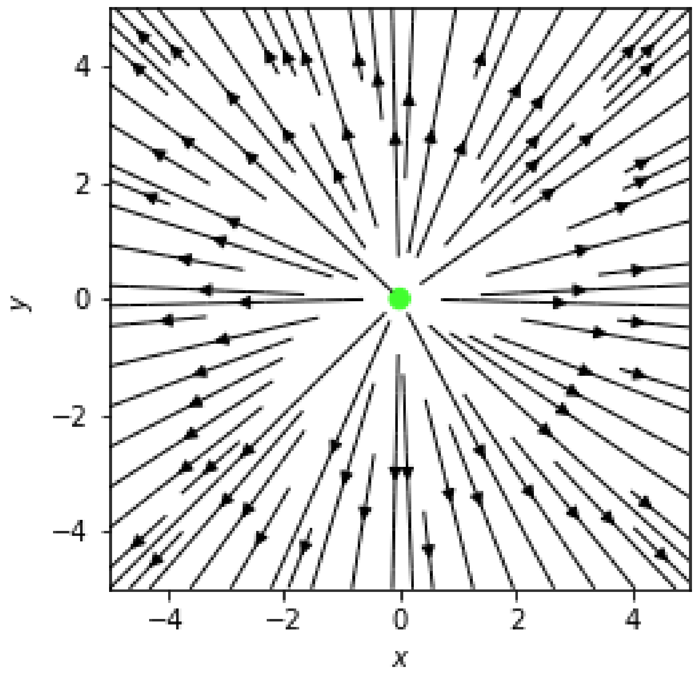
Assignment4

February 13, 2018

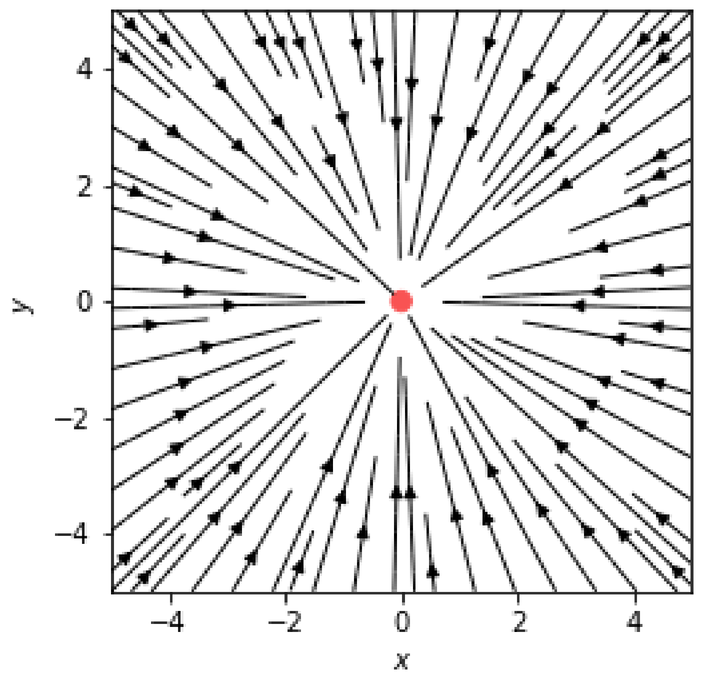
```
In [54]: #Exercise 1
         import matplotlib.pyplot as plt
         import numpy as np
         np.seterr(divide='ignore', invalid='ignore')
         from matplotlib.patches import Circle

         Q = 10**(-9)

         def E(q, r0, x, y):
             """Return the electric field vector E=(Ex,Ey) due to charge q at r0."""
             den = np.hypot(x-r0[0], y-r0[1])**3
             K = 9*10**9
             return q * K * (x - r0[0]) / den, q * K * (y - r0[1]) / den
         ny=101
         nx=101
         x = np.linspace(-5.0, 5.0, ny)
         y = np.linspace(-5.0, 5.0, nx)
         X, Y = np.meshgrid(x, y)

         charges = []
         charges.append((Q,(0.0, 0.0)))

         # Electric field vector, E=(Ex, Ey), as separate components
         Ex, Ey = np.zeros((ny, nx)), np.zeros((ny, nx))
         for charge in charges:
             ex, ey = E(*charge, x=X, y=Y)
             Ex += ex
             Ey += ey

         fig = plt.figure()
         ax = fig.add_subplot(111)

         plt.streamplot(x, y, Ex, Ey, color="black", linewidth=1,
                        density=1, arrowstyle='-|>', arrowsize=1)

         # Add filled circles for the charges themselves
         charge_colors = {True: '#41FE2E', False: '#F95353'}
```

```
        for q, pos in charges:
            ax.add_artist(Circle(pos, 0.17, color=charge_colors[q>0]))

        ax.set_xlabel('$x$')
        ax.set_ylabel('$y$')
        ax.set_xlim(-5,5)
        ax.set_ylim(-5,5)
        ax.set_aspect('equal')
        plt.show()
```



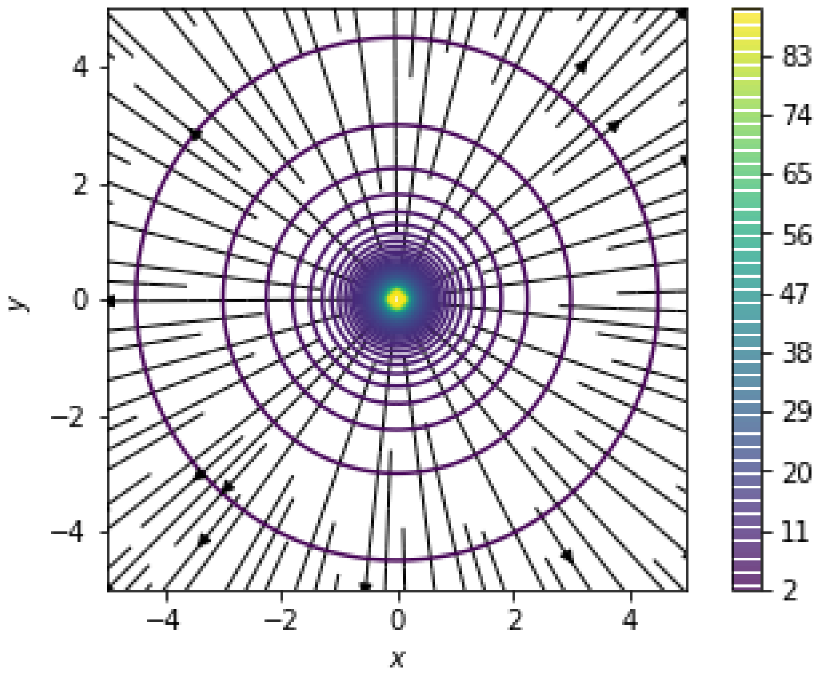
```
In [55]: #Exercise 1
         import matplotlib.pyplot as plt
         import numpy as np
         np.seterr(divide='ignore', invalid='ignore')
         from matplotlib.patches import Circle

         Q = -1*10**(-9)

         def E(q, r0, x, y):
             """Return the electric field vector E=(Ex,Ey) due to charge q at r0."""
             den = np.hypot(x-r0[0], y-r0[1])**3
             K = 9*10**9
             return q * K * (x - r0[0]) / den, q * K * (y - r0[1]) / den
         ny=101
         nx=101
```

```
x = np.linspace(-5.0, 5.0, ny)
y = np.linspace(-5.0, 5.0, nx)
X, Y = np.meshgrid(x, y)

charges = []
charges.append((Q,(0.0, 0.0)))

# Electric field vector, E=(Ex, Ey), as separate components
Ex, Ey = np.zeros((ny, nx)), np.zeros((ny, nx))
for charge in charges:
    ex, ey = E(*charge, x=X, y=Y)
    Ex += ex
    Ey += ey

fig = plt.figure()
ax = fig.add_subplot(111)

ax.streamplot(x, y, Ex, Ey, color="black", linewidth=1,
              density=1, arrowstyle='-|>', arrowsize=1)

# Add filled circles for the charges themselves
charge_colors = {True: '#41FE2E', False: '#F95353'}
for q, pos in charges:
    ax.add_artist(Circle(pos, 0.17, color=charge_colors[q>0]))

ax.set_xlabel('$x$')
ax.set_ylabel('$y$')
ax.set_xlim(-5,5)
ax.set_ylim(-5,5)
ax.set_aspect('equal')
plt.show()
```



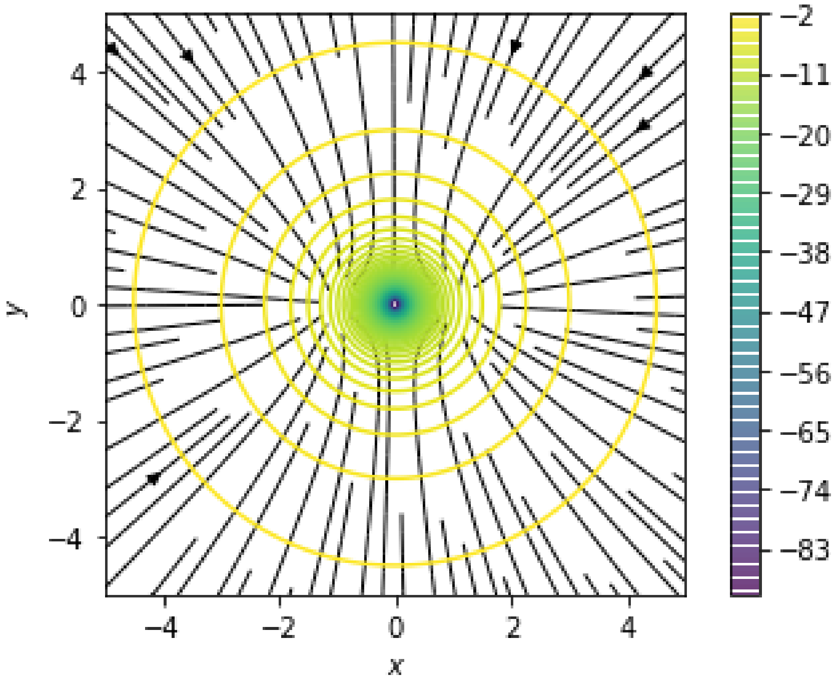
```
In [3]: #Exercise 2
        import matplotlib.pyplot as plt
        import numpy as np
        np.seterr(divide='ignore', invalid='ignore')
        ny=101
        nx=101
        K = 9*10**9
        Q = 10**(-9)

        ##
        def E(q, r0, x, y):
            """Return the electric field vector E=(Ex,Ey) due to charge q at r0."""
            den = np.hypot(x-r0[0], y-r0[1])**3
            K = 9*10**9
            return q * K * (x - r0[0]) / den, q * K * (y - r0[1]) / den
        ny=101
        nx=101
        x = np.linspace(-10.0, 10.0, ny)
        y = np.linspace(-10.0, 10.0, nx)
        X, Y = np.meshgrid(x, y)

        charges = []
        charges.append((Q,(0.0,0.0)))
```

```
# Electric field vector, E=(Ex, Ey), as separate components
Ex, Ey = np.zeros((ny, nx)), np.zeros((ny, nx))
for charge in charges:
    ex, ey = E(*charge, x=X, y=Y)
    Ex += ex
    Ey += ey

fig = plt.figure()
ax = fig.add_subplot(111)

plt.streamplot(x, y, Ex, Ey, color="black", linewidth=1,
              density=2, arrowstyle='-|>', arrowsize=1)
##


x = np.linspace(-5.0, 5.0, ny)
y = np.linspace(-5.0, 5.0, nx)
X, Y = np.meshgrid(x, y)
r0 = [0.0,0.0]
X = X+r0[0]
Y = Y+r0[1]

V = Q * K / np.sqrt(X**2 + Y**2)

axim = plt.contour(X,Y,V, 100)
fig.colorbar(axim)

ax.set_xlabel('$x$')
ax.set_ylabel('$y$')
ax.set_xlim(-5,5)
ax.set_ylim(-5,5)
ax.set_aspect('equal')

plt.show()
```



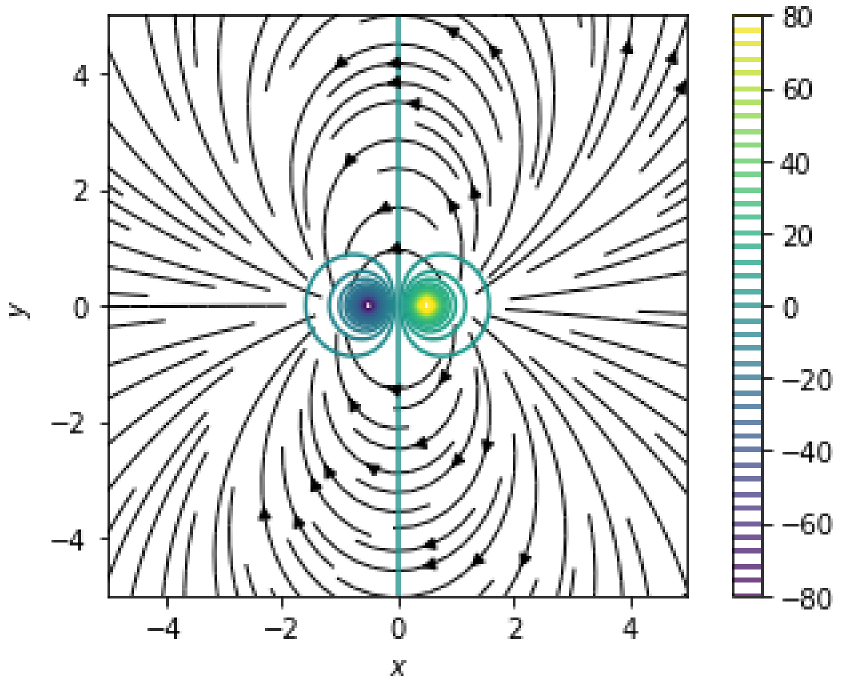
```
In [59]: #Exercise 2
         import matplotlib.pyplot as plt
         import numpy as np
         np.seterr(divide='ignore', invalid='ignore')
         ny=101
         nx=101
         K = 9*10**9
         Q = -1*10**(-9)

         ##
         def E(q, r0, x, y):
             """Return the electric field vector E=(Ex,Ey) due to charge q at r0."""
             den = np.hypot(x-r0[0], y-r0[1])**3
             K = 9*10**9
             return q * K * (x - r0[0]) / den, q * K * (y - r0[1]) / den
         ny=101
         nx=101
         x = np.linspace(-10.0, 10.0, ny)
         y = np.linspace(-10.0, 10.0, nx)
         X, Y = np.meshgrid(x, y)

         charges = []
         charges.append((Q1,(-1,0.0)))
         charges.append((Q,(1,0.0)))
```

```
# Electric field vector, E=(Ex, Ey), as separate components
Ex, Ey = np.zeros((ny, nx)), np.zeros((ny, nx))
for charge in charges:
    ex, ey = E(*charge, x=X, y=Y)
    Ex += ex
    Ey += ey

fig = plt.figure()
ax = fig.add_subplot(111)

plt.streamplot(x, y, Ex, Ey, color="black", linewidth=1,
              density=2, arrowstyle='-|>', arrowsize=1)
##

x = np.linspace(-5.0, 5.0, ny)
y = np.linspace(-5.0, 5.0, nx)
X, Y = np.meshgrid(x, y)
r0 = [0.0,0.0]
X = X+r0[0]
Y = Y+r0[1]

V = Q * K / np.sqrt(X**2 + Y**2)

axim = plt.contour(X,Y,V, 100)
fig.colorbar(axim)

ax.set_xlabel('$x$')
ax.set_ylabel('$y$')
ax.set_xlim(-5,5)
ax.set_ylim(-5,5)
ax.set_aspect('equal')

plt.show()
```



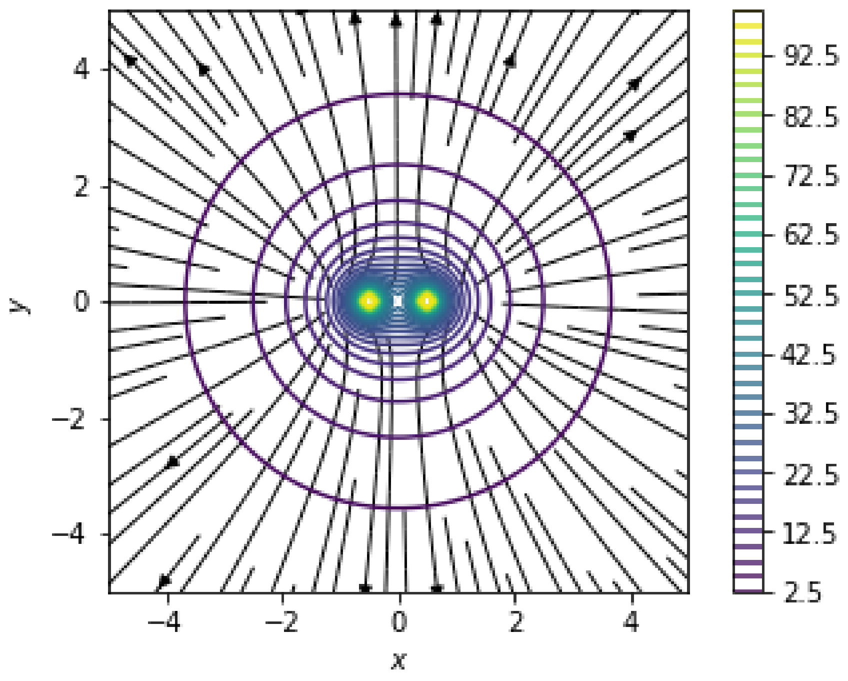
```
In [2]: #Exercise 3
        import matplotlib.pyplot as plt
        import numpy as np
        np.seterr(divide='ignore', invalid='ignore')
        ny=101
        nx=101
        K = 9*10**9

        Q = 10**(-9)
        Q1 = -1*10**(-9)

        ##
        def E(q, r0, x, y):
            """Return the electric field vector E=(Ex,Ey) due to charge q at r0."""
            den = np.hypot(x-r0[0], y-r0[1])**3
            K = 9*10**9
            return q * K * (x - r0[0]) / den, q * K * (y - r0[1]) / den
        ny=101
        nx=101
        x = np.linspace(-10.0, 10.0, ny)
        y = np.linspace(-10.0, 10.0, nx)
        X, Y = np.meshgrid(x, y)

        charges = []
```

```
charges.append((Q1,(-1,0.0)))
charges.append((Q,(1,0.0)))

# Electric field vector, E=(Ex, Ey), as separate components
Ex, Ey = np.zeros((ny, nx)), np.zeros((ny, nx))
for charge in charges:
    ex, ey = E(*charge, x=X, y=Y)
    Ex += ex
    Ey += ey

fig = plt.figure()
ax = fig.add_subplot(111)

plt.streamplot(x, y, Ex, Ey, color="black", linewidth=1,
              density=2, arrowstyle='-|>', arrowsize=1)
##

x = np.linspace(-5.0, 5.0, ny)
y = np.linspace(-5.0, 5.0, nx)
X, Y = np.meshgrid(x, y)
r0 = [-1,0.0]
X = X+r0[0]
Y = Y+r0[1]

V = Q * K / np.sqrt(X**2 + Y**2)


x1 = np.linspace(-5.0, 5.0, ny)
y1 = np.linspace(-5.0, 5.0, nx)
X1, Y1 = np.meshgrid(x1, y1)

r1 = [1,0.0]
X1 = X+r1[0]
Y1 = Y+r1[1]

V1 = Q1 * K / np.sqrt(X1**2 + Y1**2)

axim = plt.contour((X/2)+(X1/2), (Y/2)+(Y1/2), V+V1, 50)
fig.colorbar(axim)

ax.set_xlabel('$x$')
ax.set_ylabel('$y$')
ax.set_xlim(-5,5)
ax.set_ylim(-5,5)
ax.set_aspect('equal')

plt.show()
```



```
In [49]: #Exercise 3
         import matplotlib.pyplot as plt
         import numpy as np
         np.seterr(divide='ignore', invalid='ignore')
         ny=101
         nx=101
         K = 9*10**9

         Q = 10**(-9)
         Q1 = 1*10**(-9)

         ##
         def E(q, r0, x, y):
             """Return the electric field vector E=(Ex,Ey) due to charge q at r0."""
             den = np.hypot(x-r0[0], y-r0[1])**3
             K = 9*10**9
             return q * K * (x - r0[0]) / den, q * K * (y - r0[1]) / den
         ny=101
         nx=101
         x = np.linspace(-10.0, 10.0, ny)
         y = np.linspace(-10.0, 10.0, nx)
         X, Y = np.meshgrid(x, y)

         charges = []
```

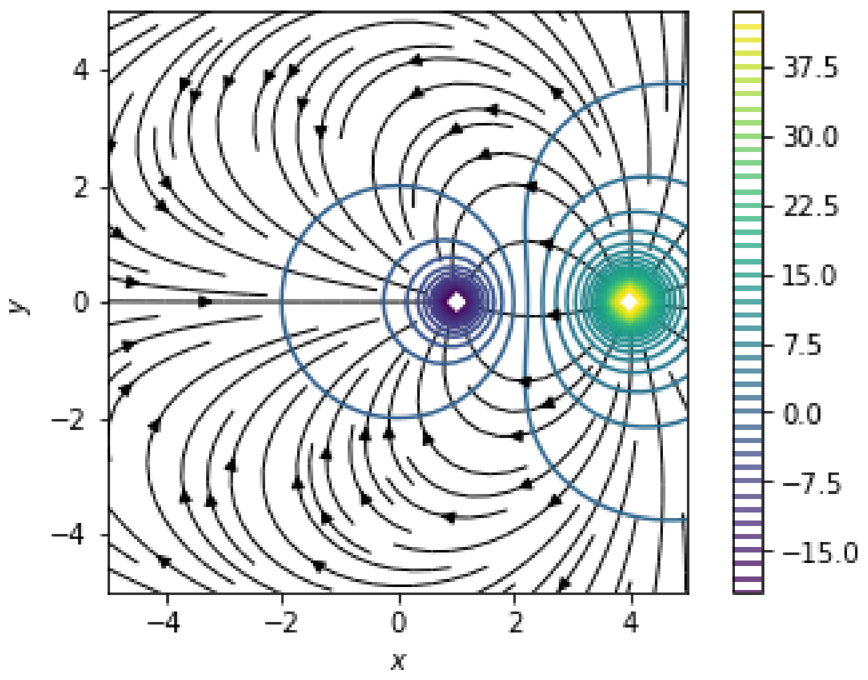
```
charges.append((Q,(-1,0.0)))
charges.append((Q1,(1,0.0)))

# Electric field vector, E=(Ex, Ey), as separate components
Ex, Ey = np.zeros((ny, nx)), np.zeros((ny, nx))
for charge in charges:
    ex, ey = E(*charge, x=X, y=Y)
    Ex += ex
    Ey += ey

fig = plt.figure()
ax = fig.add_subplot(111)

plt.streamplot(x, y, Ex, Ey, color="black", linewidth=1,
              density=2, arrowstyle='-|>', arrowsize=1)
##

x = np.linspace(-5.0, 5.0, ny)
y = np.linspace(-5.0, 5.0, nx)
X, Y = np.meshgrid(x, y)
r0 = [-1,0.0]
X = X+r0[0]
Y = Y+r0[1]

V = Q * K / np.sqrt(X**2 + Y**2)


x1 = np.linspace(-5.0, 5.0, ny)
y1 = np.linspace(-5.0, 5.0, nx)
X1, Y1 = np.meshgrid(x1, y1)

r1 = [1,0.0]
X1 = X+r1[0]
Y1 = Y+r1[1]

V1 = Q1 * K / np.sqrt(X1**2 + Y1**2)

axim = plt.contour((X/2)+(X1/2), (Y/2)+(Y1/2), V+V1, 45)
fig.colorbar(axim)

ax.set_xlabel('$x$')
ax.set_ylabel('$y$')
ax.set_xlim(-5,5)
ax.set_ylim(-5,5)
ax.set_aspect('equal')

plt.show()
```



```
In [6]: #Exercise 3.3
        import matplotlib.pyplot as plt
        import numpy as np
        np.seterr(divide='ignore', invalid='ignore')
        ny=101
        nx=101
        K = 9*10**9

        Q = 10**(-9)
        Q1 = (-1*10**(-9))/ 2
        ##


        def E(q, r0, x, y):
            """Return the electric field vector E=(Ex,Ey) due to charge q at r0."""
            den = np.hypot(x-r0[0], y-r0[1])**3
            K = 9*10**9
            return q * K * (x - r0[0]) / den, q * K * (y - r0[1]) / den
        ny=101
        nx=101
        x = np.linspace(-10.0, 10.0, ny)
        y = np.linspace(-10.0, 10.0, nx)
        X, Y = np.meshgrid(x, y)
```

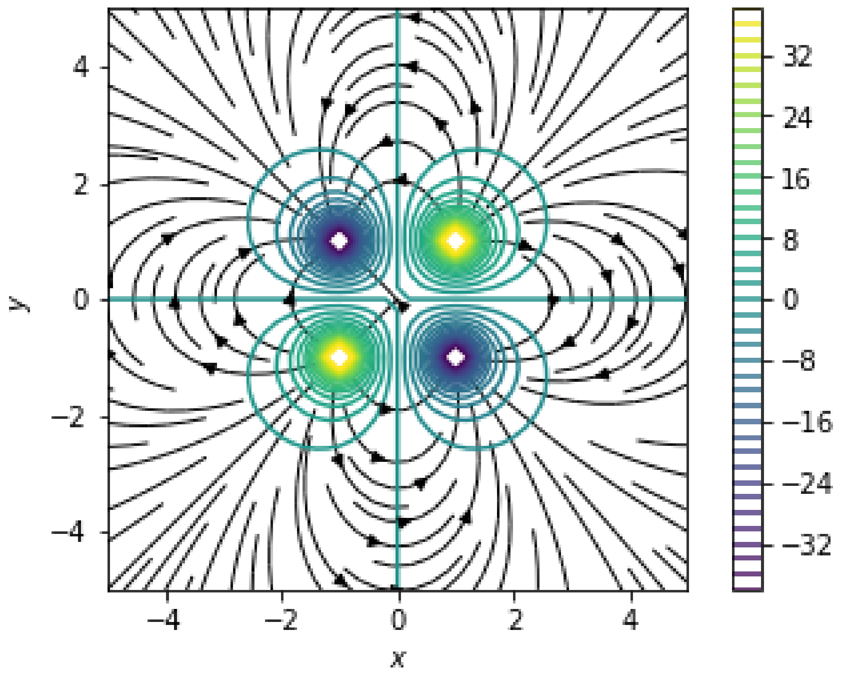
```
charges = []
charges.append((Q,(4.0,0.0)))
charges.append((Q1,(1.0,0.0)))

# Electric field vector, E=(Ex, Ey), as separate components
Ex, Ey = np.zeros((ny, nx)), np.zeros((ny, nx))
for charge in charges:
    ex, ey = E(*charge, x=X, y=Y)
    Ex += ex
    Ey += ey

fig = plt.figure()
ax = fig.add_subplot(111)

plt.streamplot(x, y, Ex, Ey, color="black", linewidth=1,
              density=2, arrowstyle='-|>', arrowsize=1)
##


x = np.linspace(-10.0, 10.0, ny)
y = np.linspace(-10.0, 10.0, nx)
X, Y = np.meshgrid(x, y)
r0 = [4.0,0.0]
X = X+r0[0]
Y = Y+r0[1]

V = Q * K / np.sqrt(X**2 + Y**2)


x1 = np.linspace(-10.0, 10.0, ny)
y1 = np.linspace(-10.0, 10.0, nx)
X1, Y1 = np.meshgrid(x1, y1)

r1 = [3.0,0.0]
X1 = X+r1[0]
Y1 = Y+r1[1]

V1 = Q1 * K / np.sqrt(X1**2 + Y1**2)


axim = plt.contour(X+4, Y, V+V1, 50)
fig.colorbar(axim)

ax.set_xlabel('$x$')
ax.set_ylabel('$y$')
ax.set_xlim(-5,5)
ax.set_ylim(-5,5)
ax.set_aspect('equal')
```

```
plt.show()
```



```
In [57]: #Exercise 3.4
         import matplotlib.pyplot as plt
         import numpy as np
         np.seterr(divide='ignore', invalid='ignore')
         ny=101
         nx=101
         K = 9*10**9
         Q = 10**(-9)

         ##
         def E(q, r0, x, y):
             """Return the electric field vector E=(Ex,Ey) due to charge q at r0."""
             den = np.hypot(x-r0[0], y-r0[1])**3
             K = 9*10**9
             return q * K * (x - r0[0]) / den, q * K * (y - r0[1]) / den
         ny=101
         nx=101
         x = np.linspace(-10.0, 10.0, ny)
         y = np.linspace(-10.0, 10.0, nx)
         X, Y = np.meshgrid(x, y)

         charges = []
         charges.append((Q,(1.0, 1.0)))
charges.append((-Q,(-1.0, 1.0)))
charges.append((-Q,(1.0, -1.0)))
charges.append((Q,(-1.0, -1.0)))

# Electric field vector, E=(Ex, Ey), as separate components
Ex, Ey = np.zeros((ny, nx)), np.zeros((ny, nx))
for charge in charges:
    ex, ey = E(*charge, x=X, y=Y)
    Ex += ex
    Ey += ey

fig = plt.figure()
ax = fig.add_subplot(111)

plt.streamplot(x, y, Ex, Ey, color="black", linewidth=1,
              density=2, arrowstyle='-|>', arrowsize=1)
##


#Charge 1  +1nC at (+1.0,+1.0)
Q = 10**(-9)
x = np.linspace(-10.0, 10.0, ny)
y = np.linspace(-10.0, 10.0, nx)
X, Y = np.meshgrid(x, y)

r0 = [1.0,1.0]
X = X+r0[0]
Y = Y+r0[1]

V = Q * K / np.sqrt(X**2 + Y**2)

#Charge 2   1nC at (1.0,+1.0)
Q1 = -1*10**(-9)
x1 = np.linspace(-10.0, 10.0, ny)
y1 = np.linspace(-10.0, 10.0, nx)
X1, Y1 = np.meshgrid(x1, y1)

r1 = [-1.0,1.0]
X1 = X1+r1[0]
Y1 = Y1+r1[1]

V1 = Q1 * K / np.sqrt(X1**2 + Y1**2)

#Charge 3 1nC at (1.0,1.0)
Q2 = 1*10**(-9)
x2 = np.linspace(-10.0, 10.0, ny)
y2 = np.linspace(-10.0, 10.0, nx)
X2, Y2 = np.meshgrid(x2, y2)
```

```
r2 = [-1.0,-1.0]
X2 = X2+r2[0]
Y2 = Y2+r2[1]

V2 = Q2 * K / np.sqrt(X2**2 + Y2**2)

#Charge 4 1nC at  (+1.0,1.0)
Q3 = -1*10**(-9)
x3 = np.linspace(-10.0, 10.0, ny)
y3 = np.linspace(-10.0, 10.0, nx)
X3, Y3 = np.meshgrid(x3, y3)

r3 = [1.0,-1.0]
X3 = X3+r3[0]
Y3 = Y3+r3[1]

V3 = Q3 * K / np.sqrt(X3**2 + Y3**2)


axim = plt.contour((X+X1+X2+X3)/4, (Y+Y1+Y2+Y3)/4, V+V1+V2+V3, 50)
fig.colorbar(axim)

ax.set_xlabel('$x$')
ax.set_ylabel('$y$')
ax.set_xlim(-5,5)
ax.set_ylim(-5,5)
ax.set_aspect('equal')
plt.show()
```


4
2
0
−2
−4
y
−4
−2
0
2
4
x
32
24
16
8
0
−8
−16
−24
−32