Florian Grabner florian.grabner@gmx.at

Gauß'scher Algorithmus

- Mathematische / Fachliche Inhalte in Stichworten:

  Lösen eines linearen Gleichungssystems mit Hilfe der Matrizenrechnung

- Kurzzusammenfassung

  Praktische Anwendung des Gauß'schen Algorithmus. Es wird das Funktionsprinzip Schritt für Schritt genau erläutert und zum Schluß eine autonome "Gauß-Funktion" erstellt, die zeigt, dass sich dieses Verfahren gut fürs rechnergestützte lösen von großen Gleichungssystemen eignet (FEM).

- Lehrplanbezug (bzw. Gegenstand / Abteilung / Jahrgang):

  Mathematik, 4.Jahrgang, alle Abteilungen

- Mathcad-Version:

  Mathcad 2000

Das zu lösende lineare Gleichungssystem

$$2x + 1y - 2z = 5$$
$$-3x + 7y + 5z = 9$$
$$x - 2y + 3z = 13$$

Dieses lineare Gleichungssystem besteht aus 3 Gleichungen mit 3 Unbekannten und ist somit lösbar (Determinante der Koeffizientenmatrix nicht Null).Der erste Schritt ist, diese 3 Gleichungen in eine Matrix umzuwandeln. Wir müssen alle x-, y- und z-Werte untereinander stellen und erhalten dann ein Matrix der folgenden Form.

$$M := \begin{pmatrix} 2 & 1 & -2 & 5 \\ -3 & 7 & 5 & 9 \\ 1 & -2 & 3 & 13 \end{pmatrix}$$

Wir müssen nun versuchen die linke Seite der Hauptdiagonalen auf Null zu bringen. Dies geschieht auf folgende Art und Weise.

$$MA_{r,c} = M_{zw_{r,c}} - K_{Gauß}$$

MA	... Arbeitsmatrix
M_{zw}	... Zwischengespeicherte Versionen von MA
r	... Aktuelle Zeilenposition von row (engl. Reihe)
c	... Aktuelle Spaltenposition von column (engl. Spalte)
$K_{Gauß}$	**... Gauß'sche Konstante (*)**

(*) Diese Bezeichnung wurde vom Autor gewählt um dem Verfahren mehr Durchsichtigkeit zu verleihen (beliebig gewählter Name)!

Diese Gauß'sche Konstante (siehe Anmerkung oben) ist das eigentliche Herzstück des Algorithmus. Sie ist genaugenommen keine konstante Zahl sondern vielmehr eine konstante Beziehung. Sie lautet:

$$K_G(M_{zw}, r, c, rr, cc) := \frac{M_{zw_{r,cc}}}{M_{zw_{rr,rr}}} \cdot M_{zw_{rr,c}}$$

r	... Aktuelle Zeilenposition
c	... Aktuelle Spaltenposition
rr,cc	... Position in der Hauptdiagonalen
M_{zw}	... Zwischengespeicherte Versionen von MA

Die beiden Variablen "rr" und "cc" beschreiben jenen Wert der Hauptdiagonale, der in der gleichen Spalte wie die zu bearbeitenden Zelle liegt. Zum Beispiel für Zelle $\mathbf{M_{1,0}}$ sind "rr" und "cc" gleich 0, da die Zelle in der nullten Spalten liegt, für $\mathbf{M_{2,1}}$ sind "rr" und "cc" gleich 1, usw.

ACHTUNG: MathCAD zählt die Zeilen und Spalten von 0 (da ORIGIN "defaultmäßig" auf 0 gesetzt)

Ab jetzt wird nur mehr mit der Arbeitsmatrix gearbeitet, um die Ausgangsmatrix als **M** gespeichert zu erhalen. Die Arbeitsmatrix ist zu Beginn gleich der zu lösenden Matrix **M**, d.h.

$$MA := M$$

Die linke Seite der Hauptgeraden besteht in unserem Fall aus 3 Zellen ($\mathbf{MA_{1,0}}$; $\mathbf{MA_{2,0}}$; $\mathbf{MA_{2,1}}$). Um die erste Zelle auf den Wert Null zu bringen, muss K_G von Zeile 1 & 2 abgezogen werden.

$$rr := 0 \qquad cc := 0$$

$$r := 1 .. 2 \qquad c := 0 .. 3$$

$$MA_{r,c} := M_{r,c} - K_G(M, r, c, rr, cc) \qquad MA = \begin{pmatrix} 2 & 1 & -2 & 5 \\ 0 & 8.5 & 2 & 16.5 \\ 0 & -2.5 & 4 & 10.5 \end{pmatrix}$$

Wir speichern dieses Zwischenergebniss ab als: $M_{zw1} := MA$

Wir sehen also, das K_G sozusagen spaltenweise vorgeht; aus diesem Grund muss K_G immer von der betreffenden und allen darunterliegenden Zeilen abgezogen werden.
Für Zelle $\mathbf{MA_{2,1}}$ muss nun die gleiche Prozedur durchgeführt werden.

$$rr := 1 \qquad cc := 1$$

$$r := 2$$

$$MA_{r,c} := M_{zw1_{r,c}} - K_G\left(M_{zw1}, r, c, rr, cc\right) \qquad MA = \begin{pmatrix} 2 & 1 & -2 & 5 \\ 0 & 8.5 & 2 & 16.5 \\ 0 & 0 & 4.588 & 15.353 \end{pmatrix}$$

Wir speichern dieses Zwischenergebniss ab als: $M_{zw2} := MA$

Die gesamte linke Seite der Hauptdiagonalen ist nun Null. Als nächster Schritt muss die Hauptdiagonale mit Einsern besetzt werden, d.h. jede Zeile wird mit dem entsprechenden Wert der Haupdiagonalen durchdividiert.

$$MA_{0,c} := \frac{MA_{0,c}}{M_{zw2_{0,0}}} \qquad MA_{1,c} := \frac{MA_{1,c}}{M_{zw2_{1,1}}} \qquad MA_{2,c} := \frac{MA_{2,c}}{M_{zw2_{2,2}}}$$

$$MA = \begin{pmatrix} 1 & 0.5 & -1 & 2.5 \\ 0 & 1 & 0.235 & 1.941 \\ 0 & 0 & 1 & 3.346 \end{pmatrix}$$

Wir speichern auch dieses Zwischenergebniss ab als: $M_{zw3} := MA$

Nun muss die rechte Seite der Hauptdiagonalen mit Nullen besetzt werden. Sie besteht in unserem Fall wieder aus 3 Zellen ($\mathbf{MA_{1,2}}$; $\mathbf{MA_{0,1}}$; $\mathbf{MA_{0,2}}$). Um die erste Zelle auf den Wert Null zu bringen, muss K_G von Zeile 1 & 0 abgezogen werden.

Die vorangegangene Prozedur kommt jetzt gedanklich "180°" verdreht zu Anwendung, d.h. K_G muss nun von der betreffenden und allen darüberliegenden Zeilen abgezogen werden.

Wir beginnen wieder mit der ersten Zelle ($\mathbf{MA_{1,2}}$). K_G muss von Zeile 1 und 0 abgezogen werden.

$rr := 2 \qquad cc := 2$

$r := 1 .. 0 \qquad c := 3 .. 0$

$MA_{r,c} := M_{zw3_{r,c}} - K_G(M_{zw3}, r, c, rr, cc) \qquad MA = \begin{pmatrix} 1 & 0.5 & 0 & 5.846 \\ 0 & 1 & 0 & 1.154 \\ 0 & 0 & 1 & 3.346 \end{pmatrix}$

Wir speichern dieses Zwischenergebniss ab als: $M_{zw4} := MA$

Es bleibt nun nur mehr Zelle $\mathbf{MA_{0,1}}$ übrig, d.h. K_G von Zelle 0 abziehen und wir erhalten das Endergebnis.

$rr := 1 \qquad cc := 1$

$r := 0$

$MA_{r,c} := M_{zw4_{r,c}} - K_G(M_{zw4}, r, c, rr, cc) \qquad MA = \begin{pmatrix} 1 & 0 & 0 & 5.269 \\ 0 & 1 & 0 & 1.154 \\ 0 & 0 & 1 & 3.346 \end{pmatrix}$

Die nun erhaltene Matrix würde folgendem Gleichungssystem entsprechen:

$1x + 0y + 0z = 5.269$

$0x + 1y + 0z = 1.154$

$0x + 0y + 1z = 3.346$

Wir sehen also, dass die letzte Spalte unseren gesuchten Unbekannten entspricht.
Wir wollen das soeben erhaltene Ergebnis mit der MathCAD-Funktion **Vorgabe - suchen** überprüfen.

$x := 1 \qquad y := 1 \qquad z := 1$

Given

$2x + 1y - 2z = 5$

$-3x + 7y + 5z = 9$

$x - 2y + 3z = 13$

$\begin{pmatrix} x \\ y \\ z \end{pmatrix} := Find(x, y, z)$

$x = 5.269 \qquad y = 1.154 \qquad z = 3.346$

Die beiden Ergebnisse stimmen überein. Mit dem soeben erlangten Wissen können wir nun eine autonom ablaufenden Funktion zum lösen linearer Gleichungssysteme erstellen.

Bei genauerer Betrachtung sehen wir, dass sich der Gesamte Lösungsablauf in drei Teile gliedert. Die Umsetzung dieser wollen wir uns nun etwas genauer ansehen.

1. Teil: Die linke Seite der Hauptdiagonalen

Mit Hilfe der Gauß'schen Konstante wird die linke Seite der Hauptdiagonalen mit Nullen besetzt, d.h. das Herzstück dieses Teiles ist die Gleichung

$$MA_{r,c} = M_{zw_{r,c}} - K_{Gauß}$$

Abhängig von der Anzahl der Gleichungen (Anzahl der Zeilen) muss diese Gleichung öfters angewendet werden um das gewünschte Ziel zu erreichen. Programmtechnisch eignet sich dafür eine FOR-Schleife.

Gehen wir wieder von einem Gleichungssystem mit 3 Gleichungen und 3 Unbekannten aus, müssen wir zwei Durchläufe machen oder "Anzahl der Zeilen - 1" Durchläufe.

$$\text{for} \quad i \in 1 .. \text{rows}(M) - 1$$

$$MA_{r,c} \leftarrow M_{r,c} - \frac{M_{r,cc}}{M_{rr,rr}} \cdot M_{rr,c}$$

Wir wissen jedoch von vorhin, dass bei einem Durchlauf die Gauß'sche Konstante von der betreffenden und allen darunterliegenden Zeilen abgezogen werden muss. Oben haben wir deshalb "r" und "c" als Laufvariablen definiert; programmtechnisch eignet sich wiederum die FOR-Schleife.

$$\text{for} \quad i \in 1 .. \text{rows}(M) - 1$$

$$\text{for} \quad r \in i .. \text{rows}(M) - 1$$

$$\text{for} \quad c \in i - 1 .. \text{cols}(M) - 1$$

$$MA_{r,c} \leftarrow M_{r,c} - \frac{M_{r,cc}}{M_{rr,rr}} \cdot M_{rr,c}$$

Anm: Der Anfang von "c" wird nicht mit 0 gesetzt um unnötige Rechenoperationen zu ersparen - schnellere Rechenzeiten.

Als nächstes müssen wir noch die Variablen "rr" bzw. "cc" und die Arbeitsmatrix MA definieren. Hier machen wir uns die Eigenschaft zu Nutze, dass beide ("rr" und "cc") immer gleich groß sind. Wir kreieren also eine neue Variable und nennen sie z.B.: "rc". "rc" beschreibt jenen Wert der Hauptdiagonalen, der in der gleichen Spalte wie die zu bearbeitenden Zelle liegt.

$$\text{for} \quad i \in 1 .. \text{rows}(M) - 1$$

$$rc \leftarrow i - 1$$

$$MA \leftarrow M$$

$$\text{for} \quad r \in i .. \text{rows}(M) - 1$$

$$\text{for} \quad c \in i - 1 .. \text{cols}(M) - 1$$

$$MA_{r,c} \leftarrow M_{r,c} - \frac{M_{r,rc}}{M_{rc,rc}} \cdot M_{rc,c}$$

Wir dürfen auch nicht vergessen, nach jedem Durchlauf die neu entstandene Matrix zu speichern (bildet die Grundlage für den nächsten Durchlauf). Programmtechnisch erfolgt das ganz einfach

$$\text{for} \quad i \in 1 .. \text{rows}(M) - 1$$

$$rc \leftarrow i - 1$$

$$MA \leftarrow M$$

$$\text{for} \quad r \in i .. \text{rows}(M) - 1$$

$$\text{for} \quad c \in i - 1 .. \text{cols}(M) - 1$$

$$MA_{r,c} \leftarrow M_{r,c} - \frac{M_{r,rc}}{M_{rc,rc}} \cdot M_{rc,c}$$

$$M \leftarrow MA$$

2.Teil: Die Hauptdiagonale

Im zweiten Teil besetzen wir die Hauptdiagonale mit Einsern inderm wir jede Zeile durch den entsprechenden Wert der Hauptdiagonalen dividieren und das Ergebnis speichern. Wieder bedienen wir uns zweier FOR-Schleifen.

for $r \in 0 .. \text{rows}(M) - 1$

  for $c \in r .. \text{cols}(M) - 1$

    $MA_{r,c} \leftarrow \frac{MA_{r,c}}{M_{r,r}}$

$M \leftarrow MA$

3. Teil: Die rechte Seite der Hauptdiagonalen

Der letzte Teil ist dem ersten Teil sehr ähnlich. Die Bereiche der FOR-Schleifen müssen den neuen Gegebenheiten angepasst werden und Der "rc" Wert ist nun nicht "i - 1" sondern " i + 1" da von unten nach oben gerechnet wird.

for $i \in \text{rows}(M) - 2 .. 0$

  $rc \leftarrow i + 1$

  for $r \in i .. 0$

    for $c \in i + 1 .. \text{cols}(M) - 1$

      $MA_{r,c} \leftarrow M_{r,c} - \frac{M_{r,rc}}{M_{rc,rc}} \cdot M_{rc,c}$

  $M \leftarrow MA$

Fügen wir nun alle drei Teile zusammen, so erhalten wir die gewünschte GAUß-Funktion

$\text{Gauß}(M) :=$

for $i \in 1 .. \text{rows}(M) - 1$

  $rc \leftarrow i - 1$

  $MA \leftarrow M$

  for $r \in i .. \text{rows}(M) - 1$

    for $c \in i - 1 .. \text{cols}(M) - 1$

      $MA_{r,c} \leftarrow M_{r,c} - \frac{M_{r,rc}}{M_{rc,rc}} \cdot M_{rc,c}$

  $M \leftarrow MA$

for $r \in 0 .. \text{rows}(M) - 1$

  for $c \in r .. \text{cols}(M) - 1$

    $MA_{r,c} \leftarrow \frac{MA_{r,c}}{M_{r,r}}$

$M \leftarrow MA$

for $i \in \text{rows}(M) - 2 .. 0$

  $rc \leftarrow i + 1$

  for $r \in i .. 0$

    for $c \in i + 1 .. \text{cols}(M) - 1$

      $MA_{r,c} \leftarrow M_{r,c} - \frac{M_{r,rc}}{M_{rc,rc}} \cdot M_{rc,c}$

  $M \leftarrow MA$

$$\text{Gauß}(M) = \begin{pmatrix} 1 & 0 & 0 & 5.269 \\ 0 & 1 & 0 & 1.154 \\ 0 & 0 & 1 & 3.346 \end{pmatrix}$$

Zusatz: Sonderfall "Division durch NULL"

Wie allgemein bekannt, darf es nie zu einer Division durch NULL kommen. Dieser Fall tritt allerdings ein wenn jeweils die ersten beiden Elemente der ersten beiden Zeilen Linearkombinationen sind oder das erste Element in der ersten Zeile eine NULL ist.
Im folgend Beispiel erläutern wir dieses Problem genauer.

Die neue Matrix **K** ist mit unserer bisherigen Gauß-Funktion nicht lösbar obwohl eine Lösung existiert.

$$K := \begin{pmatrix} 1 & 2 & 3 & 5 \\ 3 & 6 & 8 & 4 \\ 5 & -2 & 4 & 3 \end{pmatrix} \qquad \text{Gauß}(K) = \blacksquare$$

$$KD := \begin{pmatrix} 1 & 2 & 3 \\ 3 & 6 & 8 \\ 5 & -2 & 4 \end{pmatrix} \qquad b := \begin{pmatrix} 5 \\ 4 \\ 3 \end{pmatrix} \qquad KD^{-1} \cdot b = \begin{pmatrix} -11.5 \\ -8.25 \\ 11 \end{pmatrix}$$

Der Grund dafür liegt eben bei der speziellen Zahlenkombination der vier Positionen ($\mathbf{K_{00}}$, $\mathbf{K_{01}}$, $\mathbf{K_{10}}$, $\mathbf{K_{11}}$).
Bereits nach der zweiten Operation liegt das Problem offen.

1. Rechenoperation $\quad KD_{1,0} - K_G(K, 1, 0, 0, 0) = 0$

2. Rechenoperation $\quad KD_{1,1} - K_G(K, 1, 1, 0, 0) = 0$

Der Wert der zweiten Stelle der Hauptdiagonale wird NULL und dies führt bei dem zweiten Rechendurchgang unweigerlich zu einer NULL-Division.

$$K_G\left(M_{ZW}, r, c, rr, cc\right) := \frac{M_{ZW_{r,cc}}}{M_{ZW_{rr,rr}}} \cdot M_{ZW_{rr,c}}$$

Bei diesem Durchgang sind rr und cc gleich 1 (siehe Beispiel zu Beginn) und somit ist der Nenner gleich NULL.

Das Problem ist equallent wenn das erste Element der Ersten Zeile NULL ist oder die dritte Stelle der Hauptdiagonalen nach dem zweiten Rechendurchgang gleich NULL ist usw. (die Wahrscheinlichkeiten sind aber so gering und der damit verbundene Rechenaufwand um dies hundertprozentig auszuschließen so enorm, dass darauf verzichtet wird - es würde den Rahmen eindeutig sprengen und das eigentliche Ziel ins Abseits drängen).

Wir können dieses Problem umgehen, indem wir gezielt 2 Zeilen miteinander vertauschen. In unserem Fall müssen wir zum Beispiel die zweite Zeile mit der dritten Zeile vertauschen (es existiert nun keine Linearkombination mehr denn (5/1)*2 ist nicht -2 und somit steht an zweiter Stelle der Hauptdiagonalen nach dem ersten Rechendurchgang nicht NULL sondern -12).

$$K := \begin{pmatrix} 1 & 2 & 3 & 5 \\ 5 & -2 & 4 & 3 \\ 3 & 6 & 8 & 4 \end{pmatrix} \qquad \text{Gauß}(K) = \begin{pmatrix} 1 & 0 & 0 & -11.5 \\ 0 & 1 & 0 & -8.25 \\ 0 & 0 & 1 & 11 \end{pmatrix}$$

Mit einer kleinen gezielten Manipulation der Ausgangsmatrix kann das Problem der NULL-Division umgangen werden.

Für unser Programm bedeutet dies, dass wir abfragen müssen ob

1. Das Element $\mathbf{K_{0,0}}$ = NULL ist
2. Die Möglichkeit besteht, dass nach dem ersten Rechendurchgang das zweite Element der Hauptdiagonalen NULL wird

Damit die Übersichtlichkeit nicht verloren geht, schreiben wir diese beiden Abfragen in ein eigenes Programm. Die Vorgehensweise bei der ersten Abfrage ist die, dass wir eine Rechenoperation durchführen, die eine Fehlermeldung zurückgibt falls der erste Fall vorliegt. Wir lernen eine neue Funktion kennen - **"on error"**. Unsere Rechenoperation für den ersten Teil berechnet den zweiten Wert der Hauptdiagonalen nach dem ersten Rechendurchgang. Ist der Wert $\mathbf{K_{0,0}}$ gleich NULL (es würde eine NULL-Division auftreten - Fehlermeldung)so tauscht das Programm automatische die erste Zeile mit einer geeigneten Zeile aus.

```
for r ∈ 0 .. rows(K) − 1                    on error  K1,0 / K0,0 · K0,1
    continue if Kr,0 = 0
    ztmp1 ← (K^T)^⟨0⟩
    ztmp2 ← (K^T)^⟨r⟩
    for c ∈ 0 .. cols(K) − 1
        K0,c ← ztmp2c,0
        Kr,c ← ztmp1c,0
    break
```

Anm: Die Funktion **"continue"** überspringt die nachfolgenden Operationen falls die Bedingung der if-Schleife erfüllt ist.

Anwendung - Eine Zeile die mit NULL beginnt ist nicht geeignet.

"break" beendet die Anwendung vorzeitig falls eine geeignete Zeile gefunden wurde.

Für die zweite Überprüfung verwenden wir wieder eine einfache if-schleife. Diese überprüft, ob der zweite Wert der Hauptdiagonalen nach dem ersten Rechendurchgang gleich dem zweiten Wert der zweiten Zeile der Ausgangsmatrix ist. Stimmt diese Bedingung so werden wiederum die entsprechend Zeilen vertauscht.

```
for r ∈ 1 .. rows(K) − 1                    if  K1,0 / K0,0 · K0,1 = K1,1
    continue if Kr,1 = K1,0 / K0,0 · K0,1
    ztmp1 ← (K^T)^⟨1⟩
    ztmp2 ← (K^T)^⟨r⟩
    for c ∈ 0 .. cols(K) − 1
        K1,c ← ztmp2c,0
        Kr,c ← ztmp1c,0
    break
```

Anm: Anwendung von **"continue"** - Eine Zeile deren zweites Element den selben Wert hat ist nicht geeignet.

"break" beendet vorzeitig (siehe oben)

Wir fassen nun diese beiden Teile zu einem Gesamtprogramm zusammen. Dieses Programm braucht nur dann ausgeführt zu werden, falls bei der ersten Anwendung der Gauß-Funktion folgende Fehlermeldung auftritt: **"Singularität bei der Auswertung dieses Ausdrucks festgestellt. Möglicherweise dividieren sie durch NULL."**

```
Gauß_prüf(K) := | for r ∈ 0 .. rows(K) − 1
                |     continue if K_{r,0} = 0
                |     ztmp1 ← (K^T)^⟨0⟩
                |     ztmp2 ← (K^T)^⟨r⟩
                |     for c ∈ 0 .. cols(K) − 1
                |         K_{0,c} ← ztmp2_{c,0}
                |         K_{r,c} ← ztmp1_{c,0}
                |     break
                | for r ∈ 1 .. rows(K) − 1
                |     continue if K_{r,1} = (K_{1,0} / K_{0,0}) · K_{0,1}
                |     ztmp1 ← (K^T)^⟨1⟩
                |     ztmp2 ← (K^T)^⟨r⟩
                |     for c ∈ 0 .. cols(K) − 1
                |         K_{1,c} ← ztmp2_{c,0}
                |         K_{r,c} ← ztmp1_{c,0}
                |     break
                | K
```

Wir machen die Probe mit der obrigen Matrix **K**.

$$K := \begin{pmatrix} 1 & 2 & 3 & 5 \\ 3 & 6 & 8 & 4 \\ 5 & -2 & 4 & 3 \end{pmatrix}$$

Gauß(K) = ∎

$K_{neu} := \text{Gauß\_prüf}(K)$

$$\text{Gauß}(K_{neu}) = \begin{pmatrix} 1 & 0 & 0 & -11.5 \\ 0 & 1 & 0 & -8.25 \\ 0 & 0 & 1 & 11 \end{pmatrix}$$

Das Problem wurde erfolgreich gelöst.

Falls keine Lösung für das Gleichungssystem existiert, so hilft natürlich auch dieses zweite Programm nichts. Eine Kontrolle mit der Methode der Kehrmatrix ist also im Zweifelsfall empfehlenswert.
Bei Gleichungssystemen mit vielen Unbekannten (z.B. bei der FEM können mehrere Millionen auftreten) ist das durchtauschen der Zeilen natürlich nicht geeignet. Hier kann z.B. eine kleiner Wert (in unserem Fall im Bereich 10E-6) zu den Werten der Hauptdiagonalen dazuaddiert werden. Die Genauigkeit wird nur sehr gering beeinträchtigt (bei der FEM würde es sich ohnehin um ein Näherungsverfahren handeln).

Kurz wollen wir auch diese Möglichkeit ausschöpfen. Wir addieren zu den Werten der Hauptiagonale den Wert 10E-6.

$$K := \begin{pmatrix} 1 + 10^{-6} & 2 & 3 & 5 \\ 3 & 6 + 10^{-6} & 8 & 4 \\ 5 & -2 & 4 + 10^{-6} & 3 \end{pmatrix} \qquad \text{Gauß}(K) = \begin{pmatrix} 1 & 0 & 0 & -11.5 \\ 0 & 1 & 0 & -8.25 \\ 0 & 0 & 1 & 11 \end{pmatrix}$$