TP 15 Python – Dichotomie

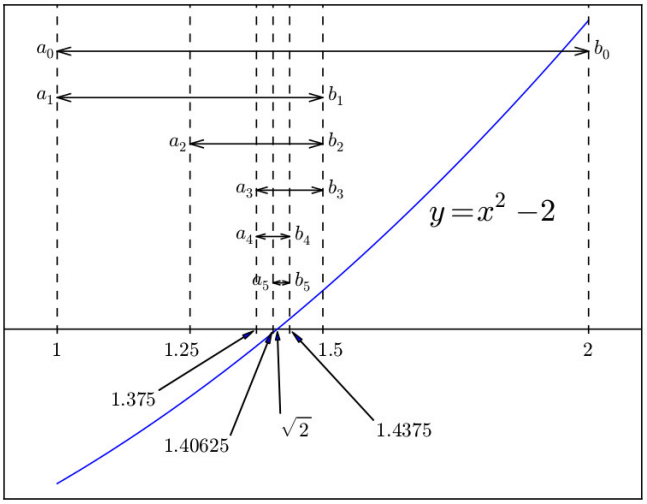
L'algorithme de dichotomie consiste à rechercher une solution en **divisant par deux à chaque étape la taille de la "zone de recherche"**.

1. Deux exemples

On cherche à résoudre l'équation $x^2 = 2$ sur $\mathbb{R}^+$, c'est-à-dire à trouver l'unique point d'annulation $\alpha \in \mathbb{R}^+$ de la fonction $f(x) = x^2 - 2$.
On sait que $\alpha \in [1, 2]$ puisque $f(1) = 1 - 2 = -1 < 0$ et $f(2) = 4 - 2 = 2 > 0$.
On rappelle le principe de l'algorithme de dichotomie pour obtenir une valeur approchée de α :

On part du segment $[a, b]$ sur lequel on cherche un zéro. Ici $a = \ldots$ et $b = \ldots$ conviennent puisque leurs images sont de signes opposés. À chaque étape :

- On considère le milieu $m = \dfrac{a+b}{2}$ du segment.
- Si $f(a)f(m) \leq 0$, alors on sait que $\alpha \in [\ldots, \ldots]$.
  On réduit donc notre intervalle de recherche en posant : $\ldots = \ldots$
- Sinon, alors on a nécessairement $f(b)f(m) \leq 0$, et donc on sait que $\alpha \in [\ldots, \ldots]$.
  On réduit donc notre intervalle de recherche en posant : $\ldots = \ldots$



On peut alors écrire le script suivant (à compléter) pour obtenir une valeur approchée de α à $10 * *(-6)$ près :

```
a, b = ..., ...
while b - a ............... :
    fa = a**2 - 2
    m = ................
    fm = ........................
    if fa * fm <= 0:
        b = m
    else:
        ........
print(.....) # pas de "return" parce qu'on n'a pas de fonction ici, juste un script
```

Deuxième exemple : Montrer que l'équation $-3x^3 - 2x + 1 = 0$ admet une unique solution réelle, et que cette solution se trouve dans l'intervalle $[0, 1]$:

Adapter le script précédent pour qu'il renvoie une valeur approchée de cette solution à 10^{-4} près :

2. L'algorithme (à connaître par cœur !)

Sur le même modère, écrire une **fonction** Python `dichotomie(f,a,b,eps)` qui prend en argument une fonction f déjà définie en Python, deux bornes a, b et un *seuil* ϵ, et qui **renvoie** une valeur approchée à ϵ près d'une solution de l'équation $f(x) = 0$ sur l'intervalle $[a, b]$.
On supposera que le théorème des valeurs intermédiaires s'applique sur cet intervalle.

Tester votre fonction avec $\epsilon = 10^{-7}$ et :

1. L'exemple précédent. On définira au préalable une fonction Python f qui renvoie la valeur de $f(x) = -3x^3 - 2x + 1$.
2. La fonction $(x - 3)(x + 5)$ sur l'intervalle $[-4, 4]$.
3. L'équation $\ln(x) = 1$. (que doit-on obtenir ? quelle fonction f définir ? quelles bornes choisir au départ ?)

3. Localiser les racines

On souhaite résoudre l'équation $x \sin(x) = e^x$.

1. Définir en Python une fonction f appropriée sur laquelle appliquer l'algorithme de dichotomie.
2. Il nous faut des bornes a et b correctes qui vérifient le TVI. Pour ce faire, **tracer la fonction** f à l'aide des modules `matplotlib.pyplot` et `numpy`.
   On prendra les abscisses x entre -10 et 10, et on restreindra les ordonnées à l'intervalle $[-10, 10]$ également.
3. À l'aide du tracé réalisé : combien de solutions doit-on chercher ? Proposer un intervalle $[a, b]$ de départ pour chacune d'entre elles.

4. À l'aide de la fonction `dichotomie`, donner une valeur approchée à 10^{-4} près de chacune des solutions cherchées.

4. Vitesse de convergence

À chaque étape, l'erreur d'approximation est divisée par 2.
Notre tout premier exemple partait avec un intervalle $[a, b] = [1, 2]$. Combien d'étapes faut-il pour obtenir une solution approchée à 10^{-6} ? .....
.....
À retenir : on a $2^{10} = 1024 \approx 1000 = 10^3$. Ainsi :

Toutes les ... itérations, l'approximation obtenue gagne ... décimales exactes supplémentaires.

De manière plus grossière : $2^3 = 8 \approx 10$, donc on gagne une décimale exacte toutes les 3 itérations environ.

5. Application

Un golfeur est situé à $342\,m$ du trou. Il frappe la balle sur un tee d'une hauteur de $0.02\,m$ à une vitesse de $90\,m/s$. On cherche à évaluer la longueur L de son tir et à le conseiller quant à l'angle initial optimal.

Si on néglige les frottements de l'air, on détermine l'équation de la trajectoire grâce aux lois de Newton et on obtient :

$$y(x) = -\frac{g}{2}\left(\frac{x}{\cos(\alpha_0)v_0}\right)^2 + \tan(\alpha_0)x + h_0$$

- $y(x)$ désigne l'altitude de la balle en fonction de sa position horizontale x.
- h_0, v_0 et α_0 sont respectivement l'altitude, la vitesse et l'angle donnés initialement au projectile.
- g est l'accélération gravitationnelle.

1. Définir en Python la fonction `altitude_balle(x, alpha0)`.
2. Analyse graphique :
   (a) Tracer sur un même graphe les trajectoires possibles pour chaque angle de 0 à 90 degrés, avec un pas de 10.
   (b) Affiner le pas et les bornes de l'angle pour conseiller graphiquement le golfeur sur l'angle à choisir.
   (c) Combien d'angles ont l'air de correspondre à la contrainte du golfeur ? Dans quels intervalles se situent-ils ?
3. À l'aide de l'algorithme de dichotomie, écrire une fonction `longueur_tir(alpha0)` qui renvoie la distance entre l'origine et le point d'impact de la balle avec le sol.
4. À l'aide de la fonction `dichotomie`, déterminer une valeur approchée au centième de degré près pour chacun des angles possibles. Lequel vous semble le plus pertinent ?