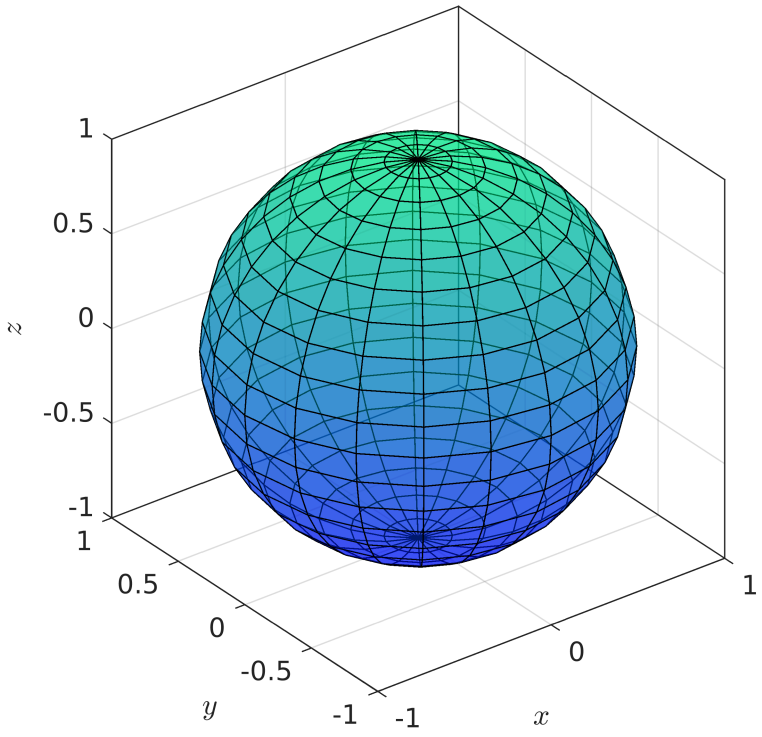
Módulo VIII. Gráficos

- Gráficas continuas y discretas.
- Mapas de color.
- Gráficas tridimensionales en coordenadas cartesianas, cilíndricas y esféricas.
- Campos vectoriales y curvas paramétricas.

Dibujar una esfera

```
clear;
clc;
set(0, 'defaultTextInterpreter', 'latex')
set(0, 'defaultLegendInterpreter', 'latex')
[x,y,z] = sphere;
% Definir centro de la esfera (xo, yo, zo):
xo = 0;
yo = 0;
zo = 0;

figure(1)
surf( x + xo, y + yo, z + zo, 'FaceAlpha', 0.5) % dibujar superficie
colormap winter
daspect([1 1 1]) % Poner los ejes del mismo tamaño
box on
hold on
xlabel('$x$')
ylabel('$y$')
zlabel('$z$')
```



Dibujar Superficie

Si se desea dibujar una superficie en el espacio tridimensional, se plantea una función de la forma $Z_{(x,y)}$ y se genera una rejilla con muchos puntos (x, y). Luego se evalúa la función Z en cada uno de estos puntos.

Ex. 1. Grafique la ecuación:

$z_{(x,y)} = y + x^2$

```
xs = -1 : 0.1 : 1; % El rango de valores de x a probar
ys = -1 : 0.1 : 1; % El rango de valores de y a probar
% Genere la rejilla de puntos (x,y) a partir de los rangos los ejes:
[X_space, Y_space] = meshgrid(xs, ys);
% Calcular la función Z:
Zs = Y_space - X_space.^2;

figure(1)
hold on
surf( X_space, Y_space, Zs, 'EdgeColor', 'w')
r = gca()
```

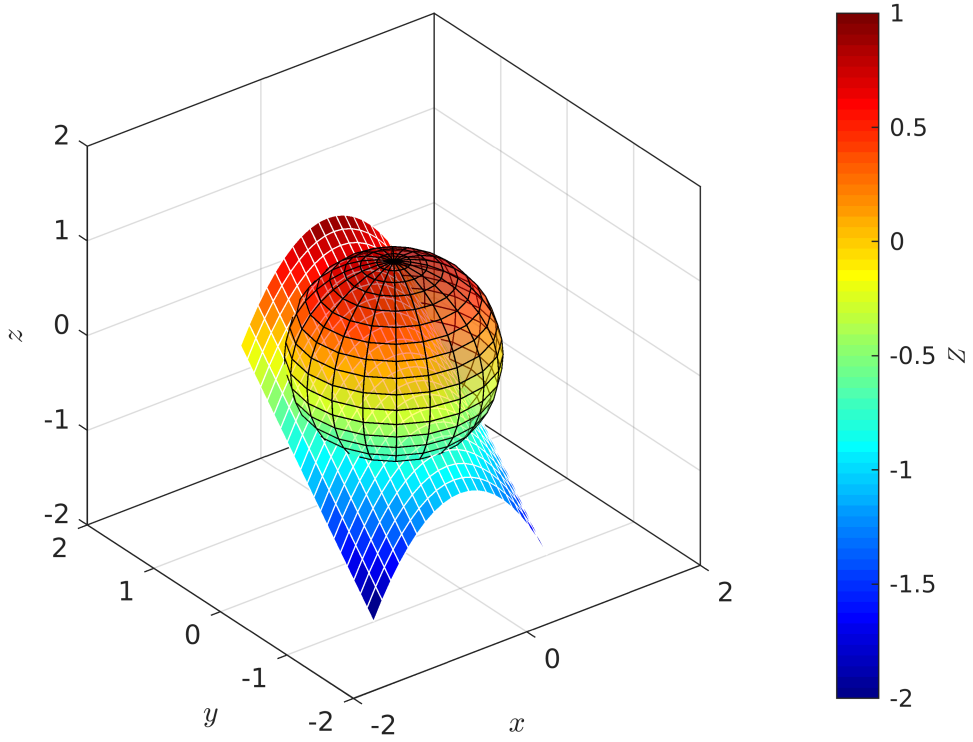
```
r =
```

```
  Axes with properties:

            XLim: [-1 1]
            YLim: [-1 1]
          XScale: 'linear'
          YScale: 'linear'
  GridLineStyle: '-'
       Position: [0.1300 0.1100 0.7750 0.8150]
          Units: 'normalized'

  Show all properties
```

```
xlim([-2, 2])
ylim([-2, 2])
zlim([-2, 2])
colormap(r, 'jet') % Cambiar mapa de colores
cb = colorbar(r)
```

```
cb =
  ColorBar with properties:

    Location: 'eastoutside'
      Limits: [-2 1]
    FontSize: 9
    Position: [0.8338 0.1100 0.0381 0.8150]
       Units: 'normalized'

  Show all properties
```

```
ylabel(cb, '$Z$', 'Interpreter', 'latex')
```



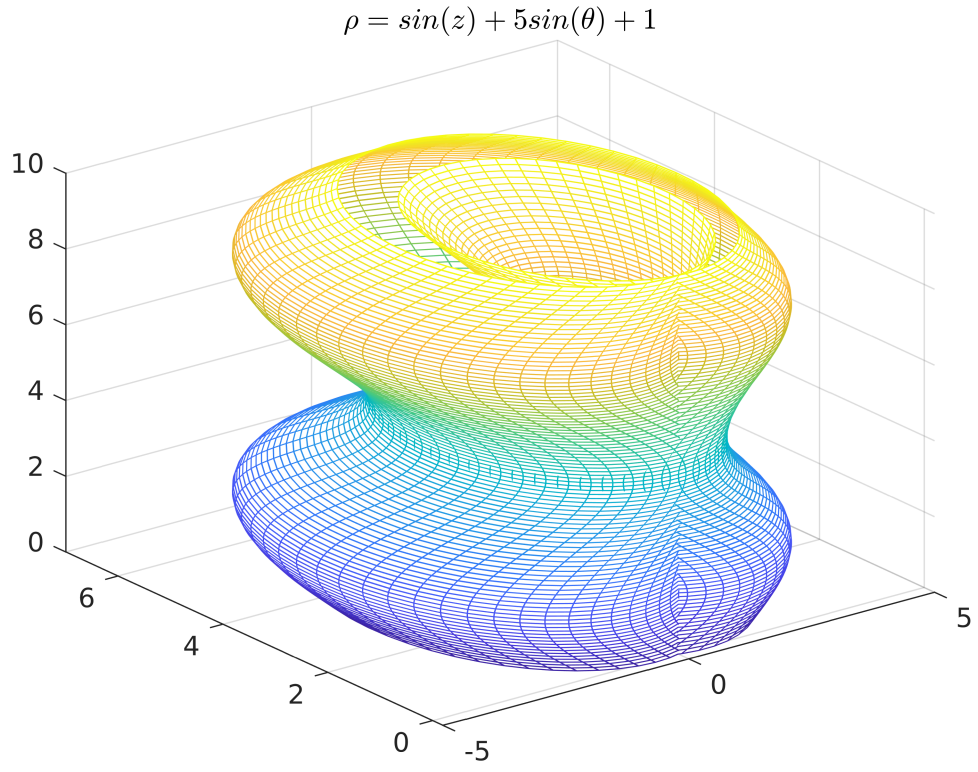
Superficie dada en coordenadas polares

Dada la función en términos de ρ, z y θ, se hace la rejilla con rangos para esas variables. Luego se transforman las coordenadas a cartesianas y se hace la gráfica.

Ex. 2. Graficar:

$\rho = sin(z) + 5sin(\theta) + 1$

```
clear; clc;
theta_0 = linspace(0,2*pi); % Crear rango de ángulos theta
z_0 = linspace(0,10); % Crear rango de alturas z
% Hacer rejilla en theta, z:
[theta_space, z_space] = meshgrid(theta_0, z_0);
% Calcular función:
rr = sin( z_space ) + 1 + 5*sin( theta_space );
% Transformar de polares a cartesianas:
[x_trans, y_trans, z_trans] = pol2cart(theta_space, rr,z_space);

figure()
mesh(x_trans, y_trans, z_trans) % Dibujar superficie
title('$\rho = sin(z) + 5sin(\theta) +1$')
```

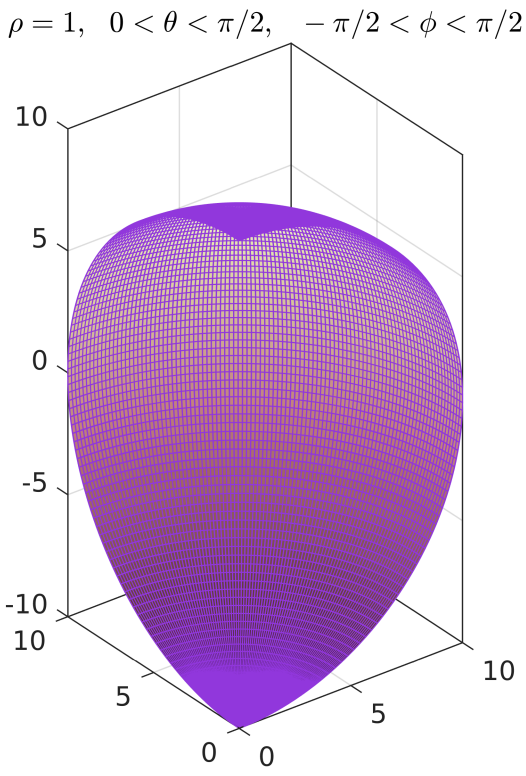


Superficie dada en coordenadas esféricas

Ex. 3. Graficar la superficie generada por los rangos:

$\rho = 1$
$0 < \theta < \pi/2$
$-\pi/2 < \phi < \pi/2$

```
theta_0 = linspace(0,pi/2); % rango de theta
phi_0 = linspace(-pi/2,pi/2); % rango de phi
r_0 = 10; % rango de radio
[theta_space, phi_space] = meshgrid(theta_0, phi_0); % Hacer rejilla en theta, phi
% Transformar de esféricas a cartesianas:
[x_trans, y_trans, z_trans] = sph2cart(theta_space, phi_space, r_0);

figure()
surf(x_trans, y_trans, z_trans, 'Edgecolor', [0.57, 0.22, 0.87])
colormap pink
daspect([1 1 1])
box on
title('$\rho = 1, ~~ 0 < \theta < \pi/2, ~~ -\pi/2 < \phi < \pi/2$')
```



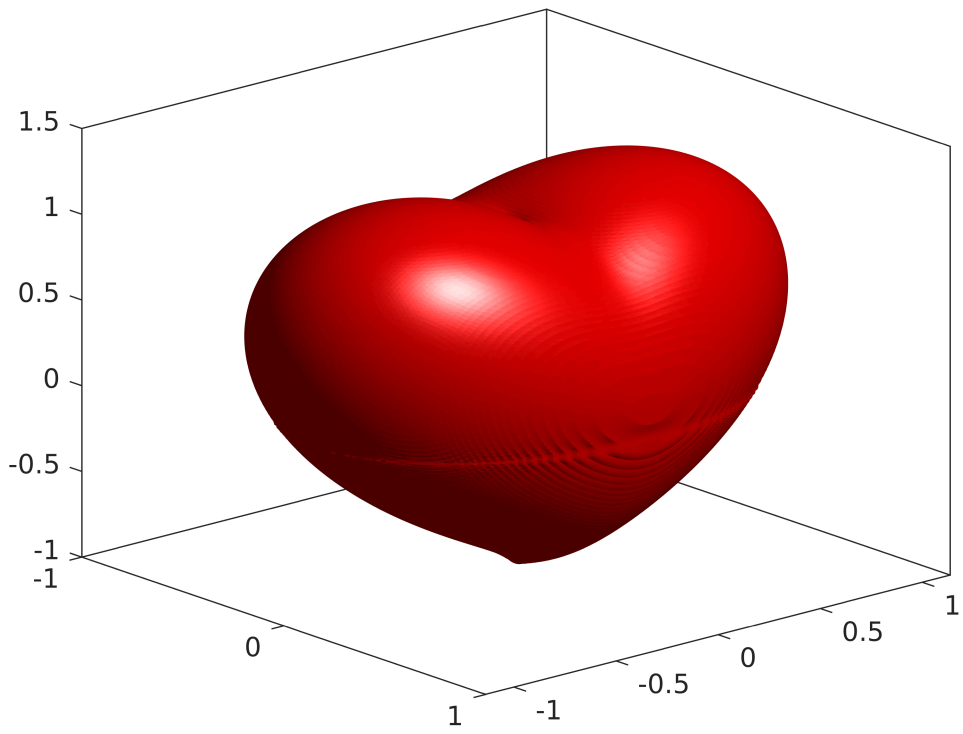
Dibujar Isosuperficie

Suponga un campo escalar en el espacio tridimensional $f_{(x,y,z)}$. Se pueden graficar todos los puntos que producen un mismo nivel del campo, es decir, $f_{(x,y,z)} = c$, siendo c una constante. Basta con generar la rejilla de puntos (x, y, z), calcular el campo f y buscar los puntos iguales a c.

Ex. 4. Graficar:

$$f_{(x,y,z)} = 0 = (y^2 + \frac{9}{4}x^2 + z^2 - 1)^3 - y^2z^3 - \frac{9}{80}x^2z^3$$

```
x_range = linspace(-2,2,300); % Rango de x
y_range = linspace(-2,2,300); % Rango de y
z_range = linspace(-2,2,300); % Rango de z
% Crear rejilla tridimensional:
[X_space, Y_space, Z_space] = meshgrid( x_range, y_range, z_range );
% Calcular campo:
f = (Y_space.^2 + 9/4*X_space.^2 + Z_space.^2-1).^3 ...
    - Y_space.^2.*Z_space.^3 ...
    - 9/80*X_space.^2.*Z_space.^3;

figure()
% Obtener caras y vértices de la isosuperficie f = 0
[faces,verts] = isosurface(X_space, Y_space, Z_space, f,.0);
% Graficar un parche con las caras y vértices sacados:
patch('Vertices',verts,'Faces',faces, 'FaceColor', 'r', 'EdgeColor', 'none')
box on
view(49,23);
camlight
```



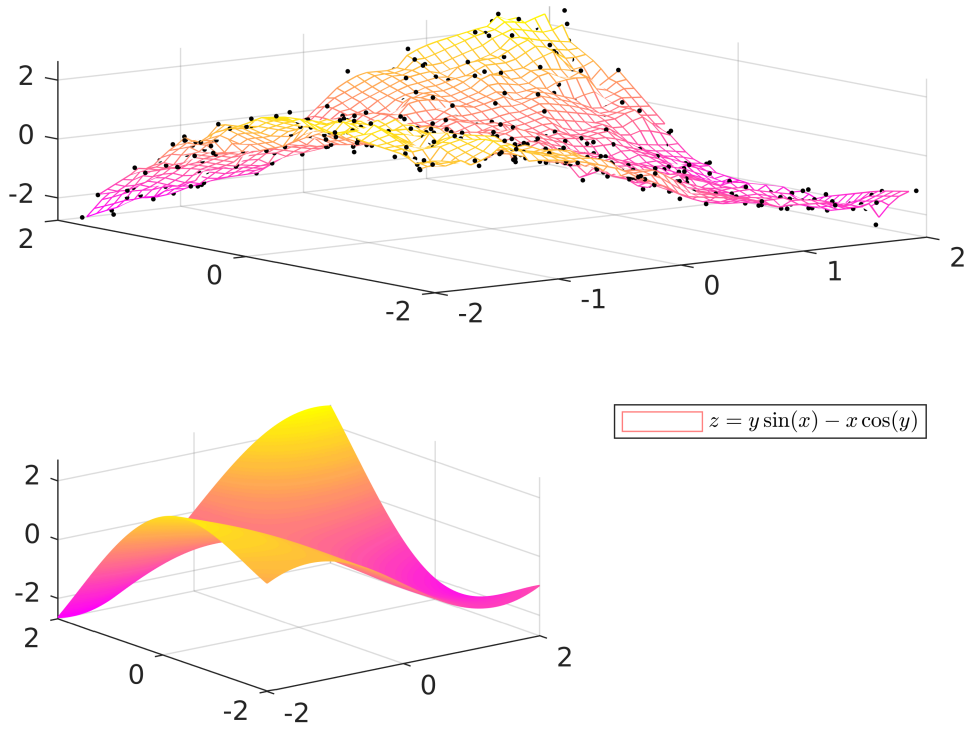
Ajustar una curva tridimensional a una serie de puntos

```
clear; clc;
x1_range = -2 : 0.01 : 2;
y1_range = -2 : 0.01 : 2;
[X, Y] = meshgrid(x1_range, y1_range);
Z = Y.*sin(X) - X.*cos(Y) + 0.2*randn( size(X) );

N = 500; % num puntos
S = [];

[rows, cols] = size( X );

for i = 1 : N

    pos_x = randi( rows );
    pos_y = randi( cols );

    S = [S; [ X(pos_x, pos_y), Y(pos_x, pos_y), Z(pos_x, pos_y) ]];
end
```

```
xs = S(:,1);
ys = S(:,2);
zs = S(:,3);

x_line = linspace(min(xs), max(xs), 50);
y_line = linspace(min(ys), max(ys), 50);
[X_space, Y_space] = meshgrid(x_line, y_line);
Z_fit = griddata(xs, ys, zs, X_space, Y_space);
```

```
Warning: Duplicate data points have been detected and removed - corresponding values have been
averaged.
```

```
Z_no_noise = Y.*sin(X) - X.*cos(Y);


figure()
subplot(2,1,1)
scatter3(xs, ys, zs, '.k')
hold on
mesh(X_space, Y_space, Z_fit)
colormap('spring')

subplot(2,1,2)
mesh(X, Y, Z_no_noise)
legend('$z = y \sin(x) - x \cos(y) $')
colormap('spring')
```



Curvas paramétricas

Múltiples ecuaciones que dependen de un mismo parámetro.

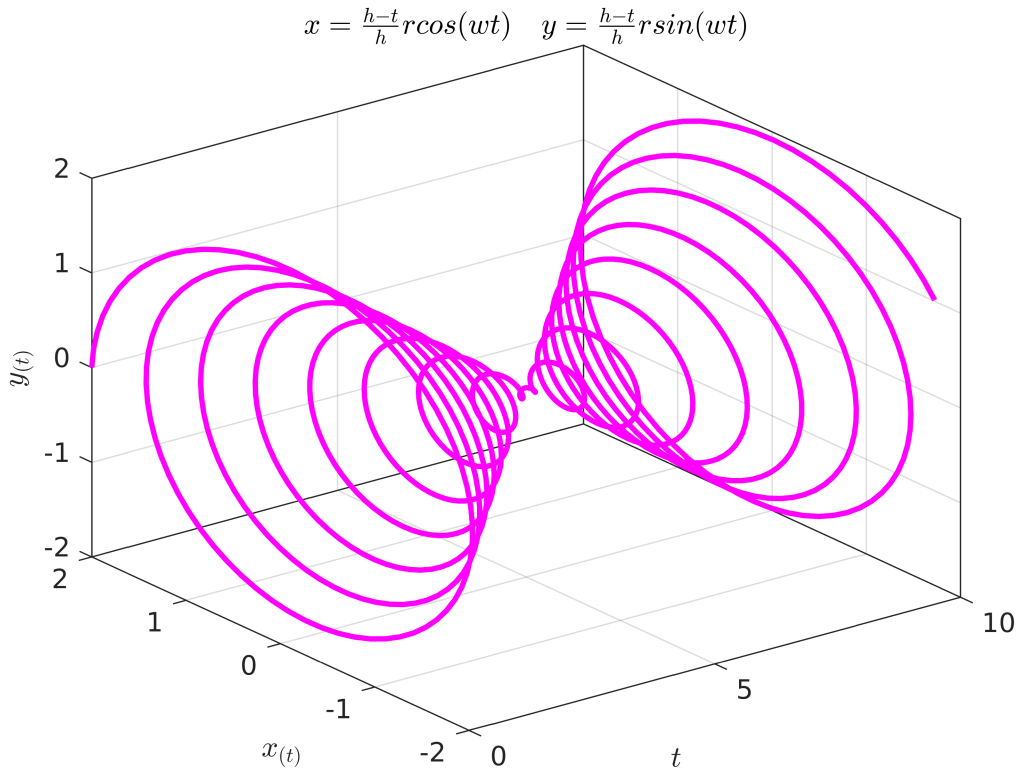
Ex. 5. Dibujar las curvas paramétricas:

$$x = \frac{h-t}{h} rcos(wt)$$

$$y = \frac{h-t}{h} rsin(wt)$$

```
h = 5;
r = 2;
w = 10;
t = 0 : 0.01 : 10;
x_t = ( (h - t)./h )*r.*cos(w*t);
y_t = ( (h - t)./h )*r.*sin(w*t);

figure()
plot3(t, x_t, y_t, 'Color', 'm', 'Linewidth', 2)
xlabel('$t$')
ylabel('$x_{(t)}$')
zlabel('$y_{(t)}$')
grid on
box on
```

```
title('$x = \frac{h-t}{h}rcos(wt) ~~~ y = \frac{h-t}{h}rsin(wt) $')
```

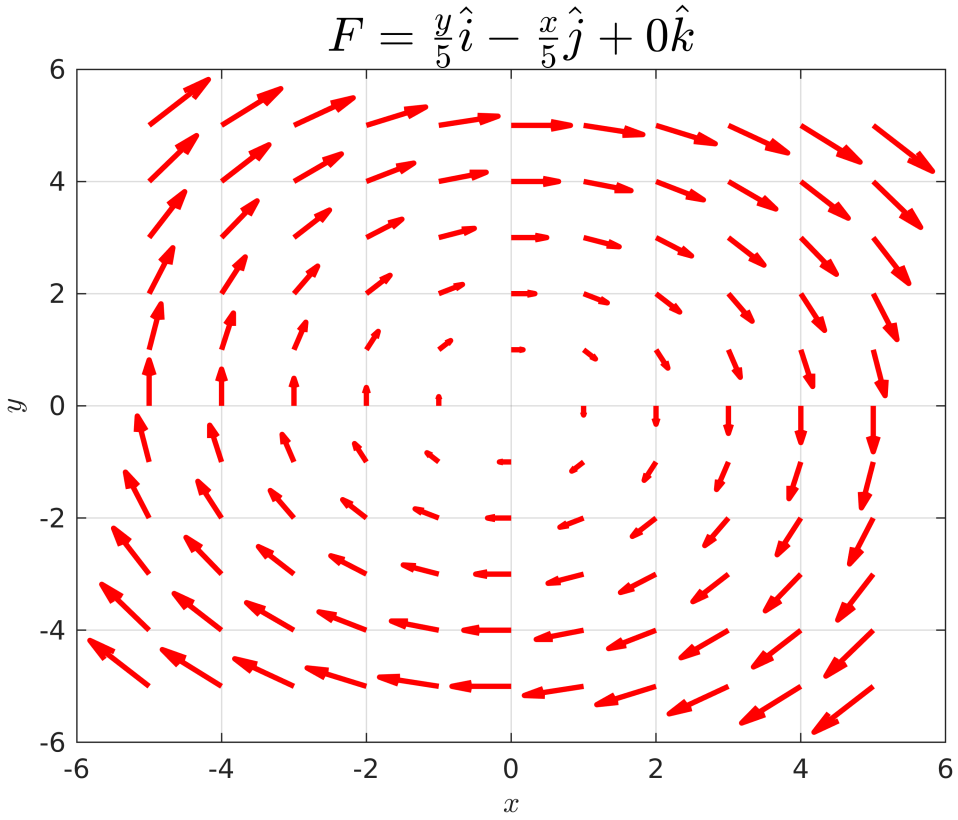


Campo vectorial

Ex. 6. Dibujar el campo vectorial:

$$F = \frac{y}{5}\hat{i} - \frac{x}{5}\hat{j} + 0\hat{k}$$

```
clear; clc;
x_range = -5 : 1 : 5;
y_range = -5 : 1 : 5;
[X_space, Y_space] = meshgrid(x_range, y_range);
figure(10)
quiver(X_space, Y_space, Y_space/5, -X_space/5, 'Color', 'r', 'LineWidth',2);
grid on
xlabel('$x$'), ylabel('$y$')
title('$F = \frac{y}{5}\hat{i} - \frac{x}{5}\hat{j} + 0\hat{k}$', 'Fontsize',18)
```

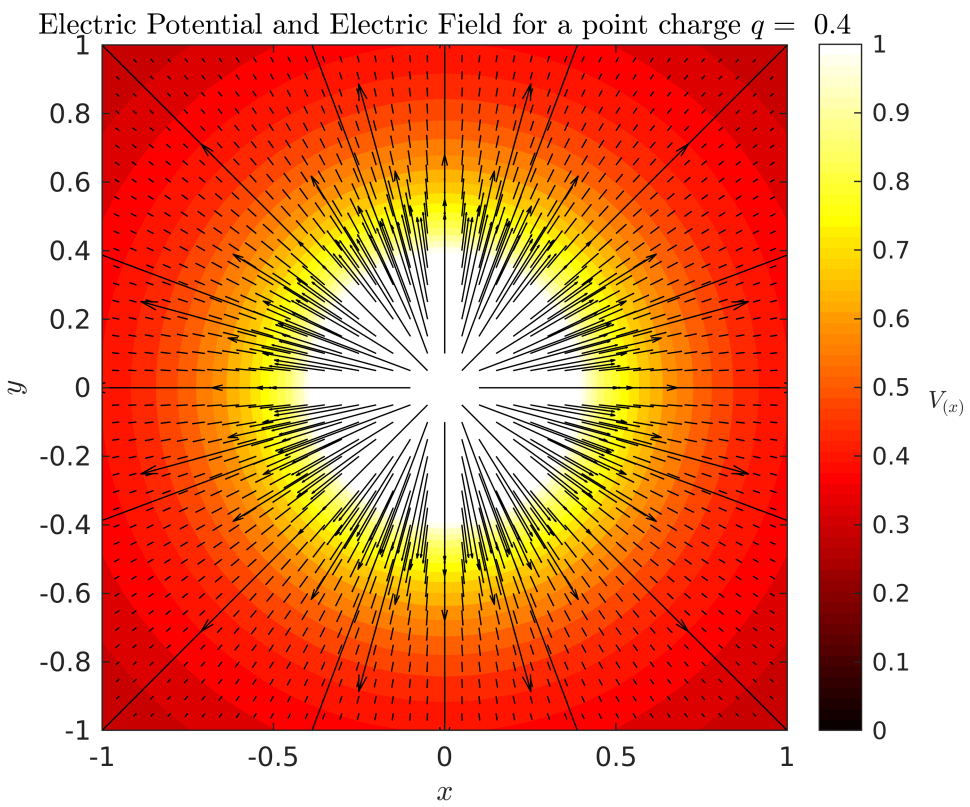


Mapa de Contorno

Ex. 7. Dibujar el mapa de contorno del potencial eléctrico generado por una carga eléctrica puntual.

```
q = 0.4;
k = 1;
x0 = 0; %ubicación de la carga
y0 = 0;
pos_0 = [x0, y0];

x_range = -1 : 0.05 : 1;
y_range = -1 : 0.05 : 1;
[X_space, Y_space] = meshgrid(x_range, y_range);
V_space = zeros( length(x_range) );

for i = 1 : length(x_range)
    for j = 1 : length(y_range)
        % Cakcular el potencial eléctrico
        % en el punto actual de la matriz
        pos_act = [x_range(i), y_range(j)];
        Vij = electric_pot(q, k, pos_0, pos_act);
        V_space(i,j) = Vij;
    end
end
```

```
% Calcular campo eléctrico con el gradiente del potencial
[Ex, Ey] = gradient(V_space);
set(0, 'defaultTextInterpreter', 'latex');
figure(1)
clf
contourf(X_space, Y_space, V_space, 200, 'LineColor','none')
caxis([0 1])
hold on
N = length(y_range);
for i = 1 : 1 : N
    for j = 1 : 1 : N
        % Graficar cada vector
        quiver(X_space(i,j), Y_space(i,j), ...
            -Ex(i, j), -Ey(i,j), 'k')
    end
end
xlabel('$ x $'), ylabel('$ y $');
xlim([-1, 1]), ylim([-1, 1])
colormap('hot')
cb = colorbar;

ylabel(cb, '$ V_{(x)} $', 'Interpreter', 'latex', ...
    'rotation', 0, 'position', [3, 0.5, 0])
daspect([1, 1, 1])
title( ['Electric Potential and Electric Field for a point charge $q = ~$', num2str(q)
```



```
function V_r = electric_pot(q, k, pos_0, pos_act)
    r = norm(pos_act - pos_0);
    V_r = k*q/r;
end
```