TD9 - Méthodes de recherche

1 Solutions approchées d'équations numériques

On considère les équations de la forme

$$f(x) = 0,$$

où l'inconnue est $x \in \mathbb{R}$ et où $f : I \to \mathbb{R}$ est une fonction et I un intervalle préalablement déterminés.
Si f est inversible et si on connaît sa fonction inverse, alors il existe une unique solution : $f^{-1}(0)$.
Pour certaines fonctions polynômiales, on connaît des méthodes pour trouver les racines.
Supposons donc qu'on a une fonction f dont on ne peut pas déterminer l'inverse par le calcul (analytiquement). Il faut trouver une approximation numérique des solutions de l'équation. On rentre dans le domaine de l'analyse numérique.
On introduit ici deux méthodes, la dichotomie et la méthode de Newton.

1.1 Dichotomie

Si on dispose d'une fonction continue f sur un intervalle I et de $u < v \in I$ tels que $f(u)f(v) < 0$, alors d'après le TVI, il existe un zéro de f dans $[u, v]$.
On veut pouvoir approcher un zéro de f avec une précision choisie.
La méthode de dichotomie consiste à considérer le milieu de $[u, v]$ et à évaluer f en ce milieu. Suivant le signe de $f(\frac{u+v}{2})$, on aura l'existence d'un zéro de f dans $\left[u, \frac{u+v}{2}\right]$ ou $\left[\frac{u+v}{2}, v\right]$. On réitère la démarche, la taille de l'intervalle étant divisée par 2 à chaque étape.
On emploie pour cette méthode un *critère d'arrêt*, qui stoppe les itérations lorsque les bornes de l'intervalle réduit sont suffisamment proches.

1 – Ecrire une fonction `dichot` telle que si on a une fonction `f`, deux flottants `u` et `v` et un flottant `eps`> 0, alors `dichot(f,u,v,eps)` renvoie une approximation d'un zéro de `f` entre `u` et `v` avec une précision de `eps`, par la méthode de dichotomie.

2 – En utilisant `dichot`, déterminer une approximation numérique à 10^{-10} près de $\sqrt{2}$.

1.2 Méthode de Newton

L'idée de l'algorithme de Newton est d'utiliser une approximation affine d'une fonction dérivable f.

Voici une description schématique de l'algorithme :
- On se fixe une précision ϵ.
- On évalue f pour en un point x_n.
- Si $f(x_n) \neq 0$ et $f'(x_n) \neq 0$, on définit $x_{n+1} = x_n - \dfrac{f(x_n)}{f'(x_n)}$.
- On réitère jusqu'à ce qu'on obtienne une approximation d'une racine à la précision ϵ.

3 – Déterminer à quoi correspond graphiquement cette méthode.

4 – Ecrire une fonction `newton` telle que `newton(f,fprim,x0,n)` renvoie l'approximation d'une racine de `f` (de dérivée `fprim`) par la méthode de Newton, en prenant comme approximation initiale `x0` et en itérant la méthode `n` fois.

5 – Ici, on suppose que si deux approximations correspondant à deux itérées successives par la méthode de Newton sont proches, alors elles sont proches de la racine recherchée.

Ecrire une fonction `newtonbis` telle que `newtonbis(f,fprim,x0,eps)` renvoie l'approximation d'une racine de `f` (de dérivée `fprim`) par la méthode de Newton, en prenant comme approximation initiale `x0` et en arrêtant les itérations lorsque deux approximations successives sont proches à `eps` près.

6 – En utilisant `newtonbis`, déterminer une approximation numérique à 10^{-10} près de $\sqrt{2}$.

1.3 Comparaison des méthodes

5 – Pour la fonction `f = lambda x : x*x-2`, utiliser les méthodes de dichotomie et de Newton pour déterminer des valeurs approchées à 10^{-10} près de $\sqrt{2}$.

6 – Créer des fonctions `dichot2` et `newton2` qui comptent le nombre de boucles nécessaires pour atteindre le résultat précedent pour chaque méthode.

7 – Tracer un graphe comparant pour chaque méthode le nombre de boucles nécessaires pour obtenir une précision de 10^{-n} en fonction de n.

2 Calcul de la racine carrée

On veut écrire une fonction `racinecarree` qui à un `float` renvoie sa racine carrée, en utilisant uniquement les opérateurs algébriques `+`, `-`, `*` ,`/` et l'opérateur `<`.

On veut minimiser le temps de calcul. Pour cela, on va chercher des *méthodes mixtes*, c'est-à-dire combinant plusieurs méthodes standard. Pour l'étude qui suit, on ne pourra pas forcément tout expliciter théoriquement, et on va donc utiliser des techniques pour évaluer "expérimentalement" le temps de calcul ou le nombre d'étapes. Il s'agit d'un questionnement ouvert, et on utilisera les quelques pistes suivantes :

1 – On pourra fixer le nombre d'itérations en argument de la fonction.

2 – On pourra également se servir de la fonction `print` pour voir l'évolution de l'approximation au cours des itérations.

3 – Tester la fonction avec des nombres de différentes tailles.

4 – Utiliser la fonction `time` du module `time`.

5 – Utiliser une boucle `for` pour répéter 10^4, 10^5 ou 10^6 fois un calcul.

6 – Envisager une méthode mixte ou l'on commence avec une autre méthode que Newton avant d'aboutir par Newton. Proposer plusieurs méthodes mixtes.

7 – Enfin, on pourra comparer les vitesses de calcul des différentes fonctions créées, entre elles ainsi qu'avec la fonction intégrée `sqrt`.

Faire un bilan formel des résultats et comparaisons obtenus.

3 Recherche dans un tableau trié

Appliquer la méthode de dichotomie à la recherche d'un élément dans une liste triée (dans l'ordre croissant). On prendra ici garde à la manipulation des indices, qui sont entiers, à la fois pour les itérations et pour bien écrire le critère d'arrêt.

8 – Ecrire une fonction `recherchelistedichot` qui pour une liste triée l et un flottant a détermine si a apparaît dans l.

9 – Ecrire une fonction `indicelistedichot` qui à une liste `l` (supposée triée dans l'ordre croissant) et à une valeur `a` renvoie le plus grand entier `i` tel que `l[i]`$\leqslant$`a`, et zéro si `l[0]`$>$`a`.

{