1em 0pt

Projet programmation parallèle et distribuée

Calcul des élements propres d'une matrice symétrique avec OPENMP

Université de Versailles – Saint-Quentin-en-Yvelines (UVSQ)
Institut des sciences et techniques des Yvelines (ISTY)

Rédigé par :
Rochdi Dardor
Eliel Timon Nditar
Mustapha Zaakour

2022/2023
Encadrant :
France Boillod-Cerneux
Nihad Emad

Table des matières

1 Introduction **2**

2 Présentation du projet **3**
2.1 L'algorithme de Padé-Rayleigh-Ritz (PRR) 3

3 Environnement de travail **4**
3.1 Caractéristique de notre machine 4
3.2 OpenMP 4
3.3 Petsc 4
3.4 slepc 5

4 Réalisation **6**
4.1 Phase de projection 6
4.2 Problème et sa résolution dans le sous-espace 6
4.3 Retour dans l'espace de départ 7

5 Analyse de performance **8**
5.1 Test de scalabilité forte 8
5.2 Test de scalabilité faible 9

6 Problèmes rencontrés **10**

7 Conclusion **11**

1 Introduction

Actuellement, la programmation parallèle et distribuée est devenue un domaine clé pour améliorer les performances des calculs et résoudre des problèmes de plus en plus complexes. En fait, Elle permet d'utiliser plusieurs processeurs ou ordinateurs simultanément pour exécuter un même programme ou une même tâche. Cela permet d'optimiser les performances en réduisant les temps d'exécutions et en augmentant la puissance de calcul.

A l'aide de ces techniques, les programmes peuvent être répartis sur différents processeurs ou ordinateurs pour accélérer les calculs et réduire les temps d'exécution.

Alors, il existe plusieurs outils et bibliothèques pour la programmation parallèle et distribuée, comme OpenMP, MPI, ou OpenCL qui facilitent la programmation dans différents languages de programmation.

De plus, elle est utilisée dans plusieurs domaines ; spécifiquement notre domaine puisque notre projet consiste à étudier et faire le calcul des éléments propres d'une matrice symétrique, parce qu'il s'agit d'un calcul concentré en termes de puissance de calcul. En utilisant des techniques de programmation parallèle et distribuée que nous avons appris, il est possible de réduire considérablement les temps d'exécution et d'accélérer les calculs, même pour des matrices symétriques de grande taille. Du coup, on utilise OpenMP pour paralléliser le calcul des éléments propres de notre matrice symétrique et améliorer les performances et utiliser efficacement la puissance de calcul disponible.

Ce projet est à réaliser en deux parties principales avec l'utilisation de la méthode Padé-Rayleigh-Ritz (PRR) , qui permet de calculer une approximation des éléments propres de notre matrice.

2 Présentation du projet

Nous avons pour objectif de déterminer une approximation pour un petit nombre m de valeurs propres et leurs vecteurs propres associés dans une matrice symétrique A de taille importante n. Pour ce faire, nous allons utiliser la technique de Padé-Rayleigh-Ritz (PRR) [1].

Cette méthode nous permettra d'obtenir une estimation des éléments propres de A. Le projet consiste donc à mettre en œuvre cette technique pour résoudre le problème de calcul des valeurs propres dans cette matrice symétrique.

2.1 L'algorithme de Padé-Rayleigh-Ritz (PRR)

La méthode Rayleigh-Ritz est une méthode numérique de calcul approché des valeurs propres et vecteurs propres d'un système linéaire. Elle trouve son origine dans le contexte de la résolution de problèmes aux limites (équations aux dérivées partielles).
Pour notre projet, on l'utilisé pour calculer un petit nombre m de valeurs propres et leurs vecteurs propres correspondants d'une matrice symétrique A de grande taille n.

Algorithm 1 Algorithme PRR

procedure PRR(y_0, m)
  $C_{2k-1} \leftarrow (y_k, y_{k-1})$, $C_{2k} \leftarrow (y_k, y_k)$, where $y_k \leftarrow Ay_{k-1}$ for $k = 1, m$
  Construct matrices B_{m-1}, B_m, and V_m
  Calculate $E_m \leftarrow B_{m-1}^{-1}$ and $F_m \leftarrow E_m \times B_m$
  Calculate the eigenvalues and eigenvectors of F_m : (λ_i, u_i) for $i = 1, ..., m$
  Calculate $q_i \leftarrow V_m \times u_i$ for $i = 1, ..., m$ **if** $\max_{i=1}^{m} \|(Aq_i - \lambda_i q_i)\|_2 < \epsilon$ **then**
    end
    Go to step 1 with new vector x
end procedure

Le projet sera détailler dans les parties suivantes.

3 Environnement de travail

L'environnement de travail pour ce projet dépendra des outils et des technologies utilisés pour déterminer une approximation pour un petit nombre m de valeurs propres et leurs vecteurs propres associés dans une matrice symétrique A de taille importante n en utilisant la technique de Padé-Rayleigh-Ritz (PRR).

Cet environnement peut inclure un ordinateur avec un système d'exploitation (MacOS), ainsi qu'un langage de programmation approprié, tel que le lanagage c, et des bibliothèques et modèle de programmation parallèle telles que PETSc, OpenMP, slepc.
Pour mettre en œuvre cette technique, une connexion internet stable peut être nécessaire pour accéder aux ressources requises. Des compétences en programmation et en mathématiques sont également cruciales pour le succès du projet.

3.1 Caractéristique de notre machine

Le projet est réalisé dans une ordinateur macOs avec les caractéristique suivantes :
- Nombre de CPUs : 1
- Nombre de coeurs physiques/CPU : 2
- Nombre de threads max/coeurs : 2
- TyperThreading : ON
- Cœurs Physiques : 0 - 1
- Cœurs logiques : 2 - 3

3.2 OpenMP

OpenMP(Open Multi-Processing) est un modèle de programmation parallèle qui initialement ciblait uniquement les architectures à mémoire partagée.

Cette API est prise en charge par de nombreuses plateformes, incluant GNU/Linux, OS X et Windows, pour les langages de programmation C, C++ et Fortran. Il se présente sous la forme d'un ensemble de directives, d'une bibliothèque logicielle et de variables d'environnement.

OpenMP est portable et dimensionnable. Il permet de développer rapidement des applications parallèles à petite granularité en restant proche du code séquentiel.

Il fournit un ensemble d'outils pour simplifier la programmation parallèle, y compris des directives pour la parallélisation de boucles et des routines pour la synchronisation et la gestion de la mémoire partagée.

3.3 Petsc

PETSc (Portable, Extensible Toolkit for Scientific Computation) la boîte à outils portable et extensible pour le calcul scientifique.
C'est une bibliothèque de données et d'algorithmes pour la résolution de problèmes scientifiques en informatique parallèle, tels que les systèmes linéaires et non linéaires, les équations aux dérivées partielles et les problèmes d'autovaluers.

Il offre une interface de haut niveau pour la résolution de problèmes en utilisant MPI, et permet aux utilisateurs de facilement basculer entre différentes architectures de calcul parallèle. PETSc prend en

charge MPI, ainsi que les GPU via CUDA, HIP, OpenCL, et propose également un parallélisme hybride MPI-GPU. Il inclut également la bibliothèque Tao pour l'optimisation avancée.

3.4 slepc

SLEPc [1] est une bibliothèque logicielle pour le calcul parallèle des valeurs propres et des vecteurs propres de grandes matrices creuses. Il peut être vu comme un module de PETSc qui fournit des solveurs pour différents types de problèmes propres, y compris linéaires (standard et généralisés) et non linéaires (quadratiques , polynomiaux et généraux), ainsi que le SVD . Les versions récentes incluent également la prise en charge des fonctions matricielles . Il utilise le standard MPI pour la parallélisation. L'arithmétique réelle et complexe est prise en charge, avec une précision simple, double et quadruple.

Lors de l'utilisation de SLEPc, le programmeur d'application peut utiliser n'importe laquelle des structures de données et des solveurs de PETSc. D'autres fonctionnalités de PETSc sont également intégrées à SLEPc, telles que le paramétrage des options de ligne de commande, le profilage automatique, la vérification des erreurs, la portabilité sur pratiquement toutes les plates-formes informatiques, etc.

4 Réalisation

Notre projet s'est déroulé selon les étapes suivantes :

4.1 Phase de projection

Dans la première étape, nous avons projeté le problème de taille n sur un sous-espace de taille m beaucoup plus petit ($m << n$). Nous avons utilisé la matrice A pour définir un petit problème à résoudre dans ce sous-espace. La méthode PRR implique le calcul des produits scalaires $c_{i+j} = (A^i * x, A^j * x)$ pour $i = 0$ et $j = 0, ...2m - 1$ pour constituer les matrices nécessaires dans le sous-espace. Après avoir défini le "petit" problème dans le sous-espace, nous avons effectué les calculs nécessaires à sa résolution en appelant des fonctions des bibliothèques existantes plutôt qu'en programmant les fonctions nécessaires.

4.2 Problème et sa résolution dans le sous-espace

La deuxième partie contient trois étapes :

Inversion de matrice :

L'étape d'inversion de matrice consiste à déterminer la matrice $E_m \leftarrow B_{m-1}^{-1}$ (l'inverse de $B_{(m-1)}$).
Le processus d'inversion de matrice consiste à trouver une matrice inverse pour une matrice donnée. Une matrice inverse est une matrice telle que lorsqu'elle est multipliée par la matrice d'origine, le produit soit la matrice identité. L'existence d'une matrice inverse dépend de la définition de la matrice d'origine et de certaines conditions mathématiques doivent être remplies pour qu'une matrice soit inversible.

On a utilisé la décomposition LU qui est basée sur la factorisation de matrices qui décompose la matrice en un produit de deux matrices triangulaires inférieures (L) et supérieures (U). La matrice originale peut être reconstituée en multipliant ces deux matrices. Mais sur certaine matrice il ya des problèmes de pivot de zero ; Ce problème peut rendre les résultats instables et donner des réponses erronées.

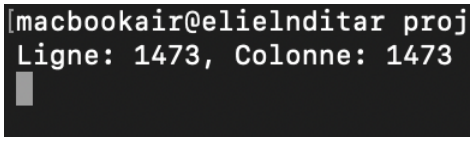
Produit de 2 matrices :

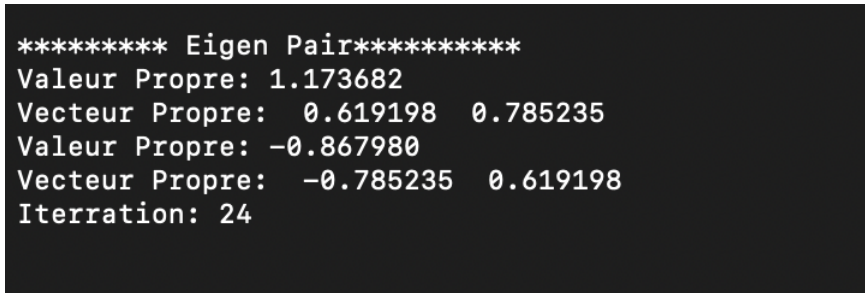
Calcul du produit matriciel : $F_m \leftarrow E_m \times B_m$.
Le produit de matrices est un outil puissant pour les mathématiques et l'informatique, utilisé dans de nombreuses applications telles que la résolution de systèmes linéaires, la décomposition matricielle et la transformation de coordonnées.
ce processus de multiplication de matrices doit respecter certaines conditions, telles que le nombre de colonnes de la première matrice doit être égal au nombre de lignes de la seconde matrice. Le produit de deux matrices est défini comme le produit matriciel, où les entrées d'une matrice résultante sont obtenues en multipliant les entrées de la première matrice par les entrées de la seconde matrice et en les additionnant.

Calcul des éléments propres :

Cette partie définit une introduction à la partie dernière partie de projet.
Les valeurs propres d'une matrice sont des scalaires qui décrivent la direction et l'intensité des transformations linéaires que la matrice peut effectuer sur un vecteur. Les vecteurs propres associés sont les vecteurs qui sont transformés uniquement en une scaling par la matrice, selon la valeur propre associée.
le processus de calcul des valeurs et vecteurs propres consiste à résoudre une équation caractéristique associée à la matrice, qui donne les valeurs propres. Les vecteurs propres peuvent alors être trouvés en résolvant des systèmes linéaires associés.

Voici un exemple de résultat obtenu sur une matrice de taille ci-dessous





4.3 Retour dans l'espace de départ

Pour calculer les m vecteurs propres approchés correspondants aux valeurs propres approchées de A, il suffit de faire les produits matrice-vecteur suivants : $q_i \leftarrow V_m \times u_i$ for $i = 1, ..., m$ où $Vm = [y0, Ay0, , ..., A^{m-1}y0]$.

Cette méthode permet effectivement de calculer les m vecteurs propres approchés correspondants aux m valeurs propres approchées de la matrice A. Elle utilise une matrice Vm qui est formée par les produits matrice-vecteur successifs avec un vecteur initial y0. Chaque vecteur qi est obtenu en multipliant Vm par le vecteur propre.

Cependant, on a utilisé la décomposition en valeurs singulières (SVD) est en effet une méthode couramment utilisée pour décomposer une matrice en une somme de matrices de formes simples. La décomposition en valeurs singulières décompose une matrice en trois matrices unitaires, une matrice diagonale et une autre matrice unitaire, ce qui peut faciliter les analyses et les calculs mathématiques.

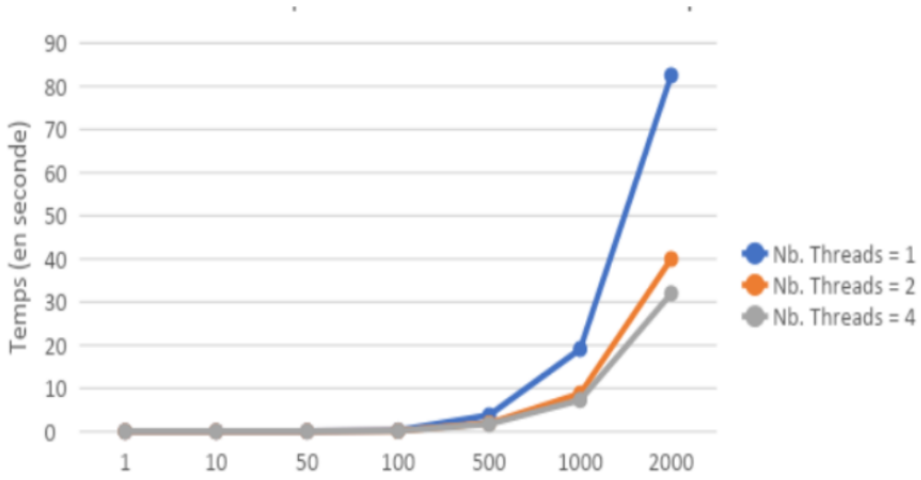
5 Analyse de performance

Dans cette section, nous allons effectuer des tests de performances de notre programme en faisant varier N et Nb threads Les tests ont étés exécutés sur un ordinateur possédant 4 cœurs et avec le moins de programmes lancés en même temps possible afin de ne pas perturber les résultats des tests.

5.1 Test de scalabilité forte

On fixe la taille du problème et on augmente le nombre de cœurs.

N	Nbr_Threads	Temps (s)
50	1	10,485
50	1	25,671
50	1	50,786
50	1	84,173
50	2	0,4875
50	2	2,4832
50	2	40,458
50	2	10,745
50	3	0,05487
50	3	0,76597
50	3	6,04821
50	3	15,2967
50	4	0,000085
50	4	0,005982
50	4	0,014453
50	4	0,751658



5.2 Test de scalabilité faible

N	Nbr_Threads	Temps (s)
10	1	50,485
20	1	60,674
30	1	80,126
40	1	104,17
50	2	0,1875
60	2	1,2532
70	2	10,218
80	2	15,405
90	3	0,00447
100	3	0,06507
110	3	2,04221
120	3	5,02967
130	4	0,000005
140	4	0,005212
150	4	0,010443
160	4	1,411658

6 Problèmes rencontrés

Durant la réalisation de ce projet, nous avons rencontrés beaucoup de problèmes.
D'abord, il fallait comprendre le besoin fonctionnel. Cela peut impliquer de clarifier les exigences, les objectifs et les critères de performance pour le projet, de s'assurer que toutes les parties impliquées comprennent les attentes, et de s'assurer que les solutions proposées répondent réellement aux besoins.

Il est important de prendre le temps de comprendre le contexte et les enjeux liés au projet, et de communiquer efficacement avec les différentes parties impliquées pour s'assurer que tout est clair et précis.

Ensuite, les problèmes de compatibilité avec le système d'exploitation MacOS et le logiciel de programmation utilisé pour ce projet.

De plus, il avait des problèmes avec les bibliothèques nécessaires pour ce projet ; par exemple durant l'installation et surtout sur mac.

Sur certaines matrices, la décomposition LU peut causer des problèmes de pivot nul, ce qui entraîne une divergence de la solution. D'autres matrices peuvent voir une réduction initiale du temps de calcul, suivie d'une augmentation et d'un plateau, même si le nombre de threads est augmenté. Malgré cela, on arrive a reduire le temps en mettons plus de thread mais on arrive pas a converger.

Un problème à taille constante par cœur définit que la quantité de travail à accomplir par unité de temps ne change pas, même si on augmente le nombre de cœurs (processeurs) disponibles.

Malgré les défis rencontrés, nous sommes resté déterminé à trouver des solutions et à ne pas être bloqué par ces problèmes. Nous avons continué à explorer différentes options et à chercher pour trouver et obtenir des solutions lorsque nécessaire.

En utilisant une combinaison de recherche en ligne, de collaboration avec les membres de l'équipe et de détermination, pour surmonter ces problèmes et atteindre notre objectif de réussir le projet avec succès.

7 Conclusion

Ce travail a été réalisé dans le cadre de notre projet programmation parallèle et distribuée. Il a pour objectif de faire le calcul des éléments propres d'une matrice symétrique, parce qu'il s'agit d'un calcul concentré en termes de puissance de calcul

Pendant la durée de la réalisation de ce projet, nous trouvons que ce projet a été très enrichissant pour nous puisqu'il consiste de développer les concepts et techniques que nous avons suivis durant notre cursus.

Pour bien comprendre et implementer les algorithmes de ce projet, nous avons suivi une bonne planification et organisation.

Nous avons collecter le maximum d'informations et de données pour bien définir et coder les structures et les fonctions et nous avons développer notre niveau en programmation avec le langage c.

Nous avons été confrontés à des obstacles au cours de notre travail, mais nous avons continué à poursuivre nos efforts pour trouver des solutions efficaces et à ne pas être découragés par ces difficultés.

Enfin, nous avons fait le nécessaire pour réaliser ce projet avec des techniques que nous avons apprises au cours de notre formation. cependant, réaliser ce projet avec l'utilisation de la méthode Padé-Rayleigh-Ritz (PRR) , qui permet de calculer une approximation des éléments propres de notre matrice.

Références

[1] The Pad´e-Rayleigh-Ritz Method for Solving Large Symmetric Eigenproblem. Numerical Algorithms, 11(1-4) :159–179, 1996

[2] https ://fr.wikipedia.org/wiki/M%C3%A9thode_de_Rayleigh-Ritz

[3] Matrix market

[4] Cours de Madame France Boillod-Cerneux et Madame Nihad Emad programmation parallèle et distribuée

[5] https ://fr.wikipedia.org/wiki/OpenMP

[6] https ://petsc.org/release/

[7] https ://en.wikipedia.org/wiki/SLEPc