14 Transformation sous forme tridiagonale (ou de Hessenberg)

Avec la transformation $v = Pu$ (où est P une matrice inversible) le problème

$$Av = \lambda v$$

devient

$$P^{-1}APu = \lambda u$$

Donc, les valeurs propres de A et de $P^{-1}AP$ sont les mêmes et les vecteurs propres v_i de A se transforment par $v_i = Pu_i$. Le but de ce paragraphe est de trouver une matrice P telle que $P^{-1}AP$ devienne "plus simple". La situation idéale serait trouvée si $P^{-1}AP$ devenait diagonale ou triangulaire - mais une telle transformation nécessiterait déjà la connaissance des valeurs propres. Alors, on cherche P tel que $P^{-1}AP$ soit sous forme de Hessenberg

$$P^{-1}AP = H = \begin{pmatrix} * & * & \cdots & \cdots & * \\ * & * & \ddots & & \vdots \\ & * & \ddots & \ddots & * \\ & & \ddots & \ddots & * \\ & & & * & * \end{pmatrix} \tag{14.1}$$

c'est-à-dire, $h_{ij} = 0$ pour $i > j + 1$. Pour arriver à ce but, nous considérons deux algorithmes.

14.1 a) A l'aide des transformations élémentaires

Comme pour l'élimination de Gauss, nous utilisons les transformations pour faire apparaitre les zéros - colonne par colonne - dans (17.1). Dans un premier pas, nous choisissons $k \geq 2$ tel que $|a_{k1}| \geq |a_{j1}|$ pour $j \geq 2$ et nous permutons les lignes 2 et k, c'est-à-dire, nous formons PA o ù P est une matrice de permutation convenable. Pour ne pas changer les valeurs propres, il faut également permuter les colonnes 2 et k (ceci correspond au calcul de $A' = PAP^{-1}$ car $P^2 = I$ (et donc $P = P^{-1}$). Si $a'_{21} = 0$, on a aussi $a'_{i1} = 0$ pour $i \geq 3$ et le premier pas est terminé. Sinon, nous déterminons

$$L_2 = \begin{pmatrix} 1 & & & & \\ 0 & 1 & & & \\ 0 & -l_{32} & 1 & & \\ \vdots & \vdots & \ddots & \ddots & \\ 0 & -l_{n2} & \cdots & 0 & 1 \end{pmatrix} \quad \text{telle que} \quad L_2A' = \begin{pmatrix} a'_{11} & a'_{12} & \cdots & a'_{1n} \\ a'_{21} & a'_{22} & \cdots & a'_{2n} \\ 0 & * & \cdots & * \\ \vdots & \vdots & \ddots & \vdots \\ 0 & * & \cdots & * \end{pmatrix}$$

Pour ceci, on définit $l_{i2} = \frac{a'_{i1}}{a'_{21}}$. Une multiplication à droite avec

$$L_2^{-1} = \begin{pmatrix} 1 & & & & \\ 0 & 1 & & & \\ 0 & l_{32} & 1 & & \\ \vdots & \vdots & \ddots & \ddots & \\ 0 & l_{n2} & \cdots & 0 & 1 \end{pmatrix}$$

ne change pas la première colonne de L_2A'. On répète la même procédure avec la sous-matrice de $L_2A'L_2^{-1}$ de dimension $n-1$, et ainsi de suite. A cause des multiplications à droite avec L_i^{-1}, cet algorithme coûte deux fois plus cher que l'élimination de Gauss. Pour la matrice

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

on prend

$$L_2 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -1/2 & 1 \end{pmatrix}$$

et on obtient

$$L_2 A = \begin{pmatrix} 3 & 2 & 1 \\ 2 & 1 & 3 \\ 0 & 5/2 & -1/2 \end{pmatrix}, \quad \text{puis} \quad L_2 A L_2^{-1} = \begin{pmatrix} 3 & 5/2 & 1 \\ 2 & 5/2 & 3 \\ 0 & 9/4 & -1/2 \end{pmatrix} = H$$

Cet exemple montre un désavantage de cet algorithme : si l'on part avec une matrice symétrique A, la matrice de Hessenberg H, obtenue par cet algorithme, n'est plus symétrique en général.

14.2 b) A l'aide des transformations orthogonales

Il est souvent préférable de travailler avec des réflexions de Householder Commençons par une réflexion pour les coordonnées $2,...,n$ laissant fixe la première coordonnée :$\bar{Q}_2 = I - 2\bar{u}_2\bar{u}_2^T$ ($\|\bar{u}_2\|_2 = 1$) tel que $\bar{Q}_2\bar{A}_1 = \alpha_2 e_1$ où $\bar{A}_1 = (a_{21},...,a_{n1})$. En posant $u_2 = (0,\bar{u}_2)^T$ et $Q_2 = I - 2u_2u_2^T$, la matrice Q_2A contient des zéros dans la première colonne à partir du troisième élément. La multiplication à droite avec $Q_2^{-1} = Q_2^T = Q_2$ ne change pas cette colonne :

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \xrightarrow{Q_2A} \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ \alpha_2 & * & * \\ 0 & * & * \end{pmatrix} \xrightarrow{Q_2AQ_2} \begin{pmatrix} a_{11} & * & * \\ \alpha_2 & * & * \\ 0 & * & * \end{pmatrix}$$

Dans le pas suivant, on applique la même procédure à la sous-matrice de dimension $n-1$, etc. Finalement, on arrive à la forme de Hessenberg (17.1) avec la tranformation $P^{-1} = Q_{n-1}....Q_2$ qui est une matrice orthogonale (c'est-à-dire, $P^{-1} = P^T$). Nous avons un double avantage avec cet algorithme :

— il ne faut pas faire une recherche de pivot ;
— si A est symétrique, alors $P^{-1}AP$ est aussi symétrique, et donc tridiagonale.

14.3 Méthode de bissection pour des matrices tridiagonales

Considérons une matrice symétrique tridiagonale

$$A = \begin{pmatrix} d_1 & e_2 & & & \\ e_2 & d_2 & e_3 & & \\ & e_3 & \ddots & \ddots & \\ & & \ddots & \ddots & e_n \\ & & & e_n & d_n \end{pmatrix}$$

On observe tout d'abord que si un élément e_i est nul, la matrice A est déjà décomposée en deux sous-matrices du m ême type, qui ensemble fournissent les valeurs propres de A . On peut donc supposer, sans restreindre la généralité, que

$$c_i \neq 0 \quad \text{pour} \quad i = 2,...,n. \tag{14.2}$$

Pour cette matrice, il est possible de calculer la valeur $P_A(\lambda)$ du polynôme caractéristique sans connaitre ses coefficients. En effet, si l'on pose

$$A_1 = (d_1), \qquad A_2 = \begin{pmatrix} d_1 & e_2 \\ e_2 & d_2 \end{pmatrix}, \qquad A_3 = \begin{pmatrix} d_1 & e_2 & \\ e_2 & d_2 & e_3 \\ & e_3 & d_3 \end{pmatrix}, \qquad ...$$

et si l'on définit

$$p_i(\lambda) = \det(A_i - \lambda I),$$

on obtient

$$\begin{aligned} p_0(\lambda) &= 1 \\ p_1(\lambda) &= d_1 - \lambda \\ p_i(\lambda) &= (d_i - \lambda)p_{i-1}(\lambda) - e_i^2 p_{i-2}(\lambda), \qquad i = 2, ..., n. \end{aligned} \tag{14.3}$$

La formule de récurrence dans (17.3) est obtenue en dé veloppant le déterminant de la matrice $A_i - \lambda I$ par rapport à la dernière ligne (ou colonne). En principe, on peut maintenant calculer les valeurs propres de A (c'est- à-dire. les zéros de $p_n(\lambda)$ de la manière suivante : chercher un intervalle où $p_n(\lambda)$ change de signe et localiser une racine de $p_n(\lambda) = 0$ par bissection. Les évaluations de $p_n(\lambda)$ sont faites à l'aide de la formule (17.3). Mais il existe une astuce interessante qui permet d'améliorer cet algorithme.

Théorème 248. Si l'équation (17.2) est vérifié, les polyn ômes $p_i(\lambda)$ définis par (17.3) satisfont

a) $p_n'(\hat{\lambda})p_{n-1}(\hat{\lambda}) < 0$ si $p_n(\hat{\lambda}) = 0$ $(\hat{\lambda} \in \mathbb{R})$

b) $p_{i-1}(\hat{\lambda})p_{i+1}(\hat{\lambda}) < 0$ si $p_i(\hat{\lambda}) = 0$ pour un $\{i \in 1, 2, ..., n-1\}$

c) $p_0(\lambda)$ ne change pas de signe sur $\mathbb{R}$.

Démonstration. L'affirmation (c) est triviale. Si $p_i(\hat{\lambda}) = 0$ pour un $\{i \in 1, 2, ..., n-1\}$, la formule de récurrence (17.3) donne l'inégalité $p_{i-1}(\hat{\lambda})p_{i+1}(\hat{\lambda}) \leq 0$. Pour démontrer (b), il suffit d'exclure le cas $p_{i-1}(\hat{\lambda})p_{i+1}(\hat{\lambda}) = 0$. Si deux valeurs consécutives de la suite $\left\{p_i(\hat{\lambda})\right\}$ sont nulles, la formule de récurrence montre que $p_i(\hat{\lambda}) = 0$ pour tout i, ce qui contredit $p_0(\lambda) = 1$. Nous démontrons par récurrence que toutes les racines de $p_i(\lambda)$ sont réelles, simples et séparées par celles de $p_{i-1}(\lambda)$. Il n'y a rien à démontrer pour $i = 1$. Supposons la propriété vraie pour i et montrons qu'elle est encore vraie pour $i+1$. Comme les z éros $\lambda_1 < \lambda_2 < ... < \lambda_i$ sont séparés par ceux de $p_{i-1}(\lambda)$ et comme $p_{i-1}(-\infty) = +\infty$, nous avons $sign\ p_{i-1}(\lambda_j) = (-1)^{j+1}$. Alors, on d éduit de (b) que $sign\ p_{i+1}(\lambda_j) = (-1)^j$. Ceci et le fait que $p_{i+1}(\lambda) = (-1)^{j+1}\lambda^{i+1} + ...$ montrent que $p_{i+1}(\lambda)$ possède un zéro ré el dans chacun des intervalles ouverts $(-\infty, \lambda_1), (\lambda_1, \lambda_2), ..., (\lambda_i, \infty)$ L'affirmation (a) est maintenant une conséquence de (b) et du fait que toutes les racines de $p_{i-1}(\lambda)$ sont réelles simples ; cqfd

Définition 249 (suite de Sturm). Une suite $\{p_0, p_1, ..., p_n\}$ de polynômes à coefficients réels s'appelle une suite de Sturm, si elle vérifie les conditions (a), (b), (c) du Théorème (275)

Considérons une suite de Sturm $\{p_0, p_1, ..., p_n\}$. Si l'on définit

$$\omega(\lambda) = \text{nombre de changements de signes de } \{p_0(\lambda), p_1(\lambda), ..., p_n(\lambda)\}$$

alors le polynôme $p_n(\lambda)$ possède exactement

$$\omega(b) - \omega(a)$$

zéros dans l'intervalle $[a, b]$ (si $p_i(\lambda) = 0$, on définit $signp_i(\lambda) = sign\ p_{i-1}(\lambda)$).

Démonstration. Par continuité, l'entier $\omega(\lambda)$ peut changer sa valeur seulement si une valeur des fonctions $p_i(\lambda)$ devient nulle. La fonction $p_0(\lambda)$ ne change pas de signe. Supposons alors que $p_i(\tilde{\lambda}) = 0$ pour un $i \in \{1, 2, ..., n-1\}$. La condition (b) et la continuité de $p_j(\lambda)$ montrent que seulement les deux situations suivantes sont possibles (ε petit) :

	$\tilde{\lambda} - \varepsilon$	$\tilde{\lambda}$	$\tilde{\lambda} + \varepsilon$
$p_{i-1}(\lambda)$	+	+	+
$p_i(\lambda)$	±	0	±
$p_{i+1}(\lambda)$	−	−	−

	$\tilde{\lambda} - \varepsilon$	$\tilde{\lambda}$	$\tilde{\lambda} + \varepsilon$
$p_{i-1}(\lambda)$	−	−	−
$p_i(\lambda)$	±	0	±
$p_{i+1}(\lambda)$	+	+	+

cqfd

Chaque fois, on a $\omega(\tilde{\lambda}+\varepsilon) = \omega(\tilde{\lambda}) = \omega(\tilde{\lambda}-\varepsilon)$ et la valeur de $\omega(\lambda)$ ne change pas si λ traverse un zéro de $p_i(\lambda)$ pour $i \in \{1,2,...,n-1\}$ Il reste à étudier la fonction $\omega(\lambda)$ dans un voisinage d'un zéro $\tilde{\lambda}$ de $p_n(\lambda)$. La propriété (a) implique que pour les signes de $p_j(\lambda)$ on a seulement les deux possibilités suivantes :

	$\tilde{\lambda}-\varepsilon$	$\tilde{\lambda}$	$\tilde{\lambda}+\varepsilon$
$p_{n-1}(\lambda)$	+	+	+
$p_n(\lambda)$	+	0	−

	$\tilde{\lambda}-\varepsilon$	$\tilde{\lambda}$	$\tilde{\lambda}+\varepsilon$
$p_{n-1}(\lambda)$	−	−	−
$p_n(\lambda)$	−	0	+

c'est-à-dire, $\omega(\tilde{\lambda}+\varepsilon) = \omega(\tilde{\lambda}-\varepsilon)+1$. Ceci démontre que la fonction $\omega(\lambda)$ est constante par morceaux et augmente de 1 sa valeur si λ traverse un zéro de $p_n(\lambda)$.

14.4 Méthode de bissection.

Si l'on applique ce théorème à la suite (17.3), la diff érence $\omega(b)-\omega(a)$ est égale au nombre de valeurs propres de (17.2) dans l'intervalle $[a,b]$. On obtient toutes les valeurs propres de A de la manière suivante :

— on cherche un intervalle $[a,b]$ qui contienne toutes les valeurs propres de A (par exemple, en appliquant le théorème de Gershgorin). On a donc que $\omega(a) = 0$ et $\omega(b) = n$.

— on pose $c = \frac{(a+b)}{2}$ et on calcule $\omega(c)$. Les différences $\omega(c)-\omega(a)$ et $\omega(b)-\omega(c)$ indiquent combien de valeurs propres de A sont dans $[a,c)$ et combien sont dans $[c,b)$

— on continue à diviser les intervalles qui contiennent au moins une valeur propre de A.

On peut facilement modifier cet algorithme pour calculer la valeur propre la plus petite ou la $3^{ème}$ plus grande valeur propre, etc. Pour éviter un "overflow" dans le calcul de $p_n(\lambda)$ (si n et λ sont grands), il vaut mieux travailler avec

$$f_i(\lambda) = \frac{p_i(\lambda)}{p_{i-1}(\lambda)} \qquad i = 1,2,...,n$$

et utiliser le fait que

$$\omega(\lambda) = \text{nombre d'éléments négatifs parmi } \{f_1(\lambda), f_2(\lambda), ..., f_n(\lambda)\}$$

(attention : si $p_{i-1}(\lambda)$ est zéro, on pose $f_i(\lambda) = -\infty$; cette valeur compte pour un élément négatif). Pour une programmation de l'algorithme, on utilise la récurrence

$$\begin{aligned} f_1(\lambda) &= d_1 - \lambda \\ f_i(\lambda) &= d_i - \lambda - \begin{cases} e_i^2/f_{i-1}(\lambda) & \text{si} \quad f_{i-1}(\lambda) \neq 0 \\ |e_i|/eps & \text{si} \quad f_{i-1}(\lambda) = 0 \end{cases} \end{aligned}$$

La formule pour le cas $f_{i-1}(\lambda) \neq 0$ est une conséquence de (17.3). Si $f_{i-1}(\lambda) = 0$ (c'est-à-dire $p_{i-1}(\lambda) = 0$), on remplace cette valeur par $|e_i|.eps$. Ceci correspond à ajouter la perturbation $|e_i|.eps$ à d_{i-1}

15 L'itération orthogonale

Dans ce paragraphe, nous allons généraliser la méthode de la puissance afin de pouvoir calculer les deux (trois,) valeurs propres dominantes en même temps. Cette généralisation motivera l'ité ration QR qui constitue l'algorithme le plus important pour le calcul des valeurs propres d'une matrice.

15.1 Généralisation de la méthode de la puissance (pour calculer les deux valeurs propres dominantes).

Considérons une matrice A dont les valeurs propres satisfont

$$|\lambda_1| > |\lambda_2| > ... > |\lambda_n|. \tag{15.1}$$

La méthode de la puissance est basée sur l'itération $y_{k+1} = Ay_k$ et nous permet d'obtenir une approximation de λ_1 à l'aide du quotient de Rayleigh. Pour calculer (en même temps) la deuxième valeur propre λ_2, nous prenons deux vecteurs y_0 et z_0 satisfaisant $y_0^* z_0 = 0$ et nous considérons l'itération

$$\begin{aligned} y_{k+1} &= Ay_k \\ z_{k+1} &= Az_k - \beta_{k+1} y_{k+1} \end{aligned} \tag{15.2}$$

où β_{k+1} est déterminé par la condition $y_{k+1}^* z_{k+1} = 0$. Par induction, on voit que

$$\begin{aligned} y_k &= A^k y_0 \\ z_k &= A^k z_0 - \gamma_k y_k \end{aligned}$$

où γ_k est tel que

$$y_k^* z_k = 0 \tag{15.3}$$

Ceci signifie que le calcul de $\{z_k\}$ correspond à la m éthode de la puissance appliquée à z_0, combinée avec une orthogonalisation (projection de $A^k z_0$ sur le complément orthogonal de y_k). En exprimant les vecteurs initiaux dans la base de vecteurs propres $v_1, v_2, ..., v_n$ de la matrice A (on suppose $\|v_i\|_2 = 1$),

$$y_0 = \sum_{i=1}^{n} a_i v_i, \qquad z_0 = \sum_{i=1}^{n} b_i v_i, \tag{15.4}$$

les vecteurs y_k, z_k deviennent

$$y_k = \sum_{i=1}^{n} a_i \lambda_i^k v_i, \qquad z_0 = \sum_{i=1}^{n} (b_i - \gamma_k a_i)\, \lambda_i^k v_i,$$

Comme nous l'avons constaté précédement, pour $k \to \infty$, le terme $a_1 \lambda_1^k v_1$ est dominant dans y_k (si $a_1 \neq 0$) et on obtient une approximation du premier vecteur propre v_1 . Que peut-on dire pour la suite $\{z_k\}$? La condition (18.3) d'orthogonalité implique que

$$\sum_{i=1}^{n} \sum_{i=1}^{n} a_i \left(b_j - \gamma_k a_j\right) \bar{\lambda}_i^k \lambda_j^k v_i^* v_j = 0 \tag{15.5}$$

Cette relation définit γ_k. Comme le terme avec $i = j = 1$ est dominant, on voit que $\gamma_k \approx b_1/a_1$ Par la suite, nous allons supposer que $a_1 \neq 0$ et $a_1 b_2 - a_2 b_1 \neq 0$. En divisant (18.5) par $\bar{\lambda}_1^k$ on obtient

$$\bar{a}_1(b_1 - \gamma_k a_1)\lambda_1^k(1 + O(|\lambda_2/\lambda_1|^k)) = -\bar{a}_1(b_2 - \gamma_k a_2)\lambda_2^k(v_1^* v_2 + O(|\lambda_2/\lambda_1|^k) + O(|\lambda_3/\lambda_2|^k)).$$

Maintenant, on peut insérer cette formule dans (18.4) et on en d éduit

$$z_k = \lambda_{21}^k (b_2 - \gamma_k a_2)(v_2 - v_1^* v_2 . v_1 + O(|\lambda_2/\lambda_1|^k) + O(|\lambda_3/\lambda_2|^k)) \tag{15.6}$$

Visiblement, le vecteur z_k s'approche (pour $k \to \infty$) d'un multiple de $v_2 - v_1^* v_2 . v_1$, qui est la projection orthogonale de v_2 à l'hyperplan $v_1^{\perp}$. Concernant les valeurs propres, on a le résultat suivant.

Théorème 250. Considérons les vecteurs y_k, z_k donnés par (18.2) et notons

$$U_k = \left(y_k / \left\| y_k \right\|_2 , z_k / \| z_k \|_2 \right) \tag{15.7}$$

(observer que $U_k^* U_k = 1$). Si (18.1) est vérifié , on a que

$$U_k^* A U_k \longrightarrow \begin{pmatrix} \lambda_1 & * \\ 0 & \lambda_2 \end{pmatrix} \qquad \text{pour} \qquad k \longrightarrow \infty \tag{15.8}$$

Démonstration. L'élément (1,1) de la matrice $U_k^* A U_k$ est le quotient de Rayleigh (12.3) qui converge vers λ_1 . En utilisant (18.6), on voit que l'élément (2,2) satisfait

$$\frac{z_k^* A z_k}{z_k^* z_k} \longrightarrow \frac{\left(v_2 - v_1^* v_2 . v_1 \right)^* \left(\lambda_2 v_2 - \lambda_1 v_1^* v_2 . v_1 \right)}{\left(v_2 - v_1^* v_2 . v_1 \right)^* \left(v_2 - v_1^* v_2 . v_1 \right)} = \frac{\lambda_2 (1 - \left| v_1^* v_2 \right|^2)}{1 - \left| v_1^* v_2 \right|^2} = \lambda_2$$

De façon similaire, on obtient pour l'élément (2,1)

$$\frac{z_k^* A y_k}{\| z_k \|_2 \left\| y_k \right\|_2} \longrightarrow \frac{\left(v_2 - v_1^* v_2 . v_1 \right)^* \lambda_1 v_1}{\left\| v_2 - v_1^* v_2 . v_1 \right\|_2 \| v_1 \|_2} = 0$$

Finalement, l'élément (1,2) de $U_k^* A U_k$ satisfait

$$\frac{y_k^* A z_k}{\left\| y_k \right\|_2 \| z_k \|_2} \longrightarrow \frac{v_1^* (\lambda_2 v_2 - \lambda_1 v_1^* v_2 . v_1)}{\| v_1 \|_2 \left\| v_2 - v_1^* v_2 . v_1 \right\|_2} = \frac{(\lambda_2 - \lambda_1) v_1^* v_2}{\sqrt{1 - \left| v_1^* v_2 \right|^2}}.$$

Cette expression est en général non nulle. cqfd

Remarque 251. Avec la notation (18.7), l'itération (18.2) peut être écrite sous la forme

$$A U_k = U_{k+1} R_{k+1}$$

où R_{k+1} est une matrice 2×2 qui est triangulaire supé rieure.

15.2 Méthode de la puissance (pour le calcul de toutes les valeurs propres)

ou simplement *itération orthogonale*. La généralisation de l'algorithme précédent au cas où l'on veut calculer toutes les á valeurs propres d'une matrice est évidente : on choisit une matrice orthogonale U_0, c'est-à-dire, on choisit n vecteurs orthogonaux (les colonnes de U_0) qui jouent le rôle de y_0, z_0, etc. Puis, on effectue l'itération

```
for k=1,2,..
    Z_{k}=AU_{k+1}          (d\U{e9}composition QR)
    U_{k}R_{k}=Z_{k}
end
```

Si (18.1) est vérifié et si la matrice U_0 est bien choisie ($a_1 \neq 0, a_1 b_2 - a_2 b_1 \neq 0$, , etc), une géné ralisation du théorème précédent donne la convergence

$$T_k = U_k^* A U_k \tag{15.9}$$

vers une matrice triangulaire dont les éléments de la diagonale sont les valeurs propres de A . On a donc transformé A en forme triangulaire à l'aide d'une matrice orthogonale (*décomposition de Schur*). Il y a une possibilité intéressante pour calculer T_k de (18.9) directement à partir de T_{k-1}. D'une part, on déduit de (**??**) que

$$T_{k-1} = U_k^* A U_k = \left(U_{k-1}^* U_k \right) R_k \tag{15.10}$$

D'autre part, on a

$$T_k = U_k^* A U_k = U_k^* A U_{k-1}^* U_k = R_k \left(U_{k-1}^* U_k \right).$$

On calcule la décomposition QR de la matrice T_{k-1} et on é change les deux matrices de cette décomposition pour obtenir T_k

15.3 L' algorithme QR

La méthode QR, due à J.C.F. Francis et à V.N. Kublanovskaya, est la méthode la plus couram- ment utilisée pour le calcul de l'ensemble des valeurs propres (P.G. Ciarlet 1982) the QR iteration, and it forms the backbone of the most effective algorithm for computing the Schur decomposition. (G.H. Golub & C.F. van Loan 1989) La version simple du célèbre algorithme QR n'est rien d'autre que la méthode du paragraphe précédent. En effet, si l'on pose $Q_k = U_{k-1}^* U_k$ et si l'on commence l'itération avec $U_0 = I$, les formules (18.9) et (18.10) nous permettent d' écrire l'algorithme précédent comme suit : (décomposition QR)

```
T_{0}=A
for  k=1,2,..
    Q_{k}R_{k}=T_{k-1}
    T_{k}=R_{k}Q_{k}
end
```

Les T_k qui sont les mêmes que dans le paragraphe V.5, convergent (en général) vers une matrice triangulaire. Ceci nous permet d'obtenir toutes les valeurs propres de la matrice A car les T_k ont les mêmes valeurs propres que A (voir (18.9)). Cet algorithme important a été développé indépendamment par J.G.F. Francis (1961) et par V.N. Kublanovskaya (1961). Un algorithme similaire, qui utilise la décomposition LR à la place de la dé composition QR, a été introduit par H. Rutishauser (1958).

Exemple 252. Appliquons la méthode QR à la matrice

$$A = \begin{pmatrix} 10 & 2 & 3 & 5 \\ 3 & 6 & 8 & 4 \\ 0 & 5 & 4 & 3 \\ 0 & 0 & 4 & 3 \end{pmatrix}$$

On peut montrer que, pour une matrice de Hessenberg A , toutes les matrices T_k sont aussi sous forme de Hessenberg. Pour 'étudier la convergence vers une matrice triangulaire, il suffit alors de considérer les éléments $t_{i+1,i}^{(k)} (i = 1, 2, ..., n-1)$ de la sous-diagonale. On constate que

$$\frac{t_{i+1,i}^{(k+1)}}{t_{i+1,i}^{(k)}} \approx \frac{\lambda_{i+1}}{\lambda_i} \tag{15.11}$$

$(\lambda_1 \approx 14,3, \lambda_2 \approx 7,86, \lambda_3 \approx 2,70, \lambda_4 \approx -1,86)$. Comme, les éléments $t_{i+1,i}^{(k)}$ convergent, pour $k \longrightarrow \infty$, liné airement vers 0 (voir la figure V.4, où les valeurs sont dessinées en fonction du nombre k de l'itération).

Remarque 253. **(a)** Comme le calcul de la décomposition QR d'une matrice pleine est très coûteux ($O(n^3)$ opérations), on applique l'algorithme QR uniquement aux matrices de Hessenberg. Dans cette situation une itération nécessite seulement $O(n^3)$ opérations.

(b) La convergence est très lente en général (seulement *linéaire*). Pour rendre efficace cet algorithme, il faut absolument trouver un moyen pour accélérer la convergence.

(c) Considérons la situation où A est une matrice ré elle qui possède des valeurs propres complexes (l'hypothèse (18.1) est violée). L'algorithme QR produit une suite de matrices T_k qui sont toutes réelles. Dans cette situation, les T_k ne convergent pas vers une matrice triangulaire, mais deviennent triangulaires par blocs (sans démonstration). Comme la dimension des blocs dans la diagonale vaut en général 1 ou 2, on obtient également des approximations des valeurs propres.

15.4 Accélération de la convergence

D'après l'observation (18.11), nous savons que

$$t_{n,n-1}^{(k)} = O(|\lambda_n/\lambda_{n-1}|^k)$$

La convergence vers zéro de cet élément ne va être rapide que si $|\lambda_n| \ll |\lambda_{n-1}|$. Une idée géniale est d'appliquer l'algorithme QR à la matrice $A - pI$ où $p \approx \lambda_n$ Comme les valeurs propres de $A - pI$ sont $\lambda_i - p$, on a la propriété $|\lambda_n - p| \ll |\lambda_i - p|$ pour $i = 1, ..., n-1$ et l'élément $t_{n,n-1}^{(k)}$ va converger rapidement vers zéro. Rien ne nous empêche d'amé liorer l'approximation p après chaque itération. L'algorithme QR avec "shift" devient alors :

```
T_{0}=A
for
   k=1,2,..
determiner le parametre p_{k-1}$
Q_{k}R_{k}=T_{k-1}-p_{k-1}I    (decomposition QR)
T_{k}=R_{k}Q_{k}+p_{k-1}
end
```

Les matrices T_k de cette itération satisfont

$$Q_k^* T_{k-1} Q_k = Q_k^*(Q_k R_k + p_{k-1} I)Q_k = R_k Q_k + p_{k-1} I = T_k$$

Ceci implique que, indépendamment de la suite p_k , les matrices T_k ont toutes les mêmes valeurs propres que $T_0 = A$. Pour décrire complètement l'algorithme QR avec shift, il faut encore discuter le choix du paramètre p_k et il faut donner un critère pour arrêter l'itération.

15.4.1 Choix du "shift"-paramètre.

On a plusieurs possibilités :

— $p_k = t_{n,n}^{(k)}$: ce choix marche très bien si les valeurs propres de la matrice sont réelles.

— on considère la matrice

$$\begin{pmatrix} t_{n-1,n-1}^{(k)} & t_{n-1,n}^{(k)} \\ t_{n,n-1}^{(k)} & t_{n,n}^{(k)} \end{pmatrix} \tag{15.12}$$

Si les valeurs propres de (18.12) sont réelles, on choisit pour p_k celle qui est la plus proche de $t_{n,n}^{(k)}$. Si elles sont de la forme $\alpha \pm i\beta$ avec $\beta \neq 0$ (donc complexes), on prend d'abord $p_k = \alpha + i\beta$ et pour l'itération suivante $p_{k+1} = \alpha - i\beta$

15.5 Critère pour arrêter l'itération.

L'idée est d'itérer jusqu'à ce que $t_{n,n-1}^{(k)}$ ou $t_{n-1,n-2}^{(k)}$ soit suffisamment petit. Plus précisément, on arr ête l'itération quand

$$t_{l,l-1}^{(k)} \leq eps.(\left|t_{l-1,l-1}^{(k)}\right| + \left|t_{l,l}^{(k)}\right|) \qquad \text{pour} \quad l = n \quad \text{ou} \quad l = n-1 \tag{15.13}$$

— Si (18.13) est vérifié pour $l = n$ on accepte $t_{n,n}^{(k)}$ comme approximation de λ_n et on continue l'ité ration avec la matrice $\left(t_{i,j}^{(k)}\right)_{1 \leq i,j \leq n-1}$

— Si (18.13) est vérifié pour $l = n-1$, on accepte les deux valeurs propres de (18.12) comme approximations de λ_n et λ_{n-1} et on continue l'itération avec la matrice $\left(t_{i,j}^{(k)}\right)_{1 \leq i,j \leq n-2}$

Exemple 254. Nous avons appliqué l'algorithme QR à la matrice (**??**) avec le shift $p_k = t_{n,n}^{(k)}$. La convergence de $t_{i+1,i}^{(k)}$ vers z éro est illustrée dans la figure V.5. Une comparaison avec la figure V.4 nous montre que la convergence est beaucoup plus rapide (convergence quadratique). Après 5 itérations, on a $\left|t_{4,3}^{(k)}\right| \leq 10^{-15}$. Encore 4 itérations pour la matrice de dimension 3 donnent $\left|t_{3,2}^{(k)}\right| \leq 10^{-15}$ Il ne reste plus que 3 itérations à faire pour la matrice de dimension 2 pour avoir $\left|t_{2,1}^{(k)}\right| \leq 10^{-15}$. En tout, 12 itérations ont donné toutes les valeurs propres avec une précision de 15 chiffres.

15.6 Le "double shift" de Francis

Dans la situation où A est une matrice réelle ayant des valeurs propres complexes, il est recommandé de choisir un shift-paramètre p_k qui soit complexe. Une application directe de l'algorithme préc édent nécessite un calcul avec des matrices complexes. L'observation suivante permet d'éviter ceci.

Lemme 255. Soit T_k une matrice réelle, $p_k = \alpha + i\beta$ et $p_{k+1} = \alpha - i\beta$.Alors, on peut choisir les décompositions dans l'algorithme QR de manière à ce que T_{k+2} soit réelle.

Remarque 256. La décomposition QR d'une matrice est unique sauf qu'on peut remplacer QR par $(QD)^{-1}\left(D^{-1}R\right)$ où $D = diag(d_1, ..., d_n)$ avec $|d_i| = 1$.

Démonstration. La formule (**??**) montre que

$$T_{k+2} = (Q_{k+1}Q_{k+2})^* T_k (Q_{k+1}Q_{k+2}) \tag{15.14}$$

cqfd

Il suffit alors de démontrer que le produit $Q_{k+1}Q_{k+2}$ est ré el. Une manipulation à l'aide de formules pour T_k donne

$$\begin{aligned} Q_{k+1}Q_{k+2}R_{k+2}R_{k+1} &= Q_{k+1}(T_{k+1} - p_{k+1}I)R_{k+1} = Q_{k+1}(R_{k+1}Q_{k+1} + p_{k+1}I - p_kI)R_{k+1} = \\ &= (Q_{k+1}R_{k+1})^2 + (p_k - p_{k+1})Q_{k+1}R_{k+1} = (T_k - p_kI)^2 + (p_k - p_{k+1})(T_k - p_kI) = \\ &= T_k^2 - (p_k + p_{k+1})T_k + p_kp_{k+1}I = M \end{aligned} \tag{15.15}$$

On a donc trouvé une décomposition QR de la matrice M qui, en cons équence des hypothèses du lemme, est une matrice réelle. Si, dans l'algorithme QR, la décomposition est choisie de manière à ce que les éléments diagonaux de R_{k+1} et R_{k+2} soient r éels, alors, à cause de l'unicité de la décomposition QR, les matrices $Q_{k+1}Q_{k+2}$ et $R_{k+2}R_{k+1}$ sont réelles. Une possibilité de calculer T_{k+2} à partir de T_k est de calculer de (18.15), de faire une décomposition QR (réelle) de M et de calculer T_{k+2} à l'aide de (18.14). Cet algorithme n'est pas pratique car le calcul de T_k^2 nécessite $O(n^3)$ op érations, même si T_k est sous forme de Hessenberg. Il y a une astuce intéressante pour obtenir T_{k+2} à partir de T_k en $O(n^2)$ opérations. Elle est basée sur la propriét é suivante.

Théorème 257. Soit une matrice donnée et supposons que

$$Q^*TQ = S \tag{15.16}$$

où Q est orthogonale et S est sous forme de Hessenberg satisfaisant $s_{i,i-1} \neq 0$ pour $i = 2, ..., n$ Alors, Q et S sont déterminé es de manière "unique" par la première colonne de Q.

Remarque 258. On a "unicité" dans le sens suivant : si $\hat{Q}^*T\hat{Q}$ est de type Hessenberg avec une matrice orthogonale $\hat{Q}$ satisfaisant $\hat{Q}e_1$, alors $\hat{Q} = QD$ où $D = diag(d_1, ..., d_n)$ avec $|d_i| = 1$.

Démonstration. Notons les colonnes de Q par q_i. Alors, la relation (18.16) implique

$$Tq_i = \sum_{j=1}^{i+1} s_{ji}q_j, \qquad q_j^*Tq_i = s_{ji}. \tag{15.17}$$

Si q_1 est fixé, la valeur s_{11} est donnée par la deuxiè me formule de (18.17). Avec cette valeur, on obtient de la premiè re formule de (18.17) que q_2 est un multiple de $Tq_1 - s_{11}q_1$. Ceci détermine q_2 à une unité prè s. Maintenant, les valeurs s_{21}, s_{12}, s_{22} sont déterminées et q_3 est un multiple de $Tq_2 - s_{21}q_1 - s_{22}q_2$ etc. cqfd

Si les hypothèses du lemme précédent sont vérifiées, on peut calculer la matrice réelle T_{k+2} en $O(n^2)$ opérations de la manière suivante :

— calculer $M\varepsilon_1$, la première colonne de M (formule (18.15)) ;
— déterminer une matrice de Householder H_1 telle que $H_1(M\varepsilon_1) = \alpha e_1$
— transformer $H_1^T T_k H_1$ sous forme de Hessenberg à l'aide de matrices de Householder $H_2, ..., H_{n-1}$ (voir le paragraphe V.3) ; c'est-à-dire., calculer $H^T T_k H$ où $H = H_1 H_2 .... H_{n-1}$.

Comme $H_i e_1 = e_1$ pour $i = 2, ..., n-1$, la première colonne de H est un multiple de celle de M (observer $H_1^T = H_1$). Par la formule (18.15), la première colonne de $Q_{k+1}Q_{k+2}$ est aussi un multiple de Me_1. Par conséquent, pour un bon choix des dé compositions $Q_{k+1}R_{k+1}$ et $Q_{k+2}R_{k+2}$ on a $H = Q_{k+1}Q_{k+2}$ la matrice obtenue par cet algorithme est égale à T_{k+2} (voir (18.14)).

15.7 Etude de la convergence

Supposons d'être déjà proche de la limite et considérons, par exemple, la matrice

$$T_0 = A = \begin{pmatrix} 2 & a \\ \varepsilon & 1 \end{pmatrix}$$

où ε est un nombre petit. Avec le choix $p_0 = 1$ pour le shift-paramètre, on obtient

$$T_0 - p_0 I = \begin{pmatrix} 1 & a \\ \varepsilon & 0 \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{1+\varepsilon^2}} & -\frac{\varepsilon}{\sqrt{1+\varepsilon^2}} \\ \frac{\varepsilon}{\sqrt{1+\varepsilon^2}} & \frac{1}{\sqrt{1+\varepsilon^2}} \end{pmatrix} \begin{pmatrix} \sqrt{1+\varepsilon^2} & \frac{a}{\sqrt{1+\varepsilon^2}} \\ 0 & -\frac{a\varepsilon}{\sqrt{1+\varepsilon^2}} \end{pmatrix} = Q_1 R_1$$

et

$$T_0 - p_0 I = R_1 Q_1 = \begin{pmatrix} * & * \\ -\frac{a\varepsilon^2}{1+\varepsilon^2} & * \end{pmatrix}$$

— si A est symétrique (c'est-à-dire, $a = \varepsilon$) on a $t^{(1)}_{n,n-1} = O(\varepsilon^2)$, donc convergence *cubique*.
— si A n'est pas symétrique (p.ex. on a donc convergence quadratique.

Ces propriétés restent vraies pour des matrices générales (sans démonstration).

16 Exercices

Exercice 259. Calculer les valeurs propres de la matrice tridiagonale (dimension $n, b.c > 0$)

$$A = \begin{pmatrix} a & c & & & \\ b & a & c & & \\ & b & a & c & \\ & & b & \ddots & \ddots \\ & & & \ddots & \end{pmatrix}$$

Indication. Les composants du vecteur propre $(v_1, v_2, ..., v_n)^T$ satisfont une équation aux diffé rences finies avec.$v_0 = v_{n+1} = 0$ Vérifier que $v_j = Const.(\alpha_1^j - \alpha_2^j)$ où

$$\alpha_1 - \alpha_2 = \frac{\lambda - a}{c}, \qquad \alpha_1.\alpha_2 = \frac{b}{c}, \qquad \left(\frac{\alpha_1}{\alpha_2}\right)^{n+1} = 1$$

Résultat : $\lambda_j = a - 2\sqrt{bc}.\cos\left(\frac{j\pi}{n+1}\right), \qquad j = 1, 2, .., n.$

Exercice 260. Considérer la matrice

$$A(\varepsilon) = \begin{pmatrix} 1 & \varepsilon & 0 \\ -1 & 0 & 1 \\ 1 & -1+\varepsilon & -\varepsilon \end{pmatrix}$$

cette matrice possède une valeur propre de la forme

$$\lambda(\varepsilon) = i + \varepsilon.d + O(\varepsilon^2)$$

Calculer d et dessiner la tangente à la courbe $\lambda(\varepsilon)$ au point $\lambda(0)$

(a) Calculer par la méthode de la puissance, la plus grande valeur propre de la matrice

$$A = \begin{pmatrix} 99 & 1 & 0 \\ 1 & 100 & 1 \\ 0 & 1 & 98 \end{pmatrix}$$

(b) Pour accélerer considérablement la vitesse de convergence, appliquer la méthode de la puissance à la matrice $A - pI$ avec un choix intelligent de p.

(c) Avec quel choix de p obtient-on la valeur propre la plus petite ?

Exercice 261. Considérons la matrice tridiagonale

$$A = \begin{pmatrix} b_1 & c_1 & \\ a_1 & b_2 & c_2 \\ & a_2 & \\ & & \ddots \end{pmatrix}$$

Montrer que, si $a_i c_i > 0$ pour $i = 1, ..., n-1$, toutes les valeurs propres de A sont réelles. *Indication*. Trouver $D = diag(d_1, ..., d_n)$ telle que DAD^{-1} soit symétrique.

Exercice 262. Soit A une matrice symétrique et B quelconque. Montrer que pour chaque valeur propre λ_B de B il existe une valeur propre λ_A de A telle que

$$|\lambda_A - \lambda_B| \leq \|A - B\|_2 .$$

Indication. Montrer l'existence d'un vecteur v tel que $v = (A - \lambda_B)^{-1}(A - B)v$. En déduire que $1 \leq \left\|(A - \lambda_B)^{-1}(A - B)\right\| \leq \left\|(A - \lambda_B)^{-1}\right\| \|(A - B)\|$.

Exercice 263. (Schur, 1909). Soit A une matrice symétrique. Montrer que pour chaque indice i il existe une valeur propre λ de A telle que

$$|\lambda - a_{ii}| \leq \sqrt{\sum_{j \neq i} \left|a_{ij}\right|^2}$$

Indication. Appliquer l'exercice 5 avec une B convenable.

Exercice 264. Soit A une matrice réelle avec pour valeur propre $\alpha + i\beta$. Montrer que l'itération

$$\begin{pmatrix} \bar{\alpha} I - A & -\bar{\beta} I \\ \bar{\beta} I & \bar{\alpha} I - A \end{pmatrix} \begin{pmatrix} u_{k+1} \\ v_{k+1} \end{pmatrix} = \begin{pmatrix} u_k \\ v_k \end{pmatrix}$$

où $\bar{\alpha} \approx \alpha$ et $\bar{\beta} \approx \beta$) permet de calculer la valeur propre $\alpha + i\beta$ et le vecteur propre correspondant. Indication. Considérer les parties réelles et complexes de l'itération de Wielandt. On obtient alors

$$\frac{u_k^T A u_k + v_k^T A v_k}{u_k^T u_k + v_k^T v_k} \longrightarrow \alpha, \qquad \frac{u_k^T A v_k + v_k^T A u_k}{u_k^T u_k + v_k^T v_k} \longrightarrow \beta$$

Exercice 265. Considérons la matrice de Hilbert,

$$A = \begin{pmatrix} 1/2 & 1/3 & 1/4 \\ 1/3 & 1/4 & 1/5 \\ 1/4 & 1/5 & 1/6 \end{pmatrix}$$

(a) Transformer A en une matrice tridiagonale ayant les mêmes valeurs propres.

(b) En utilisant une suite de Sturm, montrer que toutes les valeurs propres sont positives et qu'une valeur propre est plus petite que 0.001

(c) Calculer approximativement la condition de A pour la norme Euclidienne.

Exercice 266. La formule de récurrence

$$(k+1)P_{k+1}(x) = (2k+1)xP_k(x) - kP_{k-1}(x)$$

pour les *polynômes de Legendre* ressemble à

$$p_i(\lambda) = (d_i - \lambda)p_{i-1}(\lambda) - \varepsilon_i^2 p_{i-2}(\lambda), \qquad i = 2, \ldots, n.$$

pour les polynômes $\det(A_i - \lambda I)$. Trouver une matrice tridiagonale A de dimension n telle que les valeurs propres de A sont les racines de $P_n(x)$.

Exercice 267. Soit $p(x)$ un polynôme de degré n et supposons que toutes les racines soient simples. Démontrer que la suite définie par l'algorithme d'Euclide,

$$\begin{aligned} p_n(x) &= p(x), \qquad p_{n-1}(x) = -p'(x) \\ p_i(x) &= q_i(x)p_{i-1}(x) - \gamma_i^2 p_{i-2}(x), \qquad i = n, \ldots, 2. \end{aligned}$$

est une suite de Sturm. Pour le polynôme $p(x) = x^5 - 6x^4 + 3x^3 + 3x^2 + 2x + 8$.

(a) déterminer le nombre de racines réelles.

(b) Combien de racines sont complexes ?

(c) Combien de racines sont réelles et positives ?

Exercice 268. Pour un φ donné notons $c = \cos\varphi$ et $s = \sin\varphi$. La matrice Ω_{kl}, définie par

$$(\Omega_{kl})_{ij} = \begin{cases} 1 & \text{si } i = j, j \neq k, j \neq l \\ c & \text{si } i = j = k, \text{ ou } i = j = l \\ s & \text{si } i = k, \text{ et } j = l \\ -s & \text{si } i = l, \text{ et } j = k \\ 0 & \text{sinon} \end{cases}$$

s'appelle rotation de Givens.

(a) Montrer qu'elle est orthogonale.

(b) Soit A une matrice symétrique.Déterminer φ tel que le (k,l)-ième élément de $A' = \Omega_{kl} A \Omega_{kl}^T$ s'annule.

Resultat. $\cot 2\varphi = (a_{kk} - a_{ll})/(2a_{kl})$.

Exercice 269. La méthode de Jacobi (1846) pour le calcul des valeurs propres d'une matrice symétrique :

i) on choisit a_{kl} $(k > l)$ tel que $|a_{kl}| = \max_{i>j} |a_{ij}|$;

ii) on détermine A' comme dans l'exercice 11.

Montrer que, si on répète cette procédure, on a convergence vers une matrice diagonale, dont les éléments sont les valeurs propres de A Indication. Montrer que $\sum_{i>j} |a'_{ij}|^2 = \sum_{i>j} |a_{ij}|^2 - |a_{kl}|^2$

Exercice 270. On considère la matrice

$$A = \begin{pmatrix} 7 & 0,5 \\ 0,0001 & 8 \end{pmatrix}$$

dont on cherche à calculer les valeurs propres.

(a) Faire une itération de l'algorithme QR sans shift.

(b) Faire une itération de l'algorithme QR avec shift.

(c) Estimer la position des valeurs propres de A à l'aide du Th éorème de Gershgorin.

(d) Calculer les valeurs propres de A à l'aide du polynôme caractéristique.

Exercice 271. Montrer que si la matrice $T_0 = A$ est une matrice de Hessenberg (ou tridiagonale), alors les matrices T_k, $k \geq 1$ construites par l'algorithme QR sont également des matrices de Hessenberg (tridiagonales).

Exercice 272. Donner une estimation grossière du nombre d'opérations qui sont n écessaires pour effectuer la décomposition QR d'une matrice de Hessenberg et pour calculer ensuite le produit RQ.

Exercice 273. Soit T_0 une matrice de Hessenberg dont tous les éléments de la sous-diagonale sont non-nuls. Montrer que, si p_0 est une valeur propre de T_0, une itération de l'algorithme QR avec shift p_0 donne

$$t^{(1)}_{n,n-1} = 0.$$

Exercice 274. Expliquer, comment le calcul de T_k à partir de T_{k-1}

$$Q_k R_k = T_{k-1} - p_{k-1} I, \qquad T_k = R_k Q_k + p_{k-1} I.$$

peut être effectué sans soustraire (et additionner) explicitement la matrice $p_{k-1} I$ Indication. Laissez-vous inspirer par le "double shift" algorithme de Francis.

17 TRANSFORMATION SOUS FORME TRIDIAGONALE (ou de HESSENBERG)

Avec la transformation $v = Pu$ (où est P une matrice inversible) le problème

$$Av = \lambda v$$

devient

$$P^{-1}APu = \lambda u$$

Donc, les valeurs propres de A et de $P^{-1}AP$ sont les mêmes et les vecteurs propres v_i de A se transforment par $v_i = Pu_i$. Le but de ce paragraphe est de trouver une matrice P telle que $P^{-1}AP$ devienne "plus simple". La situation idéale serait trouvée si $P^{-1}AP$ devenait diagonale ou triangulaire - mais une telle transformation nécessiterait déjà la connaissance des valeurs propres.

Alors, on cherche P tel que $P^{-1}AP$ soit sous forme de Hessenberg

$$P^{-1}AP = H = \begin{pmatrix} * & * & \cdots & \cdots & * \\ * & * & \ddots & & \vdots \\ & * & \ddots & \ddots & * \\ & & \ddots & \ddots & * \\ & & & * & * \end{pmatrix} \tag{17.1}$$

c'est-à-dire, $h_{ij} = 0$ pour $i > j + 1$. Pour arriver à ce but, nous considérons deux algorithmes.

17.1 a) A l'aide des transformations élémentaires

Comme pour l'élimination de Gauss, nous utilisons les transformations pour faire apparaitre les zéros - colonne par colonne - dans (17.1).

Dans un premier pas, nous choisissons $k \geq 2$ tel que $|a_{k1}| \geq |a_{j1}|$ pour $j \geq 2$ et nous permutons les lignes 2 et k, c'est-à-dire, nous formons PA où P est une matrice de permutation convenable. Pour ne pas changer les valeurs propres, il faut également permuter les colonnes 2 et k (ceci correspond au calcul de $A' = PAP^{-1}$ car $P^2 = I$ (et donc $P = P^{-1}$). Si $a'_{21} = 0$, on a aussi $a'_{i1} = 0$ pour $i \geq 3$ et le premier pas est terminé. Sinon, nous déterminons

$$L_2 = \begin{pmatrix} 1 & & & & \\ 0 & 1 & & & \\ 0 & -l_{32} & 1 & & \\ \vdots & \vdots & \ddots & \ddots & \\ 0 & -l_{n2} & \cdots & 0 & 1 \end{pmatrix} \quad \text{telle que} \quad L_2 A' = \begin{pmatrix} a'_{11} & a'_{12} & \cdots & a'_{1n} \\ a'_{21} & a'_{22} & \cdots & a'_{2n} \\ 0 & * & \cdots & * \\ \vdots & \vdots & \ddots & \vdots \\ 0 & * & \cdots & * \end{pmatrix}$$

Pour ceci, on définit $l_{i2} = \frac{a'_{i1}}{a'_{21}}$. Une multiplication à droite avec

$$L_2^{-1} = \begin{pmatrix} 1 & & & & \\ 0 & 1 & & & \\ 0 & l_{32} & 1 & & \\ \vdots & \vdots & \ddots & \ddots & \\ 0 & l_{n2} & \cdots & 0 & 1 \end{pmatrix}$$

ne change pas la première colonne de $L_2 A'$.

On répète la même procédure avec la sous-matrice de $L_2 A' L_2^{-1}$ de dimension $n-1$, et ainsi de suite. A cause des multiplications à droite avec L_i^{-1}, cet algorithme coûte deux fois plus cher que l'élimination de Gauss.

Pour la matrice

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

on prend

$$L_2 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -1/2 & 1 \end{pmatrix}$$

et on obtient

$$L_2 A = \begin{pmatrix} 3 & 2 & 1 \\ 2 & 1 & 3 \\ 0 & 5/2 & -1/2 \end{pmatrix}, \quad \text{puis} \quad L_2 A L_2^{-1} = \begin{pmatrix} 3 & 5/2 & 1 \\ 2 & 5/2 & 3 \\ 0 & 9/4 & -1/2 \end{pmatrix} = H$$

Cet exemple montre un désavantage de cet algorithme : si l'on part avec une matrice symétrique A, la matrice de Hessenberg H, obtenue par cet algorithme, n'est plus symétrique en général.

17.2 b) A l'aide des transformations orthogonales

Il est souvent préférable de travailler avec des réflexions de Householder Commençons par une réflexion pour les coordonnées $2,...,n$ laissant fixe la première coordonnée : $\bar{Q}_2 = I - 2\bar{u}_2\bar{u}_2^T$ ($\|\bar{u}_2\|_2 = 1$) tel que $\bar{Q}_2\bar{A}_1 = \alpha_2 e_1$ où $\bar{A}_1 = (a_{21},...,a_{n1})$. En posant $u_2 = (0,\bar{u}_2)^T$ et $Q_2 = I - 2u_2u_2^T$, la matrice Q_2A contient des zéros dans la première colonne à partir du troisième élément. La multiplication à droite avec $Q_2^{-1} = Q_2^T = Q_2$ ne change pas cette colonne :

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \xrightarrow{Q_2A} \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ \alpha_2 & * & * \\ 0 & * & * \end{pmatrix} \xrightarrow{Q_2AQ_2} \begin{pmatrix} a_{11} & * & * \\ \alpha_2 & * & * \\ 0 & * & * \end{pmatrix}$$

Dans le pas suivant, on applique la même procédure à la sous-matrice de dimension $n-1$, etc. Finalement, on arrive à la forme de Hessenberg (17.1) avec la tranformation $P^{-1} = Q_{n-1}....Q_2$ qui est une matrice orthogonale (c'est-à-dire, $P^{-1} = P^T$).

Nous avons un double avantage avec cet algorithme :

— il ne faut pas faire une recherche de pivot ;

— si A est symétrique, alors $P^{-1}AP$ est aussi symétrique, et donc tridiagonale.

17.3 Méthode de bissection pour des matrices tridiagonales

Considérons une matrice symétrique tridiagonale

$$A = \begin{pmatrix} d_1 & e_2 & & & \\ e_2 & d_2 & e_3 & & \\ & e_3 & \ddots & \ddots & \\ & & \ddots & \ddots & e_n \\ & & & e_n & d_n \end{pmatrix}$$

On observe tout d'abord que si un élément e_i est nul, la matrice A est déjà décomposée en deux sous-matrices du même type, qui ensemble fournissent les valeurs propres de A .

On peut donc supposer, sans restreindre la généralité, que

$$c_i \neq 0 \quad \text{pour} \quad i = 2,...,n. \tag{17.2}$$

Pour cette matrice, il est possible de calculer la valeur $P_A(\lambda)$ du polynôme caractéristique sans connaitre ses coefficients. En effet, si l'on pose

$$A_1 = (d_1), \qquad A_2 = \begin{pmatrix} d_1 & e_2 \\ e_2 & d_2 \end{pmatrix}, \qquad A_3 = \begin{pmatrix} d_1 & e_2 & \\ e_2 & d_2 & e_3 \\ & e_3 & d_3 \end{pmatrix}, \qquad ...$$

et si l'on définit

$$p_i(\lambda) = \det(A_i - \lambda I),$$

on obtient

$$\begin{aligned} p_0(\lambda) &= 1 \\ p_1(\lambda) &= d_1 - \lambda \\ p_i(\lambda) &= (d_i - \lambda)\, p_{i-1}(\lambda) - e_i^2 p_{i-2}(\lambda), \qquad i = 2,...,n. \end{aligned} \tag{17.3}$$

La formule de récurrence dans (17.3) est obtenue en développant le déterminant de la matrice $A_i - \lambda I$ par rapport à la dernière ligne (ou colonne).

En principe, on peut maintenant calculer les valeurs propres de A (c'est-à-dire. les zéros de $p_n(\lambda)$ de la manière suivante : chercher un intervalle où $p_n(\lambda)$ change de signe et localiser une racine de $p_n(\lambda) = 0$ par bissection. Les évaluations de $p_n(\lambda)$ sont faites à l'aide de la formule (17.3). Mais il existe une astuce interessante qui permet d'améliorer cet algorithme.

Théorème 275. Si l'équation (17.2) est vérifié, les polynômes $p_i(\lambda)$ définis par (17.3) satisfont

a) $p_n'(\hat{\lambda})p_{n-1}(\hat{\lambda}) < 0$ si $p_n(\hat{\lambda}) = 0$ $(\hat{\lambda} \in \mathbb{R})$

b) $p_{i-1}(\hat{\lambda})p_{i+1}(\hat{\lambda}) < 0$ si $p_i(\hat{\lambda}) = 0$ pour un $\{i \in 1,2,...,n-1\}$

c) $p_0(\lambda)$ ne change pas de signe sur $\mathbb{R}$.

Démonstration. L'affirmation (c) est triviale.

Si $p_i(\hat{\lambda}) = 0$ pour un $\{i \in 1,2,...,n-1\}$, la formule de récurrence (17.3) donne l'inégalité $p_{i-1}(\hat{\lambda})p_{i+1}(\hat{\lambda}) \leq 0$. Pour démontrer (b), il suffit d'exclure le cas $p_{i-1}(\hat{\lambda})p_{i+1}(\hat{\lambda}) = 0$. Si deux valeurs consécutives

de la suite $\left\{p_i(\hat{\lambda})\right\}$ sont nulles, la formule de récurrence montre que $p_i(\hat{\lambda}) = 0$ pour tout i, ce qui contredit $p_0(\lambda) = 1$.

Nous démontrons par récurrence que toutes les racines de $p_i(\lambda)$ sont réelles, simples et séparées par celles de $p_{i-1}(\lambda)$.

Il n'y a rien à démontrer pour $i = 1$. Supposons la propriété vraie pour i et montrons qu'elle est encore vraie pour $i+1$. Comme les zéros $\lambda_1 < \lambda_2 < ... < \lambda_i$ sont séparés par ceux de $p_{i-1}(\lambda)$ et comme $p_{i-1}(-\infty) = +\infty$, nous avons $sign\ p_{i-1}(\lambda_j) = (-1)^{j+1}$. Alors, on déduit de (b) que $sign\ p_{i+1}(\lambda_j) = (-1)^j$. Ceci et le fait que $p_{i+1}(\lambda) = (-1)^{j+1}\lambda^{i+1} + ...$ montrent que $p_{i+1}(\lambda)$ possède un zéro réel dans chacun des intervalles ouverts $(-\infty, \lambda_1), (\lambda_1, \lambda_2), ..., (\lambda_i, \infty)$

L'affirmation (a) est maintenant une conséquence de (b) et du fait que toutes les racines de $p_{i-1}(\lambda)$ sont réelles simples ; cqfd

Définition 276 (suite de Sturm). Une suite $\{p_0, p_1, ..., p_n\}$ de polynômes à coefficients réels s'appelle une suite de Sturm, si elle vérifie les conditions (a), (b), (c) du Théorème (275)

Considérons une suite de Sturm $\{p_0, p_1, ..., p_n\}$. Si l'on définit

$$\omega(\lambda) = \text{nombre de changements de signes de } \{p_0(\lambda), p_1(\lambda), ..., p_n(\lambda)\}$$

alors le polynôme $p_n(\lambda)$ possède exactement

$$\omega(b) - \omega(a)$$

zéros dans l'intervalle $[a, b]$ (si $p_i(\lambda) = 0$, on définit $signp_i(\lambda) = sign\ p_{i-1}(\lambda)$).

Démonstration. Par continuité, l'entier $\omega(\lambda)$ peut changer sa valeur seulement si une valeur des fonctions $p_i(\lambda)$ devient nulle. La fonction $p_0(\lambda)$ ne change pas de signe. Supposons alors que $p_i(\tilde{\lambda}) = 0$ pour un $i \in \{1, 2, ..., n-1\}$. La condition (b) et la continuité de $p_j(\lambda)$ montrent que seulement les deux situations suivantes sont possibles (ε petit) :

	$\tilde{\lambda}-\varepsilon$	$\tilde{\lambda}$	$\tilde{\lambda}+\varepsilon$
$p_{i-1}(\lambda)$	+	+	+
$p_i(\lambda)$	±	0	±
$p_{i+1}(\lambda)$	−	−	−

	$\tilde{\lambda}-\varepsilon$	$\tilde{\lambda}$	$\tilde{\lambda}+\varepsilon$
$p_{i-1}(\lambda)$	−	−	−
$p_i(\lambda)$	±	0	±
$p_{i+1}(\lambda)$	+	+	+

cqfd

Chaque fois, on a $\omega(\tilde{\lambda}+\varepsilon) = \omega(\tilde{\lambda}) = \omega(\tilde{\lambda}-\varepsilon)$ et la valeur de $\omega(\lambda)$ ne change pas si λ traverse un zéro de $p_i(\lambda)$ pour $i \in \{1, 2, ..., n-1\}$

Il reste à étudier la fonction $\omega(\lambda)$ dans un voisinage d'un zéro $\tilde{\lambda}$ de $p_n(\lambda)$. La propriété (a) implique que pour les signes de $p_j(\lambda)$ on a seulement les deux possibilités suivantes :

	$\tilde{\lambda}-\varepsilon$	$\tilde{\lambda}$	$\tilde{\lambda}+\varepsilon$
$p_{n-1}(\lambda)$	+	+	+
$p_n(\lambda)$	+	0	−

	$\tilde{\lambda}-\varepsilon$	$\tilde{\lambda}$	$\tilde{\lambda}+\varepsilon$
$p_{n-1}(\lambda)$	−	−	−
$p_n(\lambda)$	−	0	+

c'est-à-dire, $\omega(\tilde{\lambda}+\varepsilon) = \omega(\tilde{\lambda}-\varepsilon)+1$. Ceci démontre que la fonction $\omega(\lambda)$ est constante par morceaux et augmente de 1 sa valeur si λ traverse un zéro de $p_n(\lambda)$.

17.4 Méthode de bissection.

Si l'on applique ce théorème à la suite (17.3), la différence $\omega(b) - \omega(a)$ est égale au nombre de valeurs propres de (17.2) dans l'intervalle $[a, b]$. On obtient toutes les valeurs propres de A de la manière suivante :

— on cherche un intervalle $[a, b]$ qui contienne toutes les valeurs propres de A (par exemple, en appliquant le théorème de Gershgorin). On a donc que $\omega(a) = 0$ et $\omega(b) = n$.

— on pose $c = \frac{(a+b)}{2}$ et on calcule $\omega(c)$. Les différences $\omega(c) - \omega(a)$ et $\omega(b) - \omega(c)$ indiquent combien de valeurs propres de A sont dans $[a, c)$ et combien sont dans $[c, b)$
— on continue à diviser les intervalles qui contiennent au moins une valeur propre de A.

On peut facilement modifier cet algorithme pour calculer la valeur propre la plus petite ou la $3^{ème}$ plus grande valeur propre, etc.

Pour éviter un "overflow" dans le calcul de $p_n(\lambda)$ (si n et λ sont grands), il vaut mieux travailler avec

$$f_i(\lambda) = \frac{p_i(\lambda)}{p_{i-1}(\lambda)} \qquad i = 1, 2, ..., n$$

et utiliser le fait que

$$\omega(\lambda) = \text{nombre d'éléments négatifs parmi } \{f_1(\lambda), f_2(\lambda), ..., f_n(\lambda)\}$$

(attention : si $p_{i-1}(\lambda)$ est zéro, on pose $f_i(\lambda) = -\infty$; cette valeur compte pour un élément négatif).

Pour une programmation de l'algorithme, on utilise la récurrence

$$\begin{aligned} f_1(\lambda) &= d_1 - \lambda \\ f_i(\lambda) &= d_i - \lambda - \begin{cases} e_i^2 / f_{i-1}(\lambda) & \text{si} \quad f_{i-1}(\lambda) \neq 0 \\ |e_i| / eps & \text{si} \quad f_{i-1}(\lambda) = 0 \end{cases} \end{aligned}$$

La formule pour le cas $f_{i-1}(\lambda) \neq 0$ est une conséquence de (17.3). Si $f_{i-1}(\lambda) = 0$ (c'est-à-dire $p_{i-1}(\lambda) = 0$), on remplace cette valeur par $|e_i|.eps$. Ceci correspond à ajouter la perturbation $|e_i|.eps$ à d_{i-1}

18 L'ITERATION ORTHOGONALE

Dans ce paragraphe, nous allons généraliser la méthode de la puissance afin de pouvoir calculer les deux (trois,) valeurs propres dominantes en même temps. Cette généralisation motivera l'itération QR qui constitue l'algorithme le plus important pour le calcul des valeurs propres d'une matrice.

18.1 Généralisation de la méthode de la puissance (pour calculer les deux valeurs propres dominantes).

Considérons une matrice A dont les valeurs propres satisfont

$$|\lambda_1| > |\lambda_2| > ... > |\lambda_n|. \tag{18.1}$$

La méthode de la puissance est basée sur l'itération $y_{k+1} = Ay_k$ et nous permet d'obtenir une approximation de λ_1 à l'aide du quotient de Rayleigh. Pour calculer (en même temps) la deuxième valeur propre λ_2, nous prenons deux vecteurs y_0 et z_0 satisfaisant $y_0^* z_0 = 0$ et nous considérons l'itération

$$\begin{aligned} y_{k+1} &= Ay_k \\ z_{k+1} &= Az_k - \beta_{k+1} y_{k+1} \end{aligned} \tag{18.2}$$

où β_{k+1} est déterminé par la condition $y_{k+1}^* z_{k+1} = 0$. Par induction, on voit que

$$\begin{aligned} y_k &= A^k y_0 \\ z_k &= A^k z_0 - \gamma_k y_k \end{aligned}$$

où γ_k est tel que

$$y_k^* z_k = 0 \tag{18.3}$$

Ceci signifie que le calcul de $\{z_k\}$ correspond à la méthode de la puissance appliquée à z_0, combinée avec une orthogonalisation (projection de $A^k z_0$ sur le complément orthogonal de y_k).

En exprimant les vecteurs initiaux dans la base de vecteurs propres $v_1, v_2, ..., v_n$ de la matrice A (on suppose $\|v_i\|_2 = 1$),

$$y_0 = \sum_{i=1}^{n} a_i v_i, \qquad z_0 = \sum_{i=1}^{n} b_i v_i, \tag{18.4}$$

les vecteurs y_k, z_k deviennent

$$y_k = \sum_{i=1}^{n} a_i \lambda_i^k v_i, \qquad z_0 = \sum_{i=1}^{n} (b_i - \gamma_k a_i)\, \lambda_i^k v_i,$$

Comme nous l'avons constaté précédement, pour $k \to \infty$, le terme $a_1 \lambda_1^k v_1$ est dominant dans y_k (si $a_1 \neq 0$) et on obtient une approximation du premier vecteur propre v_1 . Que peut-on dire pour la suite $\{z_k\}$?

La condition (18.3) d'orthogonalité implique que

$$\sum_{i=1}^{n} \sum_{i=1}^{n} a_i \left(b_j - \gamma_k a_j\right) \bar{\lambda}_i^k \lambda_j^k v_i^* v_j = 0 \tag{18.5}$$

Cette relation définit γ_k. Comme le terme avec $i = j = 1$ est dominant, on voit que $\gamma_k \approx b_1/a_1$

Par la suite, nous allons supposer que $a_1 \neq 0$ et $a_1 b_2 - a_2 b_1 \neq 0$. En divisant (18.5) par $\bar{\lambda}_1^k$ on obtient

$$\bar{a}_1(b_1 - \gamma_k a_1)\lambda_1^k(1 + O(|\lambda_2/\lambda_1|^k)) = -\bar{a}_1(b_2 - \gamma_k a_2)\lambda_2^k(v_1^* v_2 + O(|\lambda_2/\lambda_1|^k) + O(|\lambda_3/\lambda_2|^k)).$$

Maintenant, on peut insérer cette formule dans (18.4) et on en déduit

$$z_k = \lambda_{21}^k (b_2 - \gamma_k a_2)(v_2 - v_1^* v_2 . v_1 + O(|\lambda_2/\lambda_1|^k) + O(|\lambda_3/\lambda_2|^k)) \tag{18.6}$$

Visiblement, le vecteur z_k s'approche (pour $k \to \infty$) d'un multiple de $v_2 - v_1^* v_2 . v_1$, qui est la projection orthogonale de v_2 à l'hyperplan $v_1^{\perp}$. Concernant les valeurs propres, on a le résultat suivant.

Théorème 277. Considérons les vecteurs y_k, z_k donnés par (18.2) et notons

$$U_k = \left(y_k / \left\|y_k\right\|_2, z_k / \|z_k\|_2\right) \tag{18.7}$$

(observer que $U_k^* U_k = 1$). Si (18.1) est vérifié, on a que

$$U_k^* A U_k \longrightarrow \begin{pmatrix} \lambda_1 & * \\ 0 & \lambda_2 \end{pmatrix} \qquad \text{pour} \qquad k \longrightarrow \infty \tag{18.8}$$

Démonstration. L'élément (1,1) de la matrice $U_k^* A U_k$ est le quotient de Rayleigh (12.3) qui converge vers λ_1 . En utilisant (18.6), on voit que l'élément (2,2) satisfait

$$\frac{z_k^* A z_k}{z_k^* z_k} \longrightarrow \frac{\left(v_2 - v_1^* v_2 . v_1\right)^* (\lambda_2 v_2 - \lambda_1 v_1^* v_2 . v_1)}{\left(v_2 - v_1^* v_2 . v_1\right)^* (v_2 - v_1^* v_2 . v_1)} = \frac{\lambda_2 (1 - \left|v_1^* v_2\right|^2)}{1 - \left|v_1^* v_2\right|^2} = \lambda_2$$

De façon similaire, on obtient pour l'élément (2,1)

$$\frac{z_k^* A y_k}{\|z_k\|_2 \left\|y_k\right\|_2} \longrightarrow \frac{\left(v_2 - v_1^* v_2 . v_1\right)^* \lambda_1 v_1}{\left\|v_2 - v_1^* v_2 . v_1\right\|_2 \|v_1\|_2} = 0$$

Finalement, l'élément (1,2) de $U_k^* A U_k$ satisfait

$$\frac{y_k^* A z_k}{\left\|y_k\right\|_2 \|z_k\|_2} \longrightarrow \frac{v_1^*(\lambda_2 v_2 - \lambda_1 v_1^* v_2 . v_1)}{\|v_1\|_2 \left\|v_2 - v_1^* v_2 . v_1\right\|_2} = \frac{(\lambda_2 - \lambda_1) v_1^* v_2}{\sqrt{1 - \left|v_1^* v_2\right|^2}}.$$

Cette expression est en général non nulle. cqfd

Remarque 278. Avec la notation (18.7), l'itération (18.2) peut être écrite sous la forme

$$A U_k = U_{k+1} R_{k+1}$$

où R_{k+1} est une matrice 2×2 qui est triangulaire supérieure.

18.2 Méthode de la puissance (pour le calcul de toutes les valeurs propres)

ou simplement *itération orthogonale*. La généralisation de l'algorithme précédent au cas où l'on veut calculer toutes les á valeurs propres d'une matrice est évidente : on choisit une matrice orthogonale U_0, c'est-à-dire, on choisit n vecteurs orthogonaux (les colonnes de U_0) qui jouent le rôle de y_0, z_0, etc. Puis, on effectue l'itération

```
for k=1,2,..
    Z_{k}=AU_{k+1}          (d\U{e9}composition QR)
    U_{k}R_{k}=Z_{k}
end
```

Si (18.1) est vérifié et si la matrice U_0 est bien choisie ($a_1 \neq 0, a_1 b_2 - a_2 b_1 \neq 0$, , etc), une généralisation du théorème précédent donne la convergence

$$T_k = U_k^* A U_k \tag{18.9}$$

vers une matrice triangulaire dont les éléments de la diagonale sont les valeurs propres de A . On a donc transformé A en forme triangulaire à l'aide d'une matrice orthogonale (*décomposition de Schur*).

Il y a une possibilité intéressante pour calculer T_k de (18.9) directement à partir de T_{k-1}. D'une part, on déduit de (**??**) que

$$T_{k-1} = U_k^* A U_k = \left(U_{k-1}^* U_k\right) R_k \tag{18.10}$$

D'autre part, on a

$$T_k = U_k^* A U_k = U_k^* A U_{k-1}^* U_k = R_k \left(U_{k-1}^* U_k\right).$$

On calcule la décomposition QR de la matrice T_{k-1} et on échange les deux matrices de cette décomposition pour obtenir T_k

18.3 L'algorithme QR

La méthode QR, due à J.C.F. Francis et à V.N. Kublanovskaya, est la méthode la plus couram- ment utilisée pour le calcul de l'ensemble des valeurs propres (P.G. Ciarlet 1982)

the QR iteration, and it forms the backbone of the most effective algorithm for computing the Schur decomposition. (G.H. Golub & C.F. van Loan 1989)

La version simple du célèbre algorithme QR n'est rien d'autre que la méthode du paragraphe précédent. En effet, si l'on pose $Q_k = U_{k-1}^* U_k$ et si l'on commence l'itération avec $U_0 = I$, les formules (18.9) et (18.10) nous permettent d'écrire l'algorithme précédent comme suit : (décomposition QR)

```
T_{0}=A
for  k=1,2,..
    Q_{k}R_{k}=T_{k-1}
    T_{k}=R_{k}Q_{k}
end
```

Les T_k qui sont les mêmes que dans le paragraphe V.5, convergent (en général) vers une matrice triangulaire. Ceci nous permet d'obtenir toutes les valeurs propres de la matrice A car les T_k ont les mêmes valeurs propres que A (voir (18.9)).

Cet algorithme important a été développé indépendamment par J.G.F. Francis (1961) et par V.N. Kublanovskaya (1961). Un algorithme similaire, qui utilise la décomposition LR à la place de la décomposition QR, a été introduit par H. Rutishauser (1958).

Exemple 279. Appliquons la méthode QR à la matrice

$$A = \begin{pmatrix} 10 & 2 & 3 & 5 \\ 3 & 6 & 8 & 4 \\ 0 & 5 & 4 & 3 \\ 0 & 0 & 4 & 3 \end{pmatrix}$$

On peut montrer que, pour une matrice de Hessenberg A , toutes les matrices T_k sont aussi sous forme de Hessenberg. Pour 'étudier la convergence vers une matrice triangulaire, il suffit alors de considérer les éléments $t^{(k)}_{i+1,i} (i = 1, 2, ..., n-1)$ de la sous-diagonale. On constate que

$$\frac{t^{(k+1)}_{i+1,i}}{t^{(k)}_{i+1,i}} \approx \frac{\lambda_{i+1}}{\lambda_i} \tag{18.11}$$

$(\lambda_1 \approx 14,3, \lambda_2 \approx 7,86, \lambda_3 \approx 2,70, \lambda_4 \approx -1,86)$. Comme, les éléments $t^{(k)}_{i+1,i}$ convergent, pour $k \longrightarrow \infty$, linéairement vers 0 (voir la figure V.4, où les valeurs sont dessinées en fonction du nombre k de l'itération).

Remarque 280.
- **(a)** Comme le calcul de la décomposition QR d'une matrice pleine est très coûteux ($O(n^3)$ opérations), on applique l'algorithme QR uniquement aux matrices de Hessenberg. Dans cette situation une itération nécessite seulement $O(n^3)$ opérations.
- **(b)** La convergence est très lente en général (seulement *linéaire*). Pour rendre efficace cet algorithme, il faut absolument trouver un moyen pour accélérer la convergence.
- **(c)** Considérons la situation où A est une matrice réelle qui possède des valeurs propres complexes (l'hypothèse (18.1) est violée). L'algorithme QR produit une suite de matrices T_k qui sont toutes réelles. Dans cette situation, les T_k ne convergent pas vers une matrice triangulaire, mais deviennent triangulaires par blocs (sans démonstration). Comme la dimension des blocs dans la diagonale vaut en général 1 ou 2, on obtient également des approximations des valeurs propres.

18.4 Accélération de la convergence

D'après l'observation (18.11), nous savons que

$$t^{(k)}_{n,n-1} = O(|\lambda_n/\lambda_{n-1}|^k)$$

La convergence vers zéro de cet élément ne va être rapide que si $|\lambda_n| \ll |\lambda_{n-1}|$. Une idée géniale est d'appliquer l'algorithme QR à la matrice $A - pI$ où $p \approx \lambda_n$ Comme les valeurs propres de $A - pI$ sont $\lambda_i - p$, on a la propriété $|\lambda_n - p| \ll |\lambda_i - p|$ pour $i = 1, ..., n-1$ et l'élément $t^{(k)}_{n,n-1}$ va converger rapidement vers zéro. Rien ne nous empêche d'améliorer l'approximation p après chaque itération. L'algorithme QR avec "shift" devient alors :

```
T_{0}=A
for
   k=1,2,..
determiner le parametre p_{k-1}$
Q_{k}R_{k}=T_{k-1}-p_{k-1}I   (decomposition QR)
T_{k}=R_{k}Q_{k}+p_{k-1}
end
```

Les matrices T_k de cette itération satisfont

$$Q_k^* T_{k-1} Q_k = Q_k^*(Q_k R_k + p_{k-1} I)Q_k = R_k Q_k + p_{k-1} I = T_k$$

Ceci implique que, indépendamment de la suite p_k , les matrices T_k ont toutes les mêmes valeurs propres que $T_0 = A$.

Pour décrire complètement l'algorithme QR avec shift, il faut encore discuter le choix du paramètre p_k et il faut donner un critère pour arrêter l'itération.

18.4.1 Choix du "shift"-paramètre.

On a plusieurs possibilités :

— $p_k = t_{n,n}^{(k)}$: ce choix marche très bien si les valeurs propres de la matrice sont réelles.

— on considère la matrice

$$\begin{pmatrix} t_{n-1,n-1}^{(k)} & t_{n-1,n}^{(k)} \\ t_{n,n-1}^{(k)} & t_{n,n}^{(k)} \end{pmatrix} \tag{18.12}$$

Si les valeurs propres de (18.12) sont réelles, on choisit pour p_k celle qui est la plus proche de $t_{n,n}^{(k)}$.
Si elles sont de la forme $\alpha \pm i\beta$ avec $\beta \neq 0$ (donc complexes), on prend d'abord $p_k = \alpha + i\beta$ et pour l'itération suivante $p_{k+1} = \alpha - i\beta$

18.5 Critère pour arrêter l'itération.

L'idée est d'itérer jusqu'à ce que $t_{n,n-1}^{(k)}$ ou $t_{n-1,n-2}^{(k)}$ soit suffisamment petit. Plus précisément, on arrête l'itération quand

$$t_{l,l-1}^{(k)} \leq eps.(\left|t_{l-1,l-1}^{(k)}\right| + \left|t_{l,l}^{(k)}\right|) \qquad \text{pour} \quad l = n \quad \text{ou} \quad l = n-1 \tag{18.13}$$

— Si (18.13) est vérifié pour $l = n$ on accepte $t_{n,n}^{(k)}$ comme approximation de λ_n et on continue l'itération avec la matrice $\left(t_{i,j}^{(k)}\right)_{1\leq i,j\leq n-1}$

— Si (18.13) est vérifié pour $l = n-1$, on accepte les deux valeurs propres de (18.12) comme approximations de λ_n et λ_{n-1} et on continue l'itération avec la matrice $\left(t_{i,j}^{(k)}\right)_{1\leq i,j\leq n-2}$

Exemple 281. Nous avons appliqué l'algorithme QR à la matrice (**??**) avec le shift $p_k = t_{n,n}^{(k)}$. La convergence de $t_{i+1,i}^{(k)}$ vers zéro est illustrée dans la figure V.5. Une comparaison avec la figure V.4 nous montre que la convergence est beaucoup plus rapide (convergence quadratique). Après 5 itérations, on a $\left|t_{4,3}^{(k)}\right| \leq 10^{-15}$. Encore 4 itérations pour la matrice de dimension 3 donnent $\left|t_{3,2}^{(k)}\right| \leq 10^{-15}$ Il ne reste plus que 3 itérations à faire pour la matrice de dimension 2 pour avoir $\left|t_{2,1}^{(k)}\right| \leq 10^{-15}$. En tout, 12 itérations ont donné toutes les valeurs propres avec une précision de 15 chiffres.

18.6 Le "double shift" de Francis

Dans la situation où A est une matrice réelle ayant des valeurs propres complexes, il est recommandé de choisir un shift-paramètre p_k qui soit complexe. Une application directe de l'algorithme précédent nécessite un calcul avec des matrices complexes. L'observation suivante permet d'éviter ceci.

Lemme 282. Soit T_k une matrice réelle, $p_k = \alpha + i\beta$ et $p_{k+1} = \alpha - i\beta$.Alors, on peut choisir les décompositions dans l'algorithme QR de manière à ce que T_{k+2} soit réelle.

Remarque 283. La décomposition QR d'une matrice est unique sauf qu'on peut remplacer QR par $(QD)^{-1}\left(D^{-1}R\right)$ où $D = diag(d_1,...,d_n)$ avec $|d_i| = 1$.

Démonstration. La formule (**??**) montre que

$$T_{k+2} = (Q_{k+1}Q_{k+2})^* T_k (Q_{k+1}Q_{k+2}) \tag{18.14}$$

cqfd

Il suffit alors de démontrer que le produit $Q_{k+1}Q_{k+2}$ est réel. Une manipulation à l'aide de formules pour T_k donne

$$\begin{aligned} Q_{k+1}Q_{k+2}R_{k+2}R_{k+1} &= Q_{k+1}(T_{k+1} - p_{k+1}I)R_{k+1} = Q_{k+1}(R_{k+1}Q_{k+1} + p_{k+1}I - p_kI)R_{k+1} = \\ &= (Q_{k+1}R_{k+1})^2 + (p_k - p_{k+1})Q_{k+1}R_{k+1} = (T_k - p_kI)^2 + (p_k - p_{k+1})(T_k - p_kI) = \\ &= T_k^2 - (p_k + p_{k+1})T_k + p_kp_{k+1}I = M \end{aligned} \tag{18.15}$$

On a donc trouvé une décomposition QR de la matrice M qui, en conséquence des hypothèses du lemme, est une matrice réelle. Si, dans l'algorithme QR, la décomposition est choisie de manière à ce que les éléments diagonaux de R_{k+1} et R_{k+2} soient réels, alors, à cause de l'unicité de la décomposition QR, les matrices $Q_{k+1}Q_{k+2}$ et $R_{k+2}R_{k+1}$ sont réelles.

Une possibilité de calculer T_{k+2} à partir de T_k est de calculer de (18.15), de faire une décomposition QR (réelle) de M et de calculer T_{k+2} à l'aide de (18.14). Cet algorithme n'est pas pratique car le calcul de T_k^2 nécessite $O(n^3)$ opérations, même si T_k est sous forme de Hessenberg.

Il y a une astuce intéressante pour obtenir T_{k+2} à partir de T_k en $O(n^2)$ opérations. Elle est basée sur la propriété suivante.

Théorème 284. Soit une matrice donnée et supposons que

$$Q^*TQ = S \tag{18.16}$$

où Q est orthogonale et S est sous forme de Hessenberg satisfaisant $s_{i,i-1} \neq 0$ pour $i = 2,...,n$ Alors, Q et S sont déterminées de manière "unique" par la première colonne de Q.

Remarque 285. On a "unicité" dans le sens suivant : si $\hat{Q}^*T\hat{Q}$ est de type Hessenberg avec une matrice orthogonale $\hat{Q}$ satisfaisant $\hat{Q}e_1$, alors $\hat{Q} = QD$ où $D = diag(d_1,...,d_n)$ avec $|d_i| = 1$.

Démonstration. Notons les colonnes de Q par q_i. Alors, la relation (18.16) implique

$$Tq_i = \sum_{j=1}^{i+1} s_{ji}q_j, \qquad q_j^*Tq_i = s_{ji}. \tag{18.17}$$

Si q_1 est fixé, la valeur s_{11} est donnée par la deuxième formule de (18.17). Avec cette valeur, on obtient de la première formule de (18.17) que q_2 est un multiple de $Tq_1 - s_{11}q_1$. Ceci détermine q_2 à une unité près. Maintenant, les valeurs s_{21}, s_{12}, s_{22} sont déterminées et q_3 est un multiple de $Tq_2 - s_{21}q_1 - s_{22}q_2$ etc. cqfd

Si les hypothèses du lemme précédent sont vérifiées, on peut calculer la matrice réelle T_{k+2} en $O(n^2)$ opérations de la manière suivante :

— calculer $M\varepsilon_1$, la première colonne de M (formule (18.15));
— déterminer une matrice de Householder H_1 telle que $H_1(M\varepsilon_1) = \alpha e_1$
— transformer $H_1^T T_k H_1$ sous forme de Hessenberg à l'aide de matrices de Householder $H_2, ..., H_{n-1}$ (voir le paragraphe V.3); c'est-à-dire., calculer $H^T T_k H$ où $H = H_1 H_2 .... H_{n-1}$.

Comme $H_i e_1 = e_1$ pour $i = 2, ..., n-1$, la première colonne de H est un multiple de celle de M (observer $H_1^T = H_1$). Par la formule (18.15), la première colonne de $Q_{k+1}Q_{k+2}$ est aussi un multiple de Me_1. Par conséquent, pour un bon choix des décompositions $Q_{k+1}R_{k+1}$ et $Q_{k+2}R_{k+2}$ on a $H = Q_{k+1}Q_{k+2}$ la matrice obtenue par cet algorithme est égale à T_{k+2} (voir (18.14)).

18.7 Etude de la convergence

Supposons d'être déjà proche de la limite et considérons, par exemple, la matrice

$$T_0 = A = \begin{pmatrix} 2 & a \\ \varepsilon & 1 \end{pmatrix}$$

où ε est un nombre petit. Avec le choix $p_0 = 1$ pour le shift-paramètre, on obtient

$$T_0 - p_0 I = \begin{pmatrix} 1 & a \\ \varepsilon & 0 \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{1+\varepsilon^2}} & -\frac{\varepsilon}{\sqrt{1+\varepsilon^2}} \\ \frac{\varepsilon}{\sqrt{1+\varepsilon^2}} & \frac{1}{\sqrt{1+\varepsilon^2}} \end{pmatrix} \begin{pmatrix} \sqrt{1+\varepsilon^2} & \frac{a}{\sqrt{1+\varepsilon^2}} \\ 0 & -\frac{a\varepsilon}{\sqrt{1+\varepsilon^2}} \end{pmatrix} = Q_1 R_1$$

et

$$T_0 - p_0 I = R_1 Q_1 = \begin{pmatrix} * & * \\ -\frac{a\varepsilon^2}{1+\varepsilon^2} & * \end{pmatrix}$$

— si A est symétrique (c'est-à-dire, $a = \varepsilon$) on a $t^{(1)}_{n,n-1} = O(\varepsilon^2)$, donc convergence *cubique*.
— si A n'est pas symétrique (p.ex. on a donc convergence quadratique.

Ces propriétés restent vraies pour des matrices générales (sans démonstration).

19 Exercices

Exercice 286. Calculer les valeurs propres de la matrice tridiagonale (dimension $n, b.c > 0$)

$$A = \begin{pmatrix} a & c & & & \\ b & a & c & & \\ & b & a & c & \\ & & b & \ddots & \ddots \\ & & & \ddots & \end{pmatrix}$$

Indication. Les composants du vecteur propre $(v_1, v_2, ..., v_n)^T$ satisfont une équation aux différences finies avec.$v_0 = v_{n+1} = 0$ Vérifier que $v_j = Const.(\alpha_1^j - \alpha_2^j)$ où

$$\alpha_1 - \alpha_2 = \frac{\lambda - a}{c}, \qquad \alpha_1.\alpha_2 = \frac{b}{c}, \qquad \left(\frac{\alpha_1}{\alpha_2}\right)^{n+1} = 1$$

Résultat : $\lambda_j = a - 2\sqrt{bc}.\cos\left(\frac{j\pi}{n+1}\right), \qquad j = 1, 2, .., n.$

Exercice 287. Considérer la matrice

$$A(\varepsilon) = \begin{pmatrix} 1 & \varepsilon & 0 \\ -1 & 0 & 1 \\ 1 & -1+\varepsilon & -\varepsilon \end{pmatrix}$$

cette matrice possède une valeur propre de la forme

$$\lambda(\varepsilon) = i + \varepsilon.d + O(\varepsilon^2)$$

Calculer d et dessiner la tangente à la courbe $\lambda(\varepsilon)$ au point $\lambda(0)$

(a) Calculer par la méthode de la puissance, la plus grande valeur propre de la matrice

$$A = \begin{pmatrix} 99 & 1 & 0 \\ 1 & 100 & 1 \\ 0 & 1 & 98 \end{pmatrix}$$

(b) Pour accélerer considérablement la vitesse de convergence, appliquer la méthode de la puissance à la matrice $A - pI$ avec un choix intelligent de p.

(c) Avec quel choix de p obtient-on la valeur propre la plus petite ?

Exercice 288. Considérons la matrice tridiagonale

$$A = \begin{pmatrix} b_1 & c_1 & \\ a_1 & b_2 & c_2 \\ & a_2 & \\ & & \ddots \end{pmatrix}$$

Montrer que, si $a_i c_i > 0$ pour $i = 1, ..., n-1$, toutes les valeurs propres de A sont réelles.

Indication. Trouver $D = diag(d_1, ..., d_n)$ telle que DAD^{-1} soit symétrique.

Exercice 289. Soit A une matrice symétrique et B quelconque. Montrer que pour chaque valeur propre λ_B de B il existe une valeur propre λ_A de A telle que

$$|\lambda_A - \lambda_B| \leq \|A - B\|_2 .$$

Indication. Montrer l'existence d'un vecteur v tel que $v = (A - \lambda_B)^{-1}(A - B)v$. En déduire que $1 \leq \left\|(A - \lambda_B)^{-1}(A - B)\right\| \leq \left\|(A - \lambda_B)^{-1}\right\| \|(A - B)\|$.

Exercice 290. (Schur, 1909). Soit A une matrice symétrique. Montrer que pour chaque indice i il existe une valeur propre λ de A telle que

$$|\lambda - a_{ii}| \leq \sqrt{\sum_{j \neq i} |a_{ij}|^2}$$

Indication. Appliquer l'exercice 5 avec une B convenable.

Exercice 291. Soit A une matrice réelle avec pour valeur propre $\alpha + i\beta$. Montrer que l'itération

$$\begin{pmatrix} \bar{\alpha} I - A & -\bar{\beta} I \\ \bar{\beta} I & \bar{\alpha} I - A \end{pmatrix} \begin{pmatrix} u_{k+1} \\ v_{k+1} \end{pmatrix} = \begin{pmatrix} u_k \\ v_k \end{pmatrix}$$

où $\bar{\alpha} \approx \alpha$ et $\bar{\beta} \approx \beta$) permet de calculer la valeur propre $\alpha + i\beta$ et le vecteur propre correspondant.
Indication. Considérer les parties réelles et complexes de l'itération de Wielandt. On obtient alors

$$\frac{u_k^T A u_k + v_k^T A v_k}{u_k^T u_k + v_k^T v_k} \longrightarrow \alpha, \qquad \frac{u_k^T A v_k + v_k^T A u_k}{u_k^T u_k + v_k^T v_k} \longrightarrow \beta$$

Exercice 292. Considérons la matrice de Hilbert,

$$A = \begin{pmatrix} 1/2 & 1/3 & 1/4 \\ 1/3 & 1/4 & 1/5 \\ 1/4 & 1/5 & 1/6 \end{pmatrix}$$

(a) Transformer A en une matrice tridiagonale ayant les mêmes valeurs propres.

(b) En utilisant une suite de Sturm, montrer que toutes les valeurs propres sont positives et qu'une valeur propre est plus petite que 0.001

(c) Calculer approximativement la condition de A pour la norme Euclidienne.

Exercice 293. La formule de récurrence

$$(k+1)P_{k+1}(x) = (2k+1)xP_k(x) - kP_{k-1}(x)$$

pour les *polynômes de Legendre* ressemble à

$$p_i(\lambda) = (d_i - \lambda)p_{i-1}(\lambda) - \varepsilon_i^2 p_{i-2}(\lambda), \qquad i = 2,...,n.$$

pour les polynômes $\det(A_i - \lambda I)$. Trouver une matrice tridiagonale A de dimension n telle que les valeurs propres de A sont les racines de $P_n(x)$.

Exercice 294. Soit $p(x)$ un polynôme de degré n et supposons que toutes les racines soient simples. Démontrer que la suite définie par l'algorithme d'Euclide,

$$\begin{aligned} p_n(x) &= p(x), \qquad p_{n-1}(x) = -p'(x) \\ p_i(x) &= q_i(x)p_{i-1}(x) - \gamma_i^2 p_{i-2}(x), \qquad i = n,...,2. \end{aligned}$$

est une suite de Sturm.

Pour le polynôme $p(x) = x^5 - 6x^4 + 3x^3 + 3x^2 + 2x + 8$.

(a) déterminer le nombre de racines réelles.

(b) Combien de racines sont complexes ?

(c) Combien de racines sont réelles et positives ?

Exercice 295. Pour un φ donné notons $c = \cos\varphi$ et $s = \sin\varphi$.La matrice Ω_{kl}, définie par

$$(\Omega_{kl})_{ij} = \begin{cases} 1 & \text{si } i = j, j \neq k, j \neq l \\ c & \text{si } i = j = k, \text{ ou } i = j = l \\ s & \text{si } i = k, \text{ et } j = l \\ -s & \text{si } i = l, \text{ et } j = k \\ 0 & \text{sinon} \end{cases}$$

s'appelle rotation de Givens.

(a) Montrer qu'elle est orthogonale.

(b) Soit A une matrice symétrique.Déterminer φ tel que le (k,l)-ième élément de $A' = \Omega_{kl} A \Omega_{kl}^T$ s'annule.

Resultat. $\cot 2\varphi = (a_{kk} - a_{ll})/(2a_{kl})$.

Exercice 296. La méthode de Jacobi (1846) pour le calcul des valeurs propres d'une matrice symétrique :

i) on choisit a_{kl} $(k > l)$ tel que $|a_{kl}| = \max_{i>j} |a_{ij}|$;

ii) on détermine A' comme dans l'exercice 11.

Montrer que, si on répète cette procédure, on a convergence vers une matrice diagonale, dont les éléments sont les valeurs propres de A

Indication. Montrer que $\sum_{i>j} |a'_{ij}|^2 = \sum_{i>j} |a_{ij}|^2 - |a_{kl}|^2$

Exercice 297. On considère la matrice

$$A = \begin{pmatrix} 7 & 0,5 \\ 0,0001 & 8 \end{pmatrix}$$

dont on cherche à calculer les valeurs propres.

(a) Faire une itération de l'algorithme QR sans shift.

(b) Faire une itération de l'algorithme QR avec shift.

(c) Estimer la position des valeurs propres de A à l'aide du Théorème de Gershgorin.

(d) Calculer les valeurs propres de A à l'aide du polynôme caractéristique.

Exercice 298. Montrer que si la matrice $T_0 = A$ est une matrice de Hessenberg (ou tridiagonale), alors les matrices T_k, $k \geq 1$ construites par l'algorithme QR sont également des matrices de Hessenberg (tridiagonales).

Exercice 299. Donner une estimation grossière du nombre d'opérations qui sont nécessaires pour effectuer la décomposition QR d'une matrice de Hessenberg et pour calculer ensuite le produit RQ.

Exercice 300. Soit T_0 une matrice de Hessenberg dont tous les éléments de la sous-diagonale sont non-nuls. Montrer que, si p_0 est une valeur propre de T_0, une itération de l'algorithme QR avec shift p_0 donne

$$t^{(1)}_{n,n-1} = 0.$$

Exercice 301. Expliquer, comment le calcul de T_k à partir de T_{k-1}

$$Q_k R_k = T_{k-1} - p_{k-1} I, \qquad T_k = R_k Q_k + p_{k-1} I.$$

peut être effectué sans soustraire (et additionner) explicitement la matrice $p_{k-1} I$

Indication. Laissez-vous inspirer par le "double shift" algorithme de Francis.