Lycée Louis-Le-Grand, Paris 2015/2016
MPSI 4 – Informatique commune
G. Dewaele, A. Troesch

TP n° 10 : Résolutions d'équations

Nous étudions dans ce TP plusieurs méthodes de calcul approché de solutions d'une équation du type $f(x) = 0$, f étant une fonction continue d'un intervalle I de $\mathbb{R}$ dans $\mathbb{R}$.

Exercice 1 – Toutes les méthodes implémentées devront être vérifiées à l'aide des fonctions $f_1 : x \mapsto x^2 - 2$, $f_2 : x \mapsto x - \mathrm{e}^{-x}$ et $f_3 = \sqrt{x} - \frac{1}{2}$, et une marge d'erreur de 10^{-10}.
On comparera également aux valeurs théoriques, lorsqu'elles sont calculables, et à la valeur obtenue à l'aide de la fonction `newton` du module `scipy.optimize`. La fonction `newton` prend en argument une fonction f et un réel x_0 initialisant la méthode, ainsi, qu'un troisième argument optionnel, devant être égal à la fonction dérivée f'. Si cet argument optionnel f' est donné, un zéro de f est calculé à l'aide de la méthode de Newton, sinon à l'aide de la méthode des sécantes.

1. **Dichotomie**

   Écrire une fonction `dichotomie` prenant en argument une fonction f, deux réels a et b et une marge d'erreur ε, et retournant un couple (x, n), tel que $x \in [a, b]$ et $f(x) = 0$, n étant le nombre de partages de l'intervalle $[a, b]$ qu'il a été nécessaire d'effectuer par la méthode de dichotomie. On commencera par un test permettant de s'assurer de l'existence d'une telle valeur de x.

2. **Méthode de la fausse position**

   (a) Même question avec une fonction `fausse_position` exploitant la méthode de la fausse position. On s'arrêtera lorsqu'on se sera assuré de l'existence d'un zéro ε-proche de la valeur obtenue.

   (b) Comparer les performances de `fausse_position` et de `dichotomie` sur les fonctions f_1, f_2, ainsi que sur la fonction $x \mapsto x^3$. Pour cette dernière fonction, considérer l'intervalle initial $[-\frac{1}{2}, 1]$, et la marge d'erreur $\varepsilon = 10^{-3}$.

3. **Méthode de la sécante**

   (a) Écrire une fonction `secante` prenant en paramètre une fonction f, deux réels a et b et une marge d'erreur ε, et retournant, en cas de succès, le couple (x, n) obtenu par la méthode de la sécante, x étant une valeur approchée près d'un zéro de f, n étant le nombre d'itérations nécessaires pour obtenir x, l'arrêt s'effectuant dès lors que deux valeurs successives sont ε-proches l'une de l'autre. On limitera le nombre d'itérations à 100. Si cette valeur est atteinte, on décrètera l'échec de la méthode.

   (b) En comparant aux valeurs théoriques et aux valeurs fournies par la fonction `newton`, discuter, de la validité du test d'arrêt.

   (c) Comparer les performances de cet algorithme et des précédents

4. **Méthode de Newton-Raphson**

   (a) Écrire une fonction `mon_newton` prenant en paramètre une fonction f, sa déridée f', un réel a et une marge d'erreur ε, et retournant, en cas de succès, le couple (x, n) obtenu de même que précédemment, mais cette fois par la méthode de Newton-Raphson.

   (b) En comparant aux valeurs théoriques et aux valeurs fournies par la fonction `newton`, discuter, de la validité du test d'arrêt.

   (c) Comparer les performances de cet algorithme et des précédents.

   (d) Trouver une fonction f admettant un zéro et un réel x_0, telle que la méthode de Newton associée soit bien définie, mais divergente.

5. **Dérivation numérique**

Afin de pouvoir exploiter la méthode de Newton-Raphson sans avoir à donner l'expression de la dérivée de f, nous cherchons ici à calculer numériquement la dérivée de f. Cette méthode n'est évidemment valable que si f est à variations régulières, dans un sens que nous ne précisons pas.

(a) Écrire une fonction `derivee` prenant en paramètre une fonction f et deux réels x et h, et retournant une valeur approchée de $f'(x)$, obtenue en considérant le taux d'accroissement entre $x-h$ et $x+h$.

(b) Écrire une fonction `trace_derivee` prenant en paramètre une fonction f et trois réels a, b et h, et traçant le graphe de f' sur l'intervalle $[a, b]$, h étant le réel considéré dans la question précédente pour le calcul de la dérivée (on consultera les TP précédents pour le tracé de graphes)

(c) Écrire une fonction `choix_de_h`, prenant en paramètre une fonction f, sa dérivée théorique f' et un réel x, et traçant le graphe de la fonction qui à y associe l'erreur faite sur le calcul de f' par la méthode précédente en utilisant une valeur de $h = 10^{-y}$. On fera varier y entre 4 et 8. Confirmer le choix optimal de h obtenu par le calcul dans le cours.

(d) Reprogrammer une nouvelle fonction `mon_newton_bis`, exploitant la méthode de Newton associée à la dérivation numérique.

Exercice 2 – Méthode de Newton dans le plan complexe

Étant donné un nombre complexe a, on considère le polynôme $P = (X-1)\left(X + \frac{1}{2} - a\right)\left(X + \frac{1}{2} + a\right)$ ainsi que son polynôme dérivé P'. À tout nombre complexe z on associe la suite $(u_n)_{n\in\mathbb{N}}$ définie par la condition initiale $u_0 = z$ et la relation de récurrence : $u_{n+1} = u_n - \dfrac{P(u_n)}{P'(u_n)}$.

Comme on peut s'y attendre, la suite $(u_n)_{n\in\mathbb{N}}$ va en général converger vers l'une des trois racines de P, mais dans certains cas ne va pas converger (voire cesser d'être définie à partir d'un certain rang).

Le but de cet exercice est de représenter une portion carrée du plan complexe $(\mathrm{Re}(z), \mathrm{Im}(z)) \in [u, v]^2$ dans laquelle chaque point d'affixe z sera coloré d'une façon différente suivant que la suite $(u_n)_{n\in\mathbb{N}}$ converge vers 1, $-\frac{1}{2} - a$, $-\frac{1}{2} + a$ ou ne converge pas.

Représentation graphique.

La portion carrée du plan sera représentée par une matrice $n \times n$, la valeur de n permettant de choisir une résolution plus ou moins fine compte tenu du temps qui sera nécessaire aux calculs. Chaque case de ce tableau correspond donc à une valeur initiale $z \in \mathbb{C}$ et devra contenir au final l'un des quatre entiers $0, 1, 2, 3$ représentant respectivement les quatre situations décrites ci-dessus.

Le tableau initial sera créé à l'aide de la fonction :

> `numpy.zeros((n,n))` qui définit une matrice $n \times n$ dont chacune des cases contient la valeur 0.

Une fois le tableau `tab` rempli, la fonction :

> `matplotlib.pyplot.matshow(tab)` provoque l'affichage d'une image dans laquelle chaque valeur du tableau est représentée par une couleur différente, suivant un spectre défini par défaut.

Pour chaque valeur de z, il vous faudra itérer la suite $(u_n)_{n\in\mathbb{N}}$ un nombre suffisant de fois, en choisissant un critère vous permettant d'affirmer sans trop de risque de se tromper que la suite converge.

1. La *fractale de Newton* est obtenue lorsque $a = \mathrm{i}\,\frac{\sqrt{3}}{2}$, c'est-à-dire lorsque $P = X^3 - 1$. Représentez-la dans le domaine $\left\{z \in \mathbb{C} \;\middle|\; |\mathrm{Re}(z)| \leqslant 2 \text{ et } |\mathrm{Im}(z)| \leqslant 2\right\}$.
2. La valeur $a = -0,00508 + 0,33136\,\mathrm{i}$ permet d'observer un phénomène intéressant, la présence d'un *lapin de Douady*. Faire le tracé dans les domaines

$$\left\{z \in \mathbb{C} \;\middle|\; |\mathrm{Re}(z)| \leqslant 2 \text{ et } |\mathrm{Im}(z)| \leqslant 2\right\} \qquad \text{puis:} \qquad \left\{z \in \mathbb{C} \;\middle|\; |\mathrm{Re}(z)| \leqslant \frac{1}{10} \text{ et } |\mathrm{Im}(z)| \leqslant \frac{1}{10}\right\}.$$