Titre du rapportRapport du TP2 du Projet de Calcul Scientifique et Analyse de données

Sajid Badr
Samran Fatima Zohra
Tyoubi Anass

Département Sciences du Numérique - Première année
2019-2020

Introduction

Auparavant, nous avons vu que pour réduire la dimension en utilisant l'analyse en composantes principales (ACP), nous n'avons pas besoin de toute la décomposition spectrale de la matrice de variance / covariance. En effet, nous n'avons besoin que des couples propres principales qui fournissent suffisamment d'informations sur les données.
Nous avons mis en œuvre la méthode de puissance, qui a été introduite dans le cours Calcul Scientifique, pour calculer le couple propre dominant. Comme il a été présenté lors des mêmes cours, il est possible d'ajouter un processus de déflation à cet algorithme, afin de calculer également les couples propres suivants.
Dans cette partie du projet, nous verrons que cet algorithme spécifique n'est pas efficace en termes de performances. Nous présenterons ensuite une méthode plus efficace appelée méthode d'itération subspace, basée sur un objet appelé quotient de Rayleigh. Nous étudierons quatre variantes de cette méthode.

Contents

1 Partie I : Limites de la méthode de la puissance itérée

Question 1 :
La durée de fonctionnement de la méthode deflated_power_method est bien plus longue que la durée de fonctionnement du sous-programme dysev. La méthode deflated_power_method n'est pas toujours la meilleure approche pour calculer les couples propres.

Question 2 :
La méthode de déflation utilise la méthode de la puissance itérée, qui a sa part dépend du rapport de la plus grande valeur propre et la deuxième plus grande valeur propre (Si les valeurs propres sont proches alors la convergence est sera lente). C'est pour ce la que la convergence est généralement lente.

Question 3 :
En appliquant l'algorithme sur une matrice au lieu d'un vecteur, on obtient une matrice dont les colonnes sont les même et qui sont égaux au vecteur propre associé à la plus grande valeur propre. De plus on obtient une matrice dont tous les composants convergent vers cette même valeur propre.

2 Partie II : Extension de la méthode de puissance pour calculer les sous-espaces propres associés aux valeurs propres dominants

2.1 subspace_iter_v0 : une méthode de base pour calculer les sous-espaces propres associés aux valeurs propres dominants

Question 4:
L'utilisation la décomposition spectrale de la matrice H (m*m) au lieu de la matrice A revient au faite que m ¡¡ n et que le but de cette décomposition est de trouver le m valeur propres de A. Cette méthode a pour but d'optimiser l'espace mémoire utilisé et donc améliorer la méthode de puissance itérée.

2.2 subspace_iter_v1 : version améliorée utilisant la projection de Raleigh-Ritz

Question 6 :
Voir iter_v1.f90

3 Partie III : subspace_iter_v2 and subspace_iter_v3 : vers un solveur efficace

3.1 Approche par blocs

Question 7 :
On peut montrer par récurrence que le coût en terme de flops du calcul de $A^p = (p-1) \cdot n^3$.
Pour la matrice $A^p \cdot V$ le coût en terme de flops du calcul est $(p-1) \cdot n^3 + n^2 \cdot m$.
Pour réduire le coût en terme de flops du calcul de $A^p \cdot V$ on calcule $Y = A \cdot V$ puis $V = Y$ et on la répète p fois. Comme A est de taille $n \times n$ et $A \cdot V$ et de taille $n \times m$, ce qui donne un coût égal à $p \cdot n^2 \cdot m$.
Avec cette méthode on aura besoin de moins d'espace de stockage.

3.2 Méthode de déflation

Question 9 :
On remarque une différence de précision entre la méthode subspace_iter_v1 et la méthode de l'algorithme 1. Cette différence est causé par le faite que dans cette méthode les vecteur Vc qui convergent dans le première itération subissent toujours au opération de l'algorithme mais pour l'algorithme 1 seul le vecteur qui change et une fois il converge on n'y touche plus.

Question 10 :
En utilisant cette deuxième méthode de déflation on remarque que les vecteur Vc qui convergent ne suibissent plus de changement.
On espère ainsi avoir la même précision de vecteur propre et donc résoudre le problème précédent.

4 Partie IV : Numérical expériments

Question 12 :
En augmentant la valeur de p, le nombre d'itérations de l'algorithme diminue de plus en plus. Ce qui nous permet de diminuer le temps et donc améliorer le programme.
Question 13 :
La figure montre la distribution des valeurs propres des 4 types de matrice (dimension de la matrice = 100).

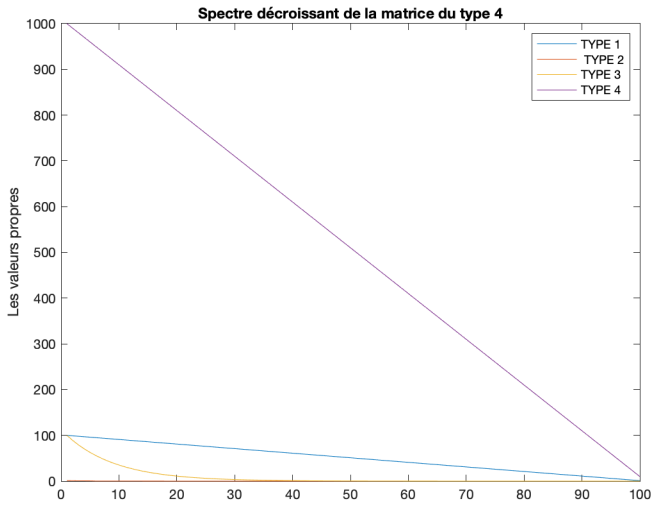

Figure 1: a distribution des valeurs propres de ces différents types

Question 14 :
Pour les 1er et 4ème types, le DSYEV est beaucoup plus rapide avec une grande matrice que les autres méthodes, la meilleure méthode dans ce cas était subspace_iter_v3, quant à lui, DSYEV a pris beaucoup moins, Pour les petites matrices, tous les méthodes ont un bon temps de fonctionnement, avec DSYEV est le meilleur suivi par le power_v11, puis les méthodes d'itérations subspace.
Pour les 2ème et 3ème types, il n'y a pas de grande différence avec une petite matrice, mais pour les plus grandes, les mêmes résultats que les types précédents pour le 3ème type; et pour le 2ème type les DSYEV, subspace_Iter_v3 et subspace_Iter_v2 sont les meilleurs.