Compte-rendu du TP2
Algèbre Linéaire Creuse : méthodes directes

Ines BESBES
Sara ROOL

March 2025

1 Introduction

1.1 Contexte

Dans ce TP, on cherche à résoudre le système linéaire : $Ax = b$, avec A une matrice creuse et symétrique en utilisant une factorisation de Cholesky. Toutefois, l'efficacité de cette méthode dépend fortement du remplissage (fill-in), c'est-à-dire du nombre de nouveaux éléments non nuls qui apparaissent dans les facteurs de A après factorisation. En effet, un fill-in excessif augmente le temps de calcul en raison du nombre croissant d'opérations arithmétiques nécessaires. De plus, cela consomme plus de mémoire, rendant problématique le traitement des matrices de grande taille.
Ainsi, pour limiter le fill-in et améliorer l'efficacité de la factorisation, on applique les stratégies de réordonnancement suivantes:

- Approximate Minimum Degree
- Symmetric Approximate Minimum Degree
- Symmetric Reverse Cuthill-McKee
- Column Approximate Minimum Degree

1.2 Objectif

L'objectif est donc d'analyser l'efficacité des stratégies de permutation pour limiter le fill-in lors de la factorisation de Cholesky. On va tester cela avec 5 matrices symétriques A différentes :

- $A = mat0$ de dimension 155*155
- $A = mat1$ de dimension 573*573
- $A = mat2$ de dimension 2201*2201
- $A = mat3$ de dimension 8625*8625
- $A = bcsstk27$ de dimension 1224*1224

2 Nombre d'opérations de la phase de résolution

Pour résoudre les deux systèmes linéaires triangulaires après une factorisation de Cholesky ($\boldsymbol{Ly} = \boldsymbol{b}$ et $\boldsymbol{L}^\top \boldsymbol{x} = \boldsymbol{y}$), le nombre total d'opérations flottantes dépend du nombre de termes non nuls (nnz) dans la matrice L.
Chaque terme non nul $L_{i,j}$ $(i \geq j)$ génère :

- 2 flops pour $\boldsymbol{Ly} = \boldsymbol{b}$ (combinaison linéaire + division)
- 2 flops pour $\boldsymbol{L}^\top \boldsymbol{x} = \boldsymbol{y}$ (même coût par symétrie)

Ainsi, on obtient:

$$\text{Total de flops} = (Ly = b) + (L^T x = y) \approx 4 \cdot \text{nnz}(L)$$

avec nnz(L) le nombre de termes non nuls dans L.

3 Qualité de la solution

Après chaque résolution, on évalue la qualité de la solution obtenue en calculant différentes erreurs.

3.1 Erreur backward

L'erreur backward mesure à quel point x_1 est une solution "exacte" pour une version légèrement perturbée de A et b.

$$\text{Erreur}_{backward} = \frac{\|Ax_1 - b\|}{\|b\|}$$

3.2 Erreur forward

L'erreur forward mesure l'écart entre la solution x_1 obtenue et la solution de référence x_{ref} (qui est calculée avec $x_{ref}= A\backslash b$ qui utilise une méthode numérique comme la factorisation de Cholesky). On la calcule comme suit :

$$\text{Erreur}_{forward} = \frac{\|x_1 - x_{\text{ref}}\|}{\|x_{\text{ref}}\|}$$

4 Résolution du système linéaire symétrique avec une factorisation de Cholesky

4.1 Vérification des matrices

Avant d'effectuer une factorisation Cholesky, il est nécessaire de vérifier que les matrices dont on dispose sont toutes symétriques définies positives. Pour ce faire, on a vérifié numériquement que l'on pouvait bien appliquer la factorisation de Cholesky à chaque matrice avec la fonction *chol(A)*. Cela a fonctionné pour toutes les matrices.

4.2 Résultats pour mat0

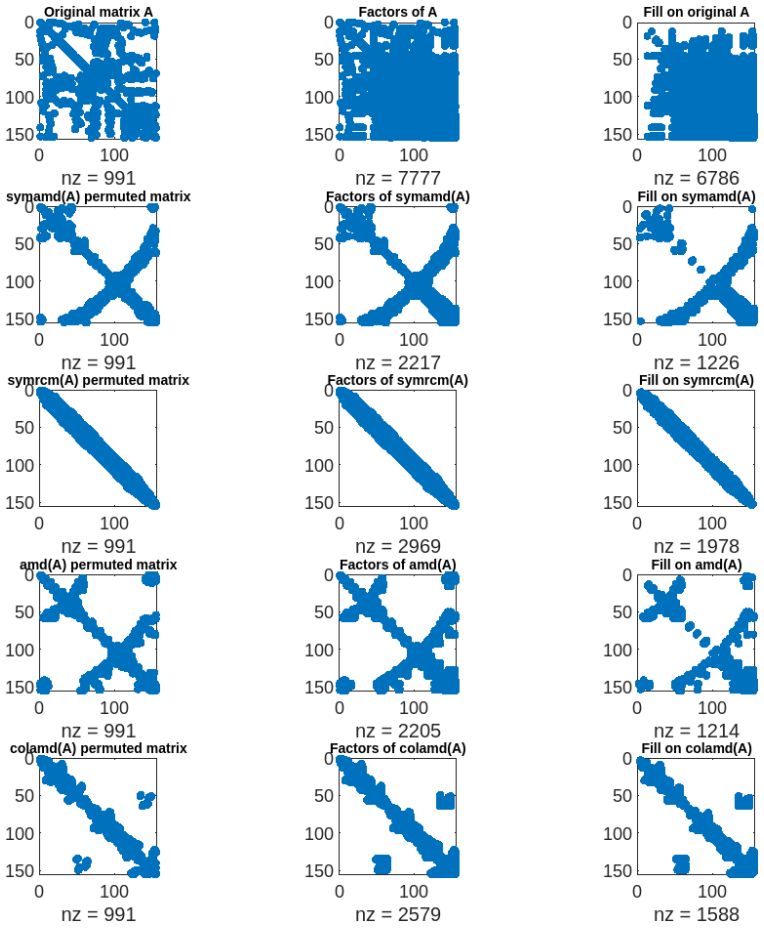

Figure 1: Visualisation des matrices et du fill-in après réordonnancement pour *mat*0

Paramètre	**Sans perm.**	**AMD**	**SymAMD**	**SymRCM**	**ColAMD**
Temps (s)	0.0005	0.0004	0.0002	0.0001	0.0001
FLOPs	1.59×10^4	4.72×10^3	4.74×10^3	6.25×10^3	5.47×10^3
Fill-in	2975	189	195	571	376
NNZ de L	3966	1180	1186	1562	1367
Mémoire (KB)	63.19	19.66	19.75	25.62	22.58
Backward error	1.23×10^{-12}	1.53×10^{-12}	1.20×10^{-12}	1.44×10^{-12}	1.72×10^{-12}
Forward error	1.08×10^{-13}	2.06×10^{-16}	1.18×10^{-13}	9.58×10^{-14}	1.51×10^{-14}
Gain FLOPs (%)	0.00%	70.25%	70.10%	60.62%	65.53%

Table 1: Comparaison des méthodes de réordonnancement pour la factorisation de Cholesky

Interprétation : On remarque qu'AMD et SymAMD sont les méthodes les plus efficaces pour réduire le nombre d'opérations (environ 70% de gain), le remplissage (moins de 200 fill-in) et l'utilisation de la mémoire (moins de 20KB), tout en restant très précises.

Globalement, toutes les permutations ont une erreur backward et forward très faible, qui varient de 10^{-12} à 10^{-16}, qui suggèrent que la solution est correcte. ColAMD est très rapide et offre un bon compromis entre performance et coût de calcul. On observe également sur la Figure 1 que SymRCM est efficace pour réduire la bande de la matrice. Il est également très rapide, mais son gain FLOPs est légèrement inférieur aux autres méthodes (environ 60% de gain).

Enfin, l'absence de permutation entraîne un coût de calcul bien plus élevé et une utilisation mémoire excessive.

4.3 Résultats pour mat1

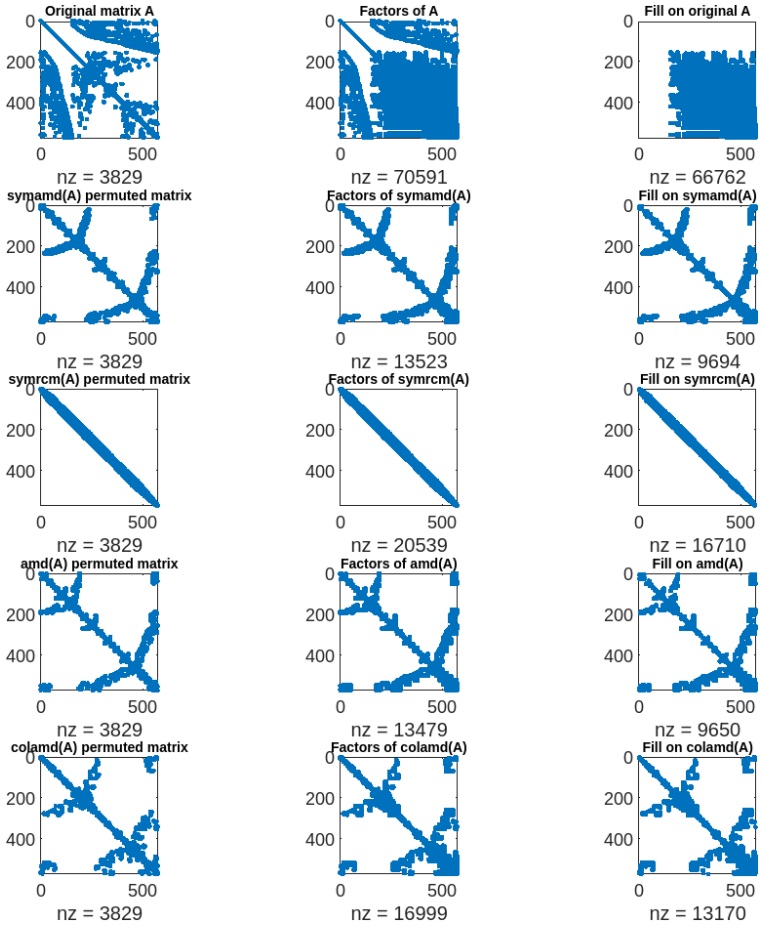

Figure 2: Visualisation des matrices et du fill-in après réordonnancement pour *mat*1

Interprétation : Comme précédemment, AMD et SymAMD sont les meilleures méthodes, réduisant fortement le nombre d'opérations, le remplissage et l'utilisation de la mémoire, tout en étant rapides.

De plus, SymAMD est plus rapide qu'AMD, probablement parce qu'elle est plus efficace sur les matrices symétriques. ColAMD est également performant, offrant un bon équilibre entre vitesse et utilisation de la mémoire, mais il a un fill in plus important (environ 5000).

Enfin, SymRCM améliore les performances par rapport à l'absence de réordonnancement, mais reste moins efficace que AMD et SymAMD. En effet, son coût mémoire (169KB) et ses FLOPs sont bien plus élévés que les autres méthodes. De plus, les backward et forward errors sont très faibles, ce qui signifie que le réordonnancement n'a pas d'impact négatif sur la précision des calculs.

Paramètre	**Sans perm.**	**AMD**	**SymAMD**	**SymRCM**	**ColAMD**
Temps (s)	0.0020	0.0006	0.0004	0.0004	0.0004
FLOPs	1.42×10^5	2.81×10^4	2.82×10^4	4.22×10^4	3.51×10^4
Fill-in	31753	3197	3219	6727	4957
NNZ de L	35582	7026	7048	10556	8786
Mémoire (KB)	6.36×10^2	114.27	114.61	169.42	141.77
Backward error	7.51×10^{-12}	5.55×10^{-12}	7.75×10^{-12}	8.76×10^{-12}	8.09×10^{-12}
Forward error	4.49×10^{-13}	7.47×10^{-16}	1.30×10^{-13}	2.71×10^{-13}	1.54×10^{-13}
Gain FLOPs (%)	0.00%	80.25%	80.19%	70.33%	75.31%

Table 2: Comparaison des méthodes de réordonnancement pour la factorisation de Cholesky

4.4 Résultats pour mat2

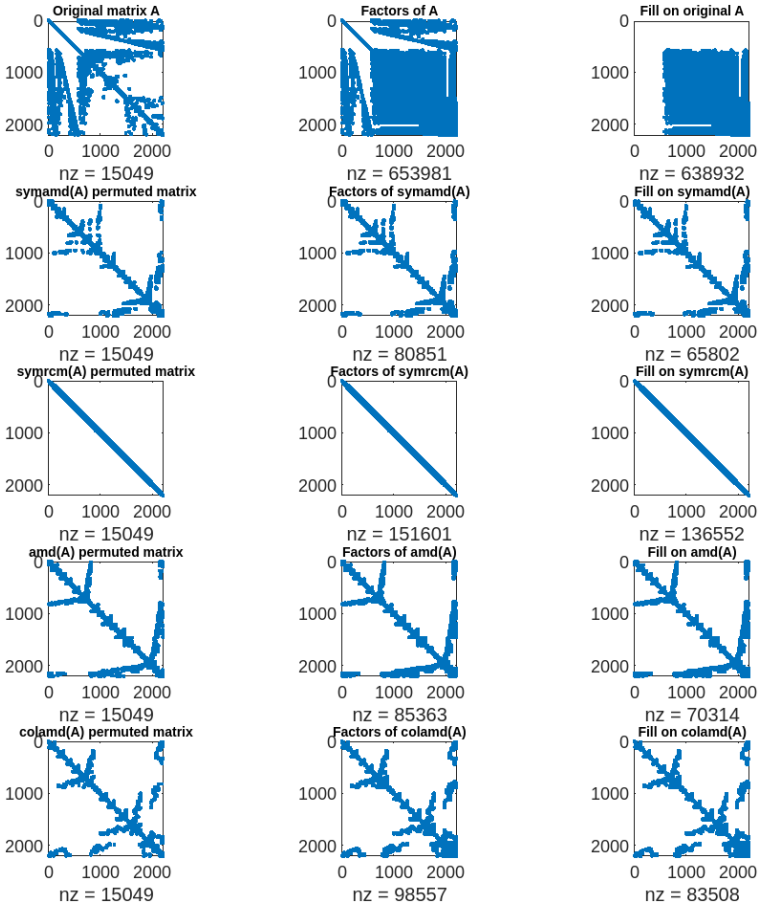

Figure 3: Visualisation des matrices et du fill-in après réordonnancement pour *mat*2

Paramètre	Sans perm.	AMD	SymAMD	SymRCM	ColAMD
Temps (s)	0.0087	0.0028	0.0016	0.0022	0.0019
FLOPs	1.31×10^6	1.75×10^5	1.66×10^5	3.08×10^5	2.02×10^5
Fill-in	312844	28733	26477	61852	35330
NNZ de L	327893	43782	41526	76901	50379
Mémoire (KB)	5.52×10^3	701.30	666.05	1218.78	804.38
Backward error	3.01×10^{-11}	2.72×10^{-11}	3.10×10^{-11}	4.24×10^{-11}	3.09×10^{-11}
Forward error	5.84×10^{-13}	6.44×10^{-14}	7.61×10^{-13}	1.14×10^{-12}	1.05×10^{-12}
Gain FLOPs (%)	0.00%	86.65%	87.34%	76.55%	84.64%

Table 3: Comparaison des méthodes de réordonnancement pour la factorisation de Cholesky

Interprétation : Pour cette matrice également, SymAMD et AMD sont les stratégies les plus performantes, offrant une réduction des FLOPs (86%), du fill-in et de la mémoire utilisée, tout en maintenant une grande précision. On remarque tout de même que SymAMD est plus performante que AMD.

D'autre part, ColAMD semble également être un bon choix car il a une bonne efficacité mémoire avec une rapidité similaire à SymAMD. SymRCM reste efficace (par rapport à sans permutation) mais génère plus de fill-in et consomme plus de mémoire que les autres méthodes.

4.5 Résultats pour mat3

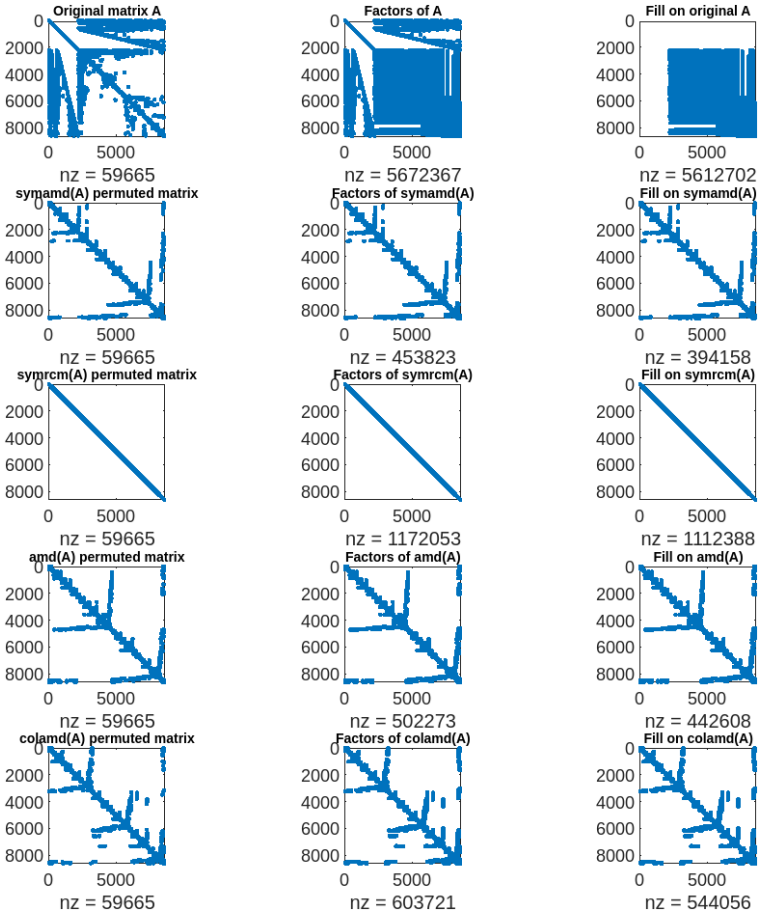

Figure 4: Visualisation des matrices et du fill-in après réordonnancement pour *mat*3

Paramètre	Sans perm.	AMD	SymAMD	SymRCM	ColAMD
Temps (s)	0.1099	0.0114	0.0103	0.0145	0.0151
FLOPs	1.13×10^7	1.02×10^6	9.25×10^5	2.36×10^6	1.22×10^6
Fill-in	2754595	195784	171559	530674	246508
NNZ de L	2814260	255449	231224	590339	306173
Mémoire (KB)	4.72×10^4	5529.39	4920.83	10404.44	6059.61
Backward error	1.29×10^{-10}	1.17×10^{-10}	1.09×10^{-10}	1.72×10^{-10}	1.28×10^{-10}
Forward error	1.58×10^{-12}	1.43×10^{-13}	1.52×10^{-12}	3.31×10^{-12}	2.44×10^{-12}
Gain FLOPs (%)	0.00%	90.92%	91.78%	79.02%	89.12%

Table 4: Comparaison des méthodes de réordonnancement pour la factorisation de Cholesky

Interprétation : Pour mat3, SymAMD et AMD sont les stratégies les plus performantes, avec SymAMD légèrement plus efficace qu'AMD. ColAMD est également un bon choix, offrant une bonne efficacité mémoire mais c'est la permutation la moins rapide (0.0151s).

De plus, SymRCM reste efficace comparé à l'absence de permutation, mais génère bien plus de fill-in et consomme plus de mémoire que les autres méthodes, ce qui réduit son efficacité globale. On peut noter que les temps de calculs pour cette matrice sont plus élévés que pour les précédentes dû à sa taille plus grande (8625*8625).

4.6 Résultats pour bcsstk27

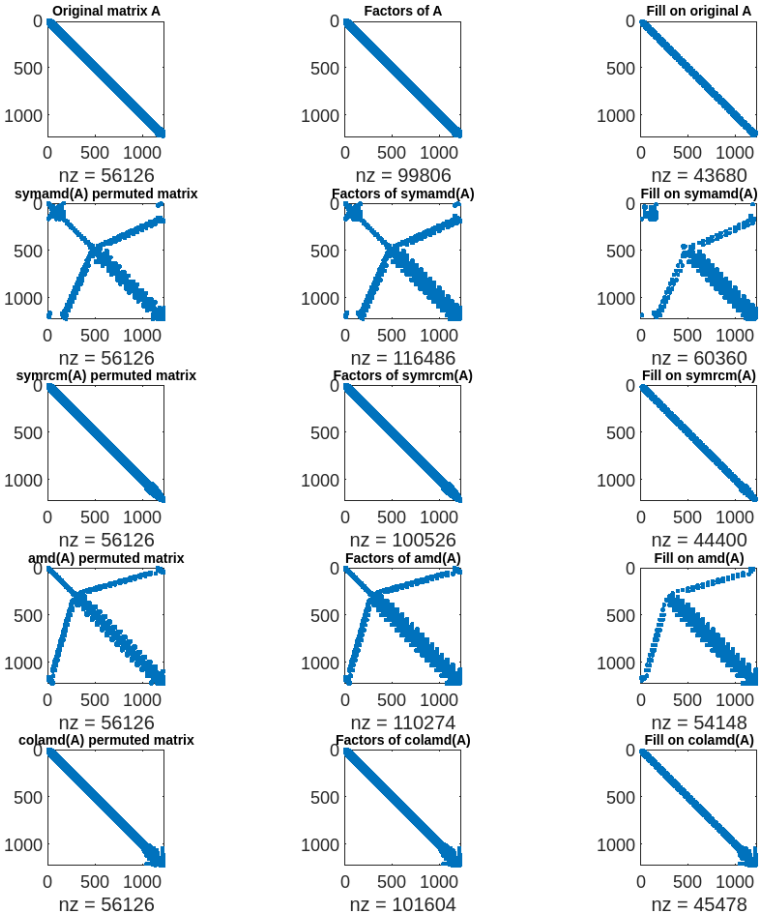

Figure 5: Visualisation des matrices et du fill-in après réordonnancement pour *bcsstk*27

Paramètre	**Sans perm.**	**AMD**	**SymAMD**	**SymRCM**	**ColAMD**
Temps (s)	0.0012	0.0014	0.0013	0.0010	0.0015
FLOPs	2.02×10^5	2.23×10^5	2.35×10^5	2.04×10^5	2.06×10^5
Fill-in	-5611	-377	2729	-5251	-4712
NNZ de L	50515	55749	58855	50875	51414
Mémoire (KB)	8.74×10^2	972.13	995.84	872.51	885.30
Backward error	4.94×10^{-14}	4.08×10^{-14}	4.11×10^{-14}	4.80×10^{-14}	3.50×10^{-14}
Forward error	1.14×10^{-14}	1.94×10^{-15}	8.11×10^{-15}	5.70×10^{-15}	8.29×10^{-15}
Gain FLOPs (%)	0.00%	-10.36%	-16.51%	-0.71%	-1.78%

Table 5: Comparaison des méthodes de réordonnancement pour la factorisation de Cholesky

Interprétation : Contrairement aux matrices précédentes, le réordonnancement de la matrice *bcsstk27* n'améliore pas le coût de calcul et peut même l'augmenter légèrement (gain FLOPs négatif).

De plus, le fill-in négatif montre que la permutation utilisée réduit le nombre de termes non nuls après les calculs. Cela ne pose pas de problème pour la mémoire utilisée ou la précision des résultats.

Cela suggère que la matrice initiale A soit déjà bien organisée. De fait, certaines façons de la réarranger peuvent donc être un peu moins efficaces. Néanmoins, les erreurs restent très petites ce qui suggère que les solutions sont correctes pour toutes les permutations.

5 Conclusion

Pour conclure, grâce à ce TP, on a pu évaluer l'impact des différentes stratégies de permutation sur la factorisation de matrices creuses. Les résultats montrent clairement que les méthodes AMD et SymAMD sont les plus performantes, offrant une réduction significative des opérations, du remplissage et de l'utilisation de la mémoire, tout en maintenant une grande précision.

On a également constaté que la méthode SymRCM, contrairement aux autres stratégies de réordonnancement, réduit la taille de la bande de la matrice en rapprochant les valeurs non nulles de la diagonale principale.