Concours Blanc : TP d'Informatique

PTSI A et B Lycée Eiffel

29 mai 2015

Le but de ce TP est de comparer différentes méthodes de résolution d'équations différentielles. La première de ces méthodes est la méthode d'Euler, que vous commencez à bien connaitre. Les deux autres que nous aborderons sont des améliorations de cette dernière, appelées respectivement méthode de Heun et méthode de Runge-Kutta (qui existe en plusieurs versions, on s'intéressera ici plus précisément à la méthode de Runge-Kutta d'ordre 4). On testera d'abord nos méthodes sur une équation différentielle d'ordre 1, puis sur des équations d'ordre 2.

Première partie : ordre 1.

Sans revenir sur l'aspect mathématique de la méthode d'Euler, rappelons-en le principe : on souhaite résoudre de façon approchée une équation de la forme $y'(t) = F(t, y(t))$ (la fonction F étant une fonction à deux variables ; par exemple, si on souhaite résoudre l'équation $y'(t) = 2ty^2(t) + t$, la fonction F sera définie par $F(t, y) = 2ty^2 + t$) sur un intervalle de résolution $[t_0, t_f]$. Pour cela, on découpe l'intervalle en n morceaux de largeur $h = \dfrac{t_f - t_0}{n}$, et de bornes $t_i = t_0 + i * h$. On calcule alors les valeurs approchées des valeurs $y(t_i)$ (notées plus simplement y_i) via la formule de récurrence :

$$y_{i+1} = y_i + h * F(t_i, y_i)$$

Les deux autres méthodes que nous programmerons fonctionnent sur le même principe, mais avec des relations de récurrence plus compliquées. Pour la méthode de Heun, on pose :

$$y_{i+1} = y_i + \frac{h}{2}(F(t_i, y_i) + F(t_{i+1}, y_i + hF(t_i, y_i)))$$

Pour la méthode de Runge-Kutta, on calcule successivement :

$$\alpha_i = y_i + \frac{h}{2}F(t_i, y_i), \quad \beta_i = y_i + \frac{h}{2}F\left(t_i + \frac{h}{2}, \alpha_i\right), \quad \gamma_i = y_i + hF\left(t_i + \frac{h}{2}, \beta_i\right)$$

et on pose alors :

$$y_{i+1} = y_i + \frac{h}{6}\left(F(t_i, y_i) + 2F\left(t_i + \frac{h}{2}, \alpha_i\right) + 2F\left(t_i + \frac{h}{2}, \beta_i\right) + F(t_{i+1}, \gamma_i)\right)$$

Bien entendu, dans tous les cas, on suppose connue la valeur initiale y_0.

1. Écrire une fonction Python **decoupage(t0,tf,n)**, qui renvoie une liste de $n+1$ valeurs obtenues comme bornes du découpage de l'intervalle $[t_0, t_f]$ en n morceaux.
2. Tracer avec le module matplotlib.pyplot une courbe de la fonction $f : t \mapsto e^{t^2}$ sur l'intervalle $[0, 2]$ en prenant successivement 10, 100 et 1000 intervalles pour le découpage (on pourra utiliser la fonction précédente). Ces deux premières questions ne sont pas nécessairement utiles pour la suite.
3. On veut résoudre l'équation différentielle $y' = 2ty(t)$, avec comme condition initiale $y(0) = 1$, à l'aide des méthodes décrites précédemment. Définir la fonction à deux variables F correspondant à cette équation.

4. Compléter le programme suivant pour la méthode d'Euler :

```
def Euler(F,y0,t0,tf,n) :
    y,y0=t,t0
    temps,ordonnee=[],[]
    h=(tf-t0)/float(n)
    for i in range(n) :
        y=y+ ...
        t= ...
        temps.append[...]
        ...
    return ordonnee
```

5. Tracer ensuite la courbe approchée de la solution de l'équation obtenue sur l'intervalle $[0,2]$ pour $n=10$, puis $n=100$ et $n=1000$. Comparer la valeur approchée obtenue pour $y(2)$ avec la valeur théorique e^4 (demandez à Python combien ça vaut).
6. Écrire le programme correspondant pour la méthode de Heun, puis pour la méthode de Runge-Kutta (utilisation du copier-coller conseillée !).
7. Tracer sur un même graphique les quatre courbes obtenues sur l'intervalle $[0,2]$ pour $n=100$ (courbe de la fonction $f(t)=e^{t^2}$, puis chacune des trois courbes approchées obtenues avec les différentes méthodes). Commenter.

Deuxième partie : ordre 2.

On souhaite adapter la méthode d'Euler (et les autres) au cas d'une équation différentielle d'ordre 2 de la forme $y''(t)=a(t)y'(t)+b(t)y(t)$ (on se contentera d'équations linéaires). L'astuce est de calculer les valeurs approchées non pas d'une seule fonction (la solution y) mais de deux (y et y') en les voyant comme solution d'un système de deux équations différentielles. Pour simplifier les choses, notons $z(t)=y'(t)$, l'équation du second ordre est alors équivalente aux deux équations $z(t)=y'(t)$, et $z'(t)=a(t)z(t)+b(t)y(t)$, système de deux équations d'ordre 1. Pour résoudre ce système par la méthode d'Euler, on supposera connues les valeurs initiales de y (notée y_0) et de y' (notée z_0), et on calculera les suivantes par les relations de récurrence :

$$\begin{cases} y_{i+1} &= y_i + h * z_i \\ z_{i+1} &= z_i + h * (a(t_i)z_i + b(t_i)y_i) \end{cases}$$

Prenons un exemple concret, avec l'équation du pendule simple : $y''(t) = -\sin(y(t))$. On obtient les relations :

$$\begin{cases} y_{i+1} &= y_i + h * z_i \\ z_{i+1} &= z_i - h * sin(y_i) \end{cases}$$

1. Écrire un programme Python permettant la résolution de l'équation du pendule par la méthode d'Euler.
2. Tracer la courbe obtenue sur l'intervalle $[0,20]$ avec les conditions initiales $y_0=0$ et $z_0=\frac{3}{2}$, pour $n=10$, puis $n=100$ et $n=1000$ (vous pourrez même tenter plus que 1000 si vous voulez, à vous d'interpréter les courbes obtenues et d'adapter). Commenter.
3. Modifier le programme précédent pour qu'il ne trace plus la courbe approchée de la solution (y en fonction de t), mais le portrait de phase (y sur un axe et y' sur l'autre). Reprendre la simulation sur $[0,20]$ avec $n=10$, $n=100$, $n=1000$, interpréter les courbes obtenues.
4. Adapter la méthode de Heun et la méthode de Runge-Kutta à cette équation du deuxième ordre, puis tracer les courbes correspondantes, et comparer.
5. Reprendre les questions précédentes avec cette fois-ci $z_0=2$. On devrait théoriquement avoir une courbe strictement croissante avec une limite égale à π en $+\infty$. Commenter les résultats obtenus.