Université de Rennes 1 Master 1 CSA
UFR Mathématiques Méthodes numériques

TP 2 - Résolution numérique d'une équation

Partie 1 : Dimension 1

Créer un répertoire TP2 (dans le répertoire où vous regroupez tous les fichiers concernant ce cours) et vous y placer pour lancer Matlab.

Exercice 1.

1) **Méthode de dichotomie**

Écrire une fonction `[y]=f(x)` pour $f(x) = x^2 - 3$ et la tester pour différentes valeurs de x. Pour cela, utiliser l'éditeur de Matlab pour enregistrer le texte suivant dans le fichier de nom `f.m` (le même nom que celui de la fonction) dans le répertoire TP2 :

```
function [y]=f(x)
% f(x) = x^2-3
y=x^2-3 ;
```

Écrire une fonction `[r,nbit]=dichotomie(f,a,b,eps)` qui, étant donné une fonction f, des valeurs initiales a et b et une précision `eps`, calcule une racine r de la fonction f entre a et b par la méthode de dichotomie (ou bissection) à une précision `eps`, avec `nbit` le nombre d'itérations. Utiliser une boucle `while` :

```
while (abs(b-a) > eps)
   ...
end
```

Appliquer à l'exemple $f(x) = x^2 - 3$, $a = 0$, $b = 2$ avec `eps= 1e-2, 1e-7 et 1e-14` puis pour la fonction $cos(x)$, en sauvegardant toutes les instructions dans un script que l'on exécutera ensuite. L'appel de la fonction `dichotomie` se fera pour $f(x)$ par

```
a=0 ; b=2 ; eps=1e-2 ; [r,nbit]=dichotomie(@f,a,b,eps)
```

et pour $\cos(x)$ (qui est une fonction prédéfinie de Matlab) par

```
[r,nbit]=dichotomie(@cos,a,b,eps)
```

2) **Méthodes itératives**

On définit les fonctions suivantes :

$$g_1(x) = \frac{5x^3 - 3x}{6x^2 - 6}\ ,\ g_2(x) = \frac{2x^3}{3x^2 - 3}\ , g_3(x) = \frac{x^3 + 9x}{3x^2 + 3}\ .$$

a. Vérifier que $\sqrt{3}$ est un point fixe de chacune de ces fonctions.
b. Écrire une fonction `[xk]=iter(g,x0,N)` qui calcule N itérés $(x_1, \ldots, x_N)$ de la fonction g pour la valeur intiale x_0. Comme résultat, x_0 et tous les itérés sont stockés dans un vecteur `xk`. Utiliser une boucle `for` :

```
for i=1 :N
   ...
end
```

c. Pour chacune des fonctions g_i, étudier la méthode itérative correspondante de calcul d'une valeur approchée de $\sqrt{3}$ en répondant aux questions suivantes :

i) Combien d'itérations sont-elles nécessaires pour avoir une précision de 7 chiffres ? (on choisira une même valeur initiale pour l'étude de toutes ces fonctions)

ii) Quel est l'ordre de la méthode ? Il peut être utile d'utiliser des commandes de la forme

```
>> r3=sqrt(3) ; x0=2 ; N=5 ;
>> xk=iter(@g3,x0,N)
>> log(abs(xk(2 :N+1)-r3)) ./ log(abs(xk(1 :N)-r3))
```

3) **Méthode de Newton**

Écrire une fonction `[r,nbit]=newton(f,fp,x0,eps,Nmax)` calculant une valeur approchée d'une racine de la fonction f. Comme paramètres d'entrée on donne la fonction `f`, sa fonction dérivée `fp`, la valeur initiale `x0`, la précision `eps` et le nombre maximal d'itérations `Nmax`. L'itération s'arrête si $|f(x_n)| < \varepsilon$ ou si le nombre maximal d'itérations est atteint. Les paramètres de sortie sont le dernier itéré `r` et le nombre d'itérations `nbit`. Utiliser une boucle `while` :

```
fx=f(x0) ;
while (abs(fx) > eps) & (nbit < Nmax)
   ...
end
```

Faire des tests avec $f(x) = x^4 - 3x^2$ et des valeurs initiales `x0 = 1.1:0.1:2`.

Partie 2 : Dimension 2

Exercice 2.

1) Écrire une fonction `[r,nbit]=newton_rd(f,Jf,x0,eps,Nmax)` calculant une valeur approchée d'une racine de la fonction f. Comme paramètres d'entrée on donne la fonction `f`, sa matrice jacobienne `Jf`, la valeur initiale `x0`, la précision `eps` et le nombre maximal d'itérations `Nmax`. L'itération s'arrête si $\left\|x^{(k+1)} - x^{(k)}\right\|_2 < \varepsilon$ (on pourrait choisir comme en dimension 1 : $\left\|f(x^{(k)})\right\|_2 < \varepsilon$) ou si le nombre maximal d'itérations est atteint. Les paramètres de sortie sont le dernier itéré r et le nombre d'itérations `nbit`. Utiliser une boucle `while` :

```
while ((norm(xkp1-xk)>eps) & (nbit<nmax))
   ...
end
```

La résolution du système linéaire $Ax = b$ s'écrivant en `Matlab` : `x = A \ b`, si l'on note `J(xk)` la matrice jacobienne de f en x_k, l'itération de la méthode de Newton comportera une instruction du type :

```
xkp1 = xk - J(xk)\f(xk) ;
```

2) Appliquer la méthode de Newton à la fonction

$$f(x) = \begin{pmatrix} (1 - x_1)(2 - x_2) \\ -1 + x_1^2 x_2^2 \end{pmatrix}$$

pour différentes valeurs de `x0`.

3) La fonction f ayant 4 racines s_1, s_2, s_3, s_4 ; selon la valeur choisie de `x0` l'algorithme convergera vers l'une ou l'autre de ces racines ou divergera. Nous allons en faire une visualisation graphique.

Nous définissons un quadrillage régulier du rectangle $[-1;\ 2.5]\times[-1;\ 2.5]$ d'abscisses `x01=-1 :pas :2.5` et d'ordonnées `x02=-1.5 :pas :2.5`. Écrire un script `fractal.m` qui, pour chaque point initial `x0=[x01(i) ; x02(j)]` applique la méthode de Newton avec `Nmax=7`, `eps=1e-6` et `pas=0.1`, puis sauvegarde dans un tableau `c(j,i)` le numéro de la solution vers laquelle a convergé la méthode et `0` s'il n'y a pas convergence.

Utiliser deux boucles `for` imbriquées :

```
n=length(x01) ;m=length(x02) ;c=zeros(m,n) ;
for i=1 :n
   for j=1 :m
      ...
   end
end
```

Utiliser l'instruction `pcolor(x01,x02,c)` ou `contourf(x01,x02,c)` pour créer le graphique.

Pour obtenir un graphique plus précis, il suffit de diminuer `pas` (l'adapter à la vitesse de calcul de l'ordinateur !).

Faire un zoom sur une partie du graphique en diminuant les dimensions du rectangle et la valeur de `pas` : par exemple $[0.5;\ 0.7]\times[1;\ 1.2]$ avec `pas=0.001` ou $[0.8;\ 1.2]\times[0;\ 0.4]$ avec `pas=0.001`.

Partie 3 : Compléments

Exercice 3.

1) Reprendre la question 2) de l'exercice 1 avec les fonctions :

$$g_4(x) = \frac{x^3+x}{2x^2-2}\ ,\ g_5(x) = -\frac{x^3}{6} + \frac{3x}{2}\ ,\ g_6(x) = x + e^{\frac{3-x^2}{4}}\left(-\frac{x^3}{6} + \frac{x}{2}\right).$$