University Mohammed VI
Polytechnic (UM6P)
College of computing

Année: 2023-2024

TP2: Newton avec région de confiance (Cas multidimensionnel)

Dans ce TP, vous allez implémenter l'algorithme de Newton avec région de confiance pour un problème d'optimisation sans contraintes de dimension n. Cet algorithme est dans l'essentiel exactement le même que celui que vous avez produit dans le TP1, la différence étant dans le calcul du minimum du modèle quadratique, maintenant de dimension n par le gradient conjugué linéaire.

1 Gradient conjugué linéaire

Le travail commence par l'implantation d'un algorithme du gradient conjugué linéaire, c'est à dire de l'appliquer à une fonction quadratique $q(\delta) = \frac{1}{2}\delta^t H\delta + c\delta$ où $H \succ 0$.

1.1 Gradient simple

Pour débuter, exécutez le `test1.sce`. Celui-ci reproduit les calculs du tableau de l'exemple de la fonction $f(x, y) = 2x^2 - 2xy + y^2 + 2x - 2y$. Si les calculs ne concordent pas, il faut trouver la source de la différence, une erreur d'installation. L'algorithme est codé dans `Grad_Mat.sci` est la descente simple du gradient.

1.2 Gradient conjugué linéaire avec matrice

Vous copiez `Grad_Mat.sci` dans un fichier `GC_Mat.sci` et faites les modifications nécessaires pour qu'il implémente l'algorithme du gradient conjugué. Pour un autre exemple de dimension $n = 15$, vous allez observer que le gradient conjugué nécessite au maximum n itérations pour trouver la solution contrairement à l'algorithme du gradient simple est encore loin de la solution après 20 itérations.

1.3 Gradient conjugué linéaire sans matrice

La prochaine étape consiste à remplacer partout l'utilisation de la matrice par un appel à une fonction `Hv(x,v)`. Le x est inutile pour l'instant puisque la matrice hessienne de la fonction q est constante. Testez en exécutant `test2.sce`, la variante étant codée dans `GC_Hv.sci`. On utilise un exemple encore avec $n = 15$ variables. Cette fois-ci, les valeurs propres de H sont toutes triples, donc on a au total 5 valeurs propres. Le gradient conjugué converge alors en 5 itérations.

2 Région de confiance

2.1 Adaptation de `GC_Hv`

Une fois que l'implantation de l'algorithme du gradient conjugué linéaire est fonctionnelle, il faut l'adapter pour l'utilisation au sein de l'algorithme de région de confiance. Maintenant, la matrice sous-jacente H n'est plus nécessairement définie positive. L'adaptation passe par deux modifications:

1. calcul d'un pas de déplacement maximum pour demeurer dans la région de confiance, le prochain point doit satisfaire $\|\delta + \theta p\| \leq \Delta$, ou encore $(\delta + \theta p)^t (\delta + \theta p) \leq \Delta^2$.

   θ_{Max} est la plus grande valeur de θ assurant cette condition.

2. détection d'une direction de courbure négative p : $p^t Hp < 0$. Cette détection s'obtient directement du signe de θ.

Si la valeur calculée de θ au sein de l'algorithme de gradient conjugué est trop grande ($> \theta_{Max}$) ou bien négative, il faut prendre θ_{Max} et terminer le calcul de gradient conjugué. Pour tester que l'adaptation semble fonctionner, exécutez `test3.sce` qui prend pour acquis que votre variante est codée dans `GC_TR.sci`. On résout trois instances de problèmes où la taille de région de confiance est ajustée pour valider que votre implantation semble fournir des résultats corrects.

2.2 Test de l'algorithme

Appliquer l'algorithme de région de confiance en utilisant votre `GC_TR.sci` sur le fameux exemple de Rosenbrock.