

Inhaltsverzeichnis

1	Einleitung	1
1.1	Grundsätzliches	1
1.2	Generierungsmöglichkeiten	1
1.3	Aufruf des Programms	2
1.4	Gewährleistung	3
2	Elementgenerierung	5
2.1	*ELTYPE	5
2.2	*COORDINATES	5
2.3	*MAPE2D	6
2.4	*MESHVL	7
2.5	*BLFMESH1	9
2.6	*BLFMESH2	11
2.7	*BLFMESH3	12
2.8	*BLFMESH4	13
2.9	*IPLMESH	13
2.10	*CRACKTIP1	14
2.11	*RIGMESH	14
2.12	*LEFMESH	15
2.13	*UPSMESH	15
2.14	*DWNMESH	15
2.15	*CGROWTH1	15
2.16	*CGROWTH2	16
2.17	*CGROWTH4	16
2.18	*HBLF MESH	17

2.19	*FBLF MESH	17
2.20	*CRACK2	18
2.21	*CNV2TRI	18
2.22	*CNV2TET	18
2.23	*MESH 9 NODE	19
2.24	*COHESIV1	19
2.25	*COHESIV3	20
2.26	*CZDS	20
2.27	*CZDOMAIN	21
2.28	*MESH3D	21
2.29	*SIDE GROOVE	22
2.30	*EXT_3D4	22
2.31	*CFRONT	23
2.32	*FLIP	23
2.33	*COPY	24
2.34	*SYSMESH	24
2.35	*RCAEDS	24
3	weitere Eingabemöglichkeiten zur Netzgenerierung	27
3.1	*TRANSLATE	27
3.2	*ROTATE	27
3.3	*SCALE	28
3.4	*COCOORD	28
3.5	*REORDER	28
3.6	*DELSURF	29
3.7	*MESH2	29
3.8	*MESHOUT	29
3.9	*TOLERANCE	29
3.10	*MULTIMAT	29
3.11	*RENEW ELSET	30
4	Definition von Element- und Knotensets	31
4.1	*ENSET	31

4.2	*SET DEFINE	33
4.2.1	Definition von Elementsets	33
4.2.2	Definition von Knotensets	33
4.2.3	Knotenset zur J -Integralberechnung	34
5	Materialparameter	35
5.1	Elastisches Material	35
5.2	plastisches Material mit exponentieller Verfestigung	35
5.3	benutzerdefinierte Materialien	36
5.4	Deformationsplastizität	37
5.5	Plastizität mit vorgegebener Spannungs-Dehnungskurve	37
5.6	Kohäsivzonenmodell	38
5.7	Zuordnung von Elementgruppen zu Materialien	38
5.8	Eingabe von beliebigen Eingabekarten	39
6	Randbedingungen	41
6.1	*BOUNDARY	41
6.2	*BOUND1	41
6.3	*MPCSUB	42
6.4	*EQUATD	42
6.5	*EQUATQ	43
6.6	*EQUATS	43
6.7	*PERIOD	44
7	Lastschrittdaten	47
7.1	Belastungsinformation	47
7.2	Ausgabe (.dat-Datei)	47
7.3	Ergebnisausgabe (.fil-Datei)	48
7.4	Bestimmung des J -Integrals	48
7.5	Direkte Ausgabe von Steuerkarten in die Input-Datei	48
7.6	Lastschrittdefinition	49
7.7	Parameter für die einzelnen Lastschritte	49

8 weitere Befehle	51
8.1 *DRAWMESH	51
8.2 *DRAW FREE EDGE	51
8.3 *MODE COMMAND	51
8.4 *END POST	52
 Schlüsselwörter	 53
 Neue Features in Version 1.1	 55

Kapitel 1

Einleitung

FEMESH ist ein batch-orientierter Preprozessor für das FE-Programm ABAQUS. Der ursprüngliche Einsatzbereich war ebenfalls der FE-Solver FECGS, der jedoch nicht mehr unterstützt wird. Zur Zeit existieren Versionen für IBM R/S 6000 Workstations unter dem Betriebssystem AIX 4.3 sowie für DEC alpha XP1000 Workstations unter Tru64 UNIX V5.

1.1 Grundsätzliches

Die Eingabedatei ist in zwei Teile aufgespalten: zunächst erfolgt die Definition des Modells, danach die Eingabe der Lasten, Randbedingungen und Analyseoptionen, zusammengefaßt unter dem Begriff *history information*. Es ist wichtig, daß zunächst das gesamte Modell vorher definiert wird, da beide Teile sequentiell abgearbeitet werden, d.h. nach der Eingabe der *history information* ist eine weitere Modelleingabe nicht mehr möglich!

Die Eingabe wird über Schlüsselwörter gesteuert. Diese beginnen in der ersten Spalte mit einem * (Asterisk), gefolgt vom Schlüsselwort ohne Leerzeichen. Schlüsselwörter können weitere Eingaben oder Optionen nach sich ziehen.

Die Eingabe kann mit großen oder kleinen Buchstaben erfolgen, es erfolgt programmintern eine Umwandlung in Großbuchstaben. (Diese Option ist noch nicht durchgängig implementiert, so daß die Eingabe von Kommandos mit großen Buchstaben sicherer ist).

An (fast) allen Stellen können Kommentarzeilen eingegeben werden. Diese sind durch ** in den ersten beiden Spalten gekennzeichnet.

1.2 Generierungsmöglichkeiten

Folgende ABAQUS-Funktionen werden unterstützt:

- Knoteneingabe: 2D und 3D, in kartesischen und in Polarkoordinaten
- Elementtypen: 2D und 3D-Solids, Vierecks-, Dreiecks-, Quader- und Tetraederelemente

- Vernetzung mit Netzverfeinerungen und Biasing, 2D und 3D, spezielle Kommandos zur Generierung von Rißspitzennetzen
- Definition von Element- und Knotensets
- Lasten: verteilte und konzentrierte Kräfte, Verschiebungen, K-Felder (mit T-Spannungen)
- Umfangreiche Funktionen zur Generierung von Rißfortschrittsnetzen mit dem Kohäsivzonenmodell (Interfaceelemente) und Boundary Layer Formulation (BLF)
- Randbedingungen: Fesselung von Freiheitsgraden, Zwangsbedingungen, periodische Randbedingungen
- Materialien: Elastisch, Plastisch, benutzerdefinierte Materialmodelle, Parametereingabe für benutzerdefinierte Elemente, Deformationsplastizität. Die Materialdaten für ein elastisch-plastisches Modell können auch von Datei gelesen werden.
- Analyseoptionen: linear, MNO, geometrisch nichtlinear, RIKS, mehrere Zeitschritte, RESTART
- Außerdem kann das Netz sowie eine Darstellung der freien Kanten (nur 2D) auf dem Bildschirm ausgegeben werden. Das Drahtmodell des Netzes kann auch als Postscript-Datei gespeichert werden.
- Kommandos, die nicht unterstützt werden, können im ABAQUS Format eingegeben werden. Diese werden dann direkt übertragen.
- benutzerdefinierte Grenzflächenelemente (Kohäsivelemente) werden für die USER ELEMENT unterstützt (siehe *MATERIAL, *MESHOUT sowie verschiedene *COHESIVE-Befehle)

1.3 Aufruf des Programms

Das Programm wird durch den Befehl

```
$> mesh.x [<problemname>]
```

aufgerufen. `problemname` bezeichnet die Eingabedatei mit den zur Generierung eines ABAQUS-Eingabefiles notwendigen Befehle. Die Eingabedatei soll die Endung `.1` haben. Die Endung selbst muß jedoch nicht angegeben werden. Wird kein Dateiname angegeben, so fragt das Programm nach einer Eingabedatei, ebenso wenn die angegebene Datei nicht existiert.

Die Ausgabedatei, d.h. die Eingabedatei für das Programmsystem ABAQUS, erhält die dort verwendete Endung `.inp`. Außerdem werden zwei weitere Dateien angelegt: Die Datei `problemname.inf` (Informationsdatei) sowie `problemname.jnl` (Journal-File). In der Informationsdatei stehen die ausgeführten Befehle. Zusätzlich sind hier weitere Informationen wie die Anzahl der generierten Knoten und Elemente enthalten¹.

Im Journal-File werden die durchlaufenen Unterprogramme dokumentiert. Diese Datei ist daher für den Anwender von untergeordneter Bedeutung.

¹Diese Information erscheint nach Beendigung des Programms auch auf dem Bildschirm

1.4 Gewährleistung

Das Programm FEMESH wird seit mehreren Jahren bei der GKSS als Werkzeug zur Generierung von Bruchmechaniknetzen von der numerischen Abteilung verwendet. Die Programmierung stammt ursprünglich von Dr. G. Lin, der bei der GKSS vorrangig mit der Weiterentwicklung des Kohäsivzonenmodells beschäftigt war. Nachdem Herr Lin die GKSS verlassen hat, wurde die Wartung des Programms von I. Scheider übernommen. Da damals noch kein Handbuch für das Programm existierte, basiert das vorhandene Wissen auf den überlassenen Eingabedateien und dem Quellcode des Programms. Da FEMESH jedoch ursprünglich nur für den Eigenbedarf bestimmt war, existierten auch im Quelltext nur wenige Kommentarzeilen, sodaß das Verständnis einiger Funktionen evtl. für immer verborgen bleibt.

Die Stabilität des Programms ist nicht besonders ausgeprägt. Bei falschen Eingaben reagiert FEMESH teilweise immer noch mit einem Programmabsturz anstatt mit einer Fehlermeldung. Der verantwortliche Autor des Handbuch ist jedoch bemüht, dieses Verhalten zu korrigieren.

In keiner Weise kann aus den genannten Gründen eine Gewährleistung für das Programm FEMESH übernommen werden. Wenn beschriebene Funktionen nicht in der dargestellten Weise funktionieren, wird darum gebeten, durch kurze Benachrichtigung an den verantwortlichen Autor zur Beseitigung von Fehlern beizutragen.

Kapitel 2

Elementgenerierung

2.1 *ELTYPE

Dieses Kommando weist den einzelnen Elementgruppen einen bestimmten Typ zu. Die Eingabe lautet:

```
*ELTYPE
anzahl_elementgruppen
eltyp_1
...
eltyp_n
```

In der ersten Folgezeile (`anzahl_elementgruppen`) stehen die Anzahl der Elementgruppen, in den weiteren Folgezeilen stehen die Elementtypzuordnungen. Bei der Eingabe für einen ABAQUS Datensatz können das alle zwei und dreidimensionalen Elemente sowie sämtliche selbstdefinierten Elemente sein, die von ABAQUS unterstützt werden.

2.2 *COORDINATES

Es können zur Definition des Modells Punkte in kartesischen sowie in Zylinderkoordinaten eingegeben werden. Diese Punkte stellen dann bei der zweidimensionalen Vernetzung die Eckpunkte von Elementnetzen dar. Allerdings werden die hier definierten Punkte nur von dem Kommando `*MAPE2D` ausgewertet. Es ist wichtig zu erwähnen, daß die vom Programm generierten Knoten nicht dieselben Nummern besitzen wie die Punkte. Ein Beispiel zur Verwendung von `*COORDINATES` erfolgt beim Befehl `*MAPE2D` weiter unten.

Nach dem Schlüsselwort `*COORDINATES` werden in den folgenden Zeilen solange Punktkoordinaten gelesen, bis in der ersten Spalte ein `*` steht, nicht gefolgt von einem weiteren `*` (dann ist es eine Kommentarzeile, die auch innerhalb der Punkteingabe möglich ist.)

Eingabeformat:

```
*COORDINATES
Nummer xp yp x0 y0
...
```

```
P Nummer rp phip x0 y0
...
```

x_p und y_p bezeichnen die kartesischen Punktkoordinaten und x_0/y_0 die globalen Koordinaten des Ursprungs des Koordinatensystems. Die Eingabe mit einem vorangestellten P kennzeichnet polare Koordinaten. Hierbei sind rp der Radius und $phip$ der Winkel, gemessen mathematisch positiv ausgehend von der positiven Horizontalen (3-Uhr-Stellung), bezogen auf den durch den mit x_0/y_0 angegebenen Ursprung des Koordinatensystems in globalen Koordinaten. Der Buchstabe P muß in der ersten Spalte stehen!

2.3 *MAPE2D: 2D-Netz durch vierseitige Flächen

Dieses Kommando wird für die Vernetzung einer vierseitigen Fläche in der x-y-Ebene verwendet. Als Ränder dieser Flächen können Kreisabschnitte gewählt werden, indem Punkte in polaren Koordinaten angegeben werden (siehe Befehl *COORDINATES).

Die Eingabe:

```
*MAPE2D, ELSET=number, NODE=nodes_per_elem
anzahleckpunkte , P1 , P2 , P3 , P4 (mind. 4 Eckpunkte!)
```

Es folgende weitere Unterkommandos, von denen das Kommando *MESHING notwendig, *BIAS optional ist. Das Unterkommando *COORDINATES ist notwendig, wenn es nicht schon vorher als Kommando eingegeben wurde.

Die Eingabe erfolgt gegen den Uhrzeigersinn. Werden mehr als 4 Eckpunkte angegeben, so bezeichnet der Punkt Nr. 5 einen Punkt auf der Kante von Punkt 1 nach Punkt 2, der Punkt Nr. 6 liegt auf der Kante von 2 nach 3, usw.. Auf diese Weise können gekrümmte Seiten auch ohne polare Koordinaten generiert werden. Ist neben den 4 notwendigen Eckknoten nur noch der Knoten Nr. 7 (zwischen Knoten 3 und 4) vorhanden, so werden die Knoten 5 und 6 mit 0 angegeben.

```
*MESHING
anzelemedge1, anzelemedge2, anzelemedge3, anzelemedge4
```

Dieses Kommando ist erforderlich, da es die Anzahl der zu generierenden Elemente angibt. Bemerkung: `anzelemedge3` und `anzelemedge4` werden automatisch auf `anzelemedge1` bzw. `anzelemedge2` gesetzt.

```
*BIAS
biasedge1 biasedge2 biasedge3 biasedge4
```

`biasedgei` gibt das Verhältnis der Elementkantenlänge von einem zum nächsten Element bei der Kante i an.

```
*COORDINATES
```

Die Folgezeilen sind unter dem gleichnamigen Kommando angegeben, siehe Kap. 2.2. Es stellt sich daher die Frage, warum überhaupt ein extra Kommando *COORDINATES existiert.

Beispiel:

Das in Abb. 2.1 dargestellte FE-Netz ist mit folgenden Befehlen erstellt worden:

```

*COORDINATE
P 1 1.0,135.0,0.0,0.0
P 2 1.0, 45.0,0.0,0.0
3 -3.0, 3.0
4 3.0, 3.0
5 -4.0, 4.0
6 4.0, 4.0
7 0.0, 12.0
MAPE2D,ELSET=1,NODE=8
4,1,2,4,3
MESHING
6,6,6,6
BIAS
1.0,1.2,1.0,1.2
MAPE2D,ELSET=1,NODE=8
7,3,4,6,5,0,0,7
MESHING
6,4,6,4
BIAS
1.0,0.7,1.0,0.7

```

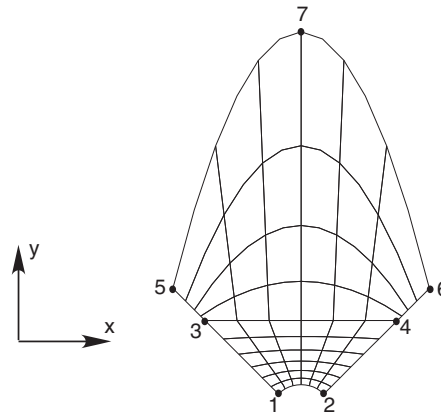


Abb. 2.1: Beispiel zur Anwendung von MAPE2D

2.4 *MESHVL: variable Netzvergrößerungen in 2D-Netzen

In der folgenden Zeile wird eines der folgenden Schlüsselwörter erwartet:

LURDL | LURD | URDL | LUR | URD | RDL | LU | UR | LR | RD | DL | UD | L | U | R | D

Die Buchstaben bezeichnen die Richtung der Elementvergrößerung: L: links, R: rechts; U: oben (engl. up); D: unten (engl. down). Hier und bei allen folgenden Befehlen ist die Definition folgendermaßen:

links → neg. x-Richtung, rechts → pos. x-Richtung

unten → neg. y-Richtung, oben → pos. y-Richtung

Eine Besonderheit ist das Kommando LURDL, da die Richtung LINKS zweifach vorkommt. Diese Möglichkeit ist für die Eingabe einer Rundumvergrößerung über einen Kerb hinweg gedacht. Der Kerb muß zur neg. x-Richtung hin öffnen. Es ist weiterhin die Eingabe von LYMIN bzw. LYMAX (s.u.) zur Positionsangabe des Kerbs notwendig.

Zusätzlich können in dieser Zeile noch Grenzen der für die Verfeinerung zu berücksichtigenden Ränder in der jeweiligen Richtung angegeben werden.

[RYMIN=value]	[UXMIN=value]
[RYMAX=value]	[UXMAX=value]
[LYMIN=value]	[DXMIN=value]
[LYMAX=value]	[DXMAX=value]

Es können mehrere dieser Schlüsselwörter in einer Zeile, getrennt durch Kommata vorkommen. Beispiel: Das FE-Netz geht von $x=0$ bis $x=10$ und soll in positiver y-Richtung

(also nach oben) vergrößert werden. Mit den erwähnten Schlüsselwörtern besteht auch die Möglichkeit, nur den Bereich von $x=3$ bis $x=7$ erweitern.

Anschließend folgt auf jeden Fall die Zeile

```
formnummer, elset, nodes_per_elem
```

(Formnummer kann die Werte 1 oder 2 annehmen; ergibt unterschiedliche Netzverfeinerungsabstufung). Die sich ergebenden Elementanordnungen (Pattern) sind in Abb. 2.2 dargestellt.

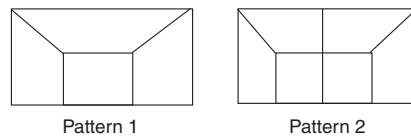


Abb. 2.2: verfügbare Module zur Netzvergrößerung: 3 auf 1 Element (Pattern 1), 4 auf 2 Elemente (Pattern 2)

Es folgen einer oder mehrere der folgenden Parameter

[TR=totalcoord]	[TU=totalcoord]
[DR=diffcoord]	[DU=diffcoord]
[TL=totalcoord]	[DD=totalcoord]
[DL=diffcoord]	[TD=diffcoord]

`totalcoord` bezeichnet die rechte / obere / untere / linke Koordinate im globalen System, bis zu der das Netz gehen soll.

`diffcoord` bezeichnet die jeweilige Koordinatendifferenz, um die das Netz in der jeweiligen Richtung erweitert wird.

Es können auch bei diesen Parametern wieder mehrere in einer Zeile, getrennt durch Kommata oder Leerzeichen, stehen

In der Richtung der Netzverfeinerung dürfen nicht 1, 2 oder 5 Elemente liegen, unabhängig von dem Pattern! (core dump!) Besonders ist darauf bei einer Einschränkung der Vergrößerung durch die `RXMIN`, `RXMAX`, etc. Parameter zu achten, da hier die Anzahl der Elemente evtl. nicht offensichtlich ist.

Zwei Beispiele zur Verdeutlichung (die grau dargestellten Elemente werden als vorhanden angenommen):

```
*MESHVL
LU
2,1,4
TL=-0.1, TU=0.3
```

Wie man sieht wird bei nicht passender Anzahl Elemente am Ende auch ein Pattern anderen Typs generiert.

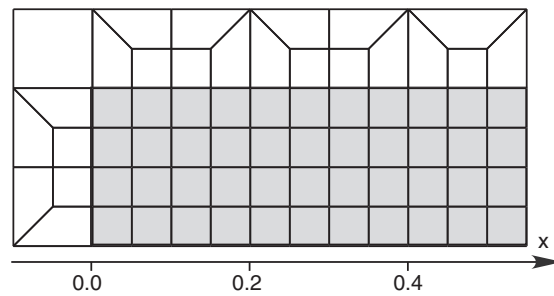


Abb. 2.3: Beispiel zur Anwendung von MESHVL

```
*MESHVL
LURDL,LYMIN=0.0
1,1,4
DL=-0.1, DU=0.1, DR=0.1,
DD=-0.1
```

Entscheidend ist, daß die Elemente an der Rifffnung tatschlich getrennt sind, da sonst ein Fehler auftritt.

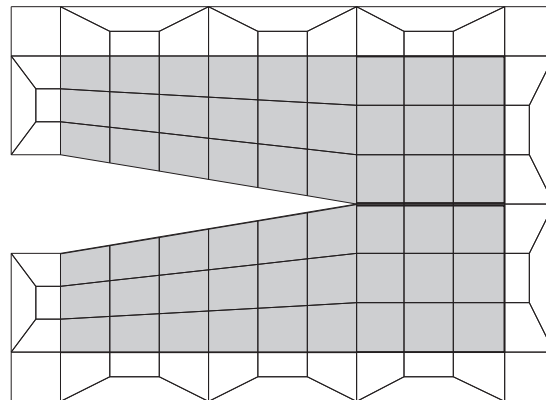


Abb. 2.4: Zweites Beispiel zur Anwendung von MESHVL

2.5 *BLFMESH1: Rispitzennetz mit runder Rispitze

Dieses Kommando erzeugt ein komplexes halbrundes Rispitzennetz der in Abb. 2.5 dargestellten Form:

Merkmal des Netzes ist eine abgerundete Rispitze (Radius `R_cracktip`). Da das erzeugte Netz eine relativ komplexe Konfiguration besitzt, besteht die Eingabe auch aus einem ganzen Satz von Parametern. Die vollstndige Eingabe hat folgendes Aussehen:

```
*BLFMESH1
titel1
elset, nodes_per_elem, NX1, NX2, NX3, NY1, XH, YH, R_tip, Len_tip, bias_tip, R_aussen
titel2
anzahl_schichten
elset, nodes_per_elem
anzahl_nodes_1, elementratio_1
...
```

`NX1`, `NX2`, `NX3` und `NY1` geben die Anzahl der Knoten in den angezeigten Bereichen der

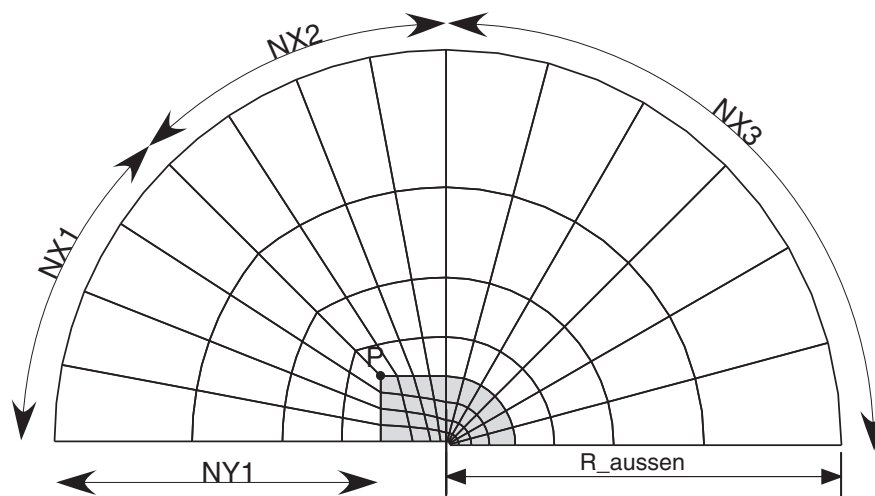


Abb. 2.5: Mit Hilfe des Befehls `*BLFMESH1` generiertes FE-Netz

Zeichnung an (Achtung, die Zahl der Elemente ist daher erst durch die Angabe von `nodes_per_elem` definiert!).

XH und YH bezeichnen den in der Abbildung eingezeichneten Eckpunkt P, die Angabe erfolgt jedoch nicht in absoluten Koordinaten, sondern beziehen sich auf die x- bzw. y-Koordinate der nach unten bzw. nach rechts weitergeführten Elementlinie. In der obigen Zeichnung besitzen XH und YH die Werte 1. Bei einem Wert von XH und YH < 1 wandert der Punkt zur Rißspitze. `R_tip` ist die Absolutkoordinate des Rißspitzenradius'. Der Ursprung der Kreises besitzt die Koordinaten 0,0/0,0. `R_aussen` ist der Außenradius des Elementpatterns. `len_tip` ist die Größe der Rißspitzenelemente selbst, und `bias_tip` ist das Elementkanten-Längenverhältnis der grau dargestellten Rißspitzenelemente (Anzahl: NX1 Elemente) in radialer Richtung. Die beiden letztgenannten Parameter bestimmen daher die Größe des Rings dieser Elemente (in der Darstellung grau schraffiert) bei der Generierung. Das Kantenlängenverhältnis der anschließenden NY1 Elemente wird aufgrund der Größe der letzten Elemente im inneren Bereich (grau schraffiert) und dem verbleibenden Platz zwischen diesem Bereich und dem angegebenen Außenradius berechnet. Würde sich in radialer Richtung eine Elementverkleinerung ergeben, wird die Anzahl der Elemente automatisch korrigiert, so daß die Größe der Elementkanten in radialer Richtung annähernd konstant bleibt. Die Anzahl der auf diese Weise erzeugten Elemente beträgt

$$\left[NY1 \times (NX1 + NX2 + NX3) + NX1 \times (NX2 + NX3) \right] / (\text{nodes_per_elem}/2)$$

Das Beispiel in Abb. 2.5 wurde durch folgende Befehlszeile erzeugt:

```
*BLFMESH1
elset, node/elem, NX1, NX2, NX3, NY1, XH, YH, R_ctip, len_ctip, bias_ctip, R_aussen
1      , 8          , 8      , 8      , 12   , 8      , 1.0,1.0, 0.01 , 0.01   , 1.2      , 1.0
Titel2: keine weitere Zeile ...
0
```

Nach dem zweiten Titel können weitere Elementschichten in radialer Richtung dem Pattern zugefügt werden. Jede Schicht - die Gesamtzahl der Schichten wird in `anzahl_schichten`

definiert - kann aus mehreren Elementen in radialer Richtung bestehen. `anzahl_nodes` gibt die Anzahl der Knoten (in radialer Richtung) an, `elementratio` definiert das Größenverhältnis der Elemente, ebenfalls in radialer Richtung. In der Abbildung sind keine solchen zusätzlichen Elementschichten definiert. Das Netz kann für die weitere Generierung mit dem Befehl *IPLMESH, Kap. 2.9, ergänzt werden. Dieser Befehl schließt Elemente derart an das bestehende Netz an, daß gerade Strukturkanten entstehen. Es ist allerdings dann darauf zu achten, daß die Anzahl der Knoten $NX3$ sowie $NX1+NX2$ durch 2, bei quadratischen Elementen sogar durch 4 teilbar ist, um ein Anschlußnetz mit *IPLMESH zu ermöglichen.

2.6 *BLFMESH2: Reißspitzennetz mit scharfer Reißspitze

Dieses Kommando erzeugt ein Reißspitzennetz ähnlich wie *BLFMESH1, allerdings mit scharfer Reißspitze. Außerdem besteht das Netz nicht unbedingt aus einem Halbkreis, sondern die Winkel sind definierbar. Abb. 2.6 verdeutlicht das Aussehen des Netzes:

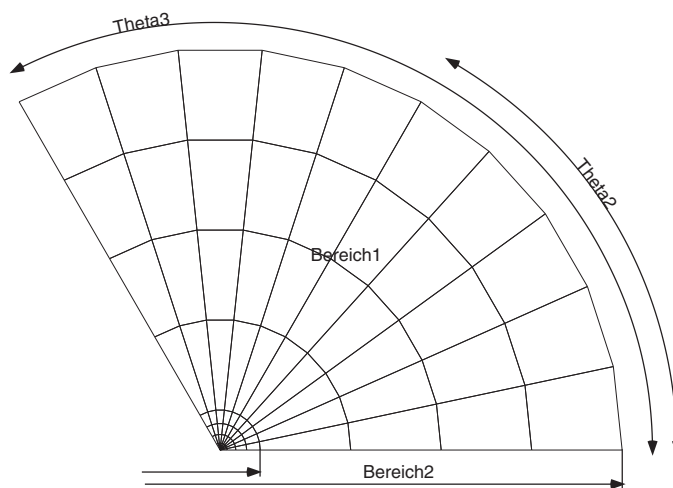


Abb. 2.6: Mit Hilfe des Befehls *BLFMESH2 generiertes FE-Netz

Der Aufruf erfolgt nach folgendem Schema:

```
*BLFMESH2
elset, nodes_per_elem, circ_elem_1, circ_elem2, Theta3, Theta2,
Theta1, X0, Y0, ivar, typeflag, Radius
anzahl_schichten
anzahl_nodes_1, elementratio_1
...
```

Die Werte `circ_elem_1` und `circ_elem_2` beziehen sich auf die Anzahl der Elemente (nicht der Knoten wie bei *BLFMESH1!) in Umfangsrichtung in den beiden Bereichen zwischen 0 und `Theta2` sowie `Theta2` und `Theta3`. Der Fächer des Netzes fängt immer in horizontaler Richtung nach rechts an. Setzt man `Theta1` > 0 , so wird zwischen der Horizontalen und `Theta1` ein Element gesetzt und zwischen `Theta1` und `Theta2` genau (`circ_elem_1` - 1) Elemente. `X0/Y0` geben die Koordinaten der Reißspitze an, die allerdings mit dem

Kommando `*ORIGIN` manipuliert werden können. Mit `Radius` wird der Außenradius des gesamten Netzes definiert.

Mit `typeflag` wird das Reißspitzennetz manipuliert. Es bedeuten

typeflag=0: Reißspitzenelement mit Viertelpunkt-Singularität (8-Knotenelemente)

typeflag=1: Reißspitzenelement mit normaler Singularität (8-Knotenelemente)

typeflag=2: Reißspitzenelement mit zusammengefaßten Reißspitzenknoten (8-Knotenelemente)

typeflag=3: Reißspitzenelement mit zusammengefaßten Reißspitzenknoten (6-Knotenelemente)

`ivar` sollte immer auf 0 stehen.

In den Folgezeilen wird definiert, wie viele Bereiche mit wie vielen Elementen in radialer Richtung generiert werden sollen. Der Radius eines Bereiches ist dabei immer 10 mal so groß wie der nächstkleinere. Bei den zwei Bereichen in der Abbildung geht der erste Bereich also bis $0,1 \times \text{Radius}$, der zweite bis $1,0 \times \text{Radius}$.

2.7 *BLFMESH3: gleichmäßiges (Reißspitzen-)netz

Mit diesem Kommando wird ein gleichmäßiges Netz mit zwei Bereichen erzeugt, definiert durch drei Angaben über die Knotenanzahl und 5 Koordinatenpunkte. Der erzeugte Bereich die in Abb. 2.7 dargestellte Form.

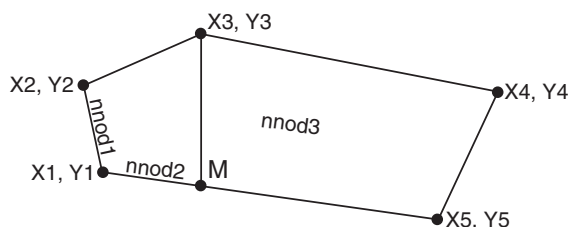


Abb. 2.7: FE-Netz, generiert mit den Befehlen `*BLFMESH3`

Die Eingabe lautet: `*BLFMESH3`

`nnod1, nnod2, nnod3, elset, nodes_per_elem`

`X1, Y1`

`X2, Y2`

`X3, Y3`

`X4, Y4`

`X5, Y5`

Die y -Koordinate des Punktes `M` wird derart generiert, daß zwischen den Punkten 1 und 5 eine gerade Linie entsteht. Die x -Koordinate des Punktes `M` ist identisch mit `Y3`.

Für die meisten Anwendungen ist das Kommando `*BLFMESH4`, `*CGROWTH1` oder `*MAPE2D` besser geeignet, einfacher oder variabler.

2.8 *BLFMESH4: gleichmäßiges (Rißspitzen-)netz mit vorgegebener Elementanzahl

```
knotenx, knoteny, elset, nodes_per_elem
X1, Y1
X2, Y2
X3, Y3
X4, Y4
```

`knotenx` und `knoteny` bezeichnen die Anzahl der Knoten in x bzw. y-Richtung (ohne Startknoten) (Beispiel: 2,2,1,8 erzeugt 1 8-knotiges Element, während 3,8,1,4 24 4-knotige Elemente generiert - 3 in x-Richtung und 8 in y-Richtung).

Achtung! Werden die 4 Koordinatenpaare rechtsherum angegeben, so werden die Elemente „linksherum“ definiert (und umgekehrt).

2.9 *IPLMESH: Anschlußnetz für *BLFMESH-Kommandos

Dieser Befehl dient der Generierung von geraden Strukturkanten im Anschluß an den Befehl *BLFMESH1, *BLFMESH2, *BLFMESH3 oder *BLFMESH4. Die Eingabe ist folgendermaßen:

```
*IPLMESH
nodes, elset, nodes_per_elem
x1, y1
x2, y2
x3, y3
x4, y4
x5, y5
```

Das hiermit generierte Netz und die einzugebenen Koordinaten sind in Abb. 2.8 ersichtlich (die mit dem vorhergegangenen Befehl *BLFMESH1 erzeugten Elemente sind grau dargestellt). Der Elementtyp (lineare oder quadratische Elemente) sollte dem mit den *BLFMESH-Kommandos generierten Typ entsprechen. In dem in Abb. 2.8 dargestellten Beispiel lautet das Kommando zum dargestellten Netz:

```
*IPLMESH
6, 1, 8
-2.5, 0.001
-2.5, 2.5
0.0, 2.5
2.5, 2.5
2.5, 0.0
```

Durch die Eingabe der 5 Koordinatenpaare werden 4 Strukturkanten des vorangegangenen *BLFMESH-Kommandos erwartet, die das mit erstellte *IPLMESH-Netz umschließt. Bei *BLFMESH4 existieren nur 3 Strukturkanten (die Kanten P1-P2, P2-P3 und P3-P4). Als Folge davon werden die Elemente der Kante P2-P3 des *BLFMESH4-Kommandos auf die Kanten P2-P3 und P3-P4 des *IPLMESH-Kommandos aufgeteilt.

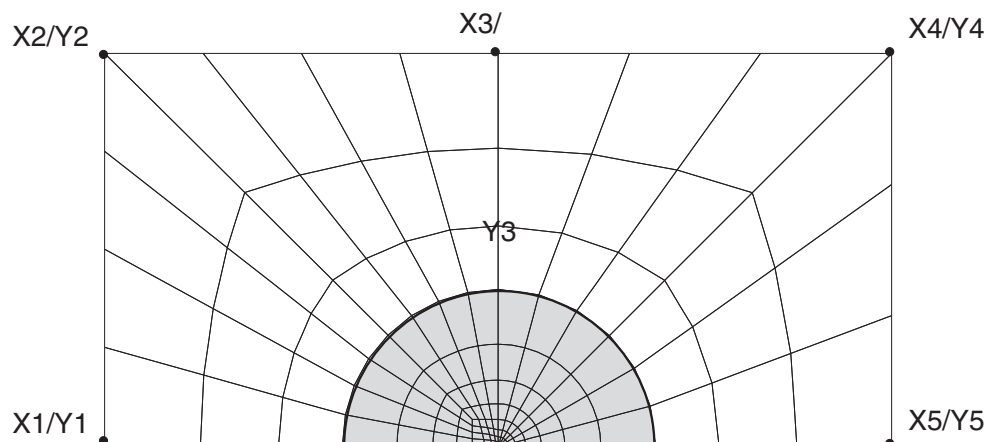


Abb. 2.8: FE-Netz, generiert mit den Befehlen *BLFMESH1 (graue Elemente) und *IPLMESH

2.10 *CRACKTIP1: Reißspitzenzelle

Diese Kommando erzeugt folgendes Elementnetz:

Die Eingabe erfolgt folgendermaßen:

*CRACKTIP1

elset, nodes_per_elem, deltaX, deltaY

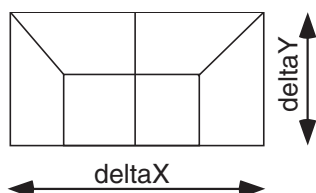


Abb. 2.9: Beispiel zur Anwendung von *CRACKTIP1

2.11 *RIGMESH: Netzerweiterung am rechten Rand

*RIGMESH [, YMIN=ymin] [, YMAX=ymax]

anzahlgruppen

(-)anzahlnodes, totalkoord, elset, nodes_per_elem

Ein negatives Vorzeichen vor `anzahlnodes` bewirkt, daß die Elemente gleichmäßig in den Zwischenraum zwischen aktuellem Rand und `totalkoord` gesetzt werden. Ohne Vorzeichen werden die Elemente ggfs. von der Elementgröße des am weitesten rechts liegenden Elementes nach rechts hin vergrößert. Eine Verfeinerung wird nicht durchgeführt.

Ist `anzahlnodes = -999`, so wird die Anzahl der zu generierenden Elemente aus der vor-

her berechnete Elementlänge am bestehenden Rand und der eingegebenen Koordinate ermittelt. Mit YMIN und YMAX können Grenzen definiert werden, in denen die Erweiterung generiert wird.

2.12 *LEFMESH: Netzerweiterung am linken Rand

Eingabe wie *RIGMESH

```
*LEFMESH [,YMIN=ymin][,YMAX=ymax]
anzahlgruppen
(-)anzahlnodes,totalkoord,elset,nodes_per_elem
```

2.13 *UPSMESH: Netzerweiterung am oberen Rand

Eingabe wie *RIGMESH

```
*UPSMESH [,XMIN=ymin][,XMAX=ymax]
anzahlgruppen
(-)anzahlnodes,totalkoord,elset,nodes_per_elem
```

2.14 *DWNMESH: Netzerweiterung am unteren Rand

Eingabe wie *RIGMESH

```
*DWNMESH [,XMIN=ymin][,XMAX=ymax]
anzahlgruppen
(-)anzahlnodes,totalkoord,elset,nodes_per_elem
```

2.15 *CGROWTH1: gleichmäßiges Reißspitzennetz für Reißfortschritt

```
*CGROWTH1
node_distance, total_length, elset, nodes_per_elem
```

Mit diesem Befehl wird ein gleichmäßiges Reißspitzennetz für die Verwendung zur Reißfortschrittsrechnung generiert. Es besteht aus zwei Elementen übereinander (in y-Richtung) und einer Anzahl Elemente nebeneinander (in x-Richtung), die sich aus dem Abstand der Knoten zueinander (`node_distance`) sowie der Gesamtlänge des generierten Netzes (`total_length`) und der Anzahl der Knoten/Element (`nodes_per_elem`) ergibt. Die Elemente sind immer quadratisch.

Ein Beispiel ist bei der Beschreibung des Befehls *CGROWTH2 angeführt, siehe Abb. 2.10.

2.16 *CGROWTH2: 2D Netzvergrößerung an CGROWTH1

```
*CGROWTH2
elset,nodes_per_elem, 0
```

Dieses Kommando erzeugt eine Netzvergrößerung nach links, rechts und oben. Voraussetzung ist ein Netz mit beliebig vielen Elementen angeordnet in zwei Reihen. Das daraufhin erstellte Pattern zeigt Abb. 2.10:

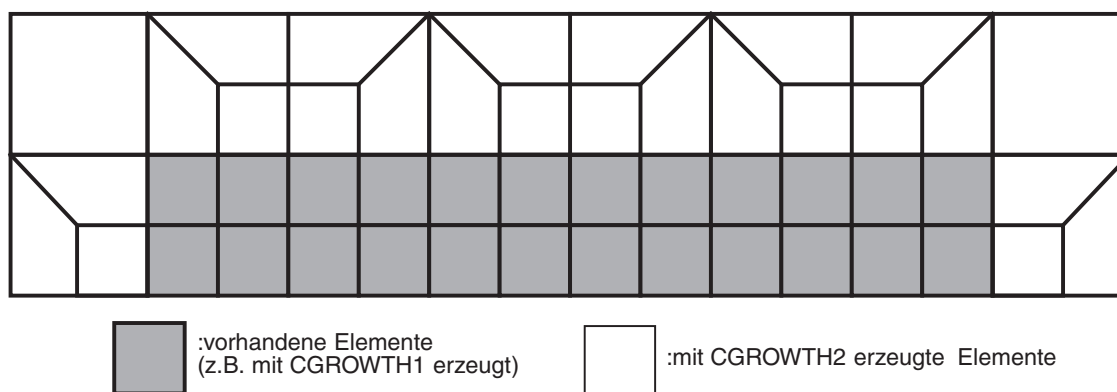


Abb. 2.10: FE-Netz, generiert mit den Befehlen *CGROWTH1 und *CGROWTH2

Das gesamte Netz in Abb. 2.10 wurde (abhängig von der Anzahl der Knoten/Element) erstellt durch

4-Knoten-Elemente

```
*CGROWTH1
```

```
0.1, 1.2, 1, 4
```

```
CGROWTH2
```

```
1, 4, 0
```

8-Knoten-Elemente

```
*CGROWTH1
```

```
0.05, 1.2, 1, 8
```

```
CGROWTH2
```

```
1, 8, 0
```

2.17 *CGROWTH4: 2D Netzvergrößerung nach links und rechts

Dieses Kommando erzeugt eine Netzvergrößerung nach links und rechts. Voraussetzung ist wieder die zweireihige Elementanordnung wie für den Befehl *CGROWTH2.

```
*CGROWTH4
elset,nodes_per_elem, 0
RA=rightabs|RD=rightdiff, LA=leftabs|LD=leftdiff
```

Die Größe des Pattern richtet sich nach der zweiten Folgezeile, in der die Größenparameter absolut oder relativ angegeben werden. Es kann durch die alleinige Eingabe des Wertes für die Erweiterung nach rechts die Vergrößerung nur nach rechts vorgenommen werden, das gleiche gilt für die Eingabe nur nach links. Bei gleichzeitiger Eingabe einer absoluten

und einer relativen Angabe in der gleichen Richtung wird der zuletzt eingegebene Wert verwendet.

2.18 *HBLF MESH: Netzerweiterung für BLF (symmetrisch)

Dieses Kommando dient zur Erweiterung eines vorhandenen Reißspitzennetzes für die Boundary Layer Formulation, speziell für symmetrische Fälle.

Die Eingabe erfolgt nach folgendem Schema:

```
*HBLF MESH
n_rad_elem, elset, nodes_per_elem
anzahl
Xo, Yo, Radius, Theta_begin, Theta_end
...
```

`n_rad_elem` gibt die Anzahl der Elemente in radialer Richtung an.

`anzahl` bezeichnet die Anzahl der einzelnen Kreisabschnitte der BLF. Sie entspricht der Anzahl der Strukturkanten des Reißspitzennetzes (i.allg. drei Kanten).

`Xo/Yo` ist das Koordinatenpaar der Reißspitze und sollte für alle nachfolgenden gleich sein. Ebenso der Außenradius des erzeugten Netzes. `Theta_begin` und `Theta_end` definieren Anfang und Ende des Kreissegments, einzugeben im Uhrzeigersinn. Für ein BLF Netz sollte das erste Segment bei 180 beginnen und das letzte bei 0 aufhören.

2.19 *FBLF MESH: Netzerweiterung für BLF (unsymm.)

Dieses Kommando ist das Pendant zum *HBLF MESH für unsymmetrische Fälle. Die Eingabe erfolgt nach dem gleichen Schema, wird aber ergänzt durch die Angabe der Reißöffnung. Insgesamt ergibt sich folgende Eingabe:

```
*FBLF MESH
n_rad_elem, elset, nodes_per_elem
anzahl
Xo, Yo, Radius, Theta_begin, Theta_end
...
FULL=L
crack_plane
```

`anzahl` bezeichnet auch hier die Anzahl der Kanten und sollte demgemäß auf 5 stehen.

Durch `FULL=L` wird angegeben, daß sich die Reißöffnung an der linken Seite befindet, und `crack_plane` gibt die y-Koordinate des Risses an.

Wichtig ist noch anzumerken, daß ein Reiß vorhanden sein muß, d.h. ie Angabe von 5 Segmenten ist ebenso notwendig wie die Definition der `crack_plane`.

2.20 *CRACK2: Auftrennen von Elementen entlang einer Linie

Mit diesem Kommando ist es möglich, ein 2D Netz durch Definition einer Linie entlang innerer Elementkanten aufzutrennen. die Eingabe erfolgt durch

```
*CRACK2
H[B]                oder                V[B]
Ybegin, Xbegin, Xende, deltaY          Xbegin, Ybegin, Yende, deltaX
```

Die Rißspitze liegt hierbei definitionsgemäß am Ende der definierten Linie, wenn nicht der Buchstabe B hinter V bzw. H eingegeben wurde. Die Rißflanken werden um `deltaX` bzw. `deltaY` verschoben. In Zusammenhang mit *BLFMESH2 kann es hierbei aber zu Problemen kommen!

2.21 *CNV2TRI: Umwandlung von Vierecks- in Dreieckselemente

Mit Hilfe dieses Befehles lassen sich aus jeweils einem viereckigen Element vier Dreieckselemente erzeugen. Die Definition der Elemente, aus denen Dreieckselemente generiert werden sollen, erfolgt genauso wie die Definition von Elementsets, siehe Kap. 4.2. Es können Vierknoten-Elemente sowie Achtknoten-Elemente verarbeitet werden.

```
*CNV2TRI, [ELSET=number]
[*POLYGON | *POINT | *CIRCLE | *POLYLINE ]
(je nach gewählter Option)
```

Die Angabe eines Elementsets ist optional und bezeichnet, den Set, in dem nach vorhandenen Elementen gesucht werden soll. Default ist das gesamte Modell. Die generierten Elemente kommen in den selben Elementset. Eventuell müssen sie noch mittels *RENEW ELSET, Kap. 3.11, in einen anderen Elementset umgewandelt werden.

2.22 *CNV2TET: Umwandlung von Bricks in Tetraederelemente

Mit Hilfe dieses Befehles lassen sich aus jeweils einem 3D-Hexaederelement sechs bzw. 24 Tetraederelemente erzeugen. Die Definition des umzuwandelnden Bereichs erfolgt genauso wie die Definition von Elementsets (siehe dort) Es können z.Zt. nur lineare (8 Knoten-) Elemente verarbeitet werden.

```
*CNV2TET, [ELSET=number] [, TYPE=1|2]
[*POLYGON | *POINT | *CIRCLE | *POLYLINE ]
(je nach gewählter Option)
```

Die Angabe eines Elementsets ist optional und bezeichnet den Set, in dem nach vorhandenen Elementen gesucht werden soll. Default ist das gesamte Modell. Die generierten

Elemente kommen in den selben Elementset. Eventuell müssen sie noch mittels *RENEW ELSET in einen anderen Elementset umgewandelt werden.

Durch die Eingabe des Typs kann die Anzahl der zu generierenden Tetraederelemente bestimmt werden. Es gilt:

TYP=1 6 lineare Tetraederelemente

TYP=2 24 lineare Tetraederelemente

Für die Definition von 24 linearen Tetraederelementen werden automatisch ein Elementmittenknoten und sechs Flächenmittelknoten generiert.

2.23 *MESH 9 NODE: Umwandlung von 8- in 9-Knoten-Elemente

Dieses Kommando wandelt bereits generierte 8-Knotenelemente in solche mit 9 Knoten um. Es werden alle 8-Knoten-Elemente einbezogen. Die Eingabe weiterer Parameter ist nicht erforderlich.

2.24 *COHESIV1: Generierung von 1D Kohäsivzonenelementen an Strukturkanten

Mit diesem Befehl lassen sich Kohäsivzonenelemente an Strukturkanten von 2D Solidelementen generieren. Es entstehen je nach der Art der bestehenden Kontinuums-elemente (linear oder quadratisch) 1 oder 2 kohäsive Elemente, deren Knoten auf der einen Seite mit den Kontinuums-elementen verbunden sind. Die Knoten auf der anderen Seite besitzen keine weitere Elementanbindung, können jedoch durch einen entsprechenden Parameter im Befehl *ENSET, Kap. 4.1, zu einem Knotenset zusammengefaßt werden, der den Namen BOTTOM trägt. Auch die Verbindung der gegenüberliegenden Knoten mit Hilfe von Zwangsbedingungen ist durch einem Parameter in *ENSET möglich.

Die Eingabe erfolgt im zweidimensionalen Raum, an dem die Strukturkante durch eine Linie, ein Polygon oder ein Kreis beschrieben werden kann.

```
*COHESIV1, NEW ELSET=number[, OLD ELSET=number]
```

```
*LINE
```

```
X1, Y1, X2, Y2
```

```
oder
```

```
*POLYGON
```

```
X1, Y1, X2, Y2
```

```
oder
```

```
*CIRCLE
```

```
Xo, Yo, R, Theta1, Theta2
```

Die Eingabe der Kohäsivzone durch *CIRCLE scheint nicht zu funktionieren.

2.25 *COHESIV3: Generierung von 2D Kohäsivzonenelementen an Strukturkanten

Dieses Kommando dient der Definition von Interface-Flächen in der Symmetrieebene einer 3D-Struktur. Es können folgende 3D-Elemente verwendet werden: C3D4, C3D8, C3D20, die Implementierung von Elementen für den Typ C3D10 ist geplant. An die Unterseite der Kontinuums-elemente werden die Kohäsivelemente generiert.

Bei der Definition von Kohäsivelementen für ABAQUS werden abhängig von der Definition der bestehenden Kontinuums-elemente 6-knotige Dreieckselemente oder 8-knotige Viereckselemente verwendet, von denen die drei bzw. vier unteren Knoten 'in der Luft hängen', aber in dem Knotenset BOTTOM zusammengefaßt sind. Dieser Set kann mit Hilfe der Setdefinition *ENSET ausgegeben werden. Dort kann auch ein Satz von Zwangsbedingungen generiert werden, um die unteren Knoten in x- und z-Richtung an die Ligamentknoten zu fesseln.

Die Eingabe lautet:

```
*COHESIV3
elset, X1, X2, Z1, Z2
```

2.26 *CZDS: Generierung von Kohäsivelementen zwischen Kontinuums-elementen (1D und 2D)

Die Kohäsivzonenelemente können als Linie für 2D-Modelle oder als Ebene für 3D-Modelle eingegeben werden. Außerdem besteht die Möglichkeit, entlang der Grenzflächen einer beliebigen Anzahl von Elementsets Kohäsivzonenelemente zu generieren. Bei den dreidimensionalen Modellen müssen die Knoten der Elemente, zwischen die die Kohäsivzone gelegt werden soll, verbunden sein, also die gleiche Knotennummer besitzen.

Die Eingabe ist folgendermaßen:

```
*CZDS[,NEW ELSET=number][,OLD ELSET=number][,NODES=nodes_per_elem]
```

Mit Hilfe der Eingabe von `nodes_per_elem` kann gesteuert werden, ob zwischen zwei Hexaederelemente jeweils ein viereckiges (NODES=8) oder vier dreieckige Interfaceelemente (NODES=6) gelegt werden.

Folgezeilen für 2D-Modelle

```
*LINE
X1, Y1, X2, Y2
```

Folgezeilen für 3D-Modelle

```
*PLANE
Xctip, Yctip, Zctip
Xcrackprop, Ycrackprop, Zcrackprop, Xnormal, Ynormal, Znormal
```

Xctip sind Koordinaten eines Punktes an der Reißfront. Mit Xcrackprop wird Richtung und Länge der zu trennenden Elemente vorgegeben, und mit Xnormal wird der Vektor normal zur Bruchfläche angegeben.

Für die Generierung von Kohäsivelementen zwischen die angrenzenden Elemente zweier Elsets lautet die Eingabe:

```
*ELSET
anzahl_elsets
elset1, elset2, ...
```

Es wird zwischen allen Elementsets nach gemeinsamen Elementoberflächen gesucht.

2.27 *CZDOMAIN: Vollständige Vernetzung einer Region mit Kohäsivelementen

Die Eingabe erfolgt analog zu der Definition von Elementsets, Kap. 4.2, also

```
*CZDOMAIN [,OLD ELSET=old_number][,NEW ELSET=new_number]
*POLYGON
X1, Y1, Z1, X2, Y2, Z2 oder
*CIRCLE
x0, y0
r1, r2, t1, t2, z1, z2
```

Es werden alle Elemente in der definierten Region, evtl. eingeschränkt durch die Vorgabe des Elementsets, gesucht. Anschließend werden alle Kanten zwischen diesen Elementen getrennt und mit Kohäsivelementen wieder verbunden.

Achtung! Es können so leicht sehr viele Elemente generiert werden.

2.28 *MESH3D: 3D Netz durch Sweepen in Normalenrichtung

Durch dieses Kommando werden die 2D Elemente in der z-Richtung aufgedickt. Die Anzahl der Elementschichten kann definiert werden. Durch das Sweepen in mehreren Schritten können unterschiedliche Verfeinerungen in Dickenrichtung eingegeben werden. Die 2D Elemente werden durch die Generierung automatisch gelöscht. Die Eingabe ist folgendermaßen:

```
*MESH3D
thickness
number
n_elem_1, bias_1, thickness_1
...
```

Wird **thickness** in der ersten Parameterzeile auf 0 gesetzt, so ergibt sich die Gesamtdicke aus der Summe der Einzeldicken, sonst werden die Einzeldicken aus der Anzahl der Elemente und der Gesamtdicke berechnet.

Die Generierung erfolgt so, daß zunächst die bestehenden Knoten auf die maximale z-Koordinate gesetzt werden, und danach mit dem angegebenen Bias die Elemente nach 0

hin generiert werden. Die letzten Knoten sind daher die bei $z=0$. Von dort aus gesehen ist das Bias daher auch invers zu verstehen.

2.29 *SIDE GROOVE: Seitenkerbengenerierung in 3D Geometrien

Dieses Kommando ist speziell für die dreidimensionale Berechnung bruchmechanischer Proben gedacht. Hier dient es der Definition von Seitenkerben. Zunächst werden die Knoten definiert, die durch das Kommando verändert werden sollen. Dies geschieht durch Angabe einer Box, die mit dem Schlüsselwort *POLYGON definiert wird. Anschließend folgt die Angabe der Kerbe selbst mit Hilfe des Kommandos *SIDE GROOVE=...

Die gesamte Eingabe zur Generierung von Seitenkerben sieht folgendermaßen aus:

```
*SIDE GROOVE
POLYGON
X1, Y1, Z1, X2, Y2, Z2
SIDE GROOVE = [YX|XZ|ZY|YZ]
X0, Y0, X1, Y1
```

Es wird jeweils nur eine Koordinate aller Knoten verändert. Dies ist bei YX die Y-Koordinate, bei XZ die X-Koordinate und bei ZY die Z-Koordinate. Dies geschieht durch

$$X_{1,neu} = X_1 + X1/X_{1,1} * X_{1,0} * (X_2 - X_{2,1})/(X_{2,0} - X_{2,1})$$

Hierbei ist X1 durch die erste im *SIDE GROOVE Befehl angegebene Koordinatenrichtung zu ersetzen und X2 durch die zweite. (Im Falle YX als X1 = Y und X2 = X)

Ein Spezialfall stellt YZ dar: Hierbei wird nur die Angabe X0 ausgewertet, die die Weite der Seitenkerbe angibt. Die anderen Werte werden von dem BOX Kommando gelesen. Die Formel zur Berechnung der neuen Y-Koordinate der Knoten ergibt sich aus:

$$Y_{neu} = Y + (Z - Z1)/(Z2 - Z1) * (Y - Y1)/(Y2 - Y1) * X0$$

2.30 *EXT_3D4: Sweepen eines vorhandenen 3D-Netzes

Mit diesem Befehl wird ein vorhandenes 3D-Netz (erzeugt z.B. mit *MESH3D) in eine Koordinatenrichtung (positiv oder negativ) erweitert. Die Eingabe erfolgt durch

```
*EXT_3D4 [ELSET=elsetnumber]
XL | XR | YD | YU | ZB | ZF
anzahl_elemente, Absolutkoord[, bias]
```

Es kann nur eine der 6 möglichen Richtungsangaben angegeben werden. Die Bezeichnungen XL, YD und ZB bezeichnen hierbei die negativen Koordinatenrichtungen. Die Angabe von `anzahl_elemente` kann positiv oder negativ sein; bei Eingabe einer negativen Elementanzahl werden die Elemente gleichmäßig verteilt, oder, bei zusätzlicher Angabe des

`bias`-Parameters, mit diesem vergrößert. Bei positiver Angabe von `anzahl_elemente` wird ebenfalls ein biasing vorgenommen, wobei die mittlere Länge der zu erweiternden Elementoberflächen als Ausgangselementlänge verwendet wird. Der Parameter `bias` hat dann keine Bedeutung. Sollte sich durch die Anzahl der Elemente `bias < 1` ergeben, also eine Verdichtung des Netzes, so wird automatisch `bias=1` eingestellt, und die Anzahl der Elemente automatisch angepaßt.

2.31 *CFRONT: Definition einer gekrümmten Rißfront

Hierdurch besteht die Möglichkeit, bei dreidimensionalen Rißberechnungen die Knotenkoordinaten so zu verschieben, daß eine ursprünglich als gerade Linie generierte Rißfront durch eine Parabel beschrieben wird. Dieses Kommando ist sehr speziell auf Risse zugeschnitten, deren Fortschrittsrichtung die positive x-Koordinate ist. Die Eingabe geschieht durch

```
*CFRONT
shift_2, shift_3, shift_4 [, thickness]
x1, x2, x3, x4, x5
```

Geändert wird nur die x-Koordinate, und zwar abhängig von der momentanen Position. Die neue x-Koordinate x_{shift} wird berechnet aus

$$\text{shift} = \text{shift}_i + (X - X_i)/(X_{i+1} - X_i) * (\text{shift}_{i+1} - \text{shift}_i)$$

$$x_{\text{shift}} = x + \text{shift} * \sqrt{1 - z^2/\text{thickness}^2}$$

`shift_1` und `shift_5` werden nicht eingegeben, sondern sind grundsätzlich 0.

Aus der Formel ergibt sich, daß bei `x_i` die Koordinaten bei `z=0` um `shift_i` verschoben werden, die Knoten bei `z=thickness` werden nicht verändert. `thickness` wird während der Ausführung des Kommandos `*MESH3D` gesetzt und hier verwendet, wenn der Wert hier nicht als Parameter angegeben wurde.

Zwischen den x-Werten wird linear interpoliert, x-Werte `< x_1` bzw. `> x_5` werden nicht verändert. Entlang der z-Achse werden die Verschiebungen in x-Richtung quadratisch interpoliert.

2.32 *FLIP: Spiegeln eines vorhandenen Netzes

Mit Hilfe dieses Befehls werden Elemente eines angebbaren Bereiches in einer Richtung gespiegelt. Die Normalenachse der Spiegelebene ist eine der Koordiantenachsen, und die Position der Ebene wird ebenfalls durch den Benutzer angegeben. Die Eingabe erfolgt durch

```
*FLIP [,DIR=X|Y|Z] [,REF=ref_plane] [,REP=nrepeat] [,REMOVE] [,NEWSET]
```

Zusätzlich kann der Bereich, für den die Spiegelung vorgenommen werden soll, durch die optionalen Folgezeilen

*DETAIL

xmin, ymin, zmin, xmax, ymax, zmax

definiert werden. Der Default ist das gesamte Modell. Eine Eliminierung doppelter Knoten wird nur bei der Angabe des Schlüsselwortes **REMOVE** vorgenommen. Mit Hilfe des optionalen Parameters **NEWSET** kann bestimmt werden, dass die gespiegelten Elemente neuen Elementsets zugeordnet werden.

2.33 *COPY: Kopieren eines vorhandenen Netzes

Mit Hilfe dieses Befehls werden Elemente in einem angebbaren an eine andere Stelle kopiert. Die Eingabe erfolgt analog zum Kommando ***FLIP** mit defierbarer Richtung und Bereichsangabe. Wie oft der Kopiervorgang wiederholt wird, kann durch den Parameter **REP** eingegeben werden. Die Eingabe erfolgt durch

*COPY [,DIR=X|Y|Z] [, REF=refencecoord] [, REP=nrepeat]

Kopiert wird immer in die positive Koordinatenrichtung! Zusätzlich kann der Bereich, für den der Vorgang vorgenommen werden soll, durch die optionalen Folgezeilen

*DETAIL

xmin, ymin, zmin, xmax, ymax, zmax

definiert werden. Eine Eliminierung doppelter Knoten wird hier nicht vorgenommen. Ebenso fehlt die Überprüfung, ob das kopierte Netz mit dem schon vorhandenen zusammenpaßt.

2.34 *SYSMESH: Spiegeln eines vorhandenen Netzes

Eingabe:

*SYSMESH

axis, ref_plane, layer (?), new_elset

Dieses Kommando funktioniert wie ***FLIP**. **axis** kann allerdings die Werte 1, 2 oder 3 annehmen. Der Set, zu dem die neuen Elemente gehören sollen, kann außerdem explizit mit **new_elset** angegeben werden.

2.35 *RCAEDS: Einlesen einer weiteren Eingabedatei

Dieser Befehl dient der Eingabe von Knotenkoordinaten und Elementdefinitionen von einem weiteren Inputfile (ursprünglich von CAEDS, daher der Name). Die Eingabe lautet:

*RCAEDS

filename

REORDER,MAGX=magfactor_x, MAGY=magfactor_y, MAGZ=magfactor_z,

ORIGIN=x_orig, y_orig, z_orig

Alle Parameter sind optional. Durch den Parameter `REORDER` (besitzt keine Option) kann man angeben, daß 2D-Elemente andersherum numeriert werden sollen. Die Parameter `MAGX`, `MAGY` und `MAGZ` geben einen Vergrößerungsfaktor für die Knotenkoordinaten an. Mit `ORIGIN` kann man den Nullpunkt zusätzlich verschieben (Es gilt $\text{coord_new} = \text{MAGi} * \text{coord} + \text{i.orig}$). Die eingelesenen Elementsets bekommen grundsätzlich neue Nummern.

Kapitel 3

weitere Eingabemöglichkeiten zur Netzgenerierung

Durch die folgenden Befehle werden keine neuen Knoten oder Elemente generiert, sondern vorhandene verändert oder sonstwie bearbeitet. Einige der Befehle können nur vor, andere nur nach dem Befehl `*END PRE` verwendet werden, bei letzteren ist dies im Text vermerkt. Einzig der Befehl `*RENEW ELSET` kann sowohl vor als auch nach dem Befehl `*END PRE` stehen.

3.1 *TRANSLATE

Mit Hilfe dieses Befehls werden alle in einem vorgegebenen Rechteck enthaltenen Knoten um einen Betrag `dx`, `dy` und `dz` verschoben. Die Eingabe wird folgendermaßen vorgenommen:

```
*TRANSLATE [,DX=dx] [,DY=dy] [,DZ=dz]
POLYGON
x1, y1, z1, x2, y2, z2
```

Diese Eingabe gehört zur Routine, die auch die Knoten für `*SET DEFINE`, Kap. 4.2, herausfindet, daher ist theoretisch auch die Eingabe von `*LINEX`, `LINEY`, `LINEZ` sowie `*POINT` möglich.

3.2 *ROTATE

Mit dem Befehl `*ROTATE` werden alle Knoten um einen vorgegebenen Winkel um eine definierte Achse gedreht. Die Eingabe lautet folgendermaßen:

```
*ROTATE
iaxis, winkel
```

`iaxis` ist die Nummer der Achse (1 → x-Achse, 2 → y-Achse, 3 → z-Achse)

Die Angabe der Rotation erfolgt mit der Eingabe von `winkel` in Grad.

3.3 *SCALE

Eingabe:

```
*SCALE  
factor
```

Der Befehl ***SCALE** darf erst nach dem Kommando ***END PRE** gesetzt werden. Mit dem Kommando können sämtliche Knotenkoordinaten mit einem Faktor multipliziert werden. Außerdem gibt es zwei Besonderheiten:

factor=-1 Es werden alle Koordinaten so skaliert, daß die maximale Ausdehnung in x-Richtung gerade 1 beträgt.

factor=-2 Es werden alle Koordinaten so skaliert, daß die maximale Ausdehnung in y-Richtung gerade 1 beträgt.

3.4 *COCOORD

eliminiert die doppelten Knoten. Anschließend werden die verbleibenden automatisch neu durchnummeriert (Aufruf von ***REORDER**, s.u.)

Es kann auch die Angabe einer Linie oder eines Bereiches angegeben. Hierzu dient in der Folgezeile eine der folgenden Angaben

```
*POLYGON  
x1,y1,z1,x2,y2,z2
```

(siehe ***SET DEFINE**) oder

```
*LINEX | LINEY | LINEZ  
y,z,x1,x2
```

siehe ebenfalls ***SET DEFINE**) oder

```
X=xval  
Y=yval  
Z=zval
```

3.5 *REORDER

Numeriert die Knoten neu (direkt aufsteigend) durch. Der Befehl braucht normalerweise nicht explizit aufgerufen werden, da nach dem Befehl ***END PRE** die Knoten automatisch neu geordnet werden. Außerdem findet nach dem Eliminieren der doppelten Knoten (***COCOORD**, s.o.) ein **REORDER** statt.

3.6 *DELSURF

Dieses Kommando ohne Parameter und Folgezeilen dient der Bestimmung der Kontinuums-elemente entlang der Reißflanken (nur sinnvoll bei scharfer Reißspitze in positiver x-Richtung)

3.7 *MESH2

Bestimmen des Ligamentknoten (erst nach *DNDSURF aufrufbar / nur sinnvoll bei scharfer Reißspitze in positiver x-Richtung).

3.8 *MESHOUT

```
*MESHOUT [,SPARSE][,USER]
```

Titel

Durch das Kommando *MESHOUT werden überhaupt erst Knoten- und Elementinformationen herausgeschrieben. Die anderen Angaben wie Setdefinitionen, Lasten, Randbedingungen und Historydefinitionen sind hiervon nicht betroffen.

In der Folgezeile wird eine Zeichenkette erwartet, der als Titel der Berechnung übernommen und unter dem ABAQUS Schlüsselwort *HEADING herausgeschrieben wird. Wichtig ist, daß das Kommando erst nach Eingabe von *END PRE eingegeben werden darf.

Für ABAQUS ist die Eingabe zusätzlicher Schlüsselwörter möglich:

Durch SPARSE wird bei ABAQUS der sparse solver aktiviert (seit 5.7 default).

Durch den Parameter USER wird ABAQUS bekanntgegeben, daß user defined elements verwendet werden. Es wird eine Ausgabe zur Definition der Elemente forciert, in die der Elementtyp eingesetzt wird, der in *ELTYPE angegeben wurde. Die Anzahl der Knoten der Elemente wird aus dem ersten Element dieses Typs gelesen und ebenfalls eingesetzt.

3.9 *TOLERANCE

Eingabe:

```
*TOLERANCE
```

```
tolerance_factor
```

Mit Hilfe dieses Befehls können alle Knotenkoordinaten, die geringer als eine in der Folgezeile des Befehls angegebene Größe sind, zu exakt 0.0D0 gesetzt werden.

3.10 *MULTIMAT

Mit diesem Kommando lassen sich schon generierte Elemente einem anderem Elementset zuordnen. Die Elemente können mit Hilfe der geometrischen Beschreibungen ARC, POLYGON

30KAPITEL 3. WEITERE EINGABEMÖGLICHKEITEN ZUR NETZGENERIERUNG

und POINT spezifiziert werden. Die Eingabe erfolgt nach folgendem Schema

```
*MULTIMAT, NEW ELSET=number[,OLD ELSET=number]
```

Anschließend können mehrere der folgenden Definitionen hintereinander erfolgen:

POLYGON

```
x1, y1, z1, x2, y2, z2
```

oder

POINT

```
x, y, z
```

oder

ARC

```
xM, yM
```

```
r1, r2, phi1, phi2, z
```

Bei POINT werden alle Elemente bestimmt, die diesen Punkt berühren, bei POLYGON werden die Elemente selektiert, deren Schwerpunkt innerhalb des mit den beiden Punkten aufgespannten Quaders liegen, bei ARC werden diejenigen Elemente selektiert, deren Schwerpunkt innerhalb eines Kreisabschnitts liegen, der mit dem Mittelpunkt x_M/y_M , dem minimalen und maximalen Radius r_1 und r_2 sowie mit Anfangs- und Endwinkel ϕ_1 und ϕ_2 angegeben sind. Die z-Koordinate(n), die man sowohl bei ARC als auch bei POLYGON eingeben muß, wird nicht abgefragt und ist daher ohne Bedeutung.

Statt diesem Kommando sollte das Kommando *RENEW ELSET (s.u.) verwendet werden.

3.11 *RENEW ELSET

Dieses Kommando hat dieselbe Funktion wie *MULTIMAT, ist aber umfangreicher. Die Eingabe erfolgt wie bei der Definition von Elementsets, Kap. 4.2, beschrieben. Es können beliebig viele Definitionen hintereinander folgen. Wurde der Parameter *NEW ELSET eingegeben, so werden alle Elemente diesem Set zugeordnet. Fehlt der Parameter, so wird für jede Definition eine neue Elementsetnummer vergeben. Die Eingabe lautet:

```
*RENEW ELSET [, OLD ELSET=elset_nummer] [,NEW ELSET=elset_nummer]
```

```
[*POLYGON | *POINT | *CIRCLE | *POLYLINE ]
```

(je nach gewählter Option)

...

Kapitel 4

Definition von Element- und Knotensets

Sämtliche definierten Elemente werden schon bei der Generierung zu Elementgruppen zugeordnet, siehe Elementgenerierung. Die Namen lauten

E0000001, E0000002, ...

Außerdem stehen für die Definition von Element- und Knotensets zwei verschiedene Befehle zur Verfügung:

4.1 *ENSET

Eingabe: *ENSET

`nxmin,nxmax,nymin,nymax,nzmin,nzmax`
`elxmin,elxmax,elymin,elymax,elzmin,elzmax`
`ligament,csurface,ctip,zset,cze3d`
`anz.folgezeilen`

In der ersten Zeile können die Knotensets mit den minimalen und maximalen x-, y- und z-Koordinaten definiert werden; eine Zahl `nxmin, ... ≠ 0` bedeutet, daß der jeweilige Set definiert werden soll. Die Namen lauten dann

`NBOUNDX1,NBOUNDX2,NBOUNDY1,NBOUNDY2,NBOUNDZ1,NBOUNDZ2`

In der zweiten Zeile stehen die zu definierenden Elementsets. Die Vorgehensweise ist analog zu den Knotensets. Die Namen enthalten außerdem die Nummer der freien Kante der äußeren Elemente für eine mögliche Definition der Flächenlasten (Druckkräfte auf Elementoberflächen).

`ELSET1_kantenr, ELSET2_kantenr, ELSET3_kantenr, ELSET4_kantenr, ELSET5_kantenr,`
`ELSET6_kantenr`

Mit `ligament=1` werden die mittels des Kommandos *MESH2, ermittelten Ligamentknoten als *NSET LIGASET herausgeschrieben.

Mit `csurface=1` werden die mittels des Kommandos `*DNDSURF` ermittelten Riffnungsknoten als `*NSET CSUFSET` herausgeschrieben.

Mit `ctip=1` wird veranlat, da Rispitzenknoten und Elementsets fr die Bestimmung des J -Integrals herausgeschrieben werden. Diese bekommen bei 2D-Rechnungen den Namen JN01 (Knotenset) bzw. JE01 (Elementset); bei 3D-Modellen werden die Namen durchnummeriert. Allerdings werden mit dieser Methode fr das J -Integral nur die Rispitzenknoten selbst herausgeschrieben. Fr die Definition eines greren Konturbereichs sollte `*SET DEFINE`, Kap. 4.2 verwendet werden.

Mit `zset=1` knnen diejenigen Knoten zu einem Set zusammengefat werden, die die z -Koordinate 0 besitzen (Name ZSET0)

`cze3d` kann die Werte 0, 1 oder 2 annehmen. Mit `cze3d=1` werden bei Verwendung der 2D-Interfacelemente fr ABAQUS mit Hilfe des Kommandos `*COHESIV3` die Ligamentknoten als Knotenset mit dem Namen BOTTOM herausgeschrieben. Durch die Angabe von `cze3d=2` wird zustzlich ein Satz von Zwangsbedingungen definiert, um die Riffnungsknoten in x - und z -Richtung an die Ligamentknoten zu fesseln.

Anschließend knnen noch weitere Knoten- und Elementsets definiert werden durch

- eine Kombination von anderen, bereits definierten Sets,
- direkte Eingabe von Knoten- bzw. Elementnummern,
- Definition von Knoten- /Elementsets durch Eingabe von umgebenden Bereichen

Die Eingabe erfolgt nach den drei ersten Zeilen des Kommandos `*ENSET`. In der ersten Folgezeile steht die Anzahl der noch kommenden Definitionen, danach folgt die Zeile

0, key, name

wobei `key` einen Wert von 1 bis 4 annehmen kann. `name` steht fr den Namen des Knoten- bzw. Elementsets. Es folgen je nach Wert von `key` bestimmte Folgezeilen, die im folgenden angegeben werden:

KEY=1 direkte Eingabe von Knoten bzw. Elementnummern

anzahl, art, generateflag, (nummer_i,i=1,anzahl)

Hierbei bedeutet `art=1` Knotenset, `art=2` weist auf ein Elementset hin. `generateflag=1` bedeutet, da bei der Eingabe das ABAQUS Schlsselwort GENERATE verwendet wird.

KEY=2 Zusammenfassen mehrerer bereits definierter Sets

0,2,setname

art, string

`art=1` \rightarrow Knotenset ; `art=2` \rightarrow elementset. In `string` stehen die zu verwendenden Sets.

KEY=3 Knotensetdefinition

Eingabe analog zum Kommando `*SET DEFINE`, Kap. 4.2 — Es wird die gleiche Routine verwendet!

KEY=4 Elementsetdefinition

C|R
 x0,y0
 phi1,phi2,r1,r2,z1,z2 bzw. x1,x2,y1,y2,z1,

Bei **art=C** wird die Eingabe eines mit Anfangs- und Endwinkel sowie r -radius und z -Koordinate erwartet, bei **art=R** müssen die Eingaben in kartesischen Koordinaten erfolgen.

4.2 *SET DEFINE

Syntax:

*SET DEFINE, NAME=setname [,OPTION=NSET|JSET] [,ELSET=nummer]

Wird der Parameter **OPTION** nicht gesetzt, werden automatisch Elementsets definiert. Die Eingabe von Knotensets erfolgt mit **OPTION=NSET**, die Option **JSET** ist eine Generierungsmöglichkeit für Knotensets speziell bei J -Integral-Berechnungen.

Bei der Eingabe der **ELSET=nummer** wird nach Elementen oder Knoten nur innerhalb dieses Sets gesucht.

4.2.1 Definition von Elementsets

Die Eingabe von Elementsets erfolgt mit Hilfe der folgenden Schlüsselwörter:

Schlüsselwort	Beschreibung
*CIRCLE x0, y0 r1, Theta1, z1, r2, Theta2, z2	r1: minimaler Radius, r2: maximaler Radius; Theta1/Theta2: Winkel (in Grad) am Anfang / Ende; z1/z2: z-Koordinate am Anfang / Ende
*POLYGON x1, y1, z1, x2, y2, z2	Die beiden Punkte definieren die Raumdiagonale eines Würfels. Alle Elemente, deren Schwerpunkt in dem durch (x1/y1/z1) und (x2/y2/z2) eingeschlossenen Würfel liegen, werden im Elementset verwendet.
*POINT x1, y1, z1	Alle Elemente, von denen ein Knoten auf diesem Punkt liegt, werden in den Elementset hineingenommen.
*POLYLINE npoints (xi, yi, zi, i=1,npoints)	Alle Elemente, deren Schwerpunkt innerhalb des mit npoints Punkten definierten Polygons liegt, werden im Elementset definiert. Es können bis zu 20 Punkte definiert werden. Die Polyline muß konvex sein!

4.2.2 Definition von Knotensets

Bei der Definition von Knotensets stehen folgende Schlüsselwörter zur Verfügung:

Schlüsselwort	Beschreibung
*LINEX y, z, x1, x2	y/z: y- und z-Koordinate der Linie; Linie bewegt sich zwischen x1 und x2
*LINEY z, x, y1, y2	siehe LINEX
*LINEZ x, y, z1, z2	siehe LINEX
*POINT x, y, z	Definition durch Koordinateneingabe
*POLYGON x1,y1,z1,x2,y2,z2	Eingabe durch Eckpunkte eines Würfels
*PLANE x1,y1,z1 x2,y2,z2,x3,y3,z3	Hiermit kann eine beliebige schiefe Ebene definiert werden. Mit (x1,y1,z1) wird ein Eckpunkt der Ebene definiert. (x2,y2,z2) sowie (x3,y3,z3) sind Richtungsvektoren in der Ebene. Alle Knoten, die ein positives Skalarprodukt mit beiden Vektoren bilden und in der Ebene liegen, werden gesammelt.
*CIRCLE x0, y0 r1, Theta1, z1, r2, Theta2, z2	r1: minimaler Radius, r2: maximaler Radius; Theta1/Theta2: Winkel (in Grad) am Anfang / Ende; z1/z2: z-Koordinate am Anfang / Ende

4.2.3 Knotenset zur J -Integralberechnung

Bei OPTION=JSET werden Knoten in allen vorhanden (d.h. mit *MESH3D generierten) z-Ebenen innerhalb eines angegebenen Rechtecks der x-y-Ebene gesucht. Die Eingabe erfolgt über:

```
*SET DEFINE, OPTION=JSET
x1, y1, x2, y2
```

Kapitel 5

Materialparameter

Die Eingabe der Materialgesetze für die einzelnen Elementgruppen erfolgt über die Anweisung

***SOLID.**

Es folgen nun hintereinander alle zu definierenden Materialgesetze. Die Eingabe der Materialien ist durch die Eingabe eines anderen Schlüsselwortes, gekennzeichnet durch „*“ in der ersten Spalte, beendet.

Die Ausgabe erfolgt nach dem Schema, daß die erste Eingabe das Material für Elementset 1 (E0000001) darstellt, der zweite Eintrag das Material für Elementset 2, etc.. Die zugehörigen Materialien bekommen den Namen M0001i, wobei i die dreistellige Ziffer der Elementsetnummer ist, zu dem Elementset 1 gehört also das Material M0001001.

Die Eingabe war ursprünglich grundsätzlich so, daß nach dem Schlüsselwort des Materialtyps die Anzahl der einzugebenden Materialparameter in der Folgezeile angegeben wurde, in einer weiteren Folgezeile gefolgt von eben diesen Werten. Neue Implementierungen weichen von dieser Vorgehensweise teilweise ab, und verwenden erforderliche und optionale Parameter hinter dem Schlüsselwort.

Folgende Materialmodelle stehen zur Verfügung:

5.1 Elastisches Material

```
*ELASTIC
2
e-modul, poisson
```

5.2 plastisches Material mit exponentieller Verfestigung

Eingabe (alte Form):

```
*PLASTIC
10
```

e-modul, poisson, sigma0, hardening_exp, real1, RMU, real3, real4
anzahl_stuetzstellen, max_stress

oder (neue Form):

```
*PLASTIC, EMODUL=e-Modul ,NU=poisson, SIG0=sigma0,
NVAL=anzahl_stuetzstellen, EPSMAX=epsmax,HARDENING=hardening_exp
```

Die Variablen `real1`, `real3` und `real4` in der alten Form werden nie verwendet, müssen jedoch aus Kompatibilitätsgründen vorhanden sein.

Mit `RMU=0` wird ein Potenzgesetz für die plastische Verfestigung mit den Parametern `sigma0` (Streckgrenze) und `hardening_exp` (Verfestigungsexponent) generiert. Die Anzahl der Stützstellen und die maximale Spannung, bis zu der die generierte Spannungs-Dehnungskurve gehen soll, werden ebenfalls vorgegeben. Die Ausgabe enthält, wie in ABAQUS üblich, den elastischen und plastischen Anteil getrennt:

```
*ELASTIC
e-modul, poisson
*PLASTIC
sigma0, 0.0
je nach Verfestigung
```

Nach der maximal zu verwendenden Spannung wird noch ein Wert mit der zehnfachen Spannung berechnet und angehängt, um ein ideal-plastisches Materialverhalten bei Erreichen der maximalen Spannung zu verhindern.

Bei `RMU≠0` wird das Materialgesetz `*DRUCKER PRAGER` generiert. Die Ausgabe sieht folgendermaßen aus:

```
*DRUCKER PRAGER, DEPENDENCIES=3
beta, 1.0, beta1
*ELASTIC
e-modul, poisson
*DRUCKER PRAGER HARDENING, TYPE=TENSION
Verfestigungskurve wie beim Potenzgesetz
```

Die neue Form steht nur für das konventionelle plastische Materialverhalten zur Verfügung. Hier werden keine Folgezeilen benötigt, und die Parameter können in beliebiger Reihenfolge eingegeben werden. Dafür sind die o.a. Schlüsselwörter erforderlich. Besitzt der Verfestigungsexponent einen Wert > 98 , so wird ein ideal plastisches Materialgesetz generiert.

5.3 benutzerdefinierte Materialien

Zur Eingabe von benutzerdefinierten Materialmodellen existieren verschiedene Befehle, die auch (teilweise unterschiedliche) Folgezeilen erfordern:

```
*POWER LAW
*GURSON
```

¹Der Winkel berechnet sich aus dem für `RMU` eingegebenen Wert wie folgt: $beta = \arctan(\sqrt{3} \times RMU)$

```
*CRYSTAL PLASTICITY
*USER MAT
```

Das Schlüsselwort `POWER LAW` entspricht nicht einem elastisch-plastischen Materialgesetz, wie man es erwarten könnte, sondern bewirkt für ABAQUS eine Definition des Materialgesetzes in Form einer `USER SUBROUTINE`, die in einer separaten Datei stehen kann. Außerdem kann eine `HOURGLASS`-Stiffness vergeben sowie die Anzahl der abhängigen Variablen eingegeben werden. Auch das Materialmodell `*GURSON` ist nicht auf das Gurson-Modell beschränkt (und bewirkt auch nicht die Verwendung des innerhalb von ABAQUS implementierten Gurson-Modells), sondern stellt ebenfalls eine Eingabehilfe für benutzerdefinierte Materialmodelle überhaupt dar.

```
*GURSON
anzahl_parameter
param_1, ...,param_8
param_9, ...
...
const_1, const_2, ..., const_n
[DEPVAR=ndepvar]
[HOURGLASS]
[USER=user_sub_name]
```

Bei der Eingabe der Parameter ist darauf zu achten, daß acht Werte pro Zeile erwartet werden. Stehen weniger Werte in einer Zeile, werden die restlichen Parameter zu null gesetzt.

5.4 Deformationsplastizität

```
*DEFORMATION PLASTICITY
6
e-modul, poisson, sigma0, hardening_exp, yield_offset, temperature
```

Hiermit ist die Eingabe des gleichlautenden ABAQUS Materialmodells mit den dort zur Verfügung stehenden Parametern möglich.

5.5 Plastizität mit vorgegebener Spannungs-Dehnungskurve

```
*TENSION
2
e-modul, poisson
material_filename
```

Hiermit ist es möglich, für den plastischen Bereich eines Materialgesetzes die Daten aus einer Datei herauszulesen. Der Filename wird nach den elastischen Parametern eingegeben. Auf der Datei stehen nur Wertepaare der wahren Spannung und der wahren plastischen Dehnung (2 Spalten), so wie es in ABAQUS verlangt wird.

5.6 Kohäsivzonenmodell

Hiermit werden die Materialparameter der Kohäsivzonenelemente für die USER ELEMENTS von ABAQUS eingegeben.

Syntax für Kohäsivzonenelemente innerhalb von ABAQUS:

```
*INTERFACE
*ABAQUS, MODEL=[1|2|3]
anzahl_parameter
parameter_1, parameter_2, ..., parameter_n
input file
```

Der MODEL-Parameter nach dem Schlüsselwort ABAQUS definiert das für die Verteilung der Maximalspannung verwendete Potential. Es bedeuten

1. Needleman (kubisches Separationsgesetz, `anzahl_parameter=5`)
2. Rose et. al (exponentielles Separationsgesetz, `anzahl_parameter=5`)
3. Lin (modifiziert) (konstante Separationsspannung, `anzahl_parameter=6[3D]` bzw. `7[2D]`)

Die ersten vier Parameter sind grundsätzlich gleich, nämlich `delta_N`, `delta_T`, `sigma_N` und `sigma_T`. Es folgt bei allen zweidimensionalen Modellen die Definition der Dicke und anschließend bei dem Modell von Lin noch die beiden Formparameter `delta_1` und `delta_2`.

5.7 Zuordnung von Elementgruppen zu Materialien

Der Befehl `*SECTION` dient der Zuordnung von Materialmodellen zu beliebigen Elementgruppen. Es bestehen zwei Möglichkeiten: Zum einen können beliebige Elementsets anderen (beliebigen) Materialnummern zugeordnet werden. Die Elementsetnummer wird dabei intern nicht hochgezählt. Die Eingabe lautet dann:

```
*SECTION
elset, materialset
```

oder es kann das aktuelle (fortlaufende) Elementset einem schon definierten Material zugeordnet werden. Dabei wird dann auch die Elementsetnummer hochgezählt. Die Eingabe hierfür lautet:

```
*SECTION, MATERIAL=oldmaterial
```

Es wird keine weitere Parameterzeile benötigt. `oldmaterial` ist die Nummer des Materials. Sind noch nicht mindestens `oldmaterial` Materialien definiert, erfolgt eine Warnung.

Mit dem Kommando `*SAME SOLID SECTION` ist es möglich, mehreren Elementsets das gleiche Material zuzuordnen. Die Eingabe erfolgt nach dem Muster

```
*SAME SOLID SECTION
*DOMAIN
anzahl
elementset_1 , matset_1,... , elementset_n, matset_n
```

Die Eingabe bewirkt, daß dem Elementset `elementset_i` das Material `matset_i` zugeordnet wird. Dieser Befehl ist allerdings etwas komplizierter anzuwenden als das Kommando `*SECTION`. Er gehört (als einziger Materialbefehl) nicht als Unterkommando zu `*SOLID`, sondern kann an anderer Stelle stehen. Dies ist besonders zu beachten, da durch diesen Befehl der `*SOLID`-Abschnitt abgeschlossen wird, und nachfolgende Materialdefinitionen nicht mehr erkannt werden.

5.8 Eingabe von beliebigen Eingabekarten

```
*COMMOND
string
```

Hierbei wird die folgende Zeile direkt in das Input-File übernommen. Dies kann im Zusammenhang mit dem Unterkommando `*SECTION` der Zuordnung einer Dicke zu 2D plain strain bzw. plain stress Elementen dienen.

Kapitel 6

Randbedingungen

Randbedingungen können in Form von vorgegebenen Verschiebungen sowie Zwangsbedingungen vorgegeben werden

6.1 *BOUNDARY

Dieser Befehl ist eine verbesserte Version des Befehls *BOUND1 (s.u.). Die Eingabe der Randbedingungen wird solange fortgesetzt, bis mit „*“ (nicht gefolgt von einem weiteren *) der nächste Befehl anfängt. Die Eingabe erfolgt mit

```
*BOUNDARY
nsetname1, direction1, factor1
...
nsetname_n, direction_n, factor_n
```

Die Randbedingungen werden über Knotensets definiert, die in der Richtung `direction` gefesselt bzw. mit einer Verschiebung vom Faktor `factor_i` beaufschlagt werden.

Es wird überprüft, daß der Knotenset vorher schon definiert worden ist. Anderenfalls erfolgt eine Warnung und die Randbedingung wird nicht definiert.

Die Eingabe von Lasten erfolgt nicht über dieses Kommando, da die Ausgabe vor den Lastschrittinformationen erfolgt. Daher sind die hier definierten Verschiebungen konstant.

6.2 *BOUND1

Dies ist der alte Befehl zur Eingabe von Randbedingungen, der nach Möglichkeit nicht mehr verwendet werden sollte. Die Eingabe lautet:

```
*BOUND1
anzahl
nsetname1, direction1, factor1
...
nsetname_n, direction_n, factor_n
```

Die Angabe der Randbedingungen erfolgt analog zum Befehl `*BOUNDARY`. Allerdings muß die Anzahl der Folgezeilen vorher durch `anzahl` mitgeteilt werden. Dabei ist `anzahl` die reale Anzahl der folgenden Datensätze + 90!

Dafür besitzt der Befehl `*BOUND1` Zusatzoptionen: Durch `anzahl=1` wird der Knotenset `NBOUNDX1` in x-Richtung gefesselt, bei `anzahl=2` wird `NBOUNDX1` in x- und `NBOUNDY1` in y-Richtung gefesselt, und bei 3 wird `NBOUNDX1` in x- und y-Richtung gefesselt.

Wichtig ist, daß `nsetname` im Format A8 eingelesen wird und daher die Richtung erst nach der 8. Stelle stehen darf!

6.3 *MPCSUB

Hiermit können Zwangsbedingungen (Multipointconstraints) vorgegeben werden. Dies geschieht in der Form, daß alle Knoten eines Knotensets in der mit `dof` angegebenen Richtung dieselbe Größe haben müssen. Die Eingabe erfolgt folgendermaßen:

```
*MPCSUB
anzahlsets
idum, dof, nodesetnumber
...
```

`dof` steht für den Freiheitsgrad, der der Zwangsbedingung unterworfen wird, `nodesetnumber` steht für die fortlaufende Nummer der mit Hilfe von `*ENSET` definierten Knotensets, wobei nur die bereits mit `*ENSET` definierten mitgezählt werden. Die Variable `idum` wird nicht mehr ausgewertet und existiert nur noch aus Kompatibilitätsgründen.

Es werden `anzahlsets` Knotensets definiert, die den Namen `MPCSETi` (*i* von 1 bis `anzahlsets`) erhalten. Alle bis auf den ersten Knoten des mit `*ENSET` definierten Sets sind hierin enthalten. An diesen ersten werden sie gefesselt. Für den Masterknoten selbst wird ebenfalls ein Nodeset namens `MPCMSTRi` definiert.

6.4 *EQUATD

Auch mit `*EQUATD` können Zwangsbedingungen definiert werden. Bei diesem Kommando liegen die betroffenen Knoten in einer horizontalen oder vertikalen Linie, die mit der y- bzw. x- Koordinate beschrieben und einer Anfangs- und Endkoordinate in der jeweils anderen Richtung begrenzt wird. Die Eingabe erfolgt nach folgendem Schema:

```
*EQUATD
anzahlsets
HORIZONTAL          VERTICAL
y_coord, x_begin, x_end    x_coord, y_begin, y_end
...(Wiederholung der zwei Zeilen für weitere Sets)
```

Alle Knoten werden hierbei in der y-Richtung (horizontale Linie) bzw. x-Richtung (vertikale Linie) an den ersten der jeweiligen Linie gefesselt.

Ist `x_begin=x_end` bzw. `y_begin=y_end`, so wird die Linie von $-\infty$ bis $+\infty$ definiert.

Im Gegensatz zu *MPCSUB wird der Knotenset nicht als solcher definiert, sondern die Knoten einzeln in Form von Equations gefesselt.

6.5 *EQUATQ

Mit diesem Kommando werden alle Knoten, die sich auf einer zu definierenden horizontalen oder vertikalen Linie befinden, mit einem Referenzknoten verbunden. Die Eingabe:

```
*EQUATQ, NAME=refnodeset
X1, Y1, Z1
anzahlsets
HORIZONTAL                VERTICAL
y_coord, x_begin, x_end   x_coord, y_begin, y_end
...(Wiederholung der zwei Zeilen für weitere Sets)
```

Es wird ein Knotenset für den Referenzknoten (Koordinaten X1, Y1, Z1) definiert, der den Namen `refnodeset` (default: EQUATQ) bekommt. Alle Knoten der anschließend definierten Linie(n) werden an diesen Knoten in der y-Richtung (horizontale Linie) bzw. x-Richtung (vertikale Linie) gefesselt. Der Unterschied zu *EQUATD besteht also nur darin, daß alle Knoten an einen Extraknoten und nicht an den ersten der Linie angebunden werden.

6.6 *EQUATS

Mit *EQUATS können Zwangsbedingungen für Biegebelastungen eingegeben werden. Es werden daraus Gleichungen generiert, die einer Rotation um ein anzugebendes Zentrum entsprechen. Die Eingabe hat folgendes Aussehen:

```
anzahlsets
string
idof1, idof2, xycoord, factor, [idof3, zequ]
kref, xref, yref
...(Wiederholung der drei Zeilen für weitere Sets)
```

`idof1` ist die Richtung, in denen die gebundenen Knoten die gleiche Koordinate (`xycoord`) besitzen. `idof2` ist dementsprechend die Richtung, entlang die Knoten gleicher `idof1` Koordinate gesucht werden. (Dementsprechend ist `idof3` die Rotationsachse. `idof3` und `zequ` brauchen in einem zweidimensionalen System nicht angegeben zu werden). `factor` beschreibt die Rotation und sollte m.E. auf -1 stehen, um vernünftige Werte zu liefern. Die Eingabe des Referenzknoten `krefs` wird nicht ausgewertet. Der Referenzknoten bekommt eine neue Nummer, die größer ist als die höchste bisher vergebene. Ich bin der Meinung, daß diese Form der Zwangsbedingung nur für Knoten entlang einer Linie gleicher y-Koordinate funktioniert (dies entspricht der Definition von „horizontal“ bei dem Kommando *EQUATD).

6.7 *PERIOD

Es existiert die Möglichkeit, periodische Randbedingungen entlang der globalen Koordinatenachsen automatisch vom Programm generieren zu lassen. Periodische Randbedingungen werden mit dem Befehl `*PERIOD` erzeugt. Anschließend werden mit dem Befehl `*PRE BOUNDARY` die Randbedingungen eingegeben. Die darauf folgende Eingabe bezieht sich zunächst auf globale Systemrandbedingungen und anschließend auf die Flächen, die mit periodischen Randbedingungen belegt werden sollen. Die grundsätzliche Eingabe sieht folgendermaßen aus:

```
*PERIOD [,ORDER3D]
PRE BOUNDARY
anzahl_fixnodes
DOF, FIXNODEi
...
anzahl_period_flaechen
DOF, FLAECHEi_1, FLAECHEi_2
...
```

Durch den optionalen Parameter `ORDER3D` wird angezeigt, daß es sich um eine dreidimensionale Struktur handelt und daher die Zwangsbedingungen in allen drei Koordinatenrichtungen generiert werden müssen. Der Befehl `*PRE BOUNDARY` muß direkt anschließend an `*PERIOD` erfolgen (Kommentarzeilen dazwischen sind erlaubt).

Mit `anzahl_fixnodes` wird die Anzahl der Systemrandbedingungen angegeben, anschließend folgen entsprechend viele Zeilen mit der Eingabe der gefesselten Knoten (analog zur Eingabe des Befehls `*BOUNDARY`).

Mit `anzahl_period_flaechen` wird die Anzahl der mit periodischen Randbedingungen belegten Flächen angegeben. Es folgen entsprechend viele Zeilen mit Knotensets, die die betreffenden Flächen enthalten (oftmals sind dies die durch `*ENSET` definierten Flächen mit minimalen bzw. maximalen Koordinaten, z.B. `NBOUNDX1`, `NBOUNDX2`).

Dabei ist `DOF` die Richtung der Flächennormale und `FLAECHEi_1` und `FLAECHEi_2` sind die Knotensätze für die Vorder- bzw. Rückfläche. Die Zwangsbedingungen zwischen den beiden Flächen und dem Referenzknoten werden immer in allen Richtungen generiert. Wenn mit `*PERIOD,ORDER3D` periodische Randbedingungen aufgebracht werden, werden automatisch `anzahl_period_flaechen` Referenzknoten mit dem Namen `EQUATP01`, ..., definiert. Durch Verschiebung dieser Knoten kann die periodische Verformung aufgebracht werden.

Beispiel:

```
*PERIOD,ORDER3D
PRE BOUNDARY
3
1, FIXNODE1
2, FIXNODE1
3, FIXNODE1
3
3, NBOUNDZ1, NBOUNDZ2
1, NBOUNDX1, NBOUNDX2
```

```
2, NBOUNDY1, NBOUNDY2
BOUNDARY
FIXNODE1, 1, 0.0
FIXNODE1, 2, 0.0
FIXNODE1, 3, 0.0
EQUATP01, 2, 0.0
EQUATP02, 3, 0.0
EQUATP03, 1, 0.0
```

In diesem Beispiel werden drei Flächenpaare mit periodischen Randbedingungen belegt, daher werden auch drei Referenzknoten EQUATP01, EQUATP02 und EQUATP03 generiert. Die Verschiebung des Knotens EQUATP02 in dem oben angegebenen Beispiel bewirkt beispielsweise die periodische Verschiebung der Flächen NBOUNDX1 und NBOUNDX2 gegeneinander. Als Systemrandbedingung wird nur ein einziger Knoten in allen drei Richtungen gefesselt. Diese Fesselung muß in den Randbedingungen noch einmal definiert werden! Die Rotation um den Knoten FIXNODE1 wird durch die Fesselung der generierten Referenzknoten unterbunden.

Anmerkung: Periodische Randbedingungen können ziemlich kompliziert werden. Aus diesem Grund empfiehlt es sich, wie in dem Beispiel immer nur einen einzigen Knoten (im Beispiel FIXNODE1) zu fesseln, so daß das Modell um diesem Knoten rotieren kann. Alle anderen Fesselungen sollte man auf die Referenzknoten aufbringen. Bei anderen Fesselungen besteht die Gefahr, daß man unsinnige Randbedingungen aufbringt.

Kapitel 7

Lastschrittdaten

Die Lastschrittdaten beginnen mit dem Schlüsselwort

```
*HISTORY
```

Innerhalb des History-Blocks, der durch das Schlüsselwort `*END MODE` (Kap. 7.7) beendet wird, sind wiederum verschiedene Schlüsselwörter zulässig. Dies sind `LOAD`, `PRINT`, `FILE`, `J-INTEGRAL`, `COMMAND`, `MODE COMMAND` und `STEP`.

7.1 Belastungsinformation

Die Lastdaten werden nach dem `LOAD`-Befehl eingegeben. Es können in der Folgezeile die Parameter `BOUNDARY`, `CLOAD`, `DLOAD`, `BLFB` sowie `BLF` stehen. In der darauffolgenden Zeile stehen die Anzahl der zu der Art gehörigen Lasten. Anschließend stehen bei `CLOAD`, `DLOAD` und `BOUNDARY` zunächst die Richtung und der Node- bzw. Elementset und nachfolgend der Lastfaktor in den beiden Folgezeilen. Existieren verschiedene Lasten, so müssen die einzelnen Lastarten jeweils mit dem vorangestellten Schlüsselwort `LOAD` aufgeführt werden. Der Block hat beispielsweise folgendes Aussehen:

```
LOAD
BOUNDARY | CLOAD | DLOAD | BLFB | BLF
anzahl_sets
direction_1, node-/elset_1
factor_1
...
```

7.2 Ausgabe (.dat-Datei)

`PRINT` steuert die Ausgabe auf das DAT-File. Es kann die Ausgabe für Knoten- und Elementsets spezifiziert werden. Bei der Option `NOPRINT` kann die gesamte Ausgabe auf die .dat-Datei unterdrückt werden. Andernfalls folgt nach dem Schlüsselwort die Anzahl der Anweisungsblöcke, jeweils bestehend aus drei oder mehr Zeilen.

PRINT NOPRINT

oder

```
PRINT
anzahl_bloecke
frequency, type, node-/elementset
anzahl_variablenzeilen
variablenzeile1
(evtl. Folgezeilen der Variablenliste)
```

Sollen Ergebnisse ausgegeben werden, so wird mit `type` angegeben, ob knotenorientierte (Wert 1) oder elementorientierte Ergebnisse (Wert 2) geschrieben werden sollen.

7.3 Ergebnisausgabe (.fil-Datei)

FILE steuert die Ausgabe auf die FIL-Datei. Die weitere Eingabe verläuft analog zur Ausgabe auf die DAT Datei, mit dem Unterschied, daß der Parameter `type` die folgenden Werte annehmen kann:

type	Element/Knotenorientiert	Besonderheit
201	Element	mit Angabe von ELSET
202	Element	mit ELSET, POSITION=NODES
203	Element	mit ELSET, POSITION=AVERAGED AT NODES
100	Knoten	mit NSET
1100	Knoten	ohne NSET

7.4 Bestimmung des J -Integrals

```
J-INTEGRAL
freq, contours, symm_flag, normal_x, normal_y, normal_z
```

Durch dieses Kommando wird die Ausgabe des Contourintegrals angefordert. Die Anzahl der Elementschichten, an denen es berechnet werden soll, wird mit `contours` angegeben, `freq` gibt die Frequenz an, mit der es auf die Ergebnisdateien herausgeschrieben werden soll, und mit `symm_flag=1` kann angegeben werden, daß es sich um eine symmetrische Struktur handelt. (Dieses Flag darf nur die Werte 1 oder 2 annehmen, da sonst das Programm einem Adressierungsfehler unterliegt!) Die `normal_x`, `-y`, `-z` Komponenten geben die Reißflächennormale an. Die Reißspitzenknoten, für die das J -Integral berechnet wird, müssen im Knotenset JN01 (und folgende) stehen. Sie können leicht mit `*SET DEFINE, OPTION=JSET` generiert werden.

7.5 Direkte Ausgabe von Steuerkarten in die Input-Datei

Zwei Kommandos dienen der direkte Übernahme in die Input-Datei, ohne vom Preprozessor interpretiert zu werden: `COMMAND` und `MODE COMMAND`. Der Unterschied besteht darin,

daß Zeilen, die nach dem Kommando `COMMAND` folgen, in der Input-Datei innerhalb des `STEP`-Blocks stehen und somit zu den Lastschrittdateien gehören, während die Zeilen, die dem Kommando `MODE COMMAND` folgen, noch vor dem `STEP`-Block erscheinen.

```
COMMAND | MODE COMMAND
number_command
zeile1
...
```

7.6 Lastschrittdefinition

Der Parameter `STEP` steuert die Berechnung selbst. Es folgen 3 Zeilen mit Eingabeparametern. Die Eingabe lautet

```
STEP
NO|YES NO|YES NO|YES
anzahl_stepzeilen, p0, pN, iriks [, maxload, inode, idof, maxdisp]
inc, nlgeom, kstep, incre
```

Die erste Zeile steuert den ABAQUS-Befehl `PREPRINT`, mit dem eine Modellausgabe auf das `DAT` file unterdrückt werden kann. Das erste Flag bezieht sich auf die Ausgabe der Input-Datei (`ECHO`), das zweite auf die Ausgabe der Lastschrittinformation (`HISTORY`) und das dritte auf die Information über die Modelldefinition (`MODEL`).

In der zweiten Zeile stehen mind. 4 Werte. `anzahl_stepzeilen` gibt die Anzahl der Zeilen nach dem Kommando `*END MODE`, siehe Kap. 7.7, an. In jeder Zeile können mehrere gleichartige Lastschritte auf einmal definiert werden, daher ergibt sich die gesamte Anzahl an Lastschritten erst dort. Der Parameter `iriks` steuert die ABAQUS Berechnungsart `IRIKS` (load displacement control). Sie wird mit `iriks>0` eingeschaltet. In diesem Falle müssen nach dem Parameter `iriks` noch 4 weitere Werte wie in ABAQUS gefordert zur Steuerung des Verfahrens angegeben werden. Der Parameter `p0` wird nur im Zusammenhang mit der `MBL`-Formulierung benötigt (im Zusammenhang mit dem Parameter `T-stress`, der nach `*END MODE` eingegeben wird); `pN` wird nicht mehr verwendet.

Die dritte Zeile dient weiteren Berechnungsparametern. Die Variablen sind 4 Integerwerte. `inc` gibt die max. zulässige Anzahl an Inkrementen an, `nlgeom=1` bedeutet, daß die Berechnung geometrisch nichtlinear durchgeführt wird, und `kstep` sowie `incre` steuern ein evtl. Lesen vom `RESTART` file zur Fortsetzung einer vorangegangenen Rechnung. Es wird ein `*RESTART, READ, STEP=kstep, INC=incre` vor dem `STEP`-Block durch die beiden letztgenannten Parameter generiert. Sonderfall: mit `kstep=-1` wird `*RESTART, READ, END STEP` eingefügt.

7.7 Parameter für die einzelnen Lastschritte

Der gesamte Historyblock wird mit dem Kommando `*END MODE` beendet. Nach dem Kommando `*END MODE` folgt ein Titel für die Lastschritte, anschließend folgt die Eingabe der Lastschrittinformationen. Die Anzahl der Zeilen, die diesem Kommando folgen, wurde mit dem Unterkommando `STEP` definiert. Die Eingabe dieses Blocks lautet:

```
*END MODE
steptitle
anzahl_steps, load_factor, therm.stress, T-stress, 1st_increment, max_increment,
msg_freq, IPLOTK, file_freq, LSEARCH, restart_freq, DISC, inc_time ...
```

(Achtung: Alle Parameter von `anzahl_steps` bis `inc_time` müssen in einer Zeile stehen!)

Der Parameter `anzahl_steps` gibt an, wie oft der in dieser Zeile definierte Lastschritt nacheinander generiert werden soll. Der anschließende Parameter `load_factor` wird von Lastschritt zu Lastschritt aufaddiert, so daß eine Zeile, beginnend mit 3, 2.0, ... dazu führt, daß drei Lastschritte mit einem Lastfaktor von 2.0 im ersten, 4.0 im zweiten und 6.0 im dritten Schritt ausgegeben werden. Wenn noch eine weitere Zeile beispielsweise mit 1, 1.0, ... folgt, so besitzt der vierte herausgeschriebene Lastschritt den Lastfaktor 7.0.

Die Lastfaktoren `therm.stress` (Temperaturlast) und `T-stress` (Constraintlast) beziehen sich auf das Verhältnis zu `sigma0`, definiert in Kap. 5.2.

Die Parameter `1st_increment` und `max_increment` beziehen sich auf die vorgegebenen Schrittweiten innerhalb des statischen Zeitschritts. Ein Analysestep ist grundsätzlich eine Zeiteinheit lang (Der erste Zeitschritt geht also bis $t=1$). `1st_increment` ist die Schrittweite des ersten Inkrements. Die maximale Schrittweite wird durch `max_increment` vorgegeben.

`restart_freq` definiert, ob und wie oft das Restart File geschrieben bzw. gelesen wird. Die Eingabe erfolgt durch `KRESTART=freq*100+overlay`. Mit `overlay=1` wird die `OVERLAY` Option aktiviert.

Mit `LSEARCH≠0` wird der Linesearch-Algorithmus eingeschaltet. Der Wert von `LSEARCH` hat allerdings keinen Einfluß auf die Ausgabe. Mit `DISC≠0` wird das ABAQUS Schlüsselwort `ANALYSIS=DISCONTINUOUS` definiert. Mit `inc_time` kann auf `CONTROLS, PARAMETER=TIME INCREMENTATION` Einfluß genommen werden. Die Zahl bedeutet `inc_time=100*I0 + IR`. `I0` ist die Anzahl der Iterationen, nach der erstmals ein Divergenzcheck durchgeführt wird, nach `IR` Iterationen wird geprüft, ob die Konvergenzgeschwindigkeit gering ist. Die maximale Anzahl iterationen wird auf `IMAX=MAX(IR, I0, 15)` gesetzt. Die Anzahl der Iterationen, ab der die alternative Konvergenzgrenze `Rp` statt `Rn` verwendet wird, beträgt ebenfalls `IMAX`.

Der Parameter `msg_freq` steuert die Häufigkeit der Ausgabe auf das `.msg file`. Mit `file_freq≤0` kann die Ausgabe auf die `.fil`-Datei für knoten- und elementorientierte Ergebnisse unterbunden werden. Der Parameter `IPLOTK` hat keine Bedeutung mehr.

Kapitel 8

weitere Befehle

8.1 *DRAWMESH

```
*DRAWMESH [,SCALE=scale] [,CX=cent_x=] [,CY=cent_y] [,PS=filename] [,MPL=filename]
```

zeichnet das Elementnetz. Die Parameter `SCALE` und `CX`, `CY`, `MPL` und `PS` sind optional. `CX` sowie `CY` geben die x- bzw. y-Koordinate des Mittelpunktes der Zeichnung in Weltkoordinaten an. Die Eingabe der Parameter `MPL` und `PS` dient der Ausgabe auf ein ABAQUS Neutralfile bzw. EPS File. Die Eingabe des Filenamens sollte ohne Endung `.ps` bzw. `.mpl` erfolgen, da diese Endung automatisch angehängt wird.

8.2 *DRAW FREE EDGE

Mit diesen Befehl werden die freien Kanten des Elementnetzes gezeichnet. Dies dient der Überprüfung, ob die einzelnen Regionen des Netzes miteinander verbunden sind, oder ob noch doppelte Knoten eliminiert werden müssen. Die Parameter `SCALE` und `CENTER` sind wiederum optional. Eine Ausgabe im Postscript- oder MPL-Format ist nicht möglich.

Die Befehle `*DRAWMESH` und `*DRAW FREE EDGE` können sowohl vor als auch nach dem Befehl `*END PRE` gebraucht werden!

8.3 *MODE COMMAND

Mit dem Befehl `*MODE COMMAND` können Kommandos ohne weitere Interpretation in das Ausgabefile übernommen werden. Die Verwendung des Befehls ist nur nach dem Befehl `*END PRE` möglich.

```
*MODE COMMAND
anzahl_zeilen
zeile_1
...
```

8.4 *END POST

Die gesamte Interpretation der Eingabedatei wird mit

***END POST**

beendet. Alle folgenden Zeilen werden nicht gelesen. Das Kommando enthält keine weiteren Parameter.

Schlüsselwörter

Das Programm FEMESH kennt eine große Anzahl von Schlüsselwörtern. Einige von diesen werden im vorliegenden Handbuch beschrieben. Von den nicht beschriebenen sind einige eindeutig nicht mehr sinnvoll, bei anderen jedoch ist die Funktion nicht ganz klar. Im folgenden werden alle Parameter aufgezählt. Die Kommandos, für die eine Beschreibung existiert, sind fett, die überflüssigen Kommandos sind kursiv gedruckt. Die Beschreibung der kursiv gedruckten, aber dennoch mit einem Häkchen versehenen Kommandos sind inzwischen wieder aus dem Handbuch entfernt worden.

Der erste Bereich der Modellerstellung umfaßt folgende Schlüsselwörter:

<i>*ARRANGE ELSET</i>	<i>*CZM3D</i>	*MESH3D
*BLFMESH1	*CZDS	<i>*MESH3DNEW</i>
*BLFMESH2	*DELSURF	*MESHVL
*BLFMESH3	<i>*DNDSURF</i>	*MULTIMAT
*BLFMESH4	*DRAW FREE EDGE	<i>*NO OUTPUT</i>
*CFRONT	*DRAWMESH	<i>*ORIGIN</i>
*CGROWTH1	*DWNMESH	*RCAEDS
*CGROWTH2	*ELTYPE	*RENEW ELSET
<i>*CGROWTH3</i>	*END PRE	*REORDER
*CGROWTH4	<i>*EXT_3D1</i>	*RIGMESH
<i>*CLOSE WI</i>	<i>*EXT_3D2</i>	*ROTATE
*CNV2TET	<i>*EXT_3D3</i>	<i>*ROTAT2</i>
*CNV2TRI	*EXT_3D4	<i>*ROTAT_3D</i>
*COCOORD	*FBLF ME	<i>*SHALLOW1</i>
*COHESIV1	*FLIP	<i>*SHALLOW2</i>
<i>*COHESIV2</i>	*HBLF ME	*SIDE GROOVE
*COHESIV3	*IPLMESH	<i>*SURFACE ELSET</i>
<i>*COHESIV4</i>	<i>*IRTMESH</i>	*SYSMESH
*COORDINATE	*LEFMESH	<i>*TITLE</i>
*COPY	*MAPE2D	*TOLER
*CRACK2	*MESH 3 NODE	*TRANSLATE
*CRACKTIP1	*MESH 9 NODE	*UPSMESH
<i>*CZM2DEL1</i>	<i>*MESH1</i>	
<i>*CZM2DEL2</i>	*MESH2	

Der erste Bereich wird mit dem Schlüsselwort ***END PRE** beendet. Es folgt die Eingabe der sog. *history information*. Hier werden folgende Schlüsselwörter erkannt (auch hier sind wieder die fett gedruckten Kommandos im Handbuch zu finden):

*BOUND1	*ENSET	*PERIOD
*BOUNDARY	*EQUATD	*SAME SOLID SECTI-
*CAEDS	*EQUATQ	ON
*CONTACT	*EQUATS	*SCALE
*CRACKED	*HISTORY	*SET DEFINE
*DRAW FREE EDGE	*INTER USER	*SOLID
*DRAWMESH	*MESHOUT	<i>*WAVEFRONT</i>
*DSLNE	*MODE COMMAND	
*END POST	*MPCSUB	

Dieser Bereich und damit die gesamte Eingabe wird mit dem Befehl ***END POST** beendet.

Neue Features in Version 1.1

1. Es existiert die neuen Kommandos `*CZDS` und `*CZDOMAIN`, welche die Kommandos `*COHESIV2` und `*COHESIV4` ersetzen. Diese Kommandos sind für die Eingabe von Kohäsivelementen als USER ELEMENT in ABAQUS geeignet.
2. Es existiert ein weiteres Kommando für die Generierung von Kohäsivzonenelementen, genannt `*CZDOMAIN`, bei dem alle Elementkanten innerhalb einer ganzen Region durch derartige Elemente vernetzt werden.
3. Zur Unterstützung der Kohäsivelemente von ABAQUS sind auch die Befehle `*MESHOUT` und `*MATERIAL` (suboption `*INTERFACE`) modifiziert worden.
4. Die Eingabemöglichkeiten für die Definition von Elementsets sind erweitert worden.
5. Die Eingabe von `*TRANSLATE` hat sich geändert
6. Es ist möglich, bei dem Kommando `*COCOORD` einen Bereich oder eine Linie anzugeben. Es werden dann nur die Knoten in diesem Bereich überprüft.
7. Das Kommando `*FLIP` nimmt inzwischen auch das Schlüsselwort `REMOVE` an. Hiermit können die doppelten Knoten sofort bei der Generierung des gespiegelten Netzes gelöscht werden.
8. Die Eingabe ist etwas komfortabler geworden.
 - Bei der Eingabe von Koordinaten im Kommando `*COORDINATES`, `*MAPE2D` (Unterkommando `*COORDINATES`) und `*BOUNDRAY` werden Kommata und Leerzeichen als Trennzeichen gewertet.
 - Folgen zwei Kommata aufeinander (evtl. mit Leerzeichen dazwischen, so wird der Wert 0 bzw. 0.0 hierfür angenommen.
 - Stehen in einer Zeile weniger Zahlen als erwartet, so werden ebenfalls alle fehlenden Zahlen als 0 bzw. 0.0 vom Programm eingesetzt.
9. Einige Kommandos werden als obsolet gekennzeichnet. Diese werden in kommenden Versionen evtl. entfernt oder verändert. Im Einzelnen sind dies
 - `*CRACKTIP1`
 - `*BLFMESH3`
 - `*CGROWTH3`
 - `*SOLID` (Option `*PLASTIC`) wird aufgeteilt in `*PLASTIC` und `*DRUCKER PRAGER`

- *COHESIVE2, *COHESIVE4, *CZM2DEL2, *CZM3D
- *ENSET, Anschlußdefinitionen mit KEY=3 und KEY=4 (werden entfernt)
- *MULTIMAT, wird ersetzt durch den erweiterten Befehl *RENEW ELSET