TP1 : Résolution de l'équation de Poisson

1 Introduction

On souhaite résoudre l'équation de Poisson avec des conditions aux limites de type Dirichlet homogène :

$$\begin{cases} -\Delta u(x,y) = f(x,y), & (x,y) \in \Omega \\ u(x,y) = 0, & (x,y) \in \partial\Omega \end{cases} \tag{1}$$

où $u(x,y)$ est la température au point (x,y), la fonction f est un terme source nul sur tout le domaine, mais valant 1 au milieu et $\Omega = [0,1] \times [0,1]$. L'équation sera discrétisée par différences finies afin d'obtenir un système linéaire. La résolution du système linéaire se fera par différentes méthodes, le but étant de comparer les méthodes entre elles.

2 Schéma numérique

On considère les deux subdivisions spatiales x_i et y_j telles que :

$$x_i = i\Delta x, \quad i \in [\![1,n]\!] \quad ; \quad y_j = j\Delta x, \quad j \in [\![1,n]\!] \tag{2}$$

avec $\Delta x = \frac{1}{n+1}$ le pas de la subdivision. On peut montrer que le schéma numérique s'écrit :

$$\frac{4u_{i,j} - u_{i+1,j} - u_{i-1,j} - u_{i,j+1} - u_{i,j-1}}{\Delta x^2} = f_{i,j}, \quad i,j \in [\![1,n]\!] \tag{3}$$

avec $u_{n+1,j} = u_{0,j} = u_{i,n+1} = u_{i,0} = 0$, $i,j \in [\![1,n]\!]$. On obtient ainsi le système linéaire suivant :

$$AU = F,\ A = \frac{1}{\Delta x^2} \begin{pmatrix} H & -I & 0 & \dots & 0 \\ -I & H & -I & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & -I & H & -I \\ 0 & \dots & 0 & -I & H \end{pmatrix} U = \begin{pmatrix} u_{1,1} \\ \vdots \\ \vdots \\ \vdots \\ u_{n,n} \end{pmatrix},\ F = \begin{pmatrix} f_{1,1} \\ \vdots \\ \vdots \\ \vdots \\ f_{n,n} \end{pmatrix} \tag{4}$$

où la matrice I est l'identité de taille n et H :

$$H = \begin{pmatrix} 4 & -1 & 0 & \dots & 0 \\ -1 & 4 & -1 & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & -1 & 4 & -1 \\ 0 & \dots & 0 & -1 & 4 \end{pmatrix} \tag{5}$$

On remarque que A est symétrique définie positive.

3 Rappels sur les méthodes itératives

Ceci est un court rappel sur l'itéré x_k des méthodes itératives que l'on va utiliser pour résoudre le système de type $Ax = b$.

Méthode de Jacobi

Pour cette méthode, l'itéré s'écrit :

$$x_{k+1} = x_k + D^{-1}(b - Ax_k) \tag{6}$$

où D est la diagonale de A. La méthode de Jacobi converge si A est à diagonale strictement dominante. Dans notre cas, elle ne l'est pas strictement, donc on peut s'attendre à ce que la méthode ne converge pas.

Méthode de relaxation

$$x_{k+1} = x_k + M^{-1}(b - Ax_k), \quad M = \frac{D}{\omega} - E \tag{7}$$

où $-E$ est la partie triangulaire inférieure de A, w est un paramètre tel que $w \in]0, 2[$. Si A est symétrique définie positive, alors la méthode converge.

Méthode du gradient conjugué

$$x_{k+1} = x_k + \alpha_k d_k \tag{8}$$

où α_k est le pas de descente et d_k est la direction de descente. Cette méthode converge si A est symétrique.

Préconditionnement

En constate que plus le conditionnement de A est élevé, plus la convergence est lente. Donc le but est de choisir une matrice M tel que le conditionnement de $M^{-1}A$ soit plus petit que celui de A. Dans notre cas, on va utiliser le préconditionneur de la factorisation de Cholesky incomplète qui consiste à chercher une approximation de la décomposition de Cholesky de A. Ce préconditionneur sera utilisé pour la méthode du gradient conjugué.

4 Résolution numérique

Le but est d'inverser la matrice A par différentes méthodes afin de trouver U. On utilisera dans notre cas seulement des méthodes itératives : le gradient conjugué sans préconditionneur et avec le préconditionneur de la factorisation de Cholesky incomplète, la méthode de Jacobi et la méthode de relaxation. L'objectif est de déterminer la meilleure méthode pour le problème considéré, c'est-à-dire dire déterminer la méthode la moins couteuse.

4.1 Matrice creuse

Le langage `Python` sera utilisé pour la résolution du problème. On peut remarquer que la matrice A est très creuse, c'est pourquoi on utilisera le module `sparse` de la bibliothèque `scipy` pour réaliser la construction de la matrice et la réalisation des produits matrice vecteur lors de l'implémentation des méthodes de résolution. Le module `sparse` permet de construire une matrice en ne stockant seulement les coefficients non nuls de la matrice, permettant ainsi d'utiliser moins de mémoire et de réaliser les produits matrice vecteur plus rapidement.

4.2 Résultat

Le vecteur U renvoyé par les quatre méthodes étant le même, seul le résultat renvoyé par le gradient conjugué sera présenté ici. Pour $n = 50$, on obtient :

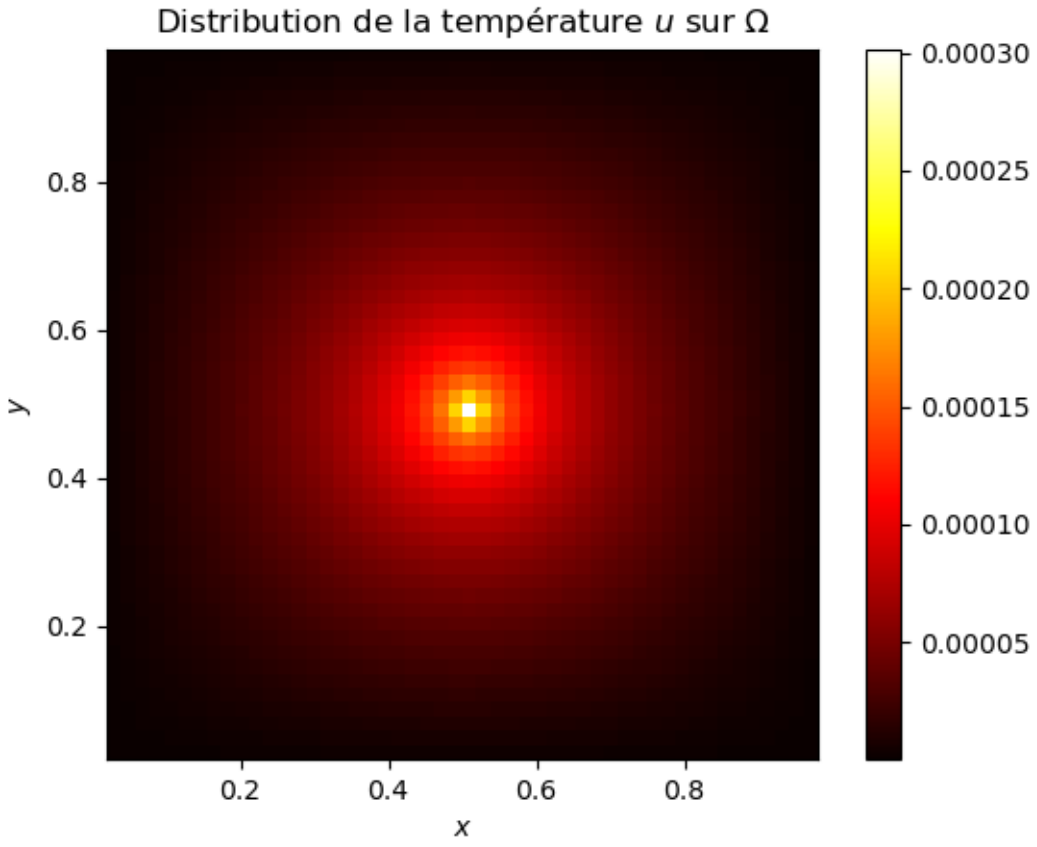


FIGURE 1 – Distribution de la température sur le domaine

On peut voir sur cette figure la diffusion de la température : plus on s'éloigne du centre, donc du terme source, plus la température tend vers zéro, car on a imposé des conditions de Dirichlet homogène sur le bord.

4.3 Comparaison

La différence entre les méthodes itératives utilisées réside dans le cout de calcul. Certaines méthodes sont plus rapides, d'autres nécessitent moins d'itérations, etc. Pour comparer ces méthodes, on peut par exemple visualiser l'évolution du résidu relatif $\frac{||r||}{||F||}$ en fonction du nombre d'itérations k ($n = 50$) :

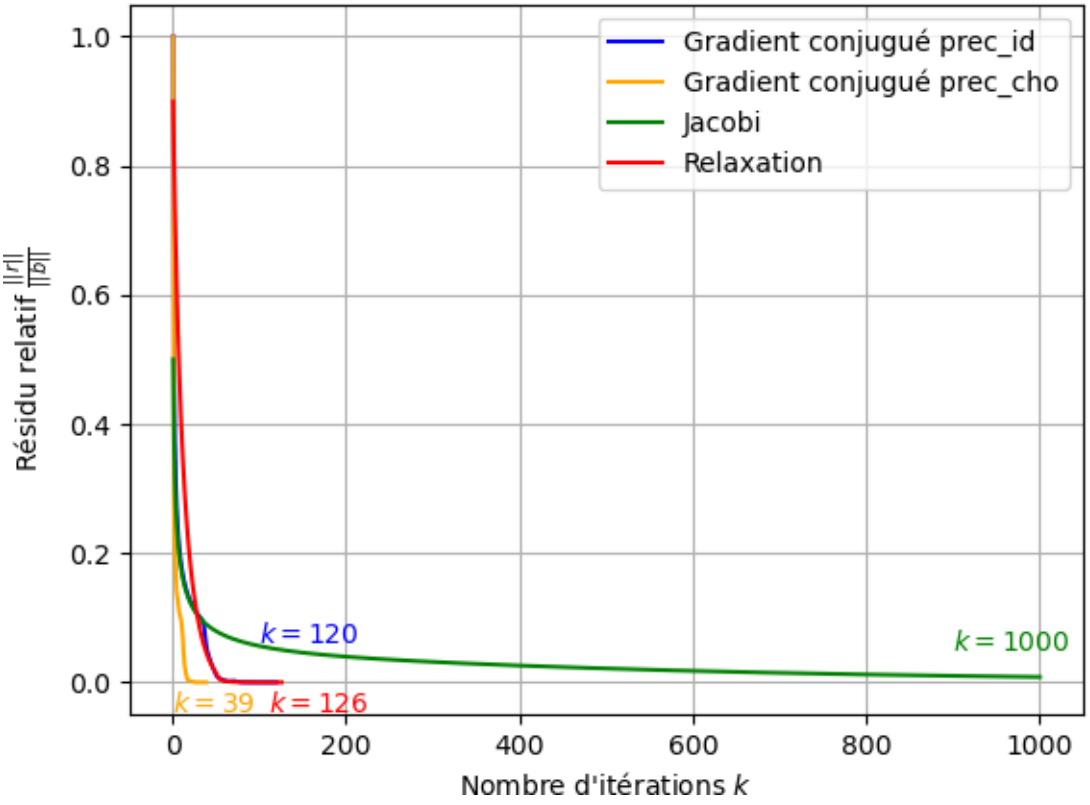


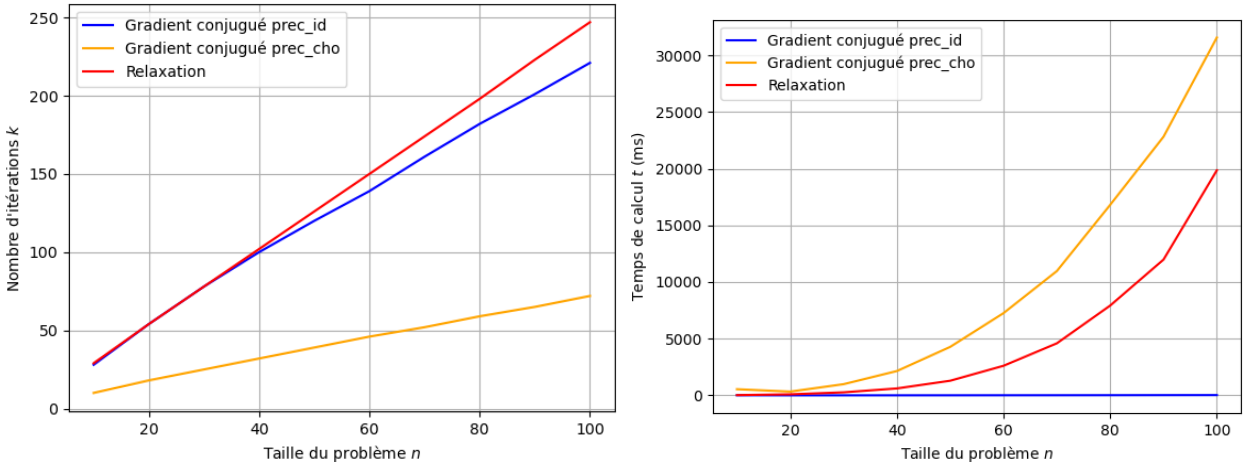
FIGURE 2 – Résidu relatif en fonction du nombre d'itérations

La méthode de relaxation a été prise avec $\omega = 2/(1 + \sin(\pi\Delta x))$, ce qui correspond à la valeur optimale vue en cours.
On peut voir sur la figure 2 que la méthode du gradient conjugué avec le préconditionneur de la factorisation de Cholesky incomplète a nécessité le moins d'itérations ($k = 39$) suivi du gradient conjugué avec le préconditionneur identité ($k = 120$) et de la méthode de relaxation ($k = 126$).
On peut observer que la méthode de Jacobi n'a pas encore convergé, en effet, le calcul s'est arrêté après 1000 itérations, ce qui correspond à limite que l'on a fixé, on peut donc supposer que cette n'est pas efficace pour le problème considéré comme prédit par la théorie, car A n'est pas à diagonale strictement dominante. On peut également calculer le temps de calcul pour les différentes méthodes afin que le résidu relatif soit inférieur à ϵ ($= 10^{-6}$). Les résultats sont référencés dans le tableau suivant pour $n = 50$:

Méthode	Nombre d'itérations k	Temps de calcul (en ms)
Gradient conjugué prec_id	120	72
Gradient conjugué prec_cho	39	4798
Jacobi	5754	18859
Relaxation	126	2130

TABLE 1 – Temps de calcul pour converger

La méthode avec le préconditionneur de Cholesky nécessite le moins d'itérations, mais n'est pas plus rapide que la méthode avec le préconditionneur identité. Cela est dû en raison du précontionneur car en passant par la factorisation de Cholesky incomplète, on a besoin de résoudre deux systèmes triangulaires, ce qui semble être couteux en temps. Dans la suite, la méthode de Jacobi ne sera plus considérée en raison des précédents résultats. On cherche à présent à déterminer le solveur le plus robuste, c'est-à-dire celui dont le nombre d'itérations varie le moins possible en fonction de la taille du système (ici n^2).



(a) Nombre d'itérations

(b) Temps de calcul

FIGURE 3 – Coût de calcul en fonction du paramètre n

Sur la figure 3a, on peut voir que le nombre d'itérations augmente avec le paramètre n de la discrétisation. Le nombre d'itérations augmente moins rapidement pour le gradient conjugué avec préconditionneur, ceci montre que le préconditionneur choisi joue bien son rôle, c'est-à-dire que le préconditionneur M utilisé est tel que $\kappa(M^{-1}A) < \kappa(A)$ avec κ le conditionnement. Au vu de ces résultats, on peut supposer que cette méthode est plus robuste que les deux autres. Cependant, sur la figure 3b on voit que le temps de calcul augmente très fortement avec n, on observe une trentaine de secondes si $n = 100$. Pour de grande valeur de n cette méthode semble être la plus lente. La méthode du gradient sans préconditionneur (préconditionneur identité) est la plus rapide, le temps de calcul semble être indépendant du paramètre n. La méthode de relaxation a une augmentation plus importante d'itérations plus n augmente et son temps de calcul est légèrement inférieur au gradient conjugué avec préconditionneur. Donc l'utilisation de cette méthode pour le problème considéré ne présente par de réel avantage.

5 Conclusion

Ce TP a permis de résoudre l'équation de Poisson par différences finies, le problème a été traité en tenant compte du fait que la matrice est creuse en utilisant des bibliothèques `Python` permettant la gestion de ce type de matrice. Le système linéaire obtenu a été résolu par plusieurs méthodes itératives. La méthode de Jacobi n'est pas adaptée à ce problème en raison de son nombre d'itérations trop élevé. L'utilisation de la méthode du gradient conjugué avec le préconditionneur de la factorisation de Cholesky incomplète est plutôt conseillé en raison du faible nombre d'itérations et semble la plus robuste pour ce problème. Le gradient conjugué sans préconditionneur peut être utilisé pour son faible temps de calcul. La méthode de relaxation ne présente pas de réel avantage par rapport au gradient conjugué. Les deux méthodes utilisant le gradient conjugué semblent être les plus appropriées au problème.