UNIVERSITÉ DE VERSAILLES
SAINT-QUENTIN-EN-YVELINES

MASTER 2 CALCUL HAUTE PERFORMANCE & SIMULATION

Rapport Projet Méthodes de Programmation Numérique Avancée

Realisé Par:

Hamid Ramdani

Sujet:

Décomposition en Valeurs Singulières
SVD

Année Universitaire:

2021/2022

Plan du Rapport

Introduction

En algèbre linéaire, la décomposition en valeurs singulières (SVD) est une factorisation d'une matrice réelle ou complexe. Elle permet la factorisation des matrices triangulaires, et peut être considérée comme la procédure de diagonalisation des matrices carrées.

L'idée de la décomposition en valeurs singulières est similaire à la décomposition en valeurs propres, mais fonctionne pour toute matrice A de taille m×n. On peut dire que la décomposition généralise la notion de valeurs propres aux matrices rectangulaires.

Les premières méthodes pratiques de calcul de SVD remontent aux années 1950s. Elles ressemblent beaucoup à l'algorithme des valeurs propres de Jacobi. Cependant, elles ont été remplacées par la méthode de Gene Golub et William Kahan publiée en 1965. l'algorithme de Golub/Kahan qui est encore le plus utilisé aujourd'hui.

Les applications mathématiques de la SVD comprennent le calcul du pseudo-inverse, l'approximation matricielle et la détermination du rang, de la plage et de l'espace nul d'une matrice. La SVD est également extrêmement utile dans tous les domaines de la science, de l'ingénierie et des statistiques, tels que le traitement du signal, l'ajustement des données par les moindres carrés et le contrôle des processus.

Dans ce travail nous nous intéresserons à la méthode décomposition en valeurs singulières, tout d'abord nous préciserons le problème à résoudre, puis nous expliquerons la méthode utilisée pour le calcul, qui représente l'approche utilisée dans le code implémenté et amélioré par l'utilisation d'outils de parallélisation afin d'avoir les meilleures performances possibles pour ce calcul.

Chapitre 1

Problématique

1.1 Relation avec la Décomposition en Valeurs Propres

Soit **A** une matrice carrée $n \times n$ avec n vecteurs propres qi linéairement indépendants (où i = 1, ..., n). Alors A peut être factorisée comme suit:

$$\mathbf{A} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^{-1}$$

Avec:

Q: est la matrice carrée n $\times$ n dont la ième colonne est le vecteur propre qi de A.
Λ: est la matrice diagonale dont les éléments diagonaux sont les valeurs propres correspondantes.

Cette méthode ne peut pas être appliquée aux matrices rectangulaires et aux matrices non diagonalisables. Il a donc fallu trouver une autre méthode applicable aux matrices de toute taille qu'elles soient diagonalisables ou non, d'où l'apparition de la décomposition en valeurs singulières.

Sa théorie est assez similaire à celle de la décomposition en valeurs propres, sauf qu'elle s'applique à n'importe quelle taille de matrice, y compris une matrice comportant beaucoup plus de lignes m que de colonnes n.

Bien que la décomposition en valeurs singulières s'applique à la fois aux matrices réelles et complexes, nous n'aborderons ici que les matrices à coefficient réel pour faciliter le processus.

1.2 Principe de la SVD

Un nombre réel non négatif $\boldsymbol{\sigma}$ est une valeur singulière pour **M** si et seulement s'il existe des vecteurs de longueur unitaire *u* dans *Km* et *v* dans *Kn* tels que:

$$\mathbf{Mv} = \sigma\mathbf{u} \;\; {}_{et} \; \mathbf{M}^*\mathbf{u} = \sigma\mathbf{v}$$

Les vecteurs *u* et *v* sont appelés respectivement vecteurs singuliers de gauche et vecteurs singuliers de droite pour $\boldsymbol{\sigma}$.

Dans toute décomposition en valeurs singulières:

$$\mathbf{M} = \mathbf{U\Sigma V}^*$$

les entrées diagonales de $\boldsymbol{\sigma}$ sont égales aux valeurs singulières de **M**. Les premières $p = min(m, n)$ colonnes de **U** et **V** sont, respectivement, des vecteurs singuliers gauche et droit pour les valeurs singulières correspondantes. Par conséquent, le théorème ci-dessus implique que:

- Une matrice $m \times n$ **M** a au plus p valeurs singulières distinctes.
- Il est toujours possible de trouver une base unitaire **U** pour *Km* avec un sous-ensemble de vecteurs de base couvrant les vecteurs singuliers de gauche de chaque valeur singulière de **M**.
- Il est toujours possible de trouver une base unitaire **V** pour *Kn* avec un sous-ensemble de vecteurs de base couvrant les vecteurs singuliers droits de chaque valeur singulière de **M**.

Le calcul des matrices U , V et d'une décomposition SVD d'une matrice rectangulaire quelconque **M** peut être fait à l'aide de la décomposition EVD des matrices carrées $\mathbf{M^TM}$ et $\mathbf{MM^T}$. En effet, en utilisant la décomposition de A, on a :

$$A^T A = X \Lambda X^T = V \Sigma^2 V^T$$

et

$$A A^T = Y \Lambda Y^T = U \Sigma^2 U^T$$

Par conséquent, $\mathbf{V} = \mathbf{X}$ (resp. $\mathbf{U} = \mathbf{Y}$) et $\mathbf{\Sigma^2} = \mathbf{\Lambda}$ sont les vecteurs et les valeurs propres de $\mathbf{M^T M}$ (resp. $\mathbf{MM^T}$).

Pour obtenir les valeurs et vecteurs (droite et gauche) singuliers d'une matrice rectangulaire M, il suffit de calculer les valeurs et vecteurs propres des matrices symétriques $\mathbf{M^T M}$ et $\mathbf{MM^T}$. Lorsque *n « m* , il est plus intéressant de calculer la EVD de la matrice $\mathbf{M^T M}$. Dans le cas contraire, il est préférable de résoudre le problème EVD de la matrice $\mathbf{MM^T}$.

Chapitre 2

Approche utilisée

La bidiagonalisation est une décomposition de matrices unitaires telles que:

$$\mathbf{U}^*\mathbf{AV} = \mathbf{B} \quad \text{,avec:}$$

- **U** et **V** sont des matrices unitaires (orthogonales).
- * désigne la transposition hermitienne.
- **B** est la bidiagonale supérieure.
- **A** peut être rectangulaire.

Pour les matrices denses, les matrices unitaires gauche et droite sont obtenues par une série de réflexions de Householder appliquées alternativement depuis la gauche et la droite. Ceci est connu sous le nom de bidiagonalisation de Golub-Kahan. Pour les matrices de grande taille, elles sont calculées de manière itérative par la méthode de Lanczos, appelée méthode de Golub-Kahan-Lanczos.

La bidiagonalisation a une structure très similaire à la décomposition en valeurs singulières (SVD). Cependant, elle est calculée dans le cadre d'opérations finies, alors que la SVD nécessite des schémas itératifs pour trouver les valeurs singulières. C'est parce que les valeurs singulières au carré sont les racines des polynômes caractéristiques de $\mathbf{A}^*\mathbf{A}$, où $\mathbf{A}$ est supposé être une matrice haute.

Dans la partie suivante, nous fournirons un algorithme basé sur la bidiagonalisation pour calculer la décomposition de la valeur singulière.

Chapitre 3

Cas séquentiel

3.1 Description de l'algorithme proposé

Nous supposons qu'on peut calcule implicitement le produit $\mathbf{A^T A}$ pour évaluer les valeurs singulières et les vecteurs singuliers de la matrice de départ A, ces valeurs ont une relation directe avec la décomposition des valeurs propres du produit $\mathbf{A^T A}$, mais malheureusement le calcul implicite de ce produit multipliera le conditionnement de la matrice, et donc nous aurons des résultats numériquement instables et le temps de calcul se multipliera.

Comme indiqué précédemment, nous commencerons par transformer la matrice initiale $\boldsymbol{A}$ en une matrice bidiagonale $\boldsymbol{B}$, puis nous utiliserons la décomposition des valeurs propres pour travailler avec la matrice bidigonale résultante.

Afin d'implémenter la SVD on utilise l'algorithme de Golub-Kahan-Lanczos pour bidiagonaliser une matrice quelconque, puis à exploiter le fait que le produit de la transposée d'une matrice bidiagonale donne une matrice symétrique tridiagonale. Il suffit ensuite de calculer la décomposition en valeurs propres de cette matrice pour extraire les valeurs singulières (la racine carrée des valeurs propres de la matrice tridiagonale) et les vecteurs singuliers correspondants (ceci est fait en accumulant les produits matriciels des matrices résultant de la bidiagonalisation et de la décomposition en valeurs propres de la matrice tridiagonale, des deux côtés).

Les routines de la bibliothèque LAPACKE, sont utilisées pour accomplir la décomposition en valeurs propres de la matrice tridiagonale dans cette technique.

3.2 Évaluation de performance théorique

L'algorithme de Golub-Kahan-Lanczos permet de bidiagonaliser une matrice d'entrée en un nombre limité d'itérations, le nombre d'itérations étant égal à la taille de la matrice.

Pour chaque itération de l'algorithme de Golub-Kahan-Lanczos, on peut remarquer que la globalité du calcul se fait dans la partie multiplication matrice-vecteur, donc si on ne considère pas cette phase, le reste de l'algorithme peut être fait en ***O(n)*** opérations arithmétiques, et la partie multiplication matrice-vecteur se fait en ***O(d∗n)*** avec d le nombre moyen d'éléments non nuls dans une ligne de la matrice, donc la complexité totale peut être estimée à ***O(d∗n∗m)***, donc cet algorithme peut être le plus adapté aux matrices creuses.

La transformation des parties dans la matrice tridiagonale correspondante prend ***O(n)***, en notant que le produit $\mathbf{B}^\mathbf{T}\mathbf{B}$ est une matrice tridiagonale symétrique avec $\mathbf{B}$ matrice bidiagonale. Par conséquent, nous allons diviser les deux parties : diagonale et sous-diagonale en deux tableaux, chacune avec une dimension.

La décomposition des valeurs propres de la matrice tridiagonale a été effectuée en utilisant la fonction LAPACKE dstevx qui est une méthode généralisée pour les problèmes de valeurs propres des matrices symétriques tridiagonales, dont la complexité n'est pas connue.

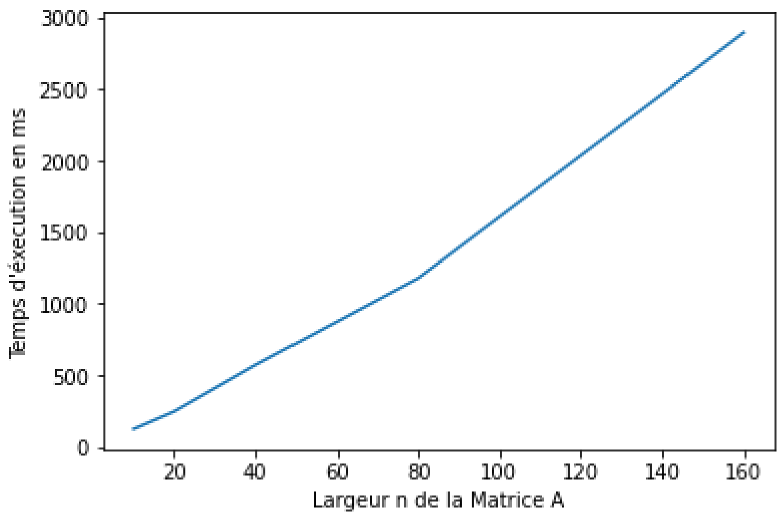
3.3 Évaluation de performance pratique

Afin d'évaluer les performances de l'algorithme SVD en utilisant l'algorithme Golub-Kahan-Lanczos, on varie à chaque fois les dimensions de la matrice $\mathbf{A}$ et on compare les temps d'exécution (ici on fixe ***m = 1000*** et on varie seulement ***n*** pour une représentation plus claire).

Pour avoir des résultats plus précis, on fait le même calcul pour un nombre bien déterminé de répétitions (ici 10) et on prend la moyenne.

les temps d'exécution ne prennent pas en compte la lecture la matrice A (initialisé aléatoirement ici), ni des opérations d'allocations de mémoire et des opérations d'affichage (qui peuvent être désactivées par l'utilisateur).

En outre, on prend aussi en compte le calcul des vecteurs singuliers dans le temps d'exécution, et pas seulement les valeurs singulières.



3.4 Conclusion

On peut remarquer que lorsque on double le taille ***m*n*** de la matrice A (on double n), on constate qu'on a un deboublement du temps d'exécution pour l'algorithme SVD avec la méthode Golub-Kahan-Lanczos. On peut conclure aussi la majorité de la hausse du temps d'exécution vient du calcul des valeurs et vecteurs propres par la fonction dstevx de LAPACKE, car la bidiagonalisation ne pose pas de problème grave vu qu'elle se fait en un nombre finit d'opération et l'algorithme de Golub-Kahan-Lanczos marche encore mieux avec les matrices larges.

Chapitre 4

Cas parallèle

4.1 Description

Dans ce chapitre, on aborde la partie parallèle de notre implémentation, qui consiste à introduire des directives OpenMP pour tirer parti des caractéristiques du modèle de mémoire partagée.

L'idée générale est de paralléliser les boucles de multiplication matrice-matrice ou matrice-vecteur, qui représentent la majorité du code, en particulier la section de bidiagonalisation.

4.2 Évaluation de performance théorique

En principe, l'algorithme de Golub-Kahan-Lanczos se parallélise bien, vu l'abondance des opérations de multiplication matricielle. Ces opérations peuvent être faites d'une manière parallèle et indépendante par plusieurs threads différents.

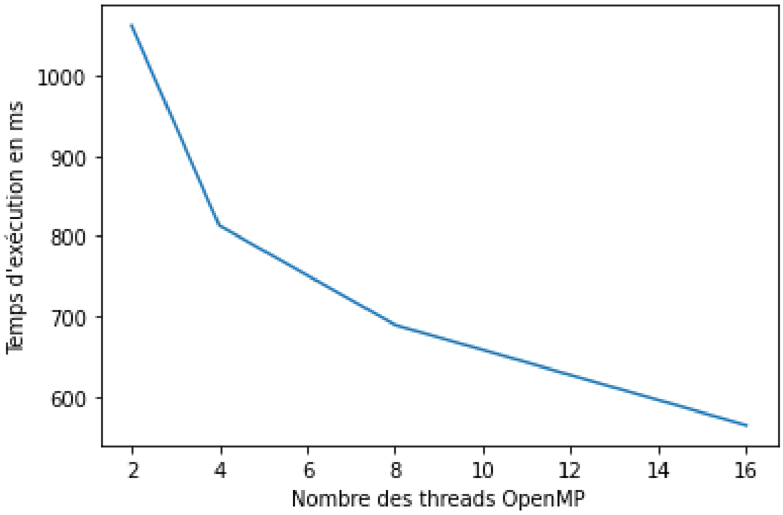
4.3 Architectures parallèles visées

Le modèle de programmation visé dans notre implémentation parallèle est la mémoire partagée avec Openmp, comme indiqué précédemment. L'objectif est de pouvoir répartir le travail entre les différents cœurs de traitement afin d'améliorer les performances de notre code.

4.4 Évaluation de performance pratique

Dans cette section, on essaye de mesurer l'effet de la parallélisation sur le temps d'exécution de la méthode SVD.

En Effet, on ne s'intéresse pas dans ce cas sur l'effet de la variation de la dimension de la matrice A (qu'on fixe ici à 1000*100), mais au nombre des threads OpenMP disponible par le programme.



4.5 Conclusion

On peut constater qu'on a une diminution du temps d'exécution, mais cette accélération est loin du décroissement linéaire de la durée en fonction du nombre des threads qu'on a pu imaginé. Ceci peut être du aux fonctions LAPACK qui prennent une grande partie du temps d'exécution et dont on a pas un accès direct à la parallélisation.

Conclusion générale

Nous avons d'abord créé une version séquentielle pour calculer la décomposition d'une matrice réelle rectangulaire en valeurs et vecteurs singuliers en utilisant la bidiagonalisation de Golub-Kahan-Lanczos, puis nous avons utilisé la décomposition spectrale de la matrice bidiagonale.

Nous avons utilisé les fonctions de la bibliothèque de Lapacke modifiées aux problèmes de valeurs propres symétriques pour la matrice tridiagonale symétrique liée à cette matrice bidigonale.

Les mesures de performance démontrent que nous devons utiliser des fonctions provenant de bibliothèques de calcul numérique standard qui ont été préalablement réglées mathématiquement et algorithmiquement.

Comme il existe de nombreux algorithmes pour la bidiagonalisation et la SVD en général, il faut choisir la meilleure approche pour chaque circonstance: pour les matrices énormes et creuses, les matrices avec un grand nombre de lignes par rapport au nombre de colonnes.