
% Substitute the point (1, -2, 3) into the divergence
div_value = subs (div_F, [x, y, z], [1, -2, 3]);
% Display the result
disp('The divergence of the vector field at (1, -2, 3) is:');
disp(double(div_value));
```

Output:

```
The divergence of the vector field at (1, -2, 3) is:
     4
```

Q5: Determine the curl of each of the vector field P, Q, T in Q19 and also write the matlab program.

Code:

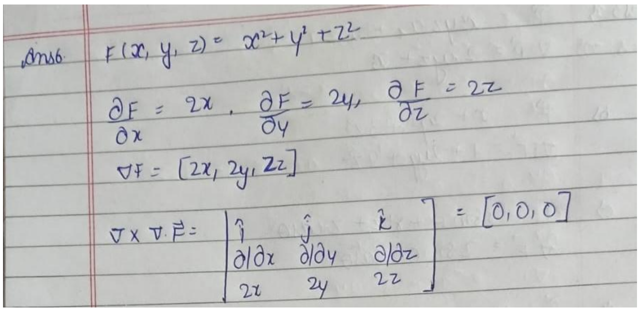
```
% Define symbolic variables
syms x y z rho phi theta r
%% Part (a) P = yz * ax + xz * az
P = [y*z, 0, x*z];
curl_P = curl (P, [x, y, z]);
disp('Curl of vector field P:')
disp(curl_P)
%% Part (b) R = x * ax + 2*y^2 * ay + 3*z^3 * az
R = [x, 2*y^2, 3*z^3];
curl_R = curl (R, [x, y, z]);
disp('Curl of vector field R:')
disp(curl_R)
%% Part (c) Q = ρ*sin(ϕ) * aρ + ρ^2*z * aϕ + z*cos(ϕ) * az
% Cylindrical coordinates (ρ, ϕ, z)
Q_rho = rho * sin(phi);
Q_phi = rho^2 * z;lkul
Q_z = z * cos(phi);
Q = [Q_rho, Q_phi, Q_z];
curl_Q = curl (Q, [rho, phi, z]);
disp('Curl of vector field Q (cylindrical coordinates):')
disp(curl_Q)
%% Part (d) T = (1/r^2) * cos(theta) * ar + r*sin(theta)*cos(phi) * az + cos(theta) * aϕ
% Spherical coordinates (r, θ, ϕ)
T_r = (1/r^2) * cos(theta);
T_theta = cos(theta);
T_phi = r * sin(theta) * cos(phi);
T = [T_r, T_theta, T_phi];
curl_T = curl (T, [r, theta, phi]);
disp('Curl of vector field T (spherical coordinates):')
disp(curl_T)
```

Output:

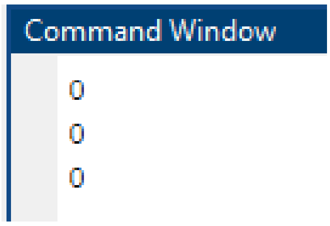
```
Curl of vector field P:
     0
y - z
   -z

Curl of vector field R:
0
0
0

Curl of vector field Q (cylindrical coordinates):
  - rho^2 - z*sin(phi)
                     0
2*rho*z - rho*cos(phi)

Curl of vector field T (spherical coordinates):
r*cos(phi)*cos(theta)
 -cos(phi)*sin(theta)
        sin(theta)/r^2
```

Q6: Find the curl of the gradient of a given scalar field of given as $\boldsymbol{f = x^2 + y^2 + z^2}$ that is $\ddot{\nabla} * \ddot{\nabla}$f.

Code:



Output:




```
syms x y z
% Define the scalar field F
F = x^2 + y^2 + z^2;
% Calculate the gradient of the scalar field
grad_F = gradient (F, [x, y, z]);
% Calculate the curl of the gradient
curl_grad_F = curl (grad_F, [x, y, z]);
```

Q7: Find the divergence of the curl of a given vector field P given in que19 that is $\ddot{\nabla} * \ddot{\nabla}$P and write the matlab program.

Code:

```
% Define symbolic variables
syms x y z rho phi theta r
%% Part (a) P = yz * ax + xz * az
P = [y*z, 0, x*z];
curl_P = curl (P, [x, y, z]);
div_curl_P = divergence (curl_P, [x, y, z]);
disp('Curl of vector field P:')
disp(curl_P)
disp('Divergence of the curl of P:')
disp(div_curl_P)
%% Part (b) R = x * ax + 2*y^2 * ay + 3*z^3 * az
R = [x, 2*y^2, 3*z^3];
curl_R = curl (R, [x, y, z]);
div_curl_R = divergence (curl_R, [x, y, z]);
disp('Curl of vector field R:')
disp(curl_R)
disp('Divergence of the curl of R:')
disp(div_curl_R)
%% Part (c) Q = ρ*sin(ϕ) * aρ + ρ^2*z * aϕ + z*cos(ϕ) * az
% Cylindrical coordinates (ρ, ϕ, z)
Q_rho = rho * sin(phi);
Q_phi = rho^2 * z;
```

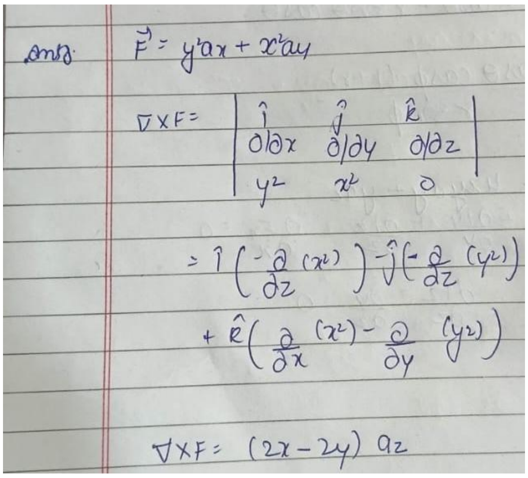
```
Q_z = z * cos(phi);
Q = [Q_rho, Q_phi, Q_z];
curl_Q = curl (Q, [rho, phi, z]);
div_curl_Q = divergence (curl_Q, [rho, phi, z]);
disp('Curl of vector field Q (cylindrical coordinates):')
disp(curl_Q)
disp('Divergence of the curl of Q:')
disp(div_curl_Q)
%% Part (d) T = (1/r^2) * cos(theta) * ar + r*sin(theta)*cos(phi) * az + cos(theta) * aϕ
% Spherical coordinates (r, θ, ϕ)
T_r = (1/r^2) * cos(theta);
T_theta = cos(theta);
T_phi = r * sin(theta) * cos(phi);
T = [T_r, T_theta, T_phi];
curl_T = curl (T, [r, theta, phi]);
div_curl_T = divergence (curl_T, [r, theta, phi]);
disp('Curl of vector field T (spherical coordinates):')
disp(curl_T)
disp('Divergence of the curl of T:')
disp(div_curl_T)
```

Output:

```
Divergence of the curl of P:
0

Curl of vector field R:
0
0
0

Divergence of the curl of R:
0

Curl of vector field Q (cylindrical coordinates):
  - rho^2 - z*sin(phi)
                     0
2*rho*z - rho*cos(phi)

Divergence of the curl of Q:
0
```

Q8: Evaluate the curl of vector field $y^2\ ax + x^2\ ay$

Code:


ans: $\vec{F} = y^2 a_x + x^2 a_y$

$$\nabla \times F = \begin{vmatrix} \hat{i} & \hat{j} & \hat{k} \\ \partial/\partial x & \partial/\partial y & \partial/\partial z \\ y^2 & x^2 & 0 \end{vmatrix}$$

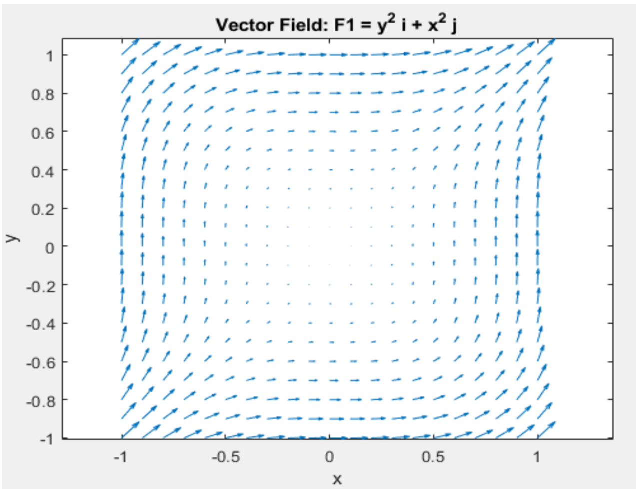
$$= \hat{i}\left(-\frac{\partial}{\partial z}(x^2)\right) - \hat{j}\left(-\frac{\partial}{\partial z}(y^2)\right) + \hat{k}\left(\frac{\partial}{\partial x}(x^2) - \frac{\partial}{\partial y}(y^2)\right)$$

$$\nabla \times F = (2x - 2y)\, a_z$$

```
F1 = [y^2, x^2, 0]; % Vector field for (a)
curl_F1 = curl (F1, [x, y, z]); % Curl of F1
disp('Curl of F1 (y^2 i + x^2 j):');
disp(curl_F1);
[X, Y] = meshgrid(-1:0.1:1, -1:0.1:1);
% Define the vector field components for both fields over the mesh
U1 = Y.^2; % x-component of F1
V1 = X.^2; % y-component of F1
% Plot the vector field for (a) F1
figure;
quiver (X, Y, U1, V1);
title ('Vector Field: F1 = y^2 i + x^2 j');
xlabel('x');
ylabel('y');
axis equal;
```

Output:

```
Curl of F1 (y^2 i + x^2 j):
        0
        0
2*x - 2*y
```

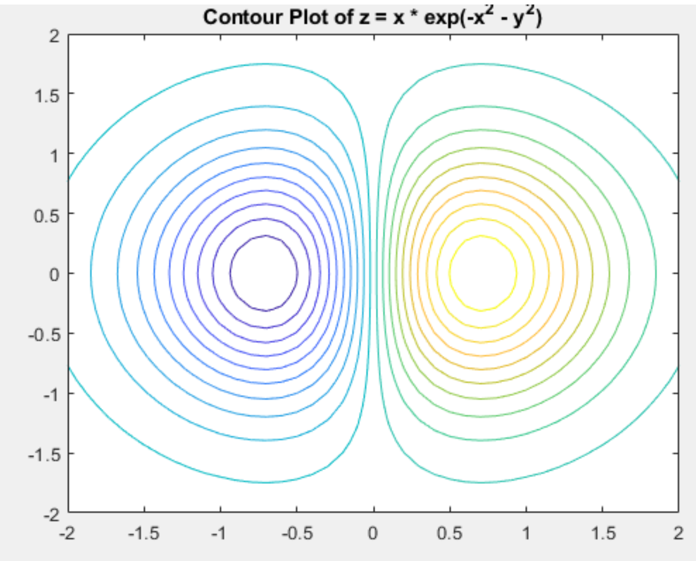


Que9: Find the numerical gradient using contour and quiver for z=xexp(-X^2-Y^2) and observe the gradient and velocity curve at the output.

Code:

```
% Create a meshgrid for x and y over the range -2:0.1:2
[x, y] = meshgrid (-2:0.1:2, -2:0.1:2);
% Define the scalar field z = x*exp (-x^2 - y^2)
z = x.* exp (-x.^2 - y.^2);
% Compute the numerical gradient of z with respect to x and y
[dz_dx, dz_dy] = gradient (z, 0.1, 0.1); % 0.1 is the step size in both directions
% Plot the contour of the scalar field
figure;
contour (x, y, z, 20); % 20 contour lines
title ('Contour Plot of z = x * exp (-x^2 - y^2)');
xlabel('x');
ylabel('y');
```

Output:



Experiment no. 5

Q1: Two-point charges of 8nC each are located at (0,0,1) and (0,0,-1). Write a matlab program and verify the answer analytically.

Output:

```
3.4110    6.8221    8.3168
```

Code**:**

Ans) Electric field at (2,3,4) $\boxed{E = k\frac{q}{r^2}\hat{r}}$

Electric field from first charge at (0,0,1)

$\vec{r_1} = 2\hat{i} + 3\hat{j} + 3k$

$|\vec{r_1}| = \sqrt{4+9+9} = \sqrt{22}$

$E_1 = k\frac{q}{r_1^2} = 9 \times 10^9 \times \frac{8\times10^{-9}}{22}$

$E_1 = \frac{72}{22} = 32.7$ N/C

Electric field vector

$E_1 = E_1 \times \frac{\vec{r_1}}{|r_1|} = \frac{327}{\sqrt{22}}(2\hat{i} + 3\hat{j} + 3\hat{k})$

Electric field from 2nd change at (0,0,-1)

$\vec{r_2} = 2\hat{i} + 3\hat{j} + 5\hat{k}$

$|\vec{r_2}| = \sqrt{4-9+25} = \sqrt{38}$

$E_2 = k\frac{q}{r_2^2} = 9\times10^9 \frac{8\times10^{-9}}{38}$

$E_2 = \frac{72}{38} = 1.89$ N/C

Electric field vector

$E_2 = E_2 \frac{\vec{r_2}}{|\vec{r_2}|} = \frac{1.89}{\sqrt{38}}(2\hat{i} + 3\hat{j} + 5\hat{k})$

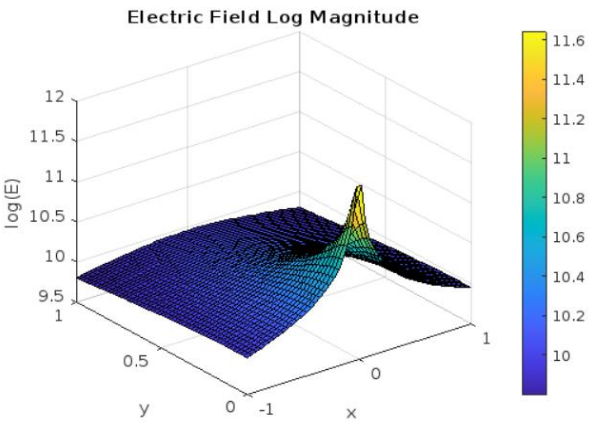
```
function E = electric_field (x, y, z)
   % Constants
   k = 8.99e9; % Coulomb's constant
   q = 8e-9; % Charge of each point charge
   % Positions of the charges
   q1 = [0 0 1];
   q2 = [0 0 -1];
   % Calculate the distance vectors from each charge to the point P
   r1 = [x y z] - q1;
   r2 = [x y z] - q2;
   % Calculate the magnitudes of the distance vectors
   r1_mag = norm(r1);
   r2_mag = norm(r2);
   % Calculate the unit vectors
   r1_hat = r1 / r1_mag;
   r2_hat = r2 / r2_mag;
   % Calculate the electric field due to each charge
   E1 = k * q * r1_hat / r1_mag^2;
   E2 = k * q * r2_hat / r2_mag^2;
   % Calculate the total electric field
 E = E1 + E2;
end
E_field = electric_field(1, 2, 3);
disp(E_field);
```

Q2: Using matlab plot the electric field due to a line charge along z axis based on equation E=1/$\sqrt{(x^2+y^2)}$. Plot it over domain-1<=x<1, 0.001<=y<=1,-1<=z<=1. Title it 'Electric field log magnitude'.

Code:

```
function plot_electric_field(x_lim, y_lim)
   % Constants
   k = 8.99e9; % Coulomb's constant
   lambda = 1; % Linear charge density
   % Create a 3D meshgrid
   [x, y] = meshgrid(linspace(x_lim(1), x_lim(2), 50), ...
                    linspace(y_lim(1), y_lim(2), 50));
   % Calculate the distance from each point to the line charge
   r = sqrt (x.^2 + y.^2);
   % Calculate the electric field magnitude
   E_mag = k * lambda. / r;
   % Take the log of the magnitude for better visualization
   E_log = log10(E_mag);
   % Plot the electric field magnitude
   figure;
   surf (x, y, E_log);
   title ('Electric Field Log Magnitude');
   xlabel('x');
   ylabel('y');
   zlabel('log(E)');
   colorbar;
   end
% Set the domain
x_lim = [-1, 1];
y_lim = [0.001, 1];
z_lim = [-1, 1];
% Plot the electric field
plot_electric_field(x_lim, y_lim, z_lim);
```

Output:



Experiment no. 6

Plots Magnetic Flux density due to Current carrying wire

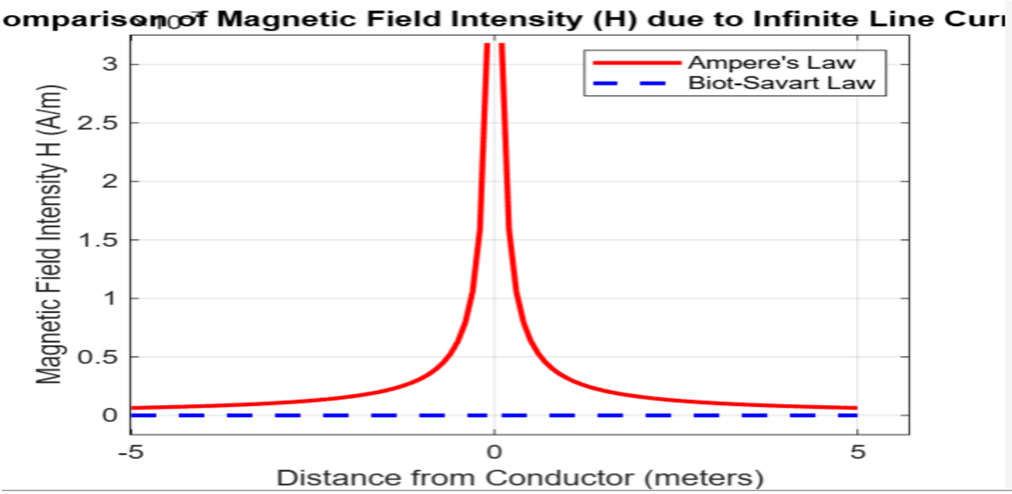
Q1: Determine the magnetic field intensity H due to a straight current-carrying conductor of an infinite length, camping current 200nA both using Ampere's circuit law and from basic principles. Also write matlab program to plot variations of H with due to this current-carrying conductor [Hint: H=I/(2pi*rho) afi, Infinite length conductor is a special case of finite length conductor]. Label the axes as 'distance from Conductors' and 'Magnetic field Intensity H'. Take distances [-5:0.1:5]. Also, Title it as 'Plots of Magnetic field Intensity due to an Infinite line Current

Code:

```
% MATLAB code to calculate and compare magnetic field intensity H due to an infinite current-
carrying conductor using two principles
% Given Parameters
I = 200e-9; % Current in amperes (200 nA)
mu0 = 4 * pi * 1e-7; % Permeability of free space (H/m)
% Define the distance range from the conductor (-5m to 5m)
rho = -5:0.1:5; % Distance from the conductor (in meters)
% Initialize magnetic field intensity arrays for both principles
H_ampere = zeros(size(rho)); % Using Ampere's Law
H_biot = zeros(size(rho)); % Using Biot-Savart Law
% Loop over each distance rho to calculate H using both methods
for i = 1: length(rho)
  if rho(i) == 0
    H_ampere(i) = Inf; % Handle singularity at rho = 0 (Ampere's Law)
    H_biot(i) = Inf; % Handle singularity at rho = 0 (Biot-Savart Law)
  else
    % Ampere's Law: H = I / (2 * pi * rho)
    H_ampere(i) = I / (2 * pi * abs(rho(i)));

    % Biot-Savart Law: H = (mu0 * I) / (2 * pi * rho)
    H_biot(i) = (mu0 * I) / (2 * pi * abs(rho(i)));
  end
end
% Display calculated values for both principles
disp('Distance from Conductor (meters) and Corresponding Magnetic Field Intensity H (A/m):');
disp('Using Ampere''s Law:');
disp(table(rho', H_ampere', 'VariableNames', {'Distance_rho_m',
'Magnetic_Field_Intensity_H_A_m_Ampere'}));
disp('Using Biot-Savart Law:');
disp(table(rho', H_biot', 'VariableNames', {'Distance_rho_m',
'Magnetic_Field_Intensity_H_A_m_Biot'}));
% Plotting the magnetic field intensity H for both methods
figure;
plot(rho, H_ampere, 'r', 'LineWidth', 2); hold on;
```

```
plot(rho, H_biot, 'b--', 'LineWidth', 2);
xlabel('Distance from Conductor (meters)');
ylabel('Magnetic Field Intensity H (A/m)');
title('Comparison of Magnetic Field Intensity (H) due to Infinite Line Current');
legend('Ampere''s Law', 'Biot-Savart Law');
grid on;
% Adjust plot to avoid infinite values
set(gca, 'YLim', [0 max(H_ampere(H_ampere < Inf))], 'XLim', [-5 5]);
```

Output:



Q2: For an infinitely long transmission line consisting of two concentric cylinders with inner conductor having radius 'a' and current I and outer conductor has radius 'b' and thickness t with return current -I, determine H everywhere that is in the following region.

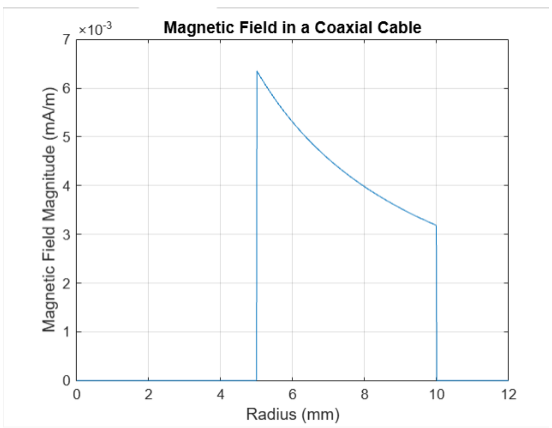
i) $0 <= \rho <= a$
ii) $a <= \rho <= b$
iii) $b <= \rho <= b+t$
iv) $\rho >= b+t$

Also write a matlab program to plot variant of |H| in different region.
Given I=200nA, a=5mm, b=10mm, t=1mm.
Label the axis and give the title appropriately.

Code:

```
% Parameters
I = 200e-9; % Current in Amperes
a = 5e-3; % Inner radius in meters
b = 10e-3; % Outer inner radius in meters
t = 1e-3; % Thickness of the outer conductor in meters
% Define the radial distance range
rho = linspace(0, b+2*t, 1000);
% Calculate the magnetic field in each region
H = zeros(size(rho));
H (rho <= a) = 0;
H (rho > a & rho <= b) = I./ (2*pi*rho (rho > a & rho <= b));
H (rho > b & rho <= b+t) = 0;
H (rho > b+t) = 0;
% Plot the magnetic field magnitude
figure;
plot (rho*1000, abs(H)*1000); % Convert to mm and mA/m
xlabel('Radius (mm)');
ylabel('Magnetic Field Magnitude (mA/m)');
title ('Magnetic Field in a Coaxial Cable');
grid on;
```

Output:



Experiment no. 7

Q1 In a nonmagnetic medium E = 4sin (2pi * 10 ^ 7 * t- 0.8x) * az V/m
Find (a) epsilon,eta (b) The time-average power carried by the wave (c) The total power crossing 100c *m ^ 2 of plane 2x + y = 5.

Code:

1 a) Since $\alpha = 0$ & $\beta \neq \omega/c$, the medium is not free space but a lossless medium:

$\beta = 0.8,\ \omega = 2\pi \times 10^7;\ \mu = \mu_0$

$\epsilon_0 = \epsilon_0 \epsilon_r$

$\beta = \omega\sqrt{\mu\epsilon} = \omega\sqrt{\mu\epsilon_0\epsilon_r} = \frac{\omega}{c}\sqrt{\epsilon_r}$

$\sqrt{\epsilon_r} = \frac{\beta c}{\omega} = \frac{0.8(3\times10^8)}{2\pi\times10^7} = \frac{12}{\pi}$ $\boxed{\epsilon_r = 14.59}$

$\eta = \sqrt{\frac{\mu}{\epsilon}} = \sqrt{\frac{\mu_0}{\epsilon_0\epsilon_r}} = \frac{120\pi}{\sqrt{\epsilon_r}} = 120\pi \times \frac{\pi}{12} = 10\pi^2$

$= 98.7\pi$

b) $P = E\times H = \frac{E_0^2}{\eta}\sin^2(\omega t - \beta x)a_x$

$P_{avg} = \frac{1}{T}\int_0^T P\,dt = \frac{E_0^2}{2\eta}a_x = \frac{16}{2\times10\pi^2}a_x$

$= 81\ a_x\ mw/m^2$

c) on plane, $2x + y = 5$ $a_n = \frac{2a_x + a_y}{\sqrt{5}}$

total power

$P_{avg} = \int P_{avg}\cdot ds = P_{avg}\cdot S a_n$

$= (81\times10^{-3}a_x)\cdot(100\times10^{-4})\left[\frac{2a_x + a_y}{\sqrt{5}}\right] = 724.5\ \mu w$

```matlab
% Given constants
mu_0 = 4 * pi * 1e-7; % Permeability of free space (H/m)
epsilon_0 = 8.854e-12; % Permittivity of free space (F/m)
c = 3e8; % Speed of light (m/s)
omega = 2 * pi * 10^7; % Angular frequency (rad/s)
beta = 0.8; % Phase constant (rad/m)
E0 = 4; % Electric field amplitude (V/m)
A_cm2 = 100; % Area in cm^2
% Convert area to m^2
A_m2 = A_cm2 * 1e-4;
% 1. Calculate relative permittivity (epsilon_r)
epsilon_r = (beta * c / omega) ^2;
% 2. Calculate intrinsic impedance (eta)
eta = sqrt (mu_0 / (epsilon_0 * epsilon_r));
% 3. Time-average power per unit area
S_avg = (E0^2) / (2 * eta); % in W/m^2
% 4. Plane normal and wave propagation direction
normal_vector = [2, 1, 0]; % Coefficients of x, y, z in plane equation
wave_vector = [1, 0, 0]; % Wave propagates along +x direction
% Cosine of the angle between wave vector and plane normal
cos_theta = dot (wave_vector, normal_vector) / ...
  (norm(wave_vector) * norm(normal_vector));
% Effective area
A_eff = A_m2 * abs(cos_theta);
% 5. Total power crossing the plane
P_total = S_avg * A_eff; % in W
% Display results
fprintf('Relative permittivity (epsilon_r): %.3f\n', epsilon_r);
fprintf('Intrinsic impedance (eta): %.3f ohms\n', eta);
fprintf('Time-average power per unit area (S_avg): %.3f mW/m^2\n', S_avg * 1e3);
fprintf('Total power crossing the plane (P_total): %.3f uW\n', P_total * 1e6);
```

Output:

```
Relative permittivity (epsilon_r): 14.590
Intrinsic impedance (eta): 98.629 ohms
Time-average power per unit area (S_avg): 81.112 mW/m^2
Total power crossing the plane (P_total): 725.490 uW
```