KU Leuven
Faculteit Ingenieurswetenschappen

Numerieke Modellering en Benadering

Practicum 1

Stijn Kuypers
Dries De Backker

Academiejaar 2015-2016

1 Projectoren en de QR factorisatie

Opgave 1

Zij $P \in \mathbb{R}^{n\times n}$ en $x \in \mathbb{R}^n$, dan $Px = \frac{x+Fx}{2}$ met $F \in \mathbb{R}^{n\times n}$ de matrix die de elementen x_1 en x_2, x_3 en x_4, ... omwisselt bij de vermenigvuldiging Fx. Dan:

$$Px = \frac{1}{2}(I+F)x \Leftrightarrow P = \frac{1}{2}(I+F) \tag{1}$$

* P is een projector indien P vierkant(gegeven) en P idempotent ($P^2 = P$). We hebben:

$$P^2 = \frac{1}{4}(I^2 + 2F + F^2) \tag{2}$$

Aangezien F het eerste en tweede element, het derde en vierde element enz... van een vector omwisselt, doet een tweede vermenigvuldiging met F deze permutatie terug teniet. Dus: $F^2 = I$. Dan:

$$P^2 = \frac{1}{4}(I + 2F + I) = \frac{1}{4}(2I + 2F) = \frac{1}{2}(I+F) = P \tag{3}$$

* P is een orthogonale projector $\Leftrightarrow P = P^*$. We hebben:

$$P^* = \frac{1}{2}(I+F)^* = \frac{1}{2}(I^* + F^*) \tag{4}$$

We kunnen nagaan dat:

$$F = \begin{bmatrix} 0 & 1 & & & & & \\ 1 & 0 & & & & & \\ & & 0 & 1 & & & \\ & & 1 & 0 & & & \\ & & & & \ddots & & \\ & & & & & 0 & 1 \\ & & & & & 1 & 0 \end{bmatrix} \tag{5}$$

zodat:

$$F^* = \begin{bmatrix} 0 & 1 & & & & & \\ 1 & 0 & & & & & \\ & & 0 & 1 & & & \\ & & 1 & 0 & & & \\ & & & & \ddots & & \\ & & & & & 0 & 1 \\ & & & & & 1 & 0 \end{bmatrix} \tag{6}$$

en dus:

$$P^* = \frac{1}{2}(I^* + F^*) = \frac{1}{2}(I+F) = P \tag{7}$$

waaruit volgt dat P een orthogonale projector is.

Opgave 2

Een Householder transformatiematrix heeft de volgende structuur:

$$Q_k = \begin{bmatrix} I & \\ & F \end{bmatrix} \tag{8}$$

met I de $(k-1) \times (k-1)$ eenheidsmatrix en F een $(m-k+1) \times (m-k+1)$ unitaire Householder reflector. Deze structuur impliceert een karakteristieke veelterm van de vorm:

$$det(\lambda I - Q_k) = (\lambda - 1)^k \cdot det(\lambda I - F) \tag{9}$$

Voor $k \geq 1$ hebben we dus steeds eigenwaarde 1 in het spectrum van Q_k met algebraïsche multipliciteit k. Aangezien dit rechtstreeks voortkomt uit het diagonaal zijn van $\begin{bmatrix} I & 0 \end{bmatrix}^T$, is k eveneens de geometrische multipliciteit. Een diagonale matrix is immers nooit defectief. We kunnen dan k lineair onafhankelijke eigenvectoren vinden voor $\lambda = 1$. Dit is gemakkelijk te verifiëren. Neem gewoon:

$$v_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \quad v_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \quad v_3 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix}, \quad \ldots \quad v_k = \ldots \tag{10}$$

Dan $Q_k v_j = v_j \quad (\forall j \leq k)$. Deze vectoren spannen de ruimte $\mathbb{R}^k$ op waartoe alle vectoren van de eerste k rijen van een matrix $A \in \mathbb{R}^{m \times n}$ behoren (aangevuld tot dimensie m met nulle. Vermits elke kolomvector van A tot en met rij k (en vervolgens aangevuld met nullen tot dimensie m) een lineaire combinatie is van bovenstaande v_j zal een vermenigvuldiging met Q_k neerkomen op een vermenigvuldiging met $\lambda = 1$. Householder laat dus de eerste k rijen van de matrix A ongemoeid zoals men hoopt te verwachten van een Householder transformatiematrix.

De Householder reflector F zal dan zorgen voor de nodige nullen zonder voorheen aangebrachte nullen te vernietigen.

Opgave 3

a deel a

Voor de uitwerking van de drie gevraagde functies verwijzen we naar de bijgevoegde MATLAB bestanden.

b deel b

Morbi luctus, wisi viverra faucibus pretium, nibh est placerat odio, nec commodo wisi enim eget quam. Quisque libero justo, consectetuer a, feugiat vitae, porttitor eu, libero. Suspendisse sed mauris vitae elit sollicitudin malesuada. Maecenas ultricies eros sit amet ante. Ut venenatis velit. Maecenas sed mi eget dui varius euismod. Phasellus aliquet volutpat odio. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Pellentesque sit amet pede ac sem eleifend consectetuer. Nullam elementum, urna vel imperdiet sodales, elit ipsum pharetra ligula, ac pretium ante justo a nulla. Curabitur tristique arcu eu metus. Vestibulum lectus. Proin mauris. Proin eu nunc eu urna hendrerit faucibus. Aliquam auctor, pede consequat laoreet varius, eros tellus scelerisque quam, pellentesque hendrerit ipsum dolor sed augue. Nulla nec lacus.

2 Iteratieve Methoden

Opgave 4

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Opgave 5

a

Het kan aangetoond worden dat het QR-algoritme (zonder shifts) equivalent is met simultane iteratie. Het QR-algoritme maakt(veelal) gebruik van het rayleighquotient bij zijn shifts. Rayleighiteratie gebruikt het rayleighquotient afgewisseld met inverse iteratie.

Zowel het QR-algoritme als simultane iteratie berekenen meerdere eigenwaarden. Rayleighquotientiteratie berekent slechts de één enkele eigenwaarde, afhankelijk van de starvector.

Opgave 6

Sed commodo posuere pede. Mauris ut est. Ut quis purus. Sed ac odio. Sed vehicula hendrerit sem. Duis non odio. Morbi ut dui. Sed accumsan risus eget odio. In hac habitasse platea dictumst. Pellentesque non elit. Fusce sed justo eu urna porta tincidunt. Mauris felis odio, sollicitudin sed, volutpat a, ornare ac, erat. Morbi quis dolor. Donec pellentesque, erat ac sagittis semper, nunc dui lobortis purus, quis congue purus metus ultricies tellus. Proin et quam. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Praesent sapien turpis, fermentum vel, eleifend faucibus, vehicula eu, lacus.

Opgave 7

Sed commodo posuere pede. Mauris ut est. Ut quis purus. Sed ac odio. Sed vehicula hendrerit sem. Duis non odio. Morbi ut dui. Sed accumsan risus eget odio. In hac habitasse platea dictumst. Pellentesque non elit. Fusce sed justo eu urna porta tincidunt. Mauris felis odio, sollicitudin sed,

volutpat a, ornare ac, erat. Morbi quis dolor. Donec pellentesque, erat ac sagittis semper, nunc dui lobortis purus, quis congue purus metus ultricies tellus. Proin et quam. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Praesent sapien turpis, fermentum vel, eleifend faucibus, vehicula eu, lacus.

Opgave 8

Sed commodo posuere pede. Mauris ut est. Ut quis purus. Sed ac odio. Sed vehicula hendrerit sem. Duis non odio. Morbi ut dui. Sed accumsan risus eget odio. In hac habitasse platea dictumst. Pellentesque non elit. Fusce sed justo eu urna porta tincidunt. Mauris felis odio, sollicitudin sed, volutpat a, ornare ac, erat. Morbi quis dolor. Donec pellentesque, erat ac sagittis semper, nunc dui lobortis purus, quis congue purus metus ultricies tellus. Proin et quam. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Praesent sapien turpis, fermentum vel, eleifend faucibus, vehicula eu, lacus.

Opgave 9

Het is mogelijk om Jacobi rotaties in parallel uit te voeren door ze op disjuncte rij-/kolomparen te laten inwerken. Zij A een symmetrische 4×4 matrix, dan kunnen we dit parallelisme voorstellen door twee rotaties in een enkele rotatiematrix te steken.
We hebben bv. de twee rotaties:

$$J_1 = \begin{bmatrix} c1 & 0 & s1 & 0 \\ 0 & 0 & 0 & 0 \\ -s1 & 0 & c1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad J_2 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & c2 & 0 & s2 \\ 0 & 0 & 0 & 0 \\ 0 & -s2 & 0 & c2 \end{bmatrix} \tag{11}$$

Deze werken in op de disjuncte rij-/kolomparen (1,3) en (2,4) en kunnen dus tegelijk uitgevoerd worden. Dit kunnen we voorstellen door:

$$R = \begin{bmatrix} c1 & 0 & s1 & 0 \\ 0 & c2 & 0 & s2 \\ -s1 & 0 & c1 & 0 \\ 0 & -s2 & 0 & c2 \end{bmatrix} \tag{12}$$

De transformatie $R^T AR$ zal dan nullen introduceren op posities (1,3), (3,1) en (2,4), (4,2) van de marix A. Voor een symmetrische matrix van dimensie $n \times n$ en n even, kunnen $n/2$ van zulke disjuncte rij-/kolomparen gevonden worden waardoor er $n/2$ operaties in parallel kunnen plaatsvinden.

Opgave 10

Het volgende algoritme in pseudocode brengt nullen aan op achtereenvolgens (1,2), (1,3), ... , (1,n); (2,3), (2,4), ... , (2,n); ... ; (n-2,n-1), (n-2,n) ; (n-1,n) en op alle elementen hiermee symmetrisch t.o.v. de hoofddiagonaal.

$Jacobi(A, tolerance)$ {

$n = width(A)$;
$J = I(n, n)$;
$imprecision = 100$;

while $(tolerance < imprecision)$ {
for $(i = 1; i = n - 1; i{+}{+})$ {
for $(j = i + 1; j = m; j{+}{+})$ {
$a = A(i, i)$;
$b = A(i, j)$;
$\theta = (1/2) \cdot atan(2d/(b - a))$;
$s = sin(\theta)$;
$c = cos(\theta)$;
$J_k = I(n, n)$;
$J_k(i, i) = c$;
$J_k(j, j) = c$;
$J_k(i, j) = s$;
$J_k(j, i) = -s$
$J = J \cdot J_k$;
$A = (J_k)^T \cdot A \cdot J_k$;
}
}

$offdiagnorm = (sumofsquares(A) - sumofdiagonalsquares(A))^{1/2}$;
$imprecision = |offdiagsum/maxdiagonal(A)|$;
}

$D = A$;
$V = J$;
return V, D;

De implementatie in MATLAB volgt bovenstaande structuur en is te vinden in het bijgeleverde bestand 'jacobi.m'.

Opgave 11

De cursus geeft aan dat de de convergentie van een symmetrische matrix met dimensie $m \leq 1000$ typisch gebeur t in minder dan tien sweeps. Dit lijkt te kloppen. De bijgevoegde matrix mat1 convergeert in 8 sweeps. De norm van de niet-diagonaalelementen, gedeeld door het grootste diagonaalelement, wordt dan immers volledig nul in MATLAB (m.a.w. de fout is van $O(\epsilon_{mach})$).

Voor een willekeurige symmetrische tridiagonale matrix is de rekenkost van dit algoritme dezelfde als van een willekeurige symmetrische niet-tridiagonale matrix, aangezien het algoritme met de structuur van de matrix, afgezien van de noodzakelijk symmetrie, geen rekening houdt. Indien we een variant zouden maken, enkel voor tridiagonale matrices, gebeurt een sweep in m-1 iteraties, nl. het aantal elementen op de diagonaal juist boven of onder de hoofddiagonaal. De rekenkost is dan $O(m)$.