Feuille de TP Informatique 09

Résolution numérique d'équations différentielles

Exercice 1.— Erreur dans la méthode d'EULER explicite

1. Saisir la fonction `Euler_explicite_ordre1` du cours [1].

On souhaite utiliser cette fonction pour résoudre numériquement, sur le segment $T = [0,5]$, le problème de CAUCHY :

$$y'(t) = 1 - y^2(t) \text{ et } y(0) = 0,$$

dont l'inconnue y est une fonction (de classe $\mathscr{C}^1$) de $\mathbb{R}$ dans $\mathbb{R}$.

On signale qu'on connaît déjà la solution du problème, qui est la fonction tangente hyperbolique th.

2. Vérifier graphiquement, en prenant successivement des subdivisions de $[0,5]$ à $n = 11$, 101, 1001 et 10001 valeurs, que la méthode d'EULER explicite converge bien vers th.

On pourra s'inpirer du listing 1 pour gérer cette famille de valeurs de n et, tracer, pour chaque valeur de n le graphique (t_k, y_k).

3. L'erreur $E(n)$ de la méthode pour une valeur donnée de n est définie, comme en cours, par :

$$E(n) = \max_{k\in\{0,\dots,n\}} \left|\text{th}(t_k) - y_k\right|.$$

On souhaite tracer en échelle logarithmique $E(n)$ en fonction de n. À cette fin, on propose en listing 1 un script à amender.

Listing 1 – python/Eulerex-acompleter.py

```
1  a = 0. ; b = 5. #Spec. intervalle travail
2  y0 = 0.         #Spece. condition. initiale
3  N = [11,101,1001,10001]; E = []
4  for n in N:
5      T = np.linspace(a,b,n)
6      Y = Euler_explicite_ordre1(F,y0,T)
7      exactes = np.tanh(T)
8      differences = XXXXX
9      E.append(max(differences))
10
11 plt.clf() ; plt.grid() #nettoyer canevas, afficher grille
12 plt.xlabel("$n$",fontsize=15) ; plt.ylabel("$E(n)$",fontsize=15)
13 plt.xscale('log') ; plt.yscale('log')
14
15 plt.plot(N,E,'bo-',lw=2)
```

Compléter la ligne 8 de ce script, l'exécuter et conclure quant à l'ordre de la méthode dans ce cas précis.

Exercice 2.— Méthode de HEUN.

La méthode de HEUN de résolution d'un problème de CAUCHY est une méthode dite de « prédiction-correction ». Avec des notations analogues à celles du cours sur la méthode d'EULER explicite, son principe est le suivant.

À un certain rang k, on appelle $\widetilde{y_{k+1}}$ la valeur y_{k+1} que donne un pas de schéma d'EULER explicite :

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

On a fait en ce sens une « prédiction » sur la valeur de y en t_{k+1}. Mais au lieu de prendre $y_{k+1} = \widetilde{y_{k+1}}$, on fait une « correction ». Sachant que :

$$y(t_{k+1}) - y(t_k) = \int_{t_k}^{t_{k+1}} y'(t)dt = \int_{t_k}^{t_{k+1}} F\left(y(t),t\right)dt$$

on approche la valeur de l'intégrale par celle du trapèze compris entre les points $\left(t_k, F\left(y_k,t_k\right)\right)$ et $\left(t_{k+1}, F\left(\widetilde{y_{k+1}}, t_{k+1}\right)\right)$ en utilisant la valeur « prédite » $\widetilde{y_{k+1}}$. Ainsi :

$$\int_{t_k}^{t_{k+1}} F\left(y(t),t\right)dt \approx \delta t \times \frac{F\left(y_k,t_k\right) + F\left(\widetilde{y_{k+1}}, t_{k+1}\right)}{2}$$

1. Pour être précis, il s'agit de la fonction `Euler_explicite_ordre1_B` du cours, que l'on doit renommer. Ses arguments sont `F`, une fonction `float,float->float`, la fonction « définissante »de l'équation différentielle $\frac{dy}{dt} = F(y,t)$, `y0`, un `float`, la configuration initiale, `T`, un `ndarray`, la plage des temps (t_k) de calcul. Elle retourne le `ndarray` des valeurs (y_k).

Finalement, le schéma numérique de la méthode de HEUN (pour une valeur initiale $y(t_0)$ donnée) est :

$$y_0 = y(t_0) \text{ et } \forall k \in \{0,\dots,n-1\}, \quad \left\{ \begin{array}{rcl} \widetilde{y_{k+1}} & = & y_k + \delta t.F\left(y_k,t_k\right) \\ \\ y_{k+1} & = & y_k + \delta t.\dfrac{F\left(y_k,t_k\right)+F\left(\widetilde{y_{k+1}},t_{k+1}\right)}{2} \end{array} \right. .$$

1. Écrire une fonction `Heun` de signature [2] calquée sur celle de `Euler_explicite_ordre1` et qui retourne une liste de valeurs approchée y_k de $y(t_k)$ aux instants t_k en utilisant la méthode de HEUN.
2. Appliquer `Heun` à l'équation différentielle :

$$y'(t) = y(t)$$

avec $y(0) = 1$, sur l'intervalle $[0,5]$. On prendra une subdivision de $[0,5]$ à 11 points.
3. En reprenant l'exemple (et les scripts !) de l'exercice 1, comparer méthode EULER et méthode de HEUN ; montrer numériquement (comme dans la dernière question de l'exercice 1) que la méthode de HEUN est d'ordre 2, c.à.d. que l'erreur est en $O(1/n^2)$.

Exercice 3.— Application de la méthode d'EULER explicite aux équations différentielles ordinaires d'ordre 2.

D'après le cours, une équation différentielle scalaire d'ordre 2, d'inconnue une fonction y à valeurs numériques, peut se mettre sous forme d'un système différentiel d'ordre 1, d'inconnue Y, à valeurs vectorielles. Le problème est alors du type :

$$\forall t \in I, \quad Y'(t) = F\left(Y(t),t\right) \quad \text{avec } Y(t) = \left(y(t),y'(t)\right)$$

où F est une fonction $\mathbb{R}^3 \to \mathbb{R}^2$.
1. On souhaite écrire une fonction calquée sur la fonction `Euler_explicite_ordre1` du cours, mais adaptée aux problèmes de CAUCHY d'ordre 2. On souhaite insister sur la nécessité de lire très attentivement les spécifications de cette fonction, définies ci-dessous, car elles ne sont pas identiques à celles de `Euler_explicite_ordre1`.

Écrire une fonction `Euler_explicite_ordre2`, qui prend en arguments :
— la fonction vectorielle `F` du problème (`F` retourne une liste ou un tableau de deux valeurs) ;
— les conditions initiales `Y0` (tableau ou liste à deux valeurs égales à $y(t_0)$ et $y'(t_0)$) ;
— deux flottants `t_0` et `t_max` (bornes de l'intervalle de résolution) ;
— le nombre `n` d'intervalles $[t_{k+1} - t_k]$ dans la subdivision régulière $(t_0 = a, t_1, \dots, t_{n-1}, , t_n = b)$, avec :

$$\forall k \in \{0,\dots,n-1\}, \quad t_{k+1} - t_k = \delta t = (t_{max} - t_0)/n$$

`Euler_explicite_ordre2` devra retourner deux listes correspondant aux valeurs approchées de y et y' aux instants « t_k ».
2. Appliquer `Euler_explicite_ordre2` au problème de CAUCHY suivant :

$$\forall t \in [0,t_{max}], \quad \frac{d^2u}{dt^2}(t) + \frac{\omega_0}{Q}\frac{du}{dt}(t) + \omega_0^2 u(t) = \omega_0^2.U_0 + \omega_0^2.U_1\cos(\omega.t)$$

avec $t_{max} = 20\text{s}$, $\omega_0 = 4\text{rad}\,\text{s}^{-1}$, $Q = 5\text{s}$, $U_0 = 6\text{V}$, $U_1 = 3\text{V}$ et $\omega = 8\text{rad}\,\text{s}^{-1}$, et avec les conditions initiales $u(0) = 0$ et $\frac{du}{dt}(0) = 0$. On pourra afficher la courbe de la solution $t \mapsto u(t)$ pour $t \in [0,t_{max}]$ ainsi que le portrait de phase de la solution.

Exercice 4.— Utilisation de `odeint`

À l'aide de la fonction `odeint` du module `scipy.integrate`, répondre aux questions suivantes.

1. Modèle de VERHULST**.**

Le modèle de VERHULST ou modèle logistique est un modèle démographique de variation d'une population d'individus (typiquement des bactéries). On note $p(t) \in \mathbb{R}^+$ le nombre d'individus [3] à un instant t, et p_0 sa valeur initiale. Le modèle tient compte de la finitude des ressources dont dispose la population dans son milieu environnant, sous la forme d'un coefficient P appelé la capacité du milieu. Le modèle tient aussi compte du taux de croissance de la population quand elle est très inférieure à P, sous la forme d'un coefficient c. Dans le modèle de VERHULST, p vérifie, sur un certain intervalle I, l'équation différentielle :

$$\forall t \in I,\ p'(t) = c.p(t)\left(1 - \frac{p(t)}{P}\right)$$

avec $P = 50\,000$ et $c = 2$ jour^{-1}. On s'intéresse à la population $p(t)$ pour t compris entre 0 et 10 jours.

2. Mêmes arguments en type et signification, même valeurs retournées.

3. *Dit comme ceci, il s'agit d'un entier, ce qui est problématique du point de vue de l'Analyse Mathématique. (Une fonction continue sur un intervalle à valeurs entières est constante). Une façon d'appréhender ces "entiers" est de comprendre que le raisonnement/la modélisation n'a pas lieu sur* une *expérience* réelle *mais sur une expérience* moyenne *qui est la moyenne (quoique cela puisse vouloir dire) de l'infinité possible d'expériences réelles soumises à aléas. On fait donc d'un problème discret un problème continu, afin de le traiter en termes différentiels. C'est la même approche que lors de l'étude de l'évolution d'une population d'atomes radioactifs (c.f. chimie).*

1.a. Après résolution numérique de cette équation différentielle dans l'intervalle I, représenter la courbe des variations de $p(t)$ pour p_0 égal à 1 000 (sous-population initiale, par rapport à la population idéale P) puis pour p_0 = 100 000 (sur-population initiale).
1.b. Comparer dans chaque cas avec la solution analytique :

$$\forall t \in I, \quad p(t) = \frac{b}{1 + \frac{b-a}{a} e^{-c.t}}$$

pour des valeurs de a, b, c à déterminer en fonction de c, P, p_0.

2. Équation de MATHIEU**.**

De multiples systèmes physiques (pendule modulé sinusoïdalement, position d'une particule chargée dans un piège électrostatique quadripolaire,...) sont régis par une équation différentielle de type équation de MATHIEU, dont la forme canonique est :

$$\frac{d^2x}{dt^2}(t) + \omega_0^2.\big(1 - \varepsilon.\cos(\omega.t)\big).x(t) = 0$$

où ω_0, ω et ε sont des constantes. On prendra par exemple ω_0 = 3 rad.s^{-1}, ω = 6,1 rad.s^{-1} et ε = 0,1.
2.a. Représenter l'évolution de $x(t)$ pour $t \in$[0 s, 100 s] avec $x(0) = 1$ (unités arbitraires) et $\frac{dx}{dt}(0) = 0$.
2.b. Représenter de même le portrait de phase sur le même intervalle de temps.

Exercice 5.— Chute avec frottements dans le champ de pesanteur.

On s'intéresse ici à la chute d'une parachutiste. L'altitude (distance au sol) à l'instant t de cette parachutiste est notée $z(t)$, et elle est lâchée depuis un avion depuis une hauteur $z(0) = h$, avec une vitesse initiale nulle. On tient compte des frottements avec l'air et de la variation de la masse volumique $\rho(z)$ de l'air avec l'altitude. On admet que dans une atmosphère où la température est supposée uniforme égale en moyenne à 0°C, ρ décroît exponentiellement avec une distance caractéristique a = 8 000 m :

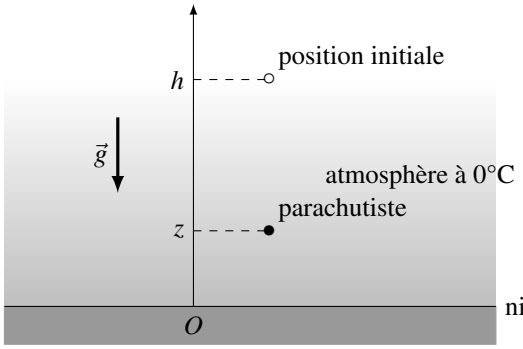
$$\rho(z) = \rho_0.e^{-z/a}$$

où ρ_0 = 1,3 kg.m^{-3} est la masse volumique de l'air au niveau du sol.

L'altitude z de la parachutiste vérifie l'équation différentielle :

$$\frac{d^2z}{dt^2} = -g + \frac{\rho_0.S.C_x}{2m} e^{-z/a} \left(\frac{dz}{dt}\right)^2$$

où g = 9,8 m.s^{-2}, C_x = 0,8 et S = 0,5 m^2 et m = 80 kg sont des constantes. Le terme en $\frac{\rho_0.S.C_x}{2m} e^{-z/a} \left(\frac{dz}{dt}\right)^2$ est dû aux frottements de l'atmosphère sur le parachutiste.



(a) Représentation de l'atmosphère.

(b) Félix BAUMGARTNER lors de son saut depuis une altitude de 39 km.

1. Écrire une fonction `trajectoire(h, dt = 0.01)` où `h` est l'altitude initiale, retournant la suite des altitudes, la suite des vitesses, la durée de la chute ainsi que la vitesse maximale atteinte de la parachutiste pendant sa chute obtenue par résolution numérique de l'EDO d'ordre 2 satisfaite par l'altitude z avec un pas de `dt`.
2. Grapher les trajectoires pour h variant sur la gamme d'altitudes 10000,20000,30000,40000,50000. On fera un graphique pour les altitudes, un graphique pour les vitesses. Expérimenter en variant la valeur du pas de résolution.
3. Selon ce modèle, d'atmosphère dite isotherme à 0°C, la vitesse du son est indépendante de l'altitude z et vaut environ c = 330 m.s^{-1}.

Déterminer (à environ 1 m près) la plus petite valeur de h permettant au parachutiste de franchir le mur du son lors de sa chute.

Exercice 6.— Modélisation de la propagation d'une épidémie à l'aide du modèle à compartiments (concours Mines-Ponts, session 2016).

Dans cet exercice, on suppose que la bibliothèque `numpy` a été importée par `import numpy as np`.

L'étude de la propagation des épidémies joue un rôle important dans les politiques de santé publique. Les modèles mathématiques ont permis de comprendre pourquoi il a été possible d'éradiquer la variole à la fin des années 1970 et pourquoi il est plus difficile d'éradiquer d'autres maladies comme la poliomyélite ou la rougeole. Ils ont également permis d'expliquer l'apparition d'épidémies de grippe tous les hivers. Aujourd'hui, des modèles de plus en plus complexes et puissants sont développés pour prédire la propagation d'épidémies à l'échelle planétaire telles que le SRAS, le virus H5N1, le virus Ebola ou la nomophobie. Ces prédictions sont utilisées par les organisations internationales pour établir des stratégies de prévention et d'intervention.

Les modèles compartimentaux sont des modèles déterministes où la population est divisée en plusieurs catégories selon leurs caractéristiques et leur état par rapport à la maladie. On considère dans cette partie un modèle à quatre compartiments disjoints : sains (S, « *susceptible* »), infectés (I, « *infected* »), rétablis (R, « *recovered* », ils sont immunisés) et décédés (D, « *dead* »). Le changement d'état des individus est gouverné par un système d'équations différentielles obtenues en supposant que le nombre d'individus nouvellement infectés (c'est-à-dire le nombre de ceux qui quittent le compartiment S) pendant un intervalle de temps donné est proportionnel au produit du nombre d'individus infectés avec le nombre d'individus sains.

En notant $S(t)$, $I(t)$, $R(t)$ et $D(t)$ la fraction de la population appartenant à chacune des quatre catégories à l'instant t, $t \in I$, l'intervalle temporel d'étude de l'épidémie, on obtient le système :

$$\forall t \in I, \begin{cases} \frac{dS}{dt}(t) &= -rS(t)I(t) \\ \frac{dI}{dt}(t) &= rS(t)I(t) - (a+b)I(t) \\ \frac{dR}{dt}(t) &= aI(t) \\ \frac{dD}{dt}(t) &= bI(t) \end{cases} \tag{1}$$

avec r le taux de contagion, a le taux de guérison et b le taux de mortalité. On suppose qu'à l'instant initial $t = 0$, on a $S(0) = 0{,}95$, $I(0) = 0{,}05$ et $R(0) = D(0) = 0$.

1. Préciser un vecteur X et une fonction f (en donnant son domaine de définition et son expression) tels que le système différentiel (1) s'écrive sous la forme :

$$\frac{dX}{dt} = f(X)$$

2. Compléter la ligne marquée `#A COMPLETER` du code suivant (on précise que `np.array` permet de créer un tableau `numpy` à partir d'une liste donnant ainsi la possibilité d'utiliser les opérateurs algébriques : somme de tableaux, produit par un scalaire, etc.) :

```
def f(X):
    """Fonction définissant l'équation différentielle"""
    global r,a,b
    return #A COMPLETER

tmax = 25.
r = 1.
a = 0.4
b = 0.1
X0 = np.array([0.95,0.05,0.,0.])

N = 250
dt = tmax/N

# Initialisations
t = 0.
X = X0
tt = [t]
XX = [X]

for i in range(N):
    t = t+dt
    X = X + dt*f(X)
    tt.append(t)
    XX.append(X)
```

3. La figure 2 représente les quatre catégories en fonction du temps obtenues en effectuant deux simulations : la première avec $N = 7$ correspond aux points (cercle, carré, losange, triangle) et la seconde avec $N = 250$ correspond aux courbes. Expliquer la différence entre ces deux simulations. Quelle simulation a nécessité le temps de calcul le plus long ?

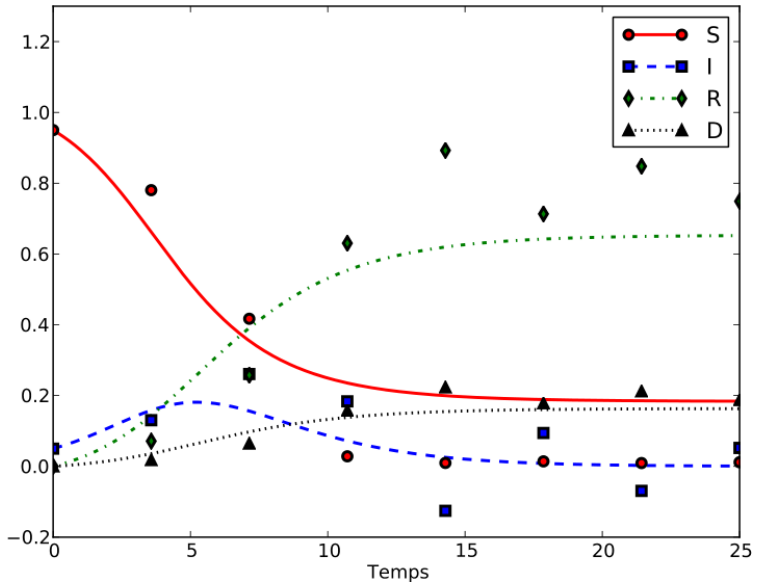


FIGURE 2 – Représentation graphique des quatre catégories S, I, R et D en fonction du temps pour $N = 7$ (points) et $N = 250$ (courbes).

En pratique, de nombreuses maladies possèdent une phase d'incubation pendant laquelle l'individu est porteur de la maladie mais ne possède pas de symptômes et n'est pas contagieux. On peut prendre en compte cette phase d'incubation à l'aide du système à retard suivant :

$$\begin{cases} \frac{dS}{dt}(t) &= -rS(t)I(t-\tau) \\ \frac{dI}{dt}(t) &= rS(t)I(t-\tau) - (a+b)I(t) \\ \frac{dR}{dt}(t) &= aI(t) \\ \frac{dD}{dt}(t) &= bI(t) \end{cases} \tag{2}$$

où τ est le temps d'incubation. On suppose alors que pour tout $t \in [-\tau, 0]$, $S(t) = 0,95$, $I(t) = 0,05$ et $R(t) = D(t) = 0$.

En notant *tmax* la durée des mesures et N un entier donnant le nombre de pas, on définit le pas de temps $dt = tmax/N$. On suppose que $\tau = p \times dt$ où p est un entier ; ainsi p est le nombre de pas de retard.

Pour résoudre numériquement ce système d'équations différentielles à retard (avec `tmax = 25`, `N = 250` et `p = 50`), on a écrit le code suivant :

```
def f(X,Itau):
    """Fonction définissant l'équation différentielle
    Itau est la valeur de I(t-p*dt)"""
    global r,a,b
    #A COMPLETER
    return #A COMPLETER

# Paramètres fonction
r = 1. ; a = 0.4 ; b = 0.1
#Paramètres Euler à retard/Cauchy
tmax = 25. ; N = 250 ; dt = tmax/N ;p = 50
t = 0.
X0 = np.array([0.95,0.05,0.,0.]) #X0
tt = [t]
XX = [X0]
# Méthode d'Euler
for i in range(N):
    t = t+dt
    #A COMPLETER
    tt.append(t)
    XX.append(X)
```

4. Compléter les blocs marqués `#A COMPLETER` du code précédent (utiliser autant de lignes que nécessaire).