DEVOIR À LA MAISON 9
Résolution d'une équation différentielle d'ordre 2 : oscillateur linéaire

Capacité exigible
À l'aide d'un langage de programmation, simuler la réponse d'un système linéaire du deuxième ordre à une excitation de forme quelconque.

1 Position du problème (de Cauchy)

On souhaite résoudre une équation différentielle d'ordre 2 de la forme :

$$\boxed{x''(t)+2\xi\omega_0 x'(t)+\omega_0^2 x(t)=f(t)}$$

sur l'intervalle $\left[t_0,t_f\right]$, avec les conditions initiales $x(t_0)=x_0$ et $x'(t_0)=x'_0$. La fonction $f(t)$ dépend de la nature et de la forme de l'excitation du système physique. On transforme l'équation différentielle d'ordre 2 en un **système de deux équations différentielles couplées d'ordre 1** en définissant les vecteurs y et y' de la façon suivante :

$$\boxed{y=\begin{pmatrix} y[0]=x \\ y[1]=x' \end{pmatrix}} \text{ et } \boxed{y'=\begin{pmatrix} x' \\ x'' \end{pmatrix}=\begin{pmatrix} y[1] \\ -2\xi\omega_0 y[1]-\omega_0^2 y[0]+f(t) \end{pmatrix}}$$

C'est un problème de Cauchy de la forme : $\boxed{y'(t)=F\left(y(t),t\right)}$ où $y(t)$ est un **vecteur**.

L'intervalle de résolution est $\left[t_0,t_f\right]$ et la condition initiale est $y(t_0)=y_0=\begin{pmatrix} x_0 \\ x'_0 \end{pmatrix}$.

2 Résolution numérique en langage Python

2.1 1ère méthode : fonction odeint

La fonction odeint du module scipy.integrate permet de résoudre un système d'équations différentielles à l'aide d'un algorithme de Runge-Kutta.

```
import scipy.integrate as sci
```

```
# Résolution numérique avec odeint
"""
scipy.integrate.odeint (f,y0,t)

Résoud un système d'équations différentielles d'ordre 1
Paramètres :
    f : f(y,t) : fonction qui calcule la dérivée de y à l'instant t
    y0 : tableau ou vecteur : condition initiale
    t : tableau d'instants pour lesquels la résolution est réalisée
Renvoie :
    y : tableau ou vecteur avec les valeurs de y calculées pour chaque instant t
    (les valeurs initiales sont sur la 1ère ligne)
"""
y = sci.odeint(derivee_y,y0,t)
```

2.2 2ème méthode : méthode d'Euler explicite

La méthode d'Euler explicite est une **méthode itérative** qui permet de déterminer numériquement une solution approchée du problème de Cauchy.
La **procédure algorithmique** est la suivante :

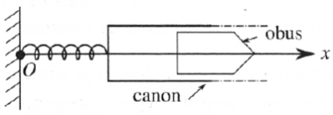
- ❖ On commence par réaliser une subdivision régulière de l'intervalle $\left[t_0, t_f\right]$ en sous-intervalles de largeur δt (δt est le **pas de résolution**), ce qui revient à générer un ensemble de points d'abscisses t_k telles que :

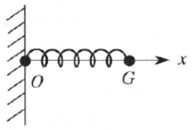
$$\boxed{t_k = t_0 + k \cdot \delta t}$$

- ❖ On cherche ensuite une valeur numérique approchée de $y\left(t_k\right)$; cette valeur approchée est notée y_k. On connaît y_0 grâce à la condition initiale. On détermine les valeurs y_k en procédant de proche en proche grâce à la **relation de récurrence** (ou **schéma numérique**) :

$$\boxed{y_{k+1} = y_k + \delta t \cdot F\left(y_k, t_k\right)}$$

3 Étude d'un oscillateur harmonique amorti : gestion du recul d'un canon (cf. TD MI2 – exercice 3)





On considère un canon de masse $M = 800$ kg. Lors du tir horizontal d'un obus de masse $m = 2{,}0$ kg avec une vitesse $\vec{v}_0 = v_0 \overrightarrow{u_x}$ ($v_0 = 600$ m.s^{-1}), le canon acquiert une vitesse initiale de recul $\vec{v}_C = -\dfrac{m}{M}\vec{v}_0$. Pour limiter la course du canon, on utilise un ressort de raideur $k_2 = 244$ N.m^{-1}, de longueur à vide L_0, dont l'une des extrémités est fixe et l'autre liée au canon. Le déplacement a lieu suivant l'axe (Ox). Le canon est assimilé à son centre de gravité G et sa position correspond à l'allongement du ressort, soit : $x = OG - L_0$. À l'instant initial, le ressort est au repos. On ajoute au système un dispositif amortisseur, exerçant une force de frottement visqueux $\overrightarrow{F_f} = -\lambda \vec{v}$, $\vec{v}$ étant la vitesse du canon.

Ce système constitue un oscillateur mécanique et l'équation différentielle vérifiée par $x(t)$ est : $\boxed{\dfrac{d^2x(t)}{dt^2} + 2\xi\omega_0 \dfrac{dx(t)}{dt} + \omega_0^2 x(t) = 0}$ avec ξ le coefficient d'amortissement tel que $\xi = \dfrac{\lambda}{2\sqrt{Mk_2}}$ et $\omega_0 = \sqrt{\dfrac{k_2}{M}}$ la pulsation propre.

1. À l'aide d'un raisonnement énergétique, retrouver l'équation différentielle du mouvement.

L'objectif est d'obtenir les solutions $x(t)$ avec les deux méthodes de résolution numériques proposées, de les comparer entre elles et avec l'expression analytique.

Les deux fichiers « DM9_MI3_Euler_odeint_niveauX.py » sont disponibles dans l'application Moodle sur l'ENT (avec X égal à 1 ou 2).
Faîtes votre choix en fonction de votre niveau ! Si vous bloquez au niveau 2, n'hésitez pas à rétrograder…
Télécharger le fichier choisi sur votre ordinateur. Le ***renommer*** *« NOM_Prénom_DM9_niveauX.py ». Lancer Pyzo puis ouvrir votre fichier.*
<u>Nota Bene</u> : Veillez à bien lire les commentaires associés à chaque ligne de code (informations, consignes, zones à compléter…)

❖ **<u>Cellule 1 : Importation des bibliothèques</u>**

2. Exécuter la « Cellule 1 » pour importer les bibliothèques (CTRL + Entrée).

❖ **<u>Cellule 2 : Données du problème physique</u>**

L'étude est d'abord réalisée en régime transitoire pseudo-périodique pour $\xi = 0{,}1$.

3. *<u>Niveaux 1 et 2</u>* : préciser les valeurs numériques ou expressions littérales des paramètres physiques du système étudié.
4. *<u>Niveaux 1 et 2</u>* : la résolution numérique est réalisée sur l'intervalle $\left[t_0, t_f\right] = \left[0, 5T_0\right]$, où T_0 est la période propre, avec $n = 200$ points. Exprimer le pas de résolution dt en fonction de n, t_0 et t_f.
5. *<u>Niveaux 1 et 2</u>* : compléter les conditions initiales et définir le vecteur initial y_0 sous forme d'un tableau (np.array).
6. *<u>Niveaux 1 et 2</u>* : générer un tableau d'instants t, constitué de n points régulièrement espacés sur l'intervalle $\left[t_0, t_f\right]$.
7. *<u>Niveaux 1 et 2</u>* : écrire la fonction derivee_y(y,t), qui prend comme arguments le vecteur y et le temps t, et qui renvoie un tableau correspondant au vecteur $y'(t) = F\left(y(t), t\right)$ du problème à résoudre. Exécuter la « Cellule 2 ».

❖ **<u>Cellule 3 : Résolution avec odeint et représentation graphique</u>**

8. *<u>Niveaux 1 et 2 :</u>* Écrire l'expression du vecteur y, solution du système d'équations différentielles obtenue avec la fonction odeint.
9. *<u>Selon le niveau 1 ou 2</u>* : compléter le code pour tracer dans deux zones graphiques distinctes d'une même fenêtre (plt.subplot), les graphes temporels de l'abscisse $x(t)$, située dans la 1ère colonne du vecteur y, et de la vitesse $x'(t)$, située dans la 2ème colonne du vecteur y. Ces deux courbes seront tracées en rouge et en trait plein (-), et identifiées avec la légende (label) « odeint ». Exécuter la « Cellule 3 ».

❖ **<u>Cellule 4 : Résolution avec Euler et représentation graphique</u>**

10. *<u>Niveau 1</u>* : compléter la fonction euler(F, y0, t, dt, n) proposée avec la relation de récurrence.
    <u>Niveau 2</u> : compléter la fonction euler(F, y0, t, dt, n) proposée avec la création et l'initialisation du vecteur y, la boucle de récurrence et la variable de sortie.

11. *Niveaux 1 et 2 :* écrire l'expression du vecteur y_euler, solution du système d'équations différentielles obtenue avec la fonction euler précédente.
12. *Selon le niveau 1 ou 2* : compléter le code pour superposer sur les graphes précédents les courbes de l'abscisse $x(t)$ et de la vitesse $x'(t)$ obtenue avec la méthode d'Euler. Ces deux courbes seront tracées en bleu et en tirets (--), et identifiées avec la légende (label) « Euler ». Exécuter la cellule.

Nota Bene* : les « Warning » générés avec** plt.subplot ***n'empêchent pas le programme de fonctionner !

Nota Bene : pour actualiser la fenêtre graphique, cliquer sur l'icône de réduction ou d'agrandissement…

13. Commenter l'allure des courbes obtenues avec les deux méthodes de résolution numériques.

- **Cellule 5 : Calcul analytique et représentation graphique**

14. *Niveaux 1 et 2 :* définir la pseudo-pulsation $\omega_P = \omega_0\sqrt{1-\xi^2}$.
15. *Niveaux 1 et 2 :* saisir les expressions analytiques de l'abscisse et de la vitesse, dans le cas d'un régime pseudo-périodique :

$$x(t) = \frac{x'_0}{\omega_P}\sin(\omega_P t)e^{-\xi\omega_0 t} \text{ et } x'(t) = x'_0\left(\cos(\omega_P t) - \xi\frac{\omega_0}{\omega_P}\sin(\omega_P t)\right)e^{-\xi\omega_0 t}$$

16. *Selon le niveau 1 ou 2* : Compléter le code pour superposer sur les graphes précédents les courbes de la position $x(t)$ et de la vitesse $x'(t)$ obtenue avec le calcul analytique. Ces deux courbes seront tracées en noir et en pointillés (:), et identifiées avec la légende (label) « Calcul ». Exécuter la cellule.
17. Comparer l'allure des courbes obtenues avec les deux méthodes de résolution numérique et avec celles obtenues par le calcul analytique.
18. Noter l'instant t_m pour lequel le recul du canon est maximal et la distance de recul d_m.

- **Influence du nombre de points**

19. Fermer la fenêtre graphique puis choisir $n = 1000$ points. Exécuter l'ensemble du fichier (*CTRL + E*). Commenter l'allure des courbes.

- **Influence du coefficient d'amortissement**

20. Fermer la fenêtre graphique puis choisir $\xi = 0{,}9$. Exécuter l'ensemble du fichier. Commenter l'allure des courbes.
21. Fermer la fenêtre graphique puis choisir $\xi = 0$ (oscillateur non amorti). Exécuter l'ensemble du fichier. Commenter l'allure des courbes.
22. Fermer la fenêtre graphique puis choisir $\xi = 1$ (régime critique). Exécuter les cellules 2, 3 et 4. Commenter l'allure des courbes. Noter l'instant t'_m pour lequel le recul du canon est maximal et la distance de recul d'_m. Comparer aux valeurs obtenues en TD.

Sauvegarder votre fichier correctement renommé et le déposer dans l'application Moodle.