

# **PROJET ANALYSE NUMÉRIQUE**

## **GM3**

### **MÉTHODES QR, TRANSFORMATION DE HOUSEHOLDER**

CLOUZEAU Emilie  
JEUNEHOMME Mona

Encadré par : Monsieur Caputo

Etablissement : Institut National des Sciences Appliquées de Rouen

Année scolaire : 2020/2021- Semestre 1

# Contents

<b>I</b>	<b>Présentation générale</b>	<b>3</b>
1	Principe de la méthode	3
2	Décomposition d'Householder	3
3	Calcul des vecteurs propres	5
<b>II</b>	<b>Analyses des résultats numériques</b>	<b>6</b>
1	Variance de p : Analyse de la matrice A	6
2	Variance de p : Analyse avec la matrice d'Hilbert de taille 4*4	7
3	Comparaison avec le programme sous Matlab	8

# Introduction

Dans ce projet, nous avons cherché à trouver les valeurs propres et les vecteurs propres d'une matrice symétrique. Pour cela, nous avons utilisé plusieurs fois le principe de la décomposition QR avec la méthode de Householder. Nous détaillerons cela dans une première partie. Nous avons ensuite codé cette méthode en fortran mais aussi en matlab. C'est pourquoi, nous comparerons les résultats obtenus à l'aide de ces deux langages dans une seconde partie en prenant comme exemple la matrice A ci-dessous et la matrice de Hilbert. Enfin, nous concluerons avec les avantages et les inconvénients d'une telle méthode suite à l'analyse de nos deux matrices tests.

$$A = \begin{pmatrix} 4 & 1 & -2 & 2 \\ 1 & 2 & 0 & 1 \\ -2 & 0 & 3 & -2 \\ 2 & 1 & -2 & -1 \end{pmatrix}$$

## Part I

# Présentation générale

## 1 Principe de la méthode

Dans toute la suite du rapport, quand nous nous référerons à la décomposition QR, celle-ci aura été établie à l'aide de la méthode de Householder. Soit  $A$  une matrice symétrique; si on applique le théorème spectral,  $A$  est diagonalisable puisqu'elle est symétrique (*ie: semblable à une matrice diagonale*<sup>1</sup> que l'on notera  $S_p$ ). Nous savons que  $S_p$  et  $A$  auront les mêmes valeurs propres. De plus, les valeurs propres de  $S_p$  se trouvent sur sa diagonale par définition d'une matrice diagonale. Aussi, nous allons commencer par chercher  $S_p$  à l'aide de la décomposition QR (cf § I.2.). Nous obtiendrons ainsi les valeurs propres de  $A$ . Enfin,  $S_p$  étant une matrice diagonale nous connaissons directement ses vecteurs propres qui sont ceux de la base canonique  $(\underline{e}_n)_{n \in N}$ . Aussi, par définition d'une valeur propre et de son vecteur propre associé,<sup>2</sup> il est facile d'obtenir les vecteurs propres de  $A$  (cf § I.3.).

## 2 Décomposition d'Householder

La décomposition QR est utilisée afin de résoudre des équations du type  $A\underline{x}=\underline{b}$ . Les conditions d'utilisation de cette méthode sont énoncées dans le théorème ci-dessous.

### **Théorème décomposition QR:**

Soit  $A$  une matrice inversible, alors il existe un unique couple de matrice  $Q$  et  $R$  avec  $Q$  une matrice orthogonale et  $R$  une matrice triangulaire supérieure tel que  $A$  égale  $QR$ .

Nous allons construire cette factorisation grâce à la matrice de transformation de Householder. L'idée de la méthode de Householder est de multiplier  $A$  par des matrices de Householder.

### **Définition matrice élémentaire de Householder:**

Soit  $\underline{v} \in \mathbb{R}^n$ , on définit la matrice élémentaire de Householder  $H(\underline{v})$  qui est symétrique et orthogonale avec  $\|\cdot\|$  la norme euclidienne par :

$$H(\underline{v}) = I - 2 \frac{\underline{v}\underline{v}^t}{\|\underline{v}\|^2} \quad (1)$$

On obtient  $H_1$  avec la formule précédente en ayant posé :

$$\underline{v} = \underline{a}_1 - \|\underline{a}_1\| \underline{e}_1, \quad \underline{a}_1 = \begin{pmatrix} a_{1,1} \\ a_{2,1} \\ a_{3,1} \\ a_{4,1} \\ \dots \\ a_{n,1} \end{pmatrix}$$

Pour obtenir  $H_2$ , on pose :

---

<sup>1</sup> $(A,B) \in (M_n(R))^2$  sont semblables si :  $\exists P \in GL_n(R), B = PAP^{-1}$

<sup>2</sup>Soit  $\lambda \in \mathbb{R}$  la valeur propre de  $A \in M_n(R)$  associée au vecteur propre  $X \in M_{n,1}(R) \setminus \{O_n\}$  alors  $AX = \lambda X$

$$\underline{v} = \underline{a_2} - \|\underline{a_2}\| \underline{e_2}, \quad \underline{a_2} = \begin{pmatrix} 0 \\ a_{2,2} \\ a_{3,2} \\ a_{4,2} \\ \dots \\ a_{n,2} \end{pmatrix}$$

On calcule alors  $H_2(\underline{v})$  et on obtient :

$$H_2 = \begin{pmatrix} 1 & . & . & . & . & 0 \\ 0 & & & & & \\ . & & & & & \\ . & & (h_2) & & & \\ . & & & & & \\ 0 & & & & & \end{pmatrix}, \text{ avec } h_2 = \begin{pmatrix} (*) \end{pmatrix} \text{ de taille } (n-1)X(n-1)$$

De manière générale, pour calculer  $H_i$ , on pose:

$$\underline{v} = \underline{a_i} - \|\underline{a_i}\| \underline{e_i}, \quad \underline{a_i} = \begin{pmatrix} 0 \\ \dots \\ 0 \\ a_{i,i} \\ \dots \\ a_{n,i} \end{pmatrix}$$

$H_i$  est alors de la forme suivante :

$$H_i = \begin{pmatrix} 1 & 0 & . & . & . & . & . & . & 0 \\ 0 & 1 & . & & & & & & . \\ . & . & . & . & & & (0) & & . \\ . & & . & . & . & & & & . \\ . & & & . & . & . & & & . \\ . & & & & . & . & 1 & 0 & \dots & 0 \\ . & & (0) & & & 0 & & & & \\ . & & & & & \dots & & (h_i) & & \\ 0 & . & . & . & . & 0 & & & & \end{pmatrix} \text{ avec } h_i = \begin{pmatrix} (*) \end{pmatrix} \text{ de taille } (n-i+1)X(n-i+1)$$

Ainsi, on multiplie A jusqu'à obtenir une matrice triangulaire supérieure  $R_1$  valant  $H_{n-1}H_{n-2} \dots H_1 A$ . Or, on sait que  $A = Q_1 R_1$  donc  $R_1 = Q_1^t A$ . Aussi on obtient que  $Q_1$  vaut  $H_1^t \dots H_{n-2}^t \dots H_{n-1}^t$

Une fois la décomposition QR obtenue, on cherche  $S_1$  tel que  $S_1 = Q_1^t A Q_1$ . Or comme  $A = Q_1 R_1$ , on a  $Q_1^t A Q_1 = Q_1^t Q_1 R_1 Q_1 = R_1 Q_1$ . Ainsi, on pose  $S_1 = R_1 Q_1$ .

On fait de nouveau une décomposition QR mais cette fois-ci sur  $S_1$  et on obtient  $S_2$  une matrice semblable à  $S_1$  qui vaut donc  $R_2 Q_2$ . On réitère ce procédé p fois jusqu'à obtenir  $S_p$  une matrice diagonale valant  $R_p Q_p$ . On obtient ainsi les valeurs propres de A.

### 3 Calcul des vecteurs propres

Nous allons expliquer comment calculer le vecteur propre  $X$  associé à la  $i$ ème valeur propre  $\lambda$  de  $A$ . Le même principe s'appliquera aux autres valeurs propres. On gardera les mêmes notations que dans la partie précédente. On rappelle que le vecteur propre associé à  $\lambda$  dans  $S_p$  est le vecteur colonne rempli de 0 et d'un 1 à la  $i$ ème position. On le note  $\underline{e}_i$ .

On a :

$$S_p \underline{e}_i = \lambda \underline{e}_i$$

Aussi, on peut réécrire cette équation de la façon suivante :

$$Q_p^t S_{p-1} Q_p \underline{e}_i = \lambda \underline{e}_i$$

En poursuivant ainsi, on obtient :

$$Q_p^t Q_{p-1}^t \dots Q_1^t A Q_1 \dots Q_{p-1} Q_p \underline{e}_i = \lambda \underline{e}_i$$

En posant  $Q_f^t = Q_p^t Q_{p-1}^t \dots Q_1^t$  on a alors  $A Q_f \underline{e}_i = \lambda Q_f \underline{e}_i$ . Aussi le vecteur propre associé à  $X$  est  $Q_f \underline{e}_i$ .

## Part II

# Analyses des résultats numériques

Dans cette partie, nous allons tester notre programme sur deux matrices, une matrice symétrique et donc bien conditionnée et une autre matrice, la matrice de Hilbert, que l'on notera B afin de ne pas interférer avec la matrice H de Householder vue précédemment. Cette dernière est réputée pour être très mal conditionnée, ce qui rend son usage très délicat en analyse numérique. Ainsi, nous allons pouvoir voir quel rôle joue le conditionnement de la matrice. Nous allons étudier le nombre d'itérations (p) nécessaires pour trouver les valeurs propres de nos matrices. Nous étudierons la différence entre  $AX_i$  et  $\lambda_i X_i$  où les  $\lambda_i$  sont les différentes valeurs propres de nos matrices trouvées par le programme et les  $X_i$  les vecteurs propres qui y sont associés ( $1 \leq i \leq 4$ ). On rappelle que  $S_p$  est la matrice équivalente à A après les décompositions QR successives effectuées. On récupère les valeurs propres de nos matrices sur la diagonale de  $S_p$  qui est une matrice diagonale pour p suffisamment grand.

## 1 Variance de p : Analyse de la matrice A

### Analyse des valeurs propres

Valeur de p	Valeurs propres ( $\lambda_i$ )	Commentaire sur S
5	6.8446089134147758 2.1758282886419096 -1.9929928845085991 0.97255568245190782	S n'est pas une matrice diagonale
50	6.8446211072349641 2.2631701755133475 -2.1921557465215926 1.0843644637732170	S n'est pas diagonale mais s'en rapproche
100	6.8446211072349641 2.2683083017115346 -2.1972938727198410 1.0843644637732222	S est diagonale si on considère que $10^{-2} \simeq 0$
500	6.8446211072349641 2.2685314064310602 -2.1975169774396703 1.0843644637732623	S est diagonale si on considère que $10^{-7} \simeq 0$
1000	6.8446211072349641 2.2685314064307520 -2.1975169774394891 1.0843644637732635	S est diagonale si on considère que $10^{-7} \simeq 0$
10000	”	”
100000	”	”

Table 1: Résultats en fonction de p pour la matrice A.

Nous remarquons alors qu'il faut environ 1000 itérations soit 1000 décompositions QR afin que  $S_n$  converge vers une matrice diagonale équivalente à A pour pouvoir trouver les valeurs propres de celle-ci. Il faut au minimum 500 itérations afin d'avoir une matrice  $S_p$  diagonale. Aussi, on remarque que cette méthode nécessite un grand nombre d'itérations pour obtenir les valeurs propres. Il est donc impossible d'appliquer cette méthode à la main. Il faut donc utiliser un ordinateur afin de la coder.

## Comparaison $AX_i$ et $\lambda_i X_i$

Dans notre programme, nous avons rajouté une partie afin de calculer  $AX_i$  et  $\lambda_i X_i$  pour connaître la différence de ces deux termes afin d'évaluer l'erreur. Nous avons pris la moyenne des quatre résultats correspondant à chacune des quatre valeurs propres. Nous avons laissé la partie du code permettant d'avoir ce résultat dans notre programme sous la forme de commentaires.

Valeur de p	Différence moyenne de $AX_i - \lambda_i X_i$
5	$9.238 \cdot 10^{-2}$
50	$2.379 \cdot 10^{-2}$
100	$4.820 \cdot 10^{-3}$
500	$-5.141 \cdot 10^{-9}$
1000	$-1.950 \cdot 10^{-8}$
10000	$-1.950 \cdot 10^{-8}$

Table 2: Comparaison  $AX_i - \lambda_i X_i$  en fonction de p

On remarque ainsi, comme précédemment, que la limite est atteinte pour  $p=1000$ . Cependant, on constate que pour  $p=500$  la différence moyenne de  $AX_i - \lambda_i X_i$  est plus faible ( $10^{-9}$  contre  $10^{-8}$ ). Nous ne savons pas expliquer ce résultat.

## 2 Variance de p : Analyse avec la matrice d'Hilbert de taille 4\*4

Nous avons choisi une matrice de Hilbert de taille 4\*4 afin de pouvoir mieux la comparer avec la matrice A qui est de la même taille. Pour rappel, nous notons cette matrice B.

### Analyse des valeurs propres

Valeur de P	Valeurs propres ( $\lambda_i$ )	Commentaire sur S
5	1.5002142891299788 0.16914122212901175 $6.7382798753769018 \cdot 10^{-3}$ $9.6704356661252589 \cdot 10^{-5}$	S est diagonale si on considère $0 \simeq 10^{-5}$
10	1.5002142893682127 0.16914122189077993 $6.7382798753745409 \cdot 10^{-3}$ $9.6704356661252589 \cdot 10^{-5}$	S est diagonale si on considère $0 \simeq 10^{-8}$
50	”	”
100	”	”
1000	”	”
10000	”	”
100000	”	”

Table 3: Résultats en fonction de p pour la matrice de Hilbert (4\*4)



Contrairement à la matrice A, la matrice d'Hilbert atteint rapidement une stagnation des valeurs à partir de 10 itérations. Au dessus de 10, l'algorithme trouve toujours les mêmes valeurs propres. Ce résultat nous a surprises. En effet, tout laissait à penser qu'il nous serait plus difficile de trouver les valeurs propres de la matrice d'Hilbert du fait de son mauvais conditionnement alors que seulement 10 itérations ont été nécessaires contre 1000 pour la matrice A. Aussi, nous avons vérifié si le programme renvoyait bien les bonnes valeurs propres, ce qui est le cas.

### Comparaison $BX_i$ et $\lambda_i X_i$

Valeur de p	Différence moyenne de $BX_i - \lambda_i X_i$
5	$2.930 \cdot 10^{-6}$
10	$4.193 \cdot 10^{-9}$
50	$4.193 \cdot 10^{-9}$
100	"
1000	"

Table 4: Comparaison  $BX_i - \lambda_i X_i$  en fonction de p

On remarque que très peu d'itérations sont nécessaires pour être proche de la solution réelle. On voit que dès 10 itérations on obtient la meilleure solution. Cela est en accord avec le résultat précédent.

## 3 Comparaison avec le programme sous Matlab

Nous avons aussi codé le programme sous Matlab afin de pouvoir comparer nos résultats.

	FORTRAN	MATLAB
Nombre de chiffres significatifs renvoyés pour $\lambda$	17	4
Nombre d'itérations p nécessaires pour les valeurs propres de A	1000	200
Nombre de chiffres significatifs renvoyés pour X	17	6
Nombre d'itérations p nécessaires pour les vecteurs propres de A	500	400
Nombre d'itérations p nécessaires pour les valeurs propres de B	5	5
Nombre d'itérations p nécessaires pour les vecteurs propres de B	10	10

Table 5: Les différences entre Fortran et Matlab

On remarque donc que, pour une plus grande précision, il est préférable de coder cette méthode en Fortran. En effet, Matlab nécessite moins d'itérations pour converger mais la précision est plus faible. Cependant, le langage choisi n'a aucune incidence pour la matrice de Hilbert comme elle converge rapidement.

# Conclusion

Nous avons découvert une méthode afin de calculer les valeurs propres et les vecteurs propres d'une matrice symétrique. En effet, nous n'avions pas vu cela en Analyse Numérique. En s'appuyant sur la décomposition QR Householder, nous avons pu appliquer les notions que nous avons vu en cours sur un exemple concret et le coder.

De plus, cette méthode est très efficace pour la matrice de Hilbert ce qui peut être suprenant étant donné son mauvais conditionnement. On retiendra que cette méthode permet de trouver toutes les valeurs propres. Seulement, elle nécessite un grand nombre d'itérations pour une matrice symétrique autre que la matrice de Hilbert.

Pour finir, ce projet a été très enrichissant pour nous. Il nous a permis de nous améliorer et de mieux manipuler les langages Matlab et Fortran mais aussi de mieux appréhender les avantages et les inconvénients de chacun. Nous avons pu remarquer qu'avec Matlab la matrice identité (eye) et la matrice de Hilbert (hilb) sont déjà codées. De plus, la déclaration des variables et l'allocation dynamique ne sont pas nécessaires sous Matlab. Enfin, hormis ces quelques différences, les langages Fortran et Matlab sont assez similaires. On constatera tout de même que Fortran apporte une solution plus précise.

# Bibliographie

- [1] Cours d'Analyse Numérique 2020 de M<sup>lle</sup> Kazakova
- [2] <http://www.ousmanethiare.com/images/cours/chapitre5ingcal.pdf>
- [3] <https://perso.univ-rennes1.fr/eric.darrigrand-lacarrieu/Teaching/PdfFiles/PolyF04cours.pdf>
- [4] <http://library.navoiy-uni.uz/files/A%20Remark%20on%20Hilbert%E2%80%99s%20Matrix.pdf>
- [5] [www-ia.lip6.fr/~hnguyen/coursan/cours5\\_imprimable.pdf](http://www-ia.lip6.fr/~hnguyen/coursan/cours5_imprimable.pdf)