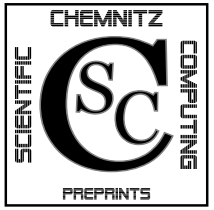
TECHNISCHE UNIVERSITÄT CHEMNITZ

Janine Glänzel

Kurzvorstellung der 3D-FEM Software SPC-PM3AdH-XX

CSC/09-03



Chemnitz Scientific Computing Preprints

Chemnitz Scientific Computing Preprints – ISSN 1864-0087

Impressum:

Chemnitz Scientific Computing Preprints — ISSN 1864-0087
(1995–2005: Preprintreihe des Chemnitzer SFB393)

Herausgeber:
Professuren für
Numerische und Angewandte Mathematik
an der Fakultät für Mathematik
der Technischen Universität Chemnitz

Postanschrift:
TU Chemnitz, Fakultät für Mathematik
09107 Chemnitz
Sitz:
Reichenhainer Str. 41, 09126 Chemnitz

http://www.tu-chemnitz.de/mathematik/csc/

Verzerrungen auf der Basis eines Substrukturkonzepts. Oktober 2008.

08-06 M. Meyer, J. Müller. Identification of mechanical strains by measurements of a deformed electrical potential field. November 2008.

08-07 M. Striebel, J. Rommes. Model order reduction of nonlinear systems: status, open issues, and applications. November 2008.

The complete list of CSC and SFB393 preprints is available via
`http://www.tu-chemnitz.de/mathematik/csc/`

TECHNISCHE UNIVERSITÄT CHEMNITZ

Chemnitz Scientific Computing Preprints

Janine Glänzel

Kurzvorstellung der 3D-FEM Software SPC-PM3AdH-XX

CSC/09-03

Zusammenfassung

In diesem Preprint wird die Weiterentwicklung der 3D-FEM Software SPC-PMAdH-XX kurz vorgestellt. Die Einleitung beschreibt schematisch den Ablauf des Programms und sechs Programmversionen. Weiterhin werden die allgemeinen Bedienungsanleitungen und die Funktionalität der einzelnen Versionen aufgeführt. Im letzten Abschnitt dem Anhang werden kurz das Standardfile und das Radiale Basisfunktionen Datenfile erklärt.

CSC/09-03 ISSN 1864-0087 **Januar 2009**

Inhaltsverzeichnis

Author's addresses:

Janine Glänzel
TU Chemnitz
Fakultät für Mathematik
D-09107 Chemnitz

http://www.tu-chemnitz.de/∼glj

Some titles in this CSC and the former SFB393 preprint series:

07-01 U. Baur, P. Benner. Gramian-Based Model Reduction for Data-Sparse Systems. February 2007.

07-02 A. Meyer. Grundgleichungen und adaptive Finite-Elemente-Simulation bei „Großen Deformationen“. Februar 2007.

07-03 P. Steinhorst. Rotationssymmetrie für piezoelektrische Probleme. Februar 2007.

07-04 S. Beuchler, T. Eibner, U. Langer. Primal and Dual Interface Concentrated Iterative Substructuring Methods. April 2007.

07-05 T. Hein, M. Meyer. Simultane Identifikation voneinander unabhängiger Materialparameter - numerische Studien. Juni 2007.

07-06 A. Bucher, U.-J. Görke, P. Steinhorst, R. Kreißig, A. Meyer. Ein Beitrag zur adaptiven gemischten Finite-Elemente-Formulierung der nahezu inkompressiblen Elastizität bei großen Verzerrungen. September 2007.

07-07 U.-J. Görke, A. Bucher, R. Kreißig Zur Numerik der inversen Aufgabe für gemischte (u/p) Formulierungen am Beispiel der nahezu inkompressiblen Elastizität bei großen Verzerrungen. October 2007.

07-08 A. Meyer, P. Steinhorst. Betrachtungen zur Spektraläquivalenz für das Schurkomplement im Bramble-Pasciak-CG bei piezoelektrischen Problemen. Oktober 2007.

07-09 T. Hein, M. Meyer. Identification of material parameters in linear elasticity - some numerical results. November 2007.

07-10 T. Hein. On solving implicitly defined inverse problems by SQP-approaches. December 2007.

08-01 P. Benner, M. Döhler, M. Pester, J. Saak. PLiCMR - Usage on CHiC. July 2008.

08-02 T. Eibner. A fast and efficient algorithm to compute BPX- and overlapping preconditioner for adaptive 3D-FEM. June 2008.

08-03 A. Meyer. Hierarchical Preconditioners and Adaptivity for Kirchhoff-Plates. September 2008.

08-04 U.-J. Görke, A. Bucher, R. Kreißig. Ein numerischer Vergleich alternativer Formulierungen des Materialmodells der anisotropen Elastoplastizität bei großen Verzerrungen. September 2008.

08-05 U.-J. Görke, R. Landgraf, R. Kreißig. Thermodynamisch konsistente Formulierung des gekoppelten Systems der Thermoelastoplastizität bei großen

Literatur

[1] S. Beuchler, A. Meyer, M. Pester, *SPC-PM3AdH v1.0-Programmer's Manual*, Preprint SFB393/01-08,TU-Chemnitz 2003.

[2] D. Lohse, *Ein Standardfile für 3D-Gebietsbeschreibungen*, Preprint SFB 393-98/11, TU-Chemnitz 1998.

[3] D. Lohse, *Ein Standard-File für 3D-Gebietsbeschreibungen-Datenbasis und Programmschnitstelle data_read.*, Preprint SFB/98-17,TU-Chemnitz 1998.

[4] A. Meyer, *Grundgleichungen und adaptive Finite-Elemente-Simulation bei "Großen Deformationen"*, Preprint CSC 07-02, TU-Chemnitz 2007.

[5] A. Meyer, P. Steinhorst, *Betrachtung zur Spektraläquvalenz für das Schukomplement im Bramble-Pasciak-CG bei piezoelektrischen Problemen*, Preprint CFC/07-08,TU-Chemnitz.

[6] A. Meyer, P. Steinhorst, *Überlegungen zur Parameterwahl im Bramble-Pasciak-CG für gemischte FEM*, Preprint SFB/05-07,TU-Chemnitz 2005.

[7] J.H.Bramble and J.E. Pasciak:, *A Preconditioning Technique for Indefinite Systems resulting from Mixed Approximations of Elliptic Problems. Math. Comput.*, 50(181):1-17,1988.

[8] A. Meyer, T. Steidten, *Improvements and Experiments on the Bramble-Pasciak Type CG for Mixed Problems in Elasticity.*, Preprint SFB393/01-13,TU-Chemnitz 2001.

[9] R. Unger, *Obstacle Description with Radial Basis Function for Contact Problems in Elasticity.*, Preprint CSC/09-01,TU-Chemnitz 2009.

1 Ausgangspunkt: Programm SPC-PM3AdH-XX

In diesem Abschnitt wird kurz auf das Programm SPC-PM3AdH eingegangen, welches im SFB 393 entwickelt wurde. Der Leistungsumfang des Finite-Elemente-Programms (FE) umfasst die adaptive Finite-Elemente-Lösung von Reaktions-Diffusions-Gleichungen und Elasitzitätsproblemen im dreidimensionalen Raum bei Hexaeder-Vernetzung. Um den Aufbau des adaptiven FE-Programms näher beschreiben zu können, wird die nachstehende Abbildungen 1 betrachtet. Die Abbildung zeigt die chronologische Arbeitsschrittfolge im Programmablauf. Sie beginnt mit dem Einlesen des Datenfiles. Im nächsten Schritt wird die Netzverfeinerung durchgeführt und danach folgt das Generieren der Elementsteifigkeitsmatizen und das Assemblieren der rechten Seite. Weiterführend wird ein Startvektor definiert, damit das erhaltene Gleichungssystem mit dem PCGM-Löser gelöst werden kann. Im letzten Schritt wird das Fehlerverhalten mit dem Fehlerschätzer berechnet. Dieser Algorithmus wird innerhalb der Verfeinerungsschleife abgearbeitet, bis das Abbruchkriterium erfüllt wird. Weiterführende Literatur ist unter [1] zu finden.

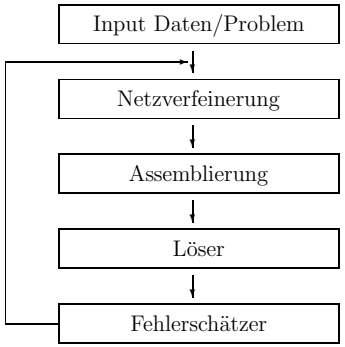


Abbildung 1: Ablauf der adaptiven Iteration

2 Erweiterungen und Fortentwicklung des FE-Programms

In diesem Kapitel werden im Wesentlichen vier Programmversionen des dreidimensionalen Programms PM3-AdH erläutert. Es gibt jeweils eine Einprozessor bzw. eine Mehrprozessorvariante für lineare Elastizitätsprobleme und für nichtli-

neare Deformationsprobleme „großer Deformationen“ und zwei Einprozessorvarianten für lineare „gemischte Probleme“. Folgende sechs Versionen sind vorhanden.

1. Programmversion für lineare Probleme als Einprozessorvariante PM3AdH
2. Programmversion für lineare Probleme als Mehrprozessorvariante PM3AdH
3. Programmversion für nichtlineare Deformationsprobleme bei großen Deformationen als Einprozessorvariante PM3AdH-NL
4. Programmversion für nichtlineare Deformationsprobleme bei großen Deformationen als Mehrprozessorvariante PM3AdH-NL
5. Programmversionen für lineare Probleme der Thermoelastizität als Einprozessorvariante PM3AdH-TEL
6. Programmversion für lineare Probleme der Piezoelektrizität als Einprozessorvariante PM3AdH-PEL

Lineare Probleme werden in diesem Fall definiert durch:

- Lösung von Reaktions-Diffusions-Gleichungen bei $NDof = 1$ (bei einem Freiheitsgrad, dass heißt es liegt eine skalare Funktion vor).
- Lösung von Elastiziätsproblemen bei $NDof = 3$ (bei drei Freiheitsgraden, dass heißt eine Vektorfunktion wird berechnet).
- Lösung von Mixed-Problemen bei $NDof = 4$ (bei vier Freiheitgraden, dass heißt eine Vektorfunktion wird berechnet ($NDof = 3$ Verschiebungfeld) und eine skalare Funktion ($NDof = 1$ Temperatur beziehungsweise elektrisches Potential))

2.1 Allgemeine Bedienungsanweisungen und Funktionalität der Programmversionen

Zunächst ist zu erwähnen, dass die sechs Versionen weitestgehend mit der Version für lineare Probleme (Einprozessorvariante) verlinkt sind. Das bedeutet bei Änderungen in der Originalversion werden alle Routinen, die in allen sechs Varianten gleich sind automatisch mit angepasst. Alle sechs Versionen sind nur unter LINUX lauffähig.

2.1.1 Einprozessorvariante für lineare Probleme-Grundbedienung

Für die erste Programmversion zur Berechnung linearer Probleme werden im Folgenden zunächst die allgemeinen Bedienungsanweisungen erklärt.


```
#
# and the rbf coefficients
#
   1  1.099784e-01
   2  1.989563e-01
   3 -4.629006e-02
   4 -9.802984e-03

    ...

 515 -1.770128e-16
 516  4.570523e-01
 517  1.926408e+00
```

N
1 $< p_1 >$ $< p_2 >$ $\cdots$ $< p_n >$

1 $< x_1 >$ $< y_1 >$ $< z_1 >$ $< f_1 >$
2 $< x_2 >$ $< y_2 >$ $< z_2 >$ $< f_2 >$
$\cdots$
N $< x_N >$ $< y_N >$ $< z_N >$ $< f_N >$
1 $< c_1 >$
2 $< c_2 >$
$\cdots$
N $< c_N >$
$N+1$ $< c_{N+1} >$
$N+2$ $< c_{N+2} >$
$N+3$ $< c_{N+3} >$
$N+4$ $< c_{N+4} >$

Zahlenbeispiel:

```
#
# Radial Basis Function -Obstacle - Examplefile
#
#  number of samplepoints
513
#
# rbf-type and 10 possible parameters p_1 ... p_10
#
# type 1 : phi(r) = r
#      2 : phi(r) = exp( -p_1 * r^2 )
#
1  1.123 2.123 3.123 4.123 5.123 6.123 7.123 8.123 9.123 10.123
#
#  now the samples

   1 -4.000000e+00  -4.000000e+00  6.904000e-01    0.000000e+00
   2 -4.000000e+00  -4.000000e+00 -1.309600e+00   -1.000000e+00
   3 -4.000000e+00  -4.000000e+00  2.690400e+00    1.000000e+00
   4 -4.000000e+00  -3.555556e+00  7.839820e-01    0.000000e+00
   5 -4.000000e+00  -3.555556e+00 -1.216018e+00   -1.000000e+00

   ...

 511  4.000000e+00   4.000000e+00 -8.090400e+00    0.000000e+00
 512  4.000000e+00   4.000000e+00 -1.009040e+01   -1.000000e+00
 513  4.000000e+00   4.000000e+00 -6.090400e+00    1.000000e+00
```


1. Der Programmaufruf erfolgt im entsprechenden Ordner über eine Shell mit dem Kommando *aquad.LINUXX*. Nach dem Aufruf läuft das Programm PM3-AdH an.

2. Nach dem Programmstart wird folgende Frage gestellt.
   "File-write for GNUPLOT? ([J]/N)"
   Diese Frage kann mit **j/[ENTER]** beantwortet werden, wenn eine grafische Darstellungen des geschätzten Fehlers mit GNUPLOT gewünscht ist. In diesem Fall entsteht das ASCII-File "fort.1" mit entsprechenden Daten.

3. Abfrage: **"Input file:"**
   Eingabe des Filenamen zum Einlesen des "Netz-Files" aus der Subdirectory ./mesh4. Dieses File enthält alle nötigen Informationen über das Start-FE-Netz sowie Randbedingungen (Materialdaten, etc.), vergleiche [2].
   Wählen zum Beispiel das Netzfile loch3D.std. Dieses Netzfile stellt ein quaderförmiges Objekt mit einer kreisrunden Aussparung in der Mitte dar. Am unterem Rand wird das Objekt mit einer Dirichlet-Randbedingung festgehalten. Und am oberen Ende des Objektes wird mit einer Neumann-Randbedingung gezogen.

4. Abfrage: **"Select element type (8, 20, 27): "**
   Bei diesem Aufruf wird die Anzahl der Knoten pro Element 8, 20 oder 27 Knoten festgelegt. Wählen für das Beispiel loch3D 8 Knotenelemente, dass heißt lineare Elemente. Nach diesem Schritt werden einige Daten zu den eingelesen Netzdaten, zum Beispiel der Name des Files, Anzahl Knoten, Kanten, Flächen, Elemente sowie Randbedingungen, verwendete Materialien und der Freiheitsgrad, aufgezeigt.

5. Bevor die Simulationsrechnung gestartet wird, können noch einzelne Parameter festgelegt werden. Diese werden über einen der Schlüsselbuchstaben angesteuert:
   "change parameters (?=help) i/e/v/d/m/s/r/T/z/? or:"
   Bedeutung der einzelnen Parameter:

   i - Iterationszahl (maximale Anzahl an PCG-Iterationen)

   e - Epsilon (Abbruch ε für den PCG)

   v - Vorkonditionierung (Art der Vorkonditionierung im PCG)

   d - Delta (Steuerungsgröße für Bramble-Pasciak-CG)

   g - Gamma (Steuerungsgröße für Bramble-Pasciak-CG)

   s - Eingabe 3 Werte für konstanten Startvektor

   r - first refinement defines coars mesh

   T - Dämpfungsfaktor für Newtonschleife

z - Zwischendruck-Steuerung

? - Help

6. Ausgabe der Simulationsrechnung als Tabelle (für $z = -1$) und der Graphik in einem Extrafenster. Für das betrachtete Beispiel loch3D ergibt sich folgendes.

NetFine # Elem	Assem: time [s]	PCGM It	PCGM time[s]	PCGM $< r, w >$	#Elems to ref.	est. Err.
14	0.001	2	0.000	9.7E-05	2	2.5E-01
168	0.010	17	0.051	5.7E-06	11	4.3E-02
1428	0.091	24	0.190	1.8E-06	80	2.1E-03
14448	1.629	35	5.911	5.8E-07	680	7.3E-05
79093	13.509	32	31.079	1.5E-07	5750	4.7E-06

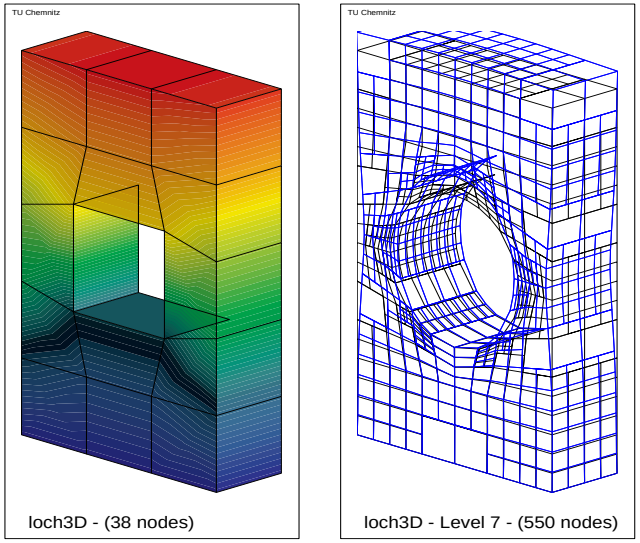


Abbildung 2: Grundnetz und 7. Verfeinerungsschritt des Beispiels loch3D

7. Steuerung der Adaptivität:

Enter - Durch Eingabe der Taste "Enter" am Ende einer Iteration wird die nächste Iteration mit entsprechend feinerem Netz berechnet. (vgl. Abb. 1)

0 - Bei Eingabe der Taste "0" wird mit der aktuellen Netzverfeinerung

- #MATERIAL: <count>
  <Mat.nr.> <Mat.parameteranz.> <Mat.parameter>
  - Thermoelastisches Material:
    1 <Mat.parameter> E-Modul ν κ α $\vec{f}$ c
  - Piezoelektrisches Material:
    1 <Mat.parameter> Materialmatrizen (2.1.7) $\vec{f}$ c
  - Bedeutung der Parameter
    ν Querkontraktionszahl (Poisson-Zahl)
    κ Diffusionskoeffizient
    α Ausdehnungskoeffizient
    $\vec{f}$ Krafttvektor
    c Quelle
- #FACE_GEO: <count>
  <fg-name> <fg-typ> <fg-count> <k-Parameter>
- #END_OF_DATA:

3.2 Radiale Basisfunktionen-Datenfile *.rbf

Das RBF-Datenfile wird wie folgt aufgebaut. Zuerst steht die Anzahl der Samplepunkte.
<count>
Dann folgt der Radiale Basisfunktions-Typ der mit der Typnummer und zehn möglichen Parametern definiert wird.

<Typ-name> $< p_1 >$ $< p_2 >$ $\cdots$ $< p_n >$

Weiterhin werden immer zeilenweise die Nummer, Koordinaten und Funktionswert der Samples beschrieben,

<Nummer> <Koordinaten> <Funktionswert>

gefolgt von den RBF-Koeffizienten

<Nummer> <Koeffizienten> .

Die allgemeine Struktur eines RBF-Datenfiles mit N Samples entspricht nachstehender Gestalt.

- Erläuterung der eingeführten Schlüsselzahlen:
  <1> Dirichlet-Randbedingung
  (z. B. eine vorgegebene feste Verschiebung)
  <200> Rutschrand
  (Vorgabe des Normalenvektors $[a_1, a_2, a_3]^T$ im Netzdatenfile)
  <210> Ebene als Hinderniss
  (Vorgabe eines Punktes $[a_1, a_2, a_3]^T$ und des Normalenvektor $[a_4, a_5, a_6]$ der Ebene)
  <230> krummlinige Hindernisse, Hindernisrand: $F(x, y, z) = 0$
  (z. B. Torus, Ballindenter,...)

  – a_{12} definiert Hindernisklassen:
    <100> F ist eine Quadrik
    $F(x) = a_1x^2 + 2a_2xy + a_3y^2 + 2a_4xz + 2a_5yz + a_6z^2$
    <200> F ist eine Kugel mit Radius a_4 und Mittelpunkt (a_1, a_2, a_3)
    <300> F ist eine Torus parallel zur xy-Ebene mit a_1 (major radius), a_2 (minor radius) und a_3 Verschiebung entlang der z-Achse
    <400> F ist mittels Radialer Basisfunktion beschrieben

- Bsp: Torus als Hindernis

Flächennr.	230	2.5	0.0	0.0	0.0
	230	0.5	0.0	0.0	0.0
	230	-4.5	0.0	0.0	300

- Bsp: Ballindenter ($a_{12} = 200$)

$$F(x, y, z) := (x - a_1)^2 + (y - a_2)^2 + (z - a_3)^2 - a_4^2$$

- Bsp: Radiale Basisfunktion als Hindernisbeschreibung [9]

Flächennr.	230	0.0	0.0	0.0	0.0
	230	0.0	0.0	0.0	0.0
	230	0.0	0.0	0.0	400.0

  Die Parameter $a_1 \cdots a_{12}$ werden dabei ignoriert, die Daten zur Definition der Radialen Basisfunktion (RBF) werden aus einem Textfile mit gleichem Namen wie das Netzfile aber mit der Endung .rbf eingelesen, welches im Abschnitt 3.2 beschrieben wird.

- #NEUMANN: <count>
  Der Aufbau der Neumann-Randbedingungen ist analog zu den Dirichlet-Randbedingungen.

nochmal gerechnet. Das heißt es erfolgt keine Netzverfeinerung, die aktuelle Lösung wird genauer.

t - Die Taste "t" bewirkt, dass das Netz gleichmäßig verfeinert wird. Wird auch Totalverfeinerung genannt.

a - Bei Eingabe der Taste "a" berechnet das Programm automatisch die Iterationen bis ungefähr 10000 Elemente.

n - Beendigung der aktuellen Simulationsrechnung. Programm spingt zum Schritt 3. und es kann ein neues Netzdatenfile eingegeben werden.

e - Beendigung des Programms

2.1.2 Einprozessorvariante für lineare Probleme-Leistungsumfang

Eine erste Erweiterung des Programms SPC-PM3AdH beinhaltet, dass für lineare Probleme auch Objekte mit Rutsch- und Kontakt-Randbedingungen simuliert werden können. In Abbildung 3 ist ein Beispiel für ein Objekt mit Rutschrand (links) und ein Objekt mit Kontaktrand (rechts) dargestellt. Das Beispiel mit der Rutschrandbedingung wurde so konstruiert, dass die geneigte Fläche entlang einer gedachten Ebene rutscht. Von oben wird eine Kraft in z-Richtung aufgegeben. Das Beispiel rechts im Bild zeigt eine Kugel, die von oben in drei gedachte Kontaktebenen fällt. Im Bild wird die Kugel aus der Sicht der Raumecke gezeigt. An den stärksten Netzverfeinerungen sind die Kontaktstellen der Kugel mit den Ebenen zu erkennen.

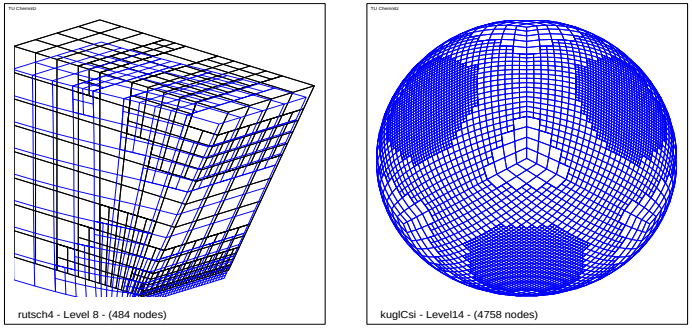


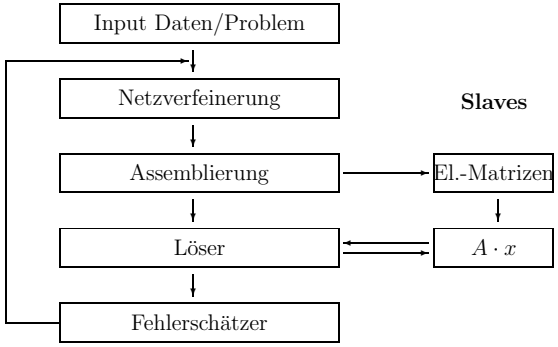
Abbildung 3: Beispiele für Rutsch- und Kontakt-Randbedingung

Die ensprechenden Datenfiles sind in des Subdirectory ./mesh4 hinterlegt. Bei einigen Netzdatenfiles mit Kontaktflächen ist es notwendig einen Nichtnull- Start-

vektor einzugeben. Und zwar dann, wenn das Datenfile zu wenige Dirichlet-Randbedingungen enthält, so dass die Struktur erst mit der Kontaktbedingung zum regulären Problem wird. Der Startvektor muss dann eine feste Starrkörperverschiebung sein, die die Kontaktbedingungen an wenigen Stellen verletzt. Hiermit wird schon im ersten Durchlauf eine gültige Lösung möglich (nichtsinguläres Gesamtproblem). Dazu muss im Schritt 5. der Parameter **"s"** eingeben und sinnvoll belegt werden. Außerdem muss teilweise der Grobgitterlöser ausgeschaltet werden, wenn eine singuläre Grobgittermatrix auftritt. Dazu ist der Parameter **"v"** auf 4 zu setzen.

2.1.3 Mehrprozessorvariante für lineare Probleme

Zum Starten der Mehrprozessorvariante muss sich auf einen Parallelrechner eingeloggt werden, zum Beispiel der CHiC-Rechner der TU-Chemnitz.
Die Mehrprozessorvariante für lineare Probleme wird analog nach den Schritten 1. bis 7. gesteuert. Es können die gleichen Simulationsrechnungen wie die in der Einprozessorvariante erfolgen. Der Hauptunterschied zwischen der Ein- bzw. Mehrprozessorvariante besteht darin, dass das Generieren der Elementmatrizen und die Matrixmultiplikation im PCGM von den Parallelrechnern (Slaves) übernommen wird. Somit wird an Rechenzeit gespart. Und insbesondere wird die Speicherplatzbeschränkung weitestgehend aufgehoben, indem auf dem Master-Prozessor zwar weiterhin die Daten für die adaptive Rechnung gespeichert bleiben, aber die große Menge der Daten für alle Elementmatrizen sich auf die „Slaves" verteilt. Die nachfolgende Abbildung zeigt im Groben den Programmablaufplan für das Farming Konzept und die eingebauten Änderungen im Vergleich zur Einprozessorvariante.



3 Anhang

3.1 Standardfile *.std

Ein Netzdatenfile *.std aus der Subdirectory ./mesh4 hat folgenden allgemeinen Aufbau. Ausführlich beschrieben ist die Definition des Standard-Files in [2] und dazu erweiterte Informationen in [3].

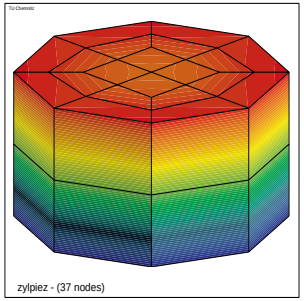
- #VERSION: <Versionserkennung>
- #PROGRAM: <Programmname>
- #DATE: <Datum>
- #DESCRIPTION: <Beschreibung>
- #USER: <Login>
- #DIMENSION: <Dimension>
- #EQN_TYPE: <Gleichungstyp>
- #DEG_OF_FREE: <NDof>
- #MAX_MAT_DATA: <count>
- #HEADER: <count>
- #VERTEX: <count>
  <Knotennr.> <x> <y> <z> - Koordinaten
- #EDGE: <count>
  <Kantennr.> <Kantentyp> <Start-> und <Endknoten>
- #FACE: <count>
  <Flächennr.> <Flächentyp> <Kantenanz.> <Kantennummern>
- #SOLID: <count>
  <Elementnr.> <Materialtyp> <Flächenanz.> <Flächennummern>

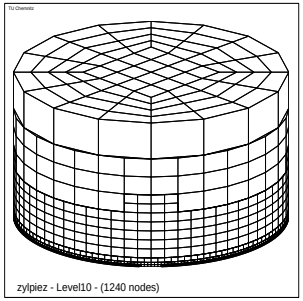
Die allgemeine Darstellung der Dirichlet-Randbedingung wird im folgendem beschrieben. Für lineare Probleme mit Rutsch- und Kontakt-Randbedingungen wurden Schlüsselzahlen (1, 200, 210, 230) im Netzdatenfile zur Erkennung von Rutsch- und Kontakträndern eingeführt.

- # DIRICHLET: <count>
  <Flächennr.>[1] <Schlüsselzahl> $< a_1 >$ $< a_4 >$ $< a_7 >$ $< a_{10} >$
  <Schlüsselzahl> $< a_2 >$ $< a_5 >$ $< a_8 >$ $< a_{11} >$
  <Schlüsselzahl> $< a_3 >$ $< a_6 >$ $< a_9 >$ $< a_{12} >$

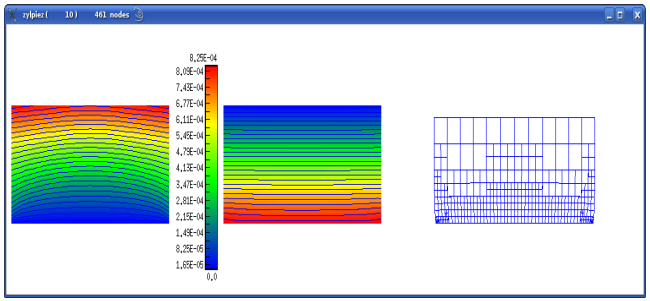
[1] gilt für $NDof = 3$; bei $NDof = 4$: vierte Schlüsselzahl und vier Parameter hinzunehmen

Nachfolgend sind noch drei Abbildungen dargestellt. Im Bild links ist der Zylinder nach der ersten Netzverfeinerung zu sehen und daneben die zu erwartende adaptive Netzverfeinerung nachfolgender Verfeinerungsschritte.





Das dritte Bild zeigt den Vollzylinder im Längsschnitt. Das linke Bild zeigt $|u|$ mit dem Verlauf der Isolinien. Im mittleren Bild ist P mit den nach oben hin fast parallel verlaufenden Isolinen zu erkennen. Das rechte Bild zeigt nochmal im Querschnitt die adaptive Vernetzung. Deutlich zu erkennen ist die verstärkte Vernetzung entlang des unteren Kreisrings entlang der Singularität.



2.1.4 Einprozessorvariante für nichtlineare Probleme bei großen Deformationen

Die nichtlineare Version des Programms PM3-AdH-NL simuliert große Deformationen von 3D-Strukturen unter Zug- und Druckeinwirkungen. Das Programm wurde dahingehend modifiziert, dass zur Berechnung von Nichtlinearitäten eine Newtonschleife hinzugefügt wurde. Weiter Hintergrundsinformationen finden sich in [4].
Die Programmbedienung erfolgt analog zu Schritt 1. bis 7. aus Abschnitt 2.1.1. Im Schritt 5. gibt es eine weitere Neuerung: Zur Berechnung von Nichtlinearitäten ist noch ein weiterer Steuerparameter **"T"** vorhanden. Dieser Parameter legt den Dämpfungsfaktor $\Delta t = \frac{1}{M}$ fest.
"give M for defining timestep Delta_T = 1/M"
"neues M ="
An dieser Stelle wird M eingegeben. Zur Darstellung der Funktionalität wird das Testbeispiel loch3LD betrachtet. Dieses Netzfile hat das selbe Grundnetz wie das Beispiel loch3D aus Abschnitt 2.1.1. Es liegen aber andere Randbedingungen und ein anderes Materialgesetz vor. Für dieses Beispiel wird $M = 10$ gewählt. Das heißt der Dämfungsfaktor ist $\Delta t = \frac{1}{10}$.
Beim Durchlauf der Simulationsrechnung werden die Newton-Iterationen der Nichtlinearität in der Rechnungstabelle zur Kontrolle und Nachvollziehung mit ausgeben. Dabei können die Größen $t - Damp$ (Aktueller Dämpfungfaktor der sich nach jeder „konvergierenden" Newton-Iteration um Δt erhöht.), $\frac{dU}{U}$ und $detF$ (Determinante des Deformationsgradienten, dieser Kontrollwert darf nicht kleiner Null werden.) abgelesen werden. In der untenstehenden Tabelle für das betrachte Beispiel loch3LD kann dies (auszugsweise) nachvollzogen werden.

NetFine:	Assem:	PCGM		NewtonNili:			est.Err.
#Elem	times[s]	It	$< r, w >$	t-damp	$\frac{dU}{U}$	Det F	
14	0.002	2	6.4E+04	0.10	INF	1.0	1.7E-01
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
154	0.019	13	1.3E+03	0.10	6.E-03	1.0	1.5E-02
882	0.142	14	1.4E+01	0.10	7.E-05	1.0	5.0E-04
2688	0.329	13	4.6E+05	0.20	1.E+00	1.0	2.4E-04
2688	0.362	19	1.0E+02	0.20	2.E-04	0.9	1.7E-04
4074	0.496	19	6.0E+01	0.20	1.E-05	0.9	7.9E-05
6314	0.769	14	6.8E+05	0.30	2.E-01	0.9	5.8E-05
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
82082	20.744	20	1.7E+06	0.90	1.E-02	0.5	1.3E-06
82082	21.468	55	5.6E+02	0.90	6.E-05	0.3	8.5E-07
115703	14.946	22	1.9E+06	1.00	1.E-02	0.3	7.6E-07

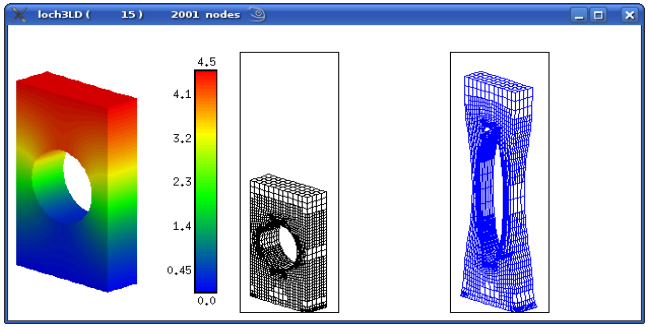


Abbildung 4: Vergleich unverformten mit verformten adaptiven Netz nach dem 15. Verfeinerungsschritt

2.1.5 Mehrprozessorvariante für nichtlineare Probleme bei großen Deformationen

Die Mehrprozessorvariante bei großen Deformationen simuliert analoge Berechnungen wie die Einprozessorvariante. Die Bedienung des Programms erfolgt wie in den Schritten 1. bis 7. aus Abschnitt 2.1.1. Der Schritt 5. hat ebenfalls die Erweiterung für den Dämpfungsfaktor der Newtonschleife. Ansonsten besitzt die Mehrprozessorvariante die selben Unterschiede zur Einprozessorvariante, wie in Abschnitt 2.1.1 für lineare Elastizitätsprobleme kurz beschrieben wurde. Der Programmaufruf kann erst nach dem Einloggen auf einen Parallelrechner erfolgen. Mit der Mehrprozessorvariante können größere adaptive Verfeinerungsschritte (über 100000 Elemente) erreicht werden. Aufgrund der Verteilung der Elementmatrizen auf die Parallelrechner (Slaves) wird ein höherer Speicherumfang erreicht und kompliziertere Rechnungen können durchgeführt werden.

Für das nachfolgende Beispiel werden folgende Materialeigenschaften ausgewählt:

$$c_{11} = 1.39 \cdot 10^{11} N/m^2, \quad c_{33} = 1.13 \cdot 10^{11} N/m^2,$$
$$c_{12} = 7.78 \cdot 10^{10} N/m^2, \quad c_{13} = 7.43 \cdot 10^{10} N/m^2,$$
$$c_{44} = 7.78 \cdot 10^{10} N/m^2$$

$$\kappa_{11} = 6.0 \cdot 10^{-9} C/(Vm), \quad \kappa_{33} = 5.470 \cdot 10^{-9} C/(Vm),$$

$$e_{15} = 13,44 C/m^2, \quad e_{31} = -6.98 C/m^2,$$
$$e_{33} = 13,84 C/m^2$$

Bemerkung: Im Netzdatenfile *.std (in ./mesh4) muss dazu die Headerzeile #MAX_MAT_DATA: 14 eingefügt werden (siehe Anhang 3.1), um die 14 Materialdaten für Piezoelektrizität einbauen zu können.

Die Erweiterungen das Programms werden am Beispiel zylpiez.std verständlicher dargestellt. Das ist ein Vollzylinder der am Boden fixiert wird und das oben beschriebene piezoelektrische Material besitzt. Die nachstehende Tabelle zeigt auszugsweise die Ergebnisse pro Schleifendurchlauf.

NetFine:	Assem:	PCGM			#ELems	est.Err.	
#Elem	times[s]	It	time[s]	$< r, w >$	to ref.		
5	0.003	6	0.004	1.9E+00	5	2.4E-01	t
40	0.006	18	0.003	7.4E-01	8	3.2E-02	t
320	0.086	22	0.074	8.4E-01	16	6.9E-03	
432	0.029	30	0.148	2.3E-01	48	2.3E-03	0
432	0.005	70	0.321	1.0E-05	48	2.3E-03	0
432	0.000	69	0.326	9.7E-10	48	2.3E-03	0
432	0.000	66	0.306	7.1E-13	48	2.3E-03	a
768	0.080	23	0.206	1.4E-01	36	9.9E-04	
⋮	⋮	⋮	⋮	⋮	⋮	⋮	
6480	0.420	24	2.012	4.6E-02	468	3.4E-05	
9812	0.847	26	3.404	4.5E-02	8	4.6E-05	

2.1.7 Einprozessorvariante für lineare Probleme der Piezoelektrizität

Um lineare piezoelektrische Probleme mit vier Freiheitsgraden zu simulieren, kann diese Programmversion genutzt werden. Es wird folgendes Gleichungssystem gelöst:

$$\begin{aligned} c(u,v) + b(\varphi,v) &= F(v) \qquad \forall v \in V \\ b(\psi,u) - k(\varphi,\psi) &= -G(\psi) \quad \forall \psi \in Q. \end{aligned}$$

Wobei u das Verschiebungsfeld und φ das elektrische Potential ist. In Matrizenschreibweise hat das Gleichungssystem folgende Gestalt.

$$\begin{pmatrix} C & B \\ B^T & -K \end{pmatrix} \begin{pmatrix} \underline{U} \\ \underline{P} \end{pmatrix} = \begin{pmatrix} \vec{f} \\ \vec{g} \end{pmatrix}$$

Die Matrizen C und K sind symmetrisch und positiv definit und B ist eine nicht symmetrische rechteckige Matrix.

Zur Lösung solcher Aufgaben mussten einige Änderungen im Hauptprogramm, in der Elementroutine, in der Materialeinleseroutine und dem PPCGM-Löser vorgenommen werden. Für weitere mathematische Informationen kann in [5] nachgelesen werden. Konkret bedeutet das, dass im Hauptprogramm zum Lösen des Gleichungssystem nicht mehr der PPCGM sondern der MixCGM [6, 7, 8] aufgerufen wird. Das heißt es wird der Löser dahingehend modifiziert, dass das Verschiebungsfeld $\vec{u} = (u_1, u_2, u_3)$ und das elektrische Potential φ in einem Gleichungssystem gelöst werden können. Zu dem werden in der Elementroutine die entsprechenden Matrizen C, B, K gebaut. Die zugehörigen Materialmatrizen besitzen folgende Gestalt:

$$\underline{\underline{C}} = \begin{pmatrix} c_{11} & c_{12} & c_{13} & 0 & 0 & 0 \\ c_{12} & c_{11} & c_{33} & 0 & 0 & 0 \\ c_{13} & c_{13} & c_{33} & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{c_{11}-c_{12}}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 & c_{44} & 0 \\ 0 & 0 & 0 & 0 & 0 & c_{44} \end{pmatrix}$$

$$\underline{\underline{B}} = \begin{pmatrix} 0 & 0 & e_{13} \\ 0 & 0 & e_{13} \\ 0 & 0 & e_{33} \\ 0 & 0 & 0 \\ 0 & e_{61} & 0 \\ e_{61} & 0 & 0 \end{pmatrix} \quad \underline{\underline{K}} = \begin{pmatrix} \kappa_{11} & 0 & 0 \\ 0 & \kappa_{11} & 0 \\ 0 & 0 & \kappa_{33} \end{pmatrix}$$

2.1.6 Einprozessorvariante für lineare Probleme der Thermoelastizität

Mit dieser Programmversion können lineare Verformungen thermoelastischer Probleme berechnet werden. Deren schwache Formulierung wie folgt aussieht:

$$\begin{aligned} a(u,v) - b(T,v) &= <\vec{f}, v> \quad \forall v \in V \\ k(T,\psi) &= <g, \psi> \quad \forall \psi \in Q \end{aligned}$$

Hierbei entsteht die Gleichung aus dem üblichen Kräftegleichgewicht

$$\int_\Omega \sigma(u,T) : \varepsilon(v) d\Omega = <\vec{f}, v>,$$

mit dem temperaturabhängigen Spannungstensor

$$\sigma(u,T) = C : \varepsilon(u) - (\alpha T) I$$

(C wie in linearer Elastizität, α ist der Ausdehnungkoeffizient).
Die zweite Gleichung entspricht der üblichen stationären-Wärmeleitgleichung

$$k(T,\psi) = \int_\Omega \kappa \nabla T \cdot \nabla \psi d\Omega.$$

Dazu mussten im Programm an mehreren Stellen (Hauptprogamm, Elementroutine, Materialeinleseroutine und Modifizierung des PPGCM-Lösers) Änderungen vorgenommen werden. Zunächst wurde im Programm grundlegend die Anzahl von drei auf vier Freiheitsgrade erhöht. Da nicht nur das Verschiebungsfeld u_x, u_y, u_z betrachtet werden, sondern zusätzlich noch ein Freiheitsgrad für die Temperaturausbreitung T. Die Elementroutine baut nach der Modifikation folgendes Gleichungssystem.

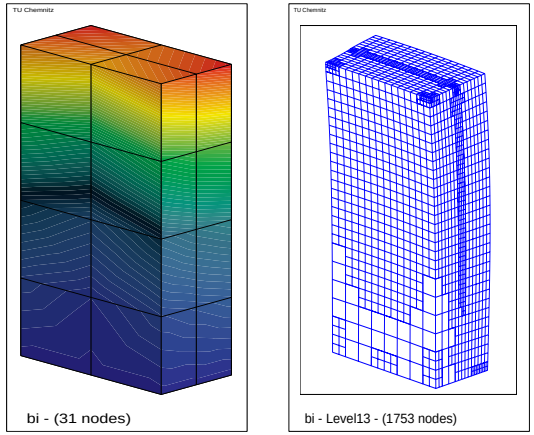
$$\begin{pmatrix} C & -B \\ 0 & K \end{pmatrix} \begin{pmatrix} \underline{U} \\ \underline{T} \end{pmatrix} = \begin{pmatrix} \underline{b}_1 \\ \underline{b}_2 \end{pmatrix}$$

In der Liste der Materialien werden folgende notwendigen Materialien (κ Diffusionskoeffizient, α Ausdehnungskoeffizient, $\vec{f}$ Kraftvektor und c eine Wärmequelle) mit aufgenommen. Der Löser PPCGM wird zweimal im Hauptprogramm aufgerufen, für drei Freiheitsgrade und für einen Freiheitsgrad. Das heißt das Programm löst zuerst die zweite Gleichung $K\underline{T} = \underline{b}_2$ dann für die erste $C\underline{U} = \underline{b}_1 + B\underline{T}$.

Am konkreten Beispiel eines Bimetallstreifens (Subdirectory /mesh4/bi.std) werden die Neuerungen näher erläutert. Das Bauteil besteht aus zwei verschiedenen Materialien (z. B. Metalle), daher existiert ein Materialsprung. Am unteren Rand (Boden) sind Dirichlet-Randbedingungen festgelegt. Aus der Tabelle kann entnommen werden, dass für thermoelastische Probleme zwei Gleichungssysteme gelöst werden. Siehe dazu Spalte PCGM. Die erste Lösung pro Verfeinerungsdurchlauf entspricht dem Gleichungssystem für $NDof = 1$ (Temperatur) und die zweite Lösung dem Gleichungssystem für $NDof = 3$ (Verschiebungsfeld).

NetFine: #Elem	Assem: times[s]	PCGM It	PCGM time[s]	PCGM $< r, w >$	# Elems to ref.	est.Err.
16	0.005	8	0.000	4.9E-01		
		13	0.000	1.2E+01	8	9.8E-02
128	0.014	13	0.000	1.5E-01		
		49	0.023	1.1E-01	34	8.7E-03
128	0.000	15	0.000	6.6E-06		
		12	0.004	1.2E+01	26	7.9E-03
128	0.000	15	0.000	9.6E-11		
		47	0.023	6.1E-02	34	8.7E-03
366	0.030	12	0.005	8.8E-03		
		12	0.015	1.7E+01	46	2.9E-03
702	0.040	13	0.010	1.7E-02		
		40	0.072	8.4E-02	22	1.8E-03
856	0.022	21	0.025	7.4E-03		
		45	0.101	2.0E-01	36	9.8E-04
⋮	⋮	⋮	⋮	⋮	⋮	⋮
6218	0.226	10	0.082	4.9E-03		
		31	0.562	4.7E-02	61	8.2E-05
6841	0.056	28	0.233	1.7E-03		
		61	1.166	1.5E-03	207	4.1E-05

Die drei Abbildungen zeigen die Verformung des Bimetallstreifens unter Einwirkung der Temperatur T. Zusätzlich treten durch die Krümmung an den oberen linken Ecken Kontaktflächen mit einer Ebene auf. Gut zu erkennen ist der Materialsprung durch die verstärkte adaptive Vernetzung zwischen den beiden Streifen.



An dieser Stelle ist das Verschiebungsfeld u_1, u_2, u_3, das Temperaturfeld T und die Vernetzung nach dem dreizehnten Verfeinerungsdurchlauf zum Vergleich in einer Abbildung dargestellt.