

Die Gauss-Elimination

Satz 1 Sei A eine (n, n) -Matrix, deren Hauptabschnittsmatrizen A_j regulär sind. Dann gibt es eine eindeutige Zerlegung $A = LR$, wobei L eine linke Dreiecksmatrix mit $l_{jj} = 1$ für $j = 1, \dots, n$ und R eine rechte Dreiecksmatrix ist.

Satz 2 Sei A eine (n, n) Matrix. Dann existieren eine Permutationsmatrix P , eine linke untere Dreiecksmatrix L mit $l_{jj} = 1$, $j = 1, \dots, n$ und eine rechte obere Dreiecksmatrix R derart, dass

$$PA = LR \quad (1)$$

gilt. (1) heißt LR-Zerlegung von A und ist in $\frac{1}{3}n^3 + \mathcal{O}(n^2)$ Operationen berechenbar. Ist A regulär, so ist auch R regulär und die Gausselimination liefert die eindeutige Lösung des Gleichungssystems $Ax = b$. Lineare Gleichungssysteme $Ax = LRx = b$ werden dann durch Lösung der Systeme $Ly = b$ und $Rx = y$ berechnet, was $\frac{1}{2}(n^2 - n)$ bzw. $\frac{1}{2}(n^2 + n)$ Rechenoperationen benötigt. So kommen wir zur Lösung von $Ax = b$ auf insgesamt $\frac{1}{3}n^3 + \mathcal{O}(n^2)$ Operationen.

Hier behandeln wir nur den ersten Satz, wir berechnen die Gausselimination ohne Pivotsuche. Sei also vorausgesetzt, dass alle Hauptabschnittsmatrizen A_j der Ausgangsmatrix A regulär sind.

Sei allgemein die Matrix A und der Lösungsvektor b durch

$$A =: A^{(1)} = \begin{pmatrix} a_{11}^{(1)} & \dots & a_{1n}^{(1)} \\ \vdots & & \vdots \\ a_{n1}^{(1)} & \dots & a_{nn}^{(1)} \end{pmatrix}, \quad b =: b^{(1)} = \begin{pmatrix} b_1^{(1)} \\ \vdots \\ b_n^{(1)} \end{pmatrix}$$

gegeben.

Beispiel:

$$A = \begin{pmatrix} 3 & 4 & 5 \\ -2 & 4 & 6 \\ 1 & 2 & 3 \end{pmatrix} = A^{(1)}, \quad b = \begin{pmatrix} -22 \\ -44 \\ -16 \end{pmatrix} = b^{(1)}$$

Allgemein gilt im j . Schritt:
Nach dem Algorithmus von Gauss wird $\frac{a_{ij}}{a_{jj}}$ -fache der Gleichung j von Gleichung i subtrahiert.

1. Schritt:

Zunächst am Beispiel:

Subtrahiere das $-\frac{2}{3}$ -fache von Zeile I von Zeile II:

$$|A^{(1)}|b^{(1)} = \left| \begin{array}{ccc|c} 3 & 4 & 5 & -22 \\ -2 & 4 & 6 & -44 \\ 1 & 2 & 3 & -16 \end{array} \right| \begin{array}{l} | \cdot -\frac{2}{3}I \leftarrow + \\ | \cdot -\frac{2}{3}I \leftarrow + \\ \hline \end{array}$$

Daraus ergibt sich

$$|A^{(2)}|b^{(2)} = \left| \begin{array}{ccc|c} 3 & 4 & 5 & -22 \\ 0 & \frac{20}{3} & \frac{28}{3} & -\frac{176}{3} \\ 0 & \frac{2}{3} & \frac{4}{3} & -\frac{26}{3} \end{array} \right|$$

Nun allgemein:

Berechne für alle Zeilen $i = 2, \dots, n$

$$l_{i1} = \frac{a_{i1}^{(1)}}{a_{11}^{(1)}}. \quad (2)$$

Berechne weiterhin für alle Spalten $k = 2, \dots, n$ in der Zeile i die Elemente

$$a_{ik}^{(2)} = a_{ik}^{(1)} - l_{i1}a_{1k}^{(1)} \quad (3)$$

und weiterhin

$$b_i^{(2)} = b_i^{(1)} - l_{i1}b_1^{(1)}. \quad (4)$$

Es folgt

$$|A^{(2)}|b^{(2)} = \left| \begin{array}{cccc|c} a_{11}^{(1)} & a_{12}^{(1)} & \dots & a_{1n}^{(1)} & b_1^{(1)} \\ 0 & a_{22}^{(2)} & \dots & a_{2n}^{(2)} & b_2^{(2)} \\ \vdots & \vdots & & \vdots & \vdots \\ 0 & a_{n2}^{(2)} & \dots & a_{nn}^{(2)} & b_n^{(2)} \end{array} \right| \begin{array}{l} | \cdot -l_{i2}II \leftarrow + \\ \hline \end{array}$$

Wir überprüfen diese Formeln an unserem Beispiel: Berechne (2) für $j = 1, i = j + 1, \dots, n$, also $i = 2, 3$:

$$l_{21} = \frac{a_{21}^{(1)}}{a_{11}^{(1)}} = \frac{-2}{3}$$

$$l_{31} = \frac{a_{31}^{(1)}}{a_{11}^{(1)}} = \frac{1}{3}$$

Damit berechnen wir $a_{ik}^{(2)}$ nach (3) für $i = 2, 3, k = 2, 3$:

$$\begin{aligned} a_{22}^{(2)} &= a_{22}^{(1)} - l_{21}a_{12}^{(1)} = 4 - \left(\frac{-2}{3}\right) \cdot 4 = \frac{20}{3} \\ a_{23}^{(2)} &= a_{23}^{(1)} - l_{21}a_{13}^{(1)} = 6 - \left(\frac{-2}{3}\right) \cdot 5 = \frac{28}{3} \\ a_{32}^{(2)} &= a_{32}^{(1)} - l_{31}a_{12}^{(1)} = 2 - \left(\frac{1}{3}\right) \cdot 4 = \frac{2}{3} \\ a_{33}^{(2)} &= a_{33}^{(1)} - l_{31}a_{13}^{(1)} = 3 - \left(\frac{1}{3}\right) \cdot 5 = \frac{4}{3} \end{aligned}$$

und $b_i^{(2)}$ für $i = 2, 3$ nach (4):

$$\begin{aligned} b_2^{(2)} &= b_2^{(1)} - l_{21}b_1^{(1)} = -44 - \left(\frac{-2}{3}\right) \cdot (-22) = -\frac{176}{3} \\ b_3^{(2)} &= b_3^{(1)} - l_{31}b_1^{(1)} = -16 - \left(\frac{1}{3}\right) \cdot (-22) = -\frac{26}{3} \end{aligned}$$

Seien nun die ersten $j - 1$ Schritte berechnet, dann haben die Matrix $A^{(j)}$ und der Vektor $b^{(j)}$ die Gestalt

$$|A^{(j)}|b^{(j)} = \left| \begin{array}{cccc|c} a_{11}^{(1)} & a_{12}^{(1)} & \dots & a_{1n}^{(1)} & b_1^{(1)} \\ & \ddots & & \vdots & \vdots \\ & & a_{jj}^{(j)} & \dots & a_{jn}^{(j)} & b_j^{(j)} \\ & & \vdots & \vdots & \vdots & \vdots \\ 0 & & a_{nj}^{(j)} & \dots & a_{nn}^{(j)} & b_n^{(j)} \end{array} \right| \begin{array}{l} \leftarrow + \\ \vdots \\ \vdots \\ \vdots \\ \vdots \end{array}$$

Nun muss $A^{(j+1)}|b^{(j+1)}$ berechnet werden. Dazu werden für alle Zeilen i mit $i = j + 1, \dots, n$ die Werte

$$l_{ij} = \frac{a_{ij}^{(j)}}{a_{jj}^{(j)}} \quad (5)$$

der Matrix L_{j+1} berechnet und dann für diese Zeilen und alle Spalten $k = j + 1, \dots, n$ die neuen Matrixelemente für $A^{(j+1)}$:

$$a_{ik}^{(j+1)} = a_{ik}^{(j)} - l_{ij}a_{jk}^{(j)}. \quad (6)$$

Weiterhin muss auch der Vektor b in allen Zeilen $i = j + 1, \dots, n$ analog zu den Berechnungen der Matrix A modifiziert werden:

$$b_i^{(j+1)} = b_i^{(j)} - l_{ij}b_j^{(j)} \quad (7)$$

Damit erhält man nach $n - 1$ Schritten die Matrix

$$|A^{(n-1)}|b^{(n-1)} = \begin{vmatrix} a_{11}^{(1)} & \dots & a_{1n}^{(1)} \\ & \ddots & \vdots \\ 0 & & a_{nn}^{(n-1)} \end{vmatrix} \begin{matrix} b_1^{(1)} \\ \vdots \\ b_n^{(n-1)} \end{matrix}$$

Die Matrix L wird durch die berechneten l_{ij} gebildet! Es werden keine Matrixprodukte LA berechnet! Alle Berechnungen, die im Programm durchgeführt werden müssen, sind mit den Gleichungen (5), (6) und (7) gegeben!

Berechnungen am Beispiel: Es wird der $j = 2$. Schritt durchgeführt, nach (5) gilt für $i = j + 1, \dots, n$, also hier $i = 3$:

$$l_{32} = \frac{a_{32}^{(2)}}{a_{22}^{(2)}} = \frac{\frac{2}{3}}{\frac{20}{3}} = \frac{1}{10}$$

Damit folgt für $k = j + 1, \dots, n$ und für $i = j + 1, \dots, n$, also hier $k = 3$ und $i = 3$

$$\begin{aligned} a_{33}^{(3)} &= a_{33}^{(2)} - l_{32}a_{23}^{(2)} = \frac{4}{3} - \frac{1}{10} \cdot \frac{28}{3} = \frac{2}{5}, \\ b_3^{(3)} &= b_3^{(2)} - l_{32}b_2^{(2)} = -\frac{26}{3} - \frac{1}{10} \cdot \frac{-176}{3} = \frac{-14}{5}. \end{aligned}$$

Das Gleichungssystem wird dann durch das so genannte Rückwärtseinsetzen gelöst: Für $j = n, n - 1, \dots, 1$ berechnet man die Komponenten x_j des Lösungsvektors x durch

$$x_j = \frac{1}{a_{jj}^{(n)}} \left(b_j^{(n)} - \sum_{k=j+1}^n a_{jk}^{(n)} x_k \right).$$

Also am Beispiel:

$$\begin{aligned} x_3 &= \frac{1}{\frac{2}{5}} \cdot \frac{14}{5} = -7 \\ x_2 &= \frac{1}{\frac{20}{3}} \left(-\frac{176}{3} - \frac{28}{3} \cdot (-7) \right) = 1 \\ x_1 &= \frac{1}{3} \left(-22 - a_{12}^{(3)} x_2 - a_{13}^{(3)} x_3 \right) = \frac{1}{3} (-22 - 4 \cdot 1 - 5 \cdot (-7)) = 3 \end{aligned}$$

Programmerläuterung:

Hier werden nur die relevanten Zeilen des Codes erläutert. Die Eingabe der Matrizen etc. sei vorausgesetzt!

Wir haben $n - 1$ Schritte, welche wir mit j bezeichnen. Im j . Schritt werden für $i = j + 1, \dots, n$ die Werte

$$l_{ij} = \frac{a_{ij}^{(j)}}{a_{jj}^{(j)}}$$

berechnet. Dies wird im j . Schritt nur für $i = j + 1, \dots, n$ gemacht, da die ersten j Zeilen ja unverändert bleiben! Dann müssen die neuen Zeilen und Spalten der Matrix A berechnet werden. Die ersten j Zeilen bleiben wie gesagt unverändert. In den ersten j Spalten werden alle Elemente unterhalb der Zeile j zu 0 berechnet. Dies machen wir aber nicht im Programm. Im Programm werden nur die Werte neu berechnet, die noch für weitere Berechnungen notwendig sind - für das aktuelle i , also in der i . Zeile sind dies die Elemente

$$a_{ik}^{(2)} = a_{ik}^{(1)} - l_{i1}a_{1k}^{(1)}$$

für $k = j + 1, \dots, n$, also die Spalten $j + 1, \dots, n$. Alle weiteren Elemente a_{ij} von A , die sich theoretisch zu 0 berechnen würden, werden mit den Werten l_{ij} überschrieben. So kann Speicherplatz gespart werden. Da wir die Matrix in jedem Schritt aktualisieren, also die Ausgangsmatrix immer wieder mit den neuen Werten überschreiben, müssen wir den Index (j) nicht weiter beachten. Weiterhin muss noch der Vektor b im j . Schritt aktualisiert werden. Die ersten j Zeilen bleiben unverändert, für $i = j + 1, \dots, n$ gilt:

$$b_i^{(j+1)} = b_i^{(j)} - l_{ij}b_j^{(j)}.$$

Bei der Formel für das Rückwärtseinsetzen,

$$x_j = \frac{1}{a_{jj}^{(n)}} \left(b_j^{(n)} - \sum_{k=j+1}^n a_{jk}^{(n)} x_k \right), \quad j = n, \dots, 1$$

kann man erkennen, dass wirklich nur auf die Elemente a_{ij} von A zugegriffen wird, für die $j \geq i$ gilt!

Es ergibt sich also der folgenden Matlab-Code:

```
% ueberschreibe die Matrix A in jedem Schritt mit den neuen
% Werten, berechne nur die Werte, die wirklich ~= 0 sind.
% Der Algorithmus setzt keine Werte von A auf Null! Da
% hinterher beim Rueckwaertseinsetzen nur noch auf die
```

```

% Elemente in der rechten oberen Dreiecksmatrix zugegriffen
% wird, ist dies nicht notwendig!!!
% In die linke untere Ecke von A wird in Zeile (*) die
% Matrix L geschrieben

for j=1:N-1
    % Schritte
    for i=j+1:N
        % Veraenderung der Zeilen
        A(i,j)= A(i,j)/A(j,j);
        % (*), Berechnung l_ij
        for k=j+1:N
            % Veraenderung der Spalten
            A(i,k)=A(i,k)-A(i,j)*A(j,k);
        end
        b(i) = b(i) - A(i,j)*b(j); % Veraenderung b
    end
end

% Rueckwaertseinsetzen
for j=N:-1:1
    summe = 0;
    % Initialisierung der Summe
    for k=j+1:N
        % Berechnung der Summe
        summe = summe + A(j,k) * x(k);
    end
    x(j) = (b(j)- summe) / A(j,j);
end

```

Erläuterung zum Rückwärtseinsetzen:

Eine Summe $\sum_{k=1}^N v_k$ kann in Matlab auf zwei Varianten berechnet werden. Zum einen verfügt Matlab über den Befehl `sum`. Ist $v = (v_1, v_2, \dots, v_n)^t$, so liefert `sum(v)` die gewünschte Summe. Dies können wir ebenso mit einer `for`-Schleife realisieren. Dazu wird ein Skalar `s` zu Beginn gleich 0 gesetzt und dann werden die einzelnen Komponenten von v hinzu addiert:

```

s=0;
for k=1:N
    s = s + v(k);
end

```

liefert das gleiche Ergebnis wie `sum(v)`.

Achtung! `for`-Schleifen sollten eigentlich vermieden werden! Sie sind sehr zeitaufwendig! Da wir aber hier nicht wirklich über alle Spalten summieren, sondern für jedes j einen anderen Beginn der Summe haben (und die anderen Elemente von A nicht auf 0 gesetzt wurden), können wir hier die `for`-Schleife nicht umgehen.