TP n° 12 de Physique
Résolution d'équations différentielles

Objectif du TP

Les équations différentielles peuvent être résolues de deux façons :
– **analytiquement**, si l'équation différentielle correspond à un type connu après mise sous forme canonique
– **numériquement** sinon.
De nombreuses équations, notamment les équations différentielles non linéaires, doivent obligatoirement être résolues numériquement. De nombreux algorithmes informatiques existent pour réaliser ces résolutions. Nous étudierons lors de cette séance le plus élémentaire algorithme, l'**algorithme d'Euler**.

1 Méthode d'Euler pour les équations d'ordre 1

La méthode d'Euler est un algorithme permettant de construire une approximation de la solution d'une équation différentielle. On étudie le problème suivant, dont on suppose l'existence et l'unicité de la solution :

$$\dot{y}(t) = F(t,y) \quad \text{avec la condition initiale} \quad y(t_0) = y_0$$

On se donne un « pas » de l'algorithme $\mathrm{d}t > 0$. Si $\mathrm{d}t$ est suffisamment petit, on peut faire l'approximation

$$y(t_0 + \mathrm{d}t) \approx y(t_0) + \dot{y}(t_0) \cdot \mathrm{d}t \quad \text{soit} \quad y(t_1) \approx y_0 + F(t_0,y_0) \cdot \mathrm{d}t$$

où $t_1 = t_0 + \mathrm{d}t$. On réitère le procédé pour obtenir $y(t_2)$, $y(t_3)$, $y(t_4)$... par

$$\left\{ \begin{array}{lcl} t_i & = & t_0 + i \cdot \mathrm{d}t \\ y_{i+1} & = & y_i + F(t_i,y_i) \cdot \mathrm{d}t \end{array} \right.$$

En reliant les points $M_n(t_n,y_n)$ par des segments de droite, on obtient la courbe d'une fonction affine par morceaux, approximation de la solution de l'équation différentielle. Il est clair qu'à chaque étape, on a probablement dévié un peu de la solution exacte. Ces erreurs peuvent se cumuler et on peut obtenir une solution approchée qui est très éloignée de la solution exacte. Cependant, cette méthode est relativement efficace à condition de prendre un pas $\mathrm{d}t$ petit (pour assurer la légitimité des approximations) et de travailler sur un intervalle pas trop long (pour limiter le nombre des approximations).

D'autres *schémas numériques* que celui d'Euler existent pour réduire l'erreur commise par l'approximation. Celui de Runge-Kutta « d'ordre 4 » (RK4) est le schéma régulièrement utilisé dans les logiciels de simulation.

2 Application sur des équations différentielles du premier ordre

2.1 Réalisation de l'algorithme

– Faire, sur papier, un dessin représentant graphiquement le fonctionnement de l'algorithme.
– Créer une fonction `euler(F,t0,tf,y0,dt)` ayant pour paramètres une fonction F de deux variables t et y correspondant à l'équation différentielle, deux réels t_0 et t_f délimitant l'intervalle de calcul, un réel y_0 égal à la condition initiale et un réel $\mathrm{d}t$ égal au pas de calcul. Cette fonction retournera deux listes t et y (ou tableaux `NumPy`) contenant les abscisses de calcul et les valeurs de la fonction solution trouvée.

2.2 Équation différentielle linéaire : modélisation d'une charge exponentielle

On souhaite mettre en place la méthode d'Euler sur une équation différentielle simple, dont on connaît la solution analytique afin de vérifier les résultats de simulation. On choisit l'équation

$$\tau\, y' + y = K \quad \text{avec la condition initiale} \quad y(0) = 0$$

qui a pour solution analytique $$y(t) = K\left(1 - \exp\left(\frac{-t}{\tau}\right)\right)$$

On prend $\tau = 2$ s et $K = 3$. Les courbes doivent être constituées de 1000 points, entre $t = 0$ et $t = 5\,\tau$.
- Afficher graphiquement la courbe solution analytique.
- Tester la fonction `euler` pour déterminer la solution numérique. Afficher le résultat sur la même figure.
- Que se passe-t-il si dt devient trop grand (par exemple d$t = 1$ s) ?

2.3 Équation différentielle non linéaire : modélisation d'une chute libre

On souhaite maintenant modéliser une chute libre, avec des frottements de la forme $-\alpha\, v^2$. L'équation différentielle est donc (axe des z vers le bas)

$$m\,\frac{\mathrm{d}v}{\mathrm{d}t} + \alpha\, v^2 = m\, g$$

Pour une bille d'acier de rayon 5 mm dans l'air, on peut mesurer $\alpha = 5 \cdot 10^{-5}\,\mathrm{kg{\cdot}m^{-1}}$ et $m = 15\,\mathrm{g}$.
- Déterminer numériquement et afficher l'évolution de la vitesse entre 0 et 20 s.

3 Résolution des équations différentielles d'ordre 2

On peut utiliser la méthode d'Euler pour les équations d'ordre 2, à condition de réaliser l'approximation successivement pour $\dot{y}(t)$ et $y(t)$. L'équation

$$\ddot{y}(t) = F(t,y,\dot{y}) \quad \text{avec les conditions initiales} \quad y(t_0) = y_0 \quad \text{et} \quad \dot{y}(t_0) = \dot{y}_0$$

peut être résolu par

$$\left\{\begin{array}{lcl} t_i & = & t_0 + i \cdot \mathrm{d}t \\ y_{i+1} & = & y_i + \dot{y}_i \cdot \mathrm{d}t \\ \dot{y}_{i+1} & = & \dot{y}_i + F(t_i,y_i) \cdot \mathrm{d}t \end{array}\right.$$

On peut aussi utiliser plus facilement la fonction `odeint` du module `scipy.integrate`. Cette fonction résout en réalité le système d'équations qui s'appliquent sur le couple $(y,\dot{y})$. À l'ordre 2, sa syntaxe est la suivante :
- elle a trois arguments : une fonction F caractérisant l'équation différentielle, une liste de 2 conditions initiales $(y_0,\dot{y}_0)$, une liste (ou tableau) t contenant tous les instants du calcul.
- elle a un retour sous forme d'un tableau de 2 colonnes, contenant les y dans la première (`sol[:,0]`) et les $\dot{y}$ dans la deuxième (`sol[:,1]`).
- la fonction F doit obligatoirement avoir deux arguments, la liste $(y_i,\dot{y}_i)$ et l'instant t de calcul. Elle retourne la dérivée sous forme de liste $(\dot{y}_i,\ddot{y}_i)$.

3.1 Équation différentielle non linéaire : modélisation d'un pendule pesant

On souhaite modéliser l'évolution, pendant 20 s, d'un pendule simple d'équation

$$\ddot{\theta} + \frac{\omega_0}{Q}\,\dot{\theta} + \omega_0^2\,\sin\theta = 0 \quad \text{avec} \quad \omega_0 = 1\,\mathrm{rad{\cdot}s^{-1}}, \quad Q = 3, \quad \theta_0 = \frac{\pi}{2}\,\mathrm{rad} \quad \text{et} \quad \dot{\theta}_0 = 0$$

- Déterminer numériquement à l'aide de `odeint` et afficher l'évolution de θ entre 0 et 20 s.