ANALYSE NUMÉRIQUE *1ère année de l'ISIMA, semestre 2*

TP Fonction unimodale

Jean-Philippe Gayon et Viet Hung Nguyen 2020–2021

Préambule. Ce TP s'intéresse a plusieurs méthodes de réduction d'intervalles pour trouver le minimum de fonctions unimodales dans un intervalle $[a, b]$. La notion de fonction unimodale a été introduite dans les notes de cours sur l'optimisation non linéaire.

Méthode de réduction d'intervalles Le principe des méthodes de réduction d'intervalle pour la recherche du minimum x^* d'une fonction unimodale f est de réduire à chaque itération k l'intervalle $[a_k, b_k]$ dans lequel x^* est présent et f reste unimodale (en considérant que $a_1 = a$ et $b_1 = b$). Quand cet intervalle devient suffisamment petit (par exemple sa taille $(b_k - a_k)$ est plus petite que 10^{-7}), on considère que la convergence est atteinte et la solution x^* peut être prise à l'une des deux extrémités de l'intervalle.

1 Méthode de recherche dichotomique

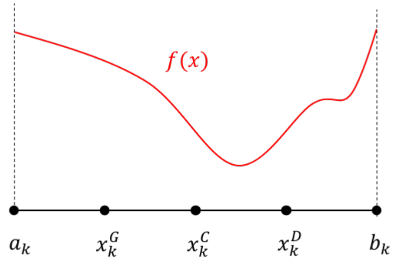
La méthode de recherche dichotomique comme son nom l'indique réduit l'intervalle $[a, b]$ de moitié après chaque itération. Soit $[a_k, b_k]$ l'intervalle réduit après les $k - 1$ premières itérations. A chaque itération k, on calcule trois points équirépartis dans l'intervalle $[a_k, b_k]$:

$$x_k^G = a_k + \frac{1}{4}(b_k - a_k)$$

$$x_k^C = a_k + \frac{1}{2}(b_k - a_k)$$

$$x_k^D = a_k + \frac{3}{4}(b_k - a_k)$$

On évalue par ailleurs $f(x_k^G)$, $f(x_k^C)$ et $f(x_k^D)$.



On réduit l'intervalle $[a_k, b_k]$ selon les 3 cas suivants :

1. si $f(x_k^C) > f(x_k^D)$ alors $a_{k+1} = x_k^C$ et $b_{k+1} = b_k$. Notons que $x_{k+1}^C = x_k^D$.
2. si $f(x_k^C) > f(x_k^G)$ alors $a_{k+1} = a_k$ et $b_{k+1} = x_k^C$. Notons que $x_{k+1}^C = x_k^G$.
3. sinon, c'est-à-dire si $f(x_k^C) \leq f(x_k^D)$ et $f(x_k^C) \leq f(x_k^G)$, alors $a_{k+1} = x_k^G$ et $b_{k+1} = x_k^D$. Notons que $x_{k+1}^C = x_k^C$.

Dans tous les cas, x_{k+1}^C peut être pris parmi les points x_k^G, x_k^C et x_k^D. On a donc besoin de calculer et évaluer uniquement x_{k+1}^G et x_{k+1}^D. La méthode de la recherche dichotomique nécessite donc 2 évaluations de la fonction f à chaque itération, à l'exception de la première.

Q1. Pour chacun des cas possibles ci-dessus, démontrer que x^* est présent et f reste unimodale dans le nouvel intervalle.

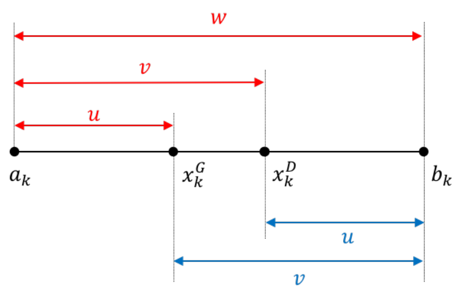
Q2. Implémenter la méthode de recherche dichotomique en Python et vérifier qu'elle donne le résultat attendu pour les quatres fonctions suivantes sur l'intervalle $[-1000, 1000]$.

$$f_1(x) = |x - 100|$$
$$f_2(x) = \begin{cases} \sqrt{x-50} & \text{si } x \geq 50 \\ \sqrt{-(x-50)} & \text{si } x \leq 50 \end{cases}$$
$$f_3(x) = \min(4x, x + 5)$$
$$f_4(x) = -x^3$$

2 Méthode du nombre d'or

Au niveau de la complexité des méthodes de réduction d'intervalle, le nombre total d'évaluations de la fonction f est un facteur qui peut être plus important encore que le nombre d'itérations. Il serait donc intéressant d'avoir une méthode qui ne demanderait qu'une évaluation de f à chaque itération au lieu de deux, quitte à moins réduire l'intervalle. La méthode du nombre d'or répond à cet objectif comme nous allons le voir dans cette partie.

On peut adapter la méthode de recherche dichotomique en ne calculant que 2 points x_k^G et x_k^D dans l'intervalle $[a_k, b_k]$, tels que $a_k < x_k^G < x_k^D < b_k$. Nous ferons l'hypothèse dans toute cette partie que la distance u entre a_k et x_k^G est la même que celle entre x_k^D et b_k, comme illustré sur la figure ci-dessous.



On réduit alors l'intervalle $[a_k, b_k]$ selon les cas de figure suivants :

1. si $f(x_k^G) > f(x_k^D)$ alors $a_{k+1} = x_k^G, b_{k+1} = b_k$
2. si $f(x_k^G) < f(x_k^D)$ alors $a_{k+1} = a_k$ et $b_{k+1} = x_k^D$
3. sinon, c.à.d. si $f(x_k^G) = f(x_k^D)$, alors $a_{k+1} = x_k^G$ et $b_{k+1} = x_k^D$.

Dans les deux premiers cas de figure, l'intervalle est divisé par un facteur

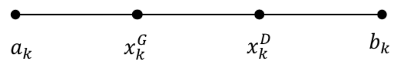
$$\alpha = \frac{w}{v}$$

Notons que le 3ème cas de figure est très peu probable.

Q3. Pour commencer, démontrer que x^* est présent et f reste unimodale dans le nouvel intervalle.

Q4. Exprimer x_k^G et x_k^D en fonction de a_k, b_k et du facteur α.

Q5. Supposons dans un premier temps que les points sont équirépartis comme sur la figure ci-dessous. Par combien est divisé l'intervalle à chaque itération pour chaque cas de figure ?



Q6. Combien d'évaluations de la fonction f doit-on effectuer à chaque itération ?

Q7. Comment positionner x_k^G et x_k^D afin de réduire au maximum l'intervalle à chaque itération pour les deux premiers cas de figure ?

La méthode nombre d'or consiste à sélectionner les points x_k^G et x_k^D de telle sorte que l'un de ces points soit repris à l'itération suivante. Il n'y aura alors qu'une seule évaluation de f à chaque itération.

Plus précisément, nous aimerions avoir :

1. si $f(x_k^G) > f(x_k^D)$, alors $x_{k+1}^G = x_k^D$
2. si $f(x_k^G) < f(x_k^D)$ alors $x_{k+1}^D = x_k^G$
3. si $f(x_k^G) = f(x_k^D)$, on accepte d'évaluer deux fois la fonction dans la mesure où cela n'arrive que très rarement

Pour avoir $x_{k+1}^G = x_k^D$ dans le 1er cas de figure et $x_{k+1}^D = x_k^G$ dans le second cas de figure, il suffit d'avoir

$$\alpha = \frac{w}{v} = \frac{v}{u}$$

Q8. Déterminer la valeur numérique de α satisfaisant l'équation précédente.

Q9. De combien est réduit l'intervalle à chaque itération pour la méthode du nombre d'or suivant chaque cas de figure ?

Q10. Combien d'évaluations de la fonction f doit-on effectuer à chaque itération pour la méthode du nombre d'or ?

Q11. Implémenter la méthode du nombre d'or en Python et vérifier qu'elle donne le résultat attendu pour les fonctions f_1, f_2, f_3, f_4.

3 Application : Problème d'abonnement avec pénalité

On considère le problème d'abonnement d'un produit de consommation régulière (eau, electricité, gaz, internet, ...). On considère en particulier un paramètre qui est la puissance maximum de consommation à tout moment. Par exemple, en électricité, en général, un particulier s'inscrit pour une puissance maximale de compteur de 6kVA. Si à un instant donné, la consommation dépasse cette puissance, le compteur va sauter ce qui entraînera une coupure de courant. Pour éviter cette coupure (qui peut être dommageable, notamment pour les entreprises), on préconise un autre type d'abonnement, dans lequel on laisse la consommation se poursuivre mais le consommateur doit payer une pénalité pour dépassement de capacité. Pour ce type de contrat, au lieu d'être imposé d'une puissance maximale souscrite, le consommateur a le droit de la choisir. Le problème qui se pose alors pour un consommateur est de choisir une puissance souscrite optimale selon ses habitudes de consommation.

Dans ce TP, nous considérons une version simplifiée de ce problème. Étant donné un relevé de consommation sous la forme d'une liste $L = \{c_1, c_2, \ldots, c_n\}$ où c_i représente la consommation à période i. Le coût que le consommateur devrait payer pour ses consommations est défini par la fonction f suivante

$$f(p) = a * p + b * \sqrt{\sum_{i=1}^{n} (\max(0, c_i - p))^2}$$

où p est la puissance souscrite. La première partie de la fonction $f(p)$ est $a * p$ où a est une constante positive qui représente le coût fixe de l'abonnement. La second partie $b * \sqrt{\sum_{i=1}^{n} (\max(0, c_i - p))^2}$ représente le coût de la pénalité de dépassement de capacité. Ce coût est calculé par la norme 2 du vecteur de dépassement multipliée par une constante positive b. Soit $C = \max\{c \mid c \in L\}$ la plus grande consommation, le problème de puissance souscrite optimale peut être formulé mathématiquement comme

$$\min_{0 \le p \le C} f(p)$$

La partie coût fixe de la fonction $f(p)$ est linéaire strictement croissante et la partie pénalité est (convexe) strictement décroissante. On peut démontrer que $f(p)$, qui est la somme de ces deux parties, est unimodale dans l'intervalle $[0, C]$. Une démonstration rigoureuse est hors sujet pour ce TP mais on peut intuitivement imaginer que l'unimodalité de la fonction f comporte la première phase décroissante où pour des petites valeurs de p, la partie pénalité domine la partie coût fixe. La fonction devient croissante à partir du moment où cette dominance s'inverse avec des grandes valeurs de p. La valeur p^* qui minimise $f(p)$ est justement le point où les deux parties sont en équilibre.

Q12. Écrire une fonction Python qui reçoit comme paramètres p, L, a et b qui calcule la valeur de $f(p)$ en utilisant des paramètres L, a et b.

Q13. Écrire une fonction Python permettant de tracer f dans l'intervalle $[0, C]$. Par exemple, la figure 1 représente la fonction f tracée avec $a = 1$, $b = 0.5$ et

$$L = [2500, 3500, 4380, 4389, 4725, 4800, 3700, 3500, 7000, 7500, 2000, 1200].$$

Q14. Écrire une fonction Python qui trouver le minimum p^* de $f(p)$ dans l'intervalle $[0, C]$ avec la méthode de recherche dichotomique.

Q15. Écrire une fonction Python qui trouver le minimum p^* de $f(p)$ dans l'intervalle $[0, C]$ avec la méthode du nombre d'or.

Q16. Comparer l'efficacité des deux méthodes en terme de temps d'exécution et en terme de nombre d'évaluations total de f en les expérimentant sur 100 instances générées de façon aléatoires avec même longueur pour la liste L et les mêmes constantes a et b.

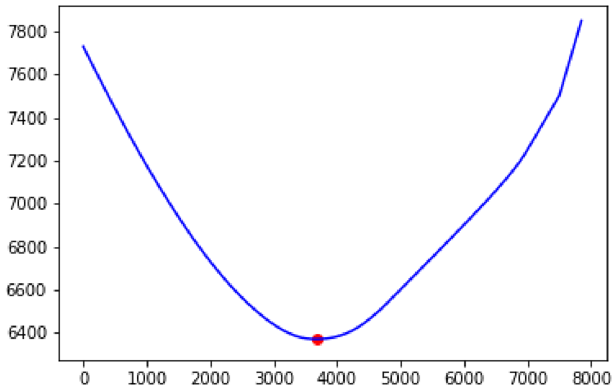


FIGURE 1 – Fonction de coût du problème d'abonnement avec pénalité