UNIVERSITé DE RENNES I
U.F.R. DE MATHÉMATIQUES

UNIVERSITÉ DE RENNES 1

Licence 2 Mathématiques et Applications **2018-2019**
U.E. Outils Informatiques B

TD 3 - Résolution numérique des équations

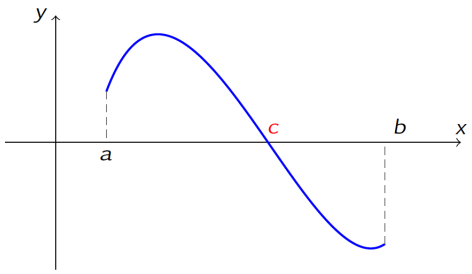
Étant donné un intervalle I de $\mathbb{R}$ et une application $f : I \to \mathbb{R}$, on cherche à trouver au moins une valeur approchée de $c \in I$ (s'il en existe) tel que

$$f(c) = 0.$$

Toutes les méthodes que nous allons présenter sont itératives et consistent en la construction d'une suite $(x_n)_{n\in\mathbb{N}}$ telle que $\lim_{n\to+\infty} x_n = c$.

1 Recherche d'une racine par dichotomie

Soit $f : [a, b] \to \mathbb{R}$ continue telle que $f(a)f(b) \leq 0$.



1. Prouver que f admet au moins une racine.
2. On considère le cas d'une racine unique. Montrer que suivant le signe de $f\left(\frac{a+b}{2}\right)$ l'une des deux relations est vérifiée

$$f(a)f\left(\frac{a+b}{2}\right) \leq 0 \text{ ou } f\left(\frac{a+b}{2}\right)f(b) \leq 0$$

3. On définit deux suites $(u_n)_{n\in\mathbb{N}}$ et $(v_n)_{n\in\mathbb{N}}$ par $u_0 = a$, $v_0 = b$ et

$$(u_{n+1}, v_{n+1}) = \begin{cases} (u_n, m) & \text{si } f(u_n)f(m) \leq 0 \\ (m, v_n) & \text{sinon} \end{cases} \quad \text{avec} \quad m = \frac{u_n + v_n}{2}$$

Montrer que ces deux suites sont convergentes et qu'elles admettent la même limite qui est la solution de l'équation $f(x) = 0$.

4. Proposer une fonction `dicho` prenant comme arguments `f, a, b, epsilon` qui itère les deux suites $(u_n)_{n\in\mathbb{N}}$ et $(v_n)_{n\in\mathbb{N}}$ jusqu'à convergence des suites.
   Exemples de résultats

```
>>> from numpy import sin
>>> dicho(sin, 3, 4)
3.141592653589214
>>> dicho(lambda x: x * x - 2, 1, 2)
1.4142135623724243
```

5. Analyser la terminaison de l'algorithme.
   Quand est-ce que l'algorithme s'arrête ?
6. Déterminer le coût de l'algorithme en fonction de la précision p choisie $\epsilon = 10^{-p}$.
   On définira le coût comme le nombre d'itérations de la boucle conditionnelle.
7. Proposer une adaptation de la fonction `dicho` qui permette d'ajouter un nombre maximal d'itérations avant d'aboutir à un échec.

Remarque, la fonction `bisect` du module `scipy.optimize` réalise une recherche dichotomique.

2 Recherche d'une racine par la méthode de la sécante

L'idée de la méthode de la sécante est très simple : pour une fonction f continue sur un intervalle $[a, b]$, et vérifiant $f(a) < 0$, $f(b) > 0$, on trace le segment $[AB]$ où $A = (a, f(a))$ et $B = (b, f(b))$. Si le segment reste au-dessus du graphe de f alors la fonction s'annule sur l'intervalle $[a', b]$ où $(a', 0)$ est le point d'intersection de la droite (AB) avec l'axe des abscisses. La droite (AB) s'appelle la sécante. On recommence en partant maintenant de l'intervalle $[a', b]$ pour obtenir une valeur a''.

Proposition
Soit $f : [a, b] \to$ une fonction continue, strictement croissante et convexe telle que $f(a) < 0$, $f(b) > 0$. Alors la suite définie par

$$a_0 = a \quad \text{et } a_{n+1} = a_n - \frac{b - a_n}{f(b) - f(a_n)} f(a_n)$$

est croissante et converge vers la solution ℓ de $(f(c) = 0)$.

1. Faire un dessin représentant la méthode de la sécante.
2. Justifier la construction de la suite récurrente.
3. Montrer que (a_n) est croissante. On pourra commencer par montrer par récurrence que $f(a_n) \leq 0$.
4. Prouver la convergence de (a_n) et déterminer sa limite.
5. Écrire une fonction `secante` pour implémenter la méthode de la sécante.

3 Recherche d'une racine par la méthode de Newton

Soit f une fonction continue et dérivable sur $[a, b]$. On construit une suite $(x_n)_{n\in\mathbb{N}}$ en remplaçant l'equation $f(x) = 0$ par l'équation affine $\phi_{x_n}(x_{n+1}) = 0$, où ϕ_{x_n} est l'équation de la tangente à f en x_n .

1. Ecrire l'équation de la tangente à f en x_n.
2. En déuire la suite puisl'algortihme de Newton.
3. Écrire une fonction `newton` pour implémenter la méthode de Newton.

4 Approximation numérique de la dérivée

Un des principaux inconvénient de la méthode de Newton est qu'elle requiert le calcul de la dérivée $f'(x)$.

1. On approche la dérivée par la différence finie

$$\frac{f(x+h) - f(x)}{h}$$

   et on cherche la valeur de h qui minimise l'erreur d'approximation.

   (a) On fait l'hypothèse que l'erreur numérique de calcul de $f(x)$ est du même ordre de grandeur que la précision relative de la machine $\epsilon = 2^{-52}$ ie l'erreur absolue sur $f(x)$ vaut $\epsilon|f(x)|$. Montrer que l'erreur numérique peut être approchée par

$$E_{num}(h) \sim 2\epsilon \left|\frac{f(x)}{h}\right|$$

   (b) En utilisant un développement de Taylor à l'ordre 2, montrer que l'erreur mathématique peut être approchée par

$$E_{math}(h) \sim \frac{|hf''(x)|}{2}$$

   où f'' est la dérivée seconde de f.

   (c) Faire l'étaude de la fonction $E(h) = E_{num}(h) + E_{math}(h)$ (tableau de variation) et en déduire une valeur de h optimale.

2. On peut aussi approcher la dérivée par

$$\frac{f(x+h) - f(x-h)}{2h}$$

   (a) En procédant comme à la question précédente, approcher l'erreur totale.

   (b) Puis trouver une valeur optimale pour h.

3. Une autre possibilité est de remplacer la dérivée par la pente de la corde reliant les points d'abscisses x_{n-1} et x_{n-2} :

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \text{ est remplacé par } x_{n+1} = x_n - \frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})}$$

   Reconnaissez-vous l'algorithme correspondant ?