TP numerique carte de champ

December 9, 2021

L'objectif de ce TP est d'utiliser une approche numérique pour représenter des cartes de champ électrostatique et de les confronter à nos connaissances sur le champ électrostatique. Reprenez l'exercice du TD Cartes de champ, nous allons calculer numériquement les cartes de champs présentes dans l'exercice. Pour cela vous avez besoin d'utiliser deux librairies numpy et matplotlib.pyplot

[ ]:
```
import numpy as np
import matplotlib.pyplot as plt
```

Vous devez ensuite définir en début de programme les variables qui vous seront utiles pour faire vos calculs numériques. A savoir ici les charges et positions de chaque point des distributions discrètes de charges.

[ ]:
```
# définissez pour chaque point de la distribution sa charge et sa position

q_1 =
x_1 =
y_1 =

# ... etc
```

Vous tracerez votre carte de champ dans un plan (Oxy) donc on peut chercher à définir les positions sur lesquelles on va calculer les différentes grandeurs du champ électrostatique. On va commencer par définir les axes x et y.

[ ]:
```
#en utilisant la fonction arange de numpy définir les bornes et l'écartement␣
 ↪entre chaque point des axes x et y qui seront numérisésous formes de listes.
x = np.arange('début','fin','pas')
y = np.arange('début','fin','pas')
```

Puis on va faire un maillage du plan en considérant chaque point ayant pour coordonnées deux éléments des listes x et y. Les premières grandeurs que l'on va calculer sont les deux coordonnées X et Y de chaque point.

[ ]:
```
# définir deux tableaux X et Y où chaque éléments du tableau correspond à la␣
 ↪coordonée selon x ou y du point considéré, on pourra utiliser la fonction␣
 ↪meshgrid de numpy
X, Y = np.meshgrid(x, y)
print(X)
print(Y)
```

On va calculer ensuite le champ électrostatique crée par chaque sources ponctuelles prises séparement

[ ]:
```
# définir un tableau pour chaque coordonée Ex et Ey du champ créé par chaque␣
↪sources ponctuelles à l'aide d'opération sur les tableaux numpy

Ex_1 = 'une fonction de' X Y x_1 y_1 q_1
Ey_1 = 'une fonction de' X Y x_1 y_1 q_1
```

Enfin on utilise le principe de superposition pour obtenir le champ de la distribution entière.

[ ]:
```
# calculer le champ de la distribution entière

Ex = 'somme de tous les' Ex_i
Ey = 'somme de tous les' Ey_i
```

Une fois le champ calculé on peut chercher à le représenter soit à l'aide de vecteur en chaque point

[ ]:
```
# utiliser la fonction quiver de matplotlib.pyplot pour représenter le champ␣
↪électrostatique

plt.quiver(X,Y,Ex,Ey)
```

soit à l'aide de ligne de champs

[ ]:
```
# utiliser la fonction streamplot de matplotlib.pyplot pour représenter les␣
↪lignes de champ

plt.streamplot(x,y,Ex,Ey)
```

en bonus on pourrait en suivant la même démarche calculer le potentiel électrostatique et le représenter à l'aide de la fonction contour de matplotlib.pyplot

[ ]:
```

```