Programmation Python pour le Calcul Scientifique

Examen (2h)

07/03/2020.

Prof: Mohamed El Ghmary
mohamed.elghmary@um5s.net.ma

***

NB : *1. Si, au cours de l''épreuve, vous repérez ce qui vous semble être une erreur d'énoncé, vous le signalez sur votre copie et poursuivez votre composition en expliquant les raisons des initiatives que vous avez amené à prendre.*
2. La qualité de la rédaction, la clarté et la précision des raisonnements entreront pour une part importante dans l'appréciation des copies.

Exercice 01

Nous allons nous intéresser dans ce problème, à quelques méthodes permettant de résoudre numériquement les équations différentielles du premier ordre, avec une condition initiale, sous la forme:

$$\begin{cases} y'(t) = f(y(t),t) \\ y(t_0) = y_0 \end{cases}$$

On note $\mathbf{I} =]\, t_0 , t_0 + T[$ l'intervalle de résolution. Pour un nombre de nœuds **N** donné, soit $t_i = t_0 + i*h$, avec $i = 0,1,2,\ldots,N$ une suite de nœuds de **I** induisant une discrétisation de **I** en sous-intervalles $I_i = [t_i, t_{i+1}]$. La longueur **h** de ces sous-intervalles est appelée pas de discrétisation, le pas **h** de discrétisation est donné par $h = T/N$. Soit y_i l'approximation au nœud t_i de la solution exacte $y(t_i)$. Les méthodes de résolution numériques, étudiées dans ce problème, s'écrire sous la forme :

$$\begin{cases} y_{i+1} = y_i + h * g(y_i, t_i) \\ t_{i+1} = t_i + h \end{cases}$$

Avec $g(y,t) = a * f(y(t),t) + b * f(y + h * f(y,t), t+h)$ où a,b sont des constantes comprises entre 0 et 1.

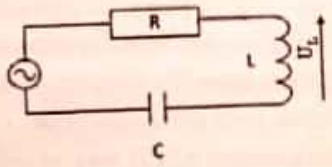
Q1. Pour quelles valeurs du couple (a,b) retrouve-t-on la méthode **d'Euler** ?

Q2. Écrire la fonction **def euler(f, t_0,T,y_0,N)** : qui prend en paramètres la fonction f, la condition initiale t_0, la valeur finale du temps **T**, y_0 la valeur de f en t_0 et le nombre de nœuds **N** et qui retourne la liste $[y_0, y_1, y_2, \ldots, y_{n-1}]$ la solution de l'équation différentielles $y'(t) = f(y(t),t)$.

Dans la suite on considère une deuxième méthode dite de **Heun**, cette méthode correspond aux valeurs du couple **(a,b)=(1/2,1/2).**

Q3. Écrire la fonction **def heun(f, t_0,T,y_0,N)** : qui prend en paramètres la fonction f, la condition initiale t_0, la valeur finale du temps **T**, y_0 la valeur de f en t_0 et le nombre de nœuds **N** et qui retourne la liste $[y_0, y_1, y_2, \ldots, y_{n-1}]$ la solution de l'équation différentielles $y'(t) = f(y(t),t)$.

Application au circuit RLC:



On souhaite travailler sur un circuit **RLC** en série qui sera constitué des éléments suivants : une résistance R(1Ω), une inductance L(1H), une capacité C(1F) et un générateur qui délivrera une source de tension sinusoïdale $V(t) = \cos(2.\pi.t)$.

On s'intéresse à la tension U_L aux bornes de la bobine qui satisfait l'équation différentielle :

$$\ddot{U}_l + \dot{U}_l + U_l = -4\pi^2 \cos(2\pi t) \qquad (E)$$

Q4. Montrez que l'équation **(E)** peut s'écrire sous la forme
$z'(t) = F(z(t),t)$ avec $F: z, t \rightarrow F(z, t)$ est une fonction à préciser (z = (z[0] , z[1])).

On suppose qu'on a exécuté les lignes de code suivantes :

from scipy.integrate import odeint
from pylab import *

Q5. Implémenter en Python l'équation différentielle (E) en utilisant la fonction odeint du module scipy.integrate, la méthode d'Euler et celle de Heun. On affichera dans une même fenêtre le résultat de ses trois méthodes avec les valeurs suivantes : **N=1000, t_0=0,T=3, $U_L(0)$=0 et $U'_L(0)$=0.**

- On ajoutera quatre types d'informations :
- le titre de la figure : 'circuit RLC, tension bobine'
- un label associé à l'axe des abscisses :'temps (s)'
- un label associé à l'axe des ordonnées :'tension (mV)'
- une légende associée au graphique :'odeint' associé à la méthode d'odeint
- une légende associée au graphique :'euler' associé à la méthode d 'Euler
- une légende associée au graphique : 'heun' associé à la méthode d'Heun

Exercice 02 (Méthode de la sécante)

La méthode de la sécante est un algorithme de recherche d'une racine (zéro) d'une fonction. Soit f cette fonction. La formule de récurrence est donnée par la relation :

$$X_{n+1} = X_n - \frac{X_n - X_{n-1}}{f(X_n) - f(X_{n-1})} f(X_n)$$

Pour appliquer cette méthode on choisit deux points x_0 et x_1 pour initialiser le processus de calcul et on arrête l'algorithme lorsque l'écart entre deux valeurs consécutives de la suite est majoré par une valeur fixée, par exemple 10^{-10}.

Q6. Expliquer pourquoi la méthode de la **sécante** est une méthode dérivée de la méthode de **Newton**.

Q7. Les deux points x_0 et x_1 peuvent-ils être choisis d'une manière quelconque ? Justifier votre réponse.

Q8. Écrire la fonction **def secante(f , x_i , x_j , eps=10**(-10))**: qui permet de rechercher le zéro de l'équation **f(x)=0** par la méthode de la sécante, avec deux méthodes:

a) **Méthode itérative**
b) **Méthode récursive**

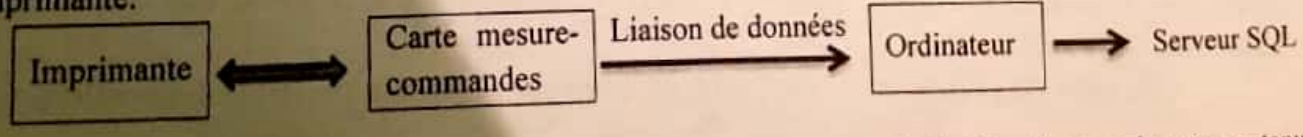
Q9. Ecrire **un programme en langage Python** qui demande à l'utilisateur de lire un nombre positif **N** au clavier, puis affiche la racine carré de **N (sans utiliser ni la fonction sqrt ni la fonction pow ni l'opérateur **)**.

EXERCICE 03 :

Les imprimantes sont des systèmes mécatroniques fabriqués en grande série dans des usines robotisées. Pour améliorer la qualité des produits vendus, il a été mis en place différents tests de fin de chaine pour valider l'assemblage des produits. Pour un de ces tests, un opérateur connecte l'outil de test sur la commande du moteur de déplacement de la tête d'impression et sur la commande du moteur d'avance papier. Une autre connexion permet de récupérer les signaux issus des capteurs de position.

Différentes commandes et mesures sont alors exécutées. Ces mesures sont envoyées par liaison de données sous la forme d'une suite de caractères ASCII vers un ordinateur.

Cet ordinateur va effectuer différentes mesures pour valider le fonctionnement de l'électromécanique de l'imprimante.



La suite des valeurs de mesure du courant en Ampère du moteur de la tête d'impression est contenue dans une liste. Les mesures ont été effectuées toutes les 2 ms. Ces mesures sont disponibles dans la liste **mesure**. Deux traitements permettent de valider le fonctionnement de l'imprimante :

Le calcul de la valeur moyenne I_{moy} du signal $I(t)$ sur la durée d'acquisition.

$$I_{moy} = \frac{1}{t_{final}} \int_0^{t_{final}} I(t)dt$$

Le calcul de l'écart type **Iec** du signal I(t) sur la durée d'acquisition.

$$I_{ec} = \sqrt{\frac{1}{t_{final}} \int_0^{t_{final}} (I(t) - I_{moy})^2 dt}$$

Q10. Ecrire une fonction en langage Python qui retourne I_{moy} après l'avoir calculée par la méthode des trapèzes.

Q11. Ecrire une fonction en langage Python qui retourne I_{ec} après l'avoir calculé en utilisant la fonction précédente.

Base de données

Lorsqu'une imprimante satisfait les critères de validation, elle est enregistrée dans la table production avec son numéro de série, la date et l'heure de sortie de production ainsi que son type.

production

Num	nSerie	dateProd	type
20	230-588ZX2547	2012-04-22 15-52-12	JETDESK-1050
21	230-588ZX2549	2012-04-22 15-53-24	JETDESK-3050
:	:	:	:

Q12. Sachant que le chemin de cette base de données dans le disque dur est "C ://TestsImpr.sqlite", et en utilisant le module sqlite3 de python, écrire un programme python permettant de se connecter à cette base de données, d'exécuter la requête "select * from production", de récupérer le résultat et de :

a) l'afficher sur l'écran.
b) Le stocker dans un fichier.

EXERCICE 04 :

Q13. Ecrire une fonction $f(x)$: qui retourne la valeur de $x^2\cos(x^5\exp(x^2)) - x - 1$.

Q14. Écrire la fonction ***derivee (f,x0,h)*** qui permet de calculer la valeur approchée de $f'(x_0)$ (la dérivée de f(x) en x_0).

Q15. Construire par compréhension de liste les listes suivantes :

a) La liste $[x_0, x_1, x_2, \ldots, x_i, \ldots, x_{n-1}]$
b) La liste $[f(x_0), f(x_1), f(x_2), \ldots, f(x_i), \ldots, f(x_{n-1})]$
c) La liste $[f'(x_0), f'(x_1), f'(x_2), \ldots, f'(x_i), \ldots, f'(x_{n-1})]$

Avec $x_i = -10+i*h$, $-10 <= x_i <= 10$ et $h=0.001$.

Q16. Écrire les lignes du code en python pour tracer dans une même fenêtre la courbe des fonctions $f(x)$ et $f'(x)$. On ajoutera quatre types d'informations:

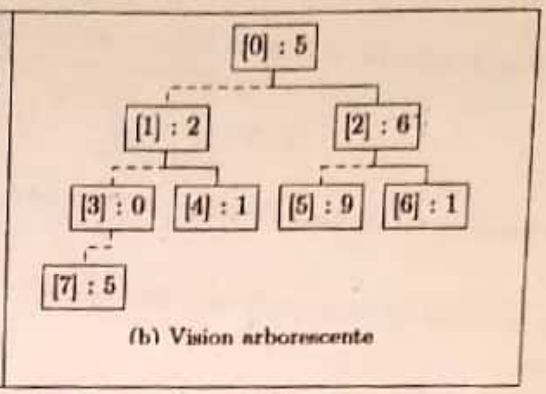
- Le titre de la figure
- Le label associé à l'axe des abscisses
- Le label associé à l'axe des ordonnées
- La légende associée à la fonction $f(x)$
- La légende associée à la fonction $f'(x)$.

Exercice 05(tri par tas(tri maximier))

Le but de l'exercice est l'écriture d'un programme de tri de tableaux basé sur la notion de tas.
Les questions se suivent logiquement, mais beaucoup sont indépendantes.

indice	0	1	2	3	4	5	6	7
T[indice]	5	2	6	0	1	9	1	5

(a) Vision tabulaire



(b) Vision arborescente

Figure 1 - Deux vues différentes d'un même tableau

La **figure 1** montre qu'un même tableau peut être dessiné avec des cases contigües, ou bien avec des cases dispersées dans une arborescence. Avec la vue contigüe, on utilise généralement une variable i qui parcourt les indices du tableau. Avec la vue arborescente, on peut évidemment utiliser une variable i qui parcourt les indices du tableau, mais on utilise également trois fonctions qui permettent de suivre les liens bidirectionnels (réels ou virtuels) de l'arborescence :

- **gauche(indice)** représente les liens pointillé du haut vers le bas de l'arborescence.
  Par exemple, dans la figure 1b, gauche(1)=3, gauche(4)=9 et gauche(2)=5.
- **droite(indice)** représente les liens en trait plein du haut vers le bas de l'arborescence.
  Par exemple, dans la figure 1b, droite(1)=4, droite(3)=8 et droite(0)=2.
- **pere(indice)** représente les liens du bas vers le haut de l'arborescence.
  Par exemple, dans la figure 1b, pere(4)=1, pere(7)=3 et pere(2)=0.

Q17. Des fonctions élémentaires pour se déplacer dans un tableau.

a) **def gauche (i) :** retourne un entier **g** tel qu'il existe un lien du haut vers le bas reliant les indices **i** à **g**.
b) **def droite(i) :** retourne un entier **d** tel qu'il existe un lien du haut vers le bas reliant les indices **i** à **d**.
c) **def pere(i):** retourne un entier **p** tel qu'il existe un lien du bas vers le haut reliant les indices i à **p**.

Q18. Construction d'un tas à partir d'un tableau.

a) **def maximum(T,i,limite):** retourne l'indice (inferieur à limite) de la plus grande des trois valeurs T[i], T[gauche(i)] et T[droite(i)].
b) **def entasser(T,i,limite):** échange des valeurs du tableau de haut en bas, en suivant une branche de l'arborescence. Cela a pour effet de faire descendre des petites valeurs, et de faire monter les grandes valeurs. Il est donc possible de construire un tas, en itérant cet algorithme sur les indices décroissants du tableau.
c) **def construireTas(T):** transforme un tableau en un tas(En utilisant entasser(T,i,limite)).
d) **def trierTas(T):** permet de trier un tas
e) **def estunTas(T)** :qui retourne **True** si le tableau T est un tas, **False** sinon.
f) **def triParTas(T):** permet de trier le tableau

Remarque:

- *Un tas est un tableau tel que pour tous les indices i >0, la valeur de T[i] <= à celle de T[pere(i)]*
- *Dans un tas, la valeur maximale est à la racine de l'arborescence, donc en T[0]. Dans le tableau trié, cette valeur doit être en T[len(T)-1]. Il suffit donc d'échanger ces deux valeurs pour progresser vers la solution. Une fois cet échange fait, si l'on exclut la dernière valeur du tableau, le tas est peu changé. En fait entasser(T,0,len(T)-1) va créer un nouveau tas pour les valeurs du tableau dont les indices sont inférieurs à limite = len(T)-1. Il suffit donc d'itérer ces deux étapes (échange, entasser) pour trier un tas.*