Phase 1 du projet

MTH8211

Ulrich Baron-Fournier, Petru Lepreanu

Remise 1 LSRN : Une méthode parallèle pour les systèmes linéaires fortement rectangulaires

Lien du Github : https ://github.com/Ulrizpascuit/Projet_MTH8211.git

Description de la problématique

Le projet s'intéresse à la résolution de grands systèmes linéaires fortement rectangulaires, c'est-à-dire des problèmes aux moindres carrés linéaires où la matrice A est de taille $m \times n$ avec un écart extrême entre m et n (bien plus d'équations que d'inconnues si $m \gg n$, cas surdéterminé, ou l'inverse $m \ll n$, cas sous-déterminé). Dans de telles situations, on cherche typiquement à calculer la solution de longueur minimale du problème $\min_x \frac{1}{2}\|Ax - b\|_2^2$ (la solution de norme minimale satisfaisant au mieux $Ax \approx b$). Ces problèmes apparaissent dans de nombreuses applications scientifiques et d'ingénierie, et la demande en solveurs plus rapides et précis s'accroît avec l'augmentation de la taille des données et des modèles considérés.

Les approches classiques pour les moindres carrés (telles que la résolution des équations normales $A^T Ax = A^T b$ ou les factorisations QR/SVD) deviennent coûteuses ou peu pratiques lorsque m et n sont très déséquilibrés et peuvent souffrir de problèmes de conditionnement numérique. Les méthodes itératives comme LSQR atténuent le coût mémoire, mais leur vitesse de convergence dépend fortement du conditionnement du système normal $A^T A$. Dans le cas de systèmes extrêmement rectangulaires (surdéterminés ou sous-déterminés), il est important de disposer d'une méthode efficace capable de préadapter le problème pour en améliorer le conditionnement. C'est dans ce contexte qu'intervient LSRN, une méthode introduisant un élément aléatoire dans le processus de résolution. En effet, LSRN utilise un échantillonnage aléatoire gaussien (projection aléatoire) pour construire un préconditionneur qui rend le système préconditionné extrêmement bien conditionné avec une probabilité élevée. Par ailleurs, cette étape de préconditionnement est hautement parallélisable et bénéficie directement de la structure des matrices creuses ou d'opérateurs linéaires rapides, ce qui la rend adaptée au calcul distribué moderne. En somme, la problématique sous-jacente est de résoudre rapidement et avec une grande précision des systèmes très rectangulaires de grande taille, un défi que les méthodes aléatoires parallèles récentes comme LSRN cherchent à relever.

Description des objectifs

Les objectifs principaux du projet sont les suivants :

1. **Compréhension de l'article LSRN (Meng et al., 2014)** — Lire et analyser en détail l'article de référence présentant LSRN. Il s'agit de bien saisir le fonctionnement de l'algorithme et ses fondements théoriques (sans nécessairement entrer dans toutes les preuves mathématiques), afin de maîtriser les idées clés de la méthode.
2. **Explication du fonctionnement de LSRN** — Fournir une explication claire avec nos propres mots du principe de LSRN et de ses principales propriétés. Cela inclut la description de l'algorithme (comment le tirage aléatoire est utilisé, quel est le rôle du préconditionnement et de l'étape itérative) ainsi que la discussion de ses avantages (conditionnement amélioré, parallélisation, précision attei-

gnable, gestion du rang déficient, etc.). L'objectif est de montrer que nous avons assimilé le contenu de l'article en pouvant le reformuler et en souligner les points marquants.

3. **Implémentation efficace de LSRN en Julia** — Développer une implémentation logicielle de l'algorithme LSRN en utilisant le langage Julia, en s'attachant à l'efficacité et à la performance. Concrètement, cela implique d'exploiter au mieux les bibliothèques optimisées disponibles : par exemple, utiliser les routines de factorisation de LAPACK et la librairie SuiteSparseQR (SPQR) pour les opérations matricielles denses ou creuses. De plus, nous comptons tirer parti de Krylov.jl pour l'étape de résolution itérative (par ex. utiliser LSQR ou une méthode de Krylov appropriée). L'implémentation devra également être écrite de façon à pouvoir exploiter le parallélisme (calcul multi-cœurs ou multi-threading), conformément à l'esprit de LSRN.
4. **Reproduction de tests numériques** — Valider notre implémentation en répliquant une partie des expériences numériques présentées dans l'article de Meng et al. (ou en réalisant des tests similaires). Il s'agira par exemple de comparer les performances de LSRN à celles d'autres solveurs de référence sur des systèmes de grande taille, tant denses que clairsemés. Les tests porteront sur la précision obtenue (erreur résiduelle, qualité de la solution) et sur les temps de calcul en fonction de la taille du problème et du nombre de cœurs utilisés. Idéalement, nous reproduirons des scénarios comparables à ceux de l'article (par exemple en testant LSRN face à DGELSD de LAPACK, à un solveur aléatoire du type Blendenpik, ou à SuiteSparseQR, selon les ressources disponibles). Ces reproductions visent à confirmer que notre implémentation atteint les performances annoncées et respecte la précision attendue.

Description du plan d'action

Pour atteindre les objectifs fixés, nous avons établi un plan d'action découpé en plusieurs étapes clés :

— **Étude approfondie de la littérature (LSRN et contexte)** : Dans un premier temps, nous allons lire en détail l'article de Meng et al. (2014) afin de comprendre le fonctionnement de LSRN. Parallèlement, nous pourrons consulter des ressources complémentaires sur les méthodes aléatoires en algèbre linéaire numérique pour bien situer LSRN par rapport à d'autres approches. Cette phase inclut la prise de notes et la clarification des concepts tels que le préconditionnement par projection aléatoire, l'utilisation de la SVD ou QR sur une matrice aléatoirement projetée, et l'itérateur LSQR/CS. Si des points restent ambigus, nous pourrons solliciter l'aide du professeur ou du chargé de laboratoire pour des clarifications, conformément aux recommandations du projet.

— **Implémentation algorithmique en Julia** : Une fois l'algorithme bien compris, nous entamerons le développement en Julia de LSRN. Cette implémentation se fera en tirant parti des bibliothèques hautes performances disponibles. Par exemple, nous utiliserons les méthpodes LAPACK (déjà accessibles via Julia/LinearAlgebra) pour effectuer des factorisations denses nécessaires, et la librairie SPQR de SuiteSparse pour les factorisations creuses si besoin. Pour la partie itérative, nous prévoyons d'utiliser Krylov.jl qui fournit notamment une implémentation de LSQR et d'autres solveurs itératifs adaptés. Nous veillerons à ce que le code exploite le parallélisme : Julia permet le calcul multi-threadé, ce qui nous permettra de tester la passage à l'échelle de notre implémentation sur plusieurs cœurs. L'implémentation sera validée sur de petits exemples pour s'assurer de son bon fonctionnement avant de passer aux grands tests.

— **Tests numériques et validation** : Après avoir une version fonctionnelle de LSRN, nous entreprendrons une plusieurs séries de tests numériques pour évaluer sa performance et vérifier qu'elle reproduit les résultats attendus. Nous commencerons par des cas simples (petites matrices denses et creuses générées aléatoirement, par exemple) pour vérifier la précision de la solution (comparaison avec une solution calculée par une méthode directe) et le comportement de convergence. Ensuite, nous passerons à des expériences à plus grande échelle, semblables à celles de l'article de référence : par exemple, résoudre des systèmes de grande dimension (plusieurs millions d'inconnues ou d'équations si possible) et mesurer le temps de calcul, la possibilité de grossir les problèmes en fonction du nombre de threads, ainsi que la précision obtenue. Nous comparerons les performances de notre implémentation LSRN à celles d'autres solveurs disponibles en Julia afin d'avoir des références de comparaison. Une attention particulière sera portée à reproduire au moins partiellement les tests de performance de Meng et al. :

par exemple, comparer LSRN à DGELSD (solveur de moindres carrés de LAPACK basé sur la SVD) sur des problèmes denses, et à SuiteSparseQR sur des problèmes non dense (sparse matrix), comme dans l'étude originale.

— **Extensions et utilisation de ressources externes** : Conformément aux suggestions du projet, nous envisageons d'enrichir le projet par des éléments complémentaires. Cela pourra inclure l'application de LSRN à un cas concret qui nous intéresse (par exemple un problème réel de régression linéaire sur de grandes données, ou un problème inverse en traitement d'images), afin de démontrer l'utilité de la méthode dans un contexte applicatif. De plus, nous pourrons exploiter des jeux de données ou matrices de test existantes, en particulier issues de la collection SuiteSparse Matrix Collection, qui propose de nombreux systèmes creux représentatifs de problèmes industriels ou académiques. L'utilisation de ces matrices réelles, en complément des cas aléatoires, permettra d'évaluer la robustesse et l'efficacité de LSRN sur des scénarios variés. Tout au long de ces étapes, nous documenterons soigneusement nos résultats et garderons la trace des configurations de test, dans un souci de reproductibilité.

Description de l'impact attendu

L'issue de ce projet devrait démontrer l'intérêt des méthodes aléatoires parallèles comme LSRN pour la résolution de très grands systèmes linéaires. En particulier, on s'attend à une amélioration significative de l'efficacité par rapport aux approches traditionnelles. Grâce à son préconditionnement aléatoire, LSRN parvient à limiter le nombre d'itérations nécessaires en assurant un faible conditionnement du système linéaire final. Couplé à une implémentation parallèle, cela permet d'obtenir des temps de calcul réduits et prévisibles même sur des problèmes de très grande taille. En effet, LSRN offre un temps d'exécution hautement prévisible (comparable à celui de méthodes directes) tout en passant à l'échelle en environnement parallèle de façon efficace. Des travaux antérieurs ont montré que sur des systèmes denses de grande dimension, LSRN rivalise avec le solveur dense DGELSD de LAPACK pour les cas surdéterminés, et s'avère nettement plus rapide pour les cas sous-déterminés. Sur des systèmes creux sans structure particulière, LSRN se révèle également significativement plus rapide que les autres solveurs existants, aussi bien pour les problèmes surdéterminés que sous-déterminés. Ces gains de performance se traduiront par la capacité de résoudre des problèmes autrefois inaccessibles ou très longs à traiter, ouvrant la voie à l'analyse de jeux de données plus vastes et à la modélisation de phénomènes plus complexes dans des délais raisonnables.

Un autre impact attendu de ce projet est de souligner l'importance de la reproductibilité des résultats numériques. En reproduisant les tests de l'article original, nous allons valider expérimentalement les propriétés avancées par Meng et al. pour LSRN, ce qui renforce la confiance dans ces conclusions. Cette démarche s'inscrit dans les bonnes pratiques de la recherche numérique : être capable de reproduire les performances annoncées par une méthode est essentiel pour en garantir la fiabilité et pour faciliter son adoption par d'autres utilisateurs. Notre rapport de projet documentera précisément les expériences réalisées et leurs résultats, de sorte qu'un tiers pourrait en principe les reproduire à l'identique. Ce souci de vérification confère une valeur scientifique supplémentaire à notre travail, au-delà de la simple implémentation.

Enfin, l'impact du projet peut se mesurer en termes d'ouverture vers les applications scientifiques et d'ingénierie. Les solveurs de systèmes linéaires sont au cœur de nombreux domaines : optimisation et apprentissage automatique (régression sur de très larges données), traitements d'images et du signal (problèmes inverses, tomographie), modélisation numérique en ingénierie (méthodes des moindres carrés en ajustement de modèles, simulations, etc.). Une méthode comme LSRN, capable de résoudre des systèmes de très grande taille de manière extensible et avec une précision élevée, représente un atout précieux pour ces domaines. En démontrant son efficacité et en disposant d'une implémentation Julia performante, nous facilitons potentiellement son transfert vers la pratique : les ingénieurs et chercheurs pourront l'utiliser pour aborder des problèmes de grande envergure qu'ils hésitaient à traiter faute d'outil adapté. En somme, ce projet contribuera à promouvoir l'utilisation des méthodes numériques aléatoires avancées pour accélérer la résolution de problèmes linéaires massifs, ce qui profitera à la communauté scientifique et technique confrontée à des défis computationnels de plus en plus importants.