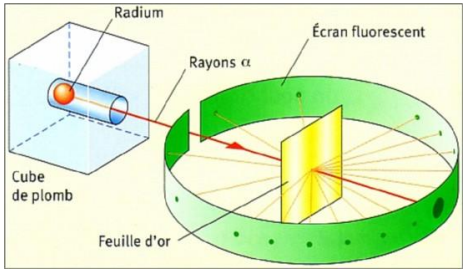
Méthodes numériques

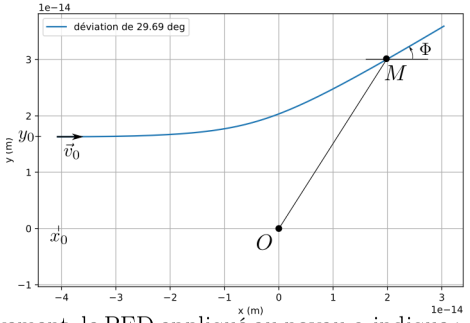
Trajectoire d'une particule dans un champ de force centrale

On s'intéresse à une expérience de type Rutherford : un noyau α (composé de deux protons et deux neutrons) est envoyé sur une fine feuille d'or. Il s'agit donc de modéliser la collision entre :

— le noyau α (point M, masse $m = 4 \times 1{,}67 \times 10^{-27}\,\text{kg}$, charge $q_1 = 2 \times 1{,}6 \times 10^{-19}\,\text{C}$) ;

— le noyau de l'atome d'or (à l'origine O, masse si importante qu'il ne bouge pas, charge $q_2 = Z \times 1{,}6 \times 10^{-19}\,\text{C}$ avec $Z = 79$).





En coordonnées polaires dans le plan du mouvement, le PFD appliqué au noyau α indique :

$$m\frac{\mathrm{d}^2\overrightarrow{OM}}{\mathrm{d}t^2} = \frac{q_1 q_2}{4\pi\varepsilon_0\,(OM)^2}\,\vec{e}_r.$$

On utilise en plus :

$$\vec{e}_r = \frac{\overrightarrow{OM}}{OM} = \frac{x\,\vec{e}_x + y\,\vec{e}_y}{OM} \quad \text{et} \quad OM = \sqrt{x^2+y^2},$$

pour avoir les équations du mouvement :

$$\text{sur } \vec{e}_x \ : \ \boxed{\ddot{x} = \frac{q_1 q_2}{4\pi\varepsilon_0\, m}\,\frac{x}{(\sqrt{x^2+y^2})^3},} \qquad \text{sur } \vec{e}_y \ : \ \boxed{\ddot{y} = \frac{q_1 q_2}{4\pi\varepsilon_0\, m}\,\frac{y}{(\sqrt{x^2+y^2})^3}.} \qquad (1)$$

Dans la suite on posera $C = \dfrac{q_1 q_2}{4\pi\varepsilon_0\, m}$ pour alléger les notations.

a/ Reformulation de l'équation en une équation d'ordre 1

Dans un cours précédent (le ressortir) nous avons vu comment résoudre une équation d'ordre 2 à l'aide de la fonction `odeint`. Celle-ci a une syntaxe particulière, qui oblige à transformer l'équation d'ordre 2 en une équation d'ordre 1.

Pour cela on pose $Y = \begin{pmatrix} x \\ y \\ \dot{x} \\ \dot{y} \end{pmatrix}$. On note $Y_0 = x$, $Y_1 = y$, $Y_2 = \dot{x}$ et $Y_3 = \dot{y}$ les quatre composantes.

En python, `Y[0]` contiendra les valeurs de $x(t)$, `Y[1]` celles de $y(t)$, etc.

1 - Dériver Y pour trouver l'équation différentielle qu'il vérifie : $\dfrac{\mathrm{d}Y}{\mathrm{d}t} = \begin{pmatrix} \dot{x} \\ \dot{y} \\ \ddot{x} \\ \ddot{y} \end{pmatrix} = \ldots$

Vous devez parvenir à l'équation du premier ordre sur Y suivante :

$$\boxed{\frac{\mathrm{d}Y}{\mathrm{d}t} = F(Y) \quad \text{avec} \quad F(Y) = \begin{pmatrix} Y_2 \\ Y_3 \\ \dfrac{CY_0}{(\sqrt{Y_0^2+Y_1^2})^3} \\ \dfrac{CY_1}{(\sqrt{Y_0^2+Y_1^2})^3} \end{pmatrix}.} \qquad (2)$$

Bilan : on est passé des équations d'ordre 2 sur x et y à l'équation ci-dessus, d'ordre 1 sur le vecteur Y.

b/ Utilisation de odeint

On utilise la fonction `odeint` de la librairie `scipy.integrate` dont voici la syntaxe :

```
import scipy.integrate as sp
Y_sol = sp.odeint(F, Y_ini, t)
```

Paramètres de odeint :

- `F` : fonction du type `F(Y,t)`, qui retourne la valeur de la dérivée de `Y` à l'instant `t`.
- `Y_ini` : valeur initiale de `Y`. Ici il a donc quatre composantes.
- `t` est un tableau qui contient les instants auxquels la solution sera calculée.

Objets retournés par odeint :

- Après exécution de `Y_sol = sp.odeint(F, Y_ini, t)`, `Y_sol` est une matrice dont la première colonne (`Y_sol[:,0]`) contient les valeurs de $Y_0(t)$ (donc dans notre exemple, de $x(t)$), la seconde colonne (`Y_sol[:,1]`) les valeurs de $Y_1(t)$ (donc dans notre exemple, de $y(t)$), etc.

2 - Ouvrez Capytale et aller sur le fichier `e529-1490298`.
Compléter le code de la fonction `F(Y,t)` pour qu'il soit en accord avec la relation (2).
Compléter ensuite la ligne `t =` en créant un tableau de `1000` instants t_i allant de `0` à `4*T0` (rappel : `t = np.linspace(a,b,N)` crée un tel tableau allant de `a` à `b` avec `1000` points).
Compléter ensuite la ligne `Y_ini =` en lui affectant le quadruplet `(x0, y0, v0, 0)`.
Enfin, compléter la ligne `Y_sol =` en utilisant la fonction `sp.odeint` avec les bons arguments.

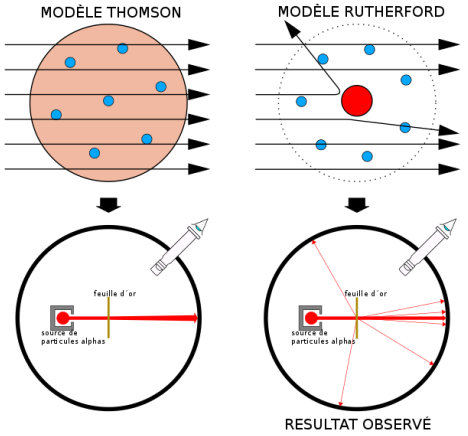
3 - Tracer ensuite la trajectoire $x(t)$, $y(t)$ obtenue à l'aide de `plt.plot`.

Pour matérialiser la position de l'atome central, il y a déjà une ligne de code qui trace un cercle de rayon `R0`.

4 - Explorer l'influence de la valeur initiale $y_0 = y(t = 0)$ (aussi appelée paramètre d'impact) sur l'allure de la trajectoire. On pourra tester `y0 = 2*R0`, ou `1*R0`, etc.

En particulier, donner la valeur de y_0 (en terme de nombre de fois R_0) à partir de laquelle le noyau α est dévié vers l'arrière.

Remarque : historiquement, ce type d'expérience a permis de valider le modèle de l'atome de Rutherford (avec un noyau chargé très petit) et de rejeter le modèle de l'atome de Thomson (atome sans noyau, où les électrons sont à l'intérieur d'un milieu chargé positivement). En effet, seul le modèle de Rutherford prédit des déviations vers l'arrière en nombre suffisant. Cf https://fr.wikipedia.org/wiki/Exp%C3%A9rience_de_Rutherford.



c/ Trajectoires dans le cas attractif

Dans cette sous-partie seulement, on prend un cas fictif où le noyau α est de charge $q_1 = -2 \times 1{,}6 \times 10^{-19}\,\mathrm{C}$ (négative, donc la force devient attractive comme pour la gravitation).

5 - Faire le changement nécessaire dans le code.

Prendre également `y0 = 5*R0`. Générer le tableau `t` jusqu'à `10*T0` (au lieu de juste `4*T0`).

Changer alors la valeur de `v0` : diminuer là d'un facteur 2 (écrire par exemple `v0=v0/2` une ligne après la définition de `v0`). Continuer à la diminuer jusqu'à obtenir un changement de nature de la trajectoire.

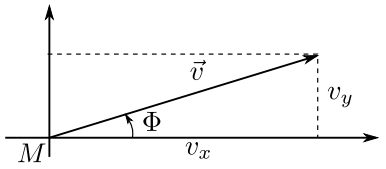
6 - Ajouter une ligne qui permet de calculer la valeur initiale de l'énergie mécanique de la charge : $E_m = \dfrac{1}{2}mv^2 + \dfrac{q_1 q_2}{4\pi\varepsilon_0\, r}$.

Vérifier alors que le signe de E_m est bien celui attendu pour une trajectoire non bornée, et pour une trajectoire bornée.

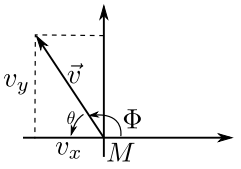
d/ Angle de déviation et exploration statistique

Pour la suite, prendre à nouveau une charge `q1 = 2 * 1.6e-19` pour le noyau α, et `v0 = (22e6*eV*2/m)**.5` comme vitesse initiale.

7 - On s'intéresse à l'angle de déviation du noyau α. Il s'agit de l'angle Φ sur la figure de début d'énoncé et sur la figure ci-dessous.



cas $v_x > 0$



cas $v_x < 0$

– Dans le cas où $v_x > 0$, montrer que $\Phi = \arctan\dfrac{v_y}{v_x}$.

– Dans le cas où $v_x < 0$, quel est l'angle donné par $\arctan\dfrac{v_y}{|v_x|}$? En déduire l'expression de Φ.

Écrire alors une fonction `deviation(vx,vy)` qui prend en argument les composantes v_x et v_y du vecteur vitesse et qui retourne la valeur de l'angle de déviation Φ (en degrés).

L'utiliser ensuite pour afficher la valeur de cette déviation : pour v_x et v_y on prendra les dernières valeurs des tableaux `xp` et `yp` (tableaux qui contiennent $\dot{x}$ et $\dot{y}$), auxquelles on accède par `xp[-1]` et `yp[-1]`.

8 - On tire aléatoirement des valeurs de y_0 comprises entre 0 et $10R_0$ à l'aide de `y0 = np.random.uniform(0,10*R0)`. Répéter ceci 1000 fois afin de construire une liste `liste_y0` de valeurs aléatoires de y_0.

9 - Tracer l'histogramme des valeurs tirées à l'aide de `plt.hist(liste_y0)`, dans une nouvelle figure.

On doit constater que `np.random.uniform` fait un tirage uniforme (pas de valeur privilégiée).

10 - Écrire une fonction `calcul_deviation(y0)` qui prend en argument une valeur initiale $y_0 = y(t = 0)$, calcule la trajectoire correspondante avec `odeint`, et retourne la valeur de l'angle de déviation. On réutilisera la fonction `deviation(vx,vy)` déjà écrite.

11 - Écrire alors une boucle qui remplit une liste `liste_deviation` qui contient les déviations associées à chaque valeurs y_0 de la liste `liste_y0`.

Tracer l'histogramme des déviations dans une nouvelle figure.

12 - Calculer la fraction de noyaux α qui sont déviés de plus de 90°.

13 - On peut éventuellement écrire une fonction qui calcule la distance minimale d'approche $r_{\min}$.

Remarque : pour des détails théoriques (angle de déviation, distance d'approche minimale), cf à la fin de la page https://femto-physique.fr/mecanique/forces-centrales.php.