Mathematik für InformatikerInnen 2

Frank-Olaf Schreyer

Universität des Saarlandes, SS 2020

Numerische Verfahren

Die Themen heute sind

- LR-Zerlegung
- Kondition einer Matrix
- QR-Zerlegung

Heute wollen wir uns mit numerischen Verfahren beschäftigen. Schon in der ersten Vorlesung hatten wir gesehen, dass der Gauß-Algorithmus, wenn wir mit Gleitkommazahlen (floating point numbers) rechnen, dramatische Fehler produzieren kann. Der Ausweg ist eine Pivotstrategie, die wir heute genauer besprechen wollen.

Gauß-Algorithmus mit Spaltenpivotstrategie

Der Algorithmus bringt eine invertierbare Matrix $A \in \mathbb{R}^{n \times n}$ sukzessive

$$A = A^{(1)} \rightsquigarrow A^{(2)} \rightsquigarrow \ldots \rightsquigarrow A^{(n)}$$

in Zeilenstufenform.

a) Im k-ten Eliminationsschritt wählen wir ein $p \in \{k, \ldots, n\}$, so dass

$$|a_{pk}^{(k)}| \geqslant |a_{jk}^{(k)}| \quad \forall j \in \{k, \ldots, n\}$$

gilt.

b) Vertauschen die p-te mit der k-ten Zeile.

c) Eliminieren Elemente unterhalb des k-ten Eintrags in der k-ten Spalte.

$$\Rightarrow \quad A^{(k+1)} = L_k P_k A^{(k)},$$

wobei P_k eine Permutationsmatrix und L_k eine untere Dreiecksmatrix ist.

Genauer gilt:

$$A^{(k+1)} = L_k P_k A^{(k)},$$

wobei

$$P_k = \begin{pmatrix} 1 & & & \\ & & 1 & \\ & 1 & & \\ & & & 1 \end{pmatrix}$$

die Permutationsmatrix zur Transposition $(p\,k)$ ist und

$$L_k = \begin{pmatrix} 1 & & & \\ & \ddots & & \\ & & 1 & \\ & & \ell_{k+1,k} & 1 \\ & & \vdots & & \ddots \\ & & \ell_{n,k} & & & 1 \end{pmatrix}$$

eine untere Dreiecksmatrix mit $|\ell_{ij}| \leqslant 1$ und Einsen auf der Diagonalen ist.

LR-Zerlegung

Folgerung. *Sei $A \in \mathrm{GL}(n, \mathbb{R})$. Dann existiert eine Permutationsmatrix P, eine untere Dreiecksmatrix*

$$L = \begin{pmatrix} 1 & & & 0 \\ \ell_{21} & 1 & & \\ \vdots & & \ddots & \\ \vdots & & & \ddots \\ \ell_{nn} & \dots & \ell_{n-1,n} & 1 \end{pmatrix} \quad \text{mit } |\ell_{ij}| \leqslant 1$$

mit Einsen auf der Diagonalen und eine obere Dreiecksmatrix

$$R = \begin{pmatrix} r_{11} & \dots & r_{1n} \\ & \ddots & \vdots \\ 0 & & r_{nn} \end{pmatrix},$$

so dass

$$PA = LR$$

gilt.

Ende Teil 1

Beweis. $P = P_{n-1} \cdots P_2 P_1$ ist das Produkt der Permutationsmatrizen aus dem obigen Algorithmus, und R ist die obere Dreiecksmatrix, die wir schließlich erhalten. Da für $k < j$ sich

$$\hat{L}_k = P_j L_k P_j$$

von L_k nur durch Reihenfolge der Einträge in der k-ten Spalte unterhalb des Diagonalelements unterscheidet, lässt sich das Ergebnis des Gauß-Algorithmus

$$L_{n-1} P_{n-1} \cdot \ldots \cdot L_2 P_2 L_1 P_1 A = R$$

umformen zu

$$\tilde{L}_{n-1} \tilde{L}_{n-2} \cdot \ldots \tilde{L}_1 PA = R,$$

wobei $\tilde{L}_k$ sich von L_k nur durch die Reihenfolge der Einträge in der k-ten Spalte unterhalb des Diagonalelements unterscheidet. Es folgt

$$PA = (\tilde{L}_1^{-1} \cdot \ldots \tilde{L}_{n-2}^{-1} \tilde{L}_{n-1}^{-1}) R.$$

Da

$$\begin{pmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & \tilde{\ell}_{k+1,k} & 1 & \\ & & \vdots & & \ddots \\ & & \tilde{\ell}_{n,k} & & 1 \end{pmatrix}^{-1} = \begin{pmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & -\tilde{\ell}_{k+1,k} & 1 & \\ & & \vdots & & \ddots \\ & & -\tilde{\ell}_{n,k} & & 1 \end{pmatrix}$$

gilt, folgt

$$L = \tilde{L}_1^{-1} \cdot \ldots \cdot \tilde{L}_{n-1}^{-1}$$

hat die gewünschte Gestalt:

$$L = \tilde{L}_1^{-1} \cdot \ldots \cdot \tilde{L}_{n-1}^{-1}$$

ist

$$L = \begin{pmatrix} 1 & & & 0 \\ -\tilde{\ell}_{21} & 1 & & \\ \vdots & & \ddots & \\ \vdots & & & \ddots \\ -\tilde{\ell}_{nn} & \ldots & -\tilde{\ell}_{n-1,n} & 1 \end{pmatrix} \text{ mit } |\tilde{\ell}_{ij}| \leqslant 1.$$

□

Check:

$$\begin{pmatrix} 1 & 0 & 0 \\ a & 1 & 0 \\ b & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & c & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ a & 1 & 0 \\ b & c & 1 \end{pmatrix}$$

Wenn man diese Rechnung als eine Rechnung mit Blockmatrizen auffasst, kommt man auf die Idee, wie man die Formel für das Produkt mit Induktion beweisen kann.

Kondition einer Matrix

Wir hatten zu einer Vektornorm $\| \ \|$ auf $\mathbb{R}^n$ die zugehörge Matrixnorm auf $\mathbb{R}^{n\times n}$ durch

$$\|A\| = \max_{\|x\|=1} \|Ax\|$$

definiert.

Definition. Die **Kondition** einer Matrix $A \in \mathrm{GL}(n, \mathbb{R})$ ist

$$\kappa(A) = \|A\|\|A^{-1}\|.$$

Es gilt $\kappa(A) \in [1, \infty[$, da $1 = \|E\| = \|AA^{-1}\| \leqslant \|A\|\|A^{-1}\|$ wegen der Submultiplizität von Matrixnormen.

Mit Hilfe der mehrdimensionalen Analysis lässt sich zeigen, dass die Kondition ein gutes Maß dafür ist, wie fehleranfällig numerische Rechnungen mit A sind. Je näher die Kondition bei 1 liegt, um so besser ist die Situation.

Beispiele. Wir nehmen als Vektornorm im folgenden jeweils die euklidische Nom.

1. Orthogonale Matrizen haben optimale Norm 1.
2. Symmetrische invertierbare Matrizen A haben die Norm

$$\|A\| = \lambda_{\max},$$

wobei $\lambda_{\max} = \max\{|\lambda_k| \mid \lambda_k \text{ ist Eigenwert von } A\}$ ist. Die Inverse einer invertierbaren symmetrischen Matrix ist ebenfalls symmetrisch, und deren Eigenwerte sind

$$\{\frac{1}{\lambda_1}, \ldots, \frac{1}{\lambda_n}\},$$

da $UAV = D \Rightarrow V^t A^{-1} U^t = D^{-1}$ gilt. Es folgt

$$\kappa(A) = \frac{\lambda_{\max}}{\lambda_{\min}}$$

ist der Quotient aus den Beträgen des betragsmäßig größten und kleinsten Eigenwerts.

3. Allgemeiner gilt

$$\kappa(A) = \frac{\max_{\|x\|=1} \|Ax\|}{\min_{\|x\|=1} \|Ax\|}.$$

   Die Kondition misst also, inwieweit Längenverhältnisse durch A verzerrt werden.

Aus dieser Sicht ist die Matrix L in der LR-Zerlegung einer Matrix $A \in \mathrm{GL}(n, \mathbb{R})$ nicht optimal, da wir lediglich

$$\kappa(L) \leqslant \sqrt{n}$$

haben.
Besser wäre es, eine orthogonale Matrix zu verwenden. Die QR-Zerlegung tut dies.

Ende Teil 2

QR-Zerlegung

Satz. *Sei $A \in GL(n, \mathbb{R})$. Dann existiert eine orthogonale Matrix $Q \in O(n)$ und eine obere Dreiecksmatrix $R \in \mathbb{R}^{n \times n}$, so dass*

$$A = QR$$

gilt. Q lässt sich als Produkt von $(n-1)$ Spiegelungen darstellen.

Bemerkung. Statt $Ax = b$ können wir also

$$Rx = Q^t b$$

lösen, also im wesentlichen mit Rückwertseinsetzen.
Dies ist besser als mit der LR-Zerlegung $PA = LR$, da in

$$Ax = b \Leftrightarrow Rx = L^{-1}Pb$$

L^{-1} aufwendiger zu berechnen und anfälliger für Rundungfehler ist als Q^t.

Um die QR-Zerlegung zu berechnen, gibt es es zwei Methoden:

- mit Rotationen, eingeführt von Givens in die Numerik
- mit Spiegelungen, eingeführt von Householder.

Letztere ist geschickter. Wir besprechen nur diese.

Definition. Sei $v \in \mathbb{R}^n$ ein Vektor $\neq 0$. Die Abbildung

$$s_v : \mathbb{R}^n \to \mathbb{R}^n \text{ mit } s_v(x) = x - 2\frac{\langle x, v\rangle}{\langle v, v\rangle} v$$

heißt **(Householder-) Reflexion**. Sie lässt die Hyperebene $H = v^\perp$ fest und bildet v auf $-v$ ab. Die darstellende Matrix ist

$$Q_v = M_{\mathcal{E}}^{\mathcal{E}} = E - 2\frac{v \cdot v^t}{\langle v, v\rangle}$$

Es gilt:

a) Q_v ist symmetrisch,
b) $Q_v^2 = E$, d.h., Q_v ist eine **Involution**,
c) Q_v ist orthogonal.

1. Schritt der QR-Zerlegung mit Hilfe von Reflektionen

Es seien a_j die Spalten von A, also

$$A = (a_1 \dots a_n) \in \mathbb{R}^{n \times n.}$$

Wir suchen Q_v, so dass

$$Q_v A = \left(\begin{array}{c|ccc} \alpha_1 & * & & \\ \hline 0 & a_2' & \cdots & a_n' \\ & & & \end{array}\right)$$

gilt, d.h., der Vektor a_1 soll auf ein Vielfaches des Einheitsvektors e_1 abgebildet werden. Da $Q_v \in \mathrm{O}(n)$, gilt $\alpha_1 = \pm\|a_1\|$. Ferner

$$\alpha_1 e_1 \stackrel{!}{=} a_1 - 2\frac{\langle a_1, v\rangle}{\langle v, v\rangle} v \Rightarrow v \in \mathrm{Spann}(e_1, a_1).$$

Behauptung. $v = a_1 - \alpha_1 e_1$ hat die gewünschte Eigenschaft.

In der Tat:

$$\begin{aligned}\langle v, v\rangle &= \langle a_1 - \alpha_1 e_1, a_1 - \alpha_1 e_1\rangle \\ &= \|a_1\|^2 - 2\alpha_1\langle a_1, e_1\rangle + \alpha_1^2 = 2(\alpha_1^2 - \alpha_1 a_{11})\end{aligned}$$

$$\begin{aligned}Q_v(a_1) &= a_1 - 2\frac{\langle a_1, v\rangle}{\langle v, v\rangle} v = a_1 - \frac{\langle a_1, a_1 - \alpha_1 e_1\rangle}{\alpha_1^2 - \alpha_1 a_{11}}(a_1 - \alpha_1 e_1) \\ &= a_1 - \frac{\|a_1\|^2 - \alpha_1 a_{11}}{\alpha_1^2 - \alpha_1 a_{11}}(a_1 - \alpha_1 e_1) = \alpha_1 e_1\end{aligned}$$

wie gewünscht.

Bemerkung. Es ist am günstigsten $\alpha_1 = -\operatorname{sgn}(a_{11}) \cdot \|a_1\|$ zu wählen, um bei

$$v = a_1 - \alpha_1 e_1$$

Auslöschung zu vermeiden.

Algorithmus. (Algorithmus zur QR-Zerlegung)
Input. $A = (a_1 \dots a_n) \in \mathbb{R}^{n \times n}$
Output. $Q \in \mathrm{O}(n), R \in \mathbb{R}^{n \times n}$ obere Dreiecksmatrix, so dass $A = QR$ gilt. Genauer:
Vektoren $v_1, \dots, v_{n-1} \in \mathbb{R}^n$ mit $v_i \in (\mathrm{Spann}(e_1, \dots, e_{i-1}))^{\perp}$, so dass $Q = Q_{v_{n-1}} \cdot \ldots \cdot Q_{v_1}$ gilt.

1. Berechne $\alpha_1 = -\operatorname{sgn}(a_{11}) \cdot \|a_1\|$ und $v_1 = a_1 - \alpha_1 e_1$ und
$$Q_{v_1} A = \left(\begin{array}{c|c} \alpha_1 & r_1 \\ \hline 0 & A' \end{array}\right)$$
2. Rufe den Algorithmus rekursiv mit A' auf mit Ausgabe $v_2, \dots, v_{n-1}$ und
$$R' = \left(\begin{array}{c|ccc} \alpha_2 & r_2 & & \\ \hline & \alpha_3 & r_3 & \\ 0 & & \ddots & \\ & & & \alpha_{n-1} \end{array}\right)$$
3. **return** $v_1, \dots, v_{n-1}$ und $R = \left(\begin{array}{c|c} \alpha_1 & r_1 \\ \hline 0 & R' \end{array}\right)$.

Ende Teil 3

Bemerkung. Wenn die Ausgabe in der Form

$$\begin{pmatrix} v_1 & r_1 & & & & \\ & v_2 & r_2 & & & \\ & & v_3 & r_3 & & \\ & & & & & \\ & & & & v_{n-1} & r_{n-1} \\ & & & & & \alpha_n \end{pmatrix} \text{ und } \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \vdots \\ \alpha_{n-1} \end{pmatrix}$$

abspeichert, braucht der Algorithmus und die Ausgabe wenig mehr Speicherplatz als die Daten für A.

Der Aufwand für die QR-Zerlegung beträgt im ersten Schritt

1. n Multiplikationen, n Additionen und eine Quadratwurzel, um v_1 zu berechnen
2. n^2 Multiplikationen und $n(n-1)$ Additionen, um das Produkt $v_1^t A$ zu berechnen
3. Weitere $n^2 + 2n$ Multiplikationen, $n^2 + n - 1$ Additionen und eine Division für

$$Q_{v_1} A = A - \frac{2}{\langle v_1, v_1 \rangle} v_1 \cdot (v_1^t A)$$

Der Gesamtaufwand besteht also aus

$$\frac{4}{3}n^3 + O(n^2)$$

Additionen und Multiplikationen. Dieser Aufwand ist in etwa viermal so groß wie der Aufwand

$$\frac{1}{3}n^3 + O(n^2)$$

zur Berechnung der LR-Zerlegung, d.h., des Gauß-Algorithmus.

Bemerkung. Die QR-Zerlegung einer invertierbaren Matrix A ist bis auf ein Produkt mit einer Diagonalmatrix mit Einträgen ± 1 eindeutig.

$$QR = \tilde{Q}\tilde{R} \Leftrightarrow \tilde{Q}^t QR = \tilde{R},$$
$$\Rightarrow \tilde{Q}^t Q = \tilde{R}R^{-1},$$

da mit A auch R invertierbar ist. Also $\tilde{Q}^t Q = \tilde{R}R^{-1}$ ist eine orthogonale obere Dreiecksmatrix und deshalb eine Diagonalmatrix mit ± 1 auf der Diagonalen. Um diese Wahl bei der Zerlegung zu eliminieren, kann man verlangen, dass die Diagonaleinträge r_{jj} von R alle positiv sind.

Beispiel.

$$\begin{pmatrix} -1 & -81 & 78 & 90 & -54 & -99 \\ 31 & -54 & -15 & -31 & 21 & -27 \\ 75 & 15 & -88 & -97 & -86 & -94 \\ -85 & 0 & 28 & -19 & 100 & -58 \\ 13 & -61 & -5 & -34 & -90 & 10 \\ 72 & -7 & -41 & 100 & 24 & -67 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{pmatrix} = \begin{pmatrix} -34 \\ -67 \\ -88 \\ 71 \\ 80 \\ 65 \end{pmatrix}$$

$$Q = \begin{pmatrix} -.00722347 & -.702908 & .496638 & .0868635 & -.0840273 & .494584 \\ .223927 & -.443326 & -.355291 & .0667509 & .774655 & -.150136 \\ .54176 & .189891 & -.411287 & .332762 & -.147544 & .607274 \\ -.613995 & -.0678539 & -.491253 & -.399815 & .0378272 & .464537 \\ .093905 & -.518372 & -.456502 & .00458815 & -.607402 & -.380946 \\ .520089 & -.00319991 & .090767 & -.846992 & -.0266801 & .0561282 \end{pmatrix}$$

$$R = \begin{pmatrix} 138.438 & -12.7494 & -90.5823 & .339503 & -98.8676 & -54.5516 \\ 0 & 115.367 & -64.0642 & -49.3442 & 52.1078 & 62.6741 \\ 0 & 0 & 65.0664 & 129.538 & -4.77057 & 16.9329 \\ 0 & 0 & 0 & -103.788 & -92.6286 & 38.3023 \\ 0 & 0 & 0 & 0 & 91.3026 & -5.20826 \\ 0 & 0 & 0 & 0 & 0 & -136.507 \end{pmatrix}$$

$$x_{approx} = \begin{pmatrix} -4.17133 \\ -1.60137 \\ -4.68994 \\ 2.09824 \\ -.894167 \\ .395892 \end{pmatrix}, \quad x_{exakt} = \begin{pmatrix} -\frac{560727497570}{134424093523} \\ -\frac{12662523689}{7907299619} \\ -\frac{630440922733}{134424093523} \\ \frac{282053345297}{134424093523} \\ -\frac{120197591016}{134424093523} \\ \frac{53217381975}{134424093523} \end{pmatrix}$$

$$\det A = -1344240935230 = (-1) \cdot 2 \cdot 5 \cdot 17 \cdot 2137 \cdot 3700187$$

$$|\det A| = 7907299619 \cdot 17 \approx 1.34424 \cdot 10^{12}$$

$$x_{error} = \begin{pmatrix} 0 \\ 1.11022 \cdot 10^{-15} \\ 1.77636 \cdot 10^{-15} \\ 4.44089 \cdot 10^{-16} \\ -5.55112 \cdot 10^{-16} \\ -3.88578 \cdot 10^{-16} \end{pmatrix}$$

bei einer Maschinengenauigkeit auf 53 Binärstellen $2^{-53} \approx 1.11022 \cdot 10^{-16}$.

Ende Teil 4