Institut für Analysis und Numerik
Universität Magdeburg
Thomas Richter
Henry von Wahl

24.04.2020

thomas.richter@ovgu.de
henry.vonwahl@ovgu.de

Blatt 2 zur Vorlesung Numerische Mathematik
Sommersemester 2020

Aufgabe 2.1

Man erstelle die QR-Zerlegung von

$$A = \begin{pmatrix} 60 & 30 & 20 \\ 30 & 20 & 15 \\ 20 & 15 & 12 \end{pmatrix}$$

mit Hilfe der Householder-Transformationen. Dabei verwende man dreistellige Arithmethik. Man mache die Proble auf Orthogonalität und berechne $I - Q^T Q$.

Aufgabe 2.2

Ein stark vereinfachtes Modell zur Beschreibung der aktiven Fälle einer Pandemie ist durch

$$f_n = \alpha \exp\Big((\beta - \gamma n) n\Big)$$

gegeben. Dabei sind f_n die aktiven Fälle an Tag n, β beschreibt die ungebremste Ausbreitungsrate des Virus, γ beschreibt die Gesamtheit aller Faktoren, die zu einem Rückgang der Ausbreitung führt, α ist ein Normierungsfaktor.

a) Man überführe das Modell in einen linearen Zusammenhang zwischen den Unbekannten. Hierzu vergleiche man Beispiel *Exkurs 4.6: Astronomische Hauptachsenbestimmung eines Himmelskörpers*. Tipp: Logarithmieren.

b) Man erstelle ein lineares Gleichungssystem zur Bestimmung der Unbekannten auf Basis der Daten

Tag	1	4	14	23	32	35	43	48
Fälle	114	550	7057	32686	65130	72800	60387	53017

Man berechne die Least-Squares Lösung. Hierbei darf entweder das Normalgleichungssystem aufgestellt und gelöst werden, oder aber die Lösung direkt über die QR-Zerlegung bestimmt werden.

Hinweis: Beim Lösen und Aufstellen der Gleichungssysteme darf der Computer zur Hilfe genommen werden. Die wesentlichen Zwischenschritte sollen aber nachvollziehbar sein.

Programmieraufgabe 2.3

Für eine Matrix $A \in \mathbb{R}^{n \times n}$ erstelle man die QR-Zerlegung. Eingabe ist die Matrix A, Ausgabe ist die Matrix R sowie eine Matrix V, die alle Spiegelungsvektoren enthält. Die Programmvorlage `template_02.py` mit einem Algorithmus-Vorschlag folgt bis zum 26. April auf der Homepage.

a) Für die angegebene 3×3 Testmatrix gebe man die reduzierte Matrix R sowie die Spiegelungsvektoren aus.

b) Man schreibe eine Funktion `erstelleQ(V)`, welche aus den Spiegelungsvektoren die orthogonale Matrix Q rekonstruiert. Man gebe diese aus.

Abgabe per Mail (**eine Abgabe pro 5er Gruppe**) an Henry von Wahl (henry.vonwahl@ovgu.de). Abgabe bis zum jeweils folgenden Freitag.

Algorithmische Umsetzung der QR-Zerlegung

Gundlegende Idee Es sei $A \in \mathbb{R}^{n\times n}$ gegeben. Die QR-Zerlegung wird in n (es reichen eigentlich $n-1$) Schritte durchgeführt:

$$\begin{aligned} R^{(0)} &:= A \\ R^{(1)} &:= S^{(1)}R^{(0)} \\ R^{(2)} &:= S^{(2)}R^{(1)} \\ &\vdots \\ R^{(n)} &:= S^{(n)}R^{(n-1)} \end{aligned} \tag{1}$$

Dabei wird die Householder-Matrix $S^{(k)}$ als

$$S^{(k)} = I - 2v^{(k)}v^{(k),T} \tag{2}$$

gebildet. Der Spiegelungsvektor $v^{(k)}$ hängt von der Matrix $R^{(k-1)}$ ab.
Aufbau der Spiegelungsvektoren Hier ist die Umsetzung nicht trivial, da nur mit Teil-Vektoren gearbeitet wird. Zunächst sei $R^{(k-1)} = (r_1^{(k-1)}, \dots, r_n^{(k-1)})$ die Darstellung von $R^{(k-1)}$ in Spaltenvektoren. Für einen Vektor $v \in \mathbb{R}^n$ definieren wir

$$\mathcal{I}_k v := \begin{pmatrix} 0 \\ \vdots \\ 0 \\ [v_k]_k \\ [v_k]_{k+1} \\ \vdots \\ [v_k]_n \end{pmatrix}, \tag{3}$$

d.h., die ersten $k-1$ Einträge von v werden zu Null gesetzt. Hiermit wird $v^{(k)}$ mit folgendem Algorithmus definiert:

$$\begin{aligned} v^{(k)} &:= \mathcal{I}_k r_k^{(k-1)} \\ n^{(k)} &:= \|v^{(k)}\|_2 \\ \text{falls } [v^{(k)}]_k > 0 \quad &\Rightarrow \quad [v^{(k)}]_k := [v^{(k)}]_k + n^{(k)}, \\ \text{sonst} \quad &\Rightarrow \quad [v^{(k)}]_k := [v^{(k)}]_k - n^{(k)} \\ v^{(k)} &:= v^{(k)}/\|v^{(k)}\|_2 \end{aligned} \tag{4}$$

Aufbau von R Die QR-Zerlegung wird jedoch nicht direkt nach (1) durchgeführt. Die Matrizen $S^{(k)}$ werden nicht aufgebaut. Stattdessen speichern wir die Vektoren $v^{(k)}$ und führen die Matrix-Matrix Multiplikationen $R^{(k)} = S^{(k)}R^{(k-1)}$ Spaltenweise durch. Es gilt mit $R_l^{(k)} = [S^{(k)}R^{(k-1)}]_l$ zunächst

$$R_l^{(k)} = R_l^{(k-1)} \text{ für } l = 1, \dots, k-1,$$

da der obere $l \times l$-Block von $S^{(k)}$ die Einheitsmatrix ist.
Für die k-te Spalte gilt nach Konstruktion

$$[R_k^{(k)}]_l = 0 \text{ für } l = k+1, \ldots, n \tag{5}$$

und ebenso für das Diagonalelement

$$[R_k^{(k)}]_k = \begin{cases} -n^{(k)} & [R_k^{(k-1)}]_k \geqslant 0 \\ n^{(k)} & [R_k^{(k-1)}]_k < 0 \end{cases} \tag{6}$$

wobei $n^{(k)}$ die oben berechnete Norm des verkürzten k-ten Spaltenvektor von $R^{(k-1)}$ ist. Damit sind die ersten k Spaltenvektoren von $R^{(k)}$ bestimmt. Für die weiteren gilt

$$R_l^{(k)} = (I - 2v^{(k)}v^{(k),T})R_l^{(k-1)}, \quad l = k+1, \ldots, n.$$

Diese Vektoren werden nicht als Matrix-Vektor Produkt ausgerechnet sondern als:

$$R_l^{(k)} = R_l^{(k-1)} - 2v^{(k)}\langle v^{(k)}, R_l^{(k-1)}\rangle_2, \quad l = k+1, \ldots, n. \tag{7}$$

Zusammenfassung $R := A$. (In Python verwende man `R=A.copy()`. Ansonsten sind `R` und `A` lediglich unterschiedliche Namen für das gleiche Objekt. D.h. ändert man `R`, würde sich auch `A` ändern.)
Man legt eine Matrix `V=np.zeros( (n,n))` mit `n=len(A)` zum Speichern der Spiegelungs-Vektoren an.
Für $k = 1, 2, \ldots, n$ (in Python achte man auf die Index-Verschiebung ab 0)

1. Berechnet v als den k-ten Spiegelungsvektor gemäß (4), achte dabei auf das Verkürzen gemäß (3).
2. Man merke sich v in `V`, z.B. effizient als `V[:,k]=v`
3. Transformation von `R` gemäß
   a) Keine Änderung in Spalten $1, \ldots, k-1$
   b) Änderung der Spalte k gemäß (5) und (6).
   c) Änderung von Spalten $k+1, \ldots, b$ gemäß (7)

In `template_02.py` sind einige Tips zum Umgang mit numpy-Vektoren und Matrizen gegeben.