Retour à IS104 – Algorithmique Numérique (http://mfaverge.vvv.enseirb-matmeca.fr/wordpress/?page_id=159)

Projet 2 : Méthode du gradient conjugué / Application à l'équation de la chaleur

Le but de ce projet consiste à implémenter des algorithmes derésolution de systèmes linéaires de grande taille, et à les appliquer à un problème de résolution d'équation aux dérivées partielles. Dans ce devoir, on considère uniquement des systèmes linéaires **symétriques**, **définis positifs** et **creux** (ne comportant que relativement peu d'éléments non nuls), et on exploite ces trois propriétés pour obtenir des algorithmes plus efficaces.

- la première partie s'intéresse à une méthode de factorisation de matrice symétrique appelée décomposition de Cholesky;
- la seconde partie construit un algorithme appelé « méthode du gradient conjugué » pour résoudre un système linéaire.
- enfin, la dernière partie utilise les algorithmes définis dans les deux premières parties afin de résoudre une équation aux dérivées partielles, l'équation de la chaleur.

Lors des tests, il est possible de comparer les algorithmes développés avec la méthode fournie par Scipy (`linalg.solve`), qui permet de comparer les résultats obtenus.

Décomposition de Cholesky

La décomposition de Cholesky est une décomposition matricielle d'une matrice symétrique définie positive A sous la forme d'un produit $T \cdot {}^tT$ où T est une matrice triangulaire inférieure, dont les coefficients sont obtenus à l'aide des équations suivantes :

$$\begin{cases} t_{i,i}^2 & = a_{i,i} - \sum_{k=1}^{i-1} t_{i,k}^2 \\ \\ t_{j,i} & = \dfrac{a_{i,j} - \sum_{k=1}^{i-1} t_{i,k} t_{j,k}}{t_{i,i}} \qquad j \geq i \end{cases}$$

Le résultat de la factorisation est la matrice T. Ceci permet de voir que toute matrice symétrique définie positive peut se mettre sous la forme d'un produit $T \cdot {}^tT$.

1. Écrire l'algorithme de factorisation dense de Cholesky et le tester. Quelle est sa complexité ?
2. En utilisant cet algorithme, combien coûte une résolution de système linéaire dense $Ax = b$?

La factorisation dense de Cholesky, bien qu'un algorithme privilégié pour la résolution de systèmes linéaires symétriques (du fait de sa stabilité numérique) est encore trop coûteuse lorsque la matrice est creuse (i.e ne contient que peu de valeurs non nulles). Il existe un algorithme de factorisation de Cholesky dans le cas creux, mais il est complexe à implémenter. Alternativement, la **factorisation de Cholesky incomplète** consiste à ne calculer qu'un sous-ensemble des éléments de la matrice T.

Dans ce projet, la factorisation incomplète correspond à la factorisation classique, dans laquelle on ne calcule pas les éléments $T[i,j]$ lorsque $A[i,j]$ est nul. Ces éléments restent alors à zéro et n'influent pas sur la suite du calcul de la factorisation.

3. Écrire un algorithme permettant de générer des matrices symétriques définies positives creuses avec un nombre de termes extra-diagonaux non nuls réglable.
4. Écrire l'algorithme de factorisation de Cholesky incomplète et le tester (quels tests réaliser ?).

   Quelle est sa complexité ? Penser à choisir de manière appropriée en fonction de quels paramètres établir cette complexité et expliquer en quoi elle est meilleure que celle de l'algorithme précédent.

Un **préconditionneur** M est une matrice facile à inverser telle que $\mathrm{cond}(M^{-1}A)$ soit inférieur à $\mathrm{cond}(A)$, où $\mathrm{cond}(A)$ est le conditionnement de la matrice A. En général, on choisit M comme un inverse approché de A ($M^{-1} \approx A^{-1}$) relativement facile à calculer.

5. Évaluer si la matrice ${}^tT^{-1} \cdot T^{-1}$ obtenue dans les deux cas précédents est un préconditionneur de bonne ou de mauvaise qualité (penser à utiliser `np.linalg.cond`).

> **Remarque :** Pour les évaluations de complexité, on pourra considérer que le produit matrices creux / vecteur, ainsi que la résolution d'un système linéaire triangulaire supérieure creux sont de complexité linéaire en le nombre d'éléments non nuls de la matrice.

Méthode du gradient conjugué

La **méthode du gradient conjugué** est une méthode numérique pour obtenir la solution d'un système d'équations linéaires $Ax = b$ où A est une matrice symétrique définie positive. Il s'agit généralement d'une méthode itérative, qui converge vers un vecteur solution X. La convergence de cette méthode utilise les propriétés du produit scalaire associé à la matrice A. La page Wikipedia (http://en.wikipedia.org /wiki/Conjugate_gradient_method) propose l'implémentation suivante en Matlab :

```
function [x] = conjgrad(A,b,x)
    r=b-A*x;
    p=r;
    rsold=r'*r;

    for i=1:10^(6)
        Ap=A*p;
        alpha=rsold/(p'*Ap);
        x=x+alpha*p;
        r=r-alpha*Ap;
        rsnew=r'*r;
        if sqrt(rsnew) < 1e-10
              break;
        end
        p=r+rsnew/rsold*p;
        rsold=rsnew;
    end
```

Le paramètre X sert d'initialisation à l'algorithme, mais peut être choisi comme égal au vecteur nul.

1. Par quels aspects cette implémentation ne respecte t'elle pas des standards de codage très sains ?
2. Par quel aspect cette méthode est-elle différente de la méthode décrite mathématiquement juste avant ? Quel est le gain de complexité occasionné ? Est-il systématique ?
3. Implémenter la méthode du gradient conjugué et la tester.

Il est possible d'utiliser cette méthode avec un préconditionneur, comme évoqué dans le paragraphe précédent. La même page (http://en.wikipedia.org/wiki/Conjugate_gradient_method) propose un algorithme avec préconditionneur (mais sans pseudo-code).

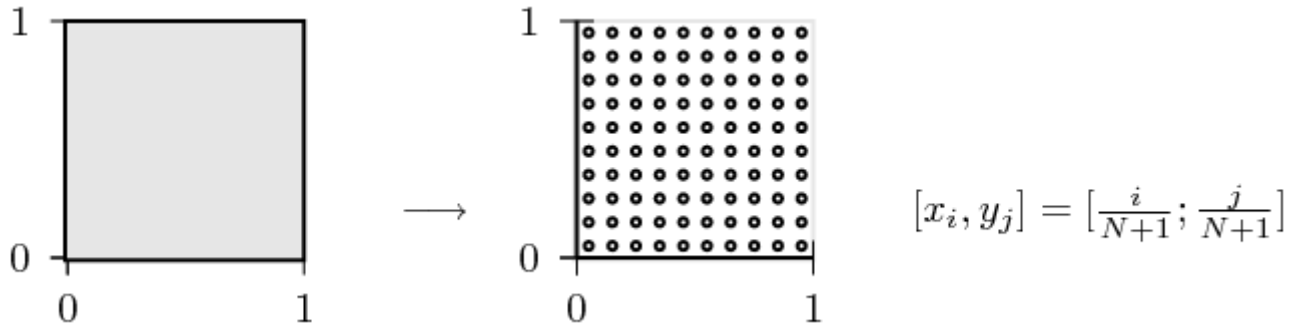
4. Implémenter la méthode du gradient conjugué avec préconditionneur et la tester.

Application à l'équation de la chaleur

Dans cette partie, on s'intéresse à la résolution d'un système linéaire particulier lié à une équation aux dérivées partielles connue : l'**équation de la chaleur** (en mode stationnaire). Concrètement, il s'agit de modéliser l'espace (plan) par un carré $[0;1] \times [0;1]$, et de calculer l'évolution de la température en chacun des points de cet espace. Si l'on note $T(x,y)$ la température en un point du carré, et $f(x,y)$ l'apport de chaleur extérieur en ce même point, alors l'équation est la suivante :

$$\left\{ \frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} = f(x,y) \qquad \forall (x,y) \in [0;1] \times [0;1] \right.$$

Notre but ici consiste à déterminer, étant donnée une fonction f, une fonction T qui soit une solution approchée de cette équation (en admettant qu'une telle solution existe). La méthode de résolution appliquée est appelée **méthode des différences finies**. Elle consiste à représenter l'espace $[0;1] \times [0;1]$ par un ensemble de $N \times N$ points choisis uniformément dans le carré.



Le domaine est alors vu comme un ensemble fini de points organisés sur un carré, et les fonctions T et f sont des matrices carrées associant à un point la valeur correspondant à ce point. De plus, les dérivées partielles sont approximées par les équations linéaires suivantes :

$$\left(\frac{\partial^2 T}{\partial x^2}\right)_{i,j} \approx \frac{T(x_i+h, y_j) + T(x_i-h, y_j) - 2T(x_i, y_j)}{h^2} = \frac{t_{i+1,j} + t_{i-1,j} - 2t_{i,j}}{h^2}$$

$$\left(\frac{\partial^2 T}{\partial y^2}\right)_{i,j} \approx \frac{T(x_i, y_j+h) + T(x_i, y_j-h) - 2T(x_i, y_j)}{h^2} = \frac{t_{i,j+1} + t_{i,j-1} - 2t_{i,j}}{h^2}$$

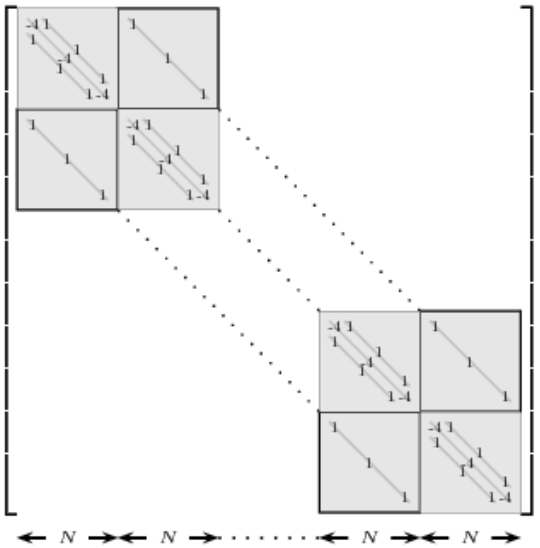
où H est la distance entre deux points consécutifs sur la même ligne ou la même colonne, c'est-à-dire ici $h = \frac{1}{N+1}$.

Conditions aux bords de Dirichlet : Pour bien définir le problème précédent, il est nécessaire de se donner des conditions aux limites sur les bords du domaine. Les conditions les plus simples sont les suivantes :

$$\begin{cases} T(0, y) = T(1, y) = 0 \\ T(x, 0) = T(x, 1) = 0 \end{cases} \qquad \forall (x, y) \in [0; 1] \times [0; 1]$$

1. Vérifier que le problème peut se mettre alors sous la forme d'un système linéaire $Ax = b$ où la matrice A est une matrice tridiagonale par blocs de taille $N^2 \times N^2$ et de la forme suivante :

$$\left(\frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial x^2}\right)\bar{T} = \frac{1}{h^2}$$



$$\cdot \bar{T} = \frac{1}{h^2} M_c \cdot \bar{T}$$

(images/p2_heat_matrix_large.png)

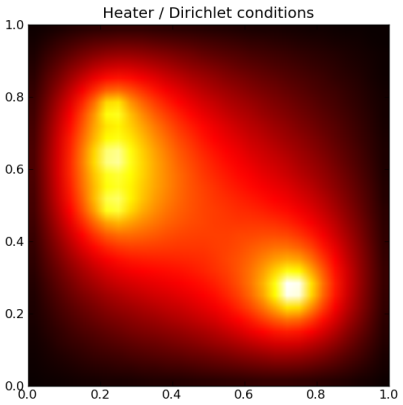
Noter que la solution X est un **vecteur** de N^2 lignes, qu'il s'agira de retransformer en une matrice

$(T_{i,j}) : N \times N$ telle que $T_{i,j} = x_{i*N+j}$.

L'entier N compte le nombre de points par lignes et par colonnes dans le domaine. Écrire un algorithme permettant de générer la matrice A et le vecteur b en fonction du nombre N.

2. Résoudre le système dans le cas d'un radiateur placé au centre du carré (quel est le vecteur image ?). Tracer l'image du résultat.
3. Résoudre le système dans le cas d'un mur chaud placé au nord. du carré (quel est le vecteur image ?). Tracer l'image du résultat.

Pour exemple, voilà le type d'image que l'on peut obtenir en résolvant certains de ces systèmes:



Dans cette section

Projet 5 : Interpolation and integration methods / Cubic splines and surface interpolation (http://mfaverge.vvv.enseirb-matmeca.fr/wordpress/?page_id=304)

Projet 6 : Résolution approchée d'équations différentielles / Modélisation de systèmes dynamiques (http://mfaverge.vvv.enseirb-matmeca.fr/wordpress/?page_id=309)

Construit avec ♥ par Thèmes Graphene (https://www.graphene-theme.com/).