Sujet de thèse

Précisions mixtes pour les méthodes itératives préconditionnées appliquées aux systèmes linéaires creux

Mots clés : calcul haute performance, algorithmique numérique, arithmétique flottante, algèbre linéaire

Contexte

Cette thèse va se dérouler dans le cadre d'une collaboration entre IFPEN (IFP Energies Nouvelles) et le laboratoire LIP6 à Sorbonne Université. Elle bénéficiera de l'encadrement suivant.

A IFPEN : Ani Anciaux-Sedrakian (ani.anciaux-sedrakian@ifpen.fr) et Thomas Guignon (thomas.guignon@ifpen.fr)
Au LIP6 : Fabienne Jézéquel (fabienne.jezequel@lip6.fr) et Théo Mary (theo.mary@lip6.fr)

A IFPEN, la simulation numérique est un outil stratégique puisqu'indispensable à la recherche et à de nombreuses applications qui requièrent la modélisation d'écoulements multiphasiques. La performance de ces simulateurs est donc un enjeu majeur. Elle a notamment un impact direct, à la fois sur la qualité des résultats, et sur la capacité à réaliser des calculs à grande échelle.

Avec l'arrivée de l'intelligence artificielle (IA), et en réponse à la demande de cette communauté de disposer d'une capacité de calcul de plus en plus élevée, les instructions de simple précision (32 bits) et de demi-précision (16 bits) ont émergé dans les unités de calculs modernes (CPU avec les instructions SIMD, GPU, TPU, etc.). En effet, la plupart des applications en lien avec l'IA n'exigent pas nécessairement la précision double et utilisent les précisions simples ou demies. Afin de pouvoir efficacement utiliser ces ressources de calcul dans les simulations numériques, il est nécessaire d'évaluer dans les algorithmes de résolution de systèmes linéaires l'influence de la dégradation de la précision et la robustesse des résultats.

Depuis une dizaine d'années, plusieurs études ont été effectuées dans cette direction. Les travaux de recherche sur **l'algèbre linéaire dense**, les **produits matriciels** et la résolution **directe** des systèmes linéaires ont montré des gains de performance très intéressants en utilisant les « Tensor Cores » disponibles sur les GPUs [AAB+20, FHM+21, ZHL+22, HM22, ABH+22]. En revanche, les travaux de recherche concernant la résolution de systèmes **creux** par des méthodes **itératives**, utilisées majoritairement dans les applications à IFPEN, sont très limités pour le moment.

Objectif(s) de la thèse

L'objectif du travail de recherche proposé est d'introduire une nouvelle approche pour exploiter les arithmétiques de précision mixte (demi, simple et double) pour les algorithmes de solveur linéaire **itératif préconditionné** fondé sur les sous-espaces de **Krylov** pour les systèmes **creux**. Plus précisément pour le problème suivant :

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

nous proposons de concevoir des algorithmes à la fois pour la construction du préconditionneur « M^{-1} » (comme SPAI, LU incomplet) et la résolution **itérative** du système linéaire **creux** (par BiCGStab par exemple), en bornant la propagation d'erreur liée aux calculs moins précis.

Les méthodes itératives fondées sur les sous-espaces de Krylov ont une complexité opérationnelle et une consommation de mémoire élevées. Par conséquent, elles ont une **extensibilité parallèle limitée** sur les supercalculateurs. L'intérêt de l'utilisation des formats flottants en **précision simple ou demi** (FP32 ou FP16) par rapport à la double précision (FP64) est donc triple, puisque la solution basée sur l'utilisation de la précision mixte nous permettra de **diminuer les entrées/sortie**s mémoire, d'**augmenter la capacité de calcul** et par conséquent de **baisser la consommation énergétique**.

L'utilisation d'un solveur préconditionné itératif en précision réduite (32 ou 16 bits) au lieu de la double précision (64 bits) nécessite cependant un examen attentif de la **propagation d'erreur** et des aspects numériques. En effet, l'utilisation à l'aveugle de la précision réduite dans un solveur itératif dégrade dans la plupart des cas, soit le niveau de précision atteignable, soit la vitesse de convergence, voire les deux à la fois.

État de l'art

Contrairement à l'intelligence artificielle, en calcul scientifique afin de garantir la robustesse des résultats de calcul, ce sont les instructions en double et en quadruple précision qui sont majoritairement utilisées. Par ailleurs les calculs en simple précision présentent des avantages par rapport aux calculs en double précision, et pas seulement parce que l'arithmétique en simple précision est généralement deux fois plus rapide que l'arithmétique en double précision, mais aussi parce que les données en simple précision nécessitent deux fois moins de stockage que les données en double précision et que les coûts de transfert de mémoire sont deux fois moins élevés. En revanche, l'erreur d'arrondi générée par chaque opération flottante est plus élevée en simple précision qu'en double, ce qui entraîne une propagation d'erreur plus importante. C'est notamment le cas lors de la résolution de systèmes linéaires mal conditionnés.

Il est donc crucial de pouvoir évaluer les erreurs d'arrondi, en particulier celles commises en précision réduite. Il existe différentes méthodes pour évaluer/analyser les erreurs d'arrondis comme l'arithmétique par intervalles [AH83], l'analyse statique [PGM03] et l'arithmétique stochastique [Vig93]. Fondée sur une approche probabiliste, l'arithmétique stochastique permet d'estimer les erreurs d'arrondi grâce à plusieurs exécutions des opérations arithmétiques avec un mode d'arrondi aléatoire. L'arithmétique stochastique peut être utilisée avec un large éventail de méthodes numériques, puisqu'elle ne nécessite pas de modification des algorithmes dans les codes à contrôler. L'arithmétique stochastique est implantée dans la bibliothèque CADNA [JC08, EBF+15, JHH21] qui permet d'estimer le nombre de chiffres corrects des résultats, notamment dans les programmes en précision mixte. Dans cette thèse, la bibliothèque CADNA pourra être utilisée pour estimer la fiabilité numérique des résultats à la fois dans la phase de préconditionnement et lors de la résolution itérative des systèmes linéaires. Une fois la propagation d'erreur maîtrisée, nous pourrons aborder les travaux concernant la conception d'algorithmes utilisant les précisions mixtes, voire faibles, afin de gagner en performance. Différentes approches permettant de développer des codes de calcul en précision mixte peuvent être distinguées.

D'abord, les approches de raffinement consistent à calculer une première solution à précision faible puis en améliorer la qualité grâce à des corrections calculées en précision plus élevée. Ces approches ont rencontré un franc succès dans les années 2000 [LLL+06, BDK+08] et ont fait l'objet d'avancées théoriques récentes motivées par l'apparition de la demie précision [CH17, CH18, ABH+22a]. Ces approches de raffinement modernes ont permis d'accélérer considérablement les méthodes directes, que ce soit pour les systèmes denses [HTD+18] ou, très récemment, creux [ABH+22b]. Leur utilisation au sein de solveurs itératifs reste en revanche ouverte, les dernières avancées dans ce domaine datant des années 2000 [BDK+08] voire 1990 [TW92].

Ensuite, les approches adaptatives s'adaptent aux données spécifiques du problème en réduisant la précision des données les moins sensibles numériquement. Par exemple, Anzt et al. [ADF+19] proposent un préconditionneur bloc Jacobi en sauvegardant les blocs dans les formats 16, 32 ou 64 bits selon leur conditionnement (les mieux conditionnées peuvent être passés en précision faible). Dans le même ordre d'idée, des approches adaptatives pour le calcul d'approximations de rang faible [ABB+22] et pour le produit matrice creuse-vecteur [GJM+22] ont été récemment proposées.

Enfin, les approches automatiques choisissent automatiquement la précision de chaque partie ou variable d'un code, quels que soient les algorithmes qui y sont implantés. Au cours de ces dernières années, plusieurs algorithmes et outils ont été proposés pour l'auto-tuning de précision. D'une part, des outils tels que FPTuner [CBB+17], Daisy [DIN+18], TAFFO [CCC+20], POP [AKM21] reposent sur une approche statique et ne sont pas destinés à être utilisés sur un code très volumineux. D'autre part, les outils dynamiques tels que Precimonious [RNN+13], HiFPTuner [GR18], FloatSmith [LVM+19], PROMISE [GJP+19, JHH21] visent à fournir une version en précision mixte de grands codes HPC. De plus, des outils ont été récemment développés pour l'auto-tuning de précision sur les GPU : AMPT-GA [KSM+19], GPUMixer [LWR+19], GRAM [HSW21]. À partir d'un code en C/C++ et d'une précision requise sur les résultats, PROMISE génère automatiquement une version du code en précision mixte. Une spécificité du logiciel PROMISE réside dans le fait qu'il fournit des programmes en précision mixte validés grâce à l'arithmétique stochastique, alors que d'autres outils dynamiques s'appuient sur un résultat de référence pouvant être affecté par des erreurs d'arrondi.

Stratégie de recherche envisagée

Nous chercherons à introduire de la précision faible et/ou mixte dans les solveurs utilisés par IFPEN, à savoir actuellement la méthode BiCGStab avec préconditionneur LU incomplet (ILU). Notre méthodologie pour ce faire consistera à combiner diverses approches (raffinement, adaptatives, etc.) à divers niveaux du calcul (notamment préconditionneur et produit matrice creuse-vecteur), tout en contrôlant rigoureusement l'erreur ainsi engendrée.

Le coût de résolution total est essentiellement donné par le coût de construction du préconditionneur plus le coût de la partie itérative (lui-même déterminé par le nombre d'itérations et le coût de chaque itération). Nous avons donc plusieurs leviers pour accélérer la résolution : réduire le coût de construction du préconditionneur, réduire le nombre d'itérations ou réduire le coût par itération.

Concernant la partie itérative, le moyen le plus immédiat de réduire son coût par itération est de passer tous les calculs en précision faible ; naturellement, cela ne permet à la méthode que d'atteindre une solution de précision faible. Pour surmonter cet obstacle, nous mettrons en œuvre les méthodes de raffinement modernes dans ce contexte nouveau de solveur itératif BiCGStab. L'idée est d'effectuer quelques appels à un BiCGStab entièrement en précision faible pour corriger la solution, ce qui devrait normalement permettre de converger à des précisions élevées tout en effectuant la plupart des itérations en précision faible. Cependant, l'applicabilité de cette technique reste une question ouverte : notamment, la convergence pourrait être ralentie, voire rendue impossible pour les problèmes les plus mal conditionnés. Dans ce cas, une possibilité que nous prévoyons sera de garder les itérations en précision plus haute et chercher à accélérer le produit matrice creuse-vecteur, qui est le plus consommateur en temps de calcul, par l'intermédiaire d'approches adaptatives notamment. Enfin, une dernière piste plus à long terme concernant la partie itérative serait de comparer BiCGStab à d'autres solveurs itératifs, comme GMRES ou sa version flexible (FMGRES, qui a l'avantage de pouvoir traiter des préconditionneurs non constants comme c'est le cas avec raffinement). Ces alternatives sont parfois plus robustes, mais aussi plus coûteuses en temps et en empreinte mémoire (base de Krylov à stocker et à maintenir orthonormale) ; dans ce contexte, la précision réduite pourrait être exploitée pour diminuer ce surcoût lié à ces méthodes, obtenant ainsi un compromis entre BiCGStab et FGMRES.

Concernant la partie préconditionneur, nous commencerons dans l'immédiat par comparer les combinaisons possibles concernant la précision pour sa construction (avant les itérations) et son application (durant les itérations, par exemple par résolution triangulaire dans le cas d'un LU incomplet). En effet, si le point de départ est d'utiliser la même précision pour la construction et l'application, il pourrait être intéressant de les dissocier. Par exemple, construire le préconditionneur en précision élevée, puis ensuite le sauvegarder en précision faible (pour réduire son coût d'application) aboutit souvent à un préconditionneur de meilleure qualité que s'il avait été directement construit en précision faible. A l'inverse, dans certains contextes il peut être utile d'appliquer le préconditionneur en précision plus élevée que celle dans laquelle il a été construit, notamment pour absorber les erreurs d'arrondi engendrées par son application [CH18, ABH+22a]. Par la suite, nous chercherons à introduire de la précision mixte à l'intérieur même du préconditionneur. Notamment, nous prévoyons de développer une version adaptative de la factorisation LU incomplète. Dans ce cadre, la question est comment choisir la précision de chaque élément des facteurs LU ; une métrique naturelle est d'utiliser leur amplitude, généralisant ainsi la factorisation ILUT (ILU à seuil, qui ne conserve que les éléments plus grands en valeur absolue qu'un certain seuil). D'autres métriques sont envisageables, liées par exemple au niveau de remplissage (généralisation des factorisations ILU(k)). Les travaux de Hook et Tisseur [HT17] visant à prédire l'ordre de grandeur des éléments des facteurs LU à l'avance pourraient être utiles dans ce contexte.

Tout au long de cette thèse, des analyses d'erreurs d'arrondi seront réalisées pour valider les algorithmes proposés. Il s'agira à la fois d'analyses a priori visant à obtenir des bornes décrivant le comportement des algorithmes d'un point de vue théorique, mais aussi leur validation pratique par des outils d'analyse d'erreur fondés sur l'arithmétique stochastique comme CADNA. La mise en œuvre de tels outils nous permettra aussi de tester les approches automatiques pour la précision mixte (via PROMISE par exemple), afin de valider le bien fondé des algorithmes proposés et de vérifier si d'autres pistes peuvent être explorées.

En conclusion, cette thèse demandera donc des recherches fondamentales à la fois sur les plans numériques (analyse d'erreur, …), algorithmiques et informatiques (mise en œuvre sur calculateurs modernes et passage à l'échelle sur problèmes de très grande taille).

Références bibliographiques

[AAB+20] A. Abdelfattah et al.; A Survey of Numerical Methods Utilizing Mixed Precision Arithmetic, https://doi.org/10.48550/arXiv.2007.06674, 2020.

[ABB+22] P. Amestoy, O. Boiteau, A. Buttari, M. Gerest, F. Jézéquel, J.-Y. l'Excellent, T. Mary, Mixed Precision Low Rank Approximations and their Application to Block Low Rank LU Factorization, IMA Journal of Numerical Analysis, https://hal.archives-ouvertes.fr/hal-03251738, 2022.

[ABH+22a] P. Amestoy, A. Buttari, N. J. Higham, J.-Y. L'Excellent, T. Mary & B. Vieublé. Five-precision GMRES-based iterative refinement, https://hal.archives-ouvertes.fr/hal-03190686, 2022.

[ABH+22b] P. Amestoy, A. Buttari, N. J. Higham, J.-Y. L'Excellent, T. Mary & B. Vieublé. Combining sparse approximate factorizations with mixed precision iterative refinement, https://hal.archives-ouvertes.fr/hal-03536031, 2022.

[ADF+19] H. Anzt, J. Dongarra, G. Flegar, N. J. Higham, E. S. Quintana-Ortí; Adaptive precision in block-Jacobi preconditioning for iterative sparse linear system solvers. Concurrency Computat Pract Exper. https://doi.org/10.1002/cpe.4460, 2019.

[AH83] G. Alefeld and J. Herzberger. Introduction to interval analysis. Academic Press, 1983.

[AKM21] A. Adjé, D. B. Khalifa, and M. Martel, Fast and Efficient Bit-Level Precision Tuning, International Static Analysis Symposium, 1-24, arXiv:2103.05241, 2021.

[BDK+08] A. Buttari, J. Dongarra, J. Kurzak, P. Luszczek, S. Tomov, Using mixed precision for sparse matrix computations to enhance the performance while achieving 64-bit accuracy, ACM TOMS 2008.

[CBB+17] W.-F. Chiang, M. Baranowski, I. Briggs, A. Solovyev, G. Gopalakrishnan, and Z. Rakamarić, Rigorous Floating-Point Mixed-Precision Tuning, in Proceedings of the 44th ACM SIGPLAN Symposium on Principles of Programming Languages, POPL 2017, New York, NY, USA, ACM, 300-315, 2017.

[CCC+20] S. Cherubin,D. Cattaneo, M. Chiari, A. D. Bello, and G. Agosta, TAFFO: Tuning Assistant for Floating to Fixed Point Optimization, IEEE Embedded Systems Letters, 12, 5-8, 2020.

[CH17] E. Carson and N. J. Higham, A new analysis of iterative refinement and its application to accurate solution of ill-conditioned sparse linear systems, SIAM SISC 2017.

[CH18] E. Carson and N. J. Higham, Accelerating the solution of linear systems by iterative refinement in three precisions, SIAM SISC 2018.

[DIN+18] E. Darulova, A. Izycheva, F. Nasir, F. Ritter, H. Becker, and R. Bastian, Daisy - Framework for Analysis and Optimization of Numerical Programs (Tool Paper), in 24th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS), vol. 10805 LNCS, Thessaloniki, Greece, 270-287, 2018.

[EBF+15] P. Eberhart, J. Brajard, P. Fortin, and F. Jézéquel, High Performance Numerical Validation using Stochastic Arithmetic, Reliable Computing, 21, pages 35-52, 2015.

[FHM+21] M. Fasi, N. J. Higham, M. Mikaitis, and S. Pranesh; Numerical behavior of NVIDIA tensor cores, PeerJ Computer Science, https://doi.org/10.7717/peerj-cs.330, 2021.

[GJM+22] S. Graillat, F. Jézéquel, T. Mary & R. Molina, Adaptive precision sparse matrix-vector product and its application to Krylov solvers, https://hal.archives-ouvertes.fr/hal-03561193, 2022.

[GJP+19] S. Graillat, F. Jézéquel, R. Picot, F. Févotte & B. Lathuilière, Auto-tuning for floating-point precision with Discrete Stochastic Arithmetic, https://doi.org/10.1016/j.jocs.2019.07.004, 2019.

[GR18] H. Guo and C. Rubio-González, Exploiting Community Structure for Floating-Point Precision Tuning, in Proceedings of the 27th ACM SIGSOFT International Symposium on Software Testing and Analysis, Amsterdam Netherlands, ACM, 333-343, July 2018.

[HM22] N. J. Higham & T. Mary. Mixed precision algorithms in numerical linear algebra, Acta Numerica, 31:347–414, https://doi.org/10.1017/S0962492922000022, 2022.

[HSW21] N.-M. Ho, H. D. Silva, and W.-F. Wong, GRAM: A Framework for Dynamically Mixing Precisions in GPU Applications, ACM Transactions on Architecture and Code Optimization, 18, 1-24, 2021.

[HT17] J. Hook and F. Tisseur, Incomplete LU preconditioner based on max-plus approximation of LU factorization, SIAM SIMAX 2017.

[HTD+18] A. Haidar, S. Tomov, J. Dongarra, N. J. Higham, Harnessing GPU tensor cores for fast FP16 arithmetic to speed up mixed-precision iterative refinement solvers, SuperComputing 2018.

[JC08] F. Jézéquel, J.-M. Chesneaux, CADNA: a library for estimating round-off error propagation, Computer Physics Communications, 178(12), pages 933-955, 2008.

[Jez21] F. Jézéquel, Benefits of stochastic arithmetic in high performance simulations and arbitrary precision codes, plenary presentation, 19th international symposium on Scientific Computing, Computer Arithmetic and Validated Numerics (SCAN 2020), virtual conference, September 2021.

[JHH21] F. Jézéquel, S. sadat Hoseininasab & T. Hilaire, Numerical validation of half precision simulations, 1st Workshop on Code Quality and Security (CQS 2021) in conjunction with WorldCIST'21 (9th World Conference on Information Systems and Technologies), https://hal.inria.fr/hal-03138494, 2021.

[KSW+19] P. V. Kotipalli, R. Singh, P. Wood, I. Laguna, and S. Bagchi, AMPT-GA: Automatic Mixed Precision Floating Point Tuning for GPU Applications, in Proceedings of the ACM International Conference on Supercomputing, Phoenix Arizona, ACM, 160-170, June 2019.

[LLL+06] J. Langou, J. Langou, P. Luszczek, J. Kurzak, A. Buttari, J. Dongarra, Exploiting the performance of 32 bit floating point arithmetic in obtaining 64 bit accuracy (revisiting iterative refinement for linear systems), Supercomputing 2006.

[LVM+19] M. O. Lam, T. Vanderbruggen, H. Menon, and M. Schordan, Tool Integration for Source-Level Mixed Precision, in 2019 IEEE/ACM 3rd International Workshop on Software Correctness for HPC Applications (Correctness), 27-35, 2019.

[LWR+19] I. Laguna, P. C. Wood, S. Ranvijay, and S. Bagchi, GPUMixer: Performance-Driven Floating-Point Tuning for GPU Scientific Applications, vol. 11501, Springer, pp. 227–246, 2019.

[PGM03] S. Putot, E. Goubault, and M. Martel. Static analysis-based validation of floating-point computations. In Numerical Software with Result Verification, volume 2991 of Lecture Notes in Computer Science, 306-313, Springer, 2003.

[RNN+13] C. Rubio-González, C. Nguyen, H. D. Nguyen, J. Demmel, W. Kahan, K. Sen, D. H. Bailey, C. Iancu, and D. Hough, Precimonious: Tuning Assistant for Floating-Point Precision, in Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis, SC'13, New York, NY, USA, ACM, 27:1-27:12, 2013.

[TW92] K. Turner and H. Walker, Efficient High Accuracy Solutions with GMRES(m), SIAM SISC 1992.

[Vig93] J. Vignes. A stochastic arithmetic for reliable scientific computation. Mathematics and Computers in Simulation, 35(3):233–261, 1993.

[ZHL+22] M. Zounon, N. J.Higham, C. Lucas and F. Tisseur; Performance impact of precision reduction in sparse linear systems solvers. PeerJ Computer Science, https://doi.org/10.7717/peerj-cs.778, 2022.