Cálculo y graficación de campos eléctricos no uniformes, como los usados en electroforesis para el diagnóstico de malaria

Integrantes:

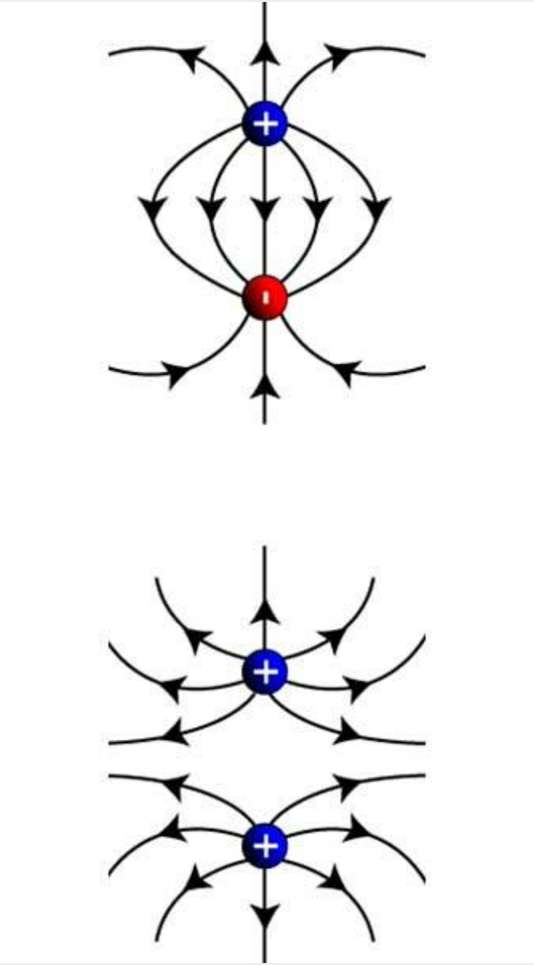
Juan Pablo David Lerma - A01283879

Abraham Cepeda Oseguera - A00827666

José Juan López Valenzuela - A01283642

Jesús Gerardo Rodríguez Tristán - A01283717

Carlos Sebastián Salinas Morales - A01283585



Contenidos

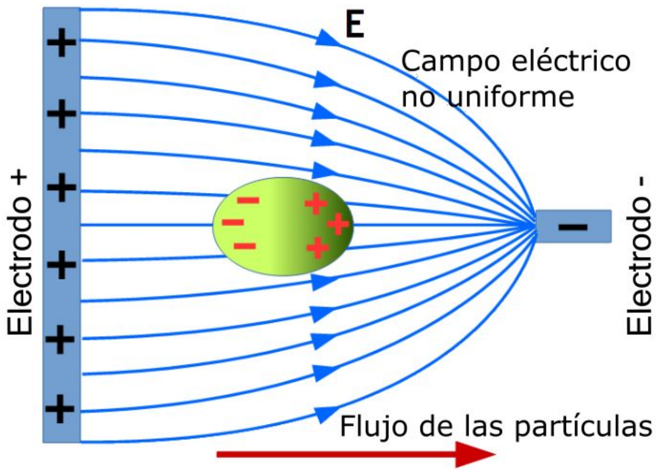
Situación problema: Diagnóstico de malaria

La malaria es una enfermedad parasitaria que involucra fiebres altas, escalofríos, síntomas similares a los de la gripe y anemia, esta es transmitida por mosquitos anofeles infectados. Un elemento importante lo es la fácil propagación de esta enfermedad, debido a eso hay diversas investigaciones científicas las cuales tienen como objetivo encontrar técnicas apropiadas para diagnosticar.

Dielectroforesis

La **dielectroforesis** es un método utilizado para diagnosticar malaria, esta técnica se basa en el uso de campos eléctricos para poder identificar y separar los glóbulos rojos sanos de los que se encuentran infectados.

Se basa en aplicar un campo eléctrico no uniforme, ejerciendo fuerzas netas distintas y permitiendo la separación de estas células.



Objetivo general del reto

- Creación de un programa el cual pueda llevar a cabo la simulación de un campo eléctrico similar a los utilizados en la Dielectroforesis para diagnosticar la malaria. Se debe hacer una representación gráfica por medio de dos electrodos con forma de placas planas, paralelas y de carga opuesta. El usuario podrá modificar el tamaño de las placas, al igual que el eje donde se van a desplegar, posteriormente se grafica el campo eléctrico no uniforme el cual debe ser representado por medio de flechas o líneas de fuerza. El usuario puede seleccionar un punto y se deben desplegar las componentes [Ex, Ey].

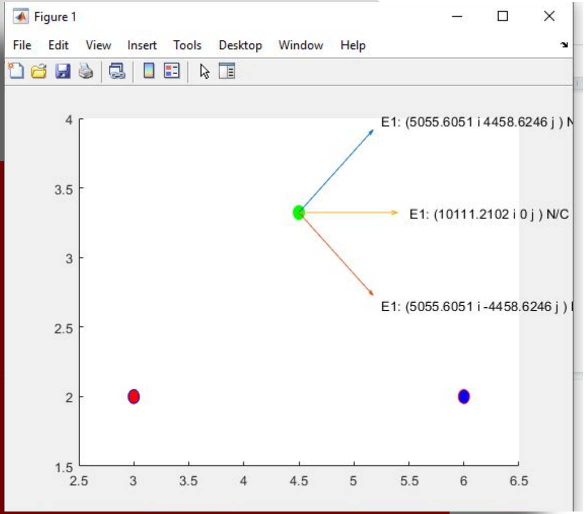
Campo eléctrico producido por un dipolo

Objetivo específico del entregable 1

Primer entregable: Simulación computacional del campo eléctrico, en MATLAB - dipolo.

- El objetivo del entregable uno es la simulación computacional (en Matlab) de las cargas eléctricas y las líneas de los campos eléctricos en 2D (coordenadas x,y) generando un dipolo. El programa debe permitir que el usuario especifique las coordenadas (x,y) de un punto en el plano 2D y se debe desplegar las componentes (Ex, Ey) del campo eléctrico total en un punto.
    - Dipolo: Hace referencia a dos cargas iguales en magnitud, pero de signo opuesto, las cuales se encuentran separadas por una distancia (2).

Procedimiento



1. Coordenadas de una de las cargas.
2. Distancia y dirección a la cual se dirigen
   a. Cálculos de coordenadas.
3. Magnitud de carga.
   a. Uso de fórmulas aprendidas en clase para poder obtener la magnitud del vector.
   b. Obtención de componentes en "x" y "y" del vector del campo eléctrico mediante identidades trigonométricas.

Ecuaciones y funciones

Ecuaciones de física:

$$E = \frac{kq}{r^2}$$

Ecuación campo eléctrico - sirve para calcular el campo eléctrico.

k → Constante de proporcionalidad.

q → Carga.

r → distancia entre las cargas.

Funciones de Matlab:

1. **quiver** → graficar vector.
2. **rectangle** → graficación de las cargas.
3. **text** → mostrar información campo eléctrico para cada vector.

Código

1

Uso de fórmulas para obtener el vector de los campos eléctricos

2

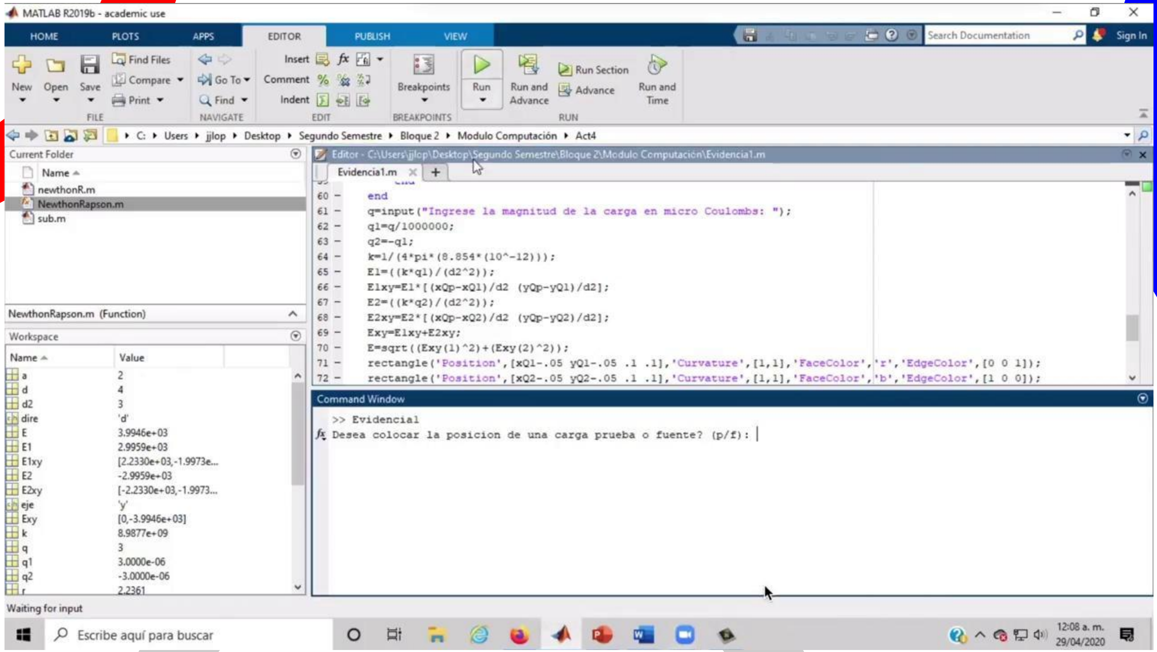
Función para graficar las cargas en el plano

```
k=1/(4*pi*(8.854*(10^-12)));
E1=((k*q1)/(d2^2));
E1xy=E1*[(xQp-xQ1)/d2 (yQp-yQ1)/d2];
E2=((k*q2)/(d2^2));
E2xy=E2*[(xQp-xQ2)/d2 (yQp-yQ2)/d2];
Exy=E1xy+E2xy;
E=sqrt((Exy(1)^2)+(Exy(2)^2));
rectangle('Position',[xQ1-.05 yQ1-.05 .1 .1],'Curvature',[1,1],'FaceColor','r','EdgeColor',[0 0 1]);
rectangle('Position',[xQ2-.05 yQ2-.05 .1 .1],'Curvature',[1,1],'FaceColor','b','EdgeColor',[1 0 0]);
rectangle('Position',[xQp-.05 yQp-.05 .1 .1],'Curvature',[1,1],'FaceColor','g','EdgeColor',[0 1 0]);
hold on
quiver(xQp,yQp,E1xy(1)/abs(E1),E1xy(2)/abs(E1))
txt = ['E1: (' num2str(E1xy(1)) ' i ' num2str(E1xy(2)) ' j ) ' 'N/C'];
text(xQp+(E1xy(1)/abs(E1)),yQp+(E1xy(2)/abs(E1)),txt);
quiver(xQp,yQp,E2xy(1)/abs(E2),E2xy(2)/abs(E2))
txt = ['E1: (' num2str(E2xy(1)) ' i ' num2str(E2xy(2)) ' j ) ' 'N/C'];
text(xQp+(E2xy(1)/abs(E2)),yQp+(E2xy(2)/abs(E2)),txt);
quiver(xQp,yQp,Exy(1)/abs(E),Exy(2)/abs(E))
txt = ['E1: (' num2str(Exy(1)) ' i ' num2str(Exy(2)) ' j ) ' 'N/C'];
text(xQp+(Exy(1)/abs(E)),yQp+(Exy(2)/abs(E)),txt);
hold off
```

Función para graficar el vector

3

Resultados - Casos



Campo eléctrico producido por cargas múltiples

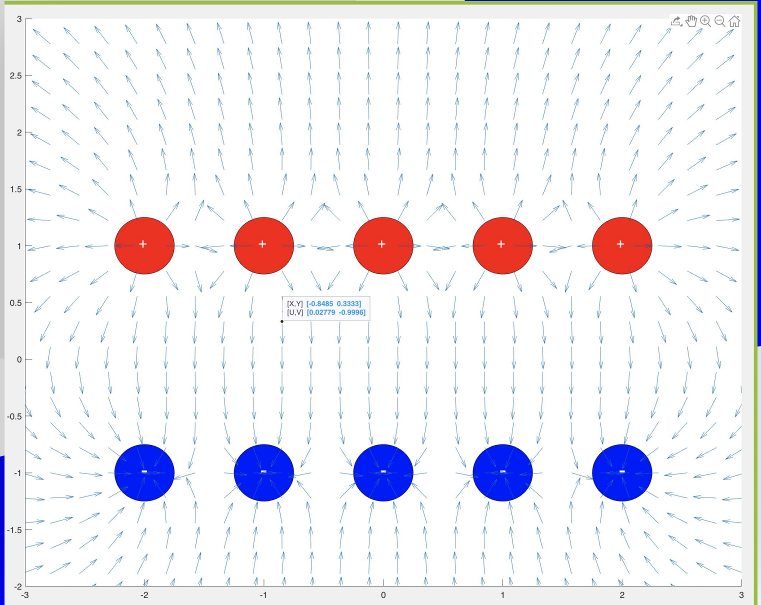
Objetivo específico del entregable 2

Segundo entregable: Simulación computacional del campo eléctrico, en MATLAB - cargas múltiples.

- El objetivo del entregable dos es la simulación computacional (en Matlab) del campo eléctrico, el cual se encuentra formado por múltiples cargas, paralelas, de cargas opuestas y del mismo tamaño, las cuales son de electrodos.
- Las cargas ahora serán múltiples, por lo que se generará una mayor cantidad de campos eléctricos y se mostrarán los vectores del campo eléctrico por todo el plano, seleccionando diferentes puntos del mismo.

Procedimiento

1. Cantidad cargas negativas y positivas.
2. Distancia entre cargas del mismo signo.
3. Distancia cargas distintas.
    a. Calcular coordenadas cargas.
    b. Calcular campos eléctricos en los puntos.
    c. Graficar campo eléctrico.



Ecuaciones y funciones

Funciones de física:

$$r = \sqrt{\Delta x^2 + \Delta y^2}$$

→ Calcular distancia

$$E = \frac{kq}{r^2}\ (cargas\ positivas)$$

$$E = \frac{-kq}{r^2}\ (cargas\ negativas)$$

→ Calcular campo eléctrico positivo o negativo.

$$E_x = E\frac{\Delta x}{r}$$

$$E_y = E\frac{\Delta y}{r}$$

→ Componentes en "x" y "y" de campo eléctrico.

$$E_{xtotal} = \sum E_x$$

$$E_{ytotal} = \sum E_y$$

→ Sumatoria campo eléctrico en "x" y "y".

$$E_{total} = \sqrt{E_{xtotal}^2 + E_{ytotal}^2}$$

→ Campo eléctrico total.

Funciones de computación:

1. **linspace** → crear espacio de puntos con separación equidistante.
   a. valor mínimo, valor máximo y particiones
2. **meshgrid** → genera matrices para coordenadas.
3. **quiver** → gráfica matriz de vectores.

Código

1

2

3

4

5

6

7

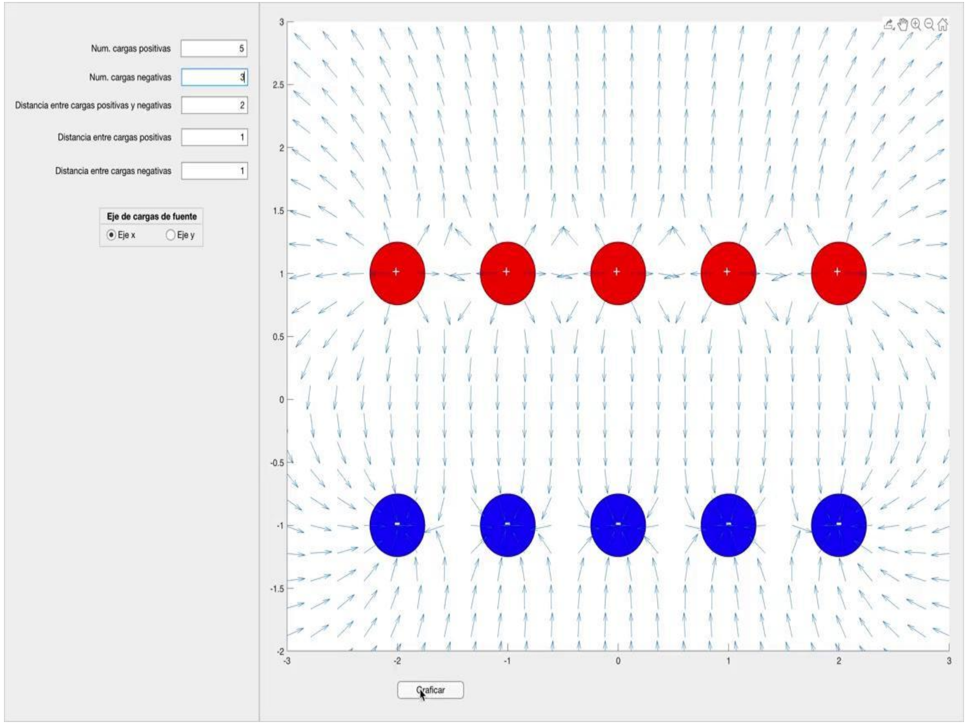
```matlab
function [Et, Ex, Ey, tx, ty] = getVectores(~, xPPos, yPPos, xNPos, yNPos, nP, nN)
    pts = 100;
    x =linspace(xPPos(1,nP)-10, xNPos(1,1)+10, pts);
    y =linspace(yNPos(1,1)-10, yPPos(1,nP)+10, pts);
    [tx,ty] = meshgrid(x,y); % Definir matriz de vectores

    k =  1/(4*pi*8.54e-12); % constante k
    q =  1.6e-19; % Valor de cargas fuente
    Ex = zeros(pts);
    Ey = zeros(pts);
    Et = zeros(pts);

    for contx = 1:pts
        for conty = 1:pts
            EpxT = 0;
            EpyT = 0;
            EnxT = 0;
            EnyT = 0;
            for i = 1:nP
                r = sqrt((tx(contx,conty) - xPPos(1,i))^2 + (ty(contx,conty) - yPPos(1,i))^2);
                Ec = k*q/r^2;
                EpxT = EpxT + Ec*(tx(contx,conty) - xPPos(1,i))/r;
                EpyT = EpyT + Ec*(ty(contx,conty) - yPPos(1,i))/r;
            end
            for i = 1:nN
                r = sqrt((tx(contx,conty) - xNPos(1,i))^2 + (ty(contx,conty) - yNPos(1,i))^2);
                Ec = -k*q/r^2;
                EnxT = EnxT + Ec*(tx(contx,conty) - xNPos(1,i))/r;
                EnyT = EnyT + Ec*(ty(contx,conty) - yNPos(1,i))/r;
            end
            Ex(contx, conty) = EpxT + EnxT;
            Ey(contx, conty) = EpyT + EnyT;
            Et(contx,conty) = sqrt(Ex(contx, conty)^2 + Ey(contx, conty)^2);
        end
    end
end

function plotVectores(~, Et, Ex, Ey, tx, ty, grafica)
    hold(grafica, "on")
    quiver(grafica, tx, ty, Ex./Et, Ey./Et, 'AutoScaleFactor',0.6)
end
```

Resultados - Casos



Campo eléctrico producido por dos placas

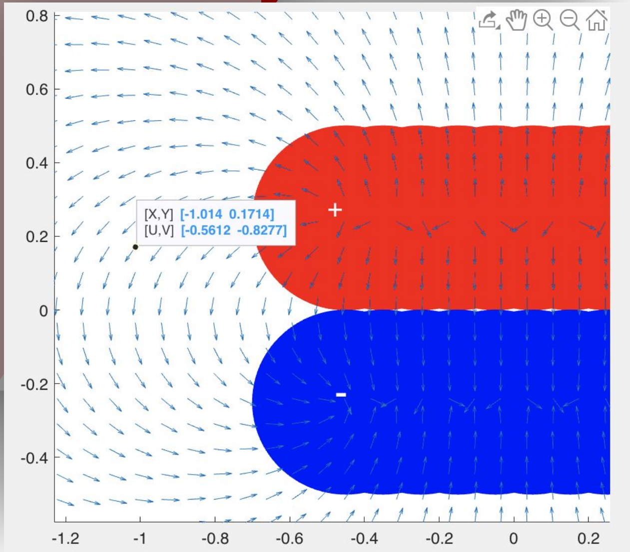
Objetivo específico del entregable 3

Tercer entregable: Simulación computacional del campo eléctrico, en MATLAB - placas.

- El objetivo del entregable tres es la simulación computacional del campo eléctrico (en Matlab) producido por dos placas planas, paralelas y de carga opuesta. El programa debe permitir que el usuario especifique el tamaño de cada una de las placas, al igual que puede seleccionar un punto en donde se deben desplegar las componentes (Ex, Ey) del campo eléctrico total en un punto.

Procedimiento

1. Longitud placas negativas y positivas.
2. Distancia entre placas.
   a. Ubicar las cargas en las placas.
   b. Calcular campos eléctricos en los puntos.
   c. Graficar campo eléctrico.



Ecuaciones y funciones

$$\int \vec{E}\cdot\vec{r} \qquad k\int_{L_{inicial}}^{L_{final}} \frac{\lambda * dl}{r^2}\cdot\vec{r} \qquad k\int_{L_{inicial}}^{L_{final}} \frac{\lambda * dl}{r^2}\cdot(\cos\theta i + \sin\theta j)$$

- Para la placa sobre el eje x:

$$E_i = k\int_{L_{inicial}}^{L_{final}} \frac{\lambda}{r^2}\cdot \cos\theta dx\ , \qquad E_j = k\int_{L_{inicial}}^{L_{final}} \frac{\lambda}{r^2}\cdot \sin\theta dx$$

$$E_i = k\int_{L_{inicial}}^{L_{final}} \frac{\lambda}{\sqrt{(a_x - x)^2 + {b_y}^2}^{\,2}}\cdot\frac{(a_x - x)}{\sqrt{(a_x - x)^2 + {b_y}^2}}dx\ , \qquad E_j = k\int_{L_{inicial}}^{L_{final}} \frac{\lambda}{\sqrt{(a_x - x)^2 + {b_y}^2}^{\,2}}\cdot\frac{b_y}{\sqrt{(a_x - x)^2 + {b_y}^2}}dx$$

$$E_i = k\int_{L_{inicial}}^{L_{final}} \frac{\lambda(a_x - x)}{((a_x - x)^2 + {b_y}^2)^{\frac{3}{2}}}\cdot dx\ , \qquad E_j = ky\int_{L_{inicial}}^{L_{final}} \frac{\lambda b_y}{((a_x - x)^2 + {b_y}^2)^{\frac{3}{2}}}dx$$

- Para la placa sobre el eje y:

$$E_i = k\int_{L_{inicial}}^{L_{final}} \frac{\lambda}{r^2}\cdot \cos\theta dy\ , \qquad E_j = k\int_{L_{inicial}}^{L_{final}} \frac{\lambda}{r^2}\cdot \sin\theta dy$$

$$E_i = k\int_{L_{inicial}}^{L_{final}} \frac{\lambda}{\sqrt{{a_x}^2 + (b_y - y)^2}^{\,2}}\cdot\frac{a_x}{\sqrt{{a_x}^2 + (b_y - y)^2}}dy\ , \qquad E_j = k\int_{L_{inicial}}^{L_{final}} \frac{\lambda}{\sqrt{{a_x}^2 + (b_y - y)^2}^{\,2}}\cdot\frac{(b_y - y)}{\sqrt{{a_x}^2 + (b_y - y)^2}}dy$$

$$E_i = k\int_{L_{inicial}}^{L_{final}} \frac{\lambda a_x}{({a_x}^2 + (b_y - y)^2)^{\frac{3}{2}}}dy\ , \qquad E_j = k\int_{L_{inicial}}^{L_{final}} \frac{\lambda(b_y - y)}{({a_x}^2 + (b_y - y)^2)^2}dy$$

Ecuaciones y funciones

$$E_i = k\lambda\left[\frac{1}{((a_x - x)^2 + b_y{}^2)^{\frac{1}{2}}}\right]_{L_{inicial}}^{L_{final}}, \qquad E_j = k\lambda\left[\frac{(x - a_x)}{b_y((a_x - x)^2 + b_y{}^2)^{\frac{1}{2}}}\right]_{L_{inicial}}^{L_{final}}$$

$$E_i = k\lambda\left[\frac{(y - b_y)}{a_x(a_x{}^2 + (b_y - y)^2)^{\frac{1}{2}}}\right]_{L_{inicial}}^{L_{final}}, \qquad E_j = k\lambda y\left[\frac{1}{(a_x{}^2 + (b_y - y)^2)^{\frac{1}{2}}}\right]_{L_{inicial}}^{L_{final}}$$

Ecuaciones y funciones

Funciones de física:

$r = \sqrt{\Delta x^2 + \Delta y^2}$ → Calcular distancia

$E = \frac{kq}{r^2}$ (cargas positivas)

$E = \frac{-kq}{r^2}$ (cargas negativas)

→ Calcular campo eléctrico positivo o negativo.

$E_x = E\frac{\Delta x}{r}$

$E_y = E\frac{\Delta y}{r}$

→ Componentes en “x” y “y” de campo eléctrico.

$E_{xtotal} = \sum E_x$

$E_{ytotal} = \sum E_y$

→ Sumatoria campo eléctrico en “x” y “y”.

$E_{total} = \sqrt{E_{xtotal}^{\;2} + E_{ytotal}^{\;2}}$

→ Campo eléctrico total.

Funciones de computación:

1. **linspace** → crear espacio de puntos con separación equidistante.
   a. valor mínimo, valor máximo y particiones
2. **meshgrid** → genera matrices para coordenadas.
3. **quiver** → gráfica matriz de vectores.

Código

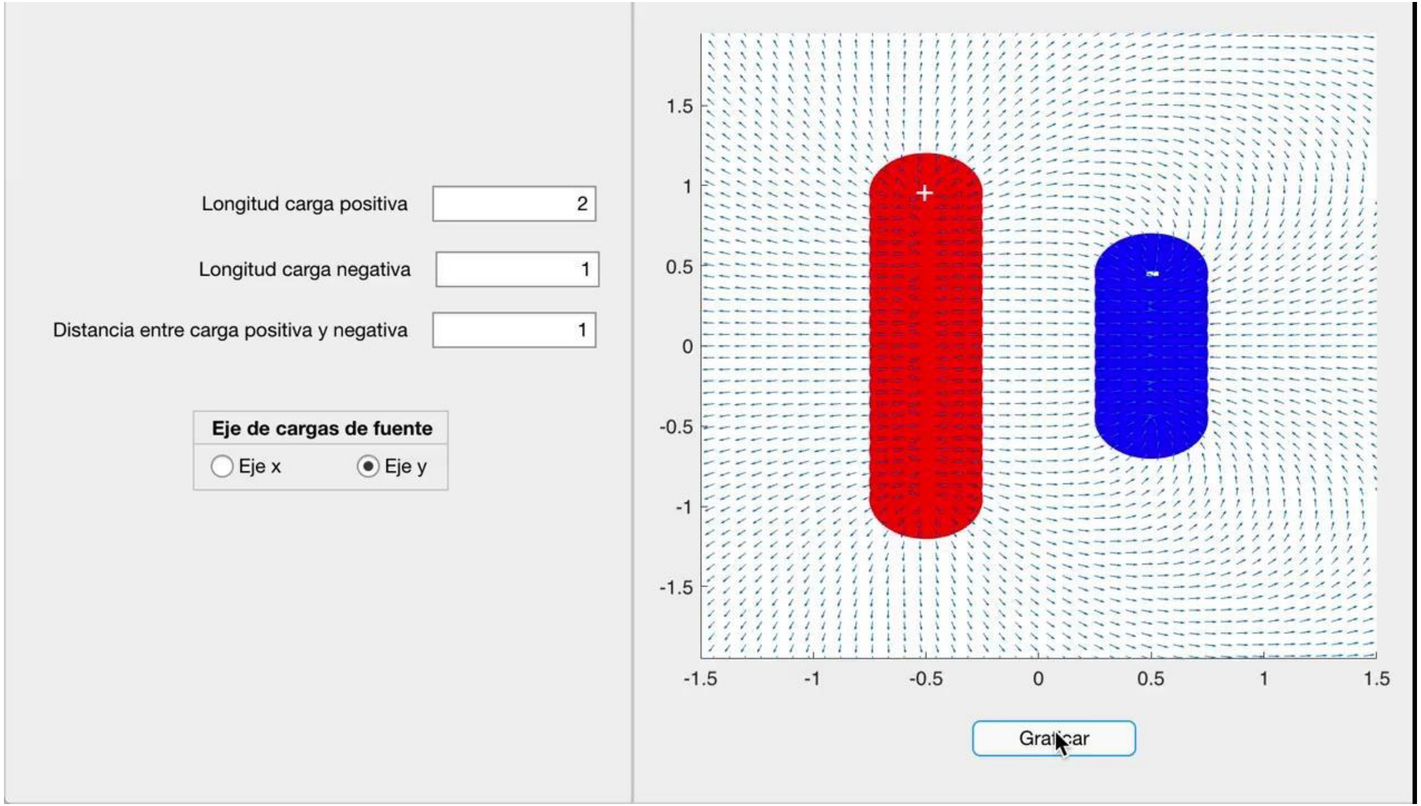
```
function [Et, Ex, Ey, tx, ty] = getVectores(~, xPPos, yPPos, xNPos, yNPos, lCP, lCN, n)
    pts = 300;
    x =linspace(xPPos(1,lCP/n)-10, xNPos(1,1)+10, pts);
    y =linspace(yNPos(1,1)-10, yPPos(1,lCP/n)+10, pts);
    [tx,ty] = meshgrid(x,y); % Definir matriz de vectores

    k =  1/(4*pi*8.54e-12); % constante k
    q =  1.6e-19; % Valor de cargas fuente
    Ex = zeros(pts);
    Ey = zeros(pts);
    Et = zeros(pts);

    for contx = 1:pts
        for conty = 1:pts
            EpxT = 0;
            EpyT = 0;
            EnxT = 0;
            EnyT = 0;
            for i = 1:lCP/n
                r = sqrt((tx(contx,conty) - xPPos(1,i))^2 + (ty(contx,conty) - yPPos(1,i))^2);
                Ec = k*q/r^2;
                EpxT = EpxT + Ec*(tx(contx,conty) - xPPos(1,i))/r;
                EpyT = EpyT + Ec*(ty(contx,conty) - yPPos(1,i))/r;
            end
            for i = 1:lCN/n
                r = sqrt((tx(contx,conty) - xNPos(1,i))^2 + (ty(contx,conty) - yNPos(1,i))^2);
                Ec = -k*q/r^2;
                EnxT = EnxT + Ec*(tx(contx,conty) - xNPos(1,i))/r;
                EnyT = EnyT + Ec*(ty(contx,conty) - yNPos(1,i))/r;
            end
            Ex(contx, conty) = EpxT + EnxT;
            Ey(contx, conty) = EpyT + EnyT;
            Et(contx,conty) = sqrt(Ex(contx, conty)^2 + Ey(contx, conty)^2);
        end
    end
end

function plotVectores(~, Et, Ex, Ey, tx, ty, grafica)
    hold(grafica, "on")
    quiver(grafica, tx, ty, Ex./Et, Ey./Et, 'AutoScaleFactor',0.5)
end
```

Resultados - Casos



Conclusión - Entregable 3

Durante la realización del programa enfocado a la entrega 3 se presentaron algunas dificultades relacionadas a la aplicación de integrales para poder obtener el campo eléctrico. Debido a eso, la estrategia utilizada fue implementar el algoritmo ya elaborado de la entrega 2 poder representar las cargas infinitas de las placas que lo componen, al igual que los cálculos pertinentes con respecto al al campo eléctrico

Conclusión General