Vorlesung 4

Alexander Mattick Kennung: qi69dube

Kapitel 1

16. Mai 2020

Ziel LGS $A\overline{x} = \overline{b}$

$A \in \mathbb{R}^{n \times n}$ und $x, b \in \mathbb{R}^n$

Inverse: $x = A^{-1}b$ ist aber langsam und numerisch hoch instabil

Gauss Elim: standard für per-Hand lösen, geht auch am Computer, essentiell LR Zerlegung.

LR Zerlegung:

$$A = \begin{bmatrix} 1 & 4 & 2 \\ 3 & -2 & 2 \\ 2 & 1 & -1 \end{bmatrix} \overset{II-3I, II-2I}{\to} \begin{bmatrix} 1 & 4 & 2 \\ 0 & -14 & -4 \\ 0 & -7 & -5 \end{bmatrix} \to \begin{bmatrix} 1 & 4 & 2 \\ 0 & -14 & -4 \\ 0 & 0 & -3 \end{bmatrix} = R$$

Dazu $L_1 = \begin{bmatrix} 1 & 0 & 0 \\ -3 & 1 & 0 \\ -2 & 0 & 1 \end{bmatrix}$ und $L_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -2 & 1 \end{bmatrix}$ Insgesamt wissen wir $R = L_2 L_1 A \to (L_2 L_1)^{-1} R = A \to L_1^{-1} L_2^{-1} R = A$

Die absoluten Werte von L ändern sich beim invertiern nicht, aber die Werte unter der diagonale werden mal (-1) genommen.

$L_1^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & 0 & 1 \end{bmatrix}$

und

$L_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 2 & 1 \end{bmatrix}$ Produkt aus beiden ist:

$L = \begin{bmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & 2 & 1 \end{bmatrix}$

Daraus folgt $A = LR$

Zum lösen:

$Ax = b \iff LRx = b$

durch substitution $Rx = y$ entsteht $Ly = b$.

$L \cdot y = b$

Forwärtseinsetzen:

$$\begin{bmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & 2 & 1 \end{bmatrix} y = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} \implies y_1 = b_1, y_2 = b_2 - 3y_1, \ldots$$

und dann normal $Rx = y$ durch Rückwärtseinsetzen

$$\begin{bmatrix} 1 & 4 & 2 \\ 0 & -14 & -4 \\ 0 & 0 & -3 \end{bmatrix} x = y \text{ und jetzt ganz normal.}$$

1 Präsenzaufgabe 1

a) tridiagonale Matrix (kommt oft in z.B. diskretisierung von DGL vor)

$$B = \begin{pmatrix} 1 & 2 & 0 & 0 \\ -3 & -8 & 3 & 0 \\ 0 & -8 & 13 & 3 \\ 0 & 0 & -2 & -4 \end{pmatrix}$$

MAN ZIEHT IMMER ZEILEN AB. Der dafür benötigte Faktor kann dann direkt in die L matrix geschrieben werden, kein (-1)* notwendig.

$[0 \quad 0 \quad -2 \quad -4]$

$$B = \begin{pmatrix} 1 & 2 & 0 & 0 \\ -3 & -8 & 3 & 0 \\ 0 & -8 & 13 & 3 \\ 0 & 0 & -2 & -4 \end{pmatrix} \xrightarrow{II - (-3) \cdot I} \begin{pmatrix} 1 & 2 & 0 & 0 \\ 0 & -2 & 3 & 0 \\ 0 & -8 & 13 & 3 \\ 0 & 0 & -2 & -4 \end{pmatrix}$$

$$L = \begin{pmatrix} 1 & 0 & 0 & 0 \\ -3 & 1 & 0 & 0 \\ 0 & 4 & 1 & 0 \\ 0 & 0 & -2 & 1 \end{pmatrix} \qquad \xrightarrow{III - \underline{4} \cdot II} \begin{pmatrix} 1 & 2 & 0 & 0 \\ 0 & -2 & 3 & 0 \\ 0 & 0 & 1 & 3 \\ 0 & 0 & -2 & -4 \end{pmatrix}$$

$$\xrightarrow{IV - \underline{(-2)} \cdot III} \begin{pmatrix} 1 & 2 & 0 & 0 \\ 0 & -2 & 3 & 0 \\ 0 & 0 & 1 & 3 \\ 0 & 0 & 0 & 2 \end{pmatrix}$$

Struktur einer Tridiagonalen Matrixlösung:

$$\begin{bmatrix} b_1 & c_1 & 0 & 0 & \cdots & 0 & 0 & 0 \\ a_1 & b_2 & c_2 & 0 & \cdots & 0 & 0 & 0 \\ 0 & a_2 & b_3 & c_3 & \cdots & 0 & 0 & 0 \\ \vdots & & \ddots & & \ddots & & \ddots & \\ 0 & 0 & 0 & 0 & \cdots & a_{n-2} & b_{n-1} & c_{n-1} \\ 0 & 0 & 0 & 0 & \cdots & 0 & a_{n-1} & b_n \end{bmatrix}$$

$$L = \begin{bmatrix} 1 & 0 & 0 & 0 & \dots \\ a_1 & 1 & 0 & 0 & \dots \\ 0 & a_2 & 1 & \dots & \dots \\ \dots & \ddots & \ddots & \ddots & \dots \end{bmatrix}$$

und $L = \begin{bmatrix} b_1 & c_1 & 0 & 0 & \dots \\ 0 & r_2 & c_2 & 0 & \dots \\ 0 & 0 & r_3 & c_3 & \dots \\ \dots & \ddots & \ddots & \ddots & \dots \end{bmatrix}$ Wobei die c's einfach aus A übernommen werden können! (über ihnen sind ja immer nur nullen,also verändern sie sich nicht).

Algorithmisch:

Erste Zeile $r_1 = b_1$

Zweite Zeile $a_1 - l_1 \cdot r_1 = 0 \implies l_1 = \frac{a_1}{r_1}$ und $r_2 = b_2 - l_1 \cdot c_1$

allgemeine Zeile:

$r_1 = b_1$

Für $k = 1, \dots, n-1$

$l_k = a_k - \frac{a_k}{r_k}$

$r_{k+1} = b_{k+1} - l_k \cdot c_k$

Laufzeit: $O(n)$

Aufwand des Lösens von $L \cdot y = b$, $R \cdot x = y$ bei tridiagonalen Matritzen

Also:

$y_1 = b_1$

$y_2 = b_2 - y_1 \cdot l_1$

Allgemein Algorithmus

$y_1 = b_1$

for $k = 1, \dots, n-1$ **do**

$y_k = b_k - l_{k-1} y_{k-1}$

end for

$R \cdot x = y$

$x_n = \frac{y_n}{r_n}$

$x_{n-1} = \frac{y_{n-1} - c_{n-1} * x_n}{r_{n-1}}$ Algorithmus:

$x_n = \frac{y_n}{r_n}$

for $k = n-1, \dots, 1$ **do**

$x_k = \frac{y_{n-1} - c_{n-1} * x_n}{r_{n-1}}$

end for

QR Zerlegung: Q Orthogonalmatrix.

R obere, rechte Dreiecksmatrix

$QRx = b \iff Rx = Q^T b$

Q Givensrotation und Housholder Spiegelungen.

Ziel: Der i-te Eintrag der j-ten spalte von A zu null.

$\rightarrow$ Nur die j-te spalte $A_j = \begin{bmatrix} a_{1,j} \\ \dots \\ a_{j,j} \\ \dots \\ a_{i,j} \\ \dots \end{bmatrix}$ dafür einfach nur die digonale und und zu eliminierendes betrachten:

$v = (a_{jj}, a_{ij})^T$

Zum lösen mit LR Zerlegung und pivot-matrix p $A = PLR$ muss man beim tauschen auch immer darauf achten, dass man die schon existierenden L-Spalten mittauscht:

LR-Zerlegung mit Pivot

Require: $A \in \mathbb{R}^{n \times n}$
  $P \leftarrow E_n$
  $L \leftarrow \mathbf{0}^{n \times n}$
  for $i \in [0..n]$ **do**
    $pivot \leftarrow$ pivot_index(A,i)
    swap_row(L,i,pivot)
    swap_row(R,i,pivot)
    swap_row(P,i,pivot)
    factor $\leftarrow 1/A[i][i]$
    for $j \in [i..n]$ **do**
      $l \leftarrow A[i][j] * factor$
      $A[j] \leftarrow A[j] - A[i] * l$
      $L[i][j] \leftarrow l$
    end for
  end for
  return P^T, $E_n + L$, A
