KfK 3958
Oktober 1985

ERATO — Ein Programm zur Berechnung induzierter Wirbelströme in dreidimensionalen leitenden Strukturen

J. Benner
Institut für Reaktorentwicklung
Projekt Kernfusion

Kernforschungszentrum Karlsruhe

KERNFORSCHUNGSZENTRUM KARLSRUHE
Institut für Reaktorentwicklung
Projekt Kernfusion

KfK 3958

ERATO - Ein Programm zur Berechnung induzierter Wirbelströme in dreidimensionalen leitenden Strukturen

J. Benner

Kernforschungszentrum Karlsruhe G.m.b.H, Karlsruhe

Kernforschungszentrum Karlsruhe GmbH
ISSN 0303-4003

ZUSAMMENFASSUNG

Das Programmsystem ERATO dient dazu, induzierte Wirbelströme in dreidimensionalen leitenden Strukturen und deren sekundäres Magnetfeld zu bestimmen. Es handelt sich bei diesem Programm um eine Weiterentwicklung des am IPP-Garching entwickelten FEDIFF-Codes. Zur Berechnung wird die Finite-Elemente-Netzwerkmethode (FEN) benutzt, wobei die leitende Struktur durch ein äquivalentes Netzwerk ersetzt wird.

Im ERATO-System sind für die Erzeugung der Finite-Elemente Einteilung, die eigentliche Wirbelstromanalyse, und die abschließende Auswertung je ein separates Programm vorgesehen. Dadurch kann erreicht werden, daß das zentrale Analyseprogramm vollständig geometrieunabhängig ist. Zur Elementeinteilung stehen zwei sog. Preprozessoren zur Verfügung, mit denen ein Torussegment und eine ebene, rechteckige Scheibe behandelt werden können. Zur abschließenden Auswertung dienen Postprozessoren, damit können die Wirbelstromverteilungen in Form übersichtlicher Listen gedruckt und mittels Computer-Graphik gezeichnet werden.

Im Bericht wird die Theorie der FEN-Methode vorgestellt, der Aufbau und die Benutzung der verschiedenen Einzelprogramme (Preprozessoren, Analyseprogramm, Postprozessoren, Hilfsprogramme) beschrieben, und zwei Anwendungsbeispiele diskutiert.

ERATO - A COMPUTER PROGRAM FOR THE CALCULATION OF INDUCED EDDY-CURRENTS IN THREE-DIMENSIONAL CONDUCTIVE STRUCTURES

ABSTRACT

The computer code ERATO is used for the calculation of eddy-currents in three-dimensinal conductive structures and their secondary magnetic field. ERATO is a revised version of the code FEDIFF, developed at IPP Garching. For the calculation the Finite-Element-Network (FEN) method is used, where the structure is simulated by an equivalent electric network.

In the ERATO-code, the calculation of the finite-element discretization, the eddy-current analysis, and the final evaluation of the results are done in separate programs. So the eddy-current analysis as the central step is perfectly independent of a special geometry. For the finite-element discretization there are two so called preprocessors, which treat a torus-segment and a rectangular, flat plate. For the final evaluation postprocessors are used, by which the current-distributions can be printed and plotted.

In the report, the theoretical foundation of the FEN-Method is discussed, the structure and the application of the programs (preprocessors, analysis-program, postprocessors, supporting programs) are shown, and two examples for calculations are presented.

INHALTSVERZEICHNIS

NOMENKLATUR

Allgemeine Symbolik

[1]	Literatur Referenz
(10)	Gleichungsnummer
c, α	Beliebige Skalare
$\underline{u}$	Beliebiger Zeilenvektor
u_i	Komponente eines Vektors
$\underline{\underline{A}}$	Beliebige Matrix
$\underline{u}^T$, $\underline{\underline{A}}^T$	Transponierter Vektor bzw. Matrix
$\lvert\underline{u}\rvert$	Absolutbetrag des Vektors $\underline{u}$
$\underline{a} \cdot \underline{b}$	Skalarprodukt zweier Vektoren
sign(x)	Signum-Funktion; $\text{sign}(x) = \begin{cases} 1 & \text{für } x > 0 \\ 0 & \text{für } x = 0 \\ -1 & \text{für } x < 0 \end{cases}$
diag($\underline{u}$)	Aus Vektor $\underline{u}$ aufgebaute Diagonalmatrix
Σ	Summenzeichen
ε { }	Element einer Menge
(x,y,z)	Kartesische Koordinaten
(r,ϕ,z)	Zylinderkoordinaten
t	Zeitkoordinate
$\dot{u}$	Zeitableitung von u
div $\underline{u}$	Divergenz des Vektors $\underline{u}$
∇ a	Gradient des Skalares a
$\Delta = \nabla^2$	Laplace-Operator

Bezeichnungen spezieller Variablen und Konstanten

$\underline{A}$	Vektorpotential
$\underline{B}$	Magnetische Induktion
b	Elementbreite
C	Flächengröße
$\underline{c}$	Mittelpunkt eines externen Stromkreises
$\underline{D}$	Dielektrische Verschiebung
d	Elementdicke
δ	Dirac-Funktion
$\underline{E}$	Elektrische Feldstärke
F	Finites Element
$\underline{H}$	Magnetische Feldstärke
$\underline{\underline{H}}$	Maschen-Inzidenzmatrix

I	Stromstärke
$\underline{i}$	Stromdichte
K	Anzahl der Knoten
$\underline{\underline{L}}$	Induktivitätsmatrizen
ℓ	Elementlänge
M	Masche, Anzahl der Maschen
μ_0	Permeabilitätskonstante
N	Anzahl der Elemente
$\underline{n}$	Normale eines externen Stromkreises
$\underline{P}$	Eckpunkte der Elemente
$\underline{Q}$	Seitenmittelpunkte der Elemente
R	Ohmscher Widerstand
r	Radius eines externen Stromkreises
ρ	Raumladungsdichte
S	Zweig
σ	Leitfähigkeit
U	Skalares Potential
V	Volumenelement

Typen von Fortran-Programmvariablen

I, I(n), I(n,m)	- INTEGER-Wert, bzw. ein- oder zweidimensionales INTEGER-Feld
R, R(n), R(n,m)	- REAL-Wert, bzw. entsprechendes Feld
C*k, C(n)*k	- CHARACTER der Länge k, bzw. entsprechendes Feld
L, L(n)	- LOGICAL-Wert, bzw. entsprechendes Feld

Vorbemerkung

Der hier beschriebene Code ERATO wurde aus dem im IPP Garching von Herrn Dr. H. Preis erstellten Code FEDIFF entwickelt. Die Übernahme von FEDIFF geschah im Rahmen der Kooperation zwischen KfK und IPP auf dem Gebiet der Fusionsreaktorentwicklung.

Der Autor hatte Gelegenheit, sich am IPP unter Anleitung von Herrn Dr. Preis in FEDIFF einzuarbeiten, und möchte H. Preis für die Unterstützung und Beratung bei der Übernahme und Weiterentwicklung seinen Dank aussprechen.

1. Einleitung

Es ist ein wesentliches Merkmal von Fusionsmaschinen, die auf dem TOKAMAK - Prinzip aufbauen, daß neben stationären Magnetfeldern auch zeitlich veränderliche Felder auftreten, die leitende Strukturen durchfluten. Dabei werden in diesen Strukturen elektrische Ströme, die sog. Wirbelströme, induziert. Für die Auslegung einer Fusionsanlage ist die genaue Kenntnis dieser Wirbelströme von großer Bedeutung.

Man kann die Auswirkung der induzierten Ströme auf die Fusionsanlage in drei Punkten zusammenfassen:

- Die Ströme bewirken eine Ohmsche Erwärmung der sie tragenden Strukturen, die bei der Auslegung der Kühlungseinrichtungen berücksichtigt werden muß.

- Die stromdurchflossenen Strukturen werden gleichzeitig auch noch von hohen statischen Magnetfeldern durchsetzt. Die dadurch hervorgerufenen Lorentzkräfte führen zu erheblichen mechanischen Belastungen, die bei schnell veränderlichen Vorgängen (z.B. Plasmadisruption) auslegungsbestimmend sind.

- Die Wirbelströme beeinflußen die Stabilität des Plasmas. So braucht z.B. das zur Plasmakontrolle eingesetzte Feld der Poloidalfeldspulen wegen des entgegengesetzt wirkenden Feldes der Wirbelströme eine gewisse Zeit, bis es in voller Stärke das Plasma erreicht hat. Genauso verändern die Wirbelstromfelder auch das zum Plasmaeinschluß benutzte Magnetfeld, sie können dadurch eventuell die Gleichgewichtskonfiguration zerstören.

Vor allem bei den zukünftigen großen Fusionstestanlagen, die mit einem massiven Brutblanket ausgerüstet sein werden, gewinnt die Frage der Wirbelstromanalyse große Bedeutung. Da dies Problem analytisch nicht und experimentell nur mit sehr starken Vereinfachungen zu behandeln ist, wird weltweit sehr intensiv an der numerischen Wirbelstromanalyse gearbeitet. Die meisten der heute verfügbaren Codes setzen allerdings zeitlich periodische Magnetfeldänderungen voraus und sind deshalb bei den typischen Wirbelstromproblemen in Fusionsreaktoren nicht anwendbar.

Von den wenigen Rechenprogrammen, die beliebige transiente Vorgänge behandeln können, benutzen die meisten die Finite-Elemente-Netzwerk- (FEN) Methode. Die wichtigsten dieser Rechenprogramme sind:

- FEDIFF (IPP-Garching [1 - 3]), benutzt bei der Auslegung von ASDEX, ASDEX-Upgrade, und (zeitweilig) TFTR;

- SPARK (PPPL [4 - 6]), benutzt zur Auslegung des TFTR;

- EDDYNET (ANL [7, 8]);

- Ein japanischer Code [9], benutzt bei Rechnungen zu INTOR.

Teilweise sind auch noch andere Berechnungsverfahren im Einsatz, wie bei NLMMAP (MIT [10]) Finite-Elemente oder COMPELL (Universität Genua [11]) Finite Differenzen. Diese Verfahren haben aber im Vergleich zur FEN - Methode den Nachteil, daß neben den stromführenden Strukturen auch das Vakuum diskretisiert werden muß, was bei 3D Problemen zu einem sehr hohen numerischen Aufwand führt. Die oben genannten Codes sind deshalb auch nur auf 2D Probleme anwendbar.

Unter den aufgeführten FEN-Codes zeichnet sich FEDIFF u.a. dadurch aus, daß nur mit ihm Wirbelstromprobleme in wirklich "dicken" Strukturen behandelt werden können (s. Kap.2). Dieser Code wurde deshalb im März 1984 vom IPP Garching übernommen und an den Rechenanlagen des KfK (Siemens 7890, CYBER 205) installiert und weiterentwickelt. Der neue Code ERATO unterscheidet sich von seinem Vorgänger in einer Vielzahl DV-technischer Details, wodurch Benutzerfreundlichkeit, Flexibilität und Anwendungsbereich des Rechenprogrammes verbessert wurden.

Im Kap. 2 werden die theoretischen Grundlagen der FEN - Methode, wie sie für die Codes FEDIFF und ERATO benötigt werden, entwickelt. Insbesondere werden die Netzwerks-Differentialgleichungen aus den Maxwell-Gleichungen abgeleitet.

Wesentliches Ziel bei der Entwicklung des ERATO-Codes ist es gewesen, eine weitestgehende Unabhängigkeit von einer speziellen Geometrie zu erreichen. Dazu wird der gesamte Code modularisiert und aufgespalten in einen Preprozessorteil, das eigentliche Wirbelstrom-Berechnungsprogramm, und einen Postprozessorteil. Die globale Struktur des Codes wird in Kap. 3 genauer erklärt.

Aufgabe des problemabhängigen Preprozessors ist es, aus einer problemangepassten Geometriebeschreibung eine Finite-Elemente Einteilung zu generieren. Diese wird vom eigentlichen Wirbelstromcode ERATO eingelesen und

weiterverarbeitet. Ausgabe dieses Programmes sind letztlich die berechneten transienten Ströme in den einzelnen Elementen. Sie können mit Hilfe der Postprozessoren in übersichtlicher Form (Liste, Plot) dargestellt werden.

In Kap. 4 dieses Berichtes werden die vorhandenen Preprozessoren näher beschrieben. Mit den vorhandenen Programmen ist es möglich, eine ebene, rechteckige Platte sowie ein Torussegment zu behandeln. Das eigentliche Berechnungsprogramm ERATO wird in Kap. 5 diskutiert.

Kap. 6 dient der Beschreibung der benutzten Postprozessoren. Hierunter fällt einmal das Programm DRUCK zum übersichtlichen Drucken der Wirbelstromverteilungen in der Struktur und des zeitlichen Verlaufes der induzierten Ströme in einzelnen Elementen. Der wichtigste Postprozessor ist das Plotprogramm ERAPLO, das hier nur kurz vorgestellt wird.

Eine DV-technische Besonderheit des ERATO-Codes ist, daß es zwei unterschiedliche Codeversionen gibt, die an die Rechenanlagen CYBER 205 (Universität Karlsruhe) und Siemens 7890/IBM 3081 (KfK) angepasst sind. Beide Codeversionen werden in einer einzigen Source-Datei gehalten, für die Verwaltung der Versionen wurde das Programm SODEMA (Kap. 7) entwickelt.

Im Kap. 8 schließlich wird die Anwendung des ERATO Programmsystems an zwei Beispielen demonstriert.

2. DIE FINITE-ELEMENTE-NETZWERKMETHODE

2.1 Die Maxwellgleichungen in quasistatischer Näherung

Wie alle Vorgänge der Elektrodynamik kann auch die Wirbelstromberechnung letztlich auf die Maxwellschen Differentialgleichungen zurückgeführt werden:

(1a) $\text{rot}\ \underline{H} - \underline{D}^{\bullet} = \underline{i}$

(1b) $\text{div}\ \underline{D} = \rho$

(1c) $\text{rot}\ \underline{E} + \underline{B}^{\bullet} = 0$

(1d) $\text{div}\ \underline{B} = 0$

Dieser Satz von 8 partiellen Differentialgleichungen für 16 physikalische Größen muß noch durch geeignete Stoffgesetze geschlossen werden. Wir wollen hier im weiteren voraussetzen, daß die Struktur lineares magnetisches Verhalten zeigt und dem ohmschen Gesetz genügt:

(2a) $\underline{B} = \mu_0 \underline{H}$

(2b) $\underline{i} = \sigma \underline{E}$

Die Annahme des Stoffgesetzes (2a) stellt eine starke Einschränkung dar, weil damit Eisen als wirbelstromführendes Material ausgeschlossen wird. Als weitere Vereinfachung wird außerdem der Term $\underline{D}^{\bullet}$ in Gleichung (1a) vernachlässigt. Diese sog. quasistatische Näherung wird in der Literatur für nicht zu hochfrequente Vorgänge als akzeptabel angesehen. Setzt man dann (2) in (1) ein, so ergeben sich die folgenden Differentialgleichungen:

(3a) $\text{rot}\ \underline{B} = \mu_0 \underline{i}$

(3b) $\text{div}\ \underline{D} = \rho$

(3c) $\text{rot}\ \underline{E} + \underline{B}^{\bullet} = 0$

(3d) $\text{div}\ \underline{B} = 0$

2.2 Die Diffusionsgleichung für das Vektorpotential

Obwohl es möglich wäre, das System (3) direkt zu behandeln, ist dies wegen der Vielzahl der auftretenden Unbekannten numerisch nicht sinnvoll. Eine Möglichkeit, das ganze System auf eine einzige Vektor-Differentialgleichung zu reduzieren, bietet die Einführung des sog. Vektorpotentials $\underline{A}$. Dies ist ein Vektorfeld mit der Eigenschaft:

$$(4) \qquad \underline{B} = \mathrm{rot}\ \underline{A}$$

dessen Existenz durch Gleichung (3d) gesichert ist. Setzt man (4) in (3c) ein, so ergibt sich:

$$(5) \qquad \mathrm{rot}\ (\underline{E} + \dot{\underline{A}}) = 0 \quad \Longleftrightarrow \quad \underline{E} + \dot{\underline{A}} = \nabla U$$

mit dem skalaren Potential U. Nach Einsetzen von (5) und (2b) in (3a) erhält man:

$$(6) \qquad \mu_0 \underline{i} = \mu_0 \sigma \underline{E} = \mu_0 \sigma (\nabla U - \dot{\underline{A}}) = \mathrm{rot\ rot}\ \underline{A} = -\Delta \underline{A} + \nabla(\mathrm{div}\ \underline{A})$$

Die Wahl der Potentiale U und $\underline{A}$ ist nicht eindeutig. Durch eine geeignete Wahl kann man deshalb stets erreichen, daß gilt:

$$(7) \qquad \mathrm{div}\ \underline{A} - \mu_0 \sigma U = 0$$

Damit ergibt sich aus (6) die Wirbelstrom-Diffusionsgleichung für $\underline{A}$:

$$(8) \qquad \mu_0 \sigma \dot{\underline{A}} - \Delta \underline{A} = 0$$

Gleichung (8) ist Ausgangspunkt vieler Verfahren zur numerischen [11] oder analytischen [12] Wirbelstromberechnung. Für 3D Probleme wird die Lösung der Diffusionsgleichung aber sehr aufwendig, da es sich bei (8) um ein System von drei gekoppelten partiellen Differentialgleichungen handelt. Zudem muß dies System im gesamten dreidimensionalen Raum gelöst werden, obwohl in Wirklichkeit nur die Lösung innerhalb der leitenden Strukturen gewünscht ist. Im weiteren wird deshalb ein Näherungsverfahren, die Finite-Elemente Netzwerkmethode, entwickelt, das (8) speziell im Bereich dieser leitenden Strukturen approximativ löst.

2.3 Ableitung der Netzwerksgleichungen

Die Idee des Näherungsverfahrens ist es, die gesamte leitende Struktur durch ein Netzwerk zu approximieren, und das Wirbelstromproblem dadurch auf ein klassisches Netzwerk-Analyseproblem zurückzuführen. Dazu wird die Struktur in N einzelne, diskrete Finite Elemente F_j (j = 1, ..., N) unterteilt. Die genaue Form dieser Elemente und die Art der Unterteilung wird in Kap. 2.5 diskutiert. Die Mittelachsen S_j der Elemente bilden die Zweige dieses Netzwerkes (s. Abb. 2.1).

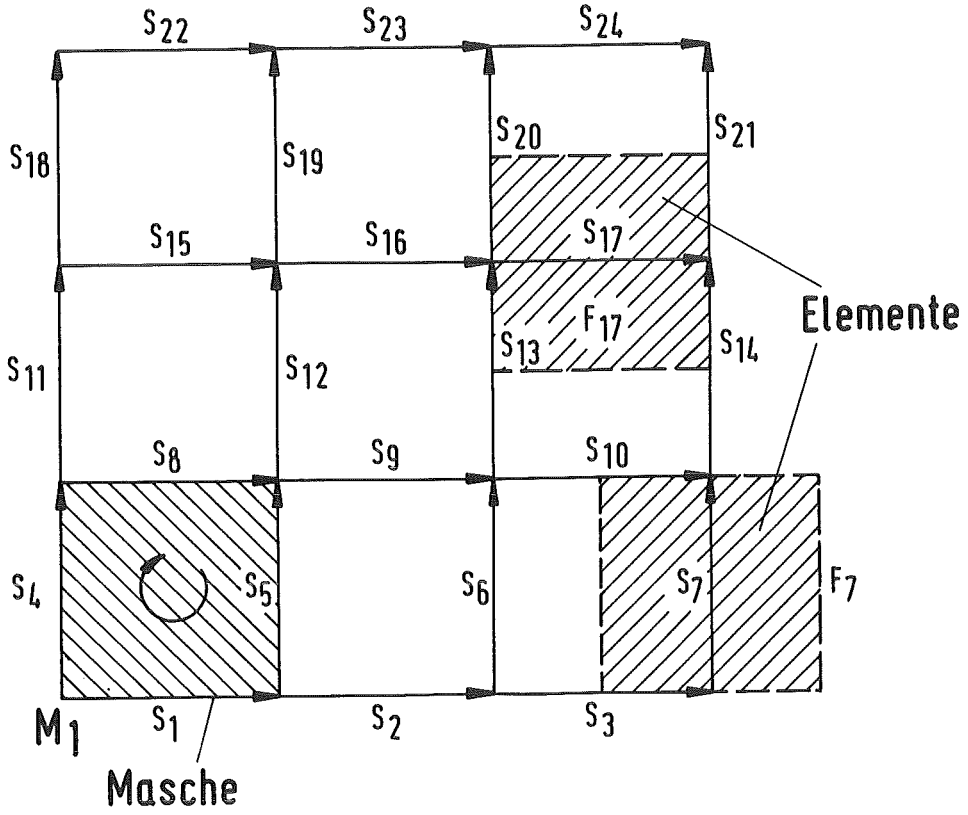


Abb. 2.1: Das benutzte äquivalente Netzwerk

Nach (3c) gilt dann für eine beliebige Masche M_k dieses Netzwerkes, die von den Zweigen $S_1^k, \ldots, S_{N_k}^k \subset \{S_1, \ldots, S_N\}$ gebildet wird:

$$\iint_{M_k} \text{rot } \underline{E}\, d\underline{F} + \iint_{M_k} \underline{B}^{\bullet}\, d\underline{F} = 0 \qquad \Longleftrightarrow \qquad (4)$$

$$\iint_{M_k} \text{rot } \underline{E}\, d\underline{F} + \iint_{M_k} \text{rot } \underline{A}^{\bullet}\, d\underline{F} = 0 \qquad \Longleftrightarrow$$

Satz v. Stokes

$$(9) \qquad \sum_{j=1}^{N_k} \left\{ \int_{S_j^k} \underline{E}\, d\underline{s} + \int_{S_j^k} \underline{A}^{\bullet}\, d\underline{s} \right\} = 0$$

Gleichung (9) muß nun für jede Masche M_k erfüllt sein. Dies ist auf jeden Fall richtig, wenn der Klammerausdruck in (9) für alle Elemente F_i ($i = 1, \ldots, N$) verschwindet:

$$(10) \qquad \int_{S_i} \underline{E}\, d\underline{s} + \int_{S_i} \underline{A}^{\bullet}\, d\underline{s} = 0 \quad \text{für } i = 1, \ldots, N$$

Zur weiteren Bestimmung des Ausdruckes (10) betrachten wir zunächst den zweiten Summanden. Das Magnetfeld im Bereich des Elementes F_i setzt sich additiv wie folgt zusammen:

(a) Aus den Magnetfeldern, die von den (vorerst noch unbekannten) Wirbelströmen $\underline{i}_j$ in allen Elementen F_j ($j = 1, \ldots, N$) erzeugt werden,

und

(b) den Magnetfeldern der N_{ex} Ströme $\underline{i}_{k,ex}$ in den externen erregenden Stromkreisen $F_{k,ex}$ ($k = 1, \ldots, N_{ex}$). Diese Ströme werden als bekannt angenommen.

Damit gilt dann:

$$(11) \qquad \int_{S_i} \underline{A}^{\bullet}\, d\underline{s} = \sum_{j=1}^{N} \int_{S_i} \underline{A}_j^{\bullet}\, d\underline{s} + \sum_{j=1}^{N_{ex}} \int_{S_i} \underline{A}_{k,ex}^{\bullet}\, d\underline{s}$$

Dabei ist $\underline{A}_j$ das Vektorpotential des Stromes $\underline{i}_j$ im Element F_j, und $\underline{A}_{k,ex}$ das entsprechende Vektorpotential des Stromes $\underline{i}_{k,ex}$ im externen Leiter $F_{k,ex}$. Die Bestimmung dieser Potentiale kann mit der Maxwellgleichung (3a) erfolgen. Danach gilt:

$$\text{rot } \underline{B}_j = \text{rot rot } \underline{A}_j = \mu_0 \sigma \underline{i}_j \qquad \Longleftrightarrow$$

$$(12) \qquad \Delta \underline{A}_j - \nabla(\text{div } \underline{A}_j) = \mu_0 \sigma \underline{i}_j$$

Durch geeignete Normierung von $\underline{A}_j$ läßt sich stets erreichen:

(13) $\quad \operatorname{div} \underline{A}_j = 0$

so daß folgt:

(14) $\quad \Delta \underline{A}_j = \mu_0 \sigma \underline{i}_j$

Nach allgemeinen Sätzen der Potentialtheorie erhält man eine Lösung von (14) durch:

(15) $$\underline{A}_j = (\mu_0/4\pi) \iiint_{F_j} \underline{i}_j(\underline{r}')/(|\underline{r} - \underline{r}'|)\, dV' \quad (j = 1, \ldots, N)$$

In ähnlicher Art und Weise läßt sich auch das Vektorpotential der externen Anregung errechnen:

(16) $$\underline{A}_{k,ex} = (\mu_0/4\pi) \iiint_{F_{k,ex}} \underline{i}_{k,ex}(\underline{r}')/(|\underline{r} - \underline{r}'|)\, dV' \quad (k = 1, \ldots, N_{ex})$$

Bis hierher ist die gesamte Ableitung exakt gewesen. Jetzt führen wir zur Vereinfachung von (15) und (16) zum ersten Mal Näherungen ein:

> Die induzierte Stromdichte $\underline{i}_j$ ist innerhalb eines Elementes F_j konstant, die Stromrichtung fällt mit der Achsrichtung S_j zusammen. Die erregenden Stromkreise $F_{k,ex}$ werden als unendlich dünn betrachtet.

Damit gilt dann für (15) und (16):

(17) $$\underline{A}_j = (\mu_0 \underline{i}_j/4\pi) \iiint_{F_j} 1/(|\underline{r} - \underline{r}'|)\, dV' \quad (j = 1, \ldots, N)$$

(18) $$\underline{A}_{k,ex} = (\mu_0 I_{k,ex}/4\pi) \int_{F_{k,ex}} 1/(|\underline{r} - \underline{r}'|)\, d\underline{s}' \quad (k = 1, \ldots, N_{ex})$$

wobei $I_{k,ex}$ den Gesamtstrom in Leiter $F_{k,ex}$ bezeichnet. Ist I_j entsprechend der Gesamtstrom durch das Element F_j, so sind die Größen

$$(19) \qquad L_{ij} = (1/I_j) \int_{S_i} \underline{A}_j \, d\underline{s} \quad (i,j = 1, \ldots, N)$$

$$(20) \qquad L_{ik,ex} = (1/I_{k,ex}) \int_{S_i} \underline{A}_{k,ex} \, d\underline{s} \quad (i = 1, \ldots, N \; ; \; k = 1, \ldots, N_{ex})$$

nur noch von der Geometrie der äußeren Anregung und der Elementeinteilung abhängig. Wird weiter angenommen, daß sich diese Geometrie durch die Wirbelströme nicht ändert, so brauchen diese Induktivitätskoeffizienten nur ein einziges Mal berechnet zu werden. Der zweite Term in Gleichung (9) schreibt sich dann:

$$(21) \qquad \int_{S_i} \underline{A}^{\bullet} \, d\underline{s} = \Sigma_{j=1}^{N} L_{ij} \, I_j^{\bullet} + \Sigma_{k=1}^{N_{ex}} L_{ik,ex} \, I_{k,ex}^{\bullet} \quad (i = 1, \ldots, N)$$

Zur Berechnung des ersten Summanden in (9) wenden wir erneut die Näherung der konstanten Stromdichte in den Elementen an. Damit ergibt sich:

$$(22) \qquad \int_{S_i} \underline{E} \, d\underline{s} = \int_{S_i} (\underline{i}_i/\sigma) \, d\underline{s} = I_i \, R_i$$

Dabei ist R_i der ohmsche Widerstand des Elementes F_i. In Matrixschreibweise ergibt sich (9) aus (21) und (22) zu:

$$(23) \qquad \underline{\underline{L}} \underline{I}^{\bullet} + \underline{\underline{R}} \underline{I} = -\underline{\underline{L}}_{ex} \underline{I}_{ex}^{\bullet}$$

mit

$$(24a) \qquad \underline{\underline{L}} = (L_{ij}) \quad (i,j = 1, \ldots, N)$$

$$(24b) \qquad \underline{\underline{L}}_{ex} = (L_{ik,ex}) \quad (i = 1, \ldots, N \; ; \; k = 1, \ldots, N_{ex})$$

$$(24c) \qquad \underline{\underline{R}} = \mathrm{diag}(R_1, \ldots, R_N)$$

$$(24d) \qquad \underline{I} = (I_1, \ldots, I_N)^T$$

$$(24e) \qquad \underline{I}_{ex} = (I_{1,ex}, \ldots, I_{N_{ex},ex})^T$$

2.4 Die Maschen-Form der Netzwerksgleichungen

Das Ergebnis des letzten Abschnittes war das System (23) von gewöhnlichen Differentialgleichungen. Da in diesem System für jeden Netzwerkszweig eine Gleichung vorhanden ist, heißt (23) auch die Zweig-Form der Netzwerksgleichungen. Entsprechend werden (24a-c) als die Zweigmatrizen von Induktivität und Widerstand, und (24d,e) als Vektoren der Zweigströme bezeichnet. Das so abgeleitete Zweig-Gleichungssystem (23) ist nun aber singulär. Der mathematische Grund für diese Singularität liegt im Übergang von Gleichung (9) auf Gleichung (10), der nicht reversibel ist. Physikalisch wird diese Singularität dadurch plausibel, daß die einzelnen Zweigströme I_j nicht vollkommen unabhängig sind, sondern vielmehr die Kirchhoff'schen Gleichungen erfüllen müssen:

(25a) Für alle Knoten: $\Sigma_j I_j = 0$

(25b) Für alle Maschen $\Sigma_j U_j = 0$

(U_j - Zweigspannungen, Σ_j - Summe über alle den jeweiligen Knoten berührenden Zweige bzw. alle Zweige in der betreffenden Masche)

Um ein eindeutig lösbares Gleichungssystem zu erhalten, muß der Übergang von (9) auf (10) rückgängig gemacht werden, also die Zweigströme auf einen geeigneten Satz von M unabhängigen Maschenströmen transformiert werden. Formal wird diese Transformation mit Hilfe der Maschen-Inzidenzmatrix $\underline{\underline{H}}$ durchgeführt.

Zur formalen Definition dieser N×M Matrix wird zunächst in allen Maschen eine eindeutige Umlaufrichtung definiert, sowie allen Netzwerkszweigen (d.h. allen Elementen F_j) eine Richtung gegeben (Abb. 2.1). In der j-ten Zeile der Matrix $\underline{\underline{H}}$ (j = 1, ..., N) ist dann verzeichnet, zu welchen Maschen der Zweig j gehört, und wie die Zweigrichtung zur Maschen-Umlaufrichtung orientiert ist:

(26) $H(j,k) = \begin{cases} 1 & \text{wenn Zweig } S_j \text{ zur Masche } M_k \text{ gehört, bei gleicher Zweig- und Maschenorientierung} \\ -1 & \text{wenn Zweig } S_j \text{ zur Masche } M_k \text{ gehört, bei verschiedener Zweig- und Maschenorientierung} \\ 0 & \text{wenn Zweig } S_j \text{ nicht zur Masche } M_k \text{ gehört} \end{cases}$

$(j = 1, \ldots, N \; ; \; k = 1, \ldots, M)$

Die Transformation von Maschenströmen $\underline{I}_m$ auf Zweigströme $\underline{I}$ kann damit einfach durch eine Matrixmultiplikation durchgeführt werden:

(27) $$\underline{I} = \underline{\underline{H}}\, \underline{I}_m$$

Aus der Definition der Maschen-Inzidenzmatrix $\underline{\underline{H}}$ folgt unmittelbar:

(28) $$\underline{\underline{H}}^T \underline{U} = 0$$

Setzt man (27) in (23) ein und multipliziert von links mit $\underline{\underline{H}}^T$, so ergibt sich:

(29) $$\underline{\underline{L}}_m \dot{\underline{I}}_m + \underline{\underline{R}}_m \underline{I}_m = -\underline{\underline{L}}_{m,ex} \dot{\underline{I}}_{ex}$$

mit

(30a) $\underline{\underline{L}}_m = \underline{\underline{H}}^T \underline{\underline{L}}\, \underline{\underline{H}}$: M×M Maschen-Induktivitätsmatrix

(30b) $\underline{\underline{R}}_m = \underline{\underline{H}}^T \underline{\underline{R}}\, \underline{\underline{H}}$: M×M Maschen Widerstandsmatrix

(30c) $\underline{\underline{L}}_{m,ex} = \underline{\underline{H}}^T \underline{\underline{L}}_{ex}$: $M \times N_{ex}$ Maschen-Induktivitätsmatrix der externen Ströme

Die Anzahl M der unabhängigen Maschen in einem Netzwerk hängt von der Anzahl N der Zweige und K der Knoten sowie von der Netzwerkstopologie ab, für ebene Netzwerke gilt stets:

(31) $M = N - K + 1$ (Eulersche Polyederformel [13])

(29) ist nun ein System von unabhängigen, gewöhnlichen Differentialgleichungen, das mit Standardmethoden gelöst werden kann.

2.5 Elementeinteilung

Im ERATO-Code werden wie in FEDIFF als Elemente rechtwinklige Quader endlicher Dicke benutzt (Platten). Nach der in Kap 2.4 gemachten Voraussetzung trägt jede Platte einen Strom, der innerhalb dieser Platte räumlich konstant ist und in Richtung der Plattenachse zeigt (Abb. 2.2). Im Gegensatz zu den finiten Elementen, wie sie z. B. in der Strukturmechanik geläufig sind, haben wir es in ERATO also mit gerichteten Elementen zu tun.

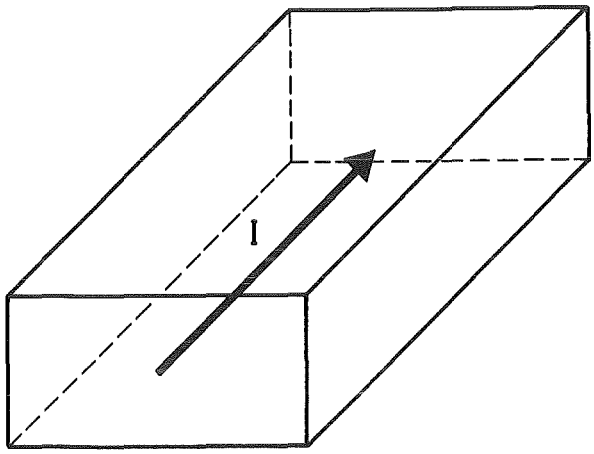


Abb. 2.2: Das ERATO-Element

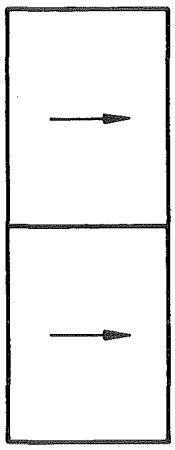
Nun ist die Richtung des induzierten Wirbelstromes aber normalerweise nicht bekannt. Diesem Umstand wird bei der Elementeinteilung dadurch Rechnung getragen, daß die gesamte Struktur mit einem Netz zueinander orthogonaler, sich gegenseitig überlappender Platten mehrfach überdeckt wird (Abb. 2.3). Das die Struktur letztlich repräsentierende Netzwerk wird von den Mittelachsen dieser Platten gebildet. Diese Überlappungsmethode wurde von Christensen [4] für ebene Strukturen eingeführt und von Preis [1] auf 3D Strukturen erweitert.

Von diesem Prinzip der mehrfachen Überdeckung wird nur dort abgegangen, wo die Stromrichtung a priori bekannt ist. Auf diese Art und Weise können sehr einfach zwei verschiedene Arten von Randbedingungen simuliert werden:

s - (streamline) Ränder: Stromlinien gehen parallel zum Rand

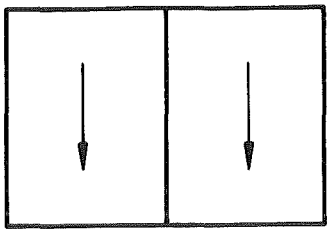
e - (equipotential) Ränder: Stromlinien gehen senkrecht zum Rand

s-Ränder treten z. B. auf, wenn ein Strukturteil elektrisch isoliert ist von seiner Umgebung, während e-Ränder i.A. Symmetrielinien der Strukturgeometrie und der Anregung sind. Abb. 2.4 zeigt die Anordnung der Platten an einem s-Rand, die Platten parallel zum Rand schließen bündig mit ihm ab, während Platten senkrecht zum Rand eine halbe Plattenbreite vorher aufhören. Bei der Platteneinteilung an einem e-Rand (Abb. 2.5) ist es genau umgekehrt. Sind die Stromverläufe zu beiden Seiten eines solchen Randes spiegelsymmetrisch, so heißen die Ränder s-Spiegel bzw. e-Spiegel. Die Bedeutung derartiger Symmetrien wird im nächsten Abschnitt behandelt.



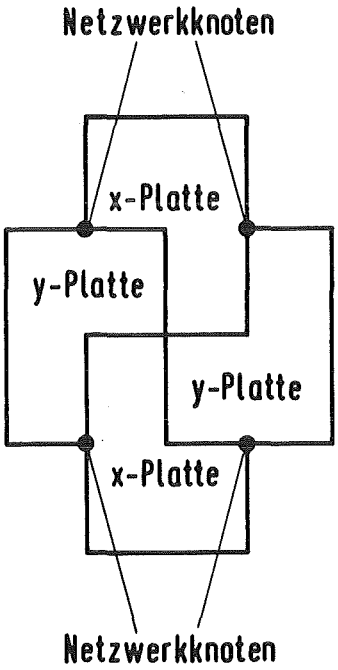
(a)

x-Platten



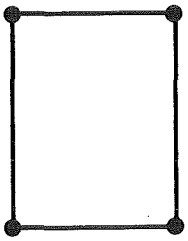
(b)

y-Platten



(c)

Überlappung der Platten



(d)

Zugehöriges Netzwerk

Abb. 2.3: Die Überlappungsmethode

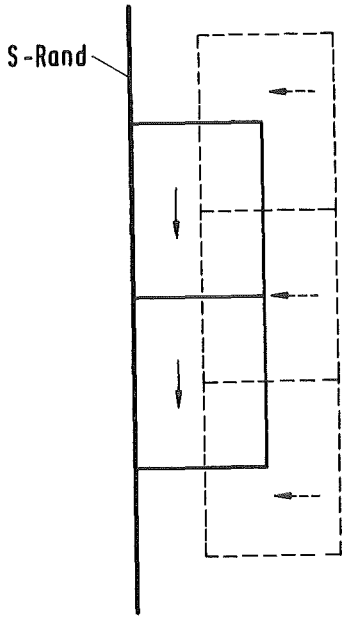


Abb. 2.4: Anordnung der Elemente an einem s-Rand

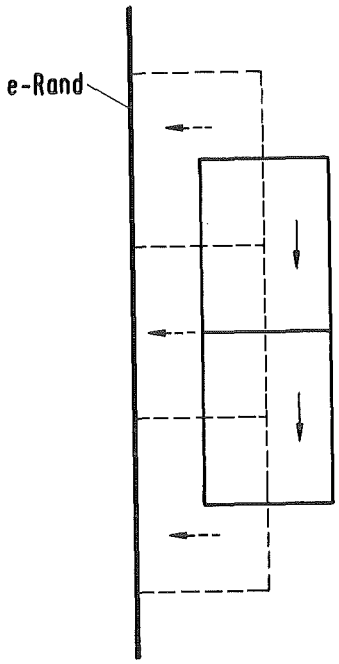


Abb. 2.5: Anordnung der Elemente an einem e-Rand

2.6 Berücksichtigung von Symmetrien

Ein häufiger Fall in den Anwendungen ist es, daß sowohl die Geometrie der wirbelstromtragenden Struktur als auch das erregende Magnetfeld bzw. die felderzeugenden Stromkreise bestimmte Symmetrien aufweisen. Hierbei kann es sich z. B. um die Spiegelsymmetrie bzgl. einer bestimmten Ebene handeln. Liegt so ein Fall vor, so weiß man, daß die induzierten Wirbelströme an bestimmten, zueinander spiegelsymmetrisch gelegenen Stellen gleichen Absolutbetrag haben, nur ihre Richtung ist eventuell, entsprechend der Symmetriebedingung, verändert. Setzt man für das FEN-Modell weiter voraus, daß die Symmetrieebenen entweder parallel oder senkrecht zu den Elementen orientiert sind, so unterscheiden sich die Ströme in symmetrisch gelegenen Elementen allenfalls durch das Vorzeichen.

Ein Beispiel für so eine Situation zeigt Abb. 2.6, die Abbildung wurde [3] entnommen. Die gesamte Struktur ist aus 4 symmetrischen Segmenten zusammengesetzt. Die Anregung durch ein externes Magnetfeld B_{ex} ist so, daß die v-Achse einem e-Spiegel und die u-Achse einem s-Spiegel entspricht. Alle anderen auftretenden Ränder sind s-Ränder. Es gibt 2 Typen zueinander orthogonaler Elemente (u-Platten und v-Platten). Die Elementeinteilungen in den Symmetriesegmenten entstehen aus den Elementen im Grundsegment durch sukzessive Spiegelung an den Achsen.

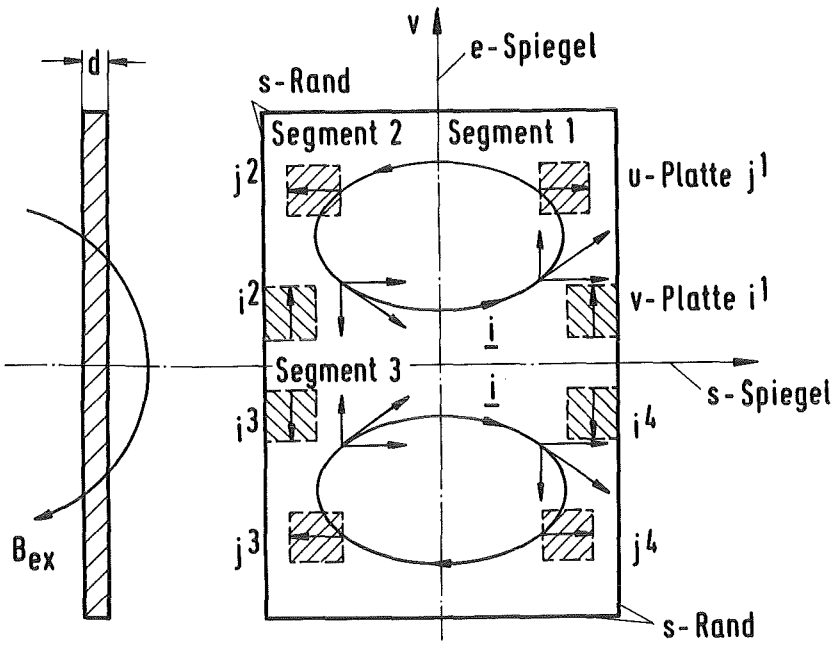


Abb. 2.6

Stromverlauf bei symmetrischer Anregung (Abbildung aus [3])

Der Vorteil ist in so einem Fall, daß bei der Finite-Elemente Diskretisierung nur noch das kleinste symmetrische Grundsegment, im Beispiel also der rechte, obere Quadrant, berücksichtigt zu werden braucht. Die Netzwerksmatrizen sind deshalb entsprechend kleiner. Bei der Berechnung der Induktivitätsmatrix $\underline{\underline{L}}$ sind die Symmetriesegmente allerdings zu berücksichtigen, da alle "fiktiven" Platten in diesen Segmenten in induktiver Wechselwirkung mit den Platten des Grundsegmentes stehen. Der Induktivitätskoeffizient L_{ij} berechnet sich deshalb als eine Summe:

$$(32) \qquad L_{ij} = \sum_{k=1}^{N_s} v(k,j) L_{ij}^{k}$$

Dabei ist N_s die Anzahl der Symmetriesektoren und L_{ij}^{k} der Induktivitätskoeffizient zwischen der Platte i^1 im Grundsegment und derjenigen Platte j^k im Segment k, die zur Platte j^1 des Grundsegmentes gehört ($k = 1, \ldots, N_s$). $v(k,j)$ bezeichnet das Vorzeichen dieses Koeffizienten, also $v(k,j) = +1$ oder -1.

Für die Bestimmung von $v(k,j)$ sind sowohl die Orientierung der Platte j^k als auch die Stromrichtung in dieser Platte zu beachten. Sind entweder die Orientierung von j^k und j^1 unterschiedlich (Segment 1 und 2 in Abb. 2.6, u-Platte), oder hat der Strom in diesen Platten unterschiedliche Richtung

(Segment 1 und 2, v-Platte), so ist v(k,j) = -1. Trifft keine dieser Bedingungen zu (z. B. bei Koppelung innerhalb des Grundsegmentes), oder treffen beide zu (Segment 1 und 4, v-Platte), so ist v(k,j) = 1. Allgemein hängt v(k,j) also von der Art der Symmetriebedingungen (e-Spiegel oder s-Spiegel) sowie von der Art der Platten, d.h. ihrer Orientierung bzgl. der Symmetrieränder ab. Im Beispiel von Abb. 2.6 gilt für alle Plattentypen: $v(1,j) = v(4,j) = 1$, $v(2,j) = v(3,j) = -1$.

2.7 Analytische Berechnung der Induktivitätskoeffizienten.

Der größte numerische Aufwand bei der praktischen Verwirklichung der FEN-Methode entsteht bei der Berechnung der Induktivitätsmatrizen (Gleichung (24a u. b)), da diese Matrizen in der Regel voll besetzt sind. Ein wesentlicher Vorteil der Verwendung quaderförmiger Elemente, wie sie Abb. 2.2 zeigt, ist es, daß in diesem Fall die Zweig-Induktivitätsmatrix $\underline{\underline{L}}$ zu einem Großteil analytisch berechnet werden kann. Es ist in diesem Fall nämlich möglich, daß Vektorpotential eines Einheitsstromes in so einem Element (Formel (17)) analytisch darzustellen. Zur entgültigen Berechnung der Induktivitätskoeffizienten muß dieses Potential dann nur noch nach (19) entlang der Netzwerkszweige integriert werden.

Berücksichtigt man eine wirkliche dreidimensionale Stromverteilung im Element, so ergibt sich nach [1] für den Betrag des Vektorpotentiales bei Vorliegen einer Stromdichte $\underline{i}$ im Element (Abb. 2.7, Richtung des Vektorpotentials = Stromrichtung):

$$
\begin{aligned}
A_{(p)} = -(\mu_0 i_x/4\pi) * \{ & \Sigma_{i=1}^{8} \operatorname{sign}[y_{(i)} z_{(i)} x_{(i)}] * \\
& [\, z_i x_i \ln(r_i + y_i) + y_i x_i \ln(r_i + z_i) + y_i z_i \ln(r_i + x_i) \\
& - \tfrac{1}{2} z_i^2 \arctan((y_i r_i + y_i^2 + z_i^2)/z_i x_i) \\
& - \tfrac{1}{2} y_i^2 \arctan((z_i r_i + y_i^2 + z_i^2)/y_i x_i) \\
& - \tfrac{1}{2} x_i^2 \arctan((y_i r_i + y_i^2 + x_i^2)/z_i x_i) \,] \,\}
\end{aligned}
\tag{33}
$$

mit

$$x_i = x_{(p)} - x_{(i)}$$

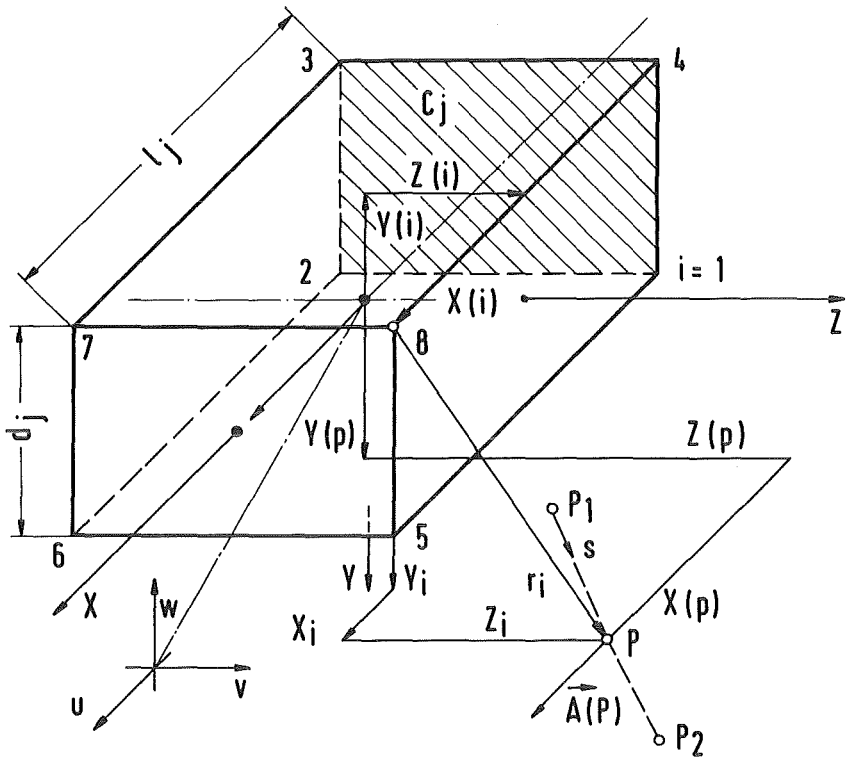


Abb. 2.7: Vektorpotential eines Einheitsstromes in einem quaderförmigen Leiter (Abbildung wurde aus [3] entnommen)

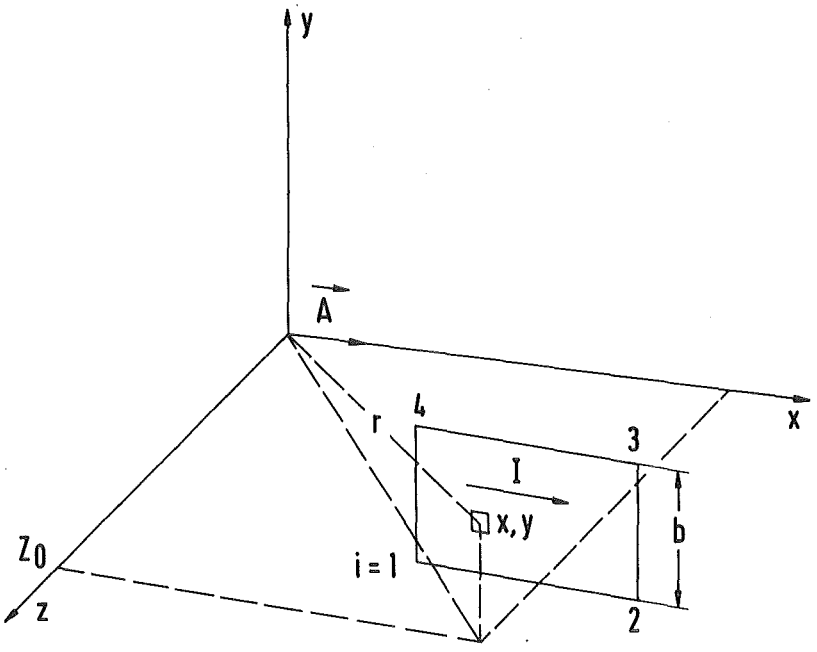


Abb. 2.8: Das Vektorpotential eines flächenhaft verteilten Einheitsstromes (Abbildung wurde aus [4] entnommen)

$$y_i = y_{(p)} - y_{(i)}$$

$$z_i = z_{(p)} - z_{(i)}$$

$$r_i = \sqrt{x_i^2 + y_i^2 + z_i^2} \qquad i = 1, \ldots, 8 \text{ Eckpunkte der Platte}$$

Dieser Ausdruck muß bei der Aufstellung der $\underline{\underline{L}}$-Matrix sehr häufig ausgewertet werden, was bei großer Elementanzahl mit einem hohen numerischen Aufwand verbunden ist. Es liegt deshalb nahe, nach Näherungen für das Vektorpotential (17) zu suchen, die auf einfachere analytische Formeln führen. Eine Möglichkeit ist es dabei, die Dicke d_j des Elementes zu vernachlässigen, also das Vektorpotential eines flächenhaft verteilten Einheitsstromes zu bestimmen. Diese Näherung wird z.B. im SPARK Code [6] benutzt. Das Potential ergibt sich dann nach [4] durch folgende Formel (Abb. 2.8):

$$\underline{A} = \underline{e}_x(\mu_0 I/4\pi b) * \{ \Sigma_{i=1}^{4}(-1)^{n+1} [x_i \ln(r_i+y_i) + y_i \ln(r_i+x_i) - z_0 \arctan((y_i x_i)/(z_0 r_i))] \} \tag{34}$$

mit

$$r_i = \sqrt{x_i^2 + y_i^2 + z_0^2} \qquad i = 1, \ldots, 4 \text{ Eckpunkte}$$

Eine noch einfachere Möglichkeit, den Koeffizienten L_{ij} zu berechnen, ergibt sich dann, wenn man das Element F_j durch ein eindimensionales Linienelement ersetzt, und gleichzeitig die Integranden in (17) und (19) durch die Funktionswerte in der Mitte der jeweiligen Integrationsbereiche S_i bzw. S_j approximiert. Es ergibt sich dann nach [4] folgende Formel für L_{ij} (Abb. 2.9):

$$L_{ij} = (\mu_0/4\pi) [(\underline{Q}_2^j - \underline{Q}_1^j)*(\underline{Q}_2^i - \underline{Q}_1^i)] / [0.5*| \underline{Q}_2^j + \underline{Q}_1^j - (\underline{Q}_2^i + \underline{Q}_1^i)|] \tag{35}$$

Die Zweig-Widerstandsmatrix $\underline{\underline{R}}$ läßt sich sehr einfach aus der Elementgeometrie berechnen (Abb. 2.7):

$$R_j = \ell_j/\sigma C_j \quad (j = 1, \ldots, N) \tag{36}$$

(ℓ_j - Länge des Zweiges S_j, C_j - Größe der Stirnfläche des Elementes F_j)

Es verbleibt schließlich noch, eine analytische Formel für die Koeffizienten der externen Induktivitätsmatrix $\underline{\underline{L}}_{ex}$ abzuleiten. Dies ist nur möglich, wenn

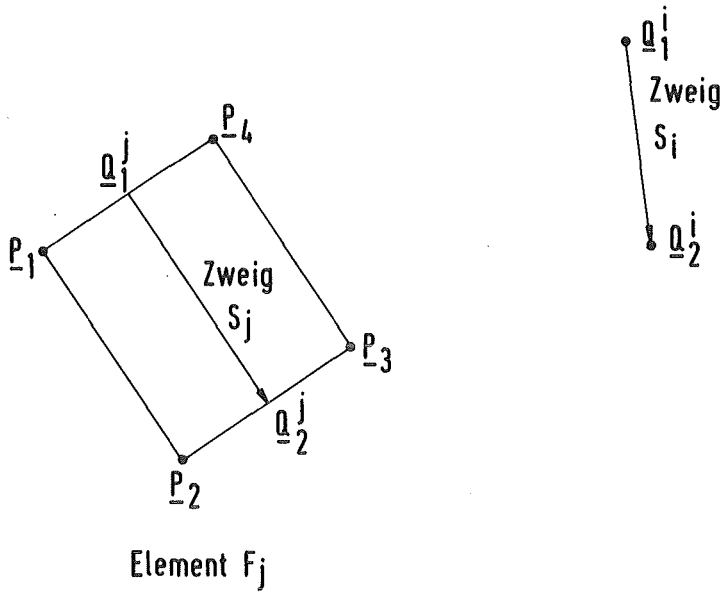


Abb. 2.9: Das Vektorpotential eines Linienstromes

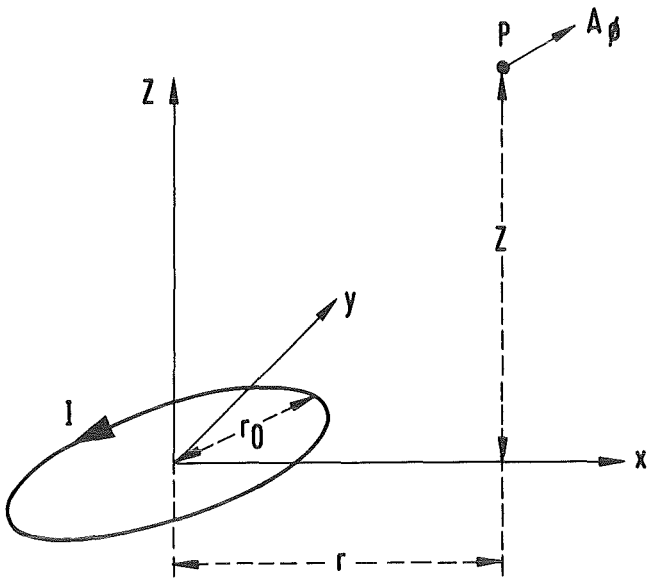


Abb. 2.10: Das Vektorpotential eines Kreisstromes

man zusätzliche einschränkende Annahmen über die Form der felderzeugenden Leiter trifft. In den Codes ERATO und FEDIFF wird deshalb angenommen, daß diese Leiter kreisförmig sind. Das nach (18) berechnete Vektorpotential hat dann nur eine einzige Komponente A_ϕ in Umfangsrichtung, die man nach [14] wie folgt berechnen kann (Abb. 2.10):

$$A_\phi = (\mu_0 I/2\pi) * \sqrt{r}\,\sqrt{z^2+(r+r_0)^2} * [F(\tfrac{1}{2}\pi,k) - E(\tfrac{1}{2}\pi,k)\,] - (\mu_0 I r_0/\pi) * 1/\sqrt{z^2+(r+r_0)^2} * F(\tfrac{1}{2}\pi,k) \tag{37}$$

mit

$k^2 = 4r_0 r/(z^2+(r+r_0)^2)$

F - vollständiges elliptisches Integral 1. Art

E - vollständiges elliptisches Integral 2. Art (siehe z.B. [15])

3. GLOBALE STRUKTUR DES PROGRAMMSYSTEMS

3.1 Zielsetzung und generelle Methode

Das generelle Ziel bei der Weiterentwicklung des FEDIFF-Codes ist gewesen, das Programm weitestgehend geometrieunabhängig zu machen. Es sollte also möglich sein, mit keinen oder zumindest möglichst wenig Anpassungen ein und dasselbe Programm zur Berechnung von Wirbelströmen in Strukturen mit ganz unterschiedlicher Geometrie und Topologie wie z. B. einer ebenen Platte und einem dreidimensionalen Torus anzuwenden. Dies Ziel ist mit dem ERATO-Code zwar nicht hundertprozentig, aber doch weitgehend verwirklicht worden.

Die Grundidee zur Erreichung dieses Zieles war es, den gesamten Ablauf der FEN-Analyse aufzuspalten in geometrieabhängige und geometrieunabhängige Einzelschritte. Jeder dieser Einzelschritte wurde dann in einem separaten Programmodul realisiert. Die einzelnen Module sind untereinander über wohldefinierte Datenschnittstellen gekoppelt. Die Grundstruktur dieser Modularisierung zeigt Abb. 3.1, ein ähnliches Schema ist auch in sehr vielen anderen Finite Elemente Programmen zu finden.

Es gibt drei Hauptmodule. Als erstes kommt ein als Preprozessor bezeichnetes Programm. Dieses hat die Aufgabe, aus einer geeigneten Geometriebeschreibung der leitenden Struktur eine Elementeinteilung zu erzeugen. Da die dabei benutzte Geometriebeschreibung sowie die zur Berechnung der Elementeinteilung benutzte Methode stark vom jeweiligen Problem abhängen, wird man normalerweise für unterschiedliche Klassen von Problemen auch unterschiedliche Preprozessoren haben.

Auf den vom Preprozessor bereitgestellten Daten über die finiten Elemente arbeitet dann das eigentliche Wirbelstromberechnungsprogramm ERATO. In diesem Programmodul wird der Hauptteil der FEN-Analyse durchgeführt, also der Aufbau der Maschen-Inzidenzmatrix (26), die Berechnung der verschiedenen Netzwerksmatrizen (24), die Zeitintegration der Netzwerks-Differential-gleichungen (29), sowie die Berechnung des von den Wirbelströmen induzierten Magnetfeldes. Dieser wichtigste und bei weitem rechenzeitaufwendigste Teil der FEN-Analyse ist nun vollkommen geometrieunabhängig. Damit können mit dem ERATO-Code jetzt Wirbelstromanalysen in jeder beliebigen dreidimensionalen Struktur durchgeführt werden, vorausgesetzt es steht ein geeigneter Preprozessor zur Verfügung, der für diese Struktur eine Finite-Elemente Einteilung erzeugt.

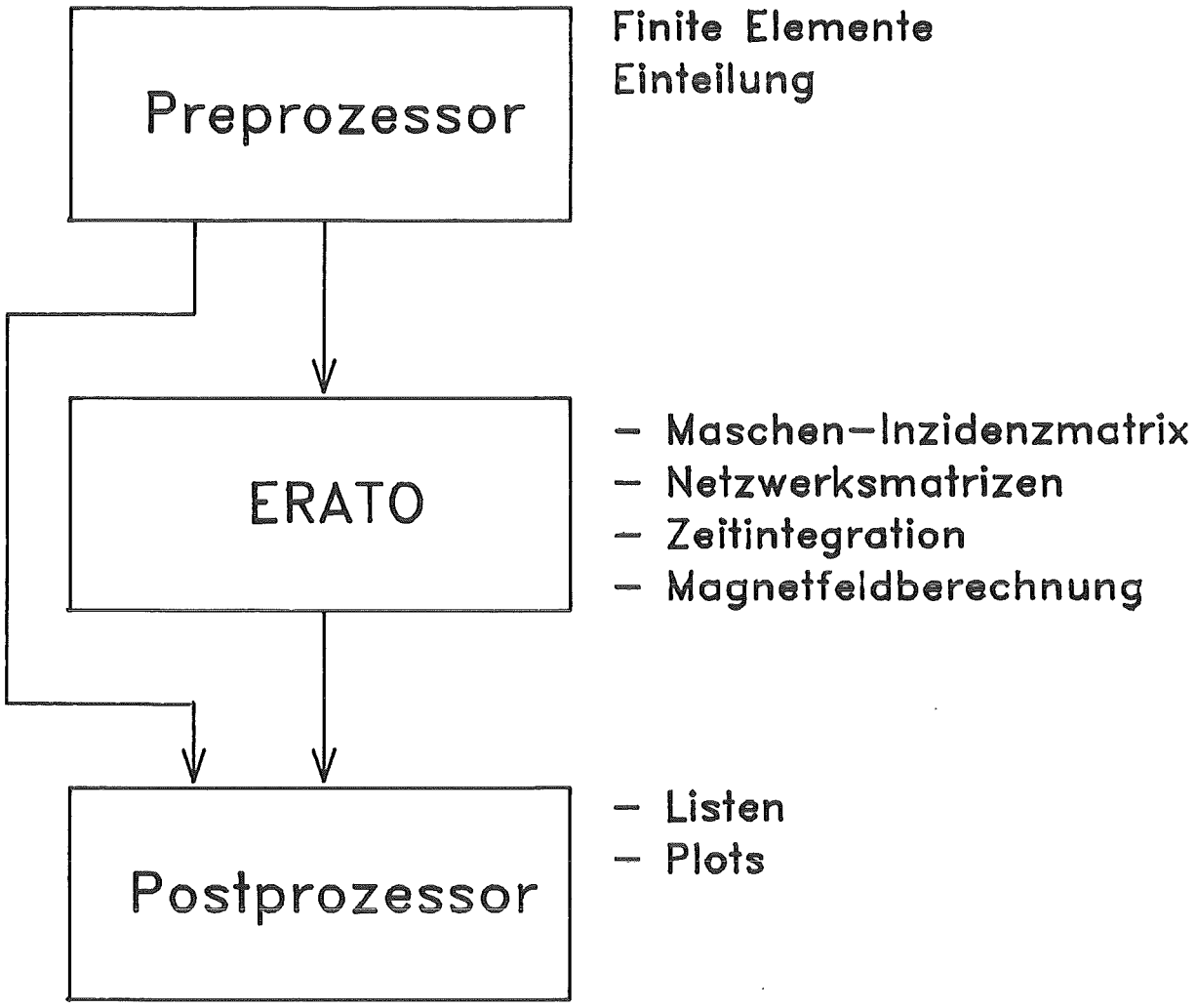


Abb. 3.1: Die Grundstruktur des ERATO-Programmsystems

Die Geometrieunabhängigkeit des eigentlichen Rechenprogrammes ist nur durch eine sehr allgemeine Methode zur Speicherung der Elementdaten zu erreichen. Programmintern wird deshalb jedes Element F_j durch seine Nummer j (j = 1, ..., N) referiert, außerdem sind jedem Element F_j (bzw. dem zugehörigen Netzwerkszweig S_j) ein Anfangsknoten k_a und ein Endknoten k_e ($1 \leq k_a, k_e \leq K$; $k_a \neq k_e$) zugeordnet. Alle notwendigen Informationen über die Elemente werden in zwei Datenfeldern gespeichert. Im ersten Feld sind die Angaben über die geometrische Lage der einzelnen Elemente enthalten, während mit dem zweiten Feld durch Angabe der zu den Elementen gehörigen Netzwerksknoten die Netzwerkstopologie beschrieben wird. Eine genaue Beschreibung dieser Datenfelder ist im Anhang 1 im Rahmen der Schnittstellendefinition Preprozessor/Rechenprogramm gegeben.

Die Ergebnisse des Rechenprogrammes ERATO sind dann im wesentlichen die in den einzelnen Elementen induzierten Wirbelströme zu verschiedenen Zeitpunkten. Diese können in einem letzten, als Postprozessor bezeichneten Programmodul übersichtlich dargestellt werden. Damit ist z. B. der Ausdruck der Zweigströme in übersichtlichen Listen möglich, vor allem aber die graphische Darstellung der errechneten Ströme mittels eines Plotprogrammes. Dieser Postprozessorteil stellt wieder einen geometrieabhängigen Schritt in der FEN-Analyse dar. Es ist vorgesehen, daß der Postprozessor sowohl vom Rechenprogramm als auch direkt vom Preprozessor Daten erhalten kann. Die Beschreibung dieser Schnittstellen für die vorhandenen Pre- und Postprozessoren findet sich in den Anhängen 2 und 3.

3.2 Globaler Aufbau und Datenfluß

Abb. 3.2 zeigt genauer die einzelnen Teilprogramme, aus denen das Programmsystem ERATO aufgebaut ist, zusammen mit ihren Ein- und Ausgabedateien. Angegeben sind auch die vordefinierten Namen dieser Dateien für die "CYBER-Version" (s.u.), sowie die zugehörige Fortran UNIT-Nummer.

Eine Besonderheit dieses Programmsystems ist es, daß der Code auf zwei verschiedenen Rechnern (CYBER 205 an der Uni Karlsruhe, IBM 3081 im KfK) mit teilweise unterschiedlichen Algorithmen läuft. Diese beiden Versionen werden im weiteren als die CYBER-Version und die IBM-Version bezeichnet. Beide Code-Versionen werden aber in einer einzigen Sourcedatei gespeichert, die sich auf dem Siemens Vorrechner der CYBER 205 befindet. Die Verwaltung dieser Versionen sowie die Generierung der jeweils benötigten Version geschieht mittels des Programmes SODEMA, das in Kap. 7.1 noch näher erläutert wird.

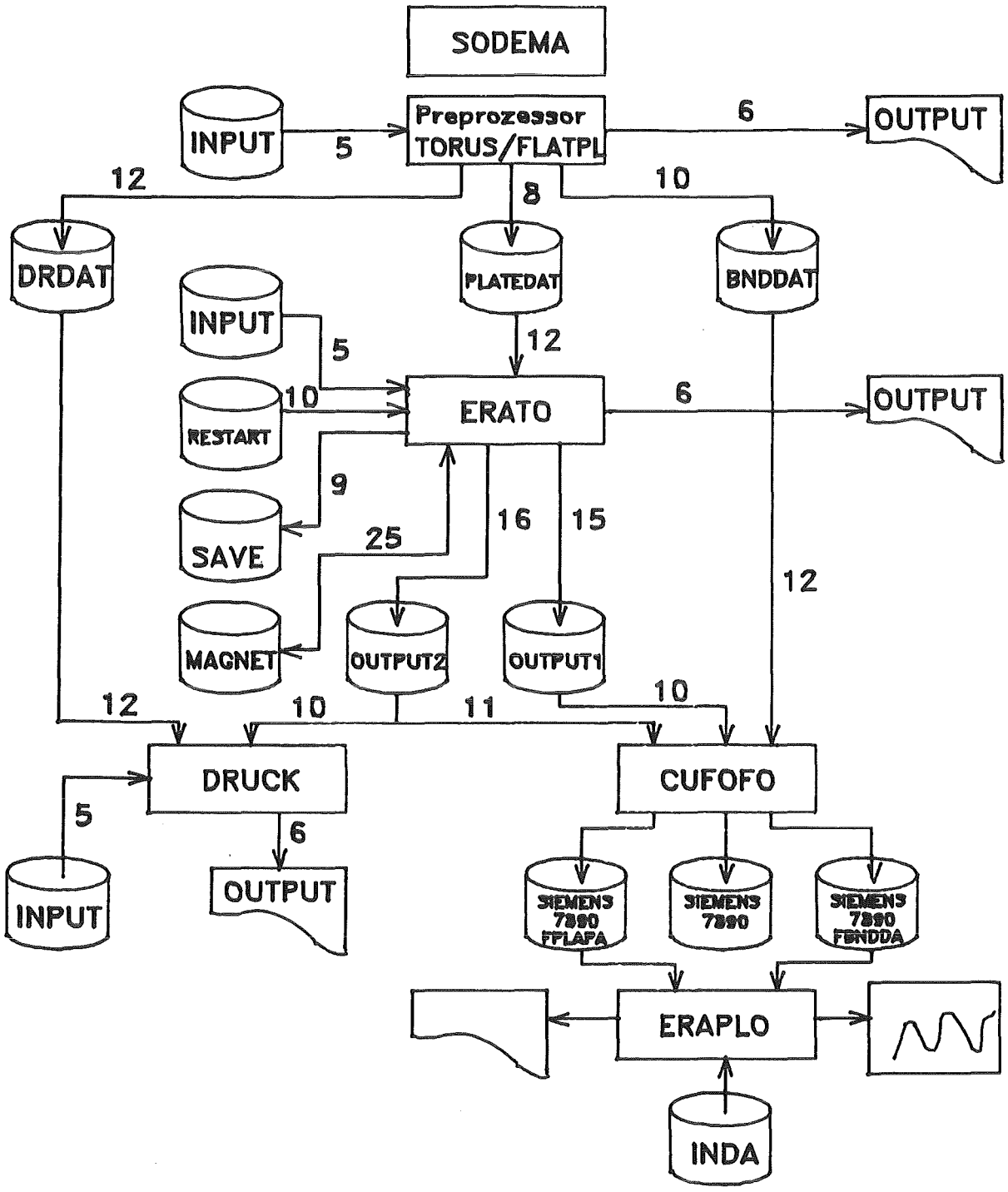


Abb. 3.2: Datenfluß innerhalb des Programmsystems ERATO

Auf der Preprozessorebene stehen z. Z. zwei verschiedene Programme zur Verfügung, TORUS (Elementeinteilung eines Torus-Segmentes) und FLATPL (Elementeinteilung einer ebenen Platte). Diese Programme werden in Kap. 4 beschrieben. Die Ausgabe dieser Preprozessoren geht einerseits auf den Drucker (Kanal 6, CYBER-Datei OUTPUT), zum anderen auf drei Plattendateien. Eine davon (Kanal 8, CYBER-Datei PLATEDAT) liefert dem Rechenprogramm ERATO die nötigen Informationen über die Elementeinteilung, die anderen beiden Dateien (Kanal 10, CYBER-Datei BNDDAT; Kanal 12, CYBER-Datei DRDAT) gehen direkt in die Postprozessoren.

Die algorithmischen Besonderheiten des Rechenprogrammes ERATO, in dem, wie schon erwähnt, der Hauptteil der Wirbelstromanalyse durchgeführt wird, werden in Kap. 5 beschrieben. Dies Programm erhält außer über den Kanal 12 (Ausgabe des Preprozessor) noch Eingabe über den Standard INPUT-Kanal 5, sowie eventuell von einem RESTART-File (Kanal 10, Cyber-Datei RESTART). Es produziert Drucker-Output über Kanal 6, sowie Ausgabe für den Postprozessorteil über die Kanäle 15 und 16 (CYBER-Dateien OUTPUT1 und OUTPUT2). Über Kanal 25 (CYBER-Datei MAGNET) wird eine als Zwischenspeicher bei der Magnetfeldberechnung benutzte Datei angesprochen. Bei vorzeitiger Unterbrechung der Wirbelstromanalyse können SAVE-Daten über Kanal 9 (CYBER-Datei SAVE) weggeschrieben werden.

Die Ausgabedatei OUTPUT2 (Kanal 16) enthält alle notwendigen Angaben über die Elementeinteilung sowie die errechneten transienten Wirbelströme in den einzelnen Elementen. Mit Hilfe des Programmes DRUCK (s. Kap. 6.1) können diese errechneten Ströme in übersichtlichen Listen ausgegeben werden. Über Kanal 16 (CYBER-Datei OUTPUT1) können die in ERATO errechneten Zweigmatrizen von Induktivität und Widerstand an den Postprozessor weitergegeben werden.

Neben dem Programm DRUCK steht als Postprozessor noch das Plotprogramm ERAPLO (s. Kap. 6.2 und [16]) zur Verfügung. Dies Programm arbeitet nicht direkt auf den verschiedenen Ausgabedateien von Preprozessor und Rechenprogramm, es ist vielmehr noch das Programm CUFOFO dazwischengeschaltet. Der Grund für diese Maßnahme ist, daß das ERAPLO-Programm wegen der benutzten speziellen Software nur auf der KfK-Anlage IBM 3081 laufen kann, während alle anderen Module hauptsächlich für die CYBER 205 konzipiert sind. Mit dem Programm CUFOFO wird deshalb der Datentransfer von der CYBER 205 auf die IBM durchgeführt. Außerdem findet eine Konversion der (unformatierten) Ausgabedaten von Preprozessor und Rechenprogramm in formatierte Daten statt, da über die Rechnerkoppelung nur Dateien im Kartenformat übertragen werden können. Eine vollständige Liste des Programmes CUFOFO ist im Anhang 4 aufgeführt.

3.3 Hinweise zur Implementierung, Rechenzeiten

Bis auf das Plotprogramm ERAPLO sind alle Teile des ERATO Programmsystems in Fortran 77 programmiert. Bei der CYBER-Version wird dabei der CYBER 200 Fortran Compiler benutzt. In einigen Unterprogrammen wird die spezielle CYBER Vektorsyntax angewandt, außerdem werden einige spezielle System- und Bibliotheksroutinen verwendet. Die JCL für einen Beispieljob der CYBER-Version findet man im Anhang 5.

In der IBM-Version wird durchweg Standard Fortran 77 verwendet. Alle Variablen und Konstanten sind einfach genau deklariert. Um mit der IBM-Version auf eine der CYBER-Version vergleichbare Genauigkeit zu kommen, muß der Siemens Fortran 77 Compiler mit der Option "AUTODBL(DBLPAD)" aufgerufen werden. Lediglich das Programm CUFOFO muß für die IBM-Version ohne diese Option übersetzt werden. Die JCL eines Beispieljobs der IBM-Version ist in Anhang 6 zu finden.

Das Plotprogramm ERAPLO ist in der Programmiersprache PL/1 programmiert und benutzt das graphische System GIPSY [17]. Eine Dokumentation dieses Programmes wird in einem anderen Bericht [16] gegeben.

Eine typische ERATO-Simulation mit 700 Elementen benötigt in der vektorisierten CYBER-Version 87 sec CPU-Zeit. Ohne Vektorisierung wächst die Rechenzeit für ein Problem dieser Größenordnung um den Faktor 6.

Dasselbe Problem wäre aus Speicherplatzgründen auf der IBM gar nicht zu behandeln (s. Kap. 5.1.2). Für den Rechenzeitvergleich CYBER-Version - IBM-Version wurde deshalb ein zweites Testproblem mit nur 500 Elementen spezifiziert. Dabei zeigte sich, daß die vektorisierte CYBER-Version hier eine um den Faktor 4 geringere Rechenzeit benötigt.

4. PREPROZESSOREN

4.1 Das Programm TORUS zur Elementeinteilung eines Torussegmentes

4.1.1 Modellierte Geometrie

Mit dem Preprozessorprogramm TORUS kann eine Finite-Elemente Einteilung für ein Segment einer Torusschale berechnet werden. Dabei wird der Begriff Torus hier allgemein für jedes geometrische Gebilde gebraucht, das durch Rotation einer Kurve in der (x,z) Ebene (der erzeugenden poloidalen Kontur) um die z-Achse entsteht (s. Abb. 4.1). Die erzeugende Konturkurve braucht dabei nicht unbedingt geschlossen zu sein. Das auf diese Art und Weise entstandene zweidimensionale Gebilde wird im weitenen meist als Konturfläche bezeichnet, sie soll die Mittelebene der in Wirklichkeit natürlich dreidimensionalen Torusschale beschreiben. Bei einer vollständigen Rotation um 360° entsteht ein in toroidaler Richtung geschlossener Torus, während bei der Rotation um den Winkel $\phi = 360°/NSEC$ ein Segment des Gesamttorus entsteht. $NSEC \geq 1$ ist dabei ein Eingabeparameter.

Die erzeugende poloidale Kontur kann sich beliebig aus Geradenstücken und Kreisbögen zusammensetzen, die im folgenden als poloidale Abschnitte bezeichnet werden. Jeder poloidale Abschnitt j ($j = 1, \ldots, NPP$) wird dabei durch Angabe seines Anfangs- und Endpunktes $\underline{P}_j$, $\underline{P}_{j+1}$ in der (x,z) Ebene sowie durch Angabe des Abschnittstyps (Gerade oder Kreisbogen) festgelegt. Für den Fall, daß es sich bei dem betreffenden Abschnitt um einen Kreisbogen handelt, ist zusätzlich die Angabe eines weiteren Punktes $\underline{Q}_j$ auf dem Kreisbogen erforderlich (s. Abb. 4.1a).

Die erzeugte Konturfläche kann auch in toroidaler Richtung in einzelne toroidale Abschnitte eingeteilt werden. Dies geschieht durch Angabe von Winkelpositionen ϕ_k ($0° \leq \phi_k \leq \phi$; $k = 0, 1, \ldots, NPT$; $NPT \geq 1$; $\phi_0 = 0°$; $\phi_{NPT} = \phi$), wie Abb. 4.1b zeigt. Das gesamte Segment ist somit mit einem Gitter aus toroidalen und poloidalen Maschenlinien überzogen. Für jede dieser Maschen kann dann ein anderer Wert für die Dicke der Torusschale und den spezifischen Widerstand des Materiales spezifiziert werden.

4.1.2 Randbedingungen in toroidaler und poloidaler Richtung

Wenn der Segmentwinkel ϕ kleiner als 360° ist (also $NSEC > 1$), so müssen an den Segmenträndern in toroidaler Richtung Randbedingungen spezifiziert werden.

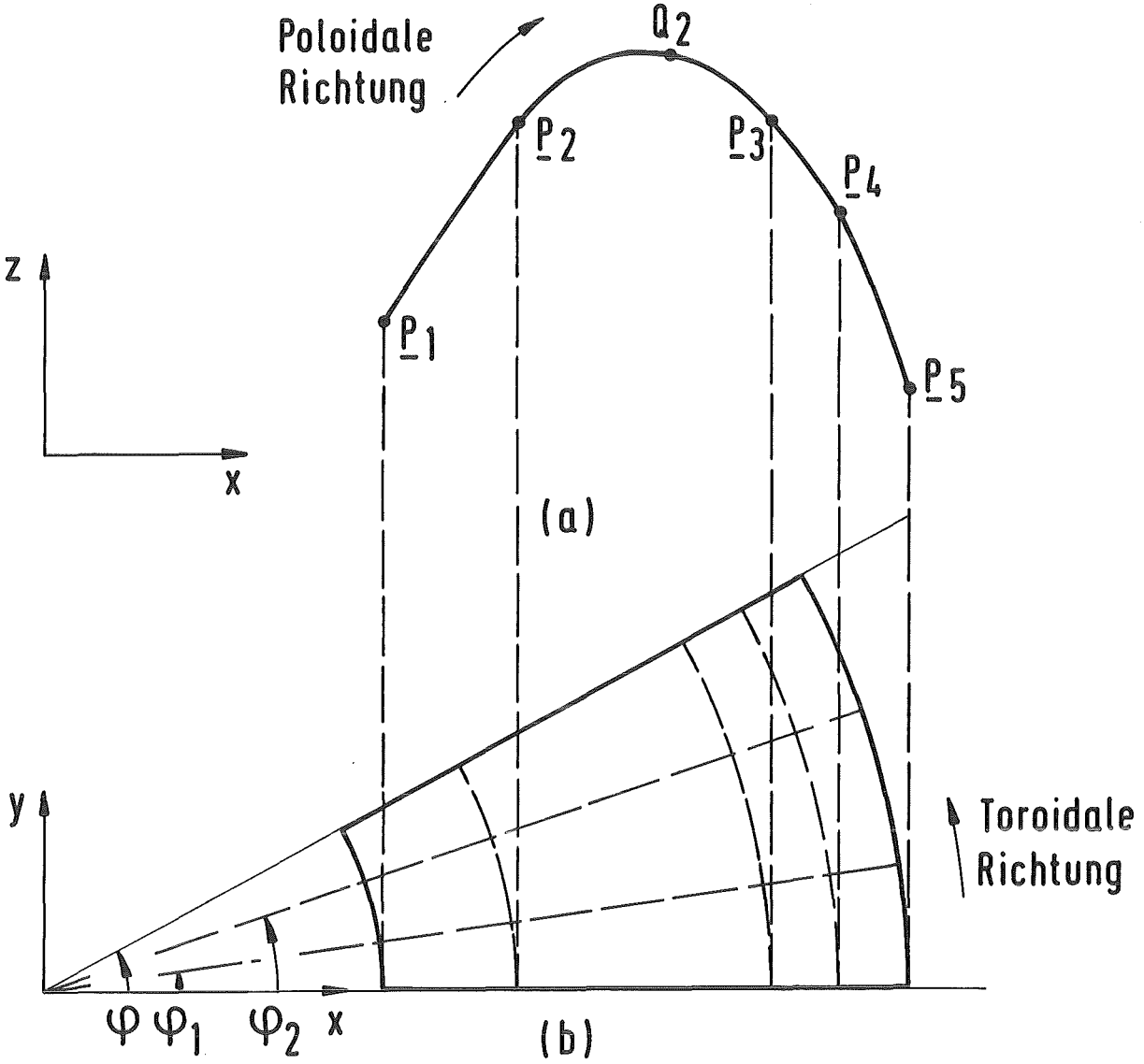


Abb. 4.1: Die Erzeugung der Torus-Mittelfläche (b) durch Rotation der erzeugenden poloidalen Kontur (a) um die Torusachse (=z-Achse)

Im Programm TORUS sind diese Ränder immer vom Typ "Spiegel" (s. Kap. 2.6). Es wird also stets angenommen, daß die leitende Struktur ein Volltorus ist, der aber in toroidaler Richtung aus NSEC identischen Segmenten zusammengesetzt ist. In diesen Segmenten sollen auch jeweils identische Wirbelströme fließen, was z.B. dann der Fall ist, wenn die erregenden Stromschleifen koaxial zur z-Achse sind.

An den Segmentgrenzen gibt es dann die Wahl zwischen folgenden Randbedingungen:

- Ideale Isolation am Segmentrand (s-Spiegel)
- Strom senkrecht zum Rand, gesamte poloidale Kontur ist leitend (e-Spiegel)
- Strom senkrecht zum Rand, Segmente teilweise isoliert (s-Spiegel/e-Spiegel)

Durch Wahl der letzten Randbedingung kann erzwungen werden, daß der Strom nur an ganz bestimmten Stellen der poloidalen Kontur von einem Segment ins nächste fließen kann und an allen anderen Stellen ein unendlich hoher Widerstand herrscht. Aus programmtechnischen Gründen darf dabei der leitende Bereich maximal zwei Zusammenhangskomponenten besitzen.

Ist die erzeugende poloidale Kontur nicht geschlossen, so müssen auch Randbedingungen für die Ränder in poloidaler Richtung vereinbart werden. Dies ist immer dann der Fall, wenn der erste und der letzte Stützpunkt der poloidalen Kontur nicht übereinstimmen, also $\underline{P}_1 \neq \underline{P}_{NPP+1}$. Trifft dies zu, so sind die betreffenden Ränder stets vom Typ "s-Rand" oder "s-Spiegel" (Eingabeparameter ISYM). Im letzteren Fall wird die (x,y)-Ebene als eine Symmetrieebene betrachtet, an der die Konturfläche gespiegelt wird.

4.1.3 Elementeinteilung

Die Festlegung der Diskretisierungsfeinheit baut auf der spezifizierten Einteilung in poloidale und toroidale Abschnitte auf. Es gibt zwei Typen von Elementen, die als toroidale Platten (Stromrichtung = toroidale Richtung) und als poloidale Platten (Stromrichtung = poloidale Richtung) bezeichnet werden. Für jeden poloidalen Abschnitt j ($1 \leq j \leq NPP$) muß die Zahl $NPPL_j$ von toroidalen Platten angegeben werden, in die er diskretisiert wird. Dabei schließen diese Platten mit den Bereichsgrenzen in poloidaler Richtung bündig ab (Abb. 4.2).

Genauso wird für jeden toroidalen Abschnitt k ($1 \leq k \leq NPT$) die Anzahl $NPTL_k$ von poloidalen Platten in diesem Abschnitt angegeben (Abb. 4.3), die jetzt mit den toroidalen Bereichsgrenzen bündig abschließen. Eine Ausnahme können hier nur die toroidalen Ränder des betrachteten Segmentes bilden, wo die jeweiligen Randbedingungen zu beachten sind. Aus diesen Festlegungen ergibt sich jetzt eindeutig die gesamte Elementeinteilung mit Hilfe des Überlappungsprinzips (Kap. 2.5).

Die Koordinaten der einzelnen Elemente (d.h. die Eckpunkte der Element-Mittelflächen) werden auch mit Hilfe der erzeugenden poloidalen Kontur berechnet. Abb. 4.4 zeigt für die schon in Abb. 4.2 gewählte Einteilung diejenigen Punkte auf der poloidalen Kontur, die durch Rotation um die z-Achse die Eckpunkte der toroidalen Platten erzeugen, und Abb. 4.5 dasselbe für die poloidalen Platten. Es liegen also bei beiden Typen von Platten die Eckpunkte stets auf der spezifizierten Konturfläche.

Dies hat nun eine wichtige Konsequenz. Handelt es sich bei dem betreffenden Bereich um eine doppelt gekrümmte Fläche, ist also das erzeugende poloidale Konturstück ein Kreisbogen, so schneiden sich die Mittelachsen von toroidalen und poloidalen Platten nicht mehr. Man kann also den Knotenpunkten des Netzwerkes als den Punkten, an denen (im Regelfall) je 2 toroidale und poloidale Platten zusammenstoßen, keinen eindeutigen geometrischen Ort mehr zuordnen.

Dieser Ort wird für die Wirbelstromberechnung selbst allerdings auch nicht benötigt. Hier kommt es allein auf die Netzwerkstopologie an, die in einer von der Geometrie unabhängigen Datenstruktur abgespeichert ist. Für die graphische Darstellung der Ergebnisse, also den Postprozessor, ist die Festlegung eines geometrischen Ortes der Knotenpunkte aber wichtig. Diese Punkte werden deshalb vom Preprozessor separat ermittelt und direkt auf die Ausgabedatei für den Postprozessor geschrieben (s. Anhang 2). Die Ermittlung der Koordinaten geschieht wieder mit Hilfe der erzeugenden poloidalen Kontur, damit sichergestellt ist, daß auch die Knotenpunkte auf der Konturfläche liegen (Abb. 4.6). Für jeden dieser Knotenpunkte werden dem Postprozessor weiterhin noch die zugehörigen Einheitsvektoren $\underline{e}_p$, $\underline{e}_t$ in poloidaler und toroidaler Richtung übermittelt.

Auf die beschriebene Art und Weise werden, wie schon erwähnt, die Mittelebenen der einzelnen Elemente berechnet. Durch einen weiteren Eingabeparameter NLAY läßt sich noch steuern, in wieviele übereinanderliegende Elementlagen die gesamte Schale eingeteilt wird (Abb. 4.7). Die einzelnen Elementlagen sind

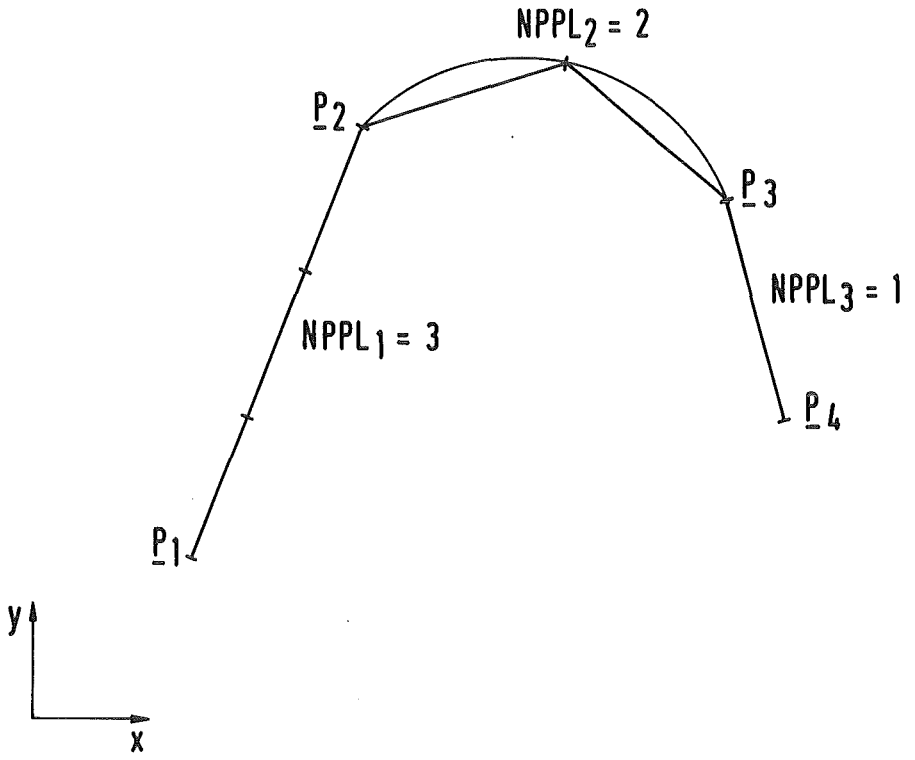


Abb. 4.2: Einteilung der poloidalen Kontur in toroidale Platten

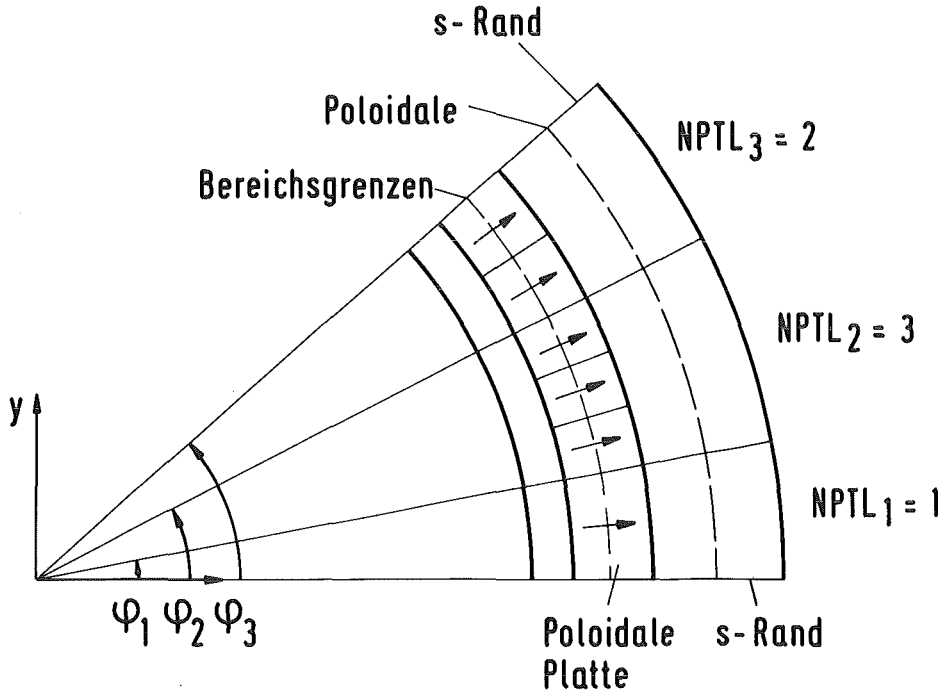


Abb. 4.3: Einteilung der toroidalen Kontur in poloidale Platten

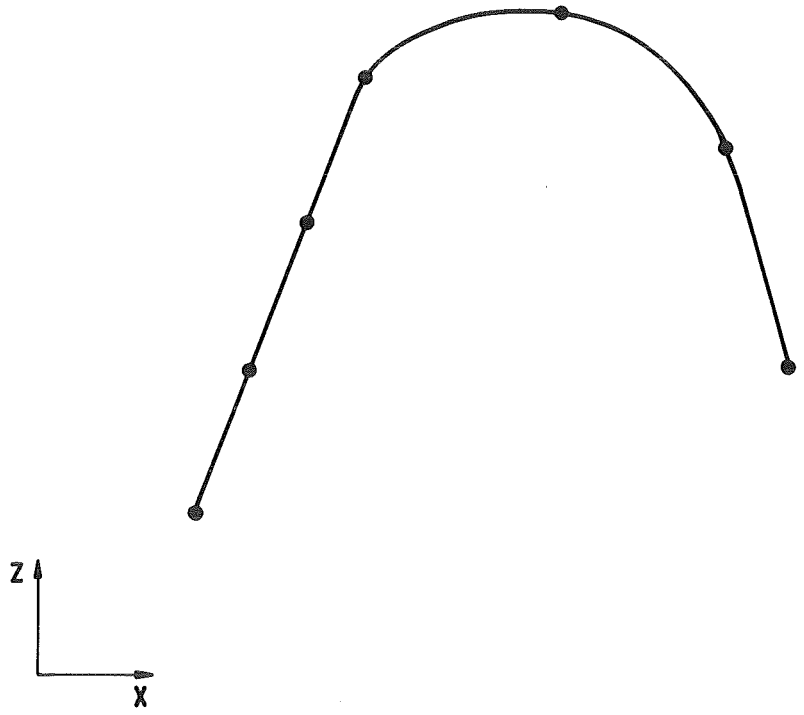


Abb. 4.4: Dieselbe poloidale Kontur wie in Abb. 4.2. Konturpunkte, die durch Rotation um die z-Achse die Eckpunkte der toroidalen Platten erzeugen

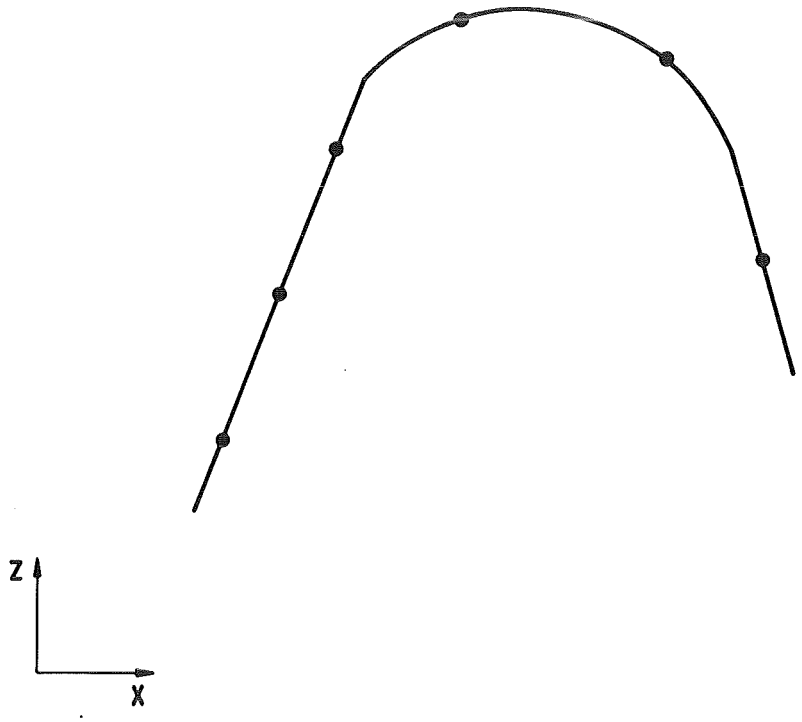


Abb. 4.5: Dasselbe wie Abb. 4.4 für die Eckpunkte der poloidalen Platten

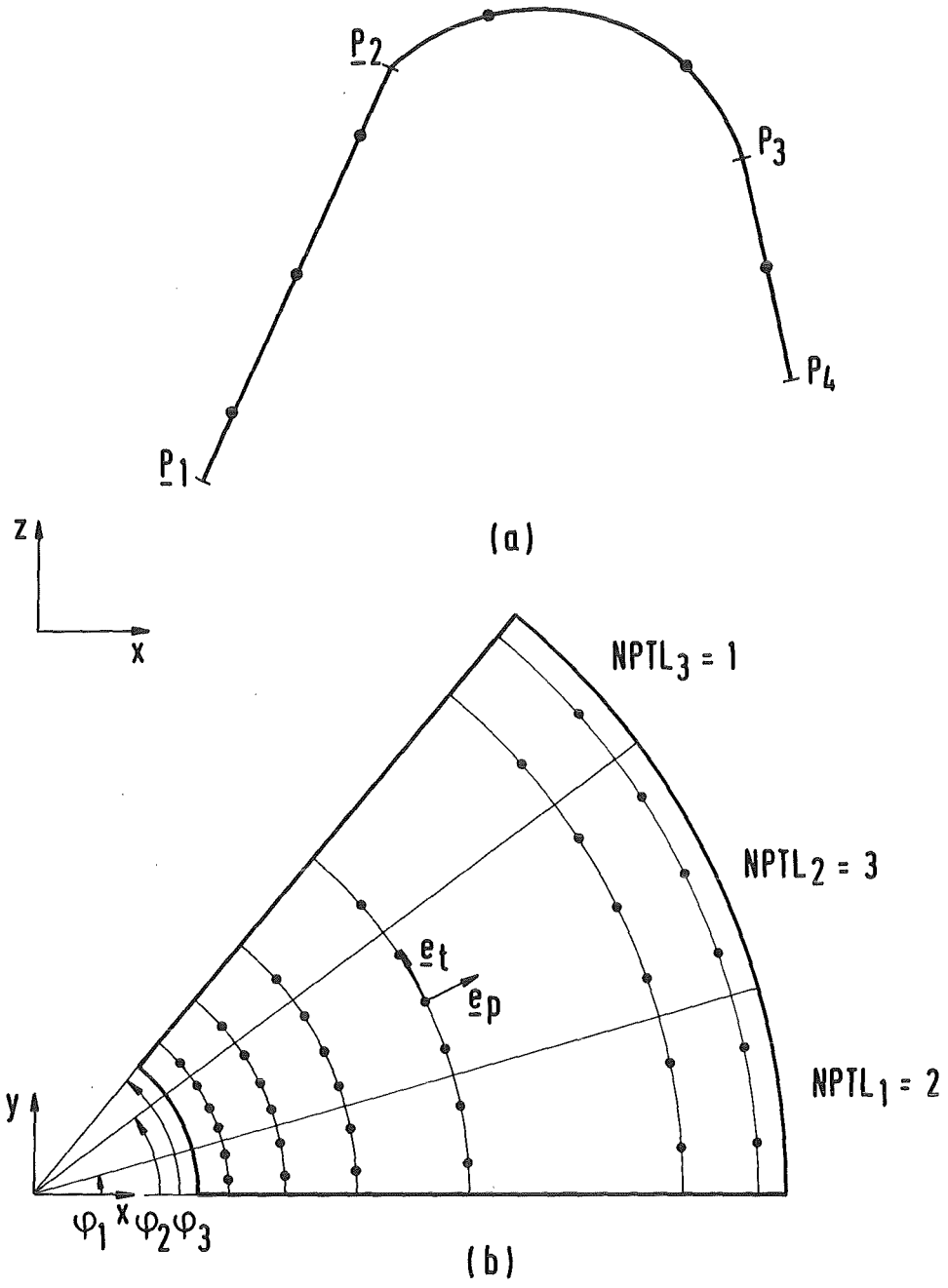


Abb. 4.6: Erzeugung (a) und geometrische Lage (b) der Knotenpunkte auf der Konturfläche

dabei allerdings elektrisch nicht direkt miteinander gekoppelt, sondern nur induktiv. Bei NLAY > 1 werden alle Elementlagen durch eine geeignete Verschiebung der Grundebene in Normalenrichtung erzeugt.

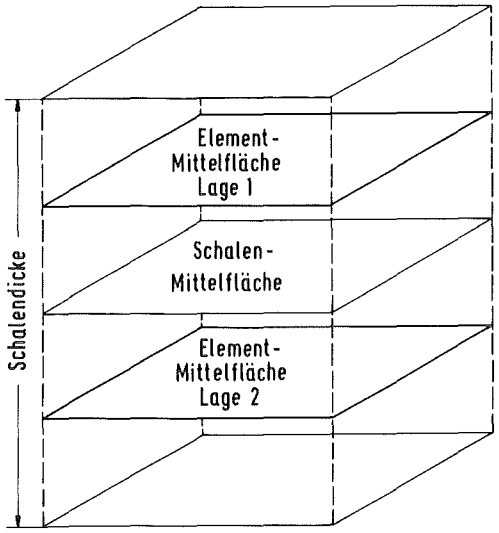


Abb. 4.7:

Diskretisierung der Schalendicke in mehrere Elementlagen

4.1.4 Programmablauf

Die modulare Struktur des Programmes TORUS zeigt Abb. 4.8. Das eigentliche Hauptprogramm ist sehr kurz:

```
PARAMETER (IDIMA = 65535)
COMMON /COA/ A(IDIMA)
CALL TORUS(A,IDIMA)
STOP
END
```

Es dient nur dazu, mittels PARAMETER-Statement ein globales Datenfeld A zu dimensionieren und das Steuerprogramm TORUS aufzurufen. Dies Feld A dient dann zur Speicherung aller während der Berechnung auftretenden Felder. Die Speicherplatzverwaltung in A geschieht in der Subroutine TORUS, sie ist so angelegt, daß für jedes Feld nur so viel Platz allokiert wird, wie für den momentanen Lauf auch gebraucht wird.

Von TORUS werden nacheinander eine Anzahl von Unterprogrammen aufgerufen, die weiter unten näher erläutert werden. Alle diese Unterprogramme erhalten über die Parameterliste die Anfangsadresse im A-Feld der von ihnen benötigten Datenfelder. Die Längen dieser Felder werden mit Hilfe des COMMON-Blockes GLOBNM übergeben.

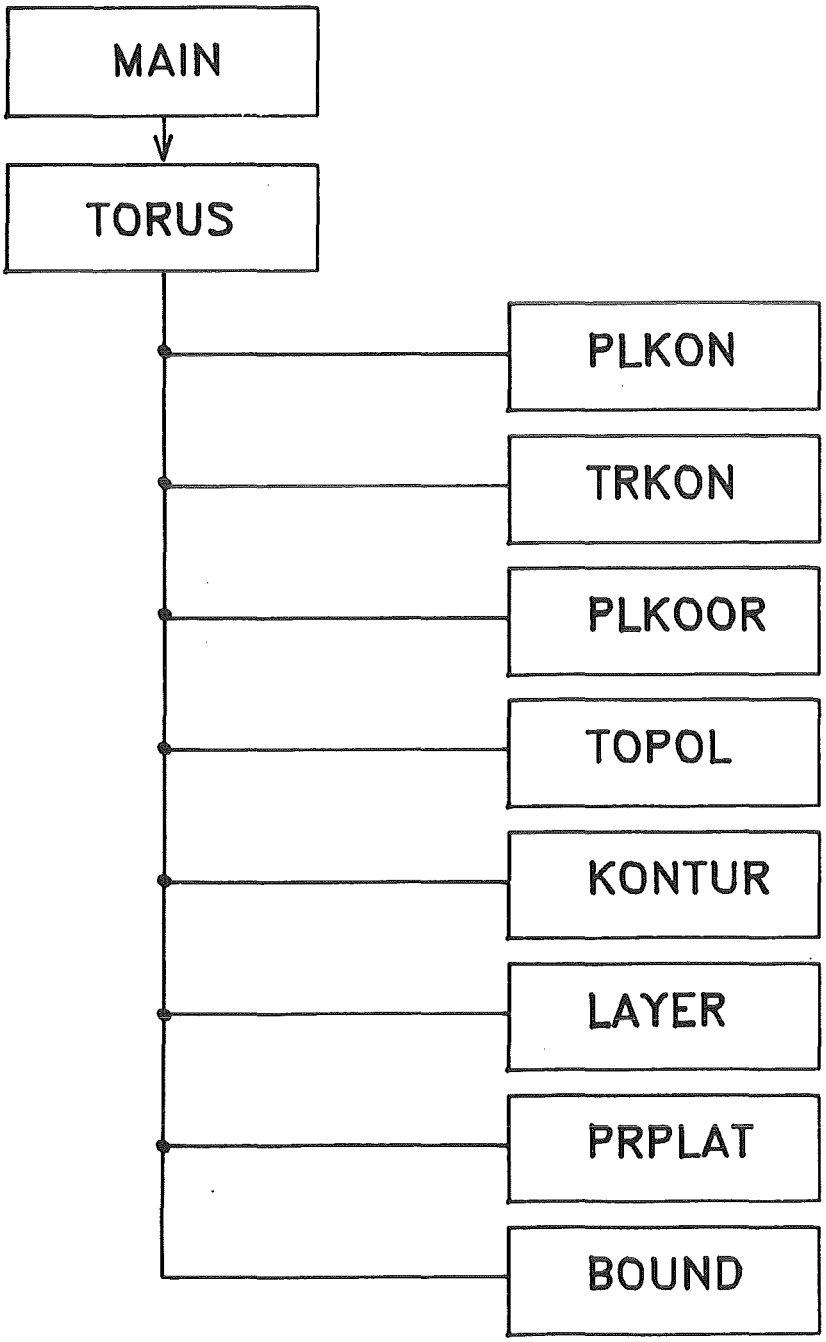


Abb. 4.8: Programmstruktur des Preprozessors TORUS

Die Steuerung des Druckeroutputs geschieht mit Hilfe des Feldes LPRINT von logischen Variablen, das sich im COMMON-Block PRT befindet. Die Belegung der einzelnen Variablen ist wie folgt:

LPRINT(1) = .T. - Testausdrucke in PLKON

LPRINT(2) = .T. - Testausdrucke in TRKON

LPRINT(3) = .T. - Testausdrucke in PLKOOR, toroidale Platten

LPRINT(4) = .T. - Testausdrucke in PLKOOR, poloidale Platten

LPRINT(5) = .T. - Testausdrucke in TOPOL

LPRINT(6) = .T. - Testausdrucke in BOUND

LPRINT(7) = .T. - Ausdruck der Daten über Platten-Geometrie

LPRINT(8) = .T. - Ausdruck der Daten über die Segmentränder (nur für Postprozessor)

LPRINT(9) = .T. - Testausdrucke in KONTUR

LPRINT(10) = .T.- Ausdruck der Daten über die Koordinaten der Knotenpunkte und die zugehörigen Einheitsvektoren (nur für Postprozessor)

LPRINT(11) - LPRINT(20) - z. Z. nicht belegt

Kurzbeschreibung der Funktion wichtiger Unterprogramme:

HEADER INPUT-Routine; Einlesen und Ausdruck der NAMELIST /NUM/; Konsistenzprüfungen

DAT INPUT-Routine; Einlesen und Ausdruck der NAMELIST's /POL/, /TOR/, /STOFF/; Konsistenzprüfungen; Berechnung der Feldlängen für den COMMON-Block GLOBNM.

PLKON (x,z)-Koordinaten der Eckpunkte poloidaler und toroidaler Platten auf der erzeugenden poloidalen Kontur

TRKON Toroidale Winkelpositionen der Eckpunkte der poloidalen und toroidalen Platten

PLKOOR Berechnung der (x,y,z)-Koordinaten aller Platten auf der Schalen-Mittelfläche

TOPOL Netzwerkstopologie

KONTUR Geometrische Orte der Netzwerksknoten und der zugehörigen Einheitsvektoren $\underline{e}_p$, $\underline{e}_t$

LAYER Wird nur bei NLAY > 1 aufgerufen, Berechnung der Element-Eckpunkte in den einzelnen Lagen durch Verschiebung senkrecht zur Schalen-Mittelfläche

PRPLAT Ausgabe der Daten über Plattengeometrie und Netzwerkstopologie an das Rechenprogramm ERATO

BOUNDS Ausgabe von Daten an den Postprozessor

4.1.5 Eingabebeschreibung

Alle Eingabedaten werden dem Preprozessor TORUS in Form von NAMELIST-Blöcken übergeben, die Bedeutung dieser Variablen ist im weiteren aufgelistet. Bei jeder Variablen ist neben einem Initialisierungswert auch der Variablentyp gemäß folgender Abkürzung angegeben:

I, I(n), I(n,m) - INTEGER-Wert, bzw. ein- oder zweidimensionales INTEGER-Feld
R, R(n), R(n,m) - REAL-Wert, bzw. entsprechendes Feld
C*k, C(n)*k - CHARACTER der Länge k, bzw. entsprechendes Feld
L, L(n) - LOGICAL-Wert, bzw. entsprechendes Feld

Ist ein einziger Wert für die Initialisierung eines Feldes angegeben, so wird das gesamte Feld mit diesem Wert initialisiert. Durchgehend wird das MKSA-Einheitensystem verwendet.

NAMELIST /NUM/ (Subroutine HEADER)

```
     NAMELIST /NUM/ RUNID , LPRINT , COMM  , NSEC , ISYM , NPP  , NPT ,
    1               NLAY  , IBNDN  , IBNDP , DI   , RHOI , NILZN , NILZP
```

RUNID (C*8;'XXXXXXXX')
Textstring zur Identifikation des Preprozessorlaufes

LPRINT (L(20);.FALSE.)
Steuerung des Druckeroutputs, die Bedeutung der einzelnen Variablen ist in Kap. 4.1.4 erklärt

COMM (C(10)*60;' ')
Beliebiger Text, der auf der ersten Seite der Job-Liste gedruckt wird

NSEC (I;1) NSEC ≥ 1
Anzahl der toroidalen Segmente

ISYM (I;0)
ISYM = 0: Ränder in poloidaler Richtung sind s-Ränder
ISYM ≠ 0: Ränder in poloidaler Richtung sind s-Spiegel

NPP (I;1) NPP ≥ 1
Anzahl der poloidalen Abschnitte

NPT (I;1) NPT ≥ 1
Anzahl der toroidalen Abschnitte

NLAY (I;1) NLAY ≥ 1
Anzahl der Lagen in Dickenrichtung der Schale

IBNDN (I;1)
IBNDN = 0: Rand in neg. toroidaler Richtung ist ein e-Spiegel
IBNDN ≠ 0: Rand in neg. toroidaler Richtung ist ein s-Spiegel

IBNDP (I;1)

IBNDP = 0: Rand in pos. toroidaler Richtung ist ein e-Spiegel
IBNDP ≠ 0: Rand in pos. toroidaler Richtung ist ein s-Spiegel

DI (R;1.E-3) DI > 0
Schalendicke

RHOI (R;1.E-7) RHOI > 0
Spezifischer Widerstand des Schalenmaterials [Ωm]

NILZN (I;1) 1 ≤ NILZN ≤ 2
Ist nur bei IBNDN = 0 wichtig: Anzahl der Zusammenhangskomponenten des leitenden Bereiches am e-Rand in negativer toroidaler Richtung

NILZP (I;1) 1 ≤ NILZP ≤ 2
Ist nur bei IBNDP = 0 wichtig: Anzahl der Zusammenhangskomponenten des leitenden Bereiches am e-Rand in positiver toroidaler Richtung

NAMELIST /POL/ (Subroutine DAT)

```
NAMELIST /POL/ PPOL  , PPOLM , IPOL , NPPL , ILEBN , ILEBP
```

Bem.: Alle in dieser NAMELIST vorkommenden Felder haben eine feste Länge, die mittels PARAMETER (NPPM in Subroutine DAT) definiert ist.

PPOL (R(2,NPP+1);0.) NPP ≤ 30
PPOL(1,j) - x-Koordinate des Punktes $\underline{P}_j$
PPOL(2,j) - z-Koordinate des Punktes $\underline{P}_j$
(s. Abb. 4.1; j = 1, ..., NPP+1)

PPOLM (R(2,NPP);0.) NPP ≤ 30
PPOLM(1,j) - x-Koordinate des Punktes $\underline{Q}_j$
PPOLM(2,j) - z-Koordinate des Punktes $\underline{Q}_j$
(s. Abb. 4.1; j = 1, ..., NPP)

IPOL (I(NPP);1) NPP ≤ 30
IPOL(j) = 1 - Konturkurve zwischen $\underline{P}_j$ und $\underline{P}_{j+1}$ ist ein Geradenstück
IPOL(j) = 2 - Konturkurve zwischen $\underline{P}_j$ und $\underline{P}_{j+1}$ ist ein Kreisbogen

(j = 1, ..., NPP)

NPPL (I(NPP);1) NPP ≤ 30

Anzahl der toroidalen Platten im j-ten poloidalen Abschnitt
(j = 1, ..., NPP)

ILEBN (I(2,2); ILEBN(*,1) = 1; ILEBN(*,2) = NPP)

Hat nur Bedeutung, wenn der Rand in neg. toroidaler Richtung ein e-Spiegel ist:

ILEBN(i,1) - Nummer des ersten poloidalen Abschnitts der i- ten Zusammenhangskomponente des leitenden Bereiches (i = 1,2)

ILEBN(i,2) - Nummer des letzten poloidalen Abschnitts der i- ten Zusammenhangskomponente des leitenden Bereiches (i = 1,2)

ILEBP (I(2,2); ILEBP(*,1) = 1; ILEBP(*,2) = NPP)

Hat nur Bedeutung, wenn der Rand in pos. toroidaler Richtung ein e-Spiegel ist:

ILEBP(i,1) - Nummer des ersten poloidalen Abschnitts der i- ten Zusammenhangskomponente des leitenden Bereiches (i = 1,2)

ILEBP(i,2) - Nummer des letzten poloidalen Abschnitts der i- ten Zusammenhangskomponente des leitenden Bereiches (i = 1,2)

NAMELIST /TOR/ (Subroutine DAT)

```
NAMELIST /TOR/ PTOR , NPTL
```

Bem.: Alle in dieser NAMELIST vorkommenden Felder haben eine feste Länge, die mittels PARAMETER (NPTM in Subroutine DAT) definiert ist.

PTOR (R(NPT-1);0.) NPT ≤ 30

PTOR(j) - Toroidale Winkelposition ϕ_j (in Grad) (s. Abb. 4.1)
(j = 1, ..., NPT-1)

NPTL (I(NPT);1) NPT ≤ 30

NPTL(j) - Anzahl der poloidalen Platten im toroidalen Abschnitt j
(j = 1, ..., NPT)

NAMELIST /STOFF/ (Subroutine DAT)

NAMELIST /STOFF/ D , RHO

Bem.: Diese NAMELIST muß nur dann eingelesen werden, wenn die Schalendicke oder der spezifische Widerstand des Schalenmaterials in einzelnen Bereichen von den in /NUM/ spezifizierten Werten DI, RHOI abweichen. Alle vorkommenden Felder haben feste Länge, die mittels PARAMETER definiert ist.

D (R(NPP,NPT);DI) NPP,NPT ≤ 30

D(i,j) - Schalendicke im poloidalen Abschnitt i, toroidalen Abschnitt j (i = 1, ..., NPP ; j= 1, ..., NPT)

RHO (R(NPP,NPT,2);RHOI) NPP,NPT ≤ 30

RHO(i,j,1) - Spezifischer Widerstand des Schalenmaterials in toroidaler Richtung im poloidalen Abschnitt i, toroidalen Abschnitt j [Ωm] (i = 1, ..., NPP ; j = 1, ..., NPT)

RHO(i,j,2) - Spezifischer Widerstand des Schalenmaterials in poloidaler Richtung im poloidalen Abschnitt i, toroidalen Abschnitt j [Ωm] (i = 1, ..., NPP ; j = 1, ..., NPT)

4.2 Preprozessor FLATPL zur Elementeinteilung einer ebenen, rechteckigen Scheibe

4.2.1 Behandelte Geometrie und Elementeinteilung

Die mit diesem Preprozessor zu behandelnde Geometrie zeigt Abb. 4.9. Es handelt sich um eine ebene, rechteckige Scheibe, deren Seiten parallel zur x- und y-Achse ausgerichtet sind, und deren Normale in z-Richtung zeigt. Wegen dieser strengen Festlegung der Koordinatenlage ist die Geometriebeschreibung der Scheibe sehr einfach. Wie beim Preprozessor TORUS (Kap. 4.1) wird wieder die Mittelebene der Struktur beschrieben, und zwar durch Angabe des linken, unteren Eckpunktes V, sowie die Ausdehnungen L_x, L_y der Scheibe in x- und y-Richtung. Außerdem kann sich in dieser Scheibe ein ebenfalls rechteckiges und parallel zu den Achsen ausgerichtetes Loch befinden, daß ebenfalls durch Angabe des linken, unteren Eckpunktes E und der Ausdehnungen $L_{o,x}$, $L_{o,y}$ spezifiziert wird.

Dies Loch kann an einer beliebigen Stelle im Inneren oder am Rand der Scheibe liegen. Die Koordinaten von E müssen dabei relativ zum Eckpunkt V der Scheibe angegeben werden (s. Abb. 4.9).

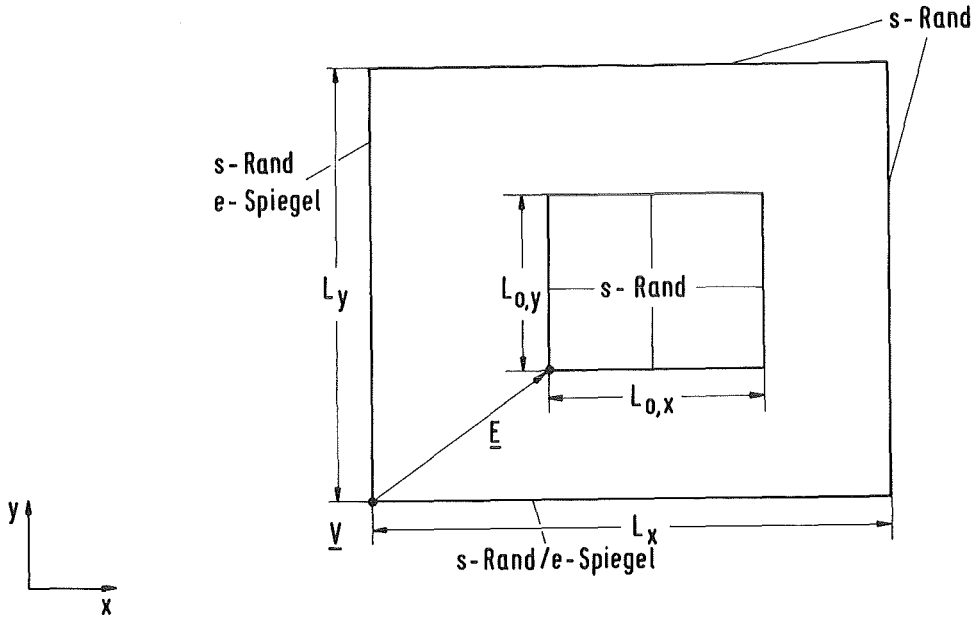


Abb. 4.9: Die mit dem Preprozessor FLATPL behandelte Geometrie

Die Randbedingungen können wie folgt gewählt werden: Der obere und der rechte Rand sind stets s-Ränder, während der untere und der linke Rand entweder als s-Ränder oder als e-Spiegel behandelt werden können. Da die letzteren beiden Randbedingungen unabhängig voneinander gewählt werden können, gibt es insgesamt vier mögliche Symmetriekonfigurationen, die Abb. 4.10 zeigt. Die Lochränder werden stets als s-Ränder betrachtet.

Bei der Elementeinteilung hat man gemäß Überlappungsprinzip zwei verschiedene Plattentypen, die x-Platten (Stromrichtung = x-Richtung) und die y-Platten (Stromrichtung = y-Richtung). Die Feinheit der Elementeinteilung wird durch drei Zahlen bestimmt: Die Anzahl der x-Platten, in die die Scheibe in x-Richtung (Parameter NX) und in y-Richtung (Parameter NY) zu unterteilen ist, sowie die Anzahl NLAY von Elementlagen in Dickenrichtung der Scheibe. Die Elementeinteilung in Dickenrichtung erfolgt dabei genauso wie bei der Torusschale (Kap. 4.1.3), die einzelnen Lagen sind nur induktiv miteinander gekoppelt.

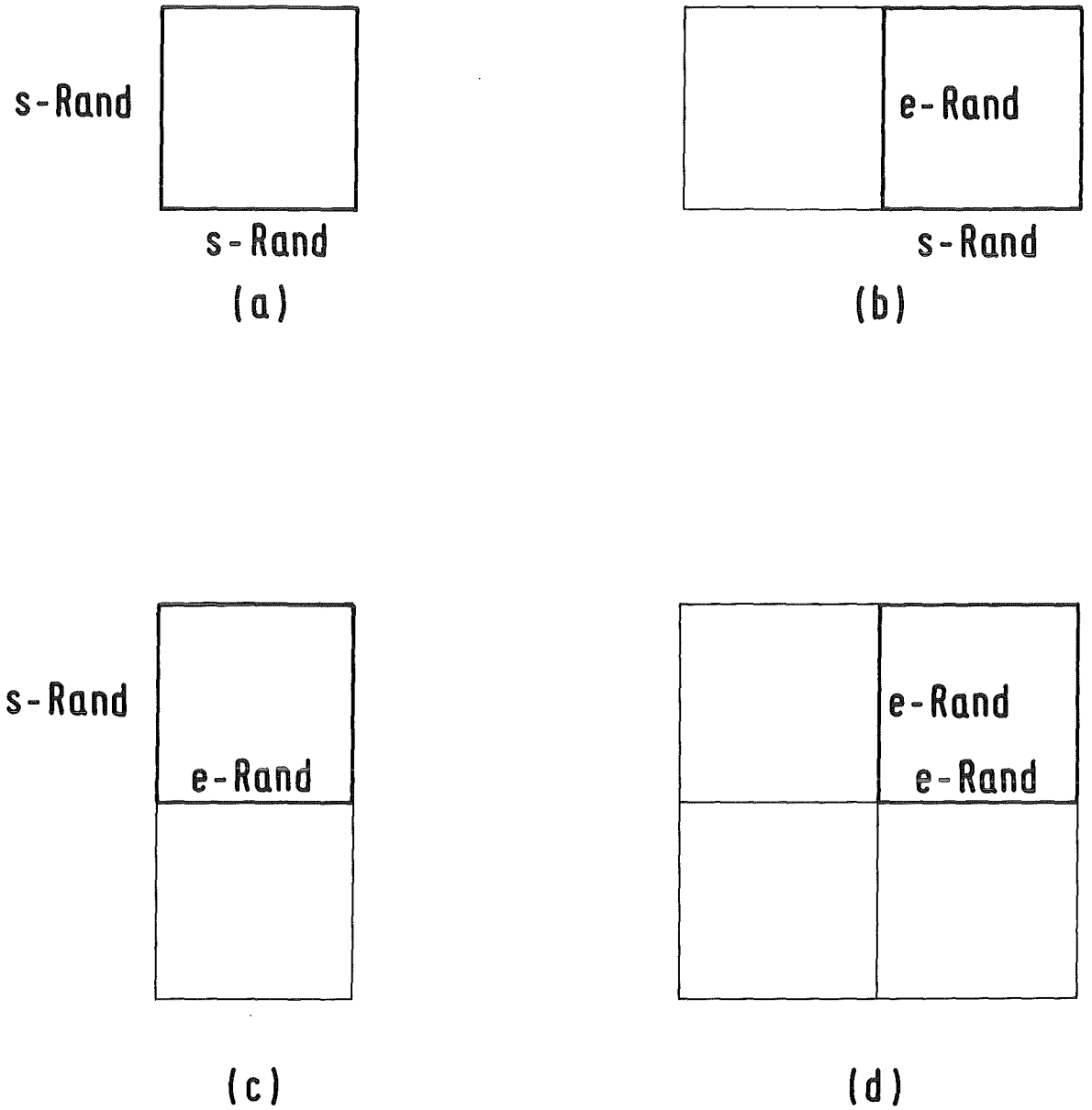
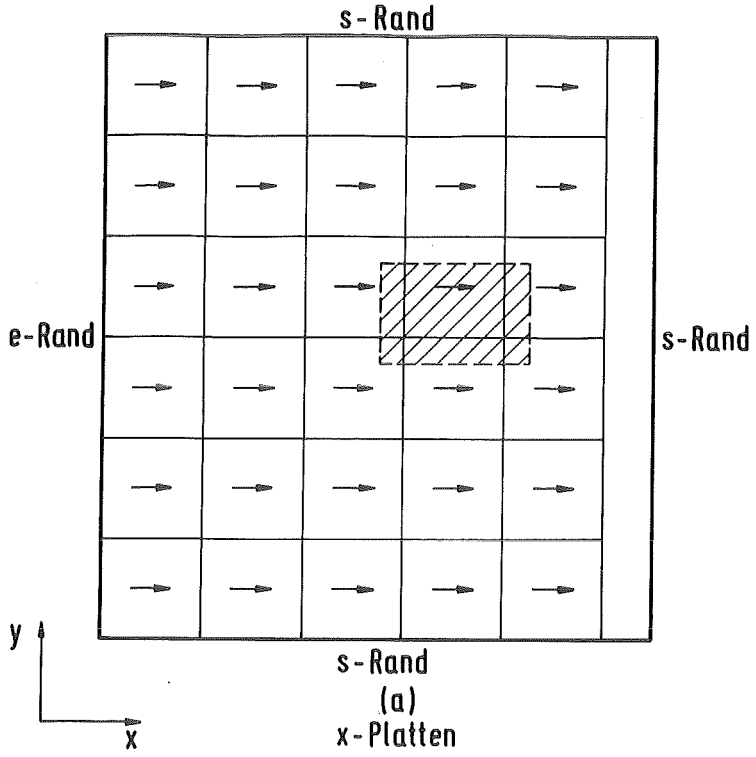
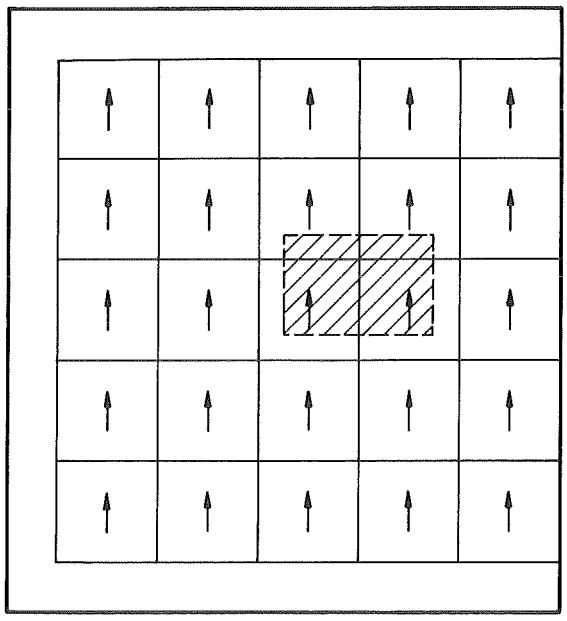


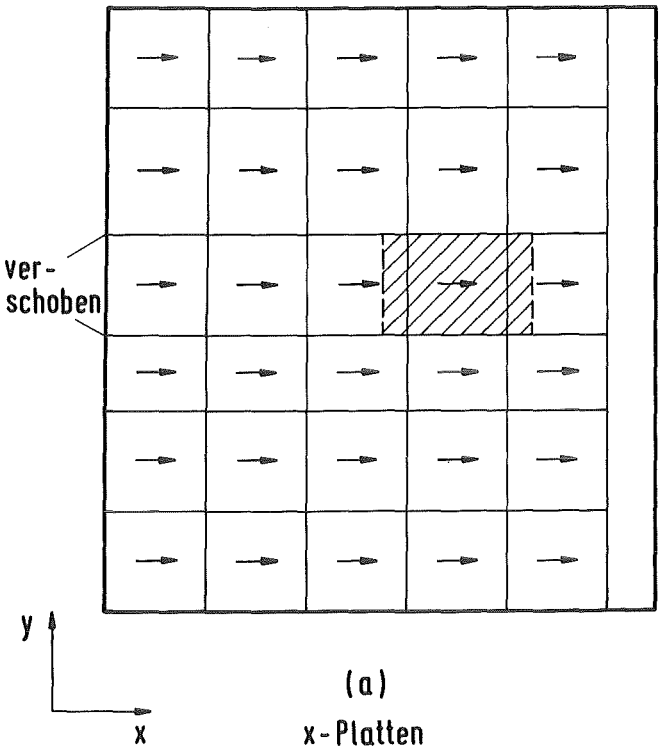
Abb. 4.10: Die 4 Symmetriekonfigurationen, die mit dem Preprozessor FLATPL behandelt werden können



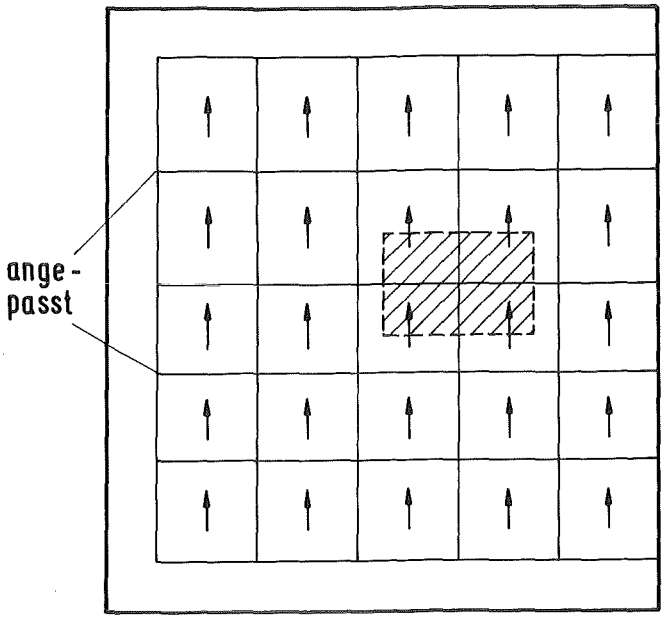


(b)
y-Platten

Abb. 4.11: Mit FLATPL erzeugte Elementeinteilung ohne Berücksichtigung eines Loches (NX = 5, NY = 6)

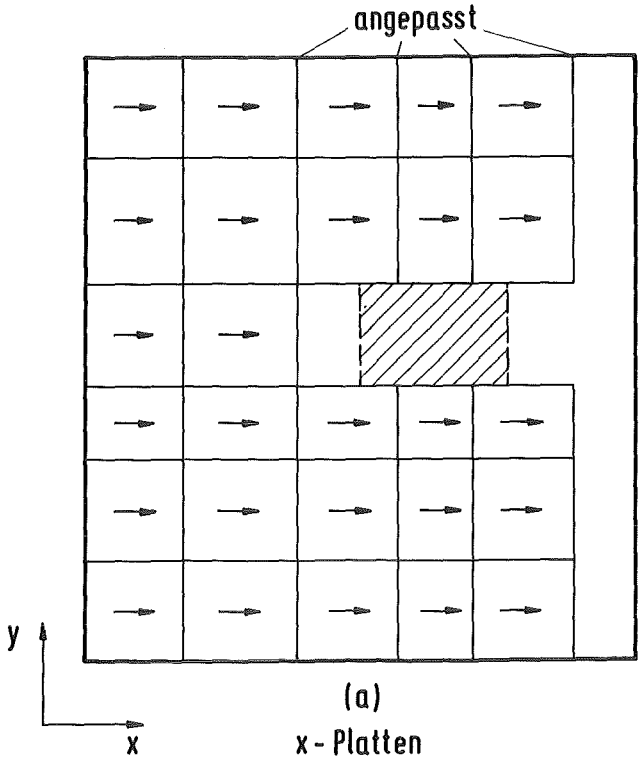


(a)
x-Platten



(b)
y-Platten

Abb. 4.12: Verschiebung der Plattenkanten in y-Richtung, so daß die y-Kanten der x-Platten bündig mit den Lochkanten übereinstimmen



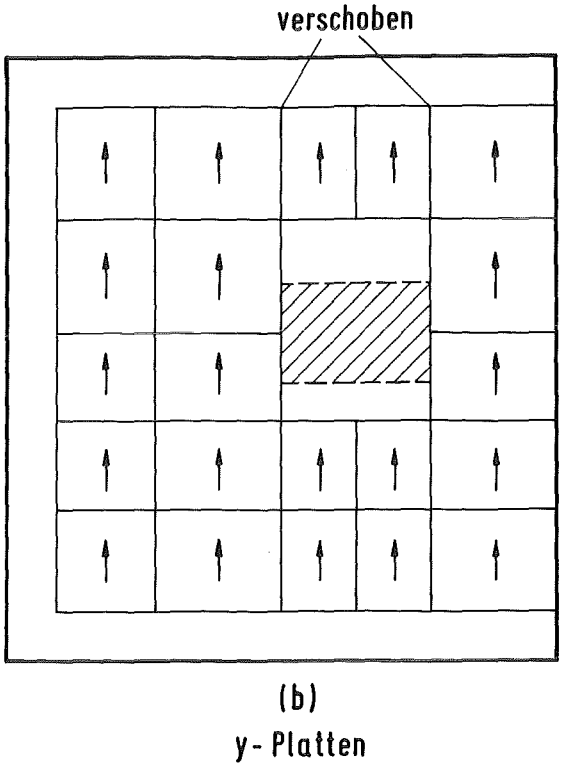


Abb. 4.13: Verschiebung der Plattenkanten in x-Richtung, so daß die x-Kanten der y-Platten bündig mit den Lochkanten übereinstimmen; Elimination der Platten im Lochbereich

Die Einteilung in x- und y-Richtung erfolgt mit äquidistanten Platten, solange kein Loch in der Scheibe vorhanden ist. Die Bestimmung einer Platteneinteilung bei Berücksichtigung eines Loches ist etwas komplizierter und soll im weiteren beschrieben werden.

Die Berechnung geht in mehreren Schritten vor sich. Zunächst wird im ersten Schritt das Vorhandensein eines Loches vollkommen ignoriert, und über die gesamte Scheibe eine äquidistante Elementeinteilung gemäß den spezifizierten Werten NX, NY, den Randbedingungen am linken und unteren Rand sowie den Forderungen des Überlappungsprinzips bestimmt (Abb. 4.11). Um das Loch zu berücksichtigen, müssen letztlich einige Platten aus dieser Einteilung entfernt werden. Vorher muß aber sichergestellt werden, daß die verbleibenden Platten richtig zu den s-Rändern des Loches liegen, also daß die Lochränder in x- (bzw. y-) Richtung bündig mit den x-Rändern der y-Platten (bzw. y-Rändern der x-Platten) übereinstimmen.

Dazu werden zunächst die y-Grenzen der x-Platten im Lochbereich so vorschoben, daß Plattengrenzen mit Lochgrenzen übereinstimmen (Abb. 4.12a). Danach werden die y-Grenzen der y-Platten so angepasst, daß das Überlappungsprinzip wieder erfüllt ist (Abb. 4.12b). Im nächsten Schritt erfolgt dann die Anpassung der x-Grenzen. Zunächst werden die x-Grenzen der y-Platten verschoben, bis Plattengrenzen mit Lochgrenzen übereinstimmen (Abb. 4.13b), und dann die x-Grenzen der x-Platten gemäß Überlappungsprinzip angepasst (Abb. 4.13a). Zum Schluß müßen nur noch die x- und y-Platten, die jetzt noch die Lochgrenzen überlappen, entfernt werden, und die Numerierung der Platten und Knoten entsprechend korrigiert werden.

4.2.2 Programmstruktur und Eingabebeschreibung

Der Preprozessor FLATPL besteht im wesentlichen aus einem einzigen Hauptprogramm. Die einzelnen Schritte bei der Erzeugung einer Elementeinteilung sind kurz zusammengefasst wie folgt:

- Dateneingabe, Konsistenzprüfengen und Ausdruck der Eingabedaten

- Berechnung der Einteilung (Plattengeometrie und Netzwerkstopologie) für eine Lage in Dickenrichtung, ohne Berücksichtigung eines Loches

- Eventuell Korrektur der Einteilung für die erste Lage, wenn ein Loch berücksichtigt werden soll (Fall LLOCH = .T.) durch Aufruf der Subroutine HOLE.

- Bei NLAY > 1 Bestimmung der Elementeinteilung für die restlichen Lagen

- Berechnung der Daten für den Postprozessor (s. Anhang 2), dabei Aufruf der Subroutine BOUND

- Ausgabe der Daten an Rechenprogramm und Postprozessor, sowie (bei LPRINT = .T.) Druckerausgabe

Die Speicherung der Daten erfolgt in Datenfeldern, deren obere Grenzen über PARAMETER-Statements fest dimensioniert werden. Es werden keine COMMON-Blöcke verwendet. Wegen der festen Dimensionierung gibt es eine Anzahl von Restriktionen, die durch Änderung des entsprechenden PARAMETER-Wertes allerdings leicht beeinflußt werden können:

- Maximale Plattenanzahl : 1000 (PARAMETER NPLMAX)
- Maximale Anzahl von Rändern : 40 (s. Anhang 2) (PARAMETER NBDMAX)
- Max. Anzahl regulärer Knoten: 500 (s. " 2) (PARAMETER NPRM)
- Max. Anz. irregulärer Knoten: 100 (s. " 2) (PARAMETER NPIRM)
- Max. Anz. entfernter Knoten : 50 (PARAMETER NKOM)
- Max. Anz. von Lagen : 5 (PARAMETER NLAYM)

Die Dateneingabe erfolgt ausschließlich über die NAMELIST /IN/. Die einzelnen Parameter werden im weiteren zusammen mit ihrem Typ und ihrer Initialisierung aufgeführt. Dabei wird durchgehend das MKSA-Einheitensystem verwendet.

NAMELIST /IN/

```
      NAMELIST /IN/ LX , LY , D , IRU , IRL , NX , NY , RHO , NLAY ,
     1      VX , VY , VZ , LPRINT , KENN , LLOX , LLOY , ELOX , ELOY ,
     2      LLOCH
```

LX (R,1.) LX > 0.
Ausdehnung der Scheibe in x-Richtung

LY (R,1.) LY > 0.

Ausdehnung der Scheibe in y-Richtung

D (R,0.01) D > 0.
Dicke der Scheibe

IRU (I,1)
IRU = 0 - Der untere Rand ist ein e-Spiegel
IRU ≠ 0 - Der untere Rand ist ein s-Rand

IRL (I,1)
IRL = 0 - Der linke Rand ist ein e-Spiegel
IRL ≠ 0 - Der linke Rand ist ein s-Rand

NX (I,1) NX > 0
Anzahl der x-Platten in x-Richtung

NY (I,1) NY > 0
Anzahl der x-Platten in y-Richtung

RHO (R,2.5E-8) RHO > 0.
Spez. Widerstand des Materials [Ωm]

NLAY (I,1) NLAY > 0
Anzahl der Lagen in Dickenrichtung der Scheibe

VX (R,0.)
x-Koordinate des unteren, linken Eckpunktes der Scheibe

VY (R,0.)
y-Koordinate des unteren, linken Eckpunktes der Scheibe

VZ (R,0.)
z-Koordinate des unteren, linken Eckpunktes der Scheibe

LPRINT (L,.TRUE.)
Bei LPRINT = .T. Ausgabe der berechneten Einteilung auf Drucker

KENN (C(8),'YYYYYYYY')
Textstring zur Identifikation des Preprozessor-Laufes

LLOCH (L,.FALSE.)

Nur bei LLOCH = .T. wird ein Loch bei der Elementeinteilung berücksichtigt

LLOX (R,0.) LLOX ≥ 0.

Ausdehnung des Loches in x-Richtung

LLOY (R,0.) LLOY ≥ 0.

Ausdehnung des Loches in y-Richtung

ELOX (R,0.) ELOX < LX

x-Koordinate der unteren, linken Ecke des Loches relativ zur unteren, linken Ecke der Scheibe V

ELOY (R,0.) ELOY < LY

y-Koordinate der unteren, linken Ecke des Loches relativ zur unteren, linken Ecke der Scheibe V

5. RECHENPROGRAMM ERATO

5.1 Globale Programmstruktur

5.1.1. MAIN-Programm

Die globale Struktur des Wirbelstrom-Berechnungsprogrammes ERATO ist ähnlich der Struktur des Preprozessors TORUS (s. Kap. 4.1.4). Das eigentliche Hauptprogramm, das bei jedem Rechenlauf mitübersetzt werden muß, ist sehr kurz:

```
      PROGRAM MAIN(UNIT5=INPUT , UNIT6=OUTPUT ,
     1  UNIT9=SAVE , UNIT10=RESTART , UNIT12=PLATEDAT , UNIT15=OUTPUT1 ,
     2  UNIT16=OUTPUT2 ,  UNIT25=MAGNET)
C
      PARAMETER (IRD = 5 , IWT = 6 , ISAVE = 9 ,
     1   IREST = 10 , IPLAT = 12 , IOUT1 = 15 , IOUT2 = 16 , IMAG = 25)
C
      PARAMETER (IDIMA = 4000000 , IDIMIA = 1000000)
C
      COMMON /COMA/  A(IDIMA)
      COMMON /COMIA/ IA(IDIMIA)
C
C
      CALL ERATO (IRD , IWT , ISAVE , IREST , IPLAT , IOUT1
     1   , IOUT2  , IMAG , A , IA , IDIMA , IDIMIA)
C
      STOP
      END
```

Es hat die folgenden Aufgaben:

- Definition und Initialisierung der globalen Speicherfelder A und IA

- Initialisierung der Unit-Nummern für die Ein/Ausgabekanäle

- Aufruf des Steuerprogrammes ERATO

Diese Aufgaben werden im folgenden präzisiert.

5.1.2 Speicherplatzverwaltung und REGION-Bedarf

Abgesehen von einigen COMMON-Blöcken und lokalen Variablen werden alle im ERATO Programm benötigten Datenfelder in den beiden Feldern A und IA gespeichert, dieselbe Technik wurde schon im Preprozessor TORUS (s. Kap. 4.1.4) benutzt. Der benötigte Speicherplatz ist von der Anzahl der finiten Elemente abhängig, außerdem ist er für die beiden verschiedenen Programmversionen (CYBER oder IBM-Version, s. Kap. 3.2) unterschiedlich. Tabelle 5.1 zeigt die in beiden Versionen benutzten Standardwerte, die damit mögliche maximale Elementanzahl, sowie den REGION-Bedarf eines Jobs mit dieser maximalen Elementanzahl. Man beachte, daß bei der IBM-Version wegen des auf maximal 6800 K beschränkten virtuellen Speicherplatzes eine Elementanzahl von 580 nicht überschritten werden kann. Bei der CYBER-Version gibt es eine derartige Beschränkung nicht.

	Größe A	Größe IA	Maximale Anzahl von Elementen	REGION-Bedarf
IBM-Version	700.000	90.000	580	6800K
CYBER-Version	4.800.000	4.000.000	1500	25000 Blöcke

Tabelle 5.1: Globale Datenspeicherung

Die dynamische Verwaltung des Speicherplatzes geschieht innerhalb des Steuerprogrammes ERATO. Dabei erhalten die von diesem Programm aufgerufenen Subroutinen per Parameterliste die Anfangsadressen im A und IA-Feld der von ihnen benötigten Datenfelder. Die zugehörigen Feldlängen werden über den COMMON-Block GLOBNM übergeben. Damit ist sichergestellt, daß in jedem Programmschritt nur so viel Speicherplatz allokiert wird, wie tatsächlich nötig ist. Die vollständige Speicherplatzbeschreibung findet sich als Kommentar am Beginn des Steuerprogrammes.

5.1.3 Ein/Ausgabekanäle

Es werden eine ganze Anzahl von Ein- und Ausgabekanälen benutzt, die

programmintern über die im MAIN-Programm definierten Variablennamen angesprochen werden. Die zugehörigen Dateien haben die folgenden Aufgaben (UNIT - Kanalnummer, IN - Variablenname, NAME - CYBER-Dateiname):

UNIT = 5 , IN = IRD , NAME = INPUT
Standard-Eingabedatei für NAMELIST-Input (s. Kap. 5.7)

UNIT = 6 , IN = IWT , NAME = OUTPUT
Drucker-Ausgabe

UNIT = 9 , IN = ISAVE , NAME = SAVE
Ausgabedatei für SAVE-Lauf

UNIT = 10 , IN = IREST , NAME = RESTART
Eingabedatei für RESTART-Lauf

UNIT = 12 , IN = IPLAT , NAME = PLATEDAT
Eingabedatei für Geometrie und Topologie der finiten Elemente
(= Ausgabedatei Preprozessor)

UNIT = 15 , IN = IOUT1 , NAME = OUTPUT1
Wird nur bei LPRINT(23) = .T. benutzt; Ausgabedatei für die berechneten Zweigmatrizen von Induktivität und Widerstand

UNIT = 16 , IN = IOUT2 , NAME = OUTPUT2
Ausgabedatei für die berechneten Wirbelströme (= Eingabedatei für die Postprozessoren)

UNIT= 25 , IN = IMAG , NAME = MAGNET
Wird nur benötigt, wenn ISTEPL = 5 ist; Ein/Ausgabedatei zur temporären Speicherung der berechneten Wirbelströme für die Magnetfeldsberechnung

5.1.4 Globale Ablaufsteuerung

Abb. 5.1 zeigt ein stark vereinfachtes Ablaufdiagramm des ERATO-Programmes. Das Gesamtprogramm läßt sich grob aufteilen in einen Dateneingabeteil sowie 5 Programmschritte, die in den folgenden Kapiteln näher behandelt werden.

Die gesamte Dateneingabe ist auf verschiedene Unterprogramme verteilt. So werden die wichtigsten Steuerungsparameter in der Subroutine HEADER (NAMELIST

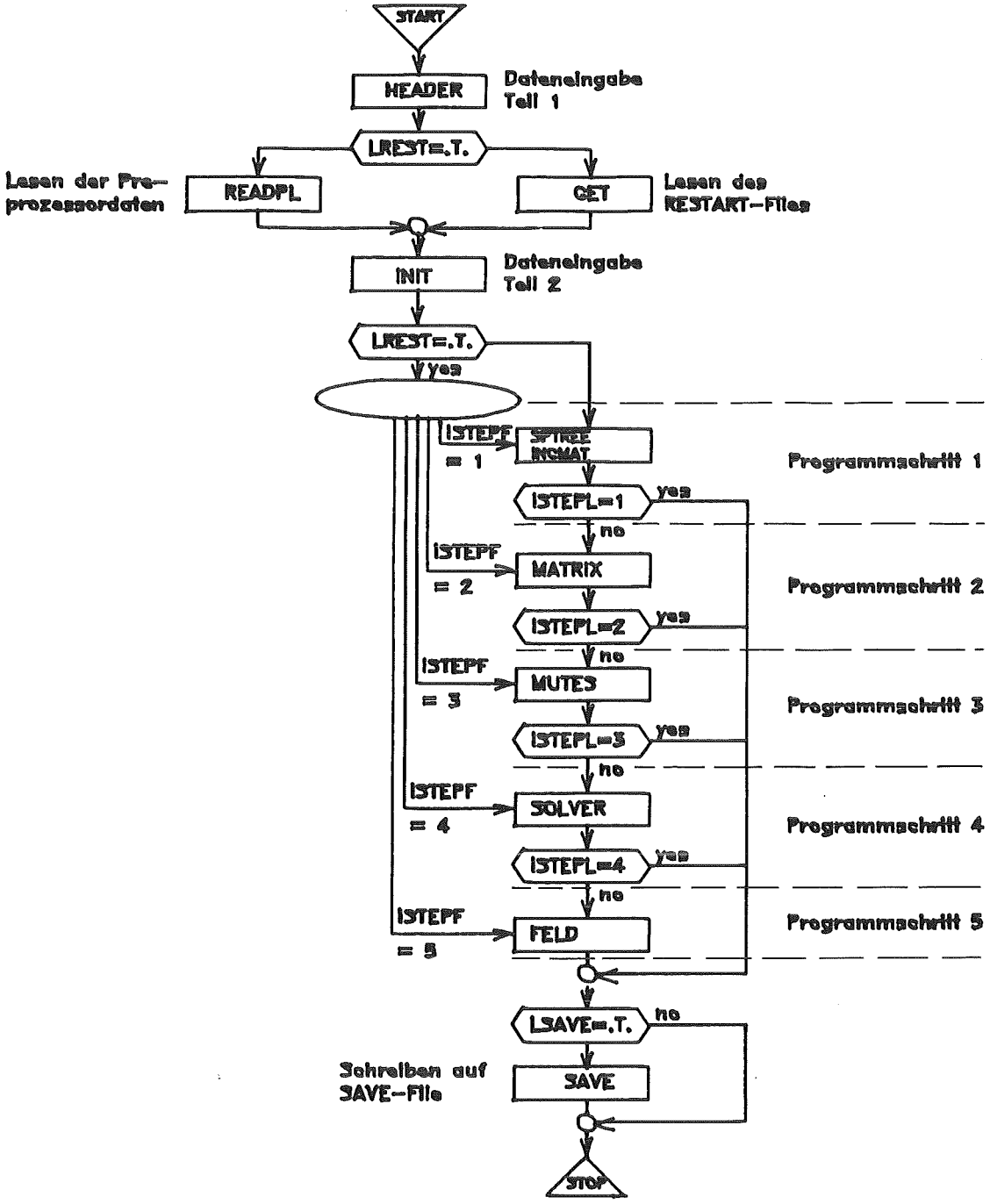


Abb. 5.1: Ablaufdiagramm des Rechenprogrammes ERATO

/DAT1/) eingelesen, und die restlichen NAMELIST-Blöcke (/DAT2/, /EXT1/, /EXT2/) in der Subroutine INIT (s. Kap. 5.7.1). Die Steuerung des globalen Programmablaufes geschieht im wesentlichen über die beiden LOGICAL-Variablen LREST und LSAVE, sowie die INTEGER-Variablen ISTEPF und ISTEPL.

Bei LREST = .T. handelt es sich um einen RESTART-Lauf, es werden dann mit der Subroutine GET die RESTART-Daten über Kanal 10 gelesen. Außerdem gibt in diesem Fall ISTEPF die Nummer des ersten auszuführenden Programmschrittes an. Bei LREST = .F. werden stattdessen die Daten des Preprozessors eingelesen (Subroutine READPL, Kanal 12) und auf Konsistenz überprüft (bei LKONS = .T.). Nach Beendigung der Dateneingabe wird in diesem Fall immer mit Programmschritt 1 begonnen. Der Parameter ISTEPL gibt die Nummer des letzten auszuführenden Programmschrittes an. Bei LSAVE = .T. werden vor Beendigung des Programmes noch alle Daten auf die SAVE-Datei über Kanal 9 geschrieben.

Weitere Steuerparameter, die insbesondere den Umfang der Druckerausgabe regeln, befinden sich im Feld LPRINT von logischen Variablen. Dies Feld gehört zum COMMON-Block PRT und ist in allen wichtigen Subroutinen verfügbar. Es kann über die NAMELIST /DAT1/ (s. Kap. 5.7.1) gesetzt werden, wobei eine Initialisierung so vorgenommen wurde, daß der Umfang der Druckerausgabe minimiert ist.

In den Programmschritten werden die folgenden Teilaufgaben der Wirbelstromanalyse durchgeführt:

Schritt 1 - Berechnung der Maschen-Inzidenzmatrix (Subroutinen SPTREE und INCMAT, s. Kap. 5.2)

Schritt 2 - Berechnung der Zweig- und Maschenmatrizen (Subroutine MATRIX, s. Kap. 5.3)

Schritt 3 - Berechnung der externen Induktivitätsmatrix (Subroutine MUTES, s. Kap. 5.4)

Schritt 4 - Zeitintegration der Netzwerksdifferentialgleichungen (Subroutine SOLVER, s. Kap. 5.5)

Schritt 5 - Berechnung des Magnetfeldes der Wirbelströme (Subroutine FELD, s. Kap. 5.6)

5.2 Programmschritt 1: Bestimmung der Maschen-Inzidenzmatrix

Die Bestimmung der Maschen-Inzidenzmatrix des elektrischen Netzwerkes geht in zwei Schritten vor sich. Zunächst wird in der Subroutine SPTREE ein spannender Baum des Netzwerksgraphen bestimmt, wobei der Algorithmus von Kruscal [13] benutzt wird. Als spannenden Baum bezeichnet man einen Untergraphen des ursprünglichen Graphen, der dieselben Knoten besitzt, aber Baumstruktur hat. Abb. 5.2 zeigt so einen spannenden Baum für ein Netzwerk, wie es typisch in ERATO vorkommt.

Eingabe für das Unterprogramm SPTREE ist der Netzwerksgraph in Form einer Kanten-Inzidenzmatrix. Dies ist ein zweidimensionales Feld, in dem für jede Kante die Nummer des zugehörigen Anfangs- und Endknotens angegeben ist; Abb. 5.3 zeigt die Kanten-Inzidenzmatrix des Graphen aus Abb. 5.2. Auch der von SPTREE berechnete Baum wird auf diese Art und Weise gespeichert. Der bearbeitete Graph braucht in übrigen nicht zusammenhängend zu sein, bei unzusammenhängenden Graphen bestimmt SPTREE einen spannenden Baum für jede Komponente.

Der so bestimmte spannende Baum (bzw. jede seiner Zusammenhangskomponenten) wird dann in der Subroutine INCMAT weiterverarbeitet, in der die Maschen-Inzidenzmatrix $\underline{\underline{H}}$ (s. Kap. 2.4, Def. (26)) aufgebaut wird. Dazu wird der Baum zunächst umgespeichert. Ein beliebiger Knoten i_0 wird als Wurzel des Baumes definiert. Nach einem bekannten Satz der Graphentheorie [13] gibt es dann von der Wurzel zu jedem beliebigen Knoten einen eindeutigen Weg. Ein Knoten i ist Vorgänger p(j) eines Knotens j (p(j) = i), wenn es eine Kante von i nach j gibt, und die Entfernung zwischen i_0 und i kleiner ist als die Entfernung zwischen i_0 und j ($1 \leq i,j \leq N$, $i \neq j$). Setzt man noch $p(i_0) = 0$, so ist mit dem Feld p die gesamte Baumstruktur eindeutig bestimmt (s. Abb. 5.4).

Zur entgültigen Berechnung der Maschen-Inzidenzmatrix werden jetzt nacheinander alle Netzwerkszweige betrachtet, die keine Baumzweige sind. Nimmt man einen derartigen Zweig in den Baum hinein, so entsteht eine Masche (s. Abb. 5.5). Die zu dieser Masche gehörigen Netzwerkszweige können folgendermaßen bestimmt werden: Man geht vom Anfangs- und Endknoten des hinzugenommenen Zweiges rückwärts in Richtung Wurzel, bis man auf einen gemeinsamen Knoten stößt (Knoten 2 in Abb. 5.5). Dies "Zurückgehen" läßt sich mit Hilfe des Feldes p (s.o.) sehr einfach ausführen. Da es sich bei dem elektrischen Netzwerk um einen gerichteten Graphen handelt, muß die Orientierung der Zweige relativ zur Maschenorientierung beachtet werden. Bei entgegengesetzter Orientierung

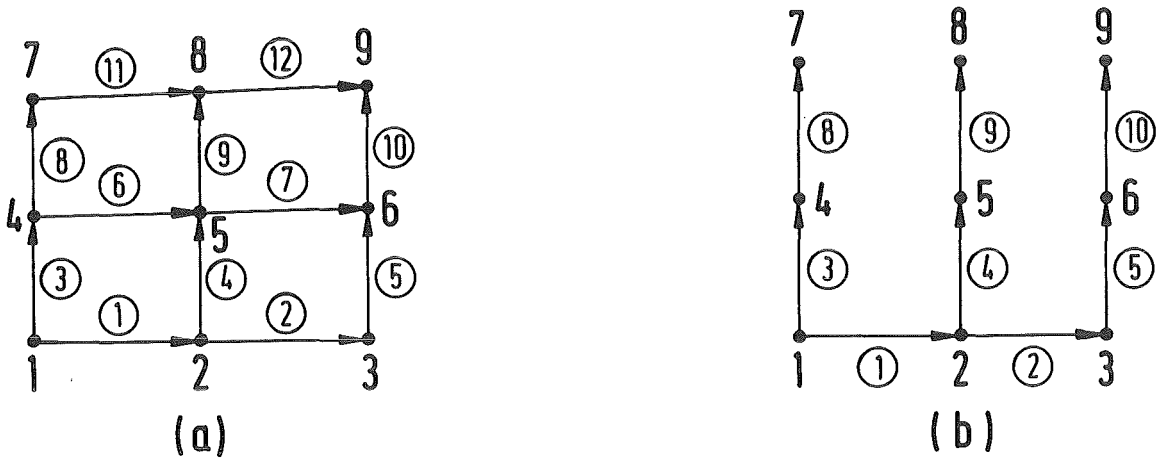


Abb. 5.2: (a) Typischer Netzwerksgraph; (b) Zugehöriger spannender Baum; Die Nummern der Zweige sind jeweils mir einem Kreis versehen

Kante	Anfangsknoten	Endknoten
1	1	2
2	2	3
3	1	4
4	2	5
5	3	6
6	4	5
7	5	6
8	4	7
9	5	8
10	6	9
11	7	8
12	8	9

Abb. 5.3: Kanten-Inzidenzmatrix des Graphen aus Abb. 5.2(a)

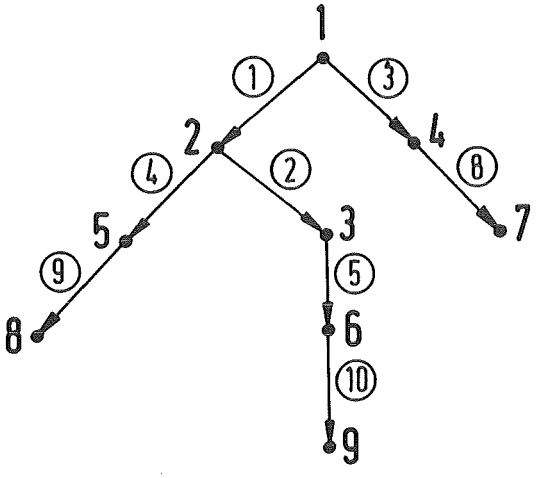


Abb. 5.4: Zu Abb. 5.2(b) isomorpher Baum; Die Struktur dieses Baumes wird durch das Feld p(9) beschrieben:
p = (0, 1, 2, 1, 2, 3, 4, 5, 6)

erhalten die entsprechenden Zweignummern ein negatives Vorzeichen (s. Abb. 5.5). Als Maschenorientierung wurde willkürlich die Orientierung des Nicht-Baumzweiges gewählt.

Die Gesamtheit der so erzeugten Maschen bildet dann einen vollständigen Satz unabhängiger Maschen [18]. Die Form, in der die Maschen-Inzidenzmatrix $\underline{\underline{H}}$ programmintern gespeichert wird, unterscheidet sich bei der CYBER- und der IBM-Version des Programmes. In der CYBER-Version werden $\underline{\underline{H}}$ und $\underline{\underline{H}}^T$ genau nach Definition (26) als schwachbesetzte N×M bzw. M×N Matrizen gespeichert. Dies ist vom Speicherplatz her zwar sehr ineffektiv, hat aber bei der Vektorisierung der Transformation von Zweig- auf Maschenmatrizen (s. Kap. 5.3.2) große Vorteile.

Für die IBM-Version wird dagegen eine kompaktere Speichermethode gewählt. Hierbei werden für jeden Zweig j (j = 1 ,..., N) die Nummern derjenigen Maschen gespeichert, zu denen dieser Zweig gehört. Die Orientierung der Zweige relativ zur Maschenorientierung wird wieder durch das Vorzeichen der Maschennummern beschrieben (s. Abb. 5.6).

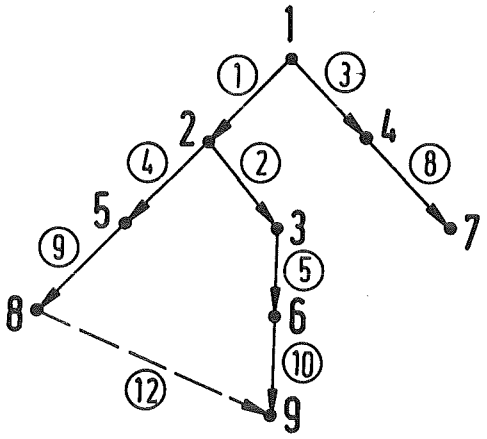


Abb.5.5: Bestimmung der Maschen-Inzidenzmatrix. Nach Hinzunahme eines Zweiges (Nr. 12) in den Baum entsteht eine Masche (Zweige 4, 9, 12, -10, -5, -2)

Zweig	Maschen	
1	-1	-3
2	-2	-4
3	1	3
4	-1	2
5	-2	-4
6	1	
7	2	
8	3	
9	-3	4
10	-4	
11	3	
12	4	

-3 4

Abb. 5.6: Maschen-Inzidenzmatrix in Zweigspeicherung. Masche 1 wird erzeugt durch Zweig 6, Masche 2 durch 7, Masche 3 durch 11, und Masche 4 durch 12

5.3 Programmschritt 2: Berechnung der Zweig- und Maschenmatrizen

Dieser Programmschritt, der von der Subroutine MATRIX gesteuert wird, ist der bei weitem rechenzeitaufwendigste Teil der gesamten Wirbelstromanalyse. Die Berechnung der Zweigmatrizen, ihre Transformation auf Maschenmatrizen und die weitere Bearbeitung ist auf mehrere Unterprogramme verteilt, die nacheinander angesprochen werden. Die dabei verwendeten Algorithmen werden im weiteren kurz vorgestellt.

5.3.1 Berechnung der Zweigmatrizen, Symmetriebedingungen

Die Zweig-Induktivitätsmatrix $\underline{\underline{L}}$ wird mit der Subroutine LBMAT berechnet. Dazu müßen die N^2 Größen L_{ij} gemäß (19) bestimmt werden, wobei sich der Integrand $\underline{A}_j$ nach der Formel (33) (bei dicken Platten, Eingabeparameter IDICK $\neq$ 0) oder der Formel (34) (bei dünnen Platten, IDICK = 0) ergibt.

Diese Berechnung stellt einen enormen numerischen Aufwand dar, wenn man bedenkt, daß sich im allgemeinen jeder Induktivitätskoeffizient L_{ij} aus einer Summe von Koeffizienten L_{ij}^{k} ergibt (Formel (32)), die die induktive Koppelung zwischen Elementen in verschiedenen Symmetriesegmenten beschreiben. Bei einer mittelfeinen Diskretisierung mit 800 Elementen und 12 Symmetriesegmenten kommt man so auf eine Zahl von 7.680.000 auszuwertenden Integralen. Hinzu kommt noch, daß die Auswertung der Integranden (33), (34) wegen der auftretenden transzendenten Funktionen Logarithmus und Arcus-Tangens numerisch relativ aufwendig ist.

Im ERATO-Programm können zwei verschiedene Symmetriebedingungen berücksichtigt werden. Dies ist einmal eine segmentweise Rotationssymmetrie in azimutaler Richtung, wie sie vor allem bei torusförmigen Objekten (s. Kap. 4.1.1) vorkommt. Die Anzahl NSEC der azimutalen Segmente wird dabei vom Preprozessor übergeben. Weiter kann mit ERATO automatisch eine Spiegelsymmetrie bezüglich der Ebene z=0 berücksichtigt werden, der zugehörige Steuerparameter ISYM kommt ebenfalls vom Preprozessor. Bei ISYM $\neq$ 0 wird die Spiegelsymmetrie von Geometrie und Anregung automatisch berücksichtigt, bei ISYM = 0 dagegen nicht. Die Berechnung der Elementeinteilung in den Symmetrieabschnitten aus der vom Preprozessor gelieferten Einteilung des Grundabschnittes geschieht in der Subroutine SECCOR.

In manchen Fällen kann es statthaft sein, sehr weit vom Grundsegment entfernte Abschnitte bei der induktiven Koppelung zu vernachlässigen, um so Rechenzeit

zu sparen. Die Gesamtzahl NIND der zu berücksichtigenden azimutalen Segmente ist deshalb ein Eingabeparameter.

Es war das Ziel bei der Entwicklung der CYBER-Version des ERATO-Codes, die Subroutine LBMAT möglichst optimal zu vektorisieren. Dies Ziel konnte durch die Verwendung einer speziellen Datenstruktur für die Geometriedaten weitgehend erreicht werden. Es ist möglich, die einzelnen Spalten der Matrix $\underline{\underline{L}}$ voll vektorisiert zu berechnen. Bei dem entwickelten Algorithmus werden extensiv die Möglichkeiten des CYBER Fortran-Compilers (Vektorsyntax, Vektorfunktionen, etc.) ausgenutzt.

Bei der IBM-Version müssen die Elemente L_{ij} sequentiell berechnet werden. Allerdings wird die Symmetrie der Induktivitätsmatrix ausgenutzt und nur das obere Dreieck der Matrix wirklich berechnet. Die numerische Auswertung der Integrale geschieht in beiden Programmversionen mit Gauss'schen Quadraturformeln [20], deren Grad von der Entfernung der zu koppelnden Elemente abhängt.

Die Bestimmung der Zweig-Widerstandsmatrix $\underline{\underline{R}}$ (Subroutine RBMAT) ist trivial, da der benötigte ohmsche Widerstand der finiten Elemente (Formel (36)) schon im Preprozessor berechnet wurde.

5.3.2: Transformation der Zweigmatrizen auf Maschenmatrizen

Auch für die in der Subroutine MESHTR durchgeführte Transformation von Zweig- auf Maschenmatrizen (Gleichung (30)) gibt es bei der IBM- und der CYBER-Version unterschiedliche Verfahren. Das Problem bei dieser Transformation ist weniger die benötigte Rechenzeit als die beteiligten sehr großen Datenmengen. Nun wird allerdings im Prinzip die Verwaltung des benötigten Speicherplatzes von den virtuellen Betriebssystemen der CYBER bzw. IBM erledigt. Bei sehr großen Datenfeldern wie im vorliegenden Fall kann die Speicherplatzverwaltung aber sehr ineffektiv werden, wenn die Daten mehr oder weniger random angesprochen werden. Es kann dann nämlich passieren, daß bei jeder neuen arithmetischen Operation ein neuer Datenbereich von der Platte in den Zentralspeicher geladen werden muß, was zu sehr langen Verweilzeiten und hohen Rechenkosten führt. Anzustreben ist deshalb, die Datenfelder möglichst sequentiell, in der Reihenfolge ihrer Abspeicherung, zu durchlaufen.

Bei der CYBER-Version hat es sich nach längeren Untersuchungen als optimal herausgestellt, sowohl die Maschen-Inzidenzmatrix $\underline{\underline{H}}$ als auch $\underline{\underline{H}}^T$ als voll besetzte Matrizen zu speichern, und die zur Transformation (30) nötigen

Matrixmultiplikationen mit der Subroutine MMUL2 durchzuführen. Dies ist ein vom Rechenzentrum der Uni Karlsruhe (Bibliothek RZLIB) geschriebenes Programm, das einerseits sehr gut vektorisiert ist, andererseits aber auch den jeweils zur Verfügung stehenden realen Speicherplatz optimal ausnutzt.

Für die IBM-Version wurde eine spezielle Matrixmultiplikationsroutine entwickelt, die die in Kap. 5.2 beschriebene kompakte Speicherung der Maschenin zidenzmatrix $\underline{\underline{H}}$ unterstützt. Im Vergleich mit dem von Christensen [4] entwickelten Verfahren hat dieser Algorithmus den Vorteil, daß die sehr große Zweig-Induktivitätsmatrix $\underline{\underline{L}}$ sequentiell durchlaufen wird.

5.3.3 Inversion der Maschen-Induktivitätsmatrix

Bei der Zeitintegration der Netzwerks-Differentialgleichungen (s. Kap. 5.5) wird das System (29) nicht direkt behandelt, sondern zunächst durch Multiplikation von links mit $(\underline{\underline{L}}_m)^{-1}$ auf eine Normalform gebracht. Zur Inversion der Maschen-Induktivitätsmatrix (Subroutine LMINV) wird $\underline{\underline{L}}_m$ als erstes so normalisiert, daß das maximale Matrixelement =1 ist. Die Inversion der normalisierten Matrix wird dann bei der CYBER-Version mit der RZLIB-Routine MINV1 durchgeführt. Auf der IBM wird das selbst geschriebene Programm MATINV verwendet, das mit einer Dreieckszerlegung ohne Pivotsuche arbeitet. Auf Wunsch (LPRINT(19) = .T.) kann auch die Qualität der Matrixinversion durch Multiplikation der invertierten Matrix mit der Ausgangsmatrix getestet werden. Es werden in diesem Falle die maximalen Fehler in der Haupt- und Nebendiagonalen ausgegeben.

5.3.4 Bestimmung von Zeitkonstanten

Es ist möglich, mit der Subroutine TKON eine Analyse der Zeitkonstanten der wirbelstromführenden Struktur durchzuführen, dazu muß der Parameter LPRINT(17) = .T. gesetzt sein. Diese Zeitkonstanten sind definiert als inverse Eigenwerte der Matrix $\underline{\underline{L}}_m^{-1}\underline{\underline{R}}_m$. Es muß an dieser Stelle allerdings darauf hingewiesen werden, daß die Aussagekraft dieser Konstanten nicht allzu hoch ist. Bei einem konkreten Felddiffusionsproblem werden nämlich i. A. viele der Modes angeregt, die durch diese Zeitkonstanten beschrieben werden, und die Wirbelströme der Einzelmodes überlagern sich. Um die Zeitkonstanten einer Struktur für eine konkrete Anregungssituation zu bestimmen, muß deshalb in der Regel das Abklingverhalten der Stromverläufe in der Struktur selbst untersucht werden.

Zur Durchführung der Eigenwertanalyse wird die Matrix $\underline{\underline{L}}_m^{-1}\underline{\underline{R}}_m$ zunächst mit einer Ähnlichkeitstransformation symmetrisiert. Als erstes wird dazu eine Cholesky-Zerlegung [19] der symmetrischen und positiv definiten Matrix $\underline{\underline{R}}_m$ vorgenommen:

$$\underline{\underline{R}}_m = \underline{\underline{U}}_m \underline{\underline{U}}_m^T \tag{38}$$

Danach wird die Ähnlichkeitstransformation mit $\underline{\underline{U}}_m^T$ durchgeführt:

$$\underline{\underline{U}}_m^T(\underline{\underline{L}}_m^{-1}\underline{\underline{R}}_m)(\underline{\underline{U}}_m^T)^{-1} = \underline{\underline{U}}_m^T(\underline{\underline{L}}_m^{-1}\underline{\underline{U}}_m\underline{\underline{U}}_m^T)(\underline{\underline{U}}_m^T)^{-1} = \underline{\underline{U}}_m^T\underline{\underline{L}}_m^{-1}\underline{\underline{U}}_m \tag{39}$$

wobei sich eine symmetrische Matrix ergibt. Da die Durchführung einer Ähnlichkeitstransformation nichts an den Eigenwerten ändert, hat diese symmetrische Matrix dieselben Eigenwerte wie $\underline{\underline{L}}_m^{-1}\underline{\underline{R}}_m$.

Die Cholesky-Zerlegung von $\underline{\underline{R}}_m$ sowie die Eigenwertanalyse der symmetrisierten Matrix werden mit Bibliotheksroutinen durchgeführt (CYBER-Version: Subroutine LGSSP (RZLIB-Bibliothek) für Zerlegung, Subroutine F02ABF (NAG-Bibliothek) für Eigenwertanalyse; IBM-Version: Subroutine LUDECP für Zerlegung, Subroutine EIGRE für Eigenwertanalyse (beide IMSL-Bibliothek)). Bei LPRINT(20) = .T. wird außerdem noch die Genauigkeit der Eigenwertanalyse überprüft.

5.4: Programmschritt 3: Wirbelstromanregung und Berechnung der externen Induktivitätsmatrix

Mit Hilfe der Matrix $\underline{\underline{L}}_{ex}$ der externen Induktivitäten (20) wird die Anregung der Wirbelströme beschrieben. Diese Matrix wird von der Subroutine MUTES berechnet. Im ERATO-Programm kann die Anregung dabei auf zwei verschiedene Art und Weisen erfolgen, wobei auch beide Anregungsarten gleichzeitig auftreten können:

(a) Durch zeitlich stetig oder sprunghaft veränderliche Ströme in ortsfesten, kreisförmigen Leitern

(b) Durch ein zeitlich stetig oder sprunghaftes veränderliches, räumlich homogenes Magnetfeld

Im Falle (a) wird jeder einzelne erregende Stromkreis durch die folgenden Größen beschrieben (s. Abb. 5.7): Die Koordinate $\underline{c}$ des Kreismittelpunktes, die

Normale $\underline{n}$ der Kreisfläche, den Radius r, sowie die Anzahl der Windungen w. Mit diesen Angaben läßt sich $\underline{\underline{L}}_{ex}$ dann nach (20) berechnen, wobei sich der Integrand aus Formel (37) ergibt.

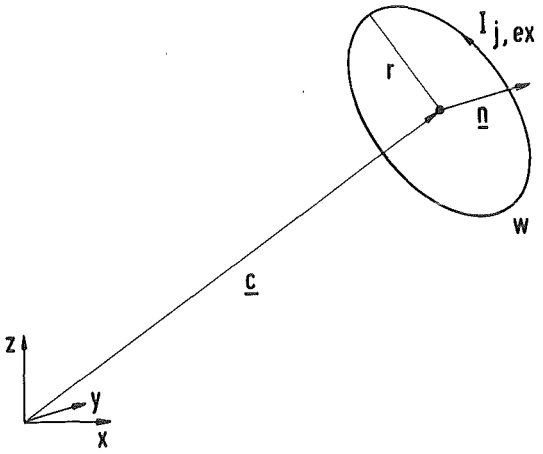


Abb. 5.7: Geometrie der äußeren Anregung: Kreisförmige Spulen beliebiger Lage und Orientierung

Neben der geometrischen Gestalt der erregenden Stromkreise muß allerdings auch noch die Zeitfunktion der Stromänderung $\underline{I}^{\bullet}_{ex}$ spezifiziert werden (s. Gleichung (29)). Dazu sind die externen Stromkreise in einzelne Kreisgruppen aufgeteilt, von denen es eine beliebige Anzahl geben kann. Alle Stromkreise innerhalb einer Kreisgruppe j (auch hier sind beliebig viele möglich) haben dieselbe Funktion $I^{\bullet}_{j,ex}$ für die zeitliche Stromänderung ($j = 1, \ldots, N_{ex}$). Für diese Funktionen muß ein eigenes Unterprogramm SAD zur Verfügung stehen, das (zusammen mit dem Hauptprogramm) bei jedem Rechenlauf mit übersetzt werden muß. Die Struktur dieses Unterprogrammes ist in Kap. 5.7.2 definiert.

Alle externen Induktivitätskoeffizienten, die zu Stromkreisen einer bestimmten Kreisgruppe j gehören, können somit bei der Aufstellung der externen Induktivitätsmatrix $\underline{\underline{L}}_{ex}$ zusammengefaßt werden, da sie alle dieselbe Stromänderungsfunktion $I^{\bullet}_{j,ex}$ aufweisen. Die Anzahl N_{ex} der Spalten von $\underline{\underline{L}}_{ex}$ entspricht also der Anzahl der Kreisgruppen, und nicht der Anzahl der externen Kreise. Die Stromkreise mit sprunghafter Stromänderung bilden formal eine eigene Kreisgruppe, ihnen ist damit eine eigene Spalte in der $\underline{\underline{L}}_{ex}$-Matrix zugeordnet. Es wird vorausgesetzt, daß diese Stromänderung in den zugehörigen externen Stromkreisen zur Zeit $t = t_0$ durch einen Sprung vom Wert $I = I_0$ auf $I = 0$ erfolgt.

Soll die Wirbelstromanregung durch direkte Vorgabe eines Magnetfeldes erfolgen (Fall (b)), so ist Gleichung (20) nicht unmittelbar zur Berechnung der Matrix $\underline{\underline{L}}_{ex}$ anwendbar. Für den Fall einen räumlich homogenen Magnetfeldes $\underline{B}$ kann man allerdings sehr einfach ein Vektorpotential $\underline{A}_B$ finden mit $\underline{B} = \mathrm{rot}(\underline{A}_B)$. Für den in ERATO vorausgesetzten Fall einen homogenen Feldes in z-Richtung ($\underline{B} = (0,0,B_z)^T$) lautet dies Potential z. B.:

$$(40) \qquad \underline{A}_B = (-\tfrac{1}{2}y, \tfrac{1}{2}x, 0)^T \, B_z$$

Man kann nun sehr einfach die Anregung durch ein homogenes Magnetfeld formal in die Zweigform (23) der Netzwerksdifferentialgleichungen integrieren. Dazu definiert man die für die Koppelung zuständigen externen Induktivitäts-koeffizienten (die es im physikalischen Sinne allerdings nicht sind) als:

$$(41) \qquad L_{i,B} = (1/B_z) \int_{S_i} \underline{A}_B \, ds \quad (i = 1, \ldots, N)$$

und erweitert $\underline{\underline{L}}_{ex}$ um diese Spalte. Genauso wird der Vektor $\underline{I}_{ex}$ (24e) formal erweitert um B_z:

$$(42) \qquad \underline{I}_{ex} = (I_{1,ex}, \ldots, I_{N_{ex},ex}, B_z)^T$$

Dadurch tritt auch ein erregendes transientes Magnetfeld im Programm formal als eine spezielle Kreisgruppe auf.

Die Berechnung des Integrales in (20) ist wegen der Komplexität des Ingegranden zumindest im Fall (a) nur numerisch möglich. Die Auswertung der in (37) auftretenden elliptischen Integrale (Subroutine DCEL1 und DCEL2) geschieht mit einem Verfahren von Bulirsch [20]. Das Integral selber wird mit einer Kombination aus Simpson-Regel und Newtons 3/8-Regel berechnet (Subroutine DQSF) [21], das Verfahren ist von 5. Ordnung.

5.5 Programmschritt 4: Zeitintegration der Netzwerksdifferentialgleichungen

In der Subroutine SOLVER wird das System (29) von gewöhnlichen Differentialgleichungen integriert. Dazu erfolgt zunächst eine Multiplikation dieser Gleichung von links mit $(\underline{\underline{L}}_m)^{-1}$:

$$(43) \qquad \underline{I}_m^{\bullet} + \underline{\underline{L}}_m^{-1} \underline{\underline{R}}_m \underline{I}_m = -\underline{\underline{L}}_m^{-1} \underline{\underline{L}}_{m,ex} \underline{I}_{ex}^{\bullet} \qquad (t \geq t_0)$$

Bei (43) handelt es sich um ein System gewöhnlicher Differentialgleichungen 1. Ordnung. Zur eindeutigen Lösbarkeit muß deshalb noch ein Anfangswert $\underline{I}_m(t=t_0)$ für den induzierten Strom bestimmt werden. Hierbei sind drei Fälle zu unterscheiden.

Findet die Anregung durch transiente Ströme in externen Leitern statt, die zur Zeit $t=t_0$ stetig von Null auf einen endlichen Wert ansteigen, so ergibt sich als Anfangswert sofort:

$$(44a) \qquad \underline{I}_m(t=t_0) = 0$$

Dieselbe Anfangsbedingung gilt auch, wenn die Wirbelstromanregung durch ein homogenes Magnetfeld mit stetiger Zeitänderungsfunktion erfolgt.

Ein anderer Fall liegt vor, wenn in diesen externen Leitern der Strom zur Zeit $t=t_0$ sprunghaft vom Wert I_0 auf 0 sinkt. Diesen Leitern sei die Kreisgruppe $i=N_{ex}$ (s. Kap. 5.4) zugeordnet. Betrachtet man dann nur die Anregung durch diese Kreisgruppe, so lautet Gleichung (43):

$$(45) \qquad \underline{I}_m^{\bullet} + \underline{\underline{L}}_m^{-1} \underline{\underline{R}}_m \underline{I}_m = \underline{\underline{L}}_m^{-1} \underline{L}_{i,ex} I_0 \, \delta(t-t_0)$$

wobei $\underline{L}_{i,ex}$ die i-te Spalte der Matrix $\underline{\underline{L}}_{m,ex}$ ist, und $\delta(t-t_0)$ die Dirac-Funktion. Durch Zeitintegration von (45) von t_0 bis t und Grenzübergang $t \to t_0$ erhält man dann:

$$(44b) \qquad \underline{I}_m(t=t_0) = \underline{\underline{L}}_m^{-1} \underline{L}_{i,ex} I_0$$

Sehr ähnlich ist der Fall, daß die Wirbelstromanregung durch ein homogenes Magnetfeld in z-Richtung erfolgt, das zur Zeit $t=t_0$ sprunghaft vom Wert B_z auf 0 abfällt. Es ergibt sich dann:

$$(44c) \qquad \underline{I}_m(t=t_0) = \underline{\underline{L}}_m^{-1} \underline{L}_{j,ex} B_z$$

wobei $\underline{L}_{j,ex}$ die entsprechende Spalte in der Matrix $\underline{\underline{L}}_{m,ex}$ ist (s. Kap. 5.4).

Nach Festlegung der Anfangswerte durch (44) kann (43) schließlich integriert werden. Da die Matrix $\underline{\underline{L}}_m$ im Allgemeinen voll besetzt ist, kann diese Integration nur numerisch erfolgen (Subroutine DREBS). Es wird dazu ein Verfahren von Bulirsch und Stoer [22] verwendet, das mit rationaler Extrapolation 6. Ordnung, Genauigkeitskontrolle und Schrittweitensteuerung arbeitet. Der Integrationszeitraum und der Genauigkeitsparameter FREL sind frei wählbar. In gleicher Weise kann per Eingabe spezifiziert werden, zu welchen Zeiten die berechneten transienten Maschenströme in Zweigströme transformiert werden und zur Postprozessor-Auswertung auf Kanal 16 geschrieben werden (Subroutine OUTP).

5.6: Programmschritt 5: Magnetfeldberechnung

Der letzte Schritt der Wirbelstromanalyse (Subroutine FELD) ist es, das von den induzierten Wirbelströmen erzeugte sekundäre Magnetfeld zu berechnen. Dazu werden die Mittelachsen der finiten Elemente als eindimensionale, geradlinige Leiter betrachtet, in denen die in Schritt 4 berechneten Zweigströme $\underline{I}$ fließen. Das Magnetfeld eines solchen Leiters mit Anfangspunkt $\underline{P}_a$ und Endpunkt $\underline{P}_e$ an einem beliebigen Aufpunkt $\underline{P}$ (Abb. 5.8) läßt sich mit einer Formel von Preis [23] berechnen:

$$(46) \quad \underline{B}_{(P)} = \frac{\mu_0 I}{4\pi} * \frac{\underline{s}_a \times \underline{s}_e}{|\underline{s}_a|\,|\underline{s}_e|} .* \frac{|\underline{s}_a| + |\underline{s}_e|}{|\underline{s}_a|\,|\underline{s}_e| + \underline{s}_a \cdot \underline{s}_e}$$

$$(\underline{s}_a = \underline{P}_a - \underline{P} \; ; \; \underline{s}_e = \underline{P}_e - \underline{P})$$

Diese Formel wird innerhalb eines Zylinderschalensegmentes ausgewertet, das von den Radien r_a, r_e, den Winkeln ϕ_a, ϕ_e, und den axialen Positionen z_a, z_e begrenzt ist (Abb. 5.9). Die Anzahlen n_r, n_ϕ, n_z der Auswertepositionen für die drei Koordinatenrichtungen sind ebenfalls Eingabeparameter, dabei ist die Unterteilung für jede Richtung äquidistant.

5.7 Eingabebeschreibung

Der Wirbelstrom-Berechnungsprogramm ERATO erhält Eingabedaten auf verschiedene Art und Weise. Der größte Teil der Daten wird über NAMELIST-Blöcke übergeben, die im Kap. 5.7.1 näher beschrieben werden. Hier ist bei jeder

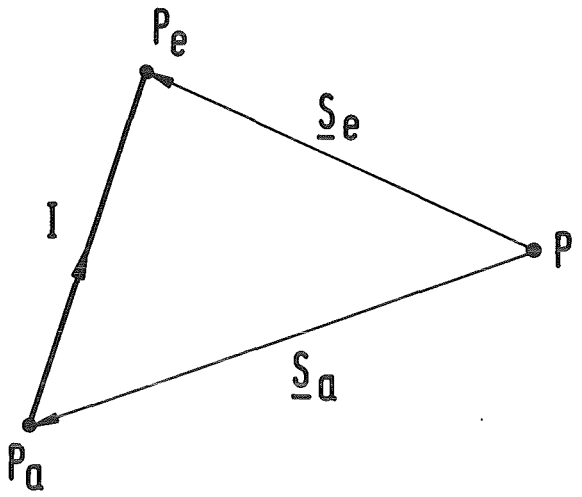


Abb. 5.8: Magnetfeld eines geradlinigen Leiters

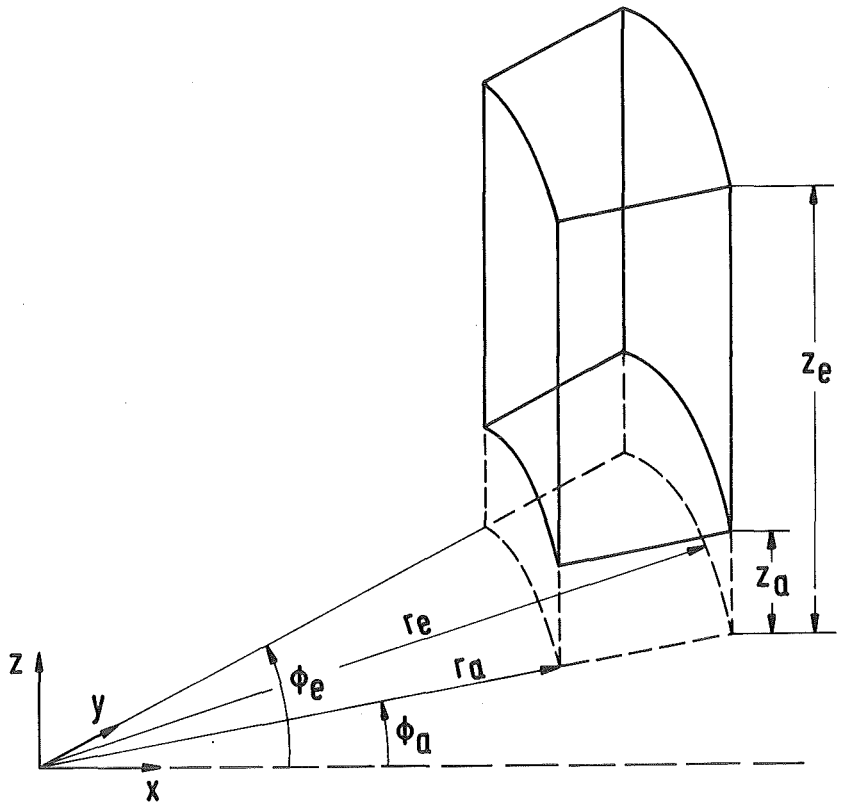


Abb. 5.9: Auswertung des induzierten sekundären Magnetfeldes

NAMELIST-Variablen neben einem Initialisierungswert auch der Variablentyp angegeben. Ist ein einziger Wert für die Initialisierung eines Feldes angegeben, so wird das gesamte Feld mit diesem Wert initialisiert. Durchgehend wird das MKSA-Einheitensystem verwendet.

Die Zeitfunktion der Stromänderung in den externen Stromkreisen wird durch die Subrouroutine SAD spezifiziert, die Struktur dieses Unterprogrammes ist in Kap. 5.7.2 näher erläutert.

Schließlich erhält ERATO noch Eingabedaten, die direkt von Rechenprogrammen erzeugt wurden. Dies sind entweder die Geometriedaten des Preprozessors (Fall LREST = .FALSE.) oder die SAVE-Daten eines vorhergegangenen ERATO-Laufes (Fall LREST = .TRUE.). Die Preprozessor-Schnittstelle wird in Anhang 1 beschrieben.

5.7.1 NAMELIST-Eingabe

NAMELIST /DAT1/ (Subroutine HEADER)

```
      NAMELIST /DAT1/ RUNIDL , LSAVE , LREST , ISTEPF , IPRINT ,
     1                ISTEPL , COMM , LKONS
```

RUNIDL (C*8;'XXXXXXXX')
Textstring zur Identifikation des ERATO-Laufes

LSAVE (L;.FALSE.)
Bei LSAVE = .TRUE. werden alle relevanten Daten vor Programmende auf die SAVE-Datei geschrieben

LREST (L;.FALSE.)
Bei LREST = .TRUE. werden Daten von der RESTART-Datei gelesen

ISTEPF (I;1) ISTEPF ε {1,2,3,4,5}
Ist nur bei LREST = .TRUE. wirksam; Nummer des ersten auszuführenden Programmschrittes

ISTEPL (I;5) ISTEPL ε {1,2,3,4,5}, ISTEPL $\geq$ ISTEPF
Nummer des letzten auszuführenden Programmschrittes

IPRINT (I(100);0)

Dient zur Initialisierung des L(100)-Feldes LPRINT, es gilt:

IPRINT(j) = 0 - LPRINT(j) = .FALSE.

IPRINT(j) ≠ 0 - LPRINT(j) = .TRUE.

IPRINT(1) ≠ 0 - Trace durch alle Subroutinen

IPRINT(2) ≠ 0 - Testausdrucke in SPTREE

IPRINT(3) ≠ 0 - Testausdrucke in INCMAT

IPRINT(4) ≠ 0 - Ausdruck der Platten-Geometrie und Topologie

IPRINT(5) ≠ 0 - Ausdruck von Informationen über den Verbrauch an CPU-Zeit und Speicherplatzverbrauch in den einzelnen Programmschritten

IPRINT(6) ≠ 0 - Ausdruck der Zweigmatrizen von Induktivität und Widerstand

IPRINT(7) ≠ 0 - Ausdruck der Maschenmatrizen von Induktivität und Widerstand

IPRINT(8) ≠ 0 - Ausdruck der Maschen-Inzidenzmatrix

IPRINT(9) ≠ 0 - Testausdrucke in LBMAT

IPRINT(10) ≠ 0 - Ausdruck der externen Induktivitätsmatrizen $\underline{\underline{L}}_{ex}$ und $\underline{\underline{L}}_{m,ex}$

IPRINT(11) ≠ 0 - Testausdrucke in SOLVER

IPRINT(12) ≠ 0 - Testausdrucke in MUTES

IPRINT(13) - Nicht belegt

IPRINT(14) ≠ 0 - Ausdruck des berechneten sekundären Magnetfeldes

IPRINT(15) ≠ 0 - Ausdruck der berechneten Wirbelströme als eindimensionales Feld von Zweigströmen

IPRINT(16) - Nicht belegt

IPRINT(17) ≠ 0 - Bestimmung der Zeitkonstanten

IPRINT(18) - Hat nur bei IPRINT(17) ≠ 0 Bedeutung:
IPRINT(18) ≠ 0 - Ausgabe aller Zeitkonstanten
IPRINT(18) = 0 - Ausgabe nur der maximalen Zeitkonstante

IPRINT(19) ≠ 0 - Inversionstest der Maschen-Induktivitätsmatrix $\underline{\underline{L}}_m$

IPRINT(20) ≠ 0 - Test der Eigenwertanalyse von $\underline{\underline{L}}_m^{-1}\underline{\underline{R}}_m$

IPRINT(21) ≠ 0 - Ausgabe der Integrationsfehler in SOLVER

IPRINT(22) ≠ 0 - CPU-Zeitmessung in MATRIX

IPRINT(23) ≠ 0 - Speicherung der Zweigmatrizen auf einer Plattendatei

IPRINT(24) ≠ 0 - Ausführliche Informationen über verbrauchte Job-Resourcen

IPRINT(25) ≠ 0 - Testausdrucke in MUNG1V (Nur CYBER-Version)

IPRINT(26) - IPRINT(100) - Nicht belegt

COMM (C(10)*60;' ')
Beliebiger Text, der auf der ersten Seite der Job-Liste ausgedruckt wird

LKONS (L;.TRUE.)
Bei LKONS = .TRUE. werden die mit READPL eingelesenen Daten des Preprozessors auf Konsistenz überprüft

NAMELIST /DAT2/ (Subroutine INIT)

```
     NAMELIST /DAT2/ IDICK , NIND ,
    1                NEX , NEXGES , IRAMP , STROM , ICONST , BO ,
```

```
2               ITSW , DT ,  FREL , TBEG , TLIM , OUTINT , IPR ,
3               TMIN , TMAX , PXA , PXE , PFIA , PFIE , PZA , PZE,
4               NX , NFI , NZ ,
```

IDICK (I;1)

IDICK ≠ 0 - Rechnung mit "dicken Platten" (Gleichung (33))

IDICK = 0 - Rechnung mit "dünnen Platten" (Gleichung (34))

NIND (I;NSEC) NIND ≤ NSEC

Anzahl der Symmetriesegmente, die bei der induktiven Koppelung mit dem Grundsegment berücksichtigt werden sollen

NEX (I;0) NEX ≥ 0

Anzahl der Kreisgruppen mit stetiger Stromänderungsfunktion

NEXGES (I;0) NEXGES ≥ 0

Gesamtanzahl der externen Stromkreise (beliebige Stromänderungsfunktion)

IRAMP (I;0)

IRAMP ≠ 0 - Externe Stromkreise mit sprunghafter Stromänderungsfunktion sind vorhanden

IRAMP = 0 - Externe Stromkreise mit sprunghafter Stromänderungsfunktion sind nicht vorhanden

STROM (R;0.)

Hat nur bei IRAMP ≠ 0 Bedeutung; Stromstärke I_0 zur Zeit $t < t_0$ in den externen Kreisen mit sprunghafter Stromänderung

ICONST (I;0)

ICONST > 0 - Anregung durch ein homogenes Magnetfeld in z-Richtung, stetige Zeitfunktion

ICONST = 0 - Keine Anregung durch homogenes Magnetfeld

ICONST < 0 - Anregung durch homogenes Magnetfeld in z-Richtung, sprunghafte Zeitfunktion

BO (R;0.)

Hat nur bei ICONST < 0 Bedeutung; Magnetfeldstärke B_z zur Zeit $t < t_0$

ITSW (I;0)
ITSW = 0 - Zeitintegration mit fester Zeitschrittweite
ITSW ≠ 0 - Zeitschrittweite wird automatisch bestimmt

DT (R;1.E-4) DT > 0.
Zeitschrittweite bei der Integration der Netzwerksdifferentialgleichungen, bei ITSW ≠ 0 wird dieser Wert im Laufe der Rechnung verändert

FREL (R;1.E-5) FREL > 0
Genauigkeitsparameter für Zeitintegrationsroutine

TBEG (R;0.)
Anfangszeit t_0 für Zeitintegration

TLIM (R;1.E-4) TLIM > TBEG
Endzeit t_{max} für Zeitintegration

OUTINT (R;1.E-4) OUTINT > 0.
Die induzierten Wirbelströme werden berechnet und auf eine Plattendatei abgespeichert zu den Zeitpunkten N*OUTINT
(N = 0, 1, 2, ...; TBEG ≤ N*OUTINT ≤ TLIM)

IPR (I;1) IPR ≥ 1
Hat nur bei IPRINT(15) ≠ 0 Bedeutung; die berechneten Wirbelströme in den einzelnen Zweigen werden zu den Zeitpunkten N*IPR*OUTINT ausgedruckt (N = 0, 1, 2, ...)

TMIN (R;0.)
Anfangszeit für die Berechnung des Magnetfeldes der induzierten Wirbelströme

TMAX (R;0.) TMAX ≥ TMIN
Endzeit für die Berechnung des Magnetfeldes der induzierten Wirbelströme

PXA (R;1.E-4) PXA > 0.

Radiale Grenze r_a für Magnetfeldberechnung (s. Abb. 5.9)

PXE (R;1.E-4) PXE ≥ PXA
Radiale Grenze r_e für Magnetfeldberechnung (s. Abb. 5.9)

PFIA (R;0.) PFIA ≥ 0
Azimutale Grenze ϕ_a (in Grad) für Magnetfeldberechnung (s. Abb. 5.9)

PFIE (R;0.) PFIE ≥ PFIA, PFIE ≤ 360°
Azimutale Grenze ϕ_e (in Grad) für Magnetfeldberechnung (s. Abb. 5.9)

PZA (R;0.)
Axiale Grenze z_a für Magnetfeldberechnung (s. Abb. 5.9)

PZE (R;0.) PZE ≥ PZA
Axiale Grenze z_e für Magnetfeldberechnung (s. Abb. 5.9)

NX (I;1) NX ≥ 1
Anzahl der Auswertepositionen in radialer Richtung für Magnetfeldberechnung

NFI (I;1) NFI ≥ 1
Anzahl der Auswertepositionen in azimutaler Richtung für Magnetfeldberechnung

NZ (I;1) NZ ≥ 1
Anzahl der Auswertepositionen in axialer Richtung für Magnetfeldberechnung

Spezifikation der anregenden externen Stromkreise

Die Spezifikation der externen Stromkreise geschieht mit Hilfe der NAMELIST-Blöcke /EXT1/ und /EXT2/, die im Anschluß an die NAMELIST /DAT2/ eingelesen werden.

```
NAMELIST /EXT1/ CENT,NORM,RAD,WIND
```

CENT (R(3);0.,0.,0.)

Mittelpunkt c eines externen Stromkreises (s. Abb. 5.7)

NORM (R(3);0.,0.,1.)
Normale n eines externen Stromkreises (s. Abb. 5.7)

RAD (R;1.)
Radius r eines externen Stromkreises (s. Abb. 5.7)

WIND (R;1.)
Windungszahl w eines externen Stromkreises (s. Abb. 5.7)

```
NAMELIST /EXT2/ NCOIL
```

NCOIL (I;-)
Anzahl der externen Stromkreise innerhalb einer Kreisgruppe

Die Geometrie der externen erregenden Stromkreisen wird nach Kreisgruppen geordnet spezifiziert. Für jede einzelne Kreisgruppe wird zunächst mit /EXT2/ die Anzahl der Kreise in dieser Kreisgruppe eingelesen. Danach wird für jeden Stromkreis in dieser Kreisgruppe die NAMELIST /EXT1/ eingelesen und so die Geometrie der Kreise festgelegt. Gibt es eine Kreisgruppe mit sprunghafter Stromänderungsfunktion (IRAMP ≠ 0), so muß diese Kreisgruppe stets als letzte eingelesen werden.

5.7.2 Zeitfunktion der Stromänderung

Die Zeitfunktion der Stromänderung $\underline{\dot{I}}_{ex}$ muß in der Subroutine SAD spezifiziert werden. Sie hat folgende Gestalt:

```
      SUBROUTINE SAD (ADX,N,T,IND)
      REAL ADX(N) , T
C
      PARAMETER (OMEG = 50. , STR = 10.0 , PI = 3.1415926535)
C
      IND = 1
      ADX(1) = -2.*PI*OMEG*STR*SIN(2.*PI*OMEG*T)
C
```

```
      RETURN
      END
```

In diesem Beispiel gibt es nur eine Kreisgruppe mit der stetigen Stromänderungsfunktion $I^{\bullet}_{ex}(t) = I^*(-2\pi\omega)\sin(2\pi\omega t)$. Die Parameter der Subroutine SAD haben folgende Bedeutung:

ADX - In diesem Datenfeld werden die berechneten Werte $I^{\bullet}_{j,ex}$ für die Kreisgruppen mit <u>stetiger</u> Stromänderungsfunktion ausgegeben. Die Kreisgruppen mit sprunghafter Zeitänderungsfunktion brauchen in SAD nicht berücksichtigt zu werden.

N - Länge des Feldes ADX. Gibt es kein homogenes Magnetfeld mit stetiger Zeitänderung (Fall ICONST ≥ 0), so ist N = NEX (s. NAMELIST /DAT2/). Im anderen Falle ist N = NEX+1, wobei der Wert $B^{\bullet}_{z}(T)$ stets im Element N des Feldes ADX stehen muß.

T - Zeitpunkt, an dem die Stromänderungsfunktionen ausgewertet werden sollen

IND - Gibt es überhaupt keine Kreisgruppe mit stetiger Stromänderungsfunktion, muß IND in SAD auf 0 gesetzt werden, anderenfalls auf einen beliebigen Wert ≠ 0.

6. POSTPROZESSOREN

6.1 Ausdrucken der Wirbelstromverteilung: Programm DRUCK

6.1.1 Programmstruktur

Mit Hilfe des Postprozessors DRUCK ist es möglich, die berechneten Wirbelströme in Form übersichtlicher Listen auszudrucken. Dabei unterstützt DRUCK sowohl die toroidale Geometrie, wie sie mit dem Preprozessor TORUS beschrieben wird (s. Kap. 4.1), als auch die mit dem Preprozessor FLATPL (Kap. 4.2) behandelte ebene Scheibe.

Die berechneten Zweigströme können auf zwei verschiedene Art und Weisen dargestellt werden. Zum einen kann zu bestimmten Zeitpunkten die Wirbelstromverteilung in der gesamten Struktur ausgedruckt werden. Außerdem kann noch die gesamte zeitliche Entwicklung des induzierten Wirbelstromes in bestimmten Elementen ausgedruckt werden. Diese Elemente werden durch die Angabe ihrer Nummer spezifiziert. Die in beiden Preprozessoren benutzte Numerierungstechnik wird im Anhang 1 beschrieben.

Bei der übersichtlichen Darstellung der räumlichen Wirbelstromverteilung in der Struktur wird ausgenutzt, daß sowohl TORUS als auch FLATPL im wesentlichen eine zweidimensionale Rechteckgeometrie beschreiben. Mit DRUCK kann deshalb die Wirbelstromverteilung innerhalb der einzelnen Lagen, jeweils getrennt nach Strömen in poloidaler und toroidaler Richtung (bei TORUS) bzw. x- und y-Richtung (bei FLATPL), in Form zweidimensionaler Felder ausgegeben werden.

Die Dimensionen dieses Rechteckschemas, also die Elementanzahlen in poloidaler/toroidaler bzw. x/y Richtung werden direkt vom Preprozessor (Kanal 12) eingelesen. Genauso werden vom Preprozessor Informationen über Fehlstellen in diesem Rechteckschema übergeben. Solche Fehlstellen treten beim Preprozessor FLATPL im Bereich des inneren Loches auf (s. Kap. 4.2.1), bei TORUS an toroidalen e-Rändern, bei denen einzelne poloidale Abschnitte als isolierend betrachtet werden (s. Kap. 4.1.2). Die genaue Spezifikation der Schnittstelle Preprozessor - Postprozessor DRUCK findet sich im Anhang 2. Im Anhang 3 wird die Schnittstelle Rechenprogramm - Postprozessor (Kanal 10) näher beschrieben, über die von DRUCK die berechneten transienten Wirbelstromverteilungen eingelesen werden.

Bei der Darstellung der zeitlichen Stromverläufe in einzelnen Elementen können

die Werte vor Ausdruck mit einem beliebigen Faktor skaliert werden. Außerdem ist es möglich, automatisch die Extremwerte der zeitlichen Stromverläufe zu bestimmen.

Alle im Programm vorkommenden Felder sind fest dimensioniert, die Feldgrenzen sind dabei über ein PARAMETER-Statement definiert. Dies hat eine Anzahl von Restriktionen zur Folge, die aber durch Änderung des PARAMETER Wertes schnell verändert werden können:

Anzahl der Elemente	≤ 2500	PARAMETER NPLMAX
Anzahl der Elemente in einer Koordinatenrichtung	≤ 100	PARAMETER NXYMAX
Anzahl der Zeitpunkte bei zeitlichen Stromverläufen	≤ 500	PARAMETER NTMAX
Anzahl der Elemente, deren zeitlicher Stromverlauf ausgedruckt wird	≤ 30	PARAMETER NPMAX

6.1.2 Eingabebeschreibung DRUCK

Neben den Daten des Preprozessors und des Rechenprogrammes kann DRUCK auch noch direkt vom Anwender spezifizierte Daten erhalten. Diese werden über die NAMELIST /AB/ übergeben, die im weiteren spezifiziert wird. Jede der NAMELIST Variablen ist vorinitialisiert. Ist die Eingabedatei (Kanal 6) leer, so wird ausschließlich mit diesen Initialisierungswerten gearbeitet.

NAMELIST /AB/

```
      NAMELIST /AB/ NPRINT , PL , LPEROD , LTORUS ,
     1              NPKT , SCALE , LTEST , IZAHLA
```

NPRINT (I;1) NPRINT ≥ 1

Die abgespeicherten Wirbelstromverteilungen werden jedes NPRINT-te Mal ausgedruckt

NPKT (I;0) NPMAX ≥ NPKT ≥ 0

Anzahl der Elemente, deren zeitliche Stromverläufe ausgedruckt werden sollen

PL (I(NPKT);0)
Nummern der Elemente, deren zeitliche Stromverläufe ausgedruckt werden sollen

LPEROD (L;.FALSE.)
Hat nur bei NPKT > 0 Bedeutung; Bei LPEROD = .TRUE. werden die Extremwerte der zeitlichen Stromverläufe in den durch PL spezifizierten Elementen bestimmt

LTORUS (L;.FALSE.)
LTORUS = .TRUE. - Ergebnisse einer Rechnung mit Preprozessor TORUS werden ausgedruckt
LTORUS = .FALSE. - Ergebnisse einer Rechnung mit Preprozessor FLATPL werden ausgedruckt

LTEST (L;.FALSE.)
Bei LTEST = .TRUE. werden ausführliche Testausdrucke ausgegeben

IZAHLA (I;1)
Der Ausdruck der Wirbelströme beginnt bei dem abgespeicherten Zeitschritt IZAHLA

SCALE (R;1.)
Hat nur bei NPKT > 0 Bedeutung; Skalierungsfaktor für die zeitlichen Stromverläufe in den durch PL spezifizierten Elementen

6.2 Plotprogramm ERAPLO

Das wichtigste Hilfsmittel zur Gewinnung qualitativer Informationen über den Wirbelstromverlauf ist das von Herrn F. Katz erstellte Plotprogramm ERAPLO. Dieses Programm unterstützt eine Geometrie, bei der die Elemente auf einzelne zweidimensionale Lagen verteilt sind, und innerhalb der Lagen (bei Wahl geeigneter Koordinaten) in einem Rechteckschema angeordnet sind. Die beiden im Kap. 4 beschriebenen Preprozessoren erzeugen derartige Elementeinteilungen.

Das Plotprogramm baut auf dem graphischen System GIPSY [17] auf und ist deshalb nur auf der IBM Rechenanlage verfügbar. Für die Übertragung der

Postprozessor-Dateien von der CYBER auf die IBM wurde deshalb ein eigenes Programm CUFOFO entwickelt, das im Kap. 7.2 beschrieben wird.

Die ausführliche Benutzerdokumentation des ERAPLO Programmes findet sich in einem separaten Bericht [16]. An dieser Stelle soll deshalb nur ein kurzer Überblick über die Fähigkeiten dieses Systems gegeben werden. Dabei kann man zwischen eindimensionalen, zweidimensionalen und dreidimensionalen Plotdarstellungen unterscheiden.

Als eindimensionale Darstellung bietet ERAPLO die Möglichkeit, den zeitlichen Verlauf des Zweigstromes in beliebigen Elementen in Form eines Strom - Zeit Diagrammes zu zeichnen. Die einzelnen Elemente werden dabei über ihre vom Preprozessor generierte Elementnummer referiert (s. Anhang 1).

Um einen wirklichen Einblick in die Stromverteilung innerhalb der gesamten Struktur zu erhalten, sind allerdings zweidimensionale und dreidimensionale Darstellungen unerlässlich. Dabei wird in diesen Fällen stets nur die Stromverteilung innerhalb einer einzelnen Elementlage gezeichnet. Die grundlegende Vorgehensweise ist, daß an den Knotenpunkten des elektrischen Netzwerkes durch eine geeignete vektorielle Addition und Mittelung Strom- bzw. Stromdichte-Vektoren berechnet werden. Dabei werden die Ströme in allen den betreffenden Knoten berührenden Netzwerkszweigen berücksichtigt. Diese Vektoren können dann auf verschiedene Art und Weise graphisch dargestellt werden.

Bei der zweidimensionalen Darstellung wird der geometrische Ort der Knotenpunkte auf der stromführenden Struktur nicht berücksichtigt. Alle Knotenpunkte werden vielmehr auf ein ebenes, äquidistantes, rechteckiges Gitter abgebildet. Die topologischen Beziehungen der einzelnen Knotenpunkte bleiben bei dieser Abbildung allerdings erhalten. In das so ermittelte Knotengitter werden dann die berechneten Stromstärkevektoren eingetragen.

Mit den dreidimensionalen Plotdarstellungen kann schließlich auch die Geometrie der Elementeinteilung sowie die räumliche Lage der Stromvektoren graphisch dargestellt werden. Zur Darstellung und Überprüfung der Elementeinteilung können alle Platten eines bestimmten Typs, also z. B. alle poloidalen oder alle toroidalen Platten, in ihrer dreidimensionalen Lage gezeichnet werden. Die Struktur kann dabei aus einem beliebigen Blickwinkel betrachtet werden, verdeckte Platten werden durch Anwendung eines Hidden-Line Algorithmus aus dem Bild entfernt. Statt der Element-Mittelflächen können auch

alle Elementachsen gezeichnet werden, in diesem Fall werden beide Elementtypen zusammen dargestellt.

Zur voll dreidimensionalen Darstellung der induzierten Wirbelströme werden allen Knotenpunkten des elektrischen Netzwerkes geometrische Orte auf der Konturfläche zugeordnet, diese Werte werden im Preprozessor berechnet und direkt dem Plotprogramm übergeben. An diesen Punkten werden dann die durch geeignete Addition und Mittelung berechneten Stromdichtevektoren eingezeichnet. Verdeckte Knoten bzw. Stromdichtevektoren werden wieder über einen Hidden-Line Algorithmus eliminiert.

Beispiele für alle hier angesprochenen Plotdarstellungen kann man im Kap. 8 finden.

7. WEITERE HILFSPROGRAMME INNERHALB DES ERATO SYSTEMS

7.1 Programm SODEMA

7.1.1 Programmbeschreibung

Das von Herrn F. Katz erstellte Programm SODEMA dient zur rationellen Verwaltung großer Sourcedateien. Dies Programm kann dabei zwei Aufgaben erfüllen:

- Unterschiedliche Versionen ein und desselben Programmsystems (z. B. für verschiedene Rechner) können in einer einzigen Sourcedatei (eventuell verteilt auf verschiedene Member) gehalten und die jeweils benötigte Version daraus erzeugt werden.

- Einzelne Source-Makros, also an vielen Stellen des Programmes immer wiederkehrende Programmteile wie COMMON-Deklarationen, können aus einer Makro-Datei in den Source-Code geladen werden.

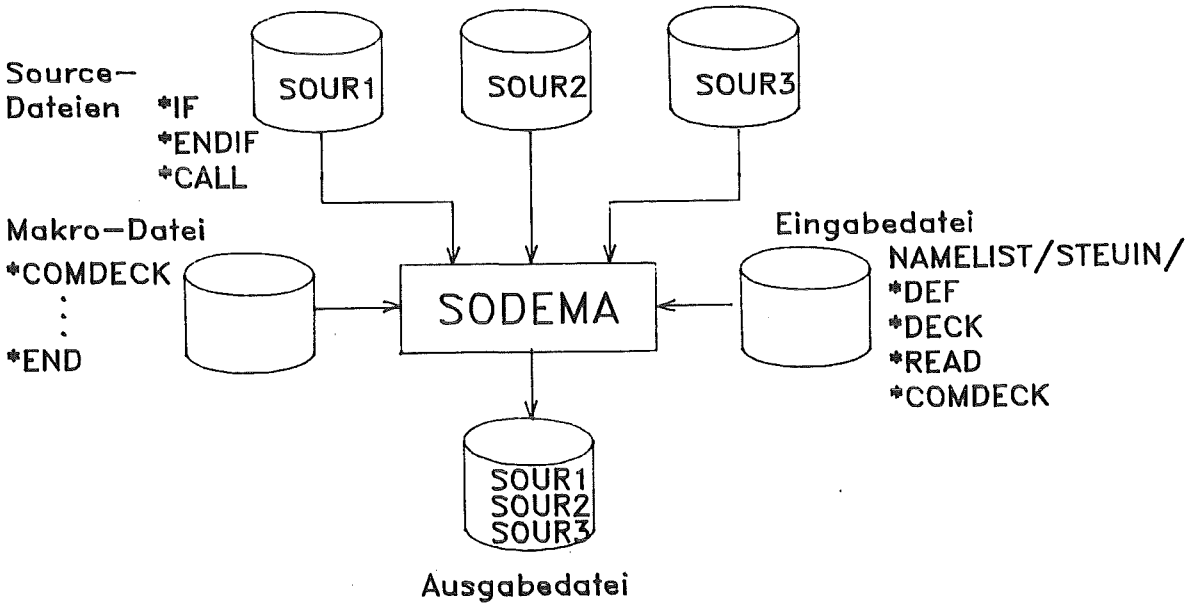


Abb. 7.1: Struktur des SODEMA-Programmes

Den Aufbau eines typischen SODEMA-Laufes zeigt Abb. 7.1. Die Steuerung des Laufes erfolgt über die NAMELIST /STEUIN/ sowie über einzelne Steuerkarten. Dadurch kann nacheinander der Inhalt verschiedener Sourcedateien eingelesen und zusammen auf eine Ausgabedatei geschrieben werden (Steuerkarten *DECK, *READ).

Für jeden SODEMA-Lauf wird eine Version definiert (Steuerkarte *DEF). Innerhalb der Source-Dateien können einzelne Kartenbereiche so gekennzeichnet werden (Steuerkarten *IF ... *ENDIF), daß sie nur bei Vorliegen bzw. Nicht-Vorliegen einer bestimmten Version in die Ausgabedatei übertragen werden. Außerdem können an beliebiger Stelle in den Sourcedateien Macros hereingeladen werden (Steuerkarte *CALL). Diese Makros können entweder in einer eigenen Makrodatei definiert sein, oder direkt in der SODEMA-Eingabedatei (Steuerkarten *COMDECK, ..., *END).

7.1.2 Beschreibung der Eingabe-NAMELIST und der Steuerkarten

NAMELIST /STEUIN/

```
NAMELIST /STEUIN/   SYSTEM , FEIN , FOUT , FCOM , NOUT
```

SYSTEM (C*8;'CYBER')
Zu erzeugende Programmversion

FEIN (I;10) $10 \leq FEIN \leq 69$
Kanalnummer der Eingabedatei

FOUT (I;9) $1 \leq FOUT \leq 9, FOUT \neq FCOM$
Kanalnummer der Ausgabedatei

FCOM (I;8) $1 \leq FCOM \leq 9, FCOM \neq FOUT$
Kanalnummer der Makro-Datei

NOUT (C*8;'00000000')
Identifikationsstring für Spalte 73 - 80 der Ausgabedatei

Steuerkarten

Alle Steuerkarten beginnen mit einem * , der stets in der 1. Spalte stehen muß. Von vielen Steuerkarten gibt es eine Kurzform, die zugehörigen Buchstaben sind in den Definitionen unterstrichen.

*DEFINE,VER

Es handelt sich um einen SODEMA-Lauf, bei dem die Source für die Programmversion VER erzeugt werden soll; VER ist vom Typ C*8.

*READ,DSN,UNIT

Es wird eine Sourcedatei über den durch UNIT (Typ I) definierten Kanal eingelesen und auf die Ausgabedatei geschrieben. DSN (Typ C*8) gibt den Namen dieser Datei an, diese Information wird in SODEMA z. Z. aber nicht verwendet. Es kann über beliebig viele Kanäle mit Nummern zwischen 10 und 69 eingelesen werden.

*DECK,DKN(,UNIT)

Der Character DKN (Typ C*4) wird zur Identifikation von einzelnen Sourcedateien oder Sourceabschnitten in der Ausgabedatei benutzt. Er erscheint in den Spalten 73 - 76 der Ausgabedatei. Bei Aufruf jedes neuen *DECK-Befehles wird die Zeilennummer (Spalte 77 - 80) auf 1 zurückgesetzt. Bei zusätzlicher Angabe von UNIT (Typ I) wird eine neue Ausgabedatei eröffnet. Wird UNIT nicht spezifiziert, ist die Ausgabedatei durch die NAMELIST /STEUIN/ festgelegt. Wenn *DECK überhaupt nicht benutzt wird, wird auch der Identifikationsstring von /STEUIN/ (Variable SYSTEM) übernommen.

*IF DEF VER
...
*ENDIF

Alle zwischen diesen beiden Steuerkarten befindlichen Karten werden nur dann in die Ausgabedatei übertragen, wenn die Version VER (Typ C*8) definiert ist.

*IF -DEF VER
...
*ENDIF

Alle zwischen diesen beiden Steuerkarten befindlichen Karten werden nur dann in die Ausgabedatei übertragen, wenn die Version VER (Typ C*8) nicht definiert ist.

*COMDECK,COM
...
*END

Alle zwischen diesen beiden Steuerkarten befindlichen Karten werden als

Makro COM (Typ C*8) definiert. Diese Definition kann sowohl in einer separaten Makro-Datei als auch in der SODEMA-Eingabedatei erfolgen.

*CALL,COM(,UNIT)

Der Makro COM wird in die Ausgabedatei geladen. Bei Angabe von UNIT (Typ I) wird vorher eine neue Makrodatei eröffnet.

7.2 Das Programm CUFOFO

Wie schon erwähnt, ist das Plotprogramm ERAPLO nur auf der IBM-Rechenanlage des KfK installiert. Deshalb ist es notwendig, die auf der CYBER erzeugten Ausgabedaten von Preprozessor und Rechenprogramm auf die IBM zu übertragen. Diese Übertragung wird durch die Tatsache kompliziert, daß über die Rechnerkopplung CYBER-Vorrechner - KfK Rechenanlage nur Daten mit der festen Recordlänge 80 (Kartendateien) übertragen werden können.

Das Programm CUFOFO hat deshalb zwei Aufgaben: Es liest die erzeugten drei Ausgabedateien (eine von Preprozessor, zwei vom Rechenprogramm, siehe Abb. 3.2) sequentiell ein und schreibt sie formatiert wieder auf 3 andere, temporäre Dateien heraus. Gleichzeitig werden am Anfang und am Ende jeder dieser Dateien (falls sie nicht leer sind) JCL-Karten eingefügt und so Jobs aufgebaut, mit denen die Datenübertragung CYBER - KfK durchgeführt werden kann. Dabei können per Eingabe (NAMELIST /IN/) die Kennung der Übertragungsjobs sowie die Zieldateien auf der IBM spezifiziert werden. Die Zieldateien können sich sowohl auf der Platte als auch auf einem DV-Band befinden, die Plattendateien müssen allerdings katalogisiert sein.

Die Anwendung des Programmes CUFOFO ist auch dann notwendig, wenn die IBM-Version des ERATO-Codes benutzt wird, da vom Plotprogramm ERAPLO alle Daten formatiert eingelesen werden. In diesem Fall führt CUFOFO allerdings nur die Umwandlung der Ausgabedaten durch. Es gibt deshalb zwei unterschiedliche Versionen von CUFOFO, die mit SODEMA verwaltet werden. Die NAMELIST /IN/ kommt bei der IBM-Version nicht vor.

Eine vollständige Liste des CUFOFO Programmes (IBM-Version) sowie ein Beispiel für den Übertragungsjob befinden sich im Anhang 4.

NAMELIST /IN/

```
NAMELIST /IN/ JOBID , DATEI , TAPE , UNIT , LABEL
```

Jede Variable in dieser NAMELIST ist ein Feld der Länge 3. Das erste Feldelement bezieht sich dabei auf den Job zur Übertragung der ersten Ausgabedatei des Rechenprogrammes ERATO (Zweigmatrizen, s. Kap. 5.3.1), das zweite Feldelement auf die zweite ERATO-Ausgabedatei (Geometrie, berechnete Wirbelströme), und des dritte Feldelement auf die Preprozessor-Datei.

```
JOBID       (C(3)*2;'A ','B ','C ')
```
Kennung des Übertragungsjobs

```
DATEI       (C(3)*30;'IRE571.ERATO.DATA(OUTOUT1)','IRE571.PLATES.OUTPUT(DATEN)',
                     'IRE571.BOUNDS.OUTPUT(DATEN)')
```
IBM-Dateiname, in die übertragen werden soll

```
UNIT        (C(3)*4;'DISK','DISK','DISK')
```
UNIT = 'DISK' - Übertragung in eine Plattendatei
UNIT = 'TAPE' - Übertragung auf ein Band

```
TAPE        (C(3)*6;'DVXXXX','DVXXXX','DVXXXX')
```
Hat nur bei UNIT = 'TAPE' Bedeutung; Nummer des DV-Bandes, auf das übertragen wird

```
LABEL       (C(3)*2;'01','01','01')
```
Hat nur bei UNIT = 'TAPE' Bedeutung, Label-Nummer der zu übertragenden Datei

8. ANWENDUNGSBEISPIELE

8.1 Wirbelströme in einem Segment eines Poloidalfeldspulengehäuses

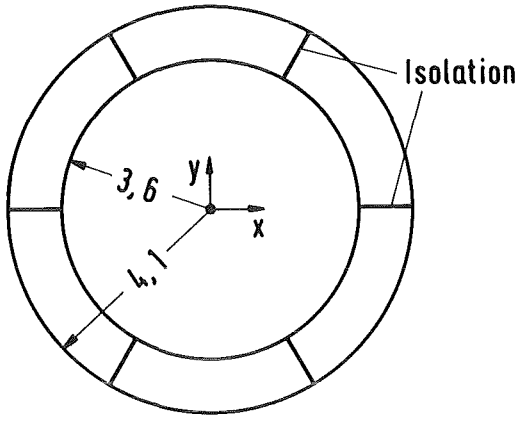
Als erstes Anwendungsbeispiel für den ERATO-Code soll eine Rechnung mit dem Preprozessor TORUS diskutiert werden. Es geht darum, den Wirbelstromverlauf in einem Segment eines Poloidalfeldspulengehäuses zu bestimmen, wie es in der POLO-Studie [24] untersucht wird. Speziell war dabei die Gehäuse-Zeitkonstante gefragt, also die charakteristische Zeit, innerhalb der in der Struktur induzierte Wirbelströme von selbst abklingen. Hier soll beispielhaft nur eine einzige Rechnung gezeigt werden, die ausführlichen Ergebnisse finden sich in [25].

Das Spulengehäuse ist ein Torus mit 6 Segmenten, wobei der poloidale Querschnitt die Form eines rechteckigen Kastens hat (s. Abb. 8.1). Für diese Rechnung wird angenommen, daß die einzelnen Segmente ideal voneinander isoliert sind. Unter Ausnutzung der Symmetrie braucht mit ERATO nur die obere Hälfte eines dieser Segmente modelliert zu werden.

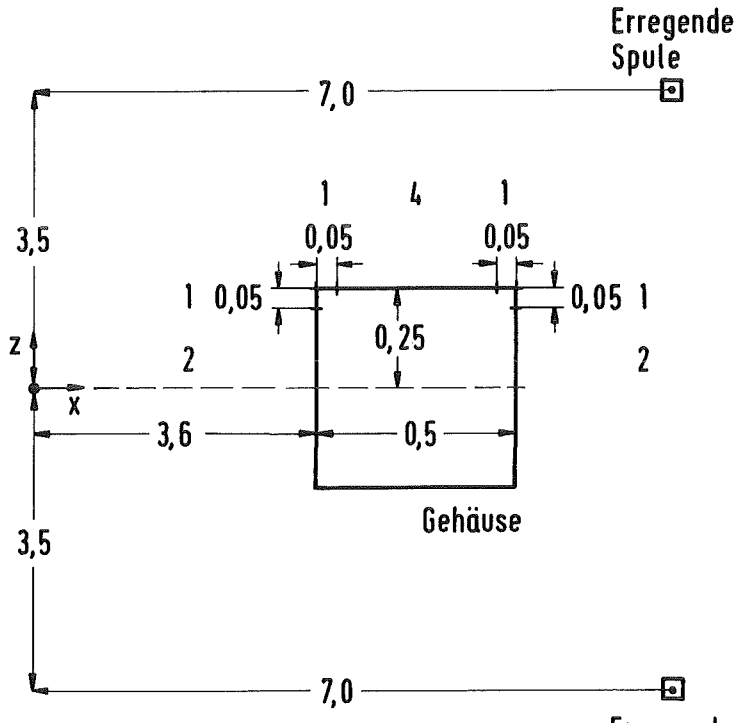
Abb. 8.1b zeigt die Unterteilung der poloidalen Kontur in einzelne Abschnitte und die jeweils gewählten Elementanzahlen. Toroidal wird nur ein einziger Abschnitt berücksichtigt, der in 12 Elemente eingeteilt ist. Die vollständige Eingabe für den Preprozessor TORUS zeigt Tabelle 8.1.

Die von TORUS erzeugte Elementeinteilung ist in Abb. 8.2 dargestellt. Auf den Bildern sind die mit ERAPLO gezeichneten Mittelflächen der toroidalen (Abb. 8.2a) und der poloidalen (Abb. 8.2b) Platten zu erkennen. Die Darstellung wurde mit einem Hidden-Line Algorithmus erzeugt, deshalb sind nur die innere Ringfläche und die Deckelfläche des Gehäuses sichtbar. Abb. 8.3 zeigt in derselben Projektion das erzeugte Netzwerk, wobei an jeden Knoten die zugehörige Nummer eingetragen ist.

In Tabelle 8.2 findet man die Eingabedaten für das Rechenprogramm ERATO. Hier wird spezifiziert, daß die Wirbelströme durch zwei externe Spulen angeregt werden, in denen zur Zeit t=0 der Strom schlagartig von $I_0=10^6$ A auf 0 sinkt. Die induzierten Ströme werden bis zu einer Zeit T_{max} = 4 ms berechnet, dabei werden die Ergebnisse alle 0.1 ms abgespeichert. Das von den Strömen erzeugte sekundäre Magnetfeld wird an den Punkten r = 2m, 4m; ϕ = 30°; z = -2m, 0m, 2m berechnet und in Listenform ausgegeben.



(a)



(b)

Abb. 8.1: Das betrachtete Poloidalfeldspulengehäuse mit Ausmaßen und Angaben zur Diskretisierung; alle Längenangaben sind in m. (a) Ansicht von oben; (b) Gehäusequerschnitt

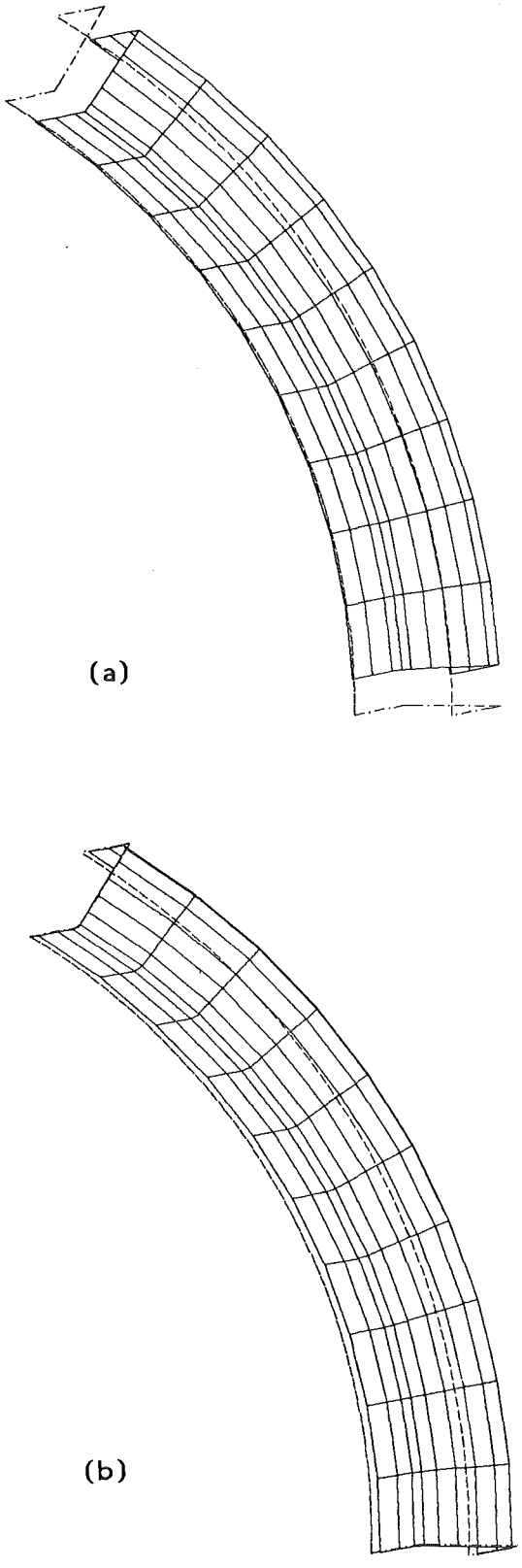


Abb. 8.2: Die von TORUS erzeugte Elementeinteilung, gezeichnet mit dem Programm ERAPLO. Zentralprojektion vom Punkt (-50,-10,50) auf die (x,y)-Ebene. (a) Toroidale Platten; (b) Poloidale Platten

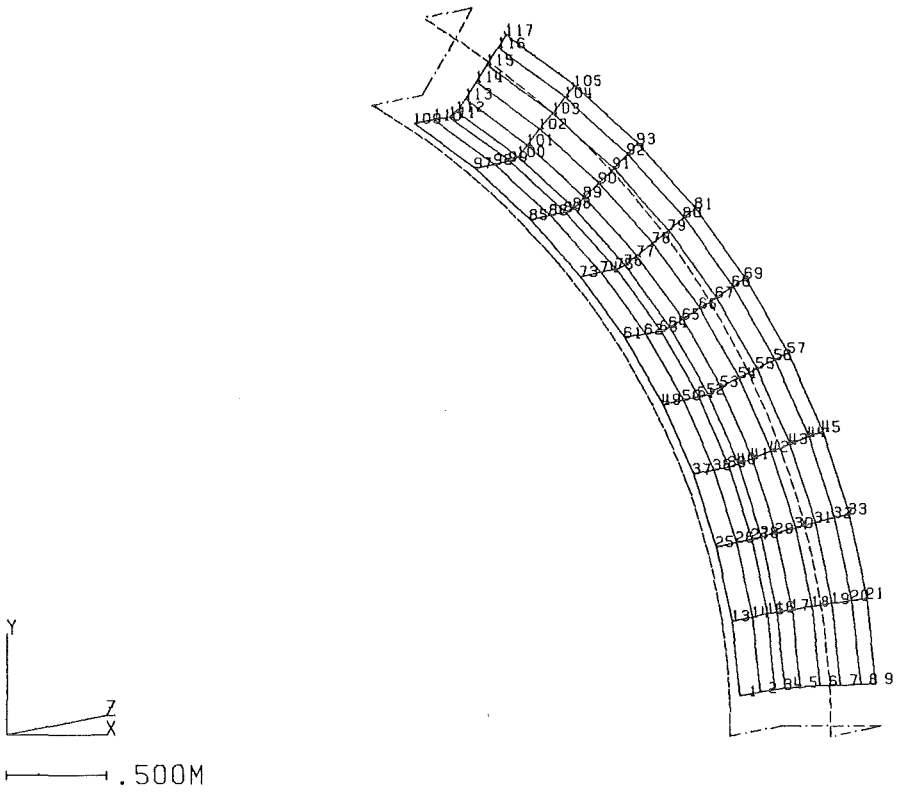


Abb. 8.3: Das zugehörige Netzwerk mit Knotennummern; dieselbe Projektion wie in Abb. 8.2.

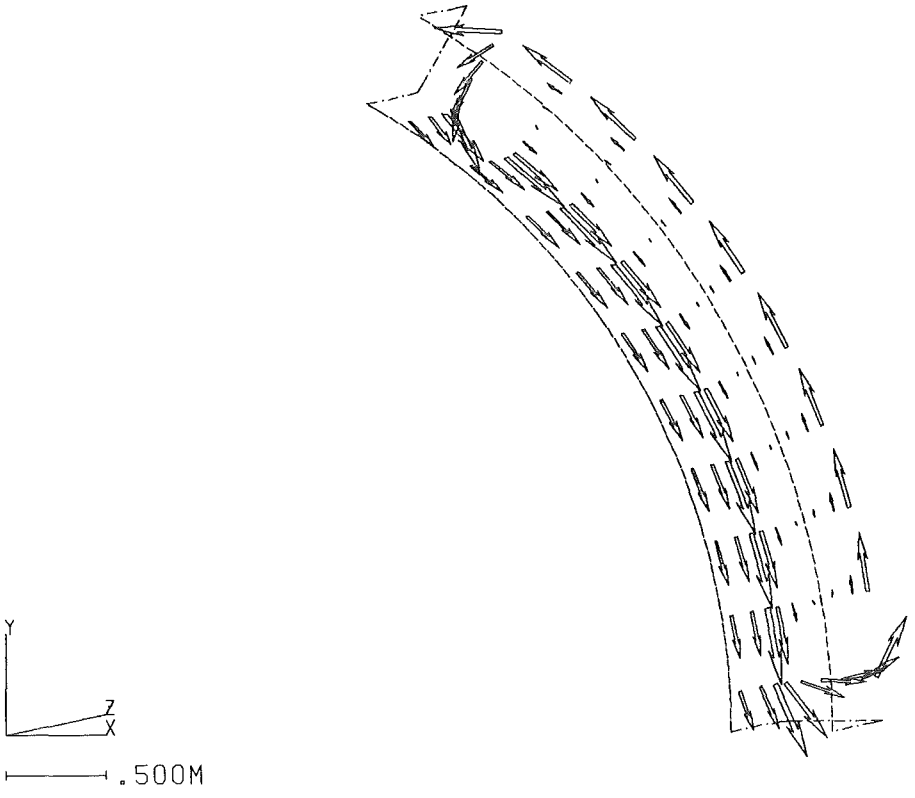


Abb. 8.4: Die induzierte Stromdichte zur Zeit t=0 in einer dreidimensionalen Darstellung; dieselbe Projektion wie in Abb 8.2.

```
---------------------------------------------------------------------
&NUM   NSEC = 6, ISYM = 1, NPP = 7, NPT = 1,  IBNDN = 1, IBNDP = 1,
       DI = 1.E-2, RHOI = 7.143E-7, RUNID = 'ITPS1C1 ',
       COMM(1) = ' GEHAEUSE EINER POLOIDALFELDSPULE     ',
       COMM(2) = ' ISOLIERTES SEGMENT                   ',
       COMM(4) = ' BEISPIEL FUER PREPROZESSOR TORUS',
       LPRINT(1) = .FALSE., LPRINT(2) = .FALSE., LPRINT(3) = .FALSE.,
       LPRINT(4) = .FALSE., LPRINT(5) = .FALSE., LPRINT(6) = .FALSE.,
       LPRINT(7) = .FALSE., LPRINT(8) = .FALSE.,
&END
&POL  PPOL(1,1) = 3.6, PPOL(2,1) = 0.00,
      PPOL(1,2) = 3.6, PPOL(2,2) = 0.2,
      PPOL(1,3) = 3.6, PPOL(2,3) = 0.25,
      PPOL(1,4) = 3.65, PPOL(2,4) = 0.25,
      PPOL(1,5) = 4.05, PPOL(2,5) = 0.25,
      PPOL(1,6) = 4.1, PPOL(2,6) = 0.25,
      PPOL(1,7) = 4.1, PPOL(2,7) = 0.2,
      PPOL(1,8) = 4.1, PPOL(2,8) = 0.00,
      IPOL(1) = 1, IPOL(2) = 1, IPOL(3) = 1,
      IPOL(4) = 1, IPOL(5) = 1, IPOL(6) = 1, IPOL(7) = 1,
      NPPL(1) = 2, NPPL(2) = 1, NPPL(3) = 1,
      NPPL(4) = 4, NPPL(5) = 1, NPPL(6) = 1, NPPL(7) = 2,
&END
&TOR  NPTL(1) = 10,
&END
---------------------------------------------------------------------
```

Tabelle 8.1: Eingabe für Preprozessor TORUS

```
---------------------------------------------------------------------
&DAT1  ISTEPL = 5, RUNIDL = 'TORBSP  ',
       COMM(1) = ' BEISPIELJOB ERATO MIT PREPROZESSOR TORUS   ',
       IPRINT(1) = 0, IPRINT(2) = 0, IPRINT(3) = 0,
       IPRINT(4) = 0, IPRINT(5) = 1, IPRINT(6) = 0, IPRINT(7) = 0,
       IPRINT(8) = 0, IPRINT(9) = 0, IPRINT(10) = 0,IPRINT(11) = 0,
       IPRINT(12) = 0, IPRINT(14) = 1, IPRINT(17) = 1, IPRINT(18) = 1,
       IPRINT(19) = 0, IPRINT(20) = 0, IPRINT(21) = 0, IPRINT(22) = 0,
       IPRINT(23) = 0, IPRINT(24)=0, IPRINT(25)=0,
&END
&DAT2  NEX = 0, IDICK = 0, DT = 1.E-3, TLIM = 4.E-3,
```

```
      OUTINT = 1.E-4, ITSW = 1, FREL = 5.E-4, TMAX = 4.E-3,
      NEXGES = 2, NX = 3, NFI = 1, NZ = 2, IRAMP = 1,
      PXA = 2., PFIA = 30., PZA = -2., STROM = 1.E6,
      PXE = 4., PFIE = 30., PZE =  2., ICONST = 0,
&END
&EXT2  NCOIL = 2  &END
&EXT1  RAD = 7.0, CENT(3) = 3.5 &END
&EXT1  RAD = 7.0, CENT(3) = -3.5 &END
```

Tabelle 8.2: Eingabedaten für das Rechenprogramm ERATO

Typische Ergebnisse einer ERATO-Simulation sind in den Abbildungen 8.4 und 8.5 dargestellt. Beide Bilder zeigen den Wirbelstromverlauf zur Zeit t=0. Wegen der vorausgesetzten unstetigen Stromänderung in den erregenden Stromkreisen fließt zu diesem Zeitpunkt bereits ein induzierter Strom. In Abb. 8.4 wurde die 3D Darstellung der Stromdichte gewählt, wegen des benutzten Blickwinkels sind wieder nur innere Ringfläche und Deckelfläche sichtbar. Abb. 8.5 benutzt die 2D Abwicklung zur Darstellung der Stromstärke in den Knotenpunkten. Man erkennt auf beiden Bildern die typischen Sattelströme, wie sie wegen der angenommenen idealen Isolierung der Segmente zu erwarten waren.

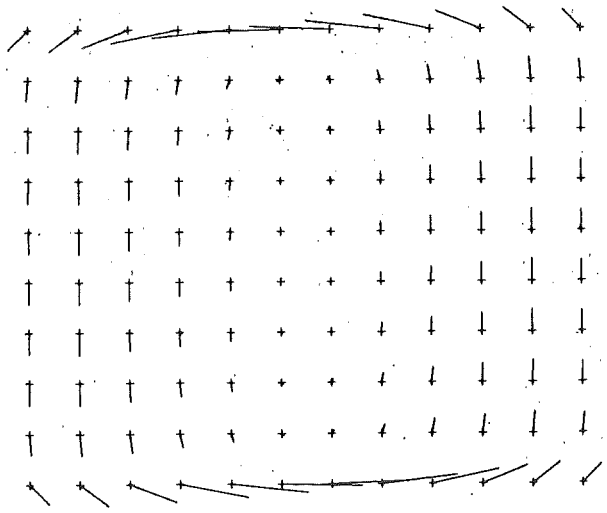


Abb. 8.5: Der induzierte Strom zur Zeit t=0 in einer zweidimensionalen Abwicklung der Schale. Der untere Bildrand entspricht dem Strukturrand in negativer toroidaler Richtung.

8.2 Wirbelströme in einer rechteckigen Scheibe

Als weiteres Beispiel für die Anwendung des Programmsystems ERATO soll noch eine Rechnung mit dem Preprozessor FLATPL präsentiert werden. Die behandelte Scheibengeometrie zeigt Abb. 8.6. Das zentrale Loch hat in der Rechnung eine Ausdehnung von h = 35.2 cm, die Scheibe ist 2 cm dick. Die Anregung der Wirbelströme erfolgt durch einen mit 50 Hz periodisch veränderlichen Strom, die Zeitänderungsfunktion wurde mit der in Kap. 5.7.2 angegebenen Subroutine SAD bereitgestellt. Die erregende Spule befindet sich in 29 cm Abstand senkrecht über dem Scheibenzentrum. Deshalb braucht nur das in Abb. 8.6 gezeigte obere rechte Viertel der Scheibe mit FLATPL diskretisiert zu werden.

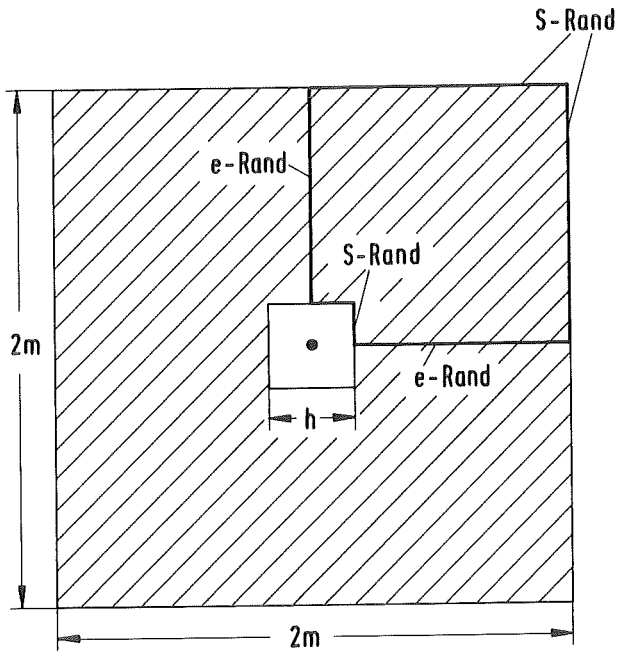


Abb. 8.6: Die behandelte rechteckige Scheibe mit zentralem Loch

Die vollständigen Eingabedaten für Preprozessor und Rechenprogramm sind in den Tabellen 8.3 und 8.4 aufgeführt. Dies Wirbelstromproblem wurde schon ausführlich in [26] untersucht. Es ist nämlich in diesem Fall mit einigen Vereinfachungen möglich, die induzierten Wirbelströme auch analytisch zu berechnen. Bei dem in [26] durchgeführten Vergleich stellte sich heraus, daß der ERATO-Code äußerst genau rechnet.

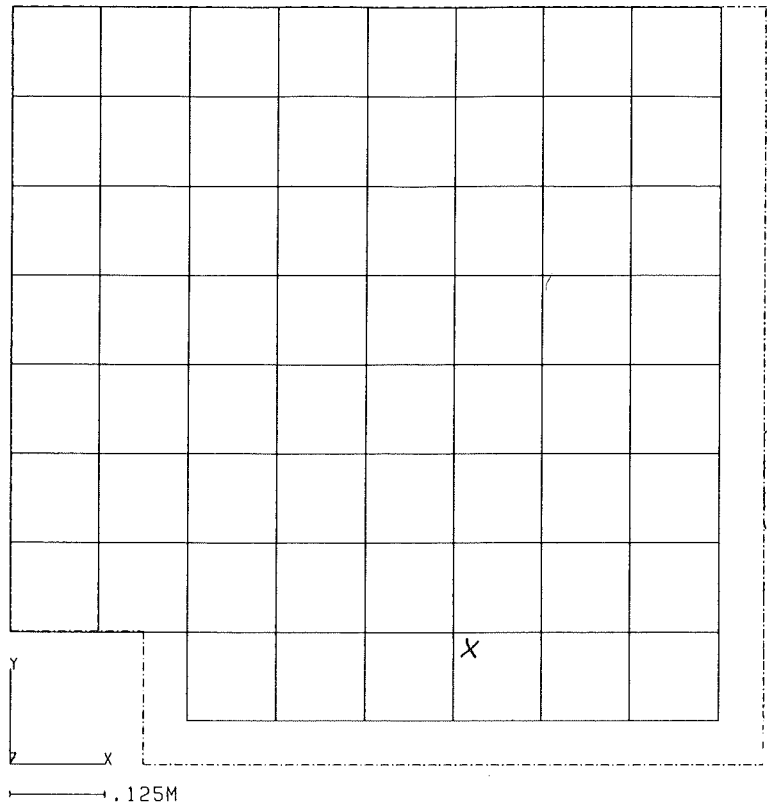
```
&IN  LX = 1., LY = 1., D = 0.02, IRU = 0, IRL = 0, NX = 8, NY = 8,
     RHO = 2.7027E-8, NLAY = 1, LPRINT = .FALSE., KENN = 'FLPLBSP',
     LLOCH = .TRUE., LLOX = 0.176, LLOY = 0.176,
&END
```

Tabelle 8.3: Eingabedaten für Preprozessor FLATPL

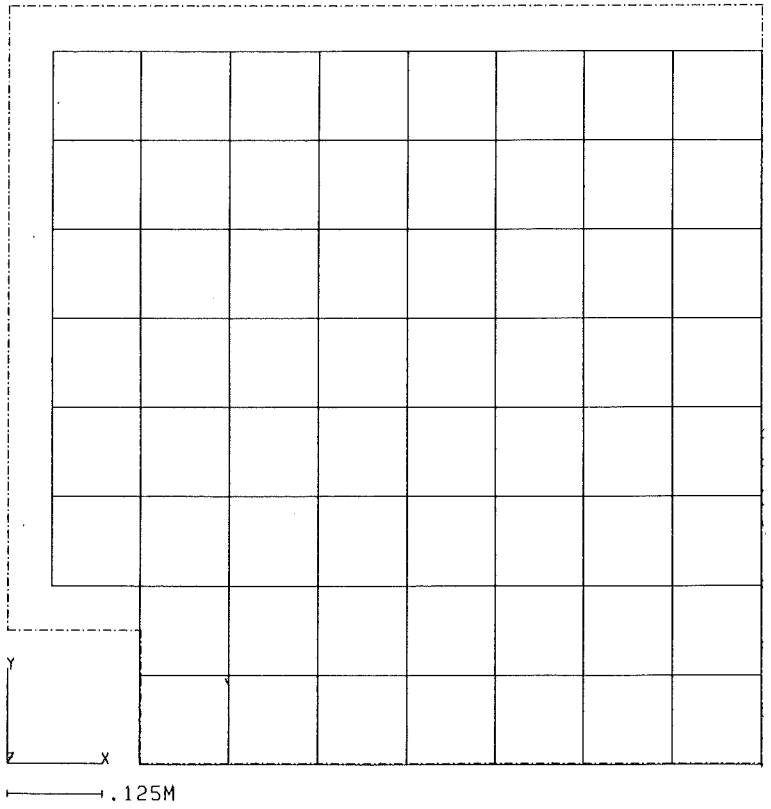
```
&DAT1  ISTEPL = 4, RUNIDL = 'FLPLBSP ',
       COMM(1) = ' BEISPIELJOB ERATO MIT PREPROZESSOR FLATPL  ',
       IPRINT(1) = 0, IPRINT(2) = 0, IPRINT(3) = 0,
       IPRINT(4) = 0, IPRINT(5) = 1, IPRINT(6) = 0, IPRINT(7) = 0,
       IPRINT(8) = 0, IPRINT(9) = 0, IPRINT(10) = 0,IPRINT(11) = 0,
       IPRINT(12) = 0, IPRINT(14) = 0, IPRINT(17) = 0, IPRINT(18) = 0,
       IPRINT(19) = 0, IPRINT(20) = 0, IPRINT(21) = 0, IPRINT(22) = 0,
       IPRINT(23) = 0, IPRINT(24)=0, IPRINT(25)=0,
&END
&DAT2  NEX = 1, IDICK = 1, DT = 1.E-3, TLIM = 80.E-3,
       OUTINT = 1.E-3, ITSW = 1, FREL = 1.E-5,
       NEXGES = 1, IRAMP = 0, ICONST = 0,
&END
&EXT2  NCOIL = 1  &END
&EXT1  RAD = 0.5, CENT(3) = 0.29 &END
```

Tabelle 8.4: Eingabedaten für Rechenprogramm ERATO

Hier sollen zur Demonstration des ERAPLO-Plotprogrammes nur beispielhaft einige Ergebnisse gezeigt werden. So sind in Abb. 8.7 die von FLATPL generierten Elementeinteilungen zu erkennen. Abb. 8.8 zeigt in einer dreidimensionalen Darstellung die zur Zeit t = 5ms induzierte Stromdichte. Abbildung 8.9 schließlich gibt für ein Element den zeitlichen Verlauf des induzierten Stromes wieder.



(a)



(b)

Abb. 8.7: Die vom Preprozessor FLATPL erzeugte Elementeinteilung in einer Projektion senkrecht zur Scheibenfläche
(a) x-Platten; (b) y-Platten

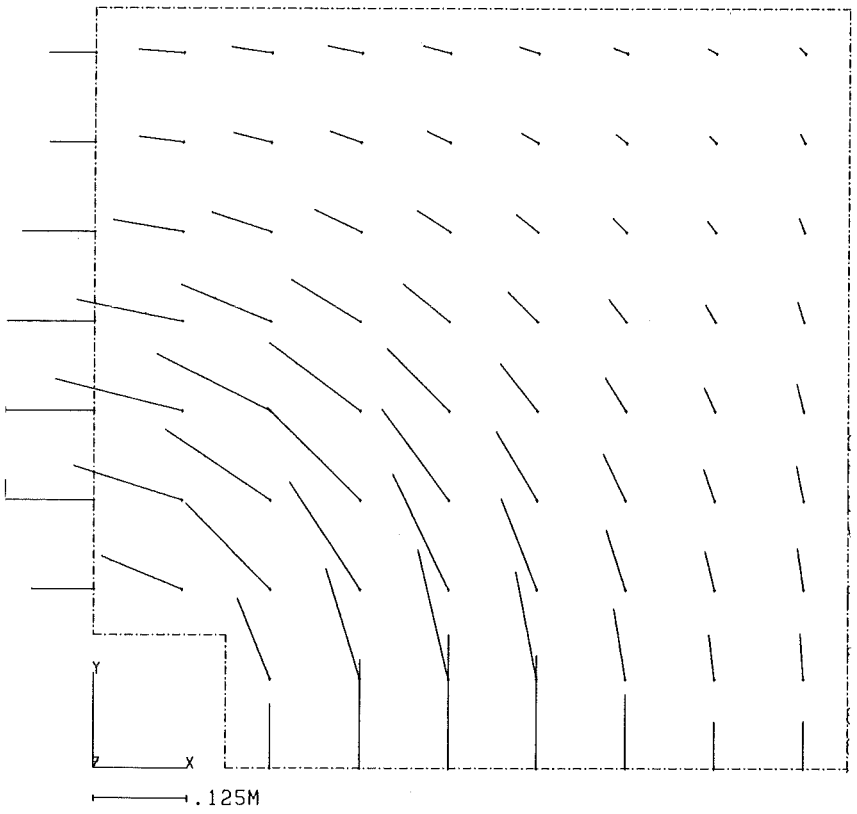


Abb. 8.8: Die zur Zeit t=5ms induzierte Stromdichteverteilung; dieselbe Projektion wie in Abb. 8.7.

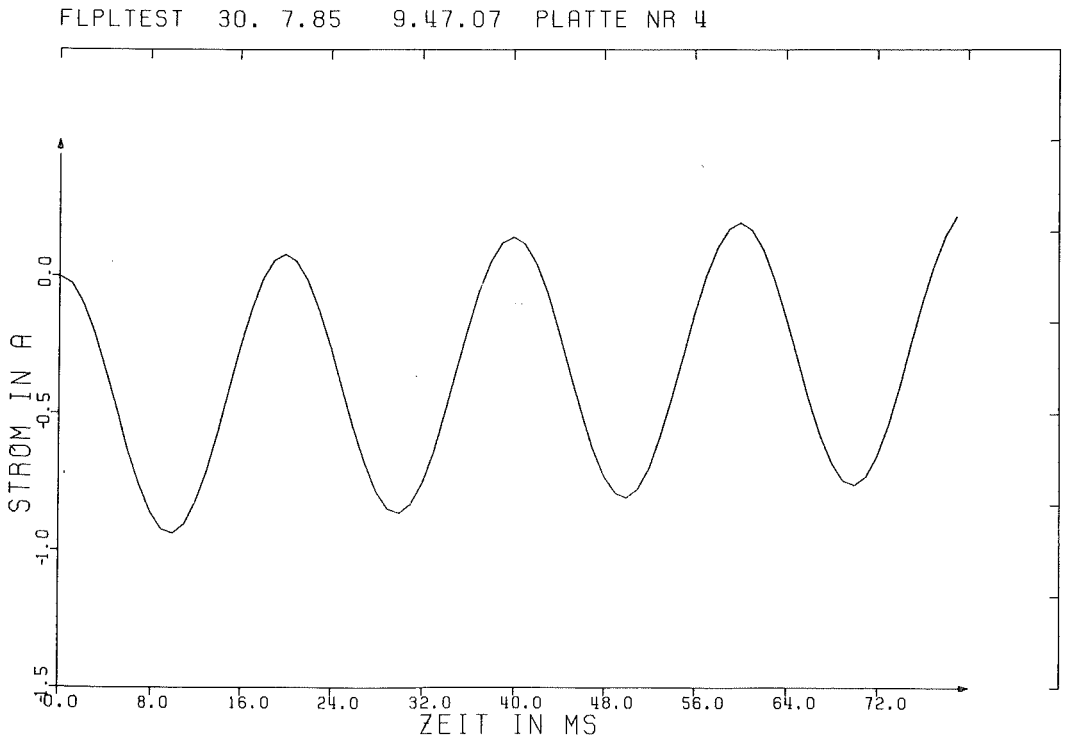


Abb. 8.9: Zeitlicher Verlauf des induzierten Stromes in Platte Nr. 4 (in Abb. 8.7(a) mit einem × gekennzeichnet)

9. LITERATUR

[1] H. Preis: Die Finite-Elemente-Netzwerkmethode für dreidimensionale Wirbelstromberechnungen in Experimenten zur kontrollierten Kernfusion; Arch. f. Elektrotechnik 65 (1982), 233 - 239.

[2] H. Preis, F. Schneider: Bestimmung von Wirbelströmen im Vakuumgefäß einer Apparatur zur Erforschung der kontrollierten Kernfusion; Arch. f. Elektrotechnik 62 (1980), 181 - 186.

[3] N. Gottardi, F. Mast, H. Preis, R. Süß: Diffusion of the Vertical Field into the Tape-Wound Magnet; IPP 1/189 (1981).

[4] U.R. Christensen: Time Varying Eddy Currents on a Conducting Surface in 3-D Using a Network Mesh Method; PPPL - 1516 (1979).

[5] D.W. Weissenburger, U.R. Christensen: Transient Eddy Currents on Finite Plane and Toroidal Conducting Surfaces; PPPL - 1517 (1979).

[6] D.W. Weissenburger: SPARK Version One: Reference Manual; PPPL - 2040 (1983).

[7] L.R. Turner, R.J. Lari: Eddy Current Calculations with the Program EDDYNET, Incorporating Loop Currents and Quadrilateral Mesh; IEEE Trans. Magn., Vol. MAG-16 (1980), 1095 - 1097.

[8] L.R. Turner, R.J. Lari: Applications and Further Developments of the Eddy Current Program EDDYNET; IEEE Trans. Magn., Vol. MAG-18 (1982), 416 - 421.

[9] A. Kameari: Transient Eddy Current Analysis on Thin Conductors with Arbitrary Connections and Shapes; J. Comp. Phys. 42 (1981), 124 - 140.

[10] R.D. Pillsbury: NLMMAP - A Two Dimensional Finite Element Program for Transient or Static, Linear or Nonlinear Magnetic Field Problems; IEEE Trans. Magn., Vol. MAG-18 (1982), 406 - 410.

[11] P. Molfino, G. Molinari, A. Viviani: A User Oriented Modular Package for the Solution of General Field Problems under Time Varying

Conditions; IEEE Trans. Magn., Vol. MAG-18 (1982), 638 - 643.

[12] C.V. Dodd, W.E. Deeds: Analytical Solutions to Eddy-Current Probe-Coil Problems; J. Appl. Phys. 39 (1968), 2829 - 2838.

[13] W. Dörfler, J. MÜhlbacher: Graphentheorie für Informatiker; Walter de Gruyter, New York (1973).

[14] K. Simonyi: Theoretische Elektrotechnik; VEB Deutscher Verlag der Wissenschaften, Berlin (1968).

[15] I.N. Bronstein, K.A. Semendjajew: Taschenbuch der Mathematik; Verlag Harry Deutsch, Zürich, 15. Aufl. (1975).

[16] F. Katz: Unveröffentlichter Bericht (1985).

[17] G. Enderle et al.: Gipsy-Handbuch; KfK 2878 (1980).

[18] N. Balabanian, T.A. Bickart: Electrical Network Theory; John Wiley & Sons Inc., New York (1969).

[19] J. Stoer: Einführung in die numerische Mathematik I; Springer Heidelberger Taschenbücher 105, Berlin (1972).

[20] R. Bulirsch: Numerical Calculation of Elliptic Integrals; Numerische Mathematik 7 (1965), pp. 78 - 80.

[21] R. Zurmühl: Praktische Mathematik für Ingenieure und Physiker; Springer, Berlin (1963).

[22] R. Bulirsch, J. Stoer: Numerical Treatment of Ordinary Differential Equations by Extrapolation Methods; Numerische Mathematik 8 (1966), pp. 1 - 13.

[23] H. Preis: Berechnung des magnetischen Feldes, der magnetischen Kräfte und des Betriebsverhaltens großer Spulensysteme für Fusions-experimente; IPP III/24 (1976).

[24] S. Förster et al.: Superconducting Poloidal Field Coil Development;

Proc. 13th. Symp. on Fusion Technology, Varese, Sept. 24. - 28. (1984), pp. 1475 - 1480.

[25] J. Benner: Unveröffentlichter Bericht (1985).

[26] J. Benner: Unveröffentlichter Bericht (1984).

Anhang 1: Schnittstelle Preprozessor - Rechenprogramm ERATO

Im weiteren wird die Schnittstelle zwischen den Preprozessoren und dem Rechenprogramm ERATO spezifiziert. Alle Ausgaben des Preprozessors sind unformatiert. Die angegebenen Recordlängen sind bei Variablen vom Typ CHARACTER (C*n) die Anzahl der Bytes, bei INTEGER (I) und REAL (R) Variablen die Anzahl der Werte.

Anzahl Records	Typ	Länge	Variablen-name	Bedeutung
1	C*8	3*8	DAT,TIM,KENN	DAT - Datum Preprozessorlauf (dd.mm.yy) TIM - Uhrzeit Preproz.lauf (hh.mm.ss) KENN - Kennung Preprozessorlauf
1	I	4	NPL, ISYM, NSEC, NNODE	NPL - Anzahl der Elemente NSEC - Anzahl der azimutalen Segmente ISYM - Symmetrie bzgl. z=0 NNODE - Anzahl der Netzwerksknoten
14	R	NPL	PLATE(NPL,14)	s.u.
3	I	NPL	BR(NPL,3)	s.u.

Erläuterungen

Jedes vom Preprozessor erzeugte Element wird über seine Nummer j ($1 \leq j \leq NPL$) referiert. Bei der Durchnumerierung wird folgendermassen vorgegangen:

- Zuerst kommen die toroidalen (x-) Platten, danach die poloidalen (y-) Platten.

- Gestartet wird mit den Elementen am Rand in negativer toroidaler Richtung (am y=0 Rand).

- Nacheinander werden die auf derselben toroidalen (y-) Position liegenden Elemente durchnumeriert, beginnend mit dem ersten poloidalen Abschnitt (bei x=0).

- Ist eine toroidale (y-) Position abgearbeitet, wird in toroidaler (y-) Richtung fortfahrend die nächste Elementreihe durchnumeriert.

- An Fehlstellen (Löchern) wird die Numerierung ohne Unterbrechung

fortgesetzt.

Die Geometrie der Elementeinteilung wird durch das REAL-Feld PLATE(NPL,14) beschrieben. Der erste Feldindex gibt dabei die Elementnummer an. In diesem Feld werden Angaben über die Koordinaten der Element-Mittelfläche, die Elementdicke senkrecht zur Mittelfläche, sowie den elektrischen Widerstand der Elemente nach folgendem Schema gespeichert (s. Abb. A1):

PLATE(J,1-3) - (x,y,z)-Koordinate des Eckpunktes 1 des J-ten Elementes
PLATE(J,4-6) - (x,y,z)-Koordinate des Eckpunktes 2 des J-ten Elementes
PLATE(J,7-9) - (x,y,z)-Koordinate des Eckpunktes 3 des J-ten Elementes
PLATE(J,10-12) - (x,y,z)-Koordinate des Eckpunktes 4 des J-ten Elementes
PLATE(13) - Elementdicke
PLATE(14) - elektrischer Widerstand

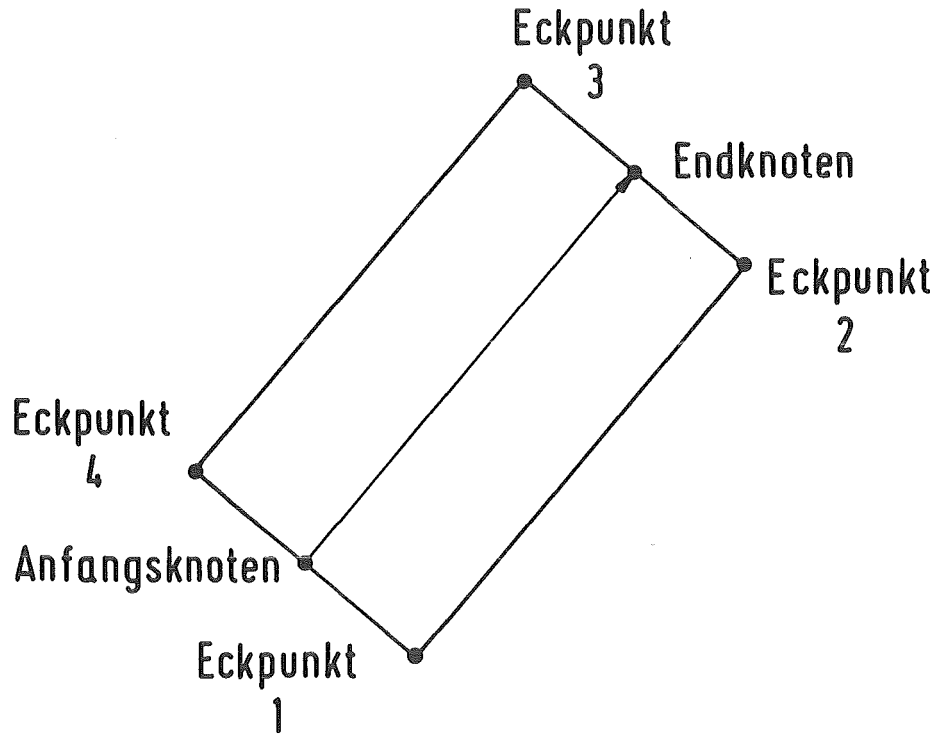


Abb. A1: Geometrische Beschreibung eines Elementes

Die Netzwerkstopologie wird durch das INTEGER-Feld BR(NPL,3) beschrieben. Hierin wird für jeden Netzwerkszweig (=Element) folgendes gespeichert:

BR(J,1) - "Typ" des J-ten Zweiges

BR(J,2) - Nummer des Anfangsknotens des J-ten Zweiges
BR(J,3) - Nummer des Endknotens des J-ten Zweiges

Dabei ist der Zweigtyp entweder Null, oder eine positive, zweistellige Zahl. Ein Typ Null bedeutet, daß der zugehörige Zweig nicht materiell vorhanden ist, sondern es sich um einen künstlichen Verbindungszweig zwischen zwei e-Rändern handelt. Ist der Typ eine zweistellige Zahl der Form "nm", so gilt:

n - Nummer der Lage, in der das Element liegt
m - Elementart; m = 1: x- bzw. poloidale Platte
m = 2: y- bzw. toroidale Platte
m > 2: undefiniert

Anhang 2: Schnittstelle Preprozessor - Postprozessor

Im weiteren werden die Schnittstellen zwischen den Preprozessoren und den Postprozessoren spezifiziert. Alle Ausgaben eines Preprozessors sind unformatiert, die Konversion auf formatierte Werte für die Eingabe in ERAPLO erfolgt mit dem Programm CUFOFO (s. Anhang 4). Die angegebenen Recordlängen sind bei Variablen vom Typ CHARACTER (C*n) die Anzahl der Bytes, bei INTEGER (I) und REAL (R) Variablen die Anzahl der Werte.

Schnittstelle Preprozessor - Plotprogramm ERAPLO

Anzahl Records	Typ	Länge	Variablen-name	Bedeutung
1	C*20	20	HEAD	HEAD = 'TOROIDALE GEOMETRIE'
1	C*8	3*8	DAT,TIM,KENN	DAT - Datum Preprozessorlauf (dd.mm.yy) TIM - Uhrzeit Preproz.lauf (hh.mm.ss) KENN - Kennung des Laufes
1	I	8	NBD, NBDPM, NE, NPR, NPIR, NLAY, NKX, NKY	NBD - Anzahl Strukturränder (werden als Polygonzüge dargestellt) NBDPM - Max. Anzahl v. Punkten auf einem Rand NE - Anzahl der e-Ränder NPR - Anzahl regulärer Knoten (s.u.) NPIR - Anzahl irregulärer Knoten (s.u.) NLAY - Anzahl Elementlagen NKX - Anzahl Knoten in x / poloidaler Richtung NKY - Anzahl Knoten in y / toroidaler Richtung
1	I	2*NBD	NBDP(NBD,2)	Für jede Randkurve j (1≤j≤NBD) gibt NBDP(j,1) die Punktanzahl auf diesem Polygonzug an; NBDP(j,2) gibt an, ob es sich um einen s-Rand (≠0) oder e-Rand (=0) handelt
NBD	R	3*NBDP (J,1)	BD(3,NBDPM, NBD)	Für Randkurve J (1≤J≤NBD) gibt BD(*,I,J) die (x,y,z) Koordinate des I-ten Punktes (1≤I≤NBDP(J,1)) an
1 (Nur bei NE > 0)	I	NE	NKE(NE)	Den e-Rändern im Netzwerk zugeordnete Knotennummern (s.u.)
1	I	NPR	IPF(NPR)	Nummern der regulären Knoten (s.u.)
1	R	9*NRP	PF(9,NPR)	Angaben zu regulären Knoten (s.u.)

1 Nur bei NPIR > 0	I	NPIR	IPFIR(NPIR)	Nummern der Netzwerkszweige, die irreguläre Knoten berühren (s.u.)
1 Nur bei NPIR > 0	R	9*NPIR	PFIR(9,NPIR)	Angaben zu irregulären Knoten (s.u.)
1	I	NLAY	NKO(NLAY)	NKO(j) (1≤j≤NLAY) gibt die Anzahl der entfernten regulären Knoten in Lage j an
NLAY	I	2*NKO(J) Nur bei NKO(J) > 0	NKOXY (2,NKOM,NLAY)	NKOM - Maximale Anzahl entfernter Knoten in einer Lage NKOXY(*,I,J) gibt die Lage des I-ten entfernten Knotens in Lage I im Rechteckgitter an

Erläuterungen

Bei den Knoten des elektrischen Netzwerkes wird unterschieden zwischen regulären und irregulären Knoten. Als irregulär werden alle diejenigen Knoten bezeichnet, die auf einem e-Rand liegen. Gemeinsames Merkmal dieser Knoten ist, daß ihnen kein eindeutiger geometrischer Ort zugeordnet werden kann, weil mit ihnen viele, sich gegenseitig nicht berührende Elemente inzident sind. Vom Preprozessor werden zur Charakterisierung dieser Knoten die folgenden Informationen an den Postprozessor übergeben:

- Die Nummern der irregulären Knoten (Feld NKE)

- Die Nummern aller Zweige, die einen irregulären Knoten als Anfangs- oder Endknoten haben (Feld IPFIR)

- Angaben über die geometrischen Orte der irregulären Knoten in den durch IPFIR bestimmten Zweigen, sowie über die Orientierung der Struktur in diesen Punkten (Feld PFIR). Das Feld PFIR ist genauso aufgebaut wie PF (s.u.).

Alle übrigen Knoten werden als regulär bezeichnet. Hierfür wird gespeichert:

- Die zugehörigen Knotennummern (Feld IPF)

- Geometrischer Ort und Strukturorientierung (Feld PF). Das Feld PF ist folgendermaßen aufgebaut: Für den regulären Knoten J bezeichnet

PF(1-3,J) - (x,y,z)-Koordinate des geometrischen Ortes des Knotens. Dieser Ort wird nach den Regeln des Überlappungsprinzips festgelegt, er liegt immer auf der erzeugenden Konturfläche

PF(4-6,J) - Einheitsvektor in x- / poloidaler Richtung am Knotenort (s. Abb. 4.6)

PF(7-9,J) - Einheitsvektor in y- / toroidaler Richtung am Knotenort (s. Abb. 4.6)

Schnittstelle Preprozessor - Postprozessor DRUCK

Anzahl Records	Typ	Länge	Variablen-name	Bedeutung
1	I	6	NLAY, NLM, NX, NY, NYX, NYY	NLAY - Anzahl der Lagen NLM - Anzahl der entfernten Platten (s. Kap. 6.1.1) NX - Anzahl der x-Platten in x-Richt. / poloidalen Platten in pol. Richt. NY - Anzahl der x-Platten in y-Richt. / poloidalen Platten in tor. Richt. NYX - Anzahl der y-Platten in x-Richt./ toroidalen Platten in pol. Richt. NYY - Anzahl der y-Platten in y-Richt./ toroidalen Platten in tor. Richt.
1 nur wenn NLM>0	I	NLM	ILM(NLM)	ILM(J) - Nummer der J-ten entfernten Platte vor der Entfernung aus dem Netzwerk

Anhang 3: Schnittstelle Rechenprogramm - Postprozessoren

Ausgabe der Zweigmatrizen

Anzahl Records	Typ	Länge	Variablen-name	Bedeutung
1	C*20	20	HEAD	HEAD = 'HEADER'
1	C*8	3*8	DAT, TIM, RUNID	DAT - Datum Rechenlauf (dd.mm.yy) TIM - Uhrzeit Rechenlauf (hh.mm.ss) RUNID - Kennung Rechenlauf
1	C*80	10*80	COMM(10)	Beliebiger Text
1	I	1	NPLM	Anzahl der Elemente, für die ein Stromwert berechnet wird
NPLM	R	NPLM	LBR (NPLM,NPLM)	Zweig-Induktivitätsmatrix
1	R	NPLM	RBR(NPLM)	Diagonale der Zweig-Widerstandsmatrix

Geometrie, Wirbelströme

Anzahl Records	Typ	Länge	Variablen-name	Bedeutung
1	C*20	20	HEAD	HEAD = 'HEADER'
1	C*8	3*8	DAT, TIM, RUNID	DAT - Datum Rechenlauf (dd.mm.yy) TIM - Uhrzeit Rechenlauf (hh.mm.ss) RUNID - Kennung Rechenlauf
1	C*80	10*80	COMM(10)	Beliebiger Text
1	C*20	20	HEAD	HEAD = 'PLATTENDATEN'
1	C*8	3*8	DAT, TIM, RUNID	DAT - Datum Preprozessorlauf (dd.mm.yy) TIM - Uhrzeit Preproz.lauf (hh.mm.ss) RUNID - Kennung Preprozessorlauf
1	I	4	NPLM, ISYM, NSEC, NNODE	NPLM - Anzahl Elemente, in denen Stromwerte berechnet werden ISYM, NSEC, NNODE - s. Anhang 1
14	R	NPLM	PLATE (NPLM,14)	s. Anhang 1
3	I	NPLM	BR(NPLM,3)	s. Anhang 1
1	C*20	20	HEAD	HEAD = 'EXTERNE SPULEN'

1	I,R	4	NEX1, NEXGES IRAMP (I), STROM (R)	NEX1 - Anzahl der Kreisgruppen NEXGES - Anzahl der externen Kreise IRAMP, STROM - s. Kap. 5.7.1
1	I	NEX1	NCOIL(NEX1)	NCOIL(J) - Anzahl externer Kreise in Kreisgruppe J
1	R	8* NEXGES	COIL (NEXGES,8)	COIL(J,1-3) - Mittelpkt. des J-ten externen Kreises COIL(J,4-6) - Normalenvektor COIL(J,7) - Radius COIL(J,8) - Windungszahl (1 ≤ j ≤ NEXGES)
Die folgender 3 Records werden so oft herausgeschrieben, wie es durch TBEG, TLIM und OUTINT (s. Kap. 5.7.1) festgelegt ist				
1	C*20	20	TEXT	TEXT = 'EDDY-CURRENTS'
1	I,R	2	IZAHL (I), TIME (R)	IZAHL - Nummer des Zeitschrittes TIME - Zeitpunkt
1	R	NPLM	AB(NPLM)	Zweigströme [A]
1	C*20	20	HEAD	HEAD = 'MAGNETFELD'
1	I	1	NPMAG	Anzahl Punkte, an denen das sekundäre Magnetfeld berechnet werden soll
1	R	3*NPMAG	P(3,NPMAG)	P(*,J) gibt die (X,y,z) Koordinate des J-ten Magnetfeldpunktes an
Der folgende Record wird so oft herausgeschrieben, wie es durch TMIN und TMAX (s. Kap. 5.7.1) festgelegt ist.				
1	R	3*NPMAG +1	TIME, B(3,NPMAG)	TIME - Zeitpunkt B(*,J) - (x,y,z) Komponente des Magnetfeldes am Punkt J

Anhang 4: Liste des Programmes CUFOFO, Datenübertragung CYBER - IBM

Anmerkung: Die folgende Liste gibt die IBM-Version des Programmes CUFOFO wieder

```
C   TORUS     UPDATE FUER SYSTEM   IBM      VOM 84-12-18
C
C KONVERSION DER AUSGABEDATEN DES PREPROZESSORS UND DES ERATO CODES
C VON UNFORMATIERTEN AUF FORMATIERTE WERTE, SOWIE UEBERTRAGUNG VON
C DER CYBER AUF DIE IBM-ANLAGE DES KFK.
C
C EIN/AUSGABEDATEIEN:
C
C    FILE OUTPUT1 (UNIT10) : AUSGABEFILE OUTPUT1 DES ERATO-CODES
C                            (L- UND R - BRANCHMATRIZEN)
C    FILE OUTPUT2 (UNIT11) : AUSGABEFILE OUTPUT2 DES ERATO-CODES
C                            (PLOTDATEN)
C    FILE BNDDAT  (UNIT12) : AUSGABEFILE BNDDAT DES PREPROZESSORS
C                            (DATEN DER RANDKURVEN)
C    FILES OUT1,OUT2,OUT3 (UNIT20,UNIT21,UNIT22) :
C            ZUGEHOERIGE AUSGABEFILES FUER DIE FORMATIERTEN WERTE
C
C EINGABEPARAMETER:
C
C     KEINE EINGABEPARAMETER
C
      PARAMETER (NPLM = 2500 , NBDM = 100 , NBDPMM = 500 , NPFM = 1000
     1           NKOM = 50)
C
      REAL*8 DH(NPLM) , DBD(3,NBDPMM) , DPF(9,NPFM) , DSTROM , DTIME
      REAL H(NPLM) , BD(3,NBDPMM) , PF(9,NPFM)
      INTEGER IH(NPLM) , NBDP(NBDM,2) , IPF(NPFM) , NKOXY(1:2,1:NKOM)
      CHARACTER JOBID(3)*2 , DATEI(3)*30 , HEAD*20 , DAT*8 ,
     1     TIM*8 , KENN*8 , COMM(10)*60 , UNIT(3)*4 , TAPE(3)*6 ,
     2     LABEL(3)*2 , DATNAM(3)*80
C
      IEIN = 10
      IOUT = 20
C
      READ (IEIN,END=3200,ERR=3250) HEAD
      IF (HEAD.NE.'HEADER              ') THEN
         WRITE (6,1120) HEAD
 1120 FORMAT(' FALSCHE KENNUNG BEI EINLESEN VON OUTPUT2: ',A8)
         GO TO 3250
      END IF
C
      WRITE (IOUT,2010) HEAD
C
      READ (IEIN,END=3200,ERR=3250) DAT , TIM , KENN
      WRITE (IOUT,2020) DAT , TIM , KENN
C
      READ (IEIN,END=3200,ERR=3250) (COMM(J) , J = 1 , 10)
      DO 28 J =  1 , 10
         WRITE(IOUT,2030) COMM(J)
 28   CONTINUE
C
      READ (IEIN,END=3200,ERR=3250) NPL
      WRITE (IOUT,2040) NPL
C
```

```
      DO 25 I = 1 , NPL
        READ (IEIN,END=3200,ERR=3250) (DH(J) , J = 1 , NPL)
        DO 6000 J = 1 , NPL
          H(J) = SNGL(DH(J))
 6000   CONTINUE
        WRITE (IOUT,2050) (H(J) , J = 1 , NPL)
 25   CONTINUE
C
        READ (IEIN,END=3200,ERR=3250) (DH(J) , J = 1 , NPL)
        DO 6010 J = 1 , NPL
          H(J) = SNGL(DH(J))
 6010   CONTINUE
        WRITE (IOUT,2050) (H(J) , J = 1 , NPL)
C
C
      CALL SPACE(6,5)
      WRITE (6,1010)
 1010 FORMAT(' DATEI OUTPUT1 WURDE KONVERTIERT, DATENUEBERTRAGUNG AUF'
     1   ' IBM VORBEREITET')
C
      GO TO 3600
C
 3200 WRITE (6,1135)
 1135 FORMAT(' END-OF-FILE BEI EINLESEN VON OUTPUT1')
      GO TO 3600
C
 3250 WRITE (6,1138)
 1138 FORMAT(' FEHLER BEIM EINLESEN VON OUTPUT1')
C
 3600 IEIN = 11
      IOUT = 21
C
      READ (IEIN,END=3000,ERR=3100) HEAD
      IF (HEAD.NE.'HEADER              ') THEN
         WRITE (6,1020) HEAD
 1020 FORMAT(' FALSCHE KENNUNG BEI EINLESEN VON OUTPUT2: ',A8)
         GO TO 3100
      END IF
C
      WRITE (IOUT,2010) HEAD
 2010 FORMAT (1X,A20)
C
      READ (IEIN,END=3000,ERR=3100) DAT , TIM , KENN
      WRITE (IOUT,2020) DAT , TIM , KENN
 2020 FORMAT(1X,3A8)
C
      READ (IEIN,END=3000,ERR=3100) (COMM(J) , J = 1 , 10)
      DO 20 J =  1 , 10
         WRITE(IOUT,2030) COMM(J)
 20   CONTINUE
 2030 FORMAT(1X,A60)
C
      READ (IEIN,END=3000,ERR=3100) HEAD
      IF (HEAD.NE.'PLATTENDATEN        ') THEN
         WRITE (6,1020) HEAD
         GO TO 3100
      END IF
C
      WRITE (IOUT,2010) HEAD
C
```

```
      READ (IEIN,END=3000,ERR=3100) DAT , TIM , KENN
      WRITE (IOUT,2020) DAT , TIM , KENN
C
      READ (IEIN,END=3000,ERR=3100) NPL , ISYM , NSEC , NNODE
      WRITE (IOUT,2040) NPL , ISYM , NSEC , NNODE
 2040 FORMAT(1X,4I5)
C
      DO 30 I = 1 , 14
         READ (IEIN,END=3000,ERR=3100) (DH(J) , J = 1 , NPL)
         DO 6020 J = 1 , NPL
           H(J) = SNGL(DH(J))
 6020    CONTINUE
         WRITE (IOUT,2050) (H(J) , J = 1 , NPL)
 2050 FORMAT(1X,6E12.4)
 30   CONTINUE
C
      DO 40 I = 1 , 3
         READ (IEIN,END=3000,ERR=3100) (IH(J) , J = 1 , NPL)
         WRITE (IOUT,2060) (IH(J) , J = 1 , NPL)
 2060 FORMAT(1X,15I5)
 40   CONTINUE
C
      READ (IEIN,END=3000,ERR=3100) HEAD
      IF (HEAD.NE.'EXTERNE SPULEN') THEN
        WRITE (6,1020) HEAD
        GO TO 3100
      END IF
C
      WRITE (IOUT,2010)HEAD
C
      READ (IEIN) NEX1 , NEXGES , IRAMP , DSTROM
      STROM = SNGL(DSTROM)
      WRITE (IOUT,2170) NEX1 , NEXGES , IRAMP , STROM
 2170 FORMAT(1X,3I4,E12.4)
C
      READ (IEIN) (IH(J) , J = 1 , NEX1)
      WRITE (IOUT,2060) (IH(J) , J = 1 , NEX1)
C
      NHHH = 8*NEXGES
      READ (IEIN) (DH(J) , J = 1 , NHHH)
      DO 6030 J = 1 , NHHH
        H(J) = SNGL(DH(J))
 6030 CONTINUE
      WRITE (IOUT,2050) (H(J) , J = 1 , NHHH)
C
 2400 READ (IEIN,END=2500,ERR=3100) HEAD
      IF (HEAD.NE.'EDDY-CURRENTS') GO TO 2500
      WRITE (IOUT,2010) HEAD
C
      READ (IEIN) IZAHL , DTIME
      TIME = SNGL(DTIME)
      WRITE (IOUT,2080) IZAHL,TIME
 2080 FORMAT(1X,I4,E12.4)
C
      READ (IEIN) (DH(J) , J = 1 , NPL)
      DO 6040 J = 1 , NPL
        H(J) = SNGL(DH(J))
 6040 CONTINUE
      WRITE (IOUT,2050) (H(J) , J = 1 , NPL)
C
```

```
      GO TO 2400
C
C
 2500 CONTINUE
      CALL SPACE(6,5)
      WRITE (6,1030)
 1030 FORMAT(' DATEI OUTPUT2 WURDE KONVERTIERT, DATENUEBERTRAGUNG',
     1    ' AUF IBM VORBEREITET')
      GO TO 3500
C
 3000 WRITE (6,1035)
 1035 FORMAT(' END-OF-FILE BEI EINLESEN VON OUTPUT2')
      GO TO 3500
C
 3100 WRITE (6,1038)
 1038 FORMAT(' FEHLER BEIM EINLESENM VON OUTPUT2')
C
 3500 IEIN = 12
      IOUT = 22
C
      READ (IEIN,END=3050,ERR=3150) HEAD
      WRITE (IOUT,2010) HEAD
C
      READ (IEIN,END=3050,ERR=3150) DAT , TIM , KENN
      WRITE (IOUT,2020) DAT , TIM , KENN
C
      READ (IEIN,END=3050,ERR=3150) NBD , NBDPM , NE , NPR , NPIR ,
     1      NLAY , NKX , NKY
      WRITE (IOUT,2070) NBD , NBDPM , NE , NPR , NPIR , NLAY , NKX , NKY
 2070 FORMAT(1X,8I5)
C
      IF (NBD.GT.0) THEN
        READ(IEIN,END=3050,ERR=3150) ( (NBDP(I,J),I = 1 , NBD) , J=1,2)
        DO 50 J = 1 , 2
          WRITE (IOUT,2150)(NBDP(I,J) , I = 1 , NBD)
 2150     FORMAT(1X,15I5)
 50     CONTINUE
C
        DO 60 K = 1 , NBD
          KK = NBDP(K,1)
          READ(IEIN,END=3050,ERR=3150) ( (DBD(I,J) , I=1,3) , J=1,KK)
          DO 6050 J = 1 , KK
          DO 6050 I = 1 , 3
            BD(I,J) = SNGL(DBD(I,J))
 6050     CONTINUE
          WRITE (IOUT,2050)( (BD(I,J) , I = 1 ,3), J = 1,KK)
 60     CONTINUE
      END IF
C
      IF (NE.GT.0) THEN
        READ (IEIN,END=3050,ERR=3150) (IH(J) , J = 1 , NE)
        WRITE (IOUT,2150) (IH(J) , J = 1 , NE)
      END IF
C
      READ (IEIN,END=3050,ERR=3150) (IPF(J) , J = 1 , NPR)
      WRITE (IOUT,2150) (IPF(J) , J = 1 , NPR)
C
      READ (IEIN,END=3050,ERR=3150) ( (DPF(K,J) , K = 1,9) , J = 1,NPR)
        DO 6060 J = 1 , NPR
        DO 6060 K = 1 , 9
```

```
          PF(K,J) = SNGL(DPF(K,J))
 6060   CONTINUE
      WRITE (IOUT,2050) ( (PF(K,J) , K = 1 , 9) , J = 1 , NPR)
C
      IF (NPIR.GT.0) THEN
        READ (IEIN,END=3050,ERR=3150) (IPF(J) , J = 1 , NPIR)
        WRITE (IOUT,2150) (IPF(J) , J = 1 , NPIR)
C
        READ (IEIN,END=3050,ERR=3150) ( (DPF(K,J) ,K = 1,9) ,J = 1,NPIR)
        DO 6070 J = 1 , NPR
        DO 6070 K = 1 , 9
          PF(K,J) = SNGL(DPF(K,J))
 6070   CONTINUE
        WRITE (IOUT,2050) ( (PF(K,J) , K = 1  , 9) , J = 1 , NPIR)
      END IF
C
      READ (IEIN,END=3050,ERR=3150) (IPF(J) , J = 1 , NLAY)
      WRITE (IOUT,2150) (IPF(J) , J = 1 , NLAY)
C
      DO 70 I = 1 , NLAY
        IF (IPF(I).GT.0) THEN
          READ (IEIN,END=3050,ERR=3150)
     1          ( (NKOXY(J,K) , J = 1 , 2) , K = 1 , IPF(I) )
          WRITE (IOUT,2150) ( (NKOXY(J,K), J = 1 ,2) , K = 1 , IPF(I))
        END IF
 70   CONTINUE
C
C
      CALL SPACE(6,5)
      WRITE (6,1040)
 1040 FORMAT(' DATEI BNDDAT WURDE KONVERTIERT, DATENUEBERTRAGUNG AUF'
     1   ' IBM VORBEREITET')
C
      GO TO 5000
C
 4000 WRITE (6,1050)
 1050 FORMAT(' FEHLER BEIM EINLESEN VON JOBID UND DATEI')
      GO TO 5000
C
 3150 WRITE (6,1070)
 1070 FORMAT(' FEHLER BEIM EINLESEN VON BNDDAT')
      GO TO 5000
C
 3050 WRITE (6,1080)
 1080 FORMAT(' END-OF-FILE BEIN EINLESEN VON BNDDAT')
C
 5000 STOP
      END
```

Datenübertragung CYBER - IBM

Mit dem folgenden Job können beliebige Daten (in Kartenformat) vom CYBER-Vorrechner auf die IBM in die Datei IRE571.XXX.DATA(ZIELDAT) übertragen werden. Die CYBER-Version des CUFOFO Programmes erzeugt automatisch einen derartigen Job.

```
//IRE571E JOB (Account-Nummer),BENNER,USER=IRE5,PASSWORD= ,
//         MSGCLASS=H
/*XEQ   DKAKFK3
//*MAIN ORG=RM006
//* BEISPIEL EINES JOBS, DER VON S7865 GESTARTET AUF KFK RECHNET.
//ST EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=*
//SYSUT1 DD DATA,DLM='|*'
       ...
  ZU UEBERTRAGENDE DATEN
       ...
       ...
|*
//SYSUT2 DD DISP=SHR,DSN=IRE571.XXX.DATA(ZIELDAT)
//SYSIN DD DUMMY
//
```

Anhang 5: Beispiel-JCL für einen CYBER-Job

Die folgende Liste zeigt die JCL für einen vollständigen ERATO-Lauf (Preprozessor, Rechenlauf, Drucken von Listen, Ergebnis-Übertragung zur IBM) auf der CYBER. Mit diesem Job wurden die in Kap. 8.2 gezeigten Ergebnisse produziert.

```
//RHF       EXEC  RHF
 SUBMIT MF=C20,FILE=C20,SYSOUT=X,HOLD=YES,NOTIFY=*
/*
//C20 DD *
JOB,STEKV.
USER(U=135XXX,AC=G6XXXXX,PA=XXX)
RESOURCE(JCAT=UP6,TL=400,WS=1540,LP=10)
COMMENT.RESOURCE(JCAT=UP4,TL=5000,WS=770,LP=6)
COMMENT.RESOURCE(JCAT=UP3,TL=10000,WS=1664,LP=13)
COMMENT.
COMMENT.    **********  PREPROZESSORLAUF F L A T P L     ***********
COMMENT.
COMMENT.    ***      FUER DEN PREPROZESSORLAUF BENOETIGTE DATEIEN ***
ATTACH,FLATPL.
REQUEST,PLATEDAT/600,RT=W.
REQUEST,BNDDAT/200,RT=W.
COMMENT.
COMMENT.    ++++++  EINGABEDATENSATZ FUER PREPROZESSORLAUF  +++++
COMMENT.
MFLINK(EING,ST=S65,JCS="GET,DSN=V571.EUTERPE.INPUT(FLPLBSP)")
COMMENT.
FLATPL(UNIT5=EING)
COMMENT.
COMMENT.    ***  ENDE DES PREPROZESSOR-LAUFES  ***
COMMENT.
COMMENT.
COMMENT.    *******  WIRBELSTROM - BERECHNUNGSPROGRAMM E R A T O  *****
COMMENT.
COMMENT.    ***  BENOETIGTE DATEIEN  ***
COMMENT.
PATTACH,RZPOOL.
ATTACH,ERATO.
REQUEST,OUTPUT1/2000,RT=W.
REQUEST,OUTPUT2/2000,RT=W.
COMMENT.
COMMENT.    ++++++  EINGABEDATENSATZ E R A T O  +++++++
COMMENT.
MFLINK(EING,ST=S65,JCS="GET,DSN=V571.ERATO.INPUT(FLPLBSP)")
COMMENT.
COMMENT.     *** UEBERSETZEN HAUPTPROGRAMM UND ZEITFUNKTION  ***
COMMENT.
MFLINK(SOUR,ST=S65,JCS="GET,DSN=V571.ERATO.DATA(MAIN)")
FORTRAN,I=SOUR,B=BIN/1000,O=BLOUV.
LOAD(BIN,LIB=ERATO,RZLIB,NAGLIB,NAGFORT,F200LIB,
CN=PROG/1000,CDF=20000,GROL=*COMA,*COMIA,L=0)
COMMENT.
PROG(UNIT5=EING)
COMMENT.
COMMENT.    ***  ENDE DES E R A T O  LAUFES  ***
COMMENT.
```

```
COMMENT.     ********  UEBERTRAGUNG DER AUSGABEDATEN AUF VORRECHNER  ****
COMMENT.     ********  UND KFK/IBM                                  ****
COMMENT.
COMMENT.     ***  BENOETIGTE DATEIEN
COMMENT.
ATTACH,CUFOFO.
REQUEST,OUT1/500.
REQUEST,OUT2/500.
REQUEST,OUT3/500.
COMMENT.
COMMENT.
CUFOFO.
COMMENT.
COMMENT.
COMMENT.      *** UEBERTRAGUNG CYBER ----> KFK/IBM                  ***
COMMENT.      *** DATEINAMEN AUF IBM MUESSEN IN DER NAMELIST  "IN" ***
COMMENT.      ***          ANGEGEBEN WERDEN                        ***
COMMENT.
COMMENT.      SWITCH,OUT1,IC=AS.
COMMENT.      MFQUEUE(OUT1,ST=S65,DD=C8,JCS="SUBMIT")
SWITCH,OUT2,IC=AS.
MFQUEUE(OUT2,ST=S65,DD=C8,JCS="SUBMIT")
SWITCH,OUT3,IC=AS.
MFQUEUE(OUT3,ST=S65,DD=C8,JCS="SUBMIT")
COMMENT.
COMMENT.      ***  ENDE DES UEBERTRAGUNGSJOBS  ***
COMMENT.
COMMENT.      *** DRUCKEN DER BERECHNETEN STROMWERTE  ***
COMMENT.
COMMENT.      ***  BENOETIGTE DATEIEN  ***
COMMENT.
COMMENT.
ATTACH,DRUCK.
COMMENT.
COMMENT.     ++++++  EINGABEDATENSATZ D R U C K  +++++++
COMMENT.
MFLINK(EING,ST=S65,JCS="GET,DSN=V571.DRUCK.INPUT(FLPLBSP)")
COMMENT.
DRUCK(UNIT5=EING).
COMMENT.
7-8-9
  &IN DATEI(2) = 'IRE571.PLATES.OUTPUT(FLPLBSP)' ,
      DATEI(3) = 'IRE571.BOUNDS.OUTPUT(FLPLBSP)' ,
  &END
7-8-9
//PURGE EXEC BTSS
SEND 'JOB V571A AN CYBER 205 ABGESCHICKT' U(V571) LOGON
CANCEL V571A PURGE
/*
```

Anhang 6: Beispiel-JCL für einen IBM-Job

Die folgende Liste zeigt die JCL für einen vollständigen ERATO-Lauf (Preprozessor, Rechenlauf, Drucken von Listen, Datenkonversion für ERAPLO) auf der IBM. Mit diesem Job wurden die in Kap. 8.1 gezeigten Ergebnisse produziert.

```
//IRE571B  JOB (Account-Nummer),BENNER,NOTIFY=IRE571,REGION=6500K,
//      TIME=(00,30),MSGCLASS=H
//**MAIN ORG=RM006
//*MAIN LINES=5
//*****************************************************************
//*
//*   PREPROZESSOR    T O R U S
//*
//*****************************************************************
//TOR EXEC F7LG,PARM.L='LIST,MAP'
//*
//L.SYSLIB DD DISP=SHR,DSN=IRE571.TORUS.LOAD
//         DD DISP=SHR,DSN=IRE571.ERATO.LOAD
//         DD DISP=SHR,DSN=SYS7.FORTLIB
//         DD DISP=SHR,DSN=SYS2.HARWELL
//         DD DISP=SHR,DSN=SYS7.FORT2LIB
//         DD DISP=SHR,DSN=SYS2.VERSATEC
//         DD DISP=SHR,DSN=SYS7.IMSL.DP
//L.SYSPRINT DD DUMMY
//L.SYSIN  DD *
 INCLUDE SYSLIB(MAIN)
//G.FT05F001 DD DISP=SHR,DSN=IRE571.EUTERPE.INPUT(TORBSP)
//G.FT06F001 DD SYSOUT=*
//G.FT08F001 DD UNIT=SYSDA,SPACE=(TRK,(50,20),RLSE),DISP=(NEW,PASS),
//   DSN=&&PLAT
//G.FT10F001 DD UNIT=SYSDA,SPACE=(TRK,(50,20),RLSE),DISP=(NEW,PASS),
//   DSN=&&BOUN
//G.FT12F001 DD UNIT=SYSDA,SPACE=(TRK,(50,20),RLSE),DISP=(NEW,PASS),
//   DSN=&&DRUCK
//*
//*****************************************************************
//*
//*    WIRBELSTROMCODE   E R A T O
//*
//*****************************************************************
//ERATO EXEC F7CLG,PARM.C='NOXREF,NOMAP,SMSG,AUTODBL(DBLPAD)',
//      PARM.L='LIST,MAP'
//*
//C.SYSPRINT DD DUMMY
//C.SYSIN  DD DISP=SHR,DSN=IRE571.ERATO.DATA(MAIN)
//*
//L.SYSLIB DD DISP=SHR,DSN=IRE571.ERATO.LOAD
//         DD DISP=SHR,DSN=SYS7.FORTLIB
//         DD DISP=SHR,DSN=SYS2.HARWELL
//         DD DISP=SHR,DSN=SYS7.FORT2LIB
//         DD DISP=SHR,DSN=SYS2.VERSATEC
//         DD DISP=SHR,DSN=SYS7.IMSL.DP
//L.SYSPRINT DD DUMMY
//G.FT05F001 DD DISP=SHR,DSN=IRE571.ERATO.INPUT(TORBSP)
//G.FT06F001 DD SYSOUT=*
//G.FT09F001 DD DUMMY
//G.FT10F001 DD DUMMY
```

```
//G.FT12F001 DD DISP=(OLD,DELETE),DSN=&&PLAT
//G.FT15F001 DD DUMMY
//G.FT16F001 DD UNIT=SYSDA,SPACE=(TRK,(500,100),RLSE),
//     DISP=(NEW,PASS),DSN=&&OUT2
//G.FT25F001 DD UNIT=SYSDA,SPACE=(TRK,(100,20),RLSE),DISP=(NEW,PASS),
//   DSN=&&MAG
//*****************************************************************
//*
//*    AUSDRUCK DER BERECHNETEN WIRBELSTROEME
//*
//*****************************************************************
//DRUCK EXEC F7LG,PARM.L='LIST,MAP'
//***
//L.SYSLIB DD DISP=SHR,DSN=IRE571.ERATO.LOAD
//       DD DISP=SHR,DSN=SYS7.FORTLIB
//       DD DISP=SHR,DSN=SYS2.HARWELL
//       DD DISP=SHR,DSN=SYS7.FORT2LIB
//       DD DISP=SHR,DSN=SYS2.VERSATEC
//       DD DISP=SHR,DSN=SYS7.IMSL.DP
//L.SYSPRINT DD DUMMY
//L.SYSIN DD *
     INCLUDE SYSLIB(DRUCK)
//G.FT05F001 DD DISP=SHR,DSN=IRE571.DRUCK.INPUT(TORBSP)
//G.FT06F001 DD SYSOUT=*
//G.FT10F001 DD DISP=(OLD,PASS),DSN=&&OUT2
//G.FT12F001 DD DISP=(OLD,DELETE),DSN=&&DRUCK
//*****************************************************************
//***
//***  KONVERSION DER AUSGABE AUF FORMATIERTE DATEN
//***
//*****************************************************************
//CUFOFO EXEC F7CLG,PARM.L='LIST,MAP'
//**
//C.SYSIN DD DISP=SHR,DSN=IRE571.ERATO.DATA(CUFOFO)
//C.SYSPRINT DD DUMMY
//L.SYSLIB DD DISP=SHR,DSN=IRE571.ERATO.LOAD
//       DD DISP=SHR,DSN=SYS7.FORTLIB
//       DD DISP=SHR,DSN=SYS2.HARWELL
//       DD DISP=SHR,DSN=SYS7.FORT2LIB
//       DD DISP=SHR,DSN=SYS2.VERSATEC
//       DD DISP=SHR,DSN=SYS7.IMSL.DP
//L.SYSPRINT DD DUMMY
//G.FT05F001 DD DUMMY
//G.FT06F001 DD SYSOUT=*
//G.FT10F001 DD DUMMY
//G.FT11F001 DD DISP=(OLD,DELETE),DSN=&&OUT2
//G.FT12F001 DD DISP=(OLD,DELETE),DSN=&&BOUN
//G.FT20F001 DD DUMMY
//G.FT21F001 DD DISP=SHR,DSN=IRE571.PLATES.OUTPUT(TORBSP)
//G.FT22F001 DD DISP=SHR,DSN=IRE571.BOUNDS.OUTPUT(TORBSP)
```