Solveurs multifrontaux exploitant des blocs de rang faible : complexité, performance et parallélisme

Théo Mary [1]

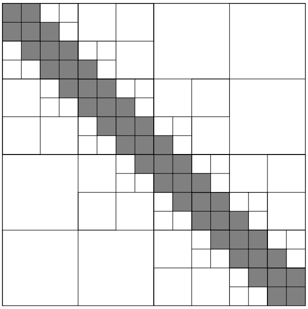
Théo Mary a soutenu sa thèse [2] en novembre 2017 à l'université de Toulouse, thèse préparée à l'Institut de recherche en informatique de Toulouse (IRIT), sous la direction de Patrick Amestoy et d'Alfredo Buttari. Il effectue actuellement un stage postdoctoral à l'université de Manchester (Royaume-Uni).

Mes travaux de thèse ont considéré la résolution de systèmes d'équations linéaires $Ax = b$, où A est une matrice creuse de très grande taille. Ceci constitue un problème important et récurrent en calcul scientifique, par exemple en mécanique des structures ou en géophysique appliquée. Deux classes de méthodes classiques existent pour résoudre de tels systèmes :

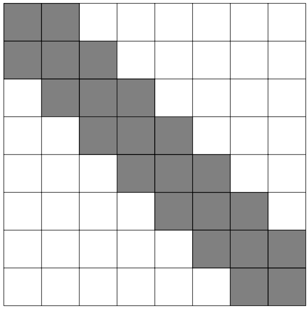
— Les méthodes *directes* calculent une factorisation de la matrice A, notamment fondée sur l'élimination de Gauss (décomposition LU), permettant ensuite d'obtenir directement la solution $x = U^{-1}L^{-1}b$ par résolutions triangulaires. Elles sont appréciées pour leur fiabilité numérique, mais possèdent un coût calculatoire important, ce qui limite la *taille* des problèmes traitables.

1. theo.mary@manchester.ac.uk

2. « Solveurs multifrontaux exploitant des blocs de rang faible : complexité, performance et parallélisme », https://tel.archives-ouvertes.fr/tel-01929478/document.



(a) Matrice $\mathscr{H}$.



(b) Matrice BLR.

FIGURE 1. Illustration des formats matriciels de rang faible $\mathscr{H}$ et BLR. Les blocs en gris sont stockés de manière exacte tandis que ceux en blanc sont approchés par des matrices de rang faible.

— Les méthodes *itératives* calculent une suite d'itérés x_k convergeant vers la solution x. Elles sont en général très économiques en coût calculatoire, mais leur fiabilité dépend fortement de la possibilité de trouver un préconditionneur qui garantit leur convergence, ce qui limite la *gamme* des problèmes traitables.

Mes travaux de thèse se sont intéressés à la conception, au développement et à l'analyse de méthodes nouvelles exploitant une perte de précision *contrôlée* pour trouver un compromis entre coût calculatoire et précision numérique, de manière à pouvoir résoudre *une large gamme de problèmes de très grande taille*. Concrètement, cette perte de précision est obtenue en exploitant des *approximations de rang faible*, qui consistent à représenter une matrice B de dimensions $m \times m$ par un produit XY^T de deux matrices X et Y de dimensions $m \times r(\varepsilon)$, où $r(\varepsilon)$ est le rang numérique de B à la précision ε souhaitée (c'est-à-dire que $\|B - XY^T\| \leq \varepsilon\|B\|$).

En effet, dans de nombreuses applications nécessitant la résolution d'un système $Ax = b$, notamment la résolution d'équations aux dérivées partielles, la matrice A et ses facteurs LU possèdent un grand nombre de blocs de rang faible. Plusieurs *formats* matriciels ont été proposés dans la littérature pour exploiter cette propriété. Ces formats diffèrent en particulier dans leur manière de partitionner la matrice en blocs (qui sont ensuite approchés par des matrices de rang faible).

Les formats *hiérarchiques*, notés $\mathscr{H}$, sont très étudiés en raison de leur faible complexité asymptotique [5]. Les matrices $\mathscr{H}$ sont définies par un partitionnement en blocs hiérarchique (voir Figure 1(a)), ce qui permet de maximiser le taux de compression obtenu grâce aux approximations de rang faible. En effet, la factorisation d'une matrice $\mathscr{H}$ dense de dimensions $n \times n$ peut avoir des coûts considérablement réduits par rapport aux coûts classiques de stockage, en $O(n^{4/3})$, et calculatoire, en

TABLE 1. Complexités asymptotiques de la factorisation d'une matrice A de dimensions $n \times n$ creuse (issue d'un problème discrétisé sur un domaine 3D régulier), selon que la matrice A est représentée par le format $\mathscr{H}$ ou BLR.

	Référence	BLR	$\mathscr{H}$
Coût de stockage	$O(n^{4/3})$	$O(n \log n)$	$O(n)$
Coût calculatoire	$O(n^2)$	$O(n^{4/3})$	$O(n)$

$O(n^2)$. Sous l'hypothèse que les rangs des blocs sont suffisamment faibles, ces coûts peuvent être tout les deux réduits jusqu'à une complexité linéaire $O(n)$. En revanche, leur partitionnement hiérarchique rend l'utilisation des matrices $\mathscr{H}$ très complexe dans un contexte parallèle, comme expliqué dans le paragraphe suivant.

D'un autre côté, le format *Block Low-Rank* (BLR) est fondé sur un partitionnement en blocs non hiérarchique (voir Figure 1(b)), ce qui le rend extrêmement simple à utiliser et offre un grand potentiel pour exploiter le parallélisme [4]. En effet, dans le format BLR tous les blocs sont indépendants, ce qui permet de paralléliser pleinement des opérations portant sur des blocs différents, alors que dans le format $\mathscr{H}$ le parallélisme est contraint par les dépendances imposées par le partitionnement hiérarchique. Une deuxième difficulté du format $\mathscr{H}$, spécifique au cas des architectures à mémoire distribuée, est que de nombreux blocs sont de dimensions très grandes (du même ordre de grandeur que celles de la matrice) et doivent donc être distribués sur plusieurs processeurs ; ceci n'est le plus souvent pas nécessaire en BLR, où les blocs sont beaucoup plus petits.

Malgré ces avantages, le format BLR a été longtemps ignoré car sa complexité théorique était inconnue. La communauté travaillant sur les formats hiérarchiques avait même conjecturé que le format BLR ne conduisait à aucun gain asymptotique par rapport à un format de rang plein standard. Un des premiers objectifs de ma thèse a été de répondre à ce problème. Nous avons montré que la théorie hiérarchique appliquée au format BLR ne permettait en effet pas de prouver de résultat satisfaisant, expliquant donc la conjecture pessimiste de la communauté. Cependant, nous avons ensuite étendu la théorie pour montrer que le format BLR obtient bien un gain asymptotique : la factorisation d'une matrice BLR dense de dimensions $n \times n$ peut atteindre, pour des rangs suffisamment faibles, un coût de stockage en $O(n \log n)$ et un coût calculatoire en $O(n^{4/3})$ [2]. Le format BLR représente donc un compromis entre la complexité asymptotique et le potentiel pour le parallélisme.

Après avoir prouvé que le format BLR parvient à réduire la complexité théorique de la factorisation de matrices, nous nous sommes tournés dans la seconde partie de ma thèse vers le problème de la conversion de cette réduction théorique en gains de performance réels sur les architectures parallèles modernes. Ceci s'avère être un problème difficile. Par exemple, sur une matrice creuse d'ordre supérieur à 20 millions issue d'un problème électromagnétique, le format BLR permet de réduire le nombre d'opérations de la factorisation d'un facteur 19.1 ; cependant, avec une première implémentation qui était standard au moment de ces travaux, ce potentiel ne se traduit que par un gain réel d'un facteur 3.9 sur une machine à 24 cœurs.

Notre analyse de performance démontre que l'introduction du format BLR dans un logiciel de haute performance pose en effet plusieurs défis, que nous avons relevés un par un en introduisant plusieurs améliorations algorithmiques [3].

— Tout d'abord, le format BLR modifie le coût relatif de chaque partie : les parties bénéficiant de son utilisation deviennent négligeables alors que celles n'en bénéficiant pas deviennent le goulet d'étranglement. Il est alors nécessaire de changer de modèle de parallélisme pour mieux paralléliser les parties du code ne bénéficiant pas du format BLR.

— Ensuite, la présence de blocs de rangs très différents change complètement le modèle d'accès à la mémoire. Nous avons montré qu'il devient critique d'effectuer les calculs dans un ordre bien précis (*left-looking* au lieu de *right-looking*) pour favoriser l'accès à des blocs de rang faible plutôt que plein et ainsi diminuer le volume total d'accès à la mémoire.

— Enfin, la factorisation d'une matrice BLR implique de nombreux calculs effectués sur des matrices de rang faible qui sont de dimensions trop petites pour permettre l'utilisation efficace des noyaux de calculs de type matrice-matrice (BLAS-3). Nous avons conçu un algorithme qui accumule des matrices de rang faible de manière à augmenter la granularité des calculs et donc leur efficacité.

Grâce à ces améliorations, le gain en temps obtenu par le format BLR augmente significativement. En revenant à l'exemple mentionné ci-dessus, le gain augmente ainsi de 3.9 à 18.8.

Ces défis sont d'autant plus importants que le nombre de cœurs utilisés est grand. Dans un contexte à mémoire distribuée avec plusieurs centaines voire milliers de cœurs, nous avons montré qu'il devient critique d'approcher les blocs par des matrices de rang faible *avant* de les transférer d'un processeur à un autre : en effet, alors que cette stratégie demande un surcroît d'opérations, elle peut s'avérer plus rapide car elle mène à un volume de communications bien plus faible. Ainsi, nous avons réussi à résoudre un problème d'imagerie sismique à haute fréquence (20Hz) qui mène à un système linéaire à 130 millions d'inconnues sur un supercalculateur disposant de 2400 cœurs. Ce problème aurait requis 150 PetaFlops et 11 TeraBytes de mémoire avec un solveur standard et n'était donc pas traitable. Grâce au format BLR, nous avons réduit l'empreinte mémoire du problème à 1.8 TeraBytes, ce qui nous a permis de le résoudre en moins d'une heure. Tout ceci étant possible sans impacter la précision de la méthode applicative.

Tous les algorithmes décrits ci-dessus ont été intégrés dans le logiciel MUMPS [3]. MUMPS est un solveur direct pour systèmes linéaires creux spécialement conçu pour les architectures à mémoire distribuée, disposant de nombreuses fonctionnalités assurant sa robustesse et sa performance. C'est également un logiciel gratuit, ouvert et distribué publiquement qui dispose de dizaines de milliers d'utilisateurs à travers le monde. De ce fait, l'impact de mes travaux de thèse s'est déjà ressenti dans plusieurs applications académiques et industrielles. Cela concerne par exemple deux applications en exploration géophysique [1, 6], ainsi que plusieurs entreprises dont EDF, Total, Airbus, Michelin, Shell, LSTC, ESI, Altair et FFT, qui utilisent tous la fonctionnalité BLR dans leurs codes de simulation numérique, notamment de mécanique des structures.

Références

[1] P. R. Amestoy, R. Brossier, A. Buttari, J.-Y. L'Excellent, T. Mary, L. Métivier, A. Miniussi, and S. Operto. Fast 3D frequency-domain full waveform inversion with a parallel Block Low-Rank multifrontal direct solver : application to OBC data from the North Sea. *Geophysics*, 81(6) :R363–R383, 2016.

[2] P. R. Amestoy, A. Buttari, J.-Y. L'Excellent, and T. Mary. On the Complexity of the Block Low-Rank Multifrontal Factorization. *SIAM J. Sci. Comput.*, 39(4) :A1710–A1740, 2017.

[3] P. R. Amestoy, A. Buttari, J.-Y. L'Excellent, and T. Mary. Performance and Scalability of the Block Low-Rank Multifrontal Factorization on Multicore Architectures. *ACM Trans. Math. Software*, 2018. To appear.

[4] Patrick Amestoy, Cleve Ashcraft, Olivier Boiteau, Alfredo Buttari, Jean-Yves L'Excellent, and Clément Weisbecker. Improving multifrontal methods by means of block low-rank representations. *SIAM J. Sci. Comput.*, 37(3) :A1451–A1474, 2015.

[5] Wolfgang Hackbusch. *Hierarchical Matrices : Algorithms and Analysis*, volume 49 of *Springer Series in Computational Mathematics*. Springer, Berlin, 2015.

[6] D. V. Shantsev, P. Jaysaval, S. de la Kethulle de Ryhove, P. R. Amestoy, A. Buttari, J.-Y. L'Excellent, and T. Mary. Large-scale 3D EM modeling with a Block Low-Rank multifrontal direct solver. *Geophys. J. Int.*, 209(3) :1558–1571, 2017.

3. `http://mumps-solver.org/`