Introduction à l'analyse numérique
Année : 2018-2019
Formation : L2 Mathématiques

TP noté 1 : équations différentielles et recherche de zéros

Ce sujet est composé en trois parties indépendantes. Pour chaque partie il faudra créer deux fichiers dans lesquels vous enregistrerez votre code : un fichier en .sci qui contiendra toutes les fonctions de la partie et un fichier en .sce qui contiendra les autres commandes, par exemple les commandes pour afficher graphiquement vos résultats. Lorsqu'une réponse doit être apportée sous forme de texte, vous l'écrirez en commentaire (précéder la ligne de "//"). N'hésitez pas à utiliser les commentaires pour faire diverses remarques, par exemple sur de possibles problèmes dans vos fonctions. **Des points seront réservés à la qualité et la présentation du code.**

A la fin du TP vous devez envoyer vos six fichiers (partie1.sci ; partie1.sce ; partie2.sci ; partie2.sce ; partie3.sci ; partie3.sce) par mail à l'adresse suivante : florian.le-manach@math.u-bordeaux.fr. N'oubliez pas d'indiquer votre nom et prénom (et celui de votre binôme si vous faites le TP à deux) dans le mail.

1 La méthode de Newton en deux dimensions

L'objectif de cette partie est de minimiser une fonction classique en optimisation, appelée souvent "banane de Rosenbrock". Cette fonction $J : \mathbb{R}^2 \longrightarrow \mathbb{R}$ est définie par

$$J(x,y) = (x-1)^2 + (x^2-y)^2 + 3 \tag{1}$$

Minimiser la fonction J revient à déterminer $(x^*, y^*) \in \mathbb{R}^2$ tel que

$$J(x^*, y^*) = \min \left\{ J(x,y),\ (x,y) \in \mathbb{R}^2 \right\}.$$

Nous admettons qu'un tel couple (x^*, y^*) est unique. De plus, selon la règle de Fermat, on a $\nabla J(x^*, y^*) = 0$.

Ainsi le problème de minimisation de la fonction J équivaut au problème de **recherche de zéro de son gradient ∇J**.

On rappelle que le gradient de la fonction J en un point (x,y) est donné par

$$\nabla J(x,y) = \begin{pmatrix} \frac{\partial J}{\partial x}(x,y) \\ \\ \frac{\partial J}{\partial y}(x,y) \end{pmatrix} \tag{2}$$

et la matrice Jacobienne du gradient ∇J (ou la matrice Hessienne de J) est donnée par :

$$D_{\nabla J}(x,y) = \begin{pmatrix} \frac{\partial^2 J}{\partial x^2}(x,y) & \frac{\partial^2 J}{\partial x \partial y}(x,y) \\ \\ \frac{\partial^2 J}{\partial y \partial x}(x,y) & \frac{\partial^2 J}{\partial y^2}(x,y) \end{pmatrix} \tag{3}$$

1. Définissez la fonction function z=J(x,y) qui prend en paramètre deux réels x, y et qui renvoie la valeur $J(x,y)$ définie dans (1). Ensuite, à l'aide de la commande fplot3d(X,Y,J), tracez la surface définie par la fonction J dans $[-2,2]^2$. On utilisera un pas de $0,1$.

2. Définissez la fonction function G=Grad(x,y) qui prend en paramètre deux réels x, y et qui renvoie la valeur du vecteur $\nabla J(x,y)$ comme définie dans (2).
   Remarque : il faudra calculer à la main les dérivées partielles de J.

3. Définissez la fonction function D=Hess(x,y) qui prend en paramètre deux réels x, y et qui renvoie la valeur de la matrice $D_{\nabla J}(x, y)$ (de taille 2×2) comme définie dans (3).
Remarque : il faudra également calculer à la main les dérivées partielles secondes de J.

4. Rappelez en commentaire la formule de Newton en deux dimensions. On va utiliser cette méthode pour trouver un zéro de ∇J. Définissez, à l'aide des questions 2 et 3, la fonction function R=Newton(x0,y0,eps) qui prend paramètre deux réels $x0$ et $y0$ correspondant à l'abscisse et à l'ordonnée du premier élément de la suite de Newton et une précision *eps* et qui renvoie R=[X ;Y] où

- X est la liste des abscisses des éléments de la suite de Newton ;
- Y est la liste des ordonnées des éléments de la suite de Newton ;
- la fonction s'arrête lorsque $\|\nabla J(x_n, y_n)\|^2 \leq eps$.

Remarque : ici $\| \cdot \|$ correspond à la norme euclidienne : $\|(a, b)\|^2 = a^2 + b^2$.

5. Testez la fonction Newton définie à la question précédente, avec $x_0 = -2$, $y_0 = 7$ et $eps = 10^{-15}$. Tracez à l'aide de plot2d la trajectoire des points obtenus.

6. En déduire en commentaire le point de minimum de la fonction J. Que vaut J en ce minimum ?

2 Résolution d'une équation différentielle

Dans cette partie on résout numériquement, par la méthode d'Euler explicite, l'équation différentielle suivante

$$(E) \ : \ y'(t) = y(t) + 5\cos(5t)e^{t/2} - 0,5\sin(5t)e^{t/2}, \quad y(0) = 0.$$

Ensuite on comparera cette solution numérique avec la solution exacte.

1. Déterminez et définissez dans Scilab la fonction function z=f(t,y) associée à cette équation différentielle de sorte à avoir $y'(t) = f(t, y(t))$.

2. Définissez la fonction function L=eulerE(f,y0,tf,dt) qui prend en paramètre la fonction f d'une equation $y'(t) = f(t, y(t))$, la condition initiale y_0 au temps $t = 0$, le temps final t_f et un pas de temps dt. Cette fonction renvoie la liste des y_n, pour $0 \leq n \leq \frac{t_f}{dt}$, vérifiant la formule d'Euler explicite.

Si vous n'arrivez pas à faire cette fonction, rappelez la formule d'Euler explicite en commentaire.

3. A l'aide des deux premières questions, résolvez numériquement l'équation (E) avec un pas de temps $dt = 0,01$ jusqu'au temps $t_f = 9$. Affichez la courbe de la solution sur un graphique.

On peut vérifier qu'une solution exacte de l'équation (E) est la fonction $g(t) = \sin(5t)e^{t/2}$.

4. Affichez la courbe de la fonction g sur un graphique.

On veut alors comparer l'erreur de l'approximation de la solution avec la méthode d'Euler explicite.

5. Affichez sur un graphique l'erreur entre la solution obtenue avec la méthode d'Euler et la solution exacte. Si y est la solution obtenue avec la méthode d'Euler et si g est la solution exacte, on appelle "erreur" la fonction $t \mapsto |y(t) - g(t)|$.
Qu'observe-t-on ? Commentez ce résultat.

3 Calcul de puissance

L'objectif de cette partie est de construire une fonction puissance qui permet, pour $x \in \mathbb{R}$ et $n \in \mathbb{N}$, de calculer x^n. Évidemment dans toute cette partie il sera interdit d'utiliser la commande scilab `x^n` pour calculer x^n.

Les trois première questions peuvent être traitées de manière indépendante.

Dans un premier temps on va construire une fonction puissance de manière naïve qui effectuera n (ou $n-1$) multiplications pour calculer x^n.

1. Définissez la fonction `function r=puiss1(x,n)` qui prend en paramètre un réel x et un entier naturel n et qui renvoie x^n. On rappelle qu'il est interdit d'utiliser le symbole ^ dans cette fonction.

On cherche maintenant a améliorer le calcul de x^n en effectuant le moins d'opération possible. Pour cela on s'intéresse à la décomposition en base 2 de n.

On rapelle que tout nombre entier n se décompose de manière unique en

$$n = \sum_{i=0}^{k} c_i 2^i = c_0 + c_1 \times 2 + c_2 \times 2^2 + \cdots + c_k 2^k$$

avec $c_i = 0$ ou $c_i = 1$.

2. Définissez la fonction `function T=decompo2(n)` qui prend en paramètre un entier naturel n et qui renvoie la liste T=[c_0, c_1,..., c_k] de sa décomposition en base 2.
Indication : On commence par déterminer c_0 en regardant la parité de n : si n est pair alors $c_0 = 0$ et si n est impair alors $c_0 = 1$. Ensuite pour déterminer c_1 on recommence cette opération avec $n/2$ si n est pair et $(n-1)/2$ si n est impair. On itère ce procéder jusqu'à obtenir $n = 0$.

3. Définissez la fonction `function L=puissListe2(x,k)` qui prend en paramètre une réel x et un entier naturel k et qui renvoie L=[x, x^2, x^{2^2},..., x^{2^k}]. Cette fonction ne doit pas faire plus de k multiplications. On rappelle qu'il est interdit d'utiliser le symbole ^ dans cette fonction.
Indication : $x^{2^{i+1}} = x^{2^i} \times x^{2^i}$.

On revient maintenant au problème initial, à savoir calculer x^n avec le moins d'opération possible. Si la décomposition de n en base 2 est

$$n = \sum_{i=0}^{k} c_i 2^i$$

alors on observe que

$$x^n = x^{c_0} \times (x^2)^{c_1} \times (x^{2^2})^{c_2} \times \cdots \times (x^{2^k})^{c_k}.$$

4. A l'aide de l'observation ci-dessus et à l'aide des fonctions définies dans les questions 2 et 3, définissez la fonction `function r=puiss2(x,n)` qui prend en paramètre une réel x et un entier naturel n et qui renvoie r=x^n.

5. Combien faut-il de multiplications pour calculer x^{12} à l'aide de la seconde méthode ?

Les questions suivantes sont hors barème.

6. Comparez, avec la commande `timer()`, le temps d'exécution des commandes `puiss1(2,10000000)` et `puiss2(2,10000000)`. Quelle fonction est la plus rapide ?

7. Écrire une fonction `function r=puiss3(x,n)` basée sur la fonction `puiss2` mais cette fois sans utiliser les fonctions des question 2 et 3 et sans utiliser de listes. Le but de cette question est de réduire l'utilisation de la mémoire.