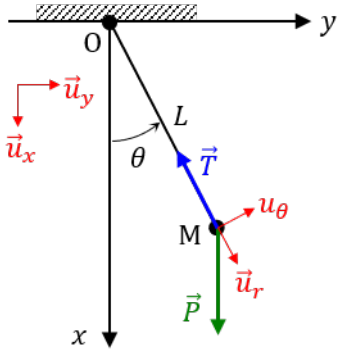
TP – Pendule simple

NOM :

On considère un pendule simple : une masse m supposée ponctuelle est attachée à l'extrémité d'un fil de longueur L et de masse négligeable. On note $\mathcal{R} = (O, \vec{u}_x, \vec{u}_y)$ le référentiel terrestre supposé galiléen et $\mathcal{B} = (O, \vec{u}_r, \vec{u}_\theta)$ la base polaire dans laquelle seront projetés les différents vecteurs.



Le PFD appliqué à la masse m dans $\mathcal{R}$ et projeté sur $\vec{u}_\theta$ donne :

$$\ddot{\theta} + \omega_0^2 \sin(\theta) = 0 \quad \text{avec :} \quad \omega_0 = \sqrt{\frac{g}{L}}$$

Aux petits angles, on obtient l'équation différentielle d'un oscillateur harmonique :

$$\ddot{\theta} + \omega_0^2 \, \theta(t) = 0$$

Dans ce TP, nous allons :

- apprendre à calculer numériquement la dérivée d'une fonction ;
- apprendre à résoudre numériquement une équation différentielle d'ordre 2 ;
- déterminer la valeur du champ de pesanteur à l'aide d'un pendule simple.

I) Conservation de l'énergie mécanique

1) Pointage vidéo

Par souci de temps, un pointage vidéo d'une dizaine d'oscillations d'un pendule simple a été réalisé en amont. Un fichier Python contient la position du pendule (x, y) en fonction du temps t.

L'expérience a été réalisée avec : $m = 200$ g et $L = 15$ cm.

🏠 Déterminer le lien entre θ, x et y.

💻 Compléter le code Python fourni et visualiser graphiquement $\theta(t)$.

2) Dérivée numérique

On souhaite obtenir numériquement la vitesse angulaire $\omega(t)$.

$$\omega(t) = \frac{d\theta}{dt} = \frac{\theta(t+dt) - \theta(t)}{(t+dt) - (t)} \quad \underset{\text{Discrétisation}}{\Rightarrow} \quad \omega[n] = \frac{\theta[n+1] - \theta[n]}{t[n+1] - t[n]}$$

Cette relation permet de calculer numériquement la dérivée $\omega[n]$, connaissant $\theta[n]$, $\theta[n+1]$, $t[n]$ et $t[n+1]$. On appelle N l'indice du dernier élément de θ et t. Attention, cette méthode ne permet pas de calculer $\omega[N]$ puisque $\theta[N+1]$ et $t[N+1]$ ne sont, par définition de N, pas définis.

Dans l'optique de travailler avec des listes qui font toutes la même taille, on pourra simplement ajouter une $N^{\text{ième}}$ valeur à la liste, identique à la précédente : $\omega[N] = \omega[N-1]$.

💻 Écrire une fonction `derivee(t, s)`, qui prend en argument deux arrays `t` et `s` et qui renvoie la dérivée de `s` par rapport à `t`. Recopier votre fonction ci-dessous.

Visualiser graphiquement $\omega(t)$. Expliquer pourquoi $\omega(t)$ est « bruité ».

3) Énergies

Exprimer l'énergie potentielle de pesanteur $\mathcal{E}_p$ (en choisissant la constante arbitraire de telle sorte que $\mathcal{E}(x=L)=0$) et l'énergie cinétique $\mathcal{E}_c$, en fonction de m, g, x, L et ω.

Calculer numériquement l'énergie potentielle $\mathcal{E}_p$, l'énergie cinétique $\mathcal{E}_c$ et l'énergie mécanique $\mathcal{E}_m$. Tracer ces trois énergies en fonction du temps. Commenter.

II) Pendule simple aux grands angles

L'objectif de cette partie est d'apprendre à résoudre une équation différentielle d'ordre 2.

Il est possible de transformer l'équation différentielle d'ordre 2 portant sur la variable θ en deux équations différentielles couplées d'ordre 1 portant sur les variables θ et ω. En effet,

$$\ddot{\theta} + \omega_0^2 \sin(\theta) = 0 \quad \Rightarrow \quad \begin{cases} \dfrac{d\theta}{dt} = \omega \\ \dfrac{d\omega}{dt} = -\omega_0^2 \sin(\theta) \end{cases}$$

Ainsi, en introduisant la liste $Y = [\theta, \omega]$, on en déduit sa dérivée :

$$\frac{dY}{dt} = \left[\frac{d\theta}{dt}, \frac{d\omega}{dt}\right] = \left[\omega, -\omega_0^2 \sin(\theta)\right] = \left[Y[1], -\omega_0^2 \sin(Y[0])\right]$$

On constate à ce stade qu'il s'agit d'une équation différentielle d'ordre 1 : $\dfrac{dY}{dt}$ s'exprime bien en fonction de Y.

Rappel :

La bibliothèque `scipy.integrate` possède une fonction `odeint` qui permet de résoudre une ED (ou plusieurs ED couplées) du premier ordre. Démarche à suivre :

- définir une fonction qui renvoie la dérivée $Y'(t)$, dont l'expression est donnée par l'équation différentielle ;
- définir la condition initiale ;
- définir un array des temps auxquels la fonction $Y(t)$ sera évaluée ;
- appeler la fonction `odeint`.

```
def deriv(Y, t):
    return # à compléter : Y → liste [*,*], t → temps

CI = [*,*] # à compléter
t = np.linspace(0, t_max, 10000) # t_max à définir
f = odeint(deriv, CI, t)
```

Résoudre l'équation différentielle du pendule simple à l'aide de la fonction `odeint`. Prendre pour conditions initiales $\theta(0) = \theta_0$ au choix et $\omega(0) = 0$. Tracer la courbe sur un temps égal à 10 T_0. *Consigner votre code ci-dessous et commenter.*

Superposer la solution obtenue avec la solution issue de l'approximation harmonique : $\theta(t) = \theta_0 \cos(\omega_0 t)$. Constater les différences aux grands angles en augmentant progressivement θ_0 jusqu'à la valeur de 179°. *Reproduire quelques courbes et commenter.*

III) Mesure de l'accélération de pesanteur

Dans l'approximation des petits angles, l'accélération de pesanteur g est reliée à la période propre T_0 du pendule par la relation :

$$\omega_0 = \sqrt{\frac{g}{L}} = \frac{2\pi}{T_0} \quad \Rightarrow \quad \boxed{g = \frac{4\pi^2 L}{T_0^2}}$$

L'objectif de cette partie est de mesurer la période T_0 d'oscillation d'un pendule de longueur L pour en déduire la valeur de g, avec son incertitude de mesure.

Justifier que l'incertitude-type de g vaut :

$$u(g) = g\sqrt{\left(\frac{u(L)}{L}\right)^2 + 2\left(\frac{u(T_0)}{T_0}\right)^2}$$

Quelles précautions expérimentales peut-on prendre afin de minimiser $u(g)$?

Réaliser une mesure de g et estimer $u(g)$.

Calculer l'écart normalisé entre votre valeur et celle d'un autre binôme. Les mesures sont-elles compatibles ?

Aides pour Python

`len(u)` renvoie la taille de la liste `u`.

`range(N)` renvoie une liste d'entiers de `0` à `N-1`.

`u.append(a)` ajoute l'élément `a` à la fin de la liste `u`.

`u[-1]` renvoie le dernier élément de la liste `u`.

`np.linspace(a, b, N)` crée un array de `N` valeurs linéairement espacées entre `a` et `b`.

`odeint(deriv, CI, t)` renvoie la solution de l'équation différentielle donnée par la fonction `deriv`, avec pour condition initiale `CI`, et évaluée aux temps `t`.

`np.cos(x)` renvoie le cosinus de l'angle `x` exprimé en radians. Faire de même pour `sin`, `tan`, `arctan`, etc.

`plt.plot(x, y, 'b-')` trace `y` en fonction de `x` avec un trait `'-'` bleu `'b'`. Il est possible de changer la couleur : rouge `'r'`, vert `'g'`, noir `'k'` et de remplacer le trait `'-'` par un trait pointillé `'--'` ou par un nuage de points `'o'`.