TP n°5 : Résolution numérique d'équations différentielles (corrigé)

Module : Calcul Scientifique (3A/B) Année universitaire : 2023-2024

1 Introduction

Les méthodes numériques employées pour résoudre les équations différentielles sont des méthodes approximatives basées sur la discrétisation de la variable du temps ainsi que sur l'utilisation de différences finies pour approcher les dérivées et les intégrales. Le problème se ramène alors à un calcul itératif, facile à automatiser à l'aide d'un programme informatique. Pour effectuer ce calcul numérique, l'utilisateur doit disposer :

— de la durée T de la simulation numérique.

— des conditions initiales.

Dans ce TP on s'intéresse aux équations différentielles de la forme :

$$x' = f(t,x)$$

où f est une fonction définie sur une partie U de $\mathbb{R}^2$, à valeurs dans $\mathbb{R}$. Une solution de cette équation est une fonction x de classe C^1 définie sur un intervalle I de $\mathbb{R}$ et à valeurs dans $\mathbb{R}$ vérifiant :

$$\forall t \in I, (t, x(t)) \in U.$$

et

$$\forall t \in I, \quad x'(t) = f(t, x(t)).$$

On ajoute à cette équation différentielle une condition initiale sous la forme d'un couple $(t_0, x_0) \in U$ et chercher à résoudre le problème de Cauchy (PC) suivant :

$$x' = f(t,x), \quad x(t_0) = x_0. \tag{1}$$

Sous certaines conditions sur f , ce problème, dit de Cauchy, admet une unique solution qui s'ecrit sous la forme :

$$x(t) = x_0 + \int_{t_0}^{t} f(s, x(s)))ds, \, \forall t \in I.$$

2 Objectif du TP

L'objectif du TP est de :

1. Résolution numérique des problèmes de Cauchy.

2. Représenter et interpréter graphiquement les résultats obtenus.

Il faudra charger ces modules, en tapant les commandes suivantes :

```python
import numpy as np
import matplotlib.pyplot as plt
```

3 Application : Évolution d'une population bactérienne

L'évolution d'une population bactérienne dans un milieu de culture est décrite par le modèle suivant

$$(PC) \begin{cases} x'(t) = r\,x(t)\,(1 - \dfrac{x(t)}{K}), & t \geq 0 \\ x(0) = x_0, \end{cases}$$

avec

- t désigne le temps exprimé en heures,
- $x(t)$ est l'effectif de la population bactérienne au temps t, donné en mg,
- r est le taux de croissance de la population ($r > 0$), donné en mg/h,
- K est le nombre de bactéries maximal que le milieu peut supporter, donné en mg,
- x_0 est l'effectif initial ($x_0 > 0$), donné en mg.

La population étudiée dans cette application a les caractéristiques suivantes

r=0.2 mg/h et K=100 mg.

L'observation de l'évolution de l'effectif $x(t)$ à chaque instant, nous amène à résoudre numériquement le problème de Cauchy (PC) par une méthode itérative décrite par le schéma $(Euler - Heun)$ suivant

$$(Euler - Heun) \begin{cases} p = x_k + hf(t_k, x_k), \\ x_{k+1} = x_k + \dfrac{h}{2}(f(t_k, x_k) + f(t_{k+1}, p)), & \forall k \in \{0,1,...,N-1\}, \\ x(0) = x_0 \text{ donnée} \end{cases}$$

où

- N est le nombre de sous-intervalles de la subdivision uniforme de $[t_0, t_0 + T]$ avec t_0 est l'instant initial et $T > 0$,
- $h = \dfrac{T}{N}$ est le pas de la discrétisation de l'intervalle $[t_0, t_0 + T]$,
- $t_k = t_0 + kh$, pour $k \in \{0,1,...,N\}$, désigne les points de la discrétisation.
- $f(t,x)$ est le second membre du problème (PC),
- x_k est la solution approchée de la solution exacte évaluée au point t_k ($x_k \simeq x(t_k)$).

Q1.a) Écrire une fonction nommée `solver(f,x0,t0,T,N)` qui prend en paramètres une fonction `f`, une valeur initiale `x0`, l'instant initial `t0`, `T` la longueur de l'intervalle `[t0, t0+T]`, et `N` le nombre de sous-intervalles de `[t0, t0+T]`. Cette fonction doit retourner la liste des valeurs `X`$= [x_0, x_1, \ldots, x_N]$.

```python
def solver(f,x0,t0,T,N):
    h=T/N
    X=[x0]
    for k in np.arange(0, N):
        p=X[k]+h*f(t0+k*h,X[k])
        X.append(X[k]+0.5*h*(f(t0+k*h,X[k])+f(t0+(k+1)*h,p)))
    return X
```

Q1.b) On se place sur l'intervalle $[0,72]$. Pour $N = 180$ et $x0 = 40$, déclarer la fonction $f(t,x)$ puis donner la valeur approchée de l'effectif de la population bactérienne à l'instant $t = 72$.

```python
# Instructions Python
r=0.2
K=100
t0=0
T=72
N=180
x0=40
f=lambda t, x: r*x*(1-x/K)
X=solver(f,x0,t0,T,N)
print("L'effectif de la population bactérienne à l'instant t=72 vaut ",X[-1])
```

```
L'effectif de la population bactérienne à l'instant t=72 vaut  99.99991519268319
```

Q1.c) Sachant que la solution exacte du problème (PC) est donnée par

$$x_{exacte}(t,x_0) = \frac{x_0 K}{x_0 + (K - x_0)e^{-rt}} \text{ , } t \geq 0 \text{ et } x_0 > 0$$

Déduire l'erreur commise par la méthode numérique $(Euler - Heun)$ à l'instant $t = 72$ et pour $x_0 = 40$.

```python
#Instructions Python
r=0.2
K=100
x_exacte = lambda t,x0 :(x0*K)/(x0+(K-x0)*np.exp(-r*t))
t=72
x0=40
print("L'erreur commise est ",abs(x_exacte(t,x0)-X[-1]))
```

```
L'erreur commise est  1.1988313275423934e-06
```

NB : Dans la suite, pour tous les tests, veuillez utiliser la fonction f déclarée dans la question Q1.b), r=0.2 et K=100.

Q2. Maintenant, on veut déterminer au bout de combien d'heures, t_k, la suite (x_k), définie dans le schéma $(Euler - Heun)$, s'est stabilisée à une tolérance de 10^{-3}. Autrement dit, on cherche l'instant t_k qui vérifie le critère d'arrêt

$$\mid x_{k+1} - x_k \mid \leq 10^{-3} \quad \forall \; k \in \{0, 1, ..., N-1\}.$$

Q2.a) Écrire une fonction `limite(X,t0,T,N)` qui prend en paramètres la liste `X` calculée par le schéma $(Euler - Heun)$, l'instant initial `t0`, `T` la longueur de l'intervalle `[t0, t0+T]` et `N` le nombre de sous-intervalles de la subdivision uniforme de `[t0, t0+T]`.

Cette fonction `limite` retourne le nombre d'heures t_k.

```python
def limite(X,t0,T,N):
    k=0
    h=T/N
    while (k <N) and (np.abs(X[k+1]-X[k])>1e-3):
        k=k+1
    return t0+k*h
```

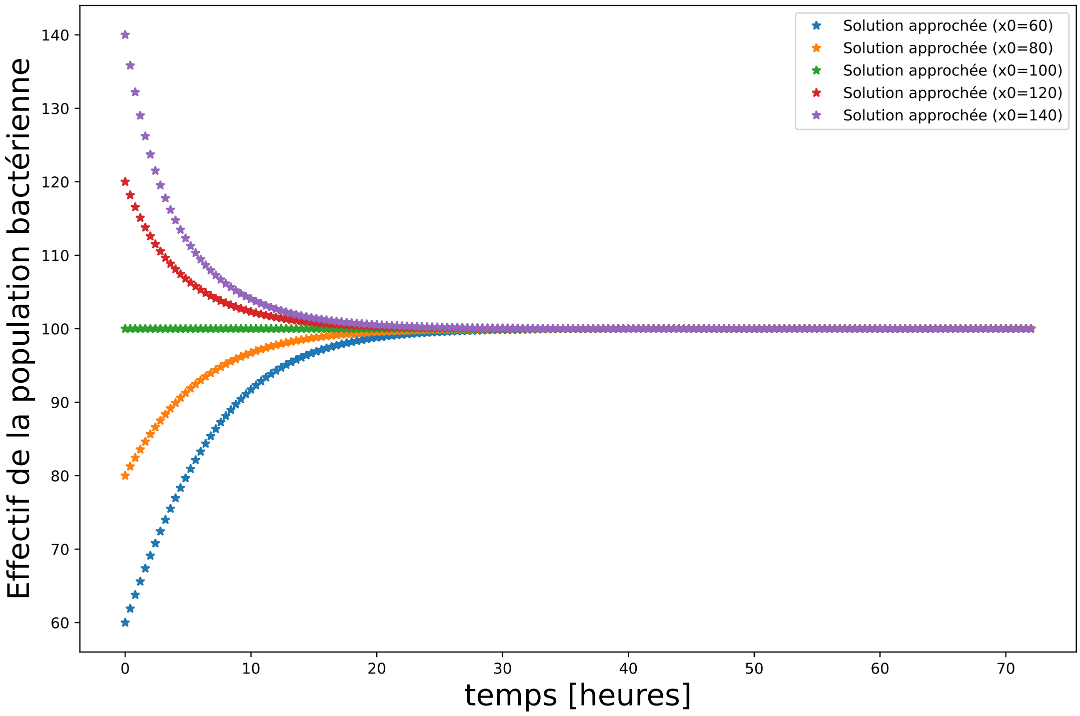
Q2.b) Pour `t0=0`, `T=72`, `N=180` et `x0=40`, déterminer l'instant de stabilisation du schéma $(Euler - Heun)$.

```
In[]: limite(X,0,72,180)
Out[]: 47.2
```

Q3.a) En utilisant la fonction `solver(f,x0,t0,T,N)`, tracer sur un même graphe, les courbes représentant l'évolution de la population bactérienne sur l'intervalle $[0,72]$ avec `N=180` pour différentes valeurs de conditions initiales `x0` $\in \{60, 80, 100, 120, 140\}$.

```python
# Instructions Python
t0=0
T=72
N=180
# on pourra mettre t = np.linspace(t0, t0+T, N+1) ailleurs de la boucle for
M = [60,80,100,120,140]
plt.figure(figsize=(12,8))
for x0 in M:
    X= solver(f,x0,t0,T,N)
    t = np.linspace(t0, t0+T, N+1)
    plt.plot(t, X, '*', label='Solution approchée (x0='+str(x0)+')')
plt.xlabel("temps [heures]", fontsize = 20)
plt.ylabel("Effectif de la population bactérienne", fontsize = 20)
plt.legend()
plt.show()
```

La figure résultante est la suivante :



Q3.b) Observez les courbes tracées et interpréter graphiquement l'évolution de la population bactérienne.

- Si $x0 \in]0, K[$ alors la suite (x_k) est croissante et converge vers K lorsque k tend vers l'infini.
- Si $x0 > K$ alors la suite (x_k) est décroissante et converge vers K lorsque k tend vers l'infini.
- Si $x0 = K$ alors la suite (x_k) est constante et égale à K.

On peut accepter aussi cette réponse : $\forall x_0 > 0$, la suite (x_k) converge vers K lorsque k tend vers l'infini.