TP n°3 :

Minimisation d'une fonction de 2 variables :
méthode du gradient et méthode de Newton

Ce TP, à réaliser en binôme, est noté. Travail à rendre 1 semaine après la séance encadrée. L'évaluation de ce travail constituera la note de TP (20% de la note finale de l'UE).

Objectifs du TP :

- Mettre en œuvre la recherche de minimum par la méthode du gradient à « pas fixe »
- Mettre en œuvre la recherche de minimum par la méthode de Newton
- Analyser le comportement et les conditions de convergence de ces méthodes

Ce TP illustre les slides 18, 28-32 du cours n°2. Voir aussi la section « Minimisation sans contrainte » du poly de 2005, pages 10 et 11.

Principe de la méthode du gradient

Le problème posé est de minimiser une certaine fonction $J(X)$ définie de $\mathbb{R}^n$ dans $\mathbb{R}$. La variable X est une vecteur : $X = (x_1, \dots, x_n) \in \mathbb{R}^n$. La fonction J est supposée différentiable à l'ordre 1[1], et on suppose que l'on connait l'expression analytique du gradient.

De manière générale, les méthodes de descente sont des méthodes itératives qui génèrent une suite de points X_n tels que : $\forall n \geq 0 : J(X_{n+1}) < J(X_n)$. Au point X_n, la direction dans laquelle la fonction décroît le plus vite est la direction opposée au gradient de J. Une technique simple pour réduire J consiste donc à rechercher un nouveau point en se déplaçant d'une certaine quantité dans cette direction.

Le schéma général de la méthode du gradient est le suivant :

$$X_{n+1} = X_n - \alpha.\nabla J(X_n) \qquad \text{avec} \qquad \alpha > 0 \text{ choisi pour avoir : } J(X_{n+1}) < J(X_n)$$

Gradient à pas optimal :

La méthode dite « à pas optimal » consiste à déterminer à chaque itération la valeur de α qui minimise J dans la direction du gradient. Il s'agit donc de minimiser $G(\alpha) = J[X_n - \alpha.\nabla J(X_n)]$ par rapport à la variable α, ce qui ramène à un problème de minimisation à une variable. On peut également formuler ce problème comme un problème d'optimisation sous contrainte égalité : il faut minimiser $J(X)$, sous contrainte $X = X_n - \alpha.\nabla J(X_n)$. Cette méthode est conceptuellement satisfaisante, mais en pratique elle n'est pas vraiment efficace. En effet, on montrera dans le cours n°3 que les directions de recherche utilisées au cours de l'algorithme à pas optimal sont deux à deux orthogonales. Elles sont entièrement déterminées par le point initial et ne sont pas nécessairement adaptées au relief de la fonction dans le voisinage du minimum. La deuxième raison est que la minimisation de J dans une direction à chaque itération a un coût de calcul qui ne peut être négligé. Pour cette raison, on préfère d'autres techniques comme le gradient à pas fixe, le gradient à pas adaptatif, ou les gradients conjugués. Dans ce TP, vous programmerez la méthode du gradient à pas fixe, puis vous proposerez une règle empirique pour faire du pas adaptatif.

[1] C'est-à-dire que le vecteur gradient est défini en tous points

Gradient à pas fixe :

La méthode dite « à pas fixe » consiste à prendre une valeur de α constante au cours des itérations. Le pas de déplacement étant proportionnel au module du gradient, il diminue automatiquement au voisinage du minimum et on peut espérer converger vers la solution, à condition de choisir une valeur de α correcte. En effet, si α est « trop grand », on peut osciller autour du minimum sans jamais l'atteindre. A l'inverse, si α est « trop petit », on est sûr de converger vers le minimum, mais avec un nombre d'itérations inutilement grand. En pratique, à moins que l'on dispose d'informations sur la dérivée seconde de la fonction, la détermination de α est empirique (essais-erreurs). Pour un problème donné, on teste différentes valeurs de α jusqu'à trouver quelque chose de satisfaisant.

Travail à réaliser :

A. Implanter la méthode du gradient à pas fixe

Schéma général de l'algorithme :

- Notations :
  - X_n et X_{n+1} : point initial et point final de l'itération n.
  - dX : pas de déplacement pour une itération
  - n et n_{max} : compteur d'itérations et nombre maximal d'itérations autorisé
  - X_0 : point de départ de l'algorithme
  - α : pas de recherche
  - ε : critère de précision
  - $converge$: indicateur booléen de convergence
- Algorithme :
  - Choix des paramètres de l'algorithme : $X_0, \alpha, \varepsilon, n_{max}$
  - Initialisation : $X_n \leftarrow X_0$, $dX \leftarrow 1, n \leftarrow 0$
  - Tant que $dX > \varepsilon$ et $n < n_{max}$:
    - $X_{n+1} \leftarrow X_n - \alpha.\nabla J(X_n)$
    - $dX \leftarrow \|X_{n+1} - X_n\|$
    - $X_n \leftarrow X_{n+1}$
    - $n \leftarrow n + 1$
  - $converge \leftarrow (dX \leq \varepsilon)$

Cahier des charges :

- Les expressions analytiques de J et son gradient ∇J sont connues
- La recherche de minimum est faite dans une fonction dont les paramètres d'entrée sont le nom de la fonction à minimiser, le nom de la fonction dérivée, le point de départ de l'algorithme, la valeur de α, la précision souhaitée ε et le nombre maximal d'itérations autorisées n_{max}.
- Affichage des résultats : Les isovaleurs de J sont tracées, de même que la suite de segments $[X_{n-1}, X_n]$. Le programme indique à l'utilisateur si la recherche de minimum a convergé, le nombre d'itérations réalisées et le dernier point obtenu (solution approchée).

Fonction à étudier :

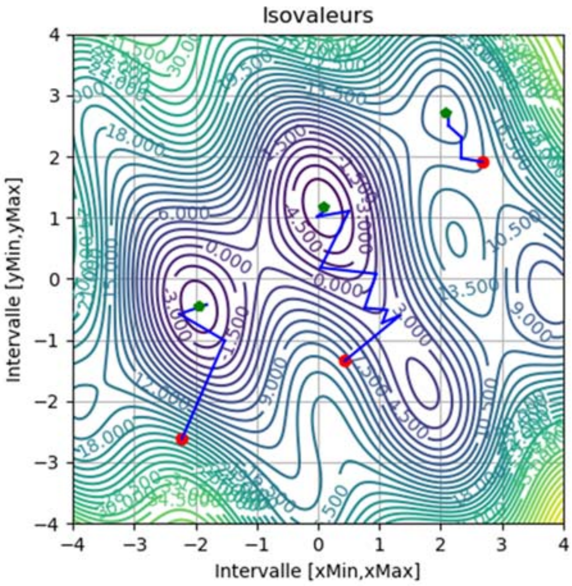
- La fonction à minimiser a pour expression : $J(X) = {x_1}^2 + 1.5{x_2}^2 - 5\sin(2x_1 + x_2) + 5\sin(x_1 - x_2)$.
- Dans un premier temps, vérifier que le mécanisme itératif est correctement implanté. Tester l'algorithme pour <u>quelques</u> itérations, en commençant avec une valeur de α petite, et pour différents points de départ.
- Une fois le mécanisme itératif validé, augmenter le nombre d'itérations pour obtenir la convergence. Tester l'algorithme pour différente valeurs de α et différents points de départ.
- Mettre en évidence le fait que l'algorithme diverge si α dépasse une certaine valeur.

- Proposer et programmer un mécanisme qui ajuste si besoin la valeur de α donnée par l'utilisateur afin d'assurer qu'on a toujours $J(X_{n+1}) < J(X_n)$. On est alors sûr d'assurer la convergence de l'algorithme.

La figure 1 montre un exemple de résultats. La figure de gauche montre les trajectoires obtenues pour 3 points initiaux différents (points rouges) et qui convergent vers 3 minimums différents (points verts). Le texte à droite correspond aux informations pour la trajectoire qui converge vers $(0.07.., 1.17..)$.

La figure 2 montre un autre exemple, obtenu pour une autre valeur de α. On y montre 100 trajectoires correspondant à des points initiaux tirées aléatoirement dans le domaine $\big[[-4,+4],[-4,+4]\big]$.

Libre à vous de faire les graphes qui vous paraissent porteurs d'information !




```
Iteration 1 / Alfa = 0.1 / dX = 1.1726397303812115
Iteration 2 / Alfa = 0.05 / dX = 0.31375725524627057
Iteration 3 / Alfa = 0.1 / dX = 0.22417638718859065
Iteration 4 / Alfa = 0.1 / dX = 0.36845091374441286
Iteration 5 / Alfa = 0.1 / dX = 0.5880632744245988
Iteration 6 / Alfa = 0.1 / dX = 0.9127923277522184
Iteration 7 / Alfa = 0.1 / dX = 1.0323413065998541
Iteration 8 / Alfa = 0.05 / dX = 0.52121220229181
Iteration 9 / Alfa = 0.05 / dX = 0.1827023849107101
Iteration 10 / Alfa = 0.05 / dX = 0.06296205616710276
Iteration 11 / Alfa = 0.05 / dX = 0.025730584192729816
Iteration 12 / Alfa = 0.05 / dX = 0.010613626093529989
Iteration 13 / Alfa = 0.1 / dX = 0.00906058396598789
Minimum trouvé après 13 itérations
Position du minimum : [0.07991947 1.17267019]
```

Figure 1 : Exemple d'affichage graphique et alpha-numérique (personnalisation et améliorations encouragées)

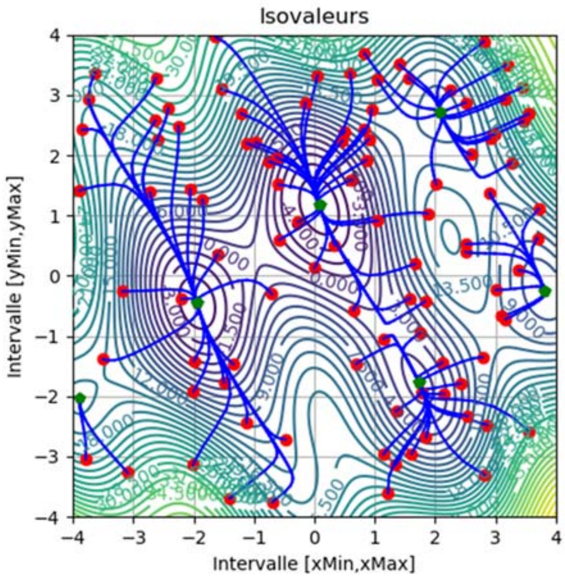


Figure 2 : Représentation des trajectoires pour 100 points initiaux différents

B. Implanter la méthode de Newton

Principe :

On souhaite trouver le minimum de la fonction J, supposée convexe sur $\mathbb{R}$. On suppose que l'on dispose des expressions analytiques du gradient ∇J et de la matrice hessienne HJ.

Soit X_n, une solution approchée. La fonction est approximée par $\tilde{J}_n$ son développement limité d'ordre 2 en X_n :

$$\widetilde{J_n}(X) = J(X_n) + \nabla J(X_n)^T.(X - X_n) + \frac{1}{2}(X - X_n)^T.HJ(X_n).(X - X_n)$$

La nouvelle solution approchée X_{n+1} est le minimum de $\widetilde{J_n}$, obtenu en annulant le gradient de $\widetilde{J_n}$.

$$\nabla\widetilde{J_n}(X) = \nabla J(X_n) + HJ(X_n).(X - X_n)$$

$$\nabla\widetilde{J_n}(X_{n+1}) = 0 \iff \nabla J(X_n) + HJ(X_n).(X_{n+1} - X_n) = 0$$

d'où : $X_{n+1} = X_n + \Delta X$, où ΔX est solution du système linéaire : $HJ(X_n).\Delta X = -\nabla J(X_n)$

Algorithme :

- Choix des paramètres de l'algorithme : X_0, ε, n_{max}
- Initialisation : $X_n \leftarrow X_0$, $dX \leftarrow 1, n \leftarrow 0$
- Tant que $dX > \varepsilon$ et $n < n_{max}$:
    - Calculer $\nabla J(X_n)$
    - Calculer $HJ(X_n)$
    - $\Delta X \leftarrow$ solution du système matriciel $HJ(X_n).\Delta X = -\nabla J(X_n)$
    - $X_n \leftarrow X_n + \Delta X$
    - $dX \leftarrow \|\Delta X\|$
    - $n \leftarrow n + 1$
- $converge \leftarrow (dX \leq \varepsilon)$

Travail à réaliser :

Mettre en œuvre l'algorithme et le tester pour la même fonction que précédemment. Etudier l'influence du point de départ. Comparer et analyser les résultats.

Suggestion : utiliser l'algorithme du gradient pour se rapprocher d'un minimum, puis l'algorithme de Newton pour déterminer précisément et rapidement le minimum.