Sorbonne Université - Université Paris Cité — Année 2023-2024
Option B
Préparation à l'agrégation de mathématiques

TP : Equations différentielles ordinaires

Dans ce TP, on aura besoin des modules suivants :

```
import numpy as np
import scipy as sc
import numpy.linalg as LA
import matplotlib.pyplot as plt
import scipy.integrate as integrate
```

Le dernier module sera utilisé pour résoudre des équations différentielles avec la commande `odeint`.

Exercice 1 Schéma d'Euler, calcul d'erreur
On considère le problème suivant :

$$\begin{cases} x'(t) & = x(t) + 2y(t) \\ y'(t) & = 2x(t) + y(t) \end{cases}$$

sur l'intervalle $[0,1]$ avec la condition initiale $x(0) = 5$ et $y(0) = 0$.

1. Faire un programme afin de déterminer la solution approchée de ce système à l'aide d'un schéma d'Euler explicite.
2. En prenant un pas de temps $\Delta t = 0.01$, tracer la solution approchée de ce système en fonction du temps.
3. On rappelle que la solution exacte de ce système est donnée par $x(t) = \frac{5}{2}(e^{-t} + e^{3t})$ et $y(t) = \frac{5}{2}(-e^{-t} + e^{3t})$. Comparer graphiquement la solution exacte et la solution approchée pour différentes valeurs de Δt.
4. Pour $0 \leq n \leq N$, on note $t_n = n\Delta t$ les points de discrétisation et X_n la solution approchée au temps t_n. Écrire un programme calculant l'erreur $\max_{0 \leq n \leq N} \|X(t_n) - X_n\|_\infty$. On pourra utiliser la commande `norm` du module `numpy.linalg` (avec l'option `np.inf`).
5. Tracer l'erreur en fonction de Δt en échelle logarithmique.
6. On observe que la pente de la droite est proche de 1. Comment l'interpréter ?
7. Reprendre les trois dernières questions pour le schéma du point milieu défini dans l'exercice 2.

> Le **schéma du point-milieu** pour approcher l'EDO $y'(t) = f(t, y(t))$ est donné par :
> $$y_{n+1} = y_n + hf\left(t_n + \frac{h}{2}, y_n + \frac{h}{2}f(t_n, y_n)\right)$$

Exercice 2 Étude d'un pendule : simulations numériques.
On considère un pendule de longueur l. L'évolution de θ l'angle du pendule par rapport à la verticale est donnée par l'équation suivante :

$$\theta''(t) + \frac{g}{l}\sin(\theta(t)) = 0.$$

On définit $\omega = \theta'$, on a ainsi le système

$$\begin{cases} \theta'(t) = \omega(t) \\ \omega'(t) = -\dfrac{g}{l}\sin(\theta(t)). \end{cases}$$

On suppose que $\theta(0)$ et $\omega(0)$ sont donnés.

Pour résoudre ce problème sur $[0, T]$, on prend $N \in \mathbb{N}^*$, puis on définit le pas de discrétisation $h = \frac{T}{N}$ et les temps de discrétisation $t_n = nh$ pour $0 \leq n \leq N$.

1. Déterminer la solution approchée de ce système à l'aide d'un schéma d'Euler explicite :

$$\begin{cases} \theta_{n+1} = \theta_n + h\omega_n \\ \omega_{n+1} = \omega_n - h\dfrac{g}{l}\sin(\theta_n). \end{cases}$$

   On prendra $T = 12,\ N = 200,\ l = 5,\ g = 9.81$ et les conditions initiales $\theta(0) = \frac{\pi}{3}$, $\omega(0) = 0$.

2. Implémenter une résolution via `solve_ivp` qui par défaut résout l'EDO via un schéma de Runge-Kutta explicite d'ordre 4.

   La syntaxe pour la commande `solve_ivp` est `sol=integrate.solve_ivp(f,(t0,tf),X0)` où
   • `f` a été définie de la façon suivante :
   `def f(t,X): ...`
   (même si `f` ne dépend pas de `t`, il faut que l'argument apparaisse en seconde position en entrée).
   • `(t0,tf)` est l'intervalle de temps où la solution est calculée
   • `X0` correspond à la condition initiale
   • `sol` est un objet contenant les informations sur la solution approchée :
   - `sol.t` contient le vecteur des temps où la solution est calculée
   - `sol.y` contient le tableau où la solution est calculée aux temps `sol.t`
   Remarque : `solve_ivp` choisit automatiquement le pas de temps, pour s'en affranchir on peut utiliser les mots-clefs suivants
   `sol=integrate.solve_ivp(f,(t0,tf), X0, max_step=dt, atol=1., rtol=1.)`
   où `dt` est le pas de temps souhaité.

3. Mettre en œuvre la méthode d'Euler implicite.
4. Comparer les différentes méthodes.
5. Illustration numérique du texte 2015-B4 : on considère l'équation

$$\theta''(t) + \omega^2(1 + \varepsilon\cos(t))\sin(\theta(t)) = 0,$$

   avec conditions initiales $\theta(0) = \frac{\pi}{8}$ et $\theta'(0) = 0$ et où ω et ε sont des constantes.
   Pour $(\omega, \varepsilon) = (\frac{1}{2}, 0.1)$ et $(\omega, \varepsilon) = (1, 0.2)$, comparer la méthode d'Euler explicite et Runge-Kutta explicite à l'ordre 4.

Exercice 3 Étude d'un pendule : Portrait de phase

Reprenons le système considéré dans l'exercice 2. On veut tracer le portrait de phase de ce problème pour les paramètres $l = 5$ et $g = 9.81$.

1. Tracer la solution de ce problème dans le plan des phases pour la donnée initiale $(\theta(0), \omega(0)) = (\theta_0, 0)$ avec $\theta_0 \in [0, 2\pi]$ (prendre de nombreuses valeurs de θ_0 et tracer toutes les solutions sur le même graphe). On pourra résoudre le système en utilisant la commande `odeint` du module `scipy.integrate`.
2. Pour compléter le portrait de phase, ajouter plusieurs courbes issues des données initiales $(-\pi, \omega_0)$ avec $\omega_0 > 0$ et $(3\pi, \omega_0)$ avec $\omega_0 < 0$.
3. Ajouter les points critiques sur cette figure.
4. Faisons maintenant un zoom près du point $(0, 0)$. Sur une autre figure, représenter le diagramme des phases près du point $(0, 0)$. Ajouter les isoclines.
5. Enfin, nous allons représenter la fonction second membre $f(\theta, \omega) = (\omega, -\frac{g}{l}\sin(\theta))$ sur cette figure. Pour cela, on pourra utiliser les commandes `np.meshgrid`, `np.hypot` et `plt.quiver`.
   NB : si vous essayez de représenter f comme ici sur la première figure, vous pourriez avoir des difficultés dues au fait que ω et θ n'ont pas le même ordre de grandeur.

Exercice 4 Schémas implicites

Dans cet exercice, nous allons utiliser les schémas d'Euler implicite et de Crank–Nicolson pour résoudre le problème de l'exercice 1.

1. Coder un programme `Newton(F,dF,X0,tol,maxiter)` qui renvoie la solution $X \in \mathbb{R}^2$ de l'équation $F(X) = 0$ calculée par la méthode de Newton (`dF` est la jacobienne de `F`).
2. En utilisant la question précédente, coder le schéma d'Euler implicite donné par $X_{n+1} = X_n + hf(t_{n+1}, X_{n+1})$.
3. Coder le schéma de Crank–Nicolson donné par $X_{n+1} = X_n + \frac{h}{2}(f(t_{n+1}, X_{n+1}) + f(t_n, X_n))$.
4. Retrouver numériquement les taux de convergence de ces deux schémas (voir la fin de l'exercice 1).

Exercice 5 Système de Lotka-Volterra

On considère le système de Lotka-Volterra sur l'intervalle $[0, 30]$:

$$\begin{cases} x'(t) & = x(t)(2 - y(t)/10) \\ y'(t) & = y(t)(x(t)/10 - 4) \end{cases}$$

avec $x(0) = 100$ et $y(0) = 70$. On pourra utiliser `odeint` pour calculer la solution de ce problème.

1. Tracer x et y en fonction du temps.
2. Tracer la solution dans l'espace des phases.
3. Tracer les isoclines dans l'espace des phases.
4. Ajouter d'autres orbites dans l'espace des phases.
5. Représenter f pour compléter le portrait de phase.