
TP 2 : Optimisation sans contrainte en dimension quelconque.

1 Résolution de l'équation de Laplace

L'objectif de ce TP est d'utiliser différentes méthodes d'optimisation afin d'approcher les solutions de l'équation suivante :

$$\begin{cases} -u''(x) = f(x), & x \in]0, 1[, \\ u(0) = u(1) = 0. \end{cases} \quad (1)$$

On se donne une subdivision de $[0, 1]$ en $N+1$ intervalles de longueur $h = \frac{1}{N+1}$, et on considère les points $x_i = ih$ de l'intervalle $[0, 1]$. On cherche alors une solution approchée sous la forme du vecteur $\mathbf{u}_N = (u_i)_{1 \leq i \leq N}$, où les u_i vérifient le schéma aux différences finies :

$$\begin{cases} \frac{-u_{i-1} + 2u_i - u_{i+1}}{h^2} = f(x_i), & \forall 1 \leq i \leq N, \\ u_0 = u_{N+1} = 0. \end{cases} \quad (2)$$

1. Montrer que l'équation (2) peut s'écrire sous la forme :

$$A_N \mathbf{u}_N = \mathbf{f}_N \quad (3)$$

où A_N et \mathbf{f}_N sont donnés par :

$$A_N = \begin{pmatrix} 2 & -1 & & 0 \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ 0 & & -1 & 2 \end{pmatrix} \quad \text{et} \quad \mathbf{f}_N = h^2 \begin{pmatrix} f(x_1) \\ \vdots \\ f(x_N) \end{pmatrix}.$$

2. Montrer que résoudre (3) revient à minimiser sur \mathbb{R}^N la fonctionnelle J_N définie par :

$$J_N(\mathbf{u}) = \frac{1}{2}(A_N \mathbf{u}, \mathbf{u}) - (\mathbf{f}_N, \mathbf{u}). \quad (4)$$

3. Calculer ∇J_N en fonction de A_N et \mathbf{f}_N .

Dans toute la suite du TP on prendra $f = 1$.

2 Méthode de Newton

Une première méthode pour calculer le minimum de J_N consiste à chercher un point \mathbf{u}_N annulant son gradient.

L'algorithme de Newton permet de trouver un point en lequel une fonction $F : \mathbb{R}^N \mapsto \mathbb{R}^N$ s'annule, connaissant une approximation \mathbf{u}^0 de ce point et un test d'arrêt ε :

Algorithme de Newton en dimension N
Initialiser le compteur k à 0, l'erreur **err** à 1.
Tant que l'erreur est plus grande que ε et que le compteur n'est pas trop grand :
— calculer le prochain candidat pour le zéro de F :
$$\mathbf{u}^{k+1} = \mathbf{u}^k - (\nabla F(\mathbf{u}^k))^{-1} F(\mathbf{u}^k),$$

— calculer l'erreur **err** = $|F(\mathbf{u}^{k+1})|$,
— incrémenter le compteur.

4. On cherche à appliquer l'algorithme de Newton à ∇J_N .
- a. Montrer que le calcul des \mathbf{u}^k est donné par la formule suivante :

$$\mathbf{u}^{k+1} = \mathbf{u}^k - A_N^{-1}(A_N \mathbf{u}^k - \mathbf{f}_N).$$

Faire une remarque !

- b. Écrire une fonction `newton.m` qui prend en argument A_N , \mathbf{f}_N , un point de départ \mathbf{u}^0 et un test d'arrêt ε , et qui renvoie le vecteur \mathbf{u}_N qui minimise J_N ainsi que le nombre d'itérations effectuées.
- c. Créer un script `scriptTP2_newton.m` et tester la fonction `newton.m` pour $\varepsilon = 10^{-12}$, et $N = 2, 5, 20, 50$. Afficher à l'aide de la fonction `fprintf` le nombre d'itérations ainsi que le temps de calcul pour chaque N . Tracer sur une même figure les solutions approchées \mathbf{u}_N , ainsi que la solution exacte de (1).
- d. Imaginer une équation, non linéaire, du même type pour laquelle l'algorithme de Newton n'est pas trivial, et l'utiliser pour la résoudre.

3 Méthodes de gradient

3.1 Méthode du gradient à pas fixe

On rappelle l'algorithme du gradient à pas fixe pour une fonctionnelle $J : \mathbb{R}^N \rightarrow \mathbb{R}$, un point de départ \mathbf{u}^0 , un pas ρ et un test d'arrêt ε préalablement définis :

Méthode du gradient à pas fixe
Initialiser le résidu r^0 à 1 et le compteur k à 0.
Tant que le résidu est plus grand que ε et que le compteur n'est pas trop grand :
— calculer la descente $\mathbf{w}^k = -\nabla J(\mathbf{u}^k)$,
— poser $\mathbf{u}^{k+1} = \mathbf{u}^k + \rho \mathbf{w}^k$,
— calculer le résidu $r^{k+1} = \|\mathbf{u}^{k+1} - \mathbf{u}^k\|$,
— incrémenter le compteur.

Créer un script `scriptTP2_fixe.m` pour répondre aux questions suivantes.

5. On se place dans le cas $N = 2$.
 - a. Calculer $J_2(u_1, u_2)$, et $\nabla J_2(u_1, u_2)$. Tracer sur une même figure les courbes de niveaux de J_2 ainsi que le champ de vecteurs ∇J_2 sur le pavé $[-10, 10] \times [-10, 10]$. On utilisera les fonctions `contour` et `quiver`.
 - b. Calculer les itérations $\mathbf{u}^k = (u_1^k, u_2^k)$ données par l'algorithme de gradient à pas fixe, et tracer sur la même figure que précédemment la ligne qui relie les \mathbf{u}^k . On prendra $\mathbf{u}^0 = (8, 4)$, $\rho = 0.1$ et $\varepsilon = 10^{-12}$.
6. On se place de nouveau dans le cas général.
 - a. Écrire une fonction `gradient_fixe.m` qui prend en argument A_N , \mathbf{f}_N , un point de départ \mathbf{u}^0 , un pas ρ et un test d'arrêt ε , et qui renvoie le vecteur \mathbf{u}_N qui minimise J_N ainsi que le nombre d'itérations effectuées.
 - b. Tester cette fonction pour $\rho = 0.1$, $\varepsilon = 10^{-12}$ et pour $N = 2, 5, 20, 50$. Afficher à l'aide de la fonction `fprintf` le nombre d'itérations ainsi que le temps de calcul pour chaque N . Tracer sur une même figure les solutions approchées \mathbf{u}_N , ainsi que la solution exacte de (1).
7. Reprendre la question 5.b. pour $\rho = 0.5$, puis $\rho = 1$. Que constate-t-on ? Peut-on choisir le pas ρ arbitrairement ?

Remarque :

Pour être plus générique, la fonction `gradient_fixe.m` peut prendre pour arguments la fonction J_N et son gradient DJ_N . On pourra alors répondre à la question 5. en faisant directement appel à la fonction `gradient_fixe.m` dont la question 5. servira alors de test de vérification.

3.2 Méthode du gradient à pas optimal

On rappelle l'algorithme du gradient à pas optimal pour une fonctionnelle $J : \mathbb{R}^N \rightarrow \mathbb{R}$, un point de départ \mathbf{u}^0 et un test d'arrêt ε préalablement définis :

Méthode du gradient à pas optimal

Initialiser le résidu r^0 à 1 et le compteur k à 0.

Tant que le résidu est plus grand que ε et que le compteur n'est pas trop grand :

- calculer la descente $\mathbf{w}^k = -\nabla J(\mathbf{u}^k)$,
- calculer $\rho^k \geq 0$ qui minimise $\rho \mapsto J(\mathbf{u}^k + \rho \mathbf{w}^k)$,
- poser $\mathbf{u}^{k+1} = \mathbf{u}^k + \rho^k \mathbf{w}^k$,
- calculer le résidu $r^{k+1} = \|\mathbf{u}^{k+1} - \mathbf{u}^k\|$,
- incrémenter le compteur.

8. Créer un script `scriptTP2_optimal.m` et reprendre les questions 5. et 6. pour la méthode du gradient à pas optimal. On utilisera par exemple l'algorithme de la section dorée vu au TP1 pour calculer le minimiseur de $\rho \mapsto J(\mathbf{u}^k + \rho \mathbf{w}^k)$ ou alors le calculer explicitement dans le cadre du problème considéré (4). Pour la question 6.a., on écrira une fonction `gradient_optimal.m` qui prend en argument A_N , \mathbf{f}_N , un point de départ \mathbf{u}^0 et un test d'arrêt ε , et qui renvoie le vecteur \mathbf{u}_N qui minimise J_N ainsi que le nombre d'itérations effectuées.

3.3 Méthode du gradient conjugué

Cette méthode n'est valable que pour des fonctionnelles de la forme $J(\mathbf{u}) = \frac{1}{2}(\mathbf{A}\mathbf{u}, \mathbf{u}) - (\mathbf{b}, \mathbf{u})$, où A est une matrice symétrique définie positive. On rappelle que l'algorithme du gradient conjugué pour une telle fonctionnelle, avec un point de départ \mathbf{u}^0 et un test d'arrêt ε préalablement définis, est donné par :

Méthode du gradient conjugué

Initialiser le résidu r^0 à $\|\mathbf{A}\mathbf{u}^0 - \mathbf{b}\|$, la descente \mathbf{w}^0 à $-(\mathbf{A}\mathbf{u}^0 - \mathbf{b})$, et le compteur k à 0.

Tant que le résidu est plus grand que ε et que le compteur n'est pas trop grand :

- calculer $\rho^k = -\frac{(\mathbf{A}\mathbf{u}^k - \mathbf{b}, \mathbf{w}^k)}{(\mathbf{A}\mathbf{w}^k, \mathbf{w}^k)}$,
- poser $\mathbf{u}^{k+1} = \mathbf{u}^k + \rho^k \mathbf{w}^k$,
- calculer la nouvelle descente $\mathbf{w}^{k+1} = -(\mathbf{A}\mathbf{u}^{k+1} - \mathbf{b}) + \frac{\|\mathbf{A}\mathbf{u}^{k+1} - \mathbf{b}\|^2}{\|\mathbf{A}\mathbf{u}^k - \mathbf{b}\|^2} \mathbf{w}^k$,
- calculer le résidu $r^{k+1} = \|\mathbf{A}\mathbf{u}^{k+1} - \mathbf{b}\|^2$,
- incrémenter le compteur.

9. Créer un script `scriptTP2_conjugué.m` et vérifier numériquement, pour quelques valeurs de N , que A_N est bien définie positive. On pourra utiliser par exemple la fonction `eig`.

10. Reprendre les questions 5. et 6. de la section 3.1 pour la méthode du gradient à pas optimal. Pour la question 6.a., on écrira une fonction `gradient_conjugué.m` qui prend en argument A_N , \mathbf{f}_N , un point de départ \mathbf{u}^0 et un test d'arrêt ε , et qui renvoie le vecteur \mathbf{u}_N qui minimise J_N ainsi que le nombre d'itérations effectuées.