5A ModIA

Rapport de TP

Méthodes de Krylov préconditionnées

Elèves :
Karima GHAMNIA
Cassandra MUSSARD

Enseignant :
Ronan GUIVARCH

12 janvier 2025

1 Introduction

L'objectif de ce TP est de résoudre une EDP avec une méthode itérative. Parmi ces méthodes, on retrouve les méthodes de Krylov, qui convergent plus rapidement lorsque le système à résoudre se rapproche de la matrice "identité". Pour se mettre dans ce cas là, nous pouvons appliquer des techniques de préconditionnement, où on résout le système linéaire équivalent suivant :

$$M^{-1}Ax = M^{-1}b$$

Avec M, la matrice préconditionneur, qui doit représenter la meilleure approximation possible de A.
Dans la suite, nous étudions l'utilisation de différents préconditionneurs (identité, Jacobi, et Cholesky)

2 Résolution d'un système linéaire associé à une matrice issue de la discrétisation d'une EDP

2.1 Vérification de l'utilisation de l'algorithme du gradient conjugué

Dans ce TP, nous souhaitons utiliser l'algorithme du gradient conjugué. Cependant, pour faire cela il faut que la matrice A (issue de l'EDP) vérifie les deux conditions suivantes :

— A doit être symétrique ($A = A^T$)
— A doit être définie positive (les valeurs propres de A sont toutes positives).

Nous avons utilisé la fonction "isequal" pour vérifier que les matrices A et A^T sont égales. De plus, on vérifie avec la fonction "eig" de Matlab que les valeurs propres sont positives (cette partie du code est commentée par la suite).
Nous avons vu que la matrice A vérifie pour chaque maillage les 2 propriétés, nous pouvons donc utiliser l'algorithme du gradient conjugué.

2.2 Résultats

Lorsque nous sommes en présence d'un système linéaire assez grand, les méthodes de factorisation sont très coûteuses. De plus, pour les méthodes itératives la convergence

peut parfois être très longue. Dans ces deux cas, l'utilisation de préconditionneurs permet d'accélérer le temps de convergence et de diminuer le coût de calcul.
Dans ce TP, nous cherchons à comparer l'efficacité (convergence, nombre d'itérations, temps de calcul du préconditionneur et la résolution) des 4 préconditionneurs suivants :

— Préconditionneur identité (pas de pré-conditionnement).

— Préconditionneur de Jacobi.

— Préconditionneur issu de la factorisation incomplète de Cholesky sans remplissage.

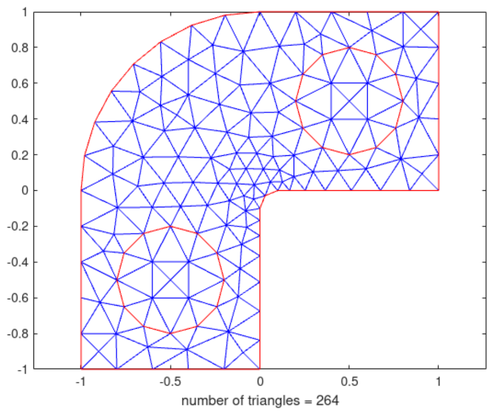
— Préconditionneur issu de la factorisation incomplète de Cholesky avec seuillage.

Nous pouvons préciser que la complexité de ces algorithmes est croissante. En effet, le préconditionneur de Jacobi est le plus simple car il utilise seulement les éléments de la diagonale. Le préconditionneur de Cholesky sans remplissage est un peu plus complexe car il essaye de conserver la structure creuse de la matrice initiale. Enfin, le dernier précondtionneur est le plus complexe car il accepte un certain degré de remplissage qui est contrôlé par un seuil.

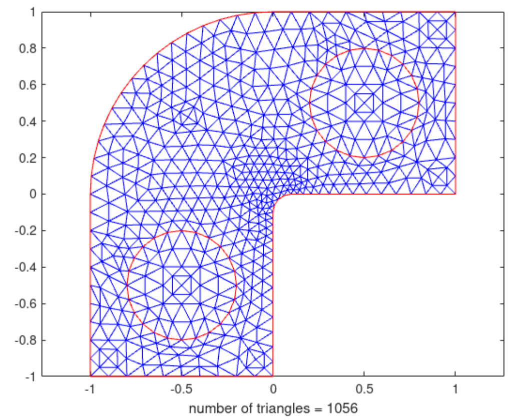
2.3 Comparaison des préconditionneurs

Nous avons a disposition 4 niveaux de raffinage (de 0 à 3), qui dans l'ordre croissant, correspondent à un maillage plus fin (voir figure 1). Ceci correspond à une augmentation de la densité des points de la grille (le nombre de triangle augmente), ainsi que la complexité du problème. Chaque niveau de raffinage est modélisé par une couleur dans les graphiques qui vont suivre.

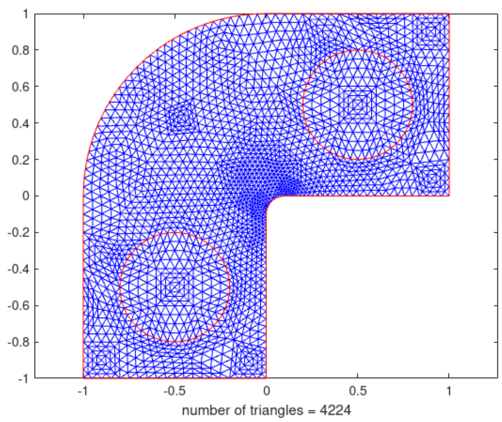
— Courbe verte ⇒ raffinage 0
— Courbe rouge ⇒ raffinage 1
— Courbe bleue ⇒ raffinage 2
— Courbe violette ⇒ raffinage 3



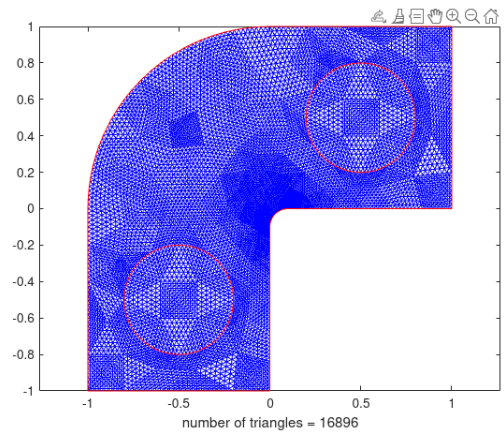
(a) Raffinage niveau 0



(b) Raffinage niveau 1



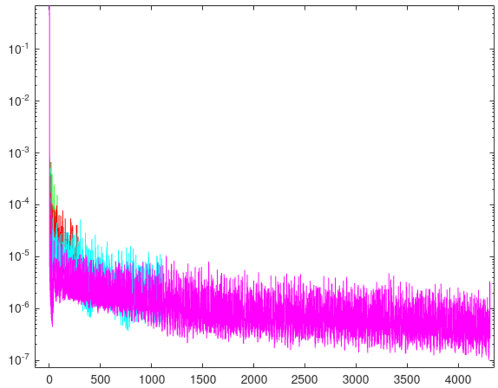
(c) Raffinage niveau 2



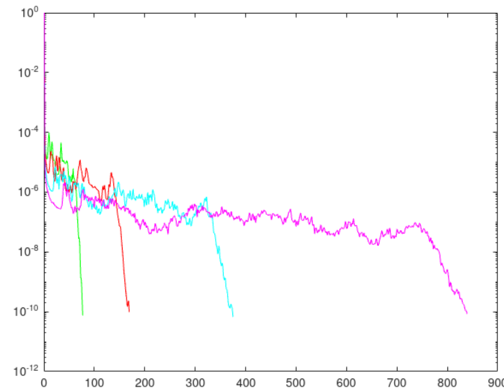
(d) Raffinage niveau 3

FIGURE 1 – Différents niveaux de raffinages

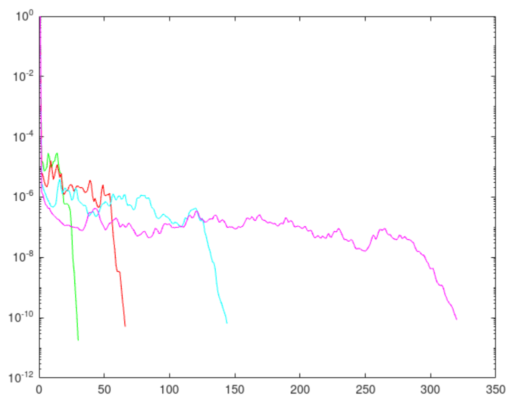
Rapport



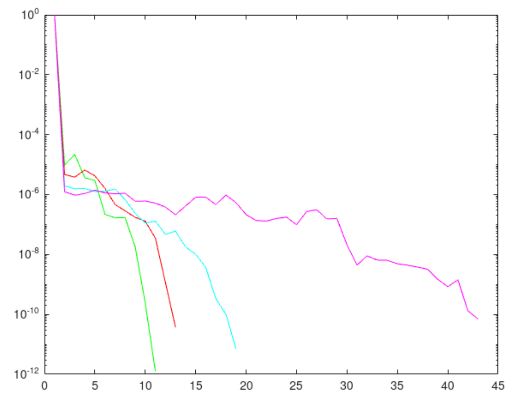
(a) sans préconditionneur



(b) préconditionneur de Jacobi(diagonal)



(c) préconditionneur issu de la factorisation Cholesky



(d) préconditionneur Cholesky incomplet avec $tol = 10^{-3}$

FIGURE 2 – Convergence de la solution avec les différents préconditionneurs

Analysons maintenant les courbes de convergence pour chacun des préconditionneurs. Nous observons sur chaque graphique de la figure 2 le nombre d'itérations effectué en fonction de l'erreur.
On constate que le nombre d'itérations effectué est décroissant par rapport à la complexité du préconditionneur choisit. En effet, sans préconditionneur on atteint plus de 4000 itérations contre environ 45 avec le préconditionneur de Cholesky avec seuil.
Ensuite, on peut aussi remarquer que plus le maillage est fin plus les différents préconditionneurs ont besoin d'itérations pour pouvoir résoudre le problème.
Enfin, on peut noter qu'on a bien convergence pour tous les préconditionneurs utilisés. Par contre, là aussi on peut voir que l'erreur commise est décroissante par rapport au préconditionneur choisit (on fait beaucoup moins d'erreur, environ 10^{-12}, avec le préconditionneur de Cholesky avec seuil). L'erreur reste assez similaire pour chaque niveau de raffinage choisit.

Convergence avec les différents préconditionneurs

Le tableau 1, présente les résultats obtenus sans utilisation de préconditionneur. Nous remarquons une augmentation significative du nombre d'itérations et du temps total (construction+résolution) avec la taille du problème (niveau de raffinage). Effectivement, comme nous l'avons constaté précédemment en analysant la figure 2, la convergence est plus lente et le problème est plus complexe lorsqu'aucun préconditionneur n'est utilisé.

Table 1 – Préconditionneur identité (aucun)

Niveau de raffinage	Taille du problème	Nb d'itérations	Temps de construction	Temps de résolution	Temps total
0	155	177	4.81e-04	3.4794e-0.2	3.5275e-02
1	573	239	4.4e-04	1.863440e-01	1.867840e-01
2	2201	769	3.105e-03	6.63803	6.641135
3	8625	4300	1.1941e-02	3.291699e+02	3.291818e+02

Le deuxième tableau 2, présente les performances du préconditionneur diagonal de Jacobi. Nous pouvons constater un nombre d'itération et un temps total assez réduits comparé au cas précedent (sans préconditionneur), ce qui indique une meilleure convergence ainsi qu'une amélioration globale des performances.

Table 2 – Préconditionneur de Jacobi (préconditionneur diagonal)

Niveau de raffinage	Taille du problème	Nb d'itérations	Temps de construction(s)	Temps de résolution(s)	Temps total(s)
0	155	76	3.196e-03	8.32e-03	1.1516e-02
1	573	168	6.18e-04	5.5958e-02	5.6576e-02
2	2201	374	3.486e-03	9.0388e-01	9.07366e-01
3	8625	838	1.3388e-02	3.126444e+01	3.127783e+01

Les troisième (Tableau 3) et quatrième (Tableau 4) tableaux montrent les résultats obtenus avec des préconditionneurs basés sur la factorisation incomplète de Cholesky (avec et sans tolérance spécifiée). Nous constatons une meilleure réduction en nombre d'itérations et en temps total, en particulier lorsque la tolérance est spécifiée. Nous obtenons ainsi le nombre d'itérations le plus bas et le temps total le plus court parmi tous les préconditionneurs examinés.

Impact de la tolérance sur les preconditionneurs IC

Les tableaux 4, 5, et 6 présentent les résultats obtenus avec un préconditionneur IC à différents niveaux de tolérance : 10^{-3}, 5.10^{-4}, et 10^{-4}. En comparant ces tableaux, nous pouvons constater que la tolérance a un impact important sur la précision des préconditionneurs IC.
Une tolérance stricte (tol très petit) améliore la précision, mais augmente le temps total (construction et résolution). Cependant, en comparant les cas "$tol = 10^{-4}$", et

TABLE 3 – Préconditionneur issu de la factorisation incomplète de Cholesky sans remplissage

Niveau de raffinage	Taille du problème	Nb d'itérations	Temps de construction(s)	Temps de résolution(s)	Temps total(s)
0	155	29	4.661e-03	7.016e-03	1.1677e-02
1	573	65	2.85e-04	2.623e-03	2.908e-03
2	2201	143	5.24e-04	1.4338e-02	1.4862e-02
3	8625	319	1.457e-03	1.27765e-01	1.29222e-01

TABLE 4 – Préconditionneur issu de la factorisation incomplète de Cholesky (IC) avec $tol = 10^{-3}$

Niveau de raffinage	Taille du problème	Nb d'itérations	Temps de construction(s)	Temps de résolution(s)	Temps total(s)
0	155	10	5.36e-03	6.78e-04	6.038e-03
1	573	12	2.2484e-02	1.185e-03	2.3669e-02
2	2201	18	1.15723e-01	5.093e-03	1.20816e-01
3	8625	42	5.08177e-01	4.196e-02	5.50137e-01

"$tol = 5.10^{-4}$" on constate qu'une tolérance trop stricte peut entraîner des coûts de calcul excessifs, car la matrice M est plus élevée que la A en raison du "fill-in" (voir figure 3, 4, et 5) sans améliorer significativement les performances du solveur. Il faut donc bien définir la tolérance selon la priorité (précision vs coût de calcul)

TABLE 5 – Préconditionneur issu de la factorisation incomplète de Cholesky (IC) avec $tol = 5.10^{-4}$

Niveau de raffinage	Taille du problème	Nb d'itérations	Temps de construction(s)	Temps de résolution(s)	Temps total(s)
0	155	9	1.0881e-02	2.6547e-02	3.7428e-02
1	573	12	2.5439e-02	7.025e-03	3.2464e-02
2	2201	16	1.28467e-01	5.231e-03	1.33698e-01
3	8625	45	6.194520e-01	5.208100e-02	6.715330e-01

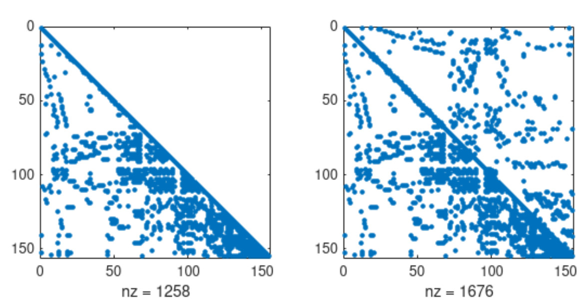


FIGURE 3 – Structure de A (à droite) et M pour $tol = 10^{-3}$

TABLE 6 – Préconditionneur issu de la factorisation incomplète de Cholesky (IC) avec $tol = 10^{-4}$

Niveau de raffinage	Taille du problème	Nb d'itérations	Temps de construction(s)	Temps de résolution(s)	Temps total(s)
0	155	6	7.563e-03	5.501e-03	1.3064e-02
1	573	8	3.2408e-02	1.244e-03	3.3652e-02
2	2201	11	1.8385e-01	3.684e-03	1.87534e-01
3	8625	16	9.54996e-01	2.4997e-02	9.79993e-01

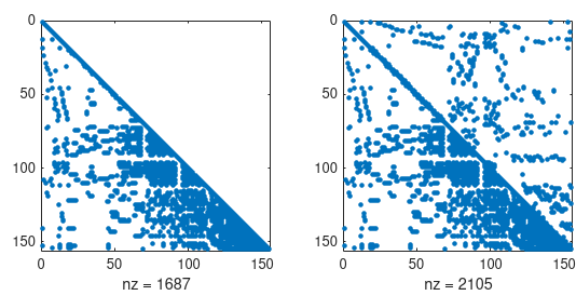


FIGURE 4 – Structure de A (à droite) et M pour $tol = 10^{-4}$

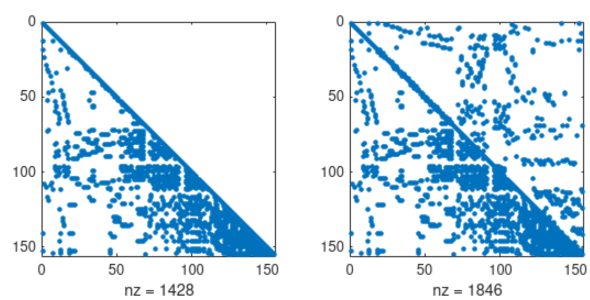


FIGURE 5 – Structure de A (à droite) et M pour $tol = 5.10^{-4}$

3 Conclusion

En conclusion, l'utilisation des préconditionneurs est une méthode efficace pour améliorer les performances des méthodes itératives (Krylov dans notre cas) dans la résolution de systèmes linéaires. L'application des préconditionneurs (identit e, Ja- cobi, ou IC) accelère la convergence, et diminue les coûts de calcul. De plus, nous avons constaté que le préconditionneur de Cholesky avec une tolérance bien choisie permet d'avoir la solution la plus précise et rapide.