Hausarbeit

Überführung eines Programms zur FEM-Berechnung von 2-Dimensional in 3-Dimensional

Pavel Malkov - 1396622

Inhaltsverzeichnis

1 Einleitung

Im Folgenden wird die Überführung des zur Verfügung gestellten Programms zur Bestimmung der Temperaturverteilung vom 2-Dimensionalen ins 3-Dimensionale beschrieben. Die theoretische Grundlage liefert das Script *Anwendung der Finite–Elemente–Methode*. Es wird die Temperaturverteilung in einem Quader berechnet, die Vernetzung des Körpers wird nicht betrachtet, es werden fertige Datensätze benutzt. Es wurde Absichtlich ein sehr einfaches Problem gewählt, sodass das Ergebnis der Berechnung mit dem theoretischen Wert verglichen werden kann. Das Programms setzt sich aus folgenden Teilen zusammen:

- Hauptprogramm: *FEM+Diri_ok.F* (Speicherverwaltung, Aufruf von Unterprogrammen)
- Unterprogramm: *fem_input.F* (Einelsen des Netzes, lokales Gleichungssystem, Assemblierung)
- Unterprogramm: *fem_randbed.F* (Einsetzen der Randbedingung ins das Gleichungssystem aus *fem_input.F*)
- Unterprogramme: *gs_solve.F*, *jac_solve.F* (iterative Löser für lineare Gleichungssysteme)
- Unterprogramm: *dsecnd.F* (Hilfsunterprogramm für LAPACK)
- Unterprogramm: *printr.f* (formatierte Ausgabe)
- Makefile: *FEM+Diri_ok+gf-LAPACK.mk*

Auszüge aus Quelltexten wurden zur besseren Lesbarkeit umformatiert.

2 Makefile

2.1 Aufbau

Das Makefile dient dazu die Compiler-Anweisung zu erstellen. Dabei wird die Zeile, mit der dieser Aufgerufen wird aus folgenden Teilen zusammengesetzt:

- CF: Compiler-Typ
- fn: Name des Hauptprogramms
- FFLAGS: Compilerschalter und Prepozessoranweisungen
- SRC: Quelldateinamen
- OBJ: Objektdateinamen

Ein Beispiel für ein einfaches Makefile mit diesen Komplonenten wäre:

```
CF=gfortran
FFLAGSPP=-g -Wall -Dfortran90 -Dplevel1 -DGaussSeidel -fdefault-real-8
fn=FEM+Diri_ok
SRC=${fn}.F fem_input.F fem_randbed.F gs_solve.F mg.f printr.f dsecnd.F
OBJ=${fn}.o fem_input.o fem_randbed.o gs_solve.o mg.o printr.o dsecnd.o
${fn}.exe: $(OBJ)
    $(CF) ${FFLAGS} -o ${fn}.exe $(OBJ)
clean:
    rm $(OBJ) ${fn}.exe
.f.o :
    $(CF) ${FFLAGS} -c $<
.F.o :
    $(CF) ${FFLAGSPP} -c $<
```

3 Hauptprogramm

3.1 Das Hauptprogramm

Das Hauptprogramm FEM+Diri_ok.F dient zur Verwaltung globaler Variablen und Felder und zum aufrufen von Unterprogrammen, die die Teilaufgaben erledigen.

3.2 Dimensionierung und Allokierung

Zunächst muss für globalen Felder variable Speichermengen bereitgestellt werden, hierzu ist es notwendig, die Größe des zu berechnenden Problems festzustellen. Dies wurde durch die Function *countEntries* implementiert:

```
function countEntries(strng) result(n)
implicit none
character(len=128) :: strng
integer(kind=4) :: i, ios, n
real(kind=4) :: dummy
write(unit = 6, fmt = *) 'countEntries: ', strng
i = 1
ios = 0
open(unit = 33, iostat = ios, file = strng, status = 'old')
if (ios .ne. 0) then
    write(unit = 6, fmt= *)'countEntries: Fehler beim Dateipfad'
else
    do while (ios .eq. 0)
        read(unit = 33, fmt = *, iostat = ios) dummy
        i = i + 1
    end do
    n = i - 2
    write(unit = 6, fmt = *) 'Anzahl der Einträge: ', n
    write(unit = 6, fmt = *) '--------'
end if
close(unit = 33, status = 'keep')
end function countEntries
```

Mit den Ergebnissen dieser Function lassen sich nun die Größen der Felder bestimmen und dadurch der Speicher allokieren:

```
    path = 'mesh.nodes'
    pdim = countEntries(path)
    path = 'mesh.elements'
    edim = countEntries(path)
    path = 'mesh.boundary'
    bdim = countEntries(path)

    allocate ( p(pdim, 3) )
    allocate ( A(pdim, pdim) )
    allocate ( ditlpu(pdim, pdim) )
    allocate ( b(pdim) )
    allocate ( blokal(4) )
    allocate ( ditb(pdim) )
    allocate ( filter(pdim) )
```

```
#ifdef LUZerlegung
    allocate ( ipiv(pdim) )
#endif
    allocate ( u(pdim) )
    allocate ( xneu(pdim) )
    allocate ( e(edim, 4) )
    allocate ( be(bdim, 3) )
    allocate ( stmp(4, 4) )
```

4 Input

4.1 Einlesen der Netzdaten

Das Unterprogramm *fem_input.F* enthält den größten Teil der Anpassungen zur Umwandlung von 2-D in 3-D. Dieses Unterprogramm liest ein und bereinigt die Netzdaten, erzeugt daraus die Lokalen Steifigkeitsmatrizen und Lastvektoren und assembliert diese zur globalen Steifigkeitsmatrix und Lastvektor. Es setzt das Grundgerüst des letztendlich zu lösenden Gleichungssystems zusammen. Die Netzdaten sind auf drei Dateien verteilt:

- *mesh.nodes*, die die Koordinaten aller Knoten enthält
- *mesh.elements*, in der Tetraederelemente als Auflistung ihrer Eckknoten zu finden sind und
- *mesh.boundary*, in der die dreieckigen Randelemente als Auflistung ihrer Eckknoten sind

Es wurde beachtet, dass in der Datei *mesh.boundary* aufgrund der Art der Erzeugung des Netzes auch Elemente im Inneren des untersuchten Körpers liegen, diese mussten vor dem setzen der Randbedingung herausgefiltert werden. Dazu wurde ein Feld namens *filter* angelegt, in dem die Liste aller Knoten abgespeichert wird, wobei die tatsächlichen Randknoten ein negatives Vorzeichen bekommen. Dies wird auch später beim Einbau der Randbedingung von Nutzen sein, da diese Methode gleichzeitig gewährleistet, dass die Wiederholung von Knoten, die sich mehrere Elemente teilen, kein Rolle mehr spielt. Da der notwendige Speicher bereits im hauptprogramm allokiert wurde, können jetzt die Daten direkt aus den Dateien in die entsprechenden Felder geschrieben werden.

```
!-------WRITE NODES-----------------------------------------
ios = 0
open(unit = 33, iostat = ios, file = 'mesh.nodes')
open(unit = 34, file = 'nodes.dem')
do ir = 1, pdim
    filter(ir) = ir
    read(unit = 33, fmt = *, iostat = ios) dummyInt, dummyReal,
    &   p(ir, 1), p(ir, 2), p(ir, 3)
    write(unit = 34, fmt = *) p(ir, 1), p(ir, 2),
    &       p(ir, 3)
end do
close(unit = 33, status = 'keep')
close(unit = 34, status = 'keep')
!------WRITE ELEMENT NODES-----------------------------------
open(unit = 33, iostat = ios, file = 'mesh.elements',
&                   status = 'old')
open(unit = 34, file = 'elements')
if (ios .ne. 0) then
    write(unit = 6, fmt = *) 'Fehler beim Dateipfad'
else
    do ir = 1, edim
        read(unit = 33, fmt = *, iostat = ios) dummyInt,
        &   dummyInt, dummyInt,
        &   e(ir, 1), e(ir, 2), e(ir, 3), e(ir, 4)
        write(unit = 34, fmt = *) e(ir, 1),  e(ir, 2),
        &           e(ir, 3), e(ir, 4)
    end do
end if
close(unit = 33, status = 'keep')
close(unit = 34, status = 'keep')
```

```
open(unit = 34, file = 'elements.dem')
do ir = 1, edim
    do jr = 1, 4
        node = e(ir, jr)
        write(unit = 34, fmt = *) p(node, 1),
        &          p(node, 2), p(node, 3)
    end do
end do
close(unit = 34, status = 'keep')
```

Um das Einlesen unterschiedlicher Quaderförmiger Netze zu ermöglichen, wurde die Bestimmung des Randes zur Bereinigung der Daten variabel gestaltet.

```
!------WRITE BOUNDARY NODES---------------------------------
xmax = -10000.0d+00
xmin =  10000.0d+00
ymax = -10000.0d+00
ymin =  10000.0d+00
zmax = -10000.0d+00
zmin =  10000.0d+00
open(unit = 33, file = 'minmax')
do i = 1, pdim
    xmax = dmax1(xmax, p(i, 1))
    xmin = dmin1(xmin, p(i, 1))
    ymax = dmax1(ymax, p(i, 2))
    ymin = dmin1(ymin, p(i, 2))
    zmax = dmax1(zmax, p(i, 3))
    zmin = dmin1(zmin, p(i, 3))
end do
write(unit =33, fmt= *) 'max xyz: ', xmax, ymax, zmax
write(unit =33, fmt= *) 'min xyz: ', xmin, ymin, zmin
close(unit =33, status = 'keep')
open(unit = 35, iostat = ios, file = 'mesh.boundary',
&                  status = 'old')
open(unit = 36, file = 'boundary')
if (ios .ne. 0) then
    write(unit = 6, fmt = *) 'Fehler beim Dateipfad'
else
    do ir = 1,bdim
        read(unit = 35, fmt = *, iostat = ios) dummyInt,
        &   dummyInt, dummyInt, dummyInt, dummyInt,
        &              be(ir, 1), be(ir, 2), be(ir, 3)
        write(unit = 36, fmt = *) be(ir, 1),  be(ir, 2),
        &           be(ir, 3)
    end do
end if
close(unit = 35, status = 'keep')
close(unit = 36, status = 'keep')
```

```
open(unit = 37, file = 'filter')
open(unit = 38, file = 'boundary.dem')
do ir = 1, bdim
    do jr = 1, 3
        node = be(ir, jr)
        if (
        &            ((p(node, 1) .gt. xmin) .and. (p(node, 1) .lt. xmax))
        &      .and. ((p(node, 2) .gt. ymin) .and. (p(node, 2) .lt. ymax))
        &      .and. ((p(node, 3) .gt. zmin) .and. (p(node, 3) .lt. zmax))
        &           )
        &  then
           cycle
        else
           if (filter(node) .lt. 0) then
             cycle
           else
           filter(node) = -filter(node)
           end if
        end if
        write(unit = 38, fmt = *) p(node, 1),
        &           p(node, 2), p(node, 3)
    end do
end do
        close(unit = 37, status = 'keep')
close(unit = 38, status = 'keep')
open(unit = 33, file = 'filter')
do i = 1, pdim
    write(unit = 33, fmt = *) filter(i)
end do
close(unit = 33, status = 'keep')
```

4.2 Erzeugung der Lokalen Steifigkeitsmatrix

Nachdem das Netz erfolgreich eingelesen wurde muss noch aus den Knotenkoordinaten für jedes Element eine Lokale Steifigkeitsmatrix erzeugt werden. (Die eingerahmten stellen entsprechen Zuweisungen im Quelltext von *fem_input.F*)

$$u(x,y,z) = \sum_i c_i \phi_i = c_1 + c_2 x + c_3 y + c_4 z = (c_1,\, c_2,\, c_3,\, c_4) \cdot \begin{pmatrix} 1 \\ x \\ y \\ z \end{pmatrix} = \underline{c} \cdot \vec{r}$$

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} x_1 \\ y_1 \\ z_1 \end{pmatrix} + \begin{pmatrix} x_2 - x_1 & x_3 - x_1 & x_4 - x_1 \\ y_2 - y_1 & y_3 - y_1 & y_4 - y_1 \\ z_2 - z_1 & z_3 - z_1 & z_4 - z_1 \end{pmatrix} \cdot \begin{pmatrix} \xi \\ \eta \\ \zeta \end{pmatrix} = \begin{pmatrix} x_1 \\ y_1 \\ z_1 \end{pmatrix} + \begin{pmatrix} (x_2 - x_1)\,\xi & (x_3 - x_1)\,\eta & (x_4 - x_1)\,\zeta \\ (y_2 - y_1)\,\xi & (y_3 - y_1)\,\eta & (y_4 - y_1)\,\zeta \\ (z_2 - z_1)\,\xi & (z_3 - z_1)\,\eta & (z_4 - z_1)\,\zeta \end{pmatrix} \cdot \begin{pmatrix} \xi \\ \eta \\ \zeta \end{pmatrix}$$

mit

$$\xi = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \quad , \quad \eta = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \quad , \quad \zeta = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \quad , \quad \vec{w} = \begin{pmatrix} \xi \\ \eta \\ \zeta \end{pmatrix}$$

$$\vec{r} = \vec{r_1} + \underline{\underline{K}} \cdot \vec{w} \Rightarrow \vec{w} = \underline{\underline{K}}^{-1} \cdot (\vec{r} - \vec{r_1}) \quad , \quad \underline{\underline{J}} = \underline{\underline{K}}^{T}$$

$$\underline{\underline{J}} = \begin{pmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} & \frac{\partial z}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} & \frac{\partial z}{\partial \eta} \\ \frac{\partial x}{\partial \zeta} & \frac{\partial y}{\partial \zeta} & \frac{\partial z}{\partial \zeta} \end{pmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix} \quad , \quad det\, \underline{\underline{J}} = aei + bfg + cdh - gec - hfa - idb$$

$$\vec{w} = \frac{\underline{\underline{J}}}{det\, \underline{\underline{J}}} \cdot Adj\, \underline{\underline{J}} \cdot (\vec{r} - \vec{r_1}) = \frac{1}{det\, \underline{\underline{J}}} \cdot (\vec{r} - \vec{r_1}) \cdot \begin{pmatrix} ei - fh & fg - di & dh - eg \\ ch - bi & ai - cg & bg - dh \\ bf - ce & cd - af & ae - bd \end{pmatrix}$$

$$= \begin{pmatrix} \varphi_{11} & \varphi_{12} & \varphi_{13} \\ \varphi_{21} & \varphi_{22} & \varphi_{23} \\ \varphi_{31} & \varphi_{32} & \varphi_{33} \end{pmatrix} \quad , \quad \tilde{\underline{\underline{J}}}\left(\underline{\underline{K}}^{-1}\right) = \begin{pmatrix} \frac{\partial \xi}{\partial x} & \frac{\partial \eta}{\partial x} & \frac{\partial \zeta}{\partial x} \\ \frac{\partial \xi}{\partial y} & \frac{\partial \eta}{\partial y} & \frac{\partial \zeta}{\partial x} \\ \frac{\partial \xi}{\partial z} & \frac{\partial \eta}{\partial z} & \frac{\partial \zeta}{\partial x} \end{pmatrix}$$

$$u_\xi = \frac{\partial u(\xi,\eta,\zeta)}{\partial \xi} = \alpha_2 \quad , \quad u_\eta = \frac{\partial u(\xi,\eta,\zeta)}{\partial \eta} = \alpha_3 \quad , \quad u_\zeta = \frac{\partial u(\xi,\eta,\zeta)}{\partial \zeta} = \alpha_4$$

$$\frac{\partial}{\partial x} u(x,y,z) = \left(\frac{\partial u(\xi,\eta,\zeta)}{\partial \xi} \cdot \frac{\partial \xi(x,y,z)}{\partial x}\right) + \left(\frac{\partial u(\xi,\eta,\zeta)}{\partial \eta} \cdot \frac{\partial \eta(x,y,z)}{\partial x}\right) + \left(\frac{\partial u(\xi,\eta,\zeta)}{\partial \zeta} \cdot \frac{\partial \zeta(x,y,z)}{\partial x}\right)$$

$$\frac{\partial}{\partial y} u(x,y,z) = \left(\frac{\partial u(\xi,\eta,\zeta)}{\partial \xi} \cdot \frac{\partial \xi(x,y,z)}{\partial y}\right) + \left(\frac{\partial u(\xi,\eta,\zeta)}{\partial \eta} \cdot \frac{\partial \eta(x,y,z)}{\partial y}\right) + \left(\frac{\partial u(\xi,\eta,\zeta)}{\partial \zeta} \cdot \frac{\partial \zeta(x,y,z)}{\partial y}\right)$$

$$\frac{\partial}{\partial z} u(x,y,z) = \left(\frac{\partial u(\xi,\eta,\zeta)}{\partial \xi} \cdot \frac{\partial \xi(x,y,z)}{\partial z}\right) + \left(\frac{\partial u(\xi,\eta,\zeta)}{\partial \eta} \cdot \frac{\partial \eta(x,y,z)}{\partial z}\right) + \left(\frac{\partial u(\xi,\eta,\zeta)}{\partial \zeta} \cdot \frac{\partial \zeta(x,y,z)}{\partial z}\right)$$

$$\frac{\partial}{\partial x} = \frac{\varphi_{11}}{det\,\underline{\underline{J}}}\alpha_2 + \frac{\varphi_{21}}{det\,\underline{\underline{J}}}\alpha_3 + \frac{\varphi_{31}}{det\,\underline{\underline{J}}}\alpha_4$$

$$\frac{\partial}{\partial y} = \frac{\varphi_{12}}{det\,\underline{\underline{J}}}\alpha_2 + \frac{\varphi_{22}}{det\,\underline{\underline{J}}}\alpha_3 + \frac{\varphi_{32}}{det\,\underline{\underline{J}}}\alpha_4$$

$$\frac{\partial}{\partial z} = \frac{\varphi_{13}}{det\,\underline{\underline{J}}}\alpha_2 + \frac{\varphi_{23}}{det\,\underline{\underline{J}}}\alpha_3 + \frac{\varphi_{33}}{det\,\underline{\underline{J}}}\alpha_4$$

$$\int\limits_A D\left\{\left(\frac{\partial u\left(\vec{r}\right)}{\partial x}\right)^2 + \left(\frac{\partial u\left(\vec{r}\right)}{\partial y}\right)^2 + \left(\frac{\partial u\left(\vec{r}\right)}{\partial z}\right)^2\right\} - q\left(\vec{r}\right)\cdot u\left(\vec{r}\right)\,d\left(\vec{r}\right) = 0$$

da die Knotenkoordinaten konstant sind, ergibt sich:

$$\begin{aligned} D\cdot I &= D\cdot\int\left[u_x^2+u_y^2+u_z^2\right]\,d^3V \\ &= D\cdot\frac{1}{det\,\underline{\underline{J}}}\int\limits_{\overline{V}}\left[\left(u_\xi\xi_x+u_\eta\eta_x+u_\zeta\zeta_x\right)^2+\left(u_\xi\xi_y+u_\eta\eta_y+u_\zeta\zeta_y\right)^2+\left(u_\xi\xi_z+u_\eta\eta_z+u_\zeta\zeta_z\right)^2\right] \\ &= D\cdot\frac{1}{det\,\underline{\underline{J}}}\left\{\alpha_2^2a+\alpha_3^2b+\alpha_4^2c+2\alpha_2\alpha_3d+2\alpha_2\alpha_4e+2\alpha_3\alpha_4f\right\}\int\limits_0^{\xi=1}\int\limits_0^{\zeta=1-\xi}\int\limits_0^{\eta=1-\xi-\zeta}1\quad d\eta d\zeta d\xi \end{aligned}$$

mit

$a = \varphi_{11}^2+\varphi_{12}^2+\varphi_{13}^2$

$b = \varphi_{21}^2+\varphi_{22}^2+\varphi_{32}^2$

$c = \varphi_{31}^2+\varphi_{32}^2+\varphi_{33}^2$

$d = \varphi_{11}\varphi_{21}+\varphi_{12}\varphi_{22}+\varphi_{13}\varphi_{23}$

$e = \varphi_{11}\varphi_{31}+\varphi_{12}\varphi_{32}+\varphi_{13}\varphi_{33}$

$f = \varphi_{21}\varphi_{31}+\varphi_{22}\varphi_{32}+\varphi_{23}\varphi_{33}$

Nun kann man das Integral über den Einheitstetraeder analytisch bestimmen und damit vermeiden es für jedes Element gesondert tun zu müssen

$$\int_0^{\xi=1}\int_0^{\zeta=1-\xi}\int_0^{\eta=1-\xi-\zeta} 1 \quad d\eta d\zeta d\xi = \int_0^1 \int_0^{1-\xi} 1-\xi-\zeta \quad d\zeta d\xi$$
$$= \int_0^1 \zeta - \xi\zeta - \frac{1}{2}\zeta^2\Big|_0^{1-\xi} \quad d\xi$$
$$= \int_0^1 \frac{1}{2} - \xi - \frac{1}{2}\xi^2 + \frac{1}{6}\xi \quad d\xi$$
$$= \frac{1}{6}$$

$$I_1 = \frac{1}{det\,\underline{\underline{J}}} \cdot \alpha_2^2 \frac{a}{6} = \frac{1}{det\,\underline{\underline{J}}} \cdot \frac{a}{6} \cdot \begin{pmatrix}\alpha_1 & \alpha_2 & \alpha_3 & \alpha_4\end{pmatrix} \cdot \underline{\underline{S}}_1' \cdot \begin{pmatrix}\alpha_1\\ \alpha_2\\ \alpha_3\\ \alpha_4\end{pmatrix}$$

mit

$$\underline{\underline{S}}_1' = \frac{a}{6}\begin{pmatrix}0&0&0&0\\0&1&0&0\\0&0&0&0\\0&0&0&0\end{pmatrix}$$

analog dazu:

$$\underline{\underline{S}}_2' = \frac{b}{6}\begin{pmatrix}0&0&0&0\\0&0&0&0\\0&0&1&0\\0&0&0&0\end{pmatrix} \quad , \quad \underline{\underline{S}}_3' = \frac{c}{6}\begin{pmatrix}0&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&0&1\end{pmatrix} \quad , \quad \underline{\underline{S}}_4' = \frac{d}{6}\begin{pmatrix}0&0&0&0\\0&0&1&0\\0&1&0&0\\0&0&0&0\end{pmatrix}$$

$$\underline{\underline{S}}_5' = \frac{e}{6}\begin{pmatrix}0&0&0&0\\0&0&0&1\\0&0&0&0\\0&1&0&0\end{pmatrix} \quad , \quad \underline{\underline{S}}_6' = \frac{f}{6}\begin{pmatrix}0&0&0&0\\0&0&0&0\\0&0&0&1\\0&0&1&0\end{pmatrix}$$

$$\begin{array}{llllll} u_1\,(0,0,0) = & +\alpha_1 & & & \\ u_2\,(1,0,0) = & +\alpha_1 & +\alpha_2 & & \\ u_3\,(0,1,0) = & +\alpha_1 & & +\alpha_3 & \\ u_4\,(0,0,1) = & +\alpha_1 & & & +\alpha_4 \end{array}$$

mit

$u_1 = \alpha_1$
$u_2 = u_1 + \alpha_2$
$u_3 = u_1 + \alpha_3$
$u_4 = u_1 + \alpha_4$

$$\Rightarrow \begin{cases} \alpha_1 = u_1 \\ \alpha_2 = u_2 - u_1 \\ \alpha_3 = u_3 - u_1 \\ \alpha_4 = u_4 - u_1 \end{cases}$$

In Matrixschreibweise:

$$\vec{u} = \underline{\underline{A}}^{-1}\vec{\alpha} = \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \alpha_4 \end{pmatrix}$$

$$\Rightarrow \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \alpha_4 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ -1 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{pmatrix} = \vec{\alpha} = \underline{\underline{A}}\vec{u}$$

$$\begin{aligned} I &= \vec{\alpha}^T \cdot \left(a\underline{\underline{S}}_1^{'} + b\underline{\underline{S}}_2^{'} + c\underline{\underline{S}}_3^{'} + d\underline{\underline{S}}_4^{'} + e\underline{\underline{S}}_5^{'} + f\underline{\underline{S}}_6^{'} \right) \cdot \vec{\alpha} \\ &= \vec{u}^T \underline{\underline{A}}^T \cdot \left(a\underline{\underline{S}}_1^{'} + b\underline{\underline{S}}_2^{'} + c\underline{\underline{S}}_3^{'} + d\underline{\underline{S}}_4^{'} + e\underline{\underline{S}}_5^{'} + f\underline{\underline{S}}_6^{'} \right) \cdot \vec{u}\underline{\underline{A}} \end{aligned}$$

$$\underline{\underline{S}} = \underline{\underline{A}}^T \cdot \left(a\underline{\underline{S}}_1^{'} + b\underline{\underline{S}}_2^{'} + c\underline{\underline{S}}_3^{'} + d\underline{\underline{S}}_4^{'} + e\underline{\underline{S}}_5^{'} + f\underline{\underline{S}}_6^{'} \right) \cdot \underline{\underline{A}}$$

damit ergeben sich die Teilsteifigkeitsmatrizen zu:

$$a\underline{\underline{S}}_1 = \frac{a}{6}\begin{pmatrix} 1 & -1 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \quad , \quad b\underline{\underline{S}}_2 = \frac{b}{6}\begin{pmatrix} 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \quad , \quad c\underline{\underline{S}}_3 = \frac{c}{6}\begin{pmatrix} 1 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 1 \end{pmatrix}$$

$$d\underline{\underline{S}}_4 = \frac{d}{6}\begin{pmatrix} 2 & -1 & -1 & 0 \\ -1 & 0 & 1 & 0 \\ -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \quad , \quad e\underline{\underline{S}}_5 = \frac{e}{6}\begin{pmatrix} 2 & -1 & 0 & -1 \\ -1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 \end{pmatrix} \quad , \quad f\underline{\underline{S}}_6 = \frac{f}{6}\begin{pmatrix} 2 & 0 & -1 & -1 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 1 \\ -1 & 0 & 1 & 0 \end{pmatrix}$$

und damit die lokale Steifigkeitsmatrix:

$$\boxed{\underline{\underline{S}}_e = \frac{1}{det\,\underline{\underline{J}}} \cdot \frac{1}{6} \cdot \begin{pmatrix} a+b+c+2d+2e+2f & -a-d-e & -b-d-f & -c-e-f \\ -a-d-e & a & d & e \\ -b-d-f & d & b & f \\ -c-e-f & e & f & c \end{pmatrix}}$$

Die eingerahmten Bereiche können nun im Quelltext von *fem_input.F* eingefügt werden:

```
      do ei = 1, edim

          x1 = p(e(ei, 1), 1)
          y1 = p(e(ei, 1), 2)
          z1 = p(e(ei, 1), 3)
          x2 = p(e(ei, 2), 1)
          y2 = p(e(ei, 2), 2)
          z2 = p(e(ei, 2), 3)
          x3 = p(e(ei, 3), 1)
          y3 = p(e(ei, 3), 2)
          z3 = p(e(ei, 3), 3)
          x4 = p(e(ei, 4), 1)
          y4 = p(e(ei, 4), 2)
          z4 = p(e(ei, 4), 3)

          a1 = x2 - x1
          b1 = y2 - y1
          c1 = z2 - z1
          d1 = x3 - x1
          e1 = y3 - y1
          f1 = z3 - z1
          g1 = x4 - x1
          h1 = y4 - y1
          i1 = z4 - z1

          det_J = a1*e1*i1 + b1*f1*g1 + c1*d1*h1 -  g1*e1*c1 - h1*f1*a1
          &     - i1*d1*b1

          phi11 = e1*i1 - f1*h1
          phi12 = f1*g1 - d1*i1
          phi13 = d1*h1 - e1*g1
          phi21 = c1*h1 - b1*i1
          phi22 = a1*i1 - c1*g1
          phi23 = b1*g1 - a1*h1
          phi31 = b1*f1 - c1*e1
          phi32 = c1*d1 - a1*f1
          phi33 = a1*e1 - b1*d1
          a2 = phi11**2 + phi12**2 + phi13**2
          b2 = phi21**2 + phi22**2 + phi23**2
          c2 = phi31**2 + phi32**2 + phi33**2
          d2 = phi11*phi21 + phi12*phi22 + phi13*phi23
          e2 = phi11*phi31 + phi12*phi32 + phi13*phi33
          f2 = phi21*phi31 + phi22*phi32 + phi23*phi33
```

```
        stmp(1,1) = a2 + b2 +c2 +2*d2 + 2*e2 + 2*f2
        stmp(1,2) = -a2 - d2 - e2
        stmp(1,3) = -b2 - a2 - f2
        stmp(1,4) = -c2 - e2 - f2
        stmp(2,1) = -a2 - d2 - e2
        stmp(2,2) = a2
        stmp(2,3) = d2
        stmp(2,4) = e2
        stmp(3,1) = -b2 - d2 - f2
        stmp(2,2) = d2
        stmp(2,3) = b2
        stmp(2,4) = f2
        stmp(1,4) = -c2 - e2 - f2
        stmp(2,2) = e2
        stmp(2,3) = f2
        stmp(2,4) = c2
        [...]
    end do
```

4.3 Erzeugung des Lastvektors

Die rechte Seite ergibt sich aus der Wärmeleitungsgleichung.
allgemein:

$$\int_V \frac{D}{2}\left[\left(\frac{\partial u(\vec{r})}{\partial x}\right)^2+\left(\frac{\partial u(\vec{r})}{\partial y}\right)^2+\left(\frac{\partial u(\vec{r})}{\partial z}\right)^2\right]-q(\vec{r})\cdot u(\vec{r})\,d^3\vec{r}=\int\frac{D}{2}\left[u_x^2+u_y^2+u_z^2\right]-q$$

im lokalen Koordinatensystem:

$$\int_V q\cdot\vec{u}\,d^3\vec{r}=\int_V q\,(\alpha_1+\alpha_2\xi+\alpha_3\eta+\alpha_4\zeta)\cdot det\underline{\underline{J}}\quad d\eta\,d\zeta\,d\xi$$

mit $q=4$

$$q\cdot\begin{pmatrix}\alpha_1 & \alpha_2 & \alpha_3 & \alpha_4\end{pmatrix}\cdot det\underline{\underline{J}}\cdot\begin{pmatrix}\left(\int_0^{\xi=1}\int_0^{\zeta=1-\xi}\int_0^{\eta=1-\xi-\zeta}1\quad d\eta\,d\zeta\,d\xi\right)\\ \left(\int_0^{\xi=1}\int_0^{\zeta=1-\xi}\int_0^{\eta=1-\xi-\zeta}\xi\quad d\eta\,d\zeta\,d\xi\right)\\ \left(\int_0^{\xi=1}\int_0^{\zeta=1-\xi}\int_0^{\eta=1-\xi-\zeta}\eta\quad d\eta\,d\zeta\,d\xi\right)\\ \left(\int_0^{\xi=1}\int_0^{\zeta=1-\xi}\int_0^{\eta=1-\xi-\zeta}\zeta\quad d\eta\,d\zeta\,d\xi\right)\end{pmatrix}=\frac{det\,\underline{\underline{J}}}{6}\cdot\vec{\alpha}\cdot\begin{pmatrix}4\\1\\1\\1\end{pmatrix}$$

$$=\frac{det\,\underline{\underline{J}}}{6}\cdot\vec{\alpha}^T\cdot\vec{b'}$$

$$=\frac{det\,\underline{\underline{J}}}{6}\cdot\vec{u}^T A^T\vec{b'}$$

$$=\frac{det\,\underline{\underline{J}}}{6}\cdot\vec{u}^T\cdot\begin{pmatrix}1\\1\\1\\1\end{pmatrix}$$

Damit ergeben sich für die linke Seite der Vorfaktor $\frac{1}{6}\cdot\frac{1}{det\,\underline{\underline{J}}}$ und für die rechte Seite $\frac{1}{6}\cdot det\,\underline{\underline{J}}$

4.4 Assemblierung

Nun werden für jedes Element die lokalen Steifigkeitsmatrizen und Lastvektoren zu globalen zusammengesetzt.

```
[...]
blokal = det_J / 6.0
stmp = stmp /(det_J*6.0)
do i = 1, 4
    l = e(ei, i)
    do j = 1, i
        r = e(ei, j)
        A(l, r) = A(l, r) + stmp(i,j)
        A(r, l) = A(l, r)
    end do
b(l) = b(l) + blokal(i)
end do
[...]
```

Es müssen noch die bekannten Randbedingungen eingesetzt werden, sodass das Gleichungssystem eindeutig lösbar ist.

5 Randbedingung

5.1 Einsetzen der Randbedingung

Die Randbedingung ist mit $r = 20 - 2y^2 + x^3y^2 - xy^3 + z^3x - zx^3$gegeben. Diese wird durch zwei Schleifen ins Gleichungssystem eingearbeitet:

```
do ki = 1, pdim
    if (filter(ki) .lt. 0) then
        do j = 1, pdim
           randwert = 20.0 - 2.0*p(ki,2)**2 + p(ki,1)**3*p(ki,2)
           &                         -
           p(ki,1)*p(ki,2)**3 + p(ki,3)**3*p(ki,1)
           &                         - p(ki,3)*p(ki,1)**3
           b(j)     = b(j) - A(j, ki) * randwert
           A(j, ki) = 0.0
           A(ki, j) = 0.0
        end do
        A(ki, ki) = 1.
    end if
end do
do ki = 1, pdim
    if (filter(ki) .lt. 0) then
        b(ki) = 20.0 - 2.0*p(ki,2)**2 + p(ki,1)**3*p(ki,2)
        &                  - p(ki,1)*p(ki,2)**3 + p(ki,3)**3*p(ki,1)
        &                  - p(ki,3)*p(ki,1)**3
    end if
end do
```

6 Löser

6.1 Bestimmung und Ausgabe des Fehlers

Das nun entstandene lineare Gleichungssystem ist eindeutig lösbar und kann beispielsweise durch den SOR-Algorithmus gelöst werden. Hierzu steht das Unterprogramm *gs_solve.F* zur Verfügung. Dadurch, dass das gewählte Problem so einfach beschaffen ist, kann man nun das durch das Programm ermittelte Ergebnis für jeden Knoten mit dem theoretischen Wert vergleichen und den entstandenen Fehler ermitteln. Dazu wurde eine Schleife ins Unterprogramm *gs_solve.F* eingefügt:

```
open(unit = 33, file = 'delta.dat')
do i = 1, ndim

    u= 20.0 - 2.0*p(i,2)**2 + p(i,1)**3*p(i,2)
    &                  - p(i,1)*p(i,2)**3 + p(i,3)**3*p(i,1)
    &                  - p(i,3)*p(i,1)**3
    delta = abs(x(i) - u)
    print*,'delta: ', i, delta
    write(unit = 33, fmt = *) i, delta

end do
close(unit = 33, status = 'keep')
call system('gnuplot delta.dem')
```

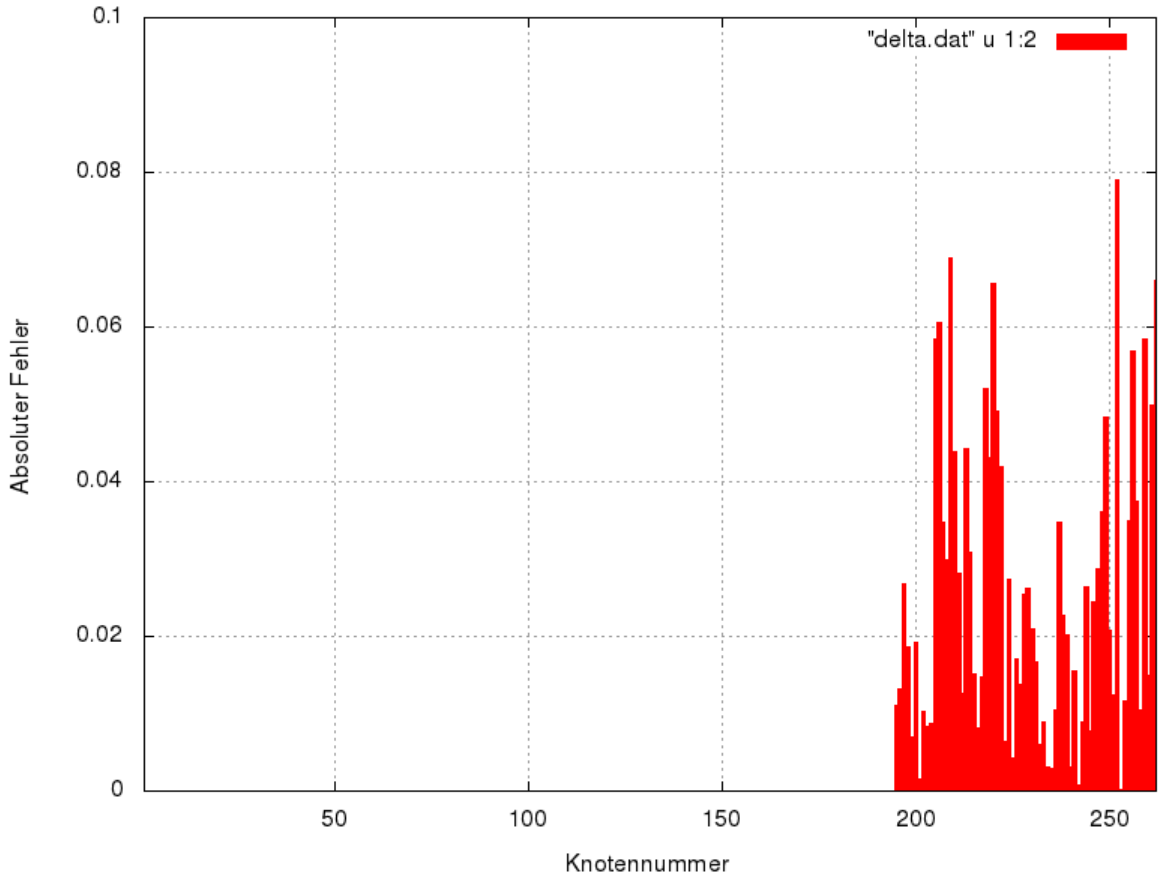
Der Absolute Fehler für jeden Knoten wird in die Datei delta.dat geschrieben, die Ausgabe erfolgt durch Gnuplot über eine Demodatei *delta.dem:*

```
set term x11
set grid
# set xrange[1:262]
# set yrange[0.0:0.1]
set style fill solid
set xlabel "Knotennummer"
set ylabel "Absoluter Fehler"
# set terminal png size 800,600
enhanced font "Helvetica, 12" #
set output "delta.png"  plot "delta.dat" u 1:2 with boxes
pause -1 "Hit return to continue"
```

7 Schlussbetrachtung

7.1 Betrachtung des ermittelten Fehlers

Die Ausgabe liefert einen Fehler der Größenordnung10^{-2} für innere Knoten und 10^{-15} für Randknoten und damit im erwarteten Bereich.



Das Programm müsste nun noch mit feineren Netzen (und auf leistungsstärkeren Maschinen) getestet werden. Eine Optimierungsmöglichkeit wäre noch das Gleichungssystem zu verkleinern, da je nach Verhältnis von Rand- und Innenknoten ein hoher Anteil an Nullzeilen (Zeilen in denen der Ausdruck $0 = 0$ steht) entstehen kann.

Literatur

[1] Prof.Dr.rer.nat. Wolfgang Piepke (2018): Anwendung der Finite–Elemente–Methode