LA-UR-87-126
c.1

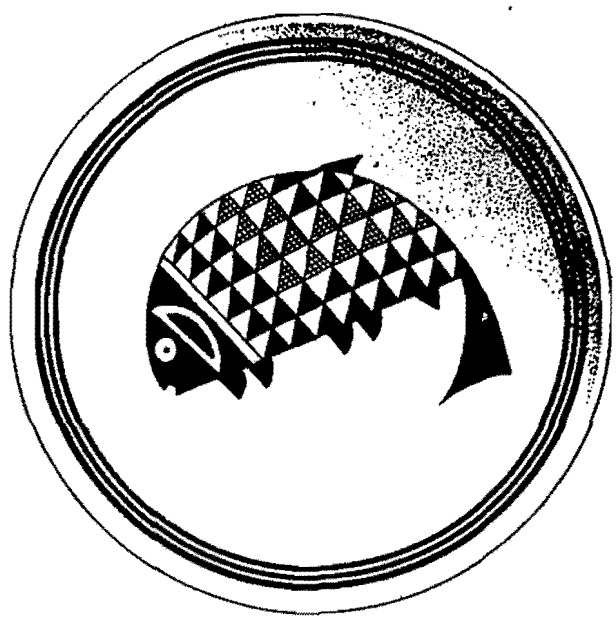
LA-UR-87-126

LOS ALAMOS ACCELERATOR CODE GROUP

DO NOT CIRCULATE

PERMANENT RETENTION

REFERENCE MANUAL FOR THE POISSON/SUPERFISH GROUP OF CODES



LOS ALAMOS NATIONAL LABORATORY
3 9338 00202 4684

Los Alamos
Los Alamos National Laboratory
Los Alamos, New Mexico 87545

POISSON/SUPERFISH REFERENCE MANUAL

Los Alamos Accelerator Code Group
MS H829
Los Alamos National Laboratory
Los Alamos, NM 87545
USA

Telephone: (505) 667–2839
FTS 843–2839

LOS ALAMOS NATL. LAB. LIBS.
3 9338 00202 4684

LA-UR-87-126

Date of Current Version: January 1, 1987

The present Accelerator Code Group Consists of:

John L. Warren (505-667-6677) Documentation
Mary T. Menzel (505-667-2841) POISSON
Grenfell Boicourt (505-667-1965) SUPERFISH
Helen Stokes (505-667-9131) Distribution
Richard K. Cooper (505-667-2839) Supervisor

If questions arise regarding these codes or the manual, please call the appropriate person or the general number 505-667-2839 and leave your name and the nature of your problem.

TABLE OF CONTENTS

A. GENERAL INTRODUCTION

1. Definition of the Problem
2. How to Use this Manual
3. How to Get Copies of the Codes and Manual
4. Short History of the Codes
5. Acknowledgements

B. MAGNET CODES — THE POISSON GROUP

1. Summary of the Basic Physics
   1. Maxwell's static equations
   2. Basic algorithms for finding the potential
   3. Calculation of auxiliary quantities
   4. Physical units used in Poisson group codes
   5. Trimming poletips with MIRT
2. Simple Example of Input and Output — H-shaped dipole
   1. Run on the Los Alamos CRAY computer
   2. Execution on Los Alamos computers
   3. Execution on systems outside of Los Alamos
3. Input to LATTICE through AUTOMESH
   1. Format-free input routine
   2. POISSON/PANDIRA inputs to LATTICE
   3. POISSON/PANDIRA input to AUTOMESH
4. Output from LATTICE
5. Input to POISSON or PANDIRA
   1. Introduction
   2. Dump numbers
   3. Changes in the CON array
   4. Entering permeability data
   5. Input of fixed potential points
   6. Input of current filaments
6. Output from POISSON or PANDIRA
7. Input and Output for TEKPLOT
   1. Input to TEKPLOT
   2. Output of TEKPLOT
   3. System-dependent plot routines in TEKPLOT

8. Input and Output for FORCE
9. Input and Output for MIRT
    1. Introduction
    2. Input to MIRT
    3. Output from MIRT
10. Diagnostic Messages
    1. Messages from AUTOMESH
    2. Messages from LATTICE
    3. Messages from POISSON
    4. Messages from PANDIRA
    5. Messages from TEKPLOT
11. Convergence and Accuracy
12. Examples
    1. POISSON example — quadrupole magnet
    2. POISSON example — electrostatic problem
    3. PANDIRA example — permanent magnet solenoid
13. Appendices
    1. Theory of electrostatics and magnetostatics
    2. Theory of auxiliary properties of magnets
    3. Harmonic analyisis of fields
    4. Use of conformal transformations
    5. Boundary conditions and meshes
    6. Numerical methods used in POISSON and PANDIRA
    7. Problem constants in numerical order
    8. Problem constants in alphabetic order
14. References
15. Index for Magnet Codes Section

C. RF CAVITY CODES — THE SUPERFISH GROUP

1. Summary of the Basic Physics
    1. TE- and TM-modes for rf structures
    2. Auxiliary properties of rf cavities
    3. Physical units
    4. Drive points and normalization
2. Simple Example of Input and Output — DTL Cell
3. Input to LATTICE through AUTOMESH
    1. Format-free input routine

2. SUPERFISH inputs to LATTICE
3. SUPERFISH inputs to AUTOMESH

4. Output from LATTICE
5. Input to SUPERFISH
6. Output from SUPERFISH
7. Input and Output for TEKPLOT
    1. Input for TEKPLOT
    2. Output of TEKPLOT
    3. System-dependent plot routines in TEKPLOT
8. Input and Output for SFO1 — Postprocesser for a DTL
9. PAN-T
    1. The Basic physics of PAN-T
    2. Preparation of input file
    3. Special input to PAN-T
10. Diagnostic Messages
    1. Messages from AUTOMESH
    2. Messages from LATTICE
    3. Messages from SUPERFISH
    4. Messages from SFO1
    5. Messages from TEKPLOT
11. Convergence and Accuracy
    1. Convergence
    2. Accuracy
12. Examples
    1. Spherical cavity
    2. Synchrotron cavity with ferrites
    3. Electron linac cavity
13. Appendices
    1. Theory of rf cavities
    2. Theory of auxiliary properties of rf cavities
    3. Boundary conditions and meshes
    4. Numerical methods used in SUPERFISH
    5. Problem constants in numerical order
    6. Problem constants in alphabetic order
14. References
15. Index for RF Cavity Codes Section

Part A

General Information

Chapter A.1

Definition of the Problem

The POISSON/SUPERFISH Group codes were set up to solve two separate problems: the design of magnets and the design of rf cavities in a two-dimensional geometry. The first stage of either problem is to describe the layout of the magnet or cavity in a way that can be used as input to solve the generalized Poisson equation for magnets or the Helmholtz equation for cavities. The computer codes require that the problems be discretized by replacing the differentials (dx, dy) by finite differences $(\delta X, \delta Y)$. Instead of defining the function everywhere in a plane, the function is defined only at a finite number of points on a mesh in the plane.

For example, consider the cross section of a long H-shaped dipole magnet as shown in Fig. A.1.1. A uniform triangular mesh of the type shown in Fig. A.1.2 is used to discretize the problem. Starting from a uniform triangular mesh, the code "relaxes" the mesh until the sides of the triangles match the boundaries and interfaces between different physical materials as closely as possible. Regions containing different materials, such as copper, iron, and vacuum or air, have to be identified and the material properties specified.

The code that generates the triangular mesh is called LATTICE. For many problems the preparation of input data for LATTICE is a tedious task, particularly for curved boundaries between different regions. A code called AUTOMESH has been written to make the preparation of input to LATTICE simpler and more physically meaningful.

The next step is to use the mesh and physical properties to find the vector potential $\mathbf{A}(x, y)$ at all mesh points. This is done in the code called POISSON. This code does not solve Poisson's equation directly, but rather works with a discretized form of Ampere's law, to obtain successive approximations to the potential. After several iterations, the code finds a solution that satisfies the boundary conditions and Eq. (A.1.1) over the entire mesh.

$$\oint_c \mathbf{H} \cdot d\mathbf{l} = \int_A \mathbf{J} \cdot d\mathbf{a} \tag{A.1.1}$$

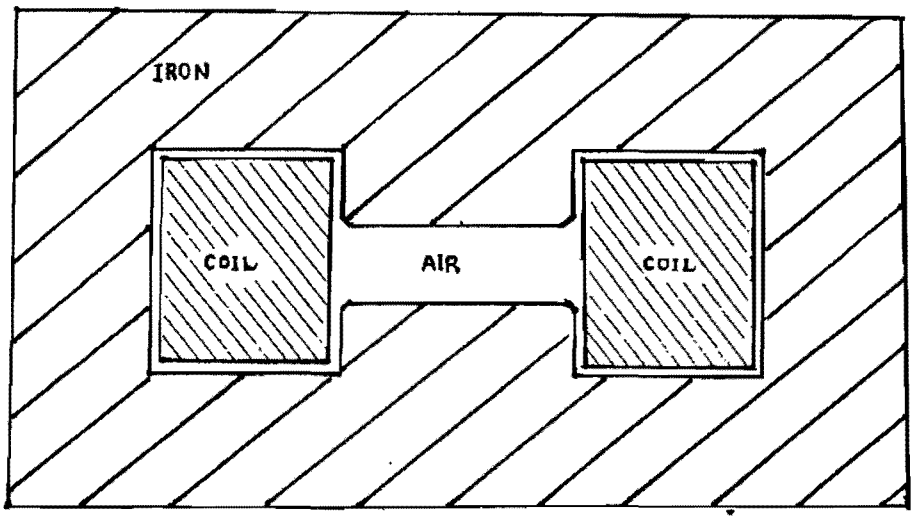


Figure A.1.1: Cross section of an H-shaped dipole magnet. The boundaries of the various regions are entered into LATTICE, which generates the mesh of discrete points shown in the next figure.

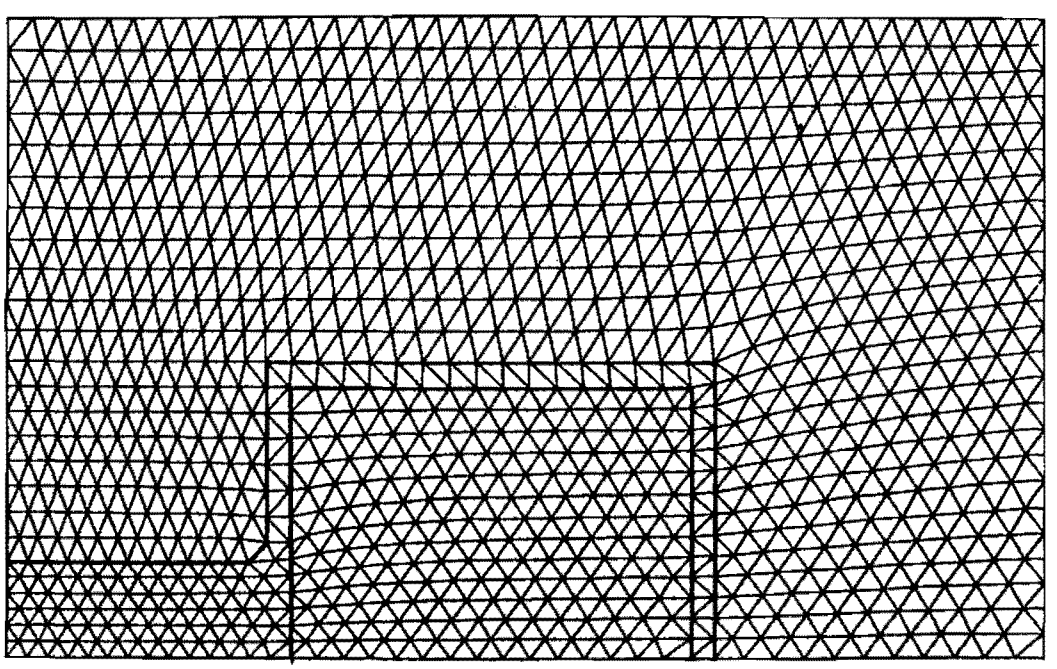


Figure A.1.2: The "physical" mesh generated by LATTICE for one-quarter of the H-shaped dipole magnet shown in Fig. A.1.1.

There are actually two codes (POISSON and PANDIRA) that can be used to find magnetic fields. POISSON finds the magnetic field caused by electric currents and soft iron. PANDIRA is primarily used to solve problems involving permanent magnet materials, although it can be used for POISSON-type problems. The numerical methods used in PANDIRA are different than those in POISSON, as is explained in Sec. B.13.6.

It should be noted that POISSON and PANDIRA can also be used to solve electrostatic problems. Currents are replaced by charges, permanent magnets are replaced by electrets, etc. It will be shown later that the parallelism between magnetostatics and electrostatics is nearly complete.

SUPERFISH solves an eigenvalue problem to determine the resonant frequencies and standing-wave electromagnetic fields in radio frequency (rf) cavities commonly used in charged particle accelerators. It can also be used to calculate properties of two-dimensional cross sections of waveguides or cylindrically symmetric waveguides. There are four other codes in the POISSON/SUPERFISH group that are postprocessors. The functions of all the codes are summarized in TABLE A.1.I.

TABLE A.1.I. A List of the POISSON/SUPERFISH Group Programs

1 AUTOMESH – prepares the input for LATTICE from geometrical data describing the problem, that is, it constructs the "logical" mesh and generates (x, y) coordinate data for straight lines, arcs of circles, and segments of hyperbolas.

2 LATTICE – generates an irregular triangular mesh (physical mesh) from input data for the "logical" and physical coordinates describing the problem, calculates "point current" terms at each mesh point in regions with distributed current density, and sets up mesh point relaxation order. LATTICE writes the information needed to solve the problem on a binary file that is read by the equation-solving codes POISSON, PANDIRA or SUPERFISH.

3 POISSON – solves Maxwell's magnetostatic (electrostatic) equations for the vector (scalar) potential with nonlinear, isotropic iron (dielectric) and electric current (charge) distributions for two-dimensional cartesian or three-dimensional cylindrical symmetry. It calculates the derivatives of the potential, namely, the fields and their gradients, calculates the stored energy, and performs harmonic (multipole) analysis of the potential. The code uses a successive over-relaxation algorithm and an iterative scheme that steps successively through the mesh points.

4 PANDIRA – is similar to POISSON, except it allows anisotropic and permanent magnet materials, for which the B vs. H (D vs. E)curve exists in the second quadrant. PANDIRA uses a different numerical method to obtain the potential.

5 TEKPLOT – plots physical meshes generated by LATTICE, and equipotential or field lines from the output of POISSON, PANDIRA, MIRT, or SUPERFISH.

6 FORCE – calculates forces and torques on coils and iron regions from POISSON or PANDIRA solutions for the potential (Presently not available).

7 MIRT – optimizes magnet profiles, coil shapes, and current densities starting from the output of POISSON, based on a field specification defined by the user.

8 SUPERFISH – solves for the TM and TE resonant frequencies and field distributions in an rf cavity with two-dimensional cartesian or three-dimensional cylindrical symmetry. Only the azimuthally symmetric modes are found for cylindrically symmetric cavities. The modes are found one at a time.

9 SFO1 – calculates auxiliary quantities from the output of SUPERFISH. These quantities include stored energy, power dissipation on the walls and tube stems, transit time factors, shunt resistance, the quality factor Q, and the maximum electric field on the boundary. The program also calculates the frequency shift of the resonant frequency caused by small displacements of segments of the cavity boundary. This code can serve as a model for creating additional SUPERFISH output codes.

Most users are interested in designing either magnets or rf cavities. For that reason we have divided this manual into three logically separate parts as illustrated in Fig. A.1.3. The codes are discussed in the order that they are normally used. The remainder of this section provides general information describing access to the codes, suggestions for using this manual, and a short history of the code development.

General Introduction

Magnet Design Programs	RF Cavity Design Programs
AUTOMESH	AUTOMESH
LATTICE	LATTICE
POISSON, PANDIRA	SUPERFISH
TEKPLOT	TEKPLOT
FORCE	SFO1
MIRT	

Figure A.1.3: General Layout of the POISSON/SUPERFISH Manual.

Chapter A.2

How to use this Manual

The POISSON/SUPERFISH codes are fairly complicated and will require some effort on the user's part before they can be mastered. This manual is intended to do three things, namely, to give the beginner a quick introduction, to supply useful summaries of input procedures for persons familiar with the code, and to give an in-depth summary of the theory that went into the writing of the codes.

The manual has been divided into two major sections, one for magnet problems and one for rf-cavity problems. The codes AUTOMESH, LATTICE and TEKPLOT are common to both problems. For the convenience of the reader, the magnet section and the rf cavity section each have their own description of these three codes. These two major sections of the manual can be physically separated without destroying the continuity of each section.

To help the beginner, both the magnet section and the rf cavity section begin with two "primer" chapters that go through the basic physics contained in the codes and display the input and output for a simple example. More examples are contained in a later chapter. Using these examples the beginner should be able to master the mechanics of running the codes.

These examples do not exercise all the options available in the codes. The details of these options are contained in later chapters. We have tried to summarize the input requirements and definitions of important input quantities in tables so that persons familiar with the mechanics of running the codes can easily remind themselves of the available options. We have also included a separate chapter on diagnostics and suggestions of what to do if these diagnostic messages are encountered.

The remainder of the material in the sections will be of some use in gaining an in-depth understanding of the theory behind the codes, the numerical methods used in the codes, and the basic limitations of the codes. No design tool should be a "black box" to the user.

Chapter A.3

How to Access the Codes.

There are several versions of the codes at various locations around the world. As a service to the user community, Los Alamos Group AT-6 has undertaken the maintenance and distribution of a "standard" version of these codes. The standard version has been installed on the Los Alamos CRAY/CTSS and VAX systems. The source codes are written in standard FORTRAN 77. The CRAY version and the VAX version differ by only a few lines that are readily identifiable from the code listings.

Copies of the source codes are available to users outside of Los Alamos by means of magnetic tape or transmission over the DECNET, ARPANET or BITNET computer networks.

A.3.1 Access outside of Los Alamos.

AT-6 will provide copies of the complete set of POISSON/SUPERFISH group codes plus sample input and output for a magnet and an rf cavity problem to any individual or institution outside of Los Alamos, provided the requestor furnishes a magnetic tape and the name of the computer on which the codes will be installed. The general characteristics of the tapes are:

For VAX/VMS: (UTILITY = COPY/LOG)
9-track, 1600 b.p.i., 80 characters/line,
512 lines/block, labeled ASCII tape
For CDC7600-CRAY or IBM (UTILITY = TAPECOPY)
9-track, 1600 b.p.i., 80 characters/line, 30 lines/block,
unlabeled ASCII (EBCDIC for IBM) tape.

Our mailing address is

Group AT-6, MS H829
Los Alamos National Laboratory
Los Alamos, NM 87545.

Once a requestor has received a copy of the codes from us, he will be informed by newsletter when corrections and improvements are made in the codes. Should difficulties arise in running of the codes, assistance is available by calling any of the numbers on page ii.

A.3.2 Access at Los Alamos.

At present the codes for the CRAY are on the Common File System (CFS); the VAX versions of the codes are stored on the CFS and on the MP-VAX complex. The following is a description of how to access the codes on these two systems. AT-6 does not maintain the CDC7600 version of the codes, but they do exist.

A.3.2.1 MP-VAX Version.

The directory AT00$DISK:[AT6HKS.VAXFILES] is the location of the source and executable files. TABLE A.3.2.I gives the names of the files.

TABLE A.3.2.I. VAX Files for the POISSON/SUPERFISH Codes

Source	Executable
AUTO.FOR	AUTOMESH.EXE
FISSO.FOR	SUPERFISH.EXE
FORSO.FOR	FORCE.EXE
LATSO.FOR	LATTICE.EXE
	MIRLIB.OLB
MIRSO.FOR	MIRT.EXE
PANSO.FOR	PANDIRA.EXE
POISO.FOR	POISSON.EXE
LIBSO.FOR	POILIB.OLB
SF1SO.FOR	SFO1.EXE
TEKSO.FOR	TEKPLOT.EXE

To run any of the executable files on the MP-VAX complex, e.g., AUTOMESH, just type:

```
RUN ATOO$DISK:[AT6HKS.VAXFILES]AUTOMESH
```

Henceforth underline means "typed by the user." The code generates an output file called TAPE73.DAT, which is used as input to LATTICE.

AUTSO.FOR is the only source file independent of other files. To compile and link it, type:

```
FORT AUTSO
LINK/EXEC=AUTOMESH AUTSO
```

The POILIB/LIB file must be linked to all other POISSON or SUPERFISH group codes. For this reason, the library must be created first if any program besides AUTOMESH is to be recompiled and if the file POILIB.OLB does not exist. To recreate the library, type:

```
FORT LIBSO
LIBR/CREATE POILIB LIBSO
```

There is an additional library used with the code MIRT. To recreate this library and run MIRT, type:

```
FORT MIRSO
LIBR/CREATE MIRLIB POISO,LIBSO
LINK/EXEC=MIRT MIRSO,MIRLIB
```

To compile and link any of the other codes, PANDIRA for example, type:

```
FORT PANSO
LINK/EXEC=PANDIRA PANSO,POILIB/LIB
```

Table A.3.2.II summarizes the commands necessary to recompile all the codes.

Table A.3.2.II. Commands for Recompilation of VAX Source Files

FORT LIB/CREATE	LIBSO POILIB LIBSO	for POILIB
FORT LINK/EXEC=AUTOMESH	AUTSO AUTSO	for AUTOMESH
FORT LINK/EXEC=LATTICE	LATSO LATSO, POILIB/LIB	for LATTICE
FORT LINK/EXEC=PANDIRA	PANSO PANSO, POILIB/LIB	for PANDIRA
FORT LINK/EXEC=POISSON	POISO POISO, POILIB/LIB	for POISSON
LIBR/CREATE	MIRLIB POISO, LIBSO	for MIRLIB
FORT LINK/EXEC=MIRT	MIRSO MIRSO, MIRLIB/LIB	for MIRT
FORT LINK/EXEC=FORCE	FORSO FORSO, POILIB/LIB	for FORCE
FORT LINK/EXEC=SFO1	SF1SO SF1SO, POILIB/LIB	for SFO1
FORT LINK/EXEC=SUPERFISH	FISSO FISSO, POILIB/LIB	for SUPERFISH
FORT LINK/EXEC=TEKPLOT	TEKSO TEKSO, POILIB/LIB, USER$OLB:PLOT10/LIB	for TEKPLOT

(This assumes access to PLOT 10.)

A.3.2.2 The CRAY system.

The codes are available as source files and executable files on the CFS and are listed in Table A.3.2.III. Executable files are for CRAY1 — CRAY1S machines. The user must recompile source files if running on CRAY XMP's.

Table A.3.2.III. CRAY Files for the POISSON/SUPERFISH Codes

Source Files Directory /lacc/poicodes/cray/src	Executable Files Directory /lacc/poicodes/cray/xeq
AUTSO	AUTOMESH
LATSO	LATTICE
FISSO	FISH
FORSO[a]	FORCE[a]
LIBSO	
MIRSO	MIRT
	MIRLIB[b]
PANSO	PANDIRA
POISO	POISSON
SF1SO	SFO1
TEKSO	TEKPLOT
	POILIB[c]

[a]At the present time, FORSO is not on the CFS.
[b]MIRLIB is the binary version of POISO. It is needed for MIRT only
[c]POILIB is the binary version of LIBSO.

There also exists a directory called **/lacc/poicodes/cray/xmp**, which contains examples of input and output files for several magnet and rf-cavity problems.

To obtain an executable program, e.g., AUTOMESH, type:

```
mass get dir=/lacc/poicodes/cray/xeq automesh
```

To execute the program, type:

```
automesh
```

The program will ask for the name of an input file that the user has created. Creation of the input file is described in Sections B.3 and C.3 below. To obtain a source program, e.g., AUTSO, type:

```
/lacc/poicodes/cray/src autso
```

The first lines of the source files are the compile instructions needed for the XEQ utility. Compilation can be done with the single command:

```
xeq autso
```

AUTOMESH is a self-contained program. All other programs use common routines from file LIBSO (the source file) or POILIB (the binary file). To compile one of these programs, e.g., LATTICE, type either

```
/lacc/poicodes/cray/src latso poilib
xeq latso
```

or

```
/lacc/poicodes/cray/src latso libso
xeq libso
xeq latso
```

The program MIRT uses an additional common file – POISO (the source file) or MIRLIB (the binary file). To compile MIRT, type either

```
/lacc/poicodes/cray/src mirso poilib mirlib
xeq mirso
```

or

```
/lacc/poicodes/cray/src mirso poiso libso
xeq libso
xeq poiso
xeq mirso
```

Chapter A.4

History of the POISSON/SUPERFISH Codes.

The POISSON/SUPERFISH group of codes really consists of two sets of codes, one for the design of magnets and another set for the design of rf cavities. These codes have been developed over a period of 15 years. In the late sixties, Jim Spoerl at Lawrence Berkeley Laboratory (LBL) began modifying a diffusion calculation code written by Alan Winslow at Lawrence Livermore National Laboratory (LLNL). The result was the TRIM codes (MESH and FIELD) capable of solving mathematical models of two-dimensional magnets, including the effects of finite permeability. MESH constructed an irregular triangular mesh to fit the geometry of the magnet. FIELD solved Poisson's equation for the potential function over the mesh.

Ron Holsinger, Klaus Halbach and other associates at LBL found the codes useful but in need of improvements. They made major changes in MESH and introduced the use of conformal transformations. In view of the extensive changes, they decided to rename the codes LATTICE, TEKPLOT, and POISSON. LATTICE is like MESH; TEKPLOT, which was split from MESH, draws plots of either the mesh or the field lines; and POISSON is like FIELD. Holsinger continued to develop these codes for two years while he was at the Swiss Institute for Nuclear Research (SIN) and the European Center for Nuclear Research (CERN). Another version of the magnet codes (LATTCR, POICR, TRIPCR and FORCCR) were created by C. Iselin while Holsinger was at CERN.

By 1975, when Holsinger arrived at Los Alamos, he had completed five programs: LATTICE, POISSON, TEKPLOT, MIRT, and FORCE. MIRT was an optimization program that iteratively changed the shape of pole faces and current distributions to obtain the field distribution specified by the user. FORCE was created to calculate the magnetic forces and torques on the iron and current-carrying coils of the magnet.

While at Los Alamos, Holsinger, in collaboration with Halbach, wrote three more programs: AUTOMESH, PANDIRA, and SUPERFISH. For many problems the in-

put data preparation for LATTICE is very tedious because one must supply both physical and logical coordinates along the boundaries. AUTOMESH eliminates the need for the user to define the logical coordinates and most of the physical coordinates. PANDIRA was written in response to difficulties encountered in using POISSON to solve problems involving permanent magnets for which the algorithm used in POISSON (successive point over-relaxation) diverges badly. PANDIRA uses the so-called "direct method" that works well for linear problems. Only a few iterations are required to make the reluctivity self-consistent with the solution for the potential.

At the urging of Don Swenson and Klaus Halbach, Holsinger wrote the rf cavity code SUPERFISH using the techniques developed for the magnet codes. SUPERFISH has many features in common with a program called MESSYMESH, which was created at the Midwest Universities Research Association (MURA) during the early sixties.

SUPERFISH not only calculates the fields but also determines the eigenfrequencies of the cavity. To solve rf cavity design problems, one uses AUTOMESH and LATTICE to describe the geometry, SUPERFISH to find the field and frequency, and TEKPLOT to display the fields in the cavity. One additional program called SFO1 has been written to calculate auxiliary quantities from the output of SUPERFISH. These quantities include transit time factors, power losses on the walls, and frequency shifts caused by perturbation of the cavity walls. SFO1 was written for Drift Tube Linac (DTL) design. Other SUPERFISH post processor codes have been written for special purposes but are not included in the code group at present.

Originally the Los Alamos version of the codes were written for the CDC6600 computers. In 1977, when Holsinger left Los Alamos, he converted all the codes to run on the VAX/750's. He continued to update and maintain the programs until 1982. At that time, he transferred the maintenance and distribution responsibility to Los Alamos.

The codes have had tremendous popularity since the early seventies, and this has resulted in a proliferation of versions of the codes. The documentation for these codes was adequate but incomplete. Until recently, Los Alamos has had very limited resources for documentation, maintenance, distribution, and consultation with users. In October of 1983, The Department of Energy (DOE-HENP) provided financial support with which we have been able to undertake the writing of a comprehensive manual and the standardization of codes. With continuing DOE support we have completed this manual, established users' groups to guide improvements of the codes, and set up a system for distributing updated versions of the codes and documentation.

Chapter A.5

Acknowledgements

We would like to acknowledge support for writing this manual from DOE-NP and DOE-HEP. The codes were in large measure written by Ronald Holsinger with the theoretical assistance of Klaus Halbach. We would also like to thank the users who, over the years, have brought coding bugs to our attention.

Holsinger's User's Guide was a masterpiece of concise exposition. What you needed to know was all there. It provided us with a firm base on which to build. Most of the text was written by Gren Boicourt, Mary Menzel and John Warren. Bernie Tice from SLAC wrote the sections on Numerical Methods. Martyn Foss assisted us in understanding the permeability tables.

The typing of this manual was not a trivial chore since it was first done on a WANG processor by Millie Steinkamp and later converted to LaTeX on the VAX by Gary Benson, Jeff Storm, Kim Serna, Patrick Byrnes and Peggy Atencio. The use of LaTeX was a real learning experience for us.

Finally, we would like to thank the people who read earlier versions of this manual and provided constructive input on content and organization.

Part B

MAGNET CODES – THE POISSON GROUP

Chapter B.1

Summary of the Basic Physics

Mathematically, the Poisson group of computer codes solves Maxwell's static equations (MSE's) in integral form and in two dimensions. When the MSE's are taken together with the boundary conditions, they are equivalent to a generalized form of Poisson's equation in two dimensions.

The first two chapters of Part B are a primer for the Poisson Group Codes. They contain a summary of the basic equations and the run sequence for a simple magnet. If you follow the steps for finding the field for this simple magnet, you will know how to run the codes for other cases. The codes have a large number of options which are explained in Chapters B.3 through B.9. Chapter B.12 contains three examples which illustrate some of these options. The rest of the chapters contain reference material that you will find useful if you run into problems or if you want to understand what the codes are doing in more detail.

B.1.1 Maxwell's static equations.

Maxwell's static equations (MSE) are derived from Maxwell's equations under the assumption that all fields are independent of time. They can be divided into three types:

Ampere's Law Type Equations:

$$\oint \mathbf{H} \cdot d\mathbf{l} = \int \mathbf{J} \cdot d\mathbf{a} \rightarrow \nabla \times \mathbf{H} = \mathbf{J}, \tag{B.1.1.1}$$

$$\oint \mathbf{E} \cdot d\mathbf{l} = 0 \rightarrow \nabla \times \mathbf{E} = 0, \tag{B.1.1.2}$$

Gauss's Law Type Equations:

$$\oint \mathbf{B} \cdot d\mathbf{a} = 0 \rightarrow \nabla \cdot \mathbf{B} = 0, \tag{B.1.1.3}$$

$$\oint \mathbf{D} \cdot d\mathbf{a} = \int \rho dv \rightarrow \nabla \cdot \mathbf{D} = \rho, \tag{B.1.1.4}$$

Material Type Equations:

Isotropic materials

$$\mathbf{H} = \gamma(|\,\mathbf{B}\,|)\mathbf{B}/\mu_o, \tag{B.1.1.5}$$

$$\mathbf{D} = \epsilon_o \kappa_e(|\,\mathbf{E}\,|)\mathbf{E}, \tag{B.1.1.6}$$

Anisotropic materials

$$\mathbf{H} = \overleftrightarrow{\gamma} \cdot \mathbf{B}/\mu_o, \tag{B.1.1.7}$$

$$\mathbf{D} = \epsilon_o \overleftrightarrow{\kappa}_e \cdot \mathbf{E}, \tag{B.1.1.8}$$

Anisotropic, permanent magnet (electret) material

$$\mathbf{H} = \overleftrightarrow{\gamma} \cdot \mathbf{B}/\mu_o - \mathbf{H}_c, \tag{B.1.1.9}$$

$$\mathbf{D} = \epsilon_o \overleftrightarrow{\kappa} \cdot \mathbf{E} - \mathbf{D}_c, \tag{B.1.1.10}$$

where $\mathbf{H}$ and $\mathbf{E}$ are the magnetic and electric field, $\mathbf{B}$ and $\mathbf{D}$ are the magnetic induction and the displacement fields, $\mathbf{J}$ and ρ are the electric current and charge densities, and γ is called the reluctivity. (It is the reciprocal of the relative permeability κ_m.) The quantity κ_e is the dielectric constant. For anisotropic materials both $\overleftrightarrow{\gamma}$ and $\overleftrightarrow{\kappa}_e$ are tensors, i.e., the magnetic field and magnetic induction are not in the same direction, for example. For permanent magnet (electret) materials there are magnetic and displacement fields that remain even when $\mathbf{B}$ and $\mathbf{E}$ are zero. Historically these are the coercive forces, $\mathbf{H}_c$ and $\mathbf{D}_c$. In the Ampere's law formulas, $d\mathbf{a}$ is an element of area times a unit vector perpendicular to that area, and $d\mathbf{l}$ is an element of path length times a unit vector tangent to the closed contour surrounding the area. In the Gauss's law formula, dv is the volume enclosed by the closed surface of the integral on the left side of the equation. The constants μ_o and ϵ_o depend on the system of physical units (meters, kilograms, seconds, Coulombs, for example) used to measure the field quantities.

One goes from the MSE's to the generalized Poisson equation by assuming that

$$\mathbf{B} = \nabla \times \mathbf{A}, \tag{B.1.1.11}$$

which follows from Eq. (B.1.1.3). For two-dimensional, cartesian geometry one can show that

$$\mathbf{A} = A_z\,\hat{\mathbf{e}}_z, \tag{B.1.1.12}$$

and

$$\mathbf{J} = J_z\,\hat{\mathbf{e}}_z, \tag{B.1.1.13}$$

where $\hat{\mathbf{e}}_z$ is a unit vector in the z-direction. From this and Eq. (B.1.1.1) it follows that

$$\frac{\partial}{\partial x}\left[\gamma(\mid \mathbf{B}(x,y) \mid)\frac{\partial}{\partial x}A_z(x,y)\right] + \frac{\partial}{\partial y}\left[\gamma(\mid \mathbf{B}(x,y) \mid)\frac{\partial}{\partial y}A_z(x,y)\right] = \mu_o J_z(x,y), \tag{B.1.1.14}$$

which is the generalized Poisson equation in cartesian coordinates.

For magnet configurations having cylindrical symmetry one can use cylindrical coordinates (r, θ, z). It is shown in Chapter B.13 that when

$$\mathbf{A} = A_\theta\,\hat{\mathbf{e}}_\theta \tag{B.1.1.15}$$

and

$$\mathbf{J} = J_\theta\,\hat{\mathbf{e}}_\theta \tag{B.1.1.16}$$

then

$$\frac{\partial}{\partial r}\left[\gamma(\mid \mathbf{B} \mid)\frac{1}{r}\frac{\partial}{\partial r}(rA_\theta)\right] + \frac{\partial}{\partial z}\left[\gamma(\mid \mathbf{B} \mid)\frac{\partial}{\partial z}A_\theta\right] = \mu_o J_\theta \tag{B.1.1.17}$$

in cylindrical coordinates. The codes find the vector potential in either cartesian coordinates or cylindrical coordinates. The generalized Poisson equations for anisotropic and permanent magnet materials are more complicated and can be found in Chapter B.13.

Problems involving anisotropic material can only be solved using the code PANDIRA. POISSON treats only isotropic materials. PANDIRA will treat both isotropic and anisotropic materials. Anisotropy is described in terms of the reluctivities $\gamma_{\|}$ along an easy axis and $\gamma_{\perp}$ along a hard axis. PANDIRA allows two geometries for the easy axis. In the first geometry, the easy axis is independent of position in the material. In the second geometry, the direction of the easy axis changes along the circumference of a circle whose center need not coincide with the origin of coordinates. There are no natural anisotropic materials of this type, but one can artificially approximate such materials by assembling a number of wedge-shaped permanent magnets. We will only describe the first geometry here; the second geometry is described in Section B.13.1.

Figure B.1.1.1 illustrates the direction of the easy axis relative to the axes of the larger problem. The easy axis makes an angle φ_E with respect to the horizontal axis of the coordinate system. The reluctivity tensor $\overset{\Rightarrow}{\gamma}$ is a symmetric tensor whose cartesian components are:

$$\gamma_{xx} = \gamma_{\|} cos^2\varphi_E + \gamma_{\perp} sin^2\varphi_E, \tag{B.1.1.18}$$

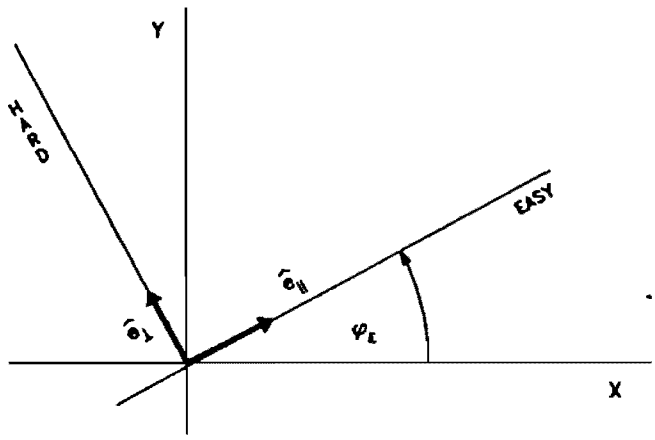


Figure B.1.1.1: Definition of coordinate axes for anisotropic materials.

$$\gamma_{xy} = \frac{1}{2}(\gamma_{\parallel} - \gamma_{\perp}) sin 2\varphi_E, \tag{B.1.1.19}$$

$$\gamma_{yy} = \gamma_{\parallel} sin^2\varphi_E + \gamma_{\perp} cos^2\varphi_E. \tag{B.1.1.20}$$

When the anisotropic material is also permanent magnet material, the coercive force H_c is also parallel to the easy axis; its components are

$$H_{cx} = H_c cos\varphi_E, \tag{B.1.1.21}$$

and

$$H_{cy} = H_c sin\varphi_E. \tag{B.1.1.22}$$

Because of the parallelism between the equations for the electric and magnetic fields, it is easy to see how one equation-solver can be used for both magnetostatic and electrostatic problems. If we let the electric field be given as the gradient of the scalar potential V,

$$\mathbf{E} = -\nabla V, \tag{B.1.1.23}$$

then it can be shown that the corresponding generalized Poisson equation for electrostatics in cartesian coordinates is

$$\frac{\partial}{\partial x}\left[\kappa_e(|\mathbf{E}|)\frac{\partial}{\partial x}V\right] + \frac{\partial}{\partial y}\left[\kappa_e(|\mathbf{E}|)\frac{\partial}{\partial y}V\right] = \rho/\epsilon_0, \tag{B.1.1.24}$$

which is of the same form as the magnetostatic equation, Eq. (B.1.1.14). The electrostatic equation in cylindrical coordinates is

$$\frac{1}{r}\frac{\partial}{\partial r}\left[r\kappa_e(|\mathbf{E}|)\frac{\partial}{\partial r}V\right] + \frac{\partial}{\partial z}\left[\kappa_e(|\mathbf{E}|)\frac{\partial}{\partial z}V\right] = \rho/\epsilon_0, \tag{B.1.1.25}$$

which is slightly different than the corresponding magnetostatic equation, Eq. (B.1.1.17). The codes do distinguish between magnetostatic and electrostatic problems for cylindrical coordinates.

Boundary conditions must be supplied to give a unique solution to the equations. Poisson's equation is an elliptic partial differential equation. This means that the most general allowed boundary conditions take the form

$$aA_b + b\partial A_b/\partial n = c \tag{B.1.1.26}$$

where A_b is for instance the value of the z-component of the vector potential evaluated on the boundary and $\partial A_b/\partial n$ is the derivative of the potential in a direction normal to the boundary curve, evaluated on the boundary. The quantities a, b and c are known functions evaluated on the boundary. The Poisson group codes do not allow this type of generality. On any segment of the boundary, the quantities a and b are either zero or one. The quantity c is always zero when a is zero. That is, only two types of boundary segments are allowed. These are referred to as Dirichlet boundaries and Neumann boundaries, which are defined as follows:

Dirichlet

$$A_b = c \tag{B.1.1.27}$$

Neumann

$$\partial A_b/\partial n = 0. \tag{B.1.1.28}$$

An easy mnemonic is to remember that "normal" derivative is "Neumann." The only known way to assign the proper boundary conditions for a problem is by "physical intuition." You must have some qualitative idea how the field is going to behave at the boundary. Because of the lack of generality built into the codes, there are some magnetic field problems that cannot be solved with these codes. Many problems can be solved only approximately.

B.1.2 Basic Algorithms for finding the potential.

The code POISSON sets up a linear equation for the potential at each point on a topologically regular, triangular mesh. For a picture of such a mesh, see Fig. B.2.7 below. By topologically regular we mean that the mesh points can be put into one-to-one correspondence with points on a mesh generated by equilateral triangles, which we will call the logical mesh since the points in this mesh can be numbered in a "logical" manner. The topologically regular mesh, which is a distortion of the logical mesh, is called the physical mesh. Each point on the logical mesh has six equidistant nearest neighbors. See Fig. B.1.2.1. The correspondence between the physical mesh and the logical mesh allows one to identify the corresponding six neighbors on the physical mesh. If we call the potential at the center of the hexagon A_0, and call the potentials at the nearest neighbor mesh points A_i, for $i = 1$ to 6,

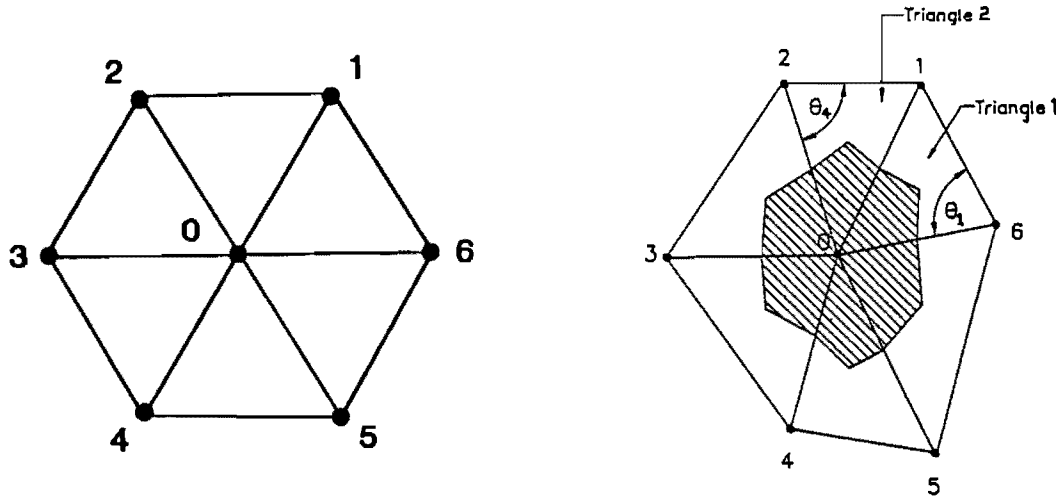


Figure B.1.2.1: Correspondence between points on the logical mesh and the physical mesh for the six nearest neighbors. The contour shown in the physical mesh is the contour used in Ampere's law, Eq. (B.1.1.1).

then Ampere's Law, when discretized, can be shown, see Appendix B.13.6, to give the equality

$$A_0 = \frac{\sum_{i=1}^{6} w_i A_i + \frac{\mu_0}{3} \sum_{i=1}^{6} J_i a_i}{\sum_{i=1}^{6} w_i} \tag{B.1.2.1}$$

where a_i is the area for each triangle enclosed by the contour integral and the w_i's are linear functions of the reluctivities γ_i's. For instance it can be shown that

$$w_1 = \frac{1}{2}\left[\gamma_1(|\,\mathbf{B}\,|)\cot(\theta_1) + \gamma_2(|\,\mathbf{B}\,|)\cot(\theta_4)\right] \tag{B.1.2.2}$$

where the angles θ_1 and θ_4 are shown in Fig. B.1.2.1. Such an equation can be set up at each mesh point. The result is a set of "linear" equations with the values of the potential at each mesh point being the unknowns. This set of equations is solved by a numerical procedure called successive substitution with overrelaxation. This procedure is described in Section B.13.6. These equations are nonlinear, since the reluctivities γ_i are functions of the A_i. The γ_i's are allowed to change during the solution process so that the final solution is self-consistent.

Having obtained a solution for the potential over the mesh, the codes will calculate auxiliary quantities such as the magnetic induction **B** and its derivatives. These calculations are discussed in the next subsection.

B.1.3 Calculation of auxiliary quantities.

In principle one could obtain the fields by numerical differentiation of the potentials, but this would not be very accurate. Derivatives of the fields, for example

$\partial B_x/\partial y$, would be even more inaccurate. POISSON gets around this limitation by analytically taking the derivatives of the potential expressed as a series of the form

$$\sum_{n=o}^{\infty}(a_n u_n + b_n v_n) \tag{B.1.3.1}$$

where the quantities u_n and v_n are polynomial solutions of Laplace's equation in either cartesian or cylindrical coordinates. The subscript n is the order of the polynomial. Table B.1.3.I gives a short list of these polynomials for cartesian coordinates.

Table B.1.3.I Harmonic Polynomials

n	u_n	v_n
1	x	y
2	$x^2 - y^2$	$2xy$
3	$x^3 - 3xy^2$	$3x^2y - y^3$

Magnet designers will recognize these as dipole, quadrupole, and sextupole magnetic potential functions. The coefficients in the multipole expansion defined by Eq. (B.1.3.1) are determined by the symmetry of the problem and by a least-square fitting procedure using the first 18 neighboring points to a given point. The details are described in Section B.13.2. The derivatives of the potential are easily expressed in terms of the a_n's, b_n's and the harmonic polynomials.

In addition to calculating the field and its derivatives, POISSON also calculates the energy in the field. The program FORCE is the postprocessor to POISSON that calculates the forces and torques on current-carrying coils and blocks of magnetic (iron) materials. Once again, the procedures used to calculate these quantities are described in Section B.13.2.

B.1.4 Physical units used in the Poisson group codes.

Maxwell's static equations Eqs. (B.1.1) through (B.1.10) have been written in the form suggestive of rationalized MKS units, but the units really depend on the values of ϵ_o and μ_o. The codes, on the other hand, do assume certain default units. These units are given in Table B.1.4.I.

Table B.1.4.I Default System of Units

Quantity	Units
Length	centimeters
Current	amperes
Induction B	gauss
Field H	amperes/cm
Potential A	gauss-cm
Derivatives of B	gauss/cm
Force	amp-cm-gauss = 10^{-6}newtons
Stored energy (cartesian)	joules/meter
Stored energy (cylindrical)	joules
μ_o	0.4 π gauss-cm/amp*
Potential V	volts
Field E	volts/cm
Velocity of light c	2.997925 x 10^{10} cm/sec
Charge	coulombs

*$\mu_o = 4\pi$ x 10^{-9} volts/(amp-cm) is also used in the codes. One can calculate $\epsilon_o = 1/(\mu_o c^2) = 8.8542$ x 10^{-14} coul /(volt-cm).

As the reader can see, this is a modified rationalized MKS system; meters are replaced by centimeters, and Tesla are replaced by gauss. The user can change the units to some extent by making use of the conversion parameter in the input to AUTOMESH or LATTICE. These codes will accept length input in any units the user desires (inches, feet, meters, furlongs, ...). This conversion parameter is known as CON(9). For example, if CON(9) = 2.54, then LATTICE expects input in inches; if CON(9) = 100, LATTICE expects meters.

B.1.5 Trimming the poletips with MIRT.

The POISSON postprocessor called MIRT is an optimization program that allows the user to trim the field in a dipole magnet. It can also be used to change the shapes of current-carrying coils, and to change the value of currents to meet user-defined field specifications. We will only describe the poletip modification capability here. The basic procedure is to: 1. Run POISSON with a preliminary estimate of the current in the coil and a simple flat poletip; 2. Specify the desired field values at points in the air gap of the magnet; 3. Specify points on the poletip boundary that can be moved to reshape the poletip; and then 4. Let MIRT change the magnitude of current in coils and the shape of the poletip to give the desired field distribution in the gap. MIRT uses the output of POISSON and an input file containing

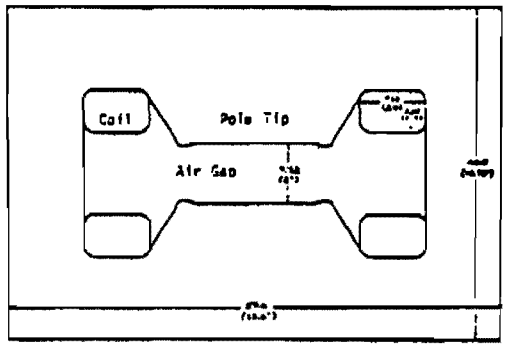


Figure: B.1.5.1: Cross section of H-shaped dipole magnet.

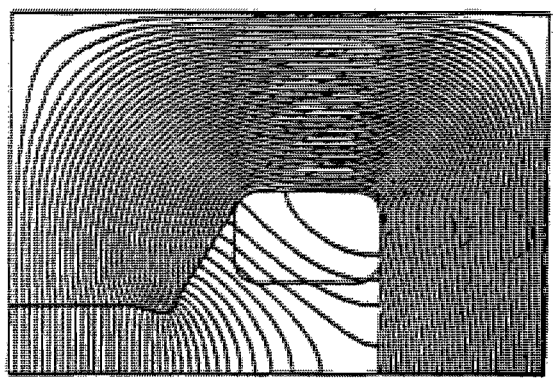


Figure B.1.5.2: Field distribution of dipole magnet; only one quarter of the magnet is shown. The poletip has been trimmed using MIRT.

the specified fields and poletip boundaries to make the required changes. Figures B.1.5.1 and B.1.5.2 shows an example of a poletip after a typical MIRT optimization.

The change in the poletip boundary is made by adding or subtracting "bumps" to the boundary. The purpose of this subsection is to describe the shape of the bumps. There are five shapes: triangular, trapezoidal, three interval, left-sided two interval, and right-sided two interval. These bumps can be either positive or negative, that is, can add or take away iron from the poletip. The triangular and two interval bumps are specified by three points on the magnet boundary and the other two bumps require four points. Figure B.1.5.3 shows the labeling of the points defining the bumps.
The solid line is the original boundary and the dotted lines illustrate a typical distortion of the boundary.

A bump is determined by one parameter, which we will call u. The formulas for the shape functions describing the two and three interval bumps are as follows,

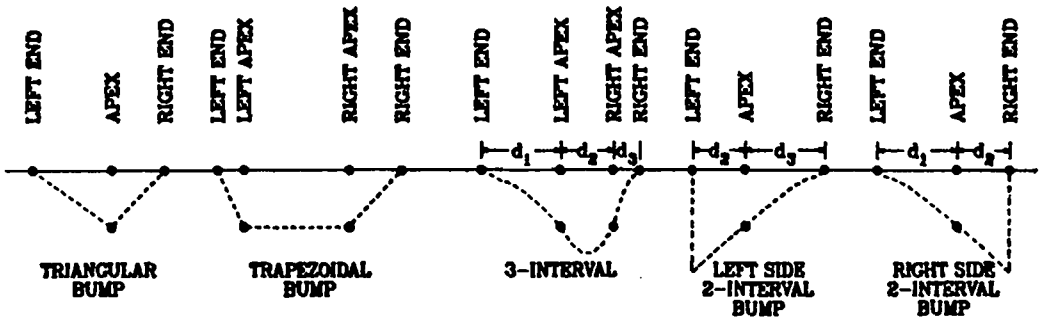


Figure B.1.5.3: Nomenclature for the five types of bumps used to trim the poletips in MIRT. Bumps can overlap and be positive or negative. The solid line is the original boundary and the dotted lines illustrate typical distortions. A left-sided two interval bump is specified by letting the left end and left apex points be the same. The right-sided two-interval bump is specified by letting the right apex and right end points be the same.

$$y = y_o - \begin{cases} a_1 x^2 & , \quad 0 \le x < d_1 \\ u - a_2(x-b)^2 & , \quad d_1 \le x < d_1 + d_2 \\ a_3(x - d_1 - d_2 - d_3)^2 & , \quad d_1 + d_2 \le x \le d_1 + d_2 + d_3 \end{cases} \tag{B.1.5.1}$$

$$y = y_0 - \begin{cases} u - a_2 x^2 & , \quad 0 \le x < d_2 \\ a_3(x - d_2 - d_3)^2 & , \quad d_2 \le x < d_2 + d_3 \end{cases} \quad , \quad d_1 = 0 \tag{B.1.5.2}$$

$$y = y_0 - \begin{cases} a_1 x^2 & , \quad 0 \le x < d_1 \\ u - a_2(x - d_1 - d_2)^2 & , \quad d_1 \le x \le d_1 + d_2 \end{cases} \quad , \quad d_3 = 0 \tag{B.1.5.3}$$

where the a's and b's are complicated functions of the length of intervals between points and are determined by matching the sections of the functions and their slopes at the apex points. (Note: the left-sided two-interval bump is incorrectly programmed in the code. It will be corrected soon.)

indentBumps may overlap and hence one can achieve rather complicated poletip shapes. The number of bumps that can be used at one time depends on the number of points in the airgap at which the field is being specified. The optimization is based on a least squares procedure and therefore the most meaningful results are obtained when the number of points to be fitted is larger than the number of parameters being adjusted.

Chapter B.2

SIMPLE EXAMPLE H-SHAPED DIPOLE MAGNET

B.2.1 Run on the Los Alamos CRAY Computer

This section contains the input necessary to calculate the magnetic field distribution in the vertical cross section of a long dipole magnet used in circular particle accelerators. Because the dipole is long, the calculation of the field far from the ends of the magnet is essentially a two dimensional problem. The magnet cross section has fourfold symmetry and therefore it is only necessary to describe one-fourth of the magnet. The first thing to do is make a diagram of the magnet cross section and assign (x, y) coordinates to the ends of straight line segments defining the boundaries between different materials as shown in Figure B.2.1.1.

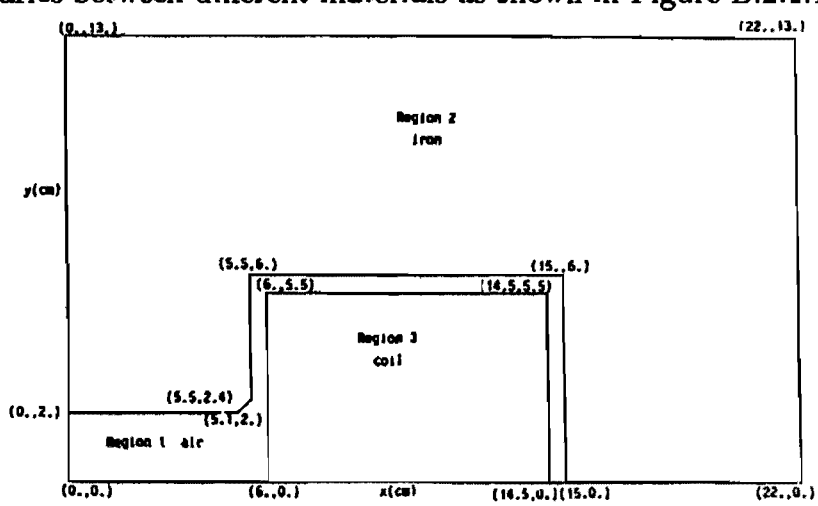


Figure B.2.1.1: Upper right quadrant of the cross section of an H-shaped dipole magnet showing iron, coil, and air regions and the boundary segments between them; (x, y) coordinates are assigned to the endpoints of line segments separating regions.

```
h-magnet test, uniform mesh 4/23/85      [title line; starts in col 2]
$reg nreg=3,dx=.45,xmax=22.,ymax=13.,npoint=5$
                    [nreg: number of regions; dx: horizontal mesh size
                     xmax,ymax: dimensions of Fig. B.2.1.1
                     npoint: number of $po lines]
$po x= 0.0,y= 0.0$
$po x=22.0,y= 0.0$
$po x=22.0,y=13.0$
$po x= 0.0,y=13.0$
$po x= 0.0,y= 0.0$
$reg mat=2, npoint=10$      [2nd region; mat=2 means material is iron]
$po x= 0.0,y= 2.0$
$po x= 5.1,y= 2.0$
$po x= 5.5,y= 2.4$
$po x= 5.5,y= 6.0$
$po x=15.0,y= 6.0$
$po x=15.0,y= 0.0$
$po x=22.0,y= 0.0$
$po x=22.0,y=13.0$
$po x= 0.0,y=13.0$
$po x= 0.0,y= 2.0$
$reg mat=1,cur=-25455.791, npoint=5$      [3rd reg.; mat=1 plus cur=(amps) means coil reg.]
$po x= 6.0,y= 0.0$
$po x=14.5,y= 0.0$
$po x=14.5,y= 5.5$
$po x= 6.0,y= 5.5$
$po x= 6.0,y= 0.0$
```

Figure B.2.1.2: The input file hmag1 for AUTOMESH. The first line is a title. Each region is described by a region line starting with the symbol $reg. The region line is followed with several point lines ($po) defining points on the boundary segments. FORTRAN namelist format is used. The title must begin in column 2. Text in boxes is comment information.

These coordinates are next used to create the input file for AUTOMESH. The program AUTOMESH constructs a triangular mesh of points over all physical regions. The program LATTICE, which runs after AUTOMESH, deforms that mesh to make lines in the mesh coincide with the boundary segments. Figure B.2.1.2 shows the AUTOMESH input file corresponding to Figure. B.2.1.1.

```
automesh      [Executes AUTOMESH; program requests name of input file]
 ? type input file name
? hmag1
 region no.  1
 ok
 region no.  2
```

```
 ok
 region no.  3
 ok
stop
      At completion AUTOMES generates 2 files:
      1. TAPE73: Input for LATTICE, 2. OUTAUT: Output listing from AUTOMES
automes ctss time  .456 seconds
cpu= .175     sys= .037     i/o+memory= .244

all done
```

Figure B.2.1.3: Terminal session with AUTOMESH on the CRAY. Underlined words are those entered by the user.

Figure B.2.1.3 shows the terminal output when AUTOMESH is executed on the CRAY. Henceforth any words underlined in examples of terminal sessions are understood to be words typed by the user. Other text is generated by the program.

The next step is to execute the program LATTICE. An interactive session on the CRAY is shown in Figure B.2.1.4.

```
lattice    Execute LATTICE with input file TAPE73 generated by AUTOMES
 ?type input file name
?  tape73
 beginning of lattice execution
 dump 0 will be set up for poisson
 h-magnet test, uniform mesh 4/23/85
 ?type input values for con(?)
?  *6 0 *46 6 s
                 Changes to problem constants in array CON( )
                 CON(6)=0 specifies internal permeability table
                 CON(46)=6 specifies symmetry type for h-magnet
 elapsed time = 0.7 sec.
 iteration converged
 elapsed time = 1.0 sec.
 generation completed
 dump number 0 has been written on tape35
 stop            2 output files generated:  TAPE35 and OUTLAT
lattice ctss time      1.457        seconds
cpu= .954     i/o= .440     mem= .062

all done
```

Figure B.2.1.4: Interactive session with program LATTICE.

The user can change program constants contained in the array CON(). The format for entering changes is described in Sec. B.3.1. The list of all CON's that

can be changed is contained in Sec. B.13.7. This detail is not important for our example. It is sufficient to note that CON's can be changed at this point. In the above example, CON(6) = 0 tells the program to use the permeability table for iron as stored in the program itself. There are options that let the user specify his own table. By specifying the symmetry type CON(46)=6, the user is making full use of the symmetry of the problem in finding the solution, hence is saving time.

Upon completion, LATTICE generates two output files: TAPE35 and OUTLAT. TAPE35 is a binary file used as input to TEKPLOT, POISSON and/or PANDIRA. The file TAPE35 can have several binary records written to it. The record coming from LATTICE is labeled "dump0." The output from POISSON can added records labeled "dump1", "dump2", etc. This will be described below. The file OUTLAT is a summary of input and output generated by LATTICE. Unless the program is generating unreasonable input to POISSON, there is seldom any reason to look at this file.

The usual way to verify that AUTOMESH and LATTICE have executed properly is to run TEKPLOT, which is a program that plots the input geometry and the mesh generated by LATTICE. Figure B.2.1.5 shows an interactive session with TEKPLOT in which one generates Figures B.2.1.6 and B.2.1.7 shown below.

```
tekplot     [Executes TEKPLOT Program; requests input data]
 ?type input data- num, itri, nphi, inap, nswxy,
?  s     [s means skip in FREE format; hence use default values]
 input data
 num= 0    itri= 0    nphi= 0    inap= 0    nswxy= 0
 plotting prob.  name = h-mag test, uniform mesh 4/23/85
 ?type input data- xmin, xmax, ymin, ymax,    [Plot limits on x and y]
?  s     [Skip; hence use default values]
 input data
 xmin=0.000    xmax= 22.000    ymin=0.000    ymax= 13.000
 ?type go or no
?  go    [After "go", screen clears clears and TEKPLOT generates Fig. B.2.1.6]
 ?type input data-num, itr, nphi, inap, nswxy,    [<CR> clears screen & gives this prompt]
?  0 1 s    [itri=1 means plot the mesh]
 input data
 num=0    itri= 1    nphi=0    inap= 0    nswxy= 0
 plotting prob.  name = h-magnet test, uniform mesh 4/23/85
 ?type input data- xmin, xmax, ymin, ymax,
?  s
 input data
 xmin=0.000    xmax= 22.000    ymin=0.000    ymax= 13.000
 ?type go or no
      [If "no", will go back to correct input; "go" clears screen and generates Fig. B.2.1.7.]
?  go
```

```
?type input data- num, itri, nphi, inap, nswxy,     <CR> gives this prompt
?  -1 s     num=-1 means dump number -1, hence quit; must type "s"
tekplot ctss time    .824 seconds
cpu= .054      i/o= .711      mem= .059
all done
```

Figure B.2.1.5: Interactive session with TEKPLOT. By choosing the default values for the parameter *itri*, one gets a plot of the magnet geometry without the mesh. On the second time around, we choose *itri*=1, which gives a plot of the mesh. *Num* stands for dump number, hence *num*=0 means get the information from the lattice dump. The program is terminated by letting *num*=-1 on the third go-around.

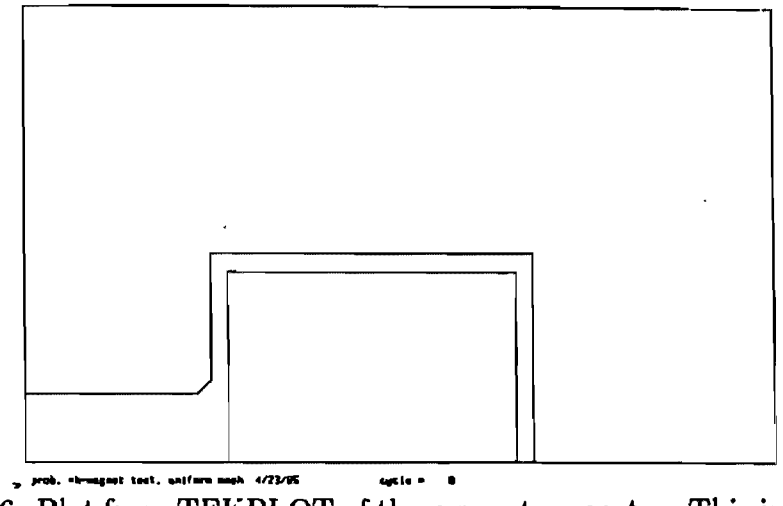


Figure B.2.1.6: Plot from TEKPLOT of the magnet geometry. This is a verification of the input data to AUTOMESH for the problem "h-magnet test, uniform mesh 4/23/85."

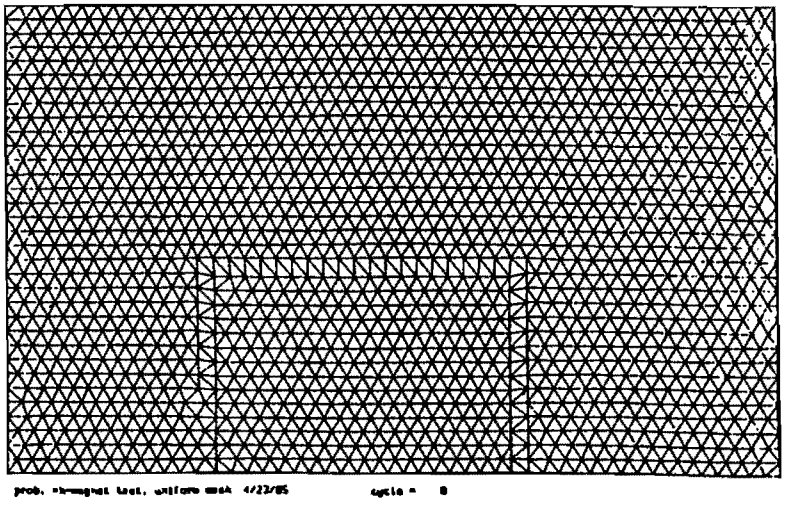


Figure B.2.1.7: Plot from TEKPLOT of the mesh generated by LATTICE for the problem "h-magnet test, uniform mesh 4/23/85."

The mesh looks good. We are now ready to run POISSON. Figure B.2.1.8 shows the interactive session.

```
poisson
 ?type ''tty'' or input file name        [POISSON can execute from file or terminal]
?  tty
 ?type input value for num        [num is dump number on TAPE35]
?  0        [In this case only dump 0 is available.]
 beginning of poisson execution from dump number 0
 prob.  name = h-magnet test, uniform mesh 4/23/85
 ?type input values for con(?)        [Make changes in CON's using FREE format]
?  s        [s=skip;make no changes.]
 elapsed time      =1.1 sec.


0 cycle      amin          amax       residual-air  eta-air  rhoair xjfact
                           gmax      residual-iron eta-iron  rhofe
0     0    0.0000e+00   0.0000e+00   1.0000e+00    1.0000   1.9000 1.0000
                        4.0000e-03   1.0000e+00    1.0000   1.0000
0    50                         rhoair optimized   0.9903   1.9556 lambda = 9.9976e-01
0    50   -4.7296e+04   0.0000e+00   5.7349e-02    0.9903   1.9558 1.0000
                        3.9026e-03   3.4407e-02    1.0039   1.0000
0   100                         rhoair optimized   0.9717   1.9576 lambda = 9.9978e-01
0   100   -1.0012e+05   0.0000e+00   2.5360e-02    0.9717   1.9578 1.0000
                        2.0634e-02   2.0906e-02    0.9834   1.0000
0   200                         rhoair optimized   0.8960   1.9478 lambda = 1.0003e+00
0   200   -1.1958e+05   0.0000e+00   1.8887e-04    0.8960   1.9478 1.0000
                        4.6352e-02   7.9338e-05    0.8900   1.0000
0   370   -1.1939e+05   0.0000e+00   3.7301e-07    0.9367   1.9478 1.0000
                        4.6392e-02   1.6135e-07    0.9328   1.0000

 solution converged in 370 iterations
 elapsed time =5.1 sec.
 dump number 1 has been written on tape35.        [POISSON writes binary record]
 ?type input value for num        [Program loops back to start]
?  -1
      [-1 stops program. Could have continued from dump 1 if wanted
      to change some CON's and run again.]
 stop
 poisson ctss time       5.664       seconds
cpu= 4.166     i/o= 1.014     mem= .485

all done
```

Figure B.2.1.8: Interactive session with POISSON on the CRAY.

POISSON produces two output files TAPE35(dump1) and OUTPOI. Data for plotting the field distribution is contained on TAPE35(dump1) and can be viewed by running TEKPLOT again. Figure B.2.1.9 is the interactive session with TEKPLOT which produces the plot shown in Figure B.2.1.10.

```
tekplot    [Executes TEKPLOT]
 ?type input data-num, itr, nphi, inap, nswxy
?  1 0 20 s    [num=1 means dump1; nphi=20 is number of potential lines plotted]
 input data
num= 1      itri= 0      nphi= 20     inap= 0      nswxy= 0
 plotting prob.  name = h-magnet test, uniform mesh 4/23/85   cycle = 370
 ?type input data- xmin, xmax, ymin, ymax,
?  s
 input data
 xmin=0.000     xmax= 22.000    ymin=0.000      ymax= 13.000
 ?type go or no
      [If "no", will go back to correct input; "go" clears screen and generates Figure B.2.1.10.]
?  go
 ?type input data- num, itri, nphi, inap, nswxy,    [Carriage return gives this prompt]
?  -1 s    [num=-1 means dump number -1, hence quit; must type "s"]
tekplot ctss time     1.480   seconds
cpu=   .797     i/o=   .611     mem=   .073

all done
```

Figure B.2.1.9: Interactive session with TEKPLOT for generation of field plot.

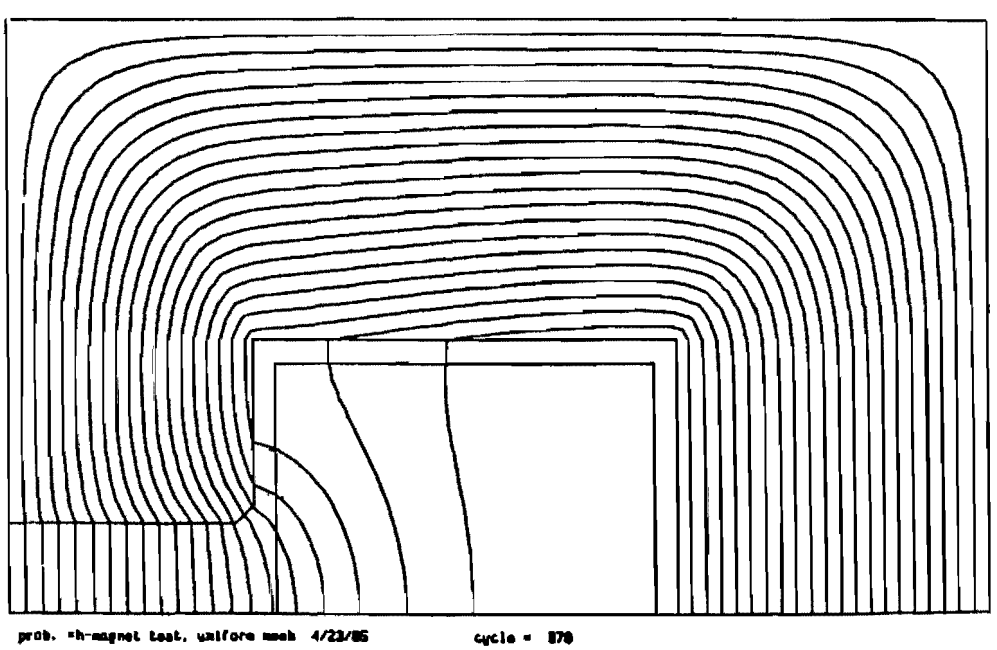


Figure B.2.1.10: Plot of magnetic field lines for problem "h-magnet test, uniform mesh 4/23/85 cycle = 370." This plot normally appears on the screen of your terminal if you have graphics capability.

least squares edit of problem , cycle370

'h' mag symmetry type
stored energy = 1.4249e+03 joules / meter or radian
xjfact= 1.000000

k	l	a(vector)	x	y	bx(gauss)	by(gauss)	bt(gauss)	dby/dy(gauss/cm)	dby/dx(gauss/cm)	afit
1	1	0.000000e+00	0.00000	0.00000	0.000	15212.250	15212.250	0.0000e+00	0.0000e+00	5.3e-04
2	1	-6.829781e+03	0.44898	0.00000	0.000	15210.825	15210.825	0.0000e+00	-6.4635e+00	-3.3e-03
3	1	-1.365822e+04	0.89796	0.00000	0.000	15206.195	15206.195	0.0000e+00	-1.4658e+01	3.7e-03
4	1	-2.048365e+04	1.34694	0.00000	0.000	15197.058	15197.058	0.0000e+00	-2.7097e+01	5.5e-03
5	1	-2.730348e+04	1.79592	0.00000	0.000	15180.665	15180.665	0.0000e+00	-4.7887e+01	6.5e-03
6	1	-3.411342e+04	2.24490	0.00000	0.000	15151.761	15151.761	0.0000e+00	-8.4536e+01	7.8e-03
7	1	-4.090583e+04	2.69388	0.00000	0.000	15100.434	15100.434	0.0000e+00	-1.5084e+02	9.2e-03
8	1	-4.766697e+04	3.14286	0.00000	0.000	15008.404	15008.404	0.0000e+00	-2.7096e+02	1.1e-02
9	1	-5.437198e+04	3.59184	0.00000	0.000	14843.504	14843.504	0.0000e+00	-4.8257e+02	2.4e-02
10	1	-6.097737e+04	4.04082	0.00000	0.000	14554.675	14554.675	0.0000e+00	-8.2909e+02	1.1e-01
11	1	-6.741291e+04	4.48980	0.00000	0.000	14077.744	14077.744	0.0000e+00	-1.3169e+03	5.1e-01
12	1	-7.358221e+04	4.93878	0.00000	0.000	13369.124	13369.124	0.0000e+00	-1.8577e+03	1.5e+00
13	1	-7.938901e+04	5.38776	0.00000	0.000	12435.718	12435.718	0.0000e+00	-2.1782e+03	-7.6e+00
14	1	-8.657337e+04	6.00000	0.00000	0.000	10893.018	10893.018	0.0000e+00	-2.8855e+03	6.7e+00
15	1	-9.114731e+04	6.44737	0.00000	0.000	9613.098	9613.098	0.0000e+00	-2.8434e+03	1.6e+01
16	1	-9.515856e+04	6.89474	0.00000	0.000	8416.533	8416.533	0.0000e+00	-2.5622e+03	5.1e+00
17	1	-9.871191e+04	7.34211	0.00000	0.000	7354.552	7354.552	0.0000e+00	-2.2169e+03	1.9e+00
18	1	-1.017952e+05	7.78947	0.00000	0.000	6439.740	6439.740	0.0000e+00	-1.8967e+03	9.3e-01
19	1	-1.044947e+05	8.23684	0.00000	0.000	5655.367	5655.367	0.0000e+00	-1.6278e+03	5.0e-01
20	1	-1.068700e+05	8.68421	0.00000	0.000	4978.523	4978.523	0.0000e+00	-1.4113e+03	2.5e-01
21	1	-1.089623e+05	9.13158	0.00000	0.000	4387.592	4387.592	0.0000e+00	-1.2403e+03	1.6e-01
22	1	-1.108059e+05	9.57895	0.00000	0.000	3864.280	3864.280	0.0000e+00	-1.1065e+03	9.6e-02
23	1	-1.124277e+05	10.02632	0.00000	0.000	3393.820	3393.820	0.0000e+00	-1.0022e+03	5.8e-02
24	1	-1.138487e+05	10.47368	0.00000	0.000	2964.550	2964.550	0.0000e+00	-9.2096e+02	3.0e-02
25	1	-1.150850e+05	10.92105	0.00000	0.000	2567.346	2567.346	0.0000e+00	-8.5778e+02	1.3e-02
26	1	-1.161495e+05	11.36842	0.00000	0.000	2195.081	2195.081	0.0000e+00	-8.0868e+02	9.6e-04
27	1	-1.170520e+05	11.81579	0.00000	0.000	1842.181	1842.181	0.0000e+00	-7.7062e+02	-9.5e-03
28	1	-1.178000e+05	12.26316	0.00000	0.000	1504.255	1504.255	0.0000e+00	-7.4126e+02	-2.0e-02
29	1	-1.183996e+05	12.71053	0.00000	0.000	1177.824	1177.824	0.0000e+00	-7.1882e+02	-3.4e-02
30	1	-1.188552e+05	13.15789	0.00000	0.000	860.097	860.097	0.0000e+00	-7.0192e+02	-5.7e-02
31	1	-1.191702e+05	13.60526	0.00000	0.000	548.874	548.874	0.0000e+00	-6.8934e+02	-9.7e-02
32	1	-1.193473e+05	14.05263	0.00000	0.000	243.246	243.246	0.0000e+00	-6.7854e+02	-2.4e-01
33	1	-1.193889e+05	14.50000	0.00000	0.000	-59.749	59.749	0.0000e+00	-6.6709e+02	-1.5e+00
34	1	-1.193575e+05	15.00000	0.00000	0.000	-60.019	60.019	0.0000e+00	0.0000e+00	-1.0e+00

Figure B.2.1.11: A section of output from file OUTPOI for problem "h-magnet test, uniform mesh 4/23/85 cycle = 370". This gives the fileds along the x-axis. Note that this printout will not give the field in iron regions.

The file OUTPOI contains a summary of the input data and a table of the field components and their gradients. We show lines 188 to 228 from that file in Figure B.2.1.11.

B.2.2 Execution on Los Alamos Computers

The POISSON group codes are on both the CRAY's and on the MP-Division VAX computers. The procedure for running these programs is slightly different in the two cases. Figure B.2.2.1 lists the commands to run the above test case on the CRAY's. It is assumed that the user knows how to sign on to the CRAY.

```
mass get dir=/lacc/poicodes/cray/exq automesh lattice tekplot poisson
mass get dir=/lacc/poicodes/cray/xmp hmagl oauthl olathl opoihl
automesh
tekplot
poisson
tekplot
fred outpoi
```

Figure B.2.2.1: Series of commands for running test problem on the LANL CRAY's. The first line gets the run files from the common file system. The second line retrieves the AUTOMESH input file hmag. The other files on this line are output files from AUTOMESH, LATTICE and POISSON. They may be useful for comparing your results with ours. Fred is an editor for looking at OUTPOI.

```
FRED AT00$DISK:[AT6HKS]HMAG1.COM
 using FREDLAST switch settings        [Editor opening information line]
$1$dua3:[AT6HKS]HMAG1.COM;7 14 lines
*T*    [Tells FRED to type all the lines in the file HMAG1.COM]
   1 $COPY AT00$DISK:[AT6HKS]HMAG1.AUT []
   2 $FRED HMAG1.AUT
   3 T*        [Type the entire AUTOMESH input file HMAG1.AUT]
   4 END       [Leave editor]
   5 $Run AT00$DISK:[AT6HKS]AUTOMESH
   6 HMAG1.AUT
   7 $RUN AT00$DISK:[AT6HKS]LATTICE
   8 TAPE73    [Tells LATTICE to use TAPE73 as input]
   9 *6 0 *46 6 s    [Change of CON's in LATTICE]
  10 $RUN AT00$DISK:[AT6HKS]POISSON
  11 TTY    [Tells POISSON that input is from the terminal]
  12 0    [dump number expected by POISSON]
  13 S    [Skip changes in the CON array]
```

```
14 -1     dump number = -1 terminates the POISSON run
$         $ is the VAX/VMS system prompt
```

Figure B.2.2.2: A listing of the command file HMAG1.COM using the FRED editor.

When running on the MP VAX/VMS machines, all executable files are under the directory AT00$DISK:[AT6HKS]. The test problem uses a command file called HMAG1.COM to execute AUTOMESH, LATTICE and POISSON; TEKPLOT is used interactively to generate the various plots. At completion the output files OUTAUT.LIS, OUTLAT.LIS and OUTPOI.LIS may be compared with the corresponding output files OAUTH1.LIS, OLATH1.LIS and OPOIH1.LIS, which are stored in directory AT00$DISK:[AT6HKS]. The step by step procedure is shown in Figure B.2.2.2.

B.2.3 Executing on Systems Outside of Los Alamos

The magnetic tape containing the POISSON Group Codes sent outside users contains the above test problem input and output files. The files will have different names depending on whether the user receives the CRAY version or the VAX/VMS version. Figure B.2.3.1 list the file names that should be on the magnetic tape.

VAX/VMS	CRAY
HMAG1.COM	-----
HMAG1.AUT	HMAG1
OAUTH1.LIS	OAUTH1
OLATH1.LIS	OLATH1
OPOIH1.LIS	OPOIH1

Figure B.2.3.1: List of input and output files included with POISSON Group Codes.

Chapter B.3

POISSON Input to LATTICE and AUTOMESH

Before the code AUTOMESH, the only way to enter data describing the physical geometry of the problem was through LATTICE. Although most users will use AUTOMESH to enter input, it is worthwhile understanding the structure of LATTICE input so that the structure of AUTOMESH makes sense. Furthermore there may be occasions when the user wishes to modify the output of AUTOMESH, which becomes the input to LATTICE. In this case, the user needs an understanding of the input to LATTICE.

B.3.1 Format-free Input Routine FREE (I,RAY1,N1,...,RAYI,NI)

The authors of the Poisson Group programs developed their own format-free input routine to make it easier to enter data into all programs except AUTOMESH. The input into AUTOMESH is via the standard FORTRAN NAMELIST method.

The other Poisson Group programs expect most of the input to be in a format that can be read by one of the following CALL statements:

```
CALL FREE(1,RAY1,N1)

CALL FREE(2,RAY1,N1,RAY2,N2)

CALL FREE(3,RAY1,N1,RAY2,N2,RAY3,N3),
```

where RAY1, RAY2, and RAY3 are arrays of length at least equal to N1, N2, and N3 respectively. In some cases the array length is variable. The CALL statements and dimensions are part of the program and hence not under the user's control,

except for arrays of variable length. Section B.3.2 spells out in detail which of the three CALL statements are being used to enter data and what freedom the user has.

The FREE format uses special characters to shorten input and to save array space. These characters and their functions are described below, and an example is given which uses all of them. When the forms in the left-hand column below are used for input, FREE interprets them as explained on the right. The case (upper or lower) for the letters R, S, and C may be important on some computers.

***I X** This notation means store the number X, in the location (I) of the current array. If there are numbers following X, they are stored in locations (I+1), (I+2), etc.

XRN This notation means store the number X, in N successive locations in the current array. A blank between X and R is optional. (Think of RN as being shorthand for "repeat N times.")

S This symbol means skip the rest of the input to the current array and go to the next array, or end the read if the current array is the last array in the CALL FREE statement.

C This symbol means count the number of values read into the current array and save the number as N1, N2 or N3 as appropriate. It also acts like S above. The purpose of this is to read in arrays of variable length.

Numbers may be either integers or floating point numbers. The latter can be in simple decimal format ± XX.XX or in scientific format ± X.XXXE ± XX. The exponent must contain a plus or a minus sign. The plus sign in front of the mantissa is optional. The only other non-numeric characters allowed in the input field are the blank and the comma. Either the blank or the comma can be used to separate input values. Comments may follow the last S or C or required numbers on any input line.

The example below illustrates all the above features. A and B are dimensioned arrays, and K is a single variable.

Calling sequence: N = 100; CALL FREE(3,A,5,B,N,K,1)

input line: `-3,4. +5.3E-2 R2 S *20 .1R10 C 13 THIS IS AN EXAMPLE.`

This input produces the array values:

A(1) = –3

A(2) = 4.0

A(3) = 0.053

A(4) = 0.053

A(5) = unchanged

B(1) thru B(19) = unchanged

B(20) thru B(29) = 0.1

N = 10

K = 13

One final important note. The FREE entry format requires all floating point numbers to have a decimal point. For example, ANGLE = 90 degrees must be entered as "90." in order to be recognized correctly. Leaving out the decimal point is a common beginner's mistake.

B.3.2 POISSON/PANDIRA Inputs to LATTICE

Logically the user would begin by making an input file to AUTOMESH, but to understand the reasoning behind AUTOMESH it is important to understand the structure of the input file to LATTICE. The input file for LATTICE is called TAPE73 for historical reasons.

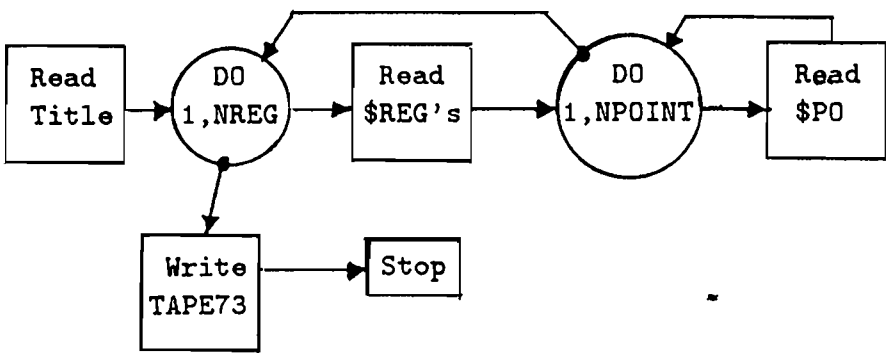


Figure B.3.2.1: Flow Diagram for Read statements in LATTICE for POISSON and PANDIRA.

The structure of the read statements in LATTICE is shown in Fig. B.3.2.1. The first data line can have anything in columns 2 through 80. If the first column is non-blank, then this data set is for a SUPERFISH problem. If the first column is blank, then this data set is for a POISSON or PANDIRA problem. The characters in columns 2 through 65 are stored and used in printouts for run identification. Because of the smaller word size on the VAX as compared with the CRAY, only columns 2 through 33 are available to the user for run identification when running on the VAX.

The reason that LATTICE distinguishes between SUPERFISH and POISSON runs is because LATTICE initializes the elements of the CON and C arrays with their default values. Some of the CON's and C's have different meanings for SUPERFISH as compared to POISSON or PANDIRA, and therefore the default values are different.

The next line (or several lines if needed) is reserved for making changes in the default values of the CON's, (elements in the CON array). Only one CON absolutely must be changed; the user must tell the program the number of regions, NREG = CON(2). NREG is the upper limit of the DO-loop for the next read statements.

There are several other CON's that should be examined at this point. Many of these can *only* be changed in LATTICE if they are to have any effect at all on the problem. A list and brief description are given in Table B.3.2.1.

Other CON's can be changed from their default values at this time even though they have no effect in LATTICE. The changes will carry through to the output file TAPE35 and be available to the other programs when needed.

The format for entering the changes in the CON's is the special free format written for this program and described in Sec. B.3.1. The following example will illustrate the power of this format. Suppose we want to change CON(2), CON(9), and CON(21) through CON(24). The input line might read as follows:

*2 10 *9 2.54 *21 1 1 0 0 S

The * occurs before the number of the element to be changed. When several elements in a row are to be changed, only the first one need be indicated by a star. The final notation S means skip the rest of the elements in the array. This same free format is used to enter elements into the C and B arrays that come next.

Table B.3.2.I CON's that can only be changed in LATTICE

Number	Name	POISSON Default	Description
CON(2)	NREG	None	Number of Regions in the problem geometry. Presently, NREG must be ≤ 31.
CON(9)	CONV	1.0	Conversion factor for the units of length in the problem. CONV = 1.0 for centimeters; CONV = 0.1 for millimeters; CONV = 2.54 for inches; etc.
CON(21)	NBSUP	0	Indicator for boundary conditions on the UPper,
CON(22)	NBSLO	1	LOwer, RighT, and LeFt boundaries of the rectangu-
CON(23)	NBSRT	0	lar region defining the problem. For magnet problems
CON(24)	NBSLF	0	a default value of 0 indicates a Dirichlet boundary condition, which means magnetic field lines are parallel to the boundary lines; a default value of 1 indicates Neumann boundary conditions, which mean magnetic field lines are perpendicular to the boundary line.
CON(32)	IPRINT	0	An indicator for print options. IPRINT = 0 gives no printout; IPRINT = −1 causes LATTICE to write the (X, Y) coordinates of mesh points to OUTLAT on the CRAY or to OUTLAT.LIS on the VAX. This parameter can be changed again in POISSON or PANDIRA and produces other print options in those programs. It is not often that this write to OUTLAT is of much use, but the option exists.
CON(37)	MAP	1	A parameter in the conformal transformation $W = Z**MAP/(MAP*RZERO**(MAP-1)$. RZERO is CON(125). LATTICE needs this value to calculate the current density in the transformed geometry. MAP $\neq 1$ – the current density is adjusted to conform to the transformed geometry in all closed regions. MAP = 1 – no current density adjustment. Note: If the user does not want any current density adjustment (the correct density for the transformed geometry has been input), MAP should not be changed until execution of POISSON/PANDIRA.

Table B.3.2.I (cont.) CON's that can only be changed in LATTICE

Number	Name	POISSON Default	Description
CON(70)	ICAL	0	Indicator for the type of formula to use in calculating the current associated with a mesh point. ICAL = 0 means use the standard formula; ICAL = 1 means use the angle formula. The latter formula gives more accurate fields near coil boundaries.
CON(79)	RHOXY	1.6	The starting over-relaxation factor for the irregular mesh generation. It is seldom that the user will want to change this.
CON(81)	NOTE	1	A flag for determining the order in which the mesh points are relaxed. NOTE = 0 gives the order: air points, interface points, then iron points. [Caution: For PANDIRA runs NOTE = 0 must be used.] NOTE = 1 gives the order: (air + interface) points then iron points.
CON(84)	EPSO	10^{-5}	The convergence criterion for mesh generation. There is seldom a reason to change this number, but if LATTICE has trouble converging, increasing EPSO may help.
CON(123)	TNEGC	0.0	A parameter used in conformal transformation. Input the total negative current in original geometry. LATTICE stores the negative transformed currents.
CON(124)	TPOSC	0.0	A parameter used in conformal transformation. Input the total positive current in original geometry LATTICE stores the positive transformed currents.
CON(125)		1.0	The scaling factor in the conformal transformation.. See CON(3) above. Normally RZERO is the aperture radius.

There are six elements in the C-array for each region; they are called "region constants". The first of these, C(1), is an arbitrary region identification number. These numbers need not be in numerical order.

The second region constant, C(2), is a material code that tells the program whether the region being defined is air or a material with magnetic or dielectric properties. The constant C(2) can take on 12 values, which are summarized in Table B.3.2.II.

Table B.3.2.II. Material Codes for a Given Region.

C(2)	Material Property
0	All points inside this region are omitted from the problem.
1	Air or current carrying coil ($\kappa_m = \kappa_e = 1$)
2	Iron using permeability table internal to POISSON or PANDIRA, or a constant reluctivity (dielectric constant) $\gamma(\kappa_e)$ when CON(6) = -1
3	Iron (dielectric) with external input table no.1, or a second constant $\gamma(\kappa_e)$
4	Iron (dielectric) with external input table no.2, or a third constant $\gamma(\kappa_e)$
5	Iron (dielectric) with external input table no.3, or a fourth constant $\gamma(\kappa_e)$
6	Permanent magnet (electret)with straight line B(H) (D(E)) function (PANDIRA only)
7 through 11 are the same as 6	

When C(2) = 0 for a region, no mesh is set up in this region. The treatment of the boundary points of such a region are determined by the region constant C(6) discussed below.

POISSON and PANDIRA have one internal table of permeability vs. the magnitude of the magnetic field H. By using options C(2) = 3,4,5, the user can set up three other tables. How these tables are entered is described in Chapter B.5. When doing problems with permanent magnetic materials, which requires using PANDIRA, one can enter up to 6 other permeability functions. These entries also are described in Chapter B.5.

The third region constant, C(3), for magnet problems is the total current in amperes at a point, along a line, or in an area. If the region consists of a point, then the current through the point is called a current filament. If the region is a line, then the current through the line is called a current sheet. It is assumed that the current is uniformly distributed along the line. Likewise when the region is an area, the current is distributed uniformly over the area. The fourth regional constant C(4) provides an alternative way of entering currents. (See below.) In electrostatic problems C(3) is the fixed electrical potential at the point or on the line. When this

region is an area, the whole area is at the constant potential C(3).

The fourth region constant, C(4), is the current density (amps/length^2) through two-dimensional regions. For electrostatic problems it is the charge density in units of Coulombs/length^2.

The fifth region constant, C(5), is an integer indicating the type of triangle to be used in defining the logical mesh for the region. There are three choices:

C(5) = 0, equal weight triangles (the default)

C(5) = 1, isosceles triangles

C(5) = 2, right triangles.

Equal weight and isosceles triangles are geometrically the same in a strictly uniform mesh. The difference comes in the relaxation process by which the *logical* mesh is deformed into the *physical* mesh. The distinction between the logical and physical mesh is discussed below when we discuss input to the B-array. The default is probably best choice.

The sixth region constant, C(6), is called a special boundary indicator. This constant is used for two purposes. It is used to indicate special fixed potential points, and it is used to indicate the boundary condition on a boundary of the problem that does not coincide with the extreme rectangular logical boundary of the problem. When it is used for the latter purpose, it can have a value of 0 or 1. C(6) = 1 indicates a Neumann boundary condition and C(6) = 0 indicates a Dirichlet boundary conditon. The default values for C(6) are C(6) = 0 for the first region and C(6) = 1 for all other regions. This latter default is suitable for all regions interior to the problem area.

When C(6) is used for the purpose of indicating fixed potential points in electrostatic problems, it takes on the value -1, and the special fixed potential value is entered as C(3), as described above. Note that setting up constant potentials (magnetic or electrostatic) can also be handled in POISSON. See subsection B.5.3.4.

The B-array requires a list of logical and physical coordinates for the boundary of each region. The first stage of any problem using LATTICE is to set up a physical picture of the geometry and assign physical coordinates (x, y) to points on the boundaries of the various regions. The second stage is to superimpose a *regular* triangular mesh on the whole area. Each mesh point can be assigned logical coordinates (K, L) as illustrated in Fig. B.3.2.2. One can now associate logical coordinates (K, L) with physical coordinates (x, y) by matching the physical point with the closest logical point. LATTICE will use this association to distort the logical mesh into the so-called physical mesh, consisting of *irregular* triangles as illustrated in Fig. B.3.2.3. The tediousness of constructing the logical mesh and making the association of coordinates is the main reason AUTOMESH was created.

Since LATTICE connects points on boundaries with straight lines, one need only specify points at the ends of long straight segments, but approximating a circular arc with straight lines requires many points.

Once again the special free format is used to enter the values in the B array. The order of the input is K(1), L(1), X(1), Y(1), K(2), L(2), X(2), (Y2), etc. The origin of coordinates in the logical mesh is (1,1), not (0,0). This input list is terminated with the free format character C, which stands for "Count the number of input values."

As indicated in Fig. B.3.2.1, the data groups for the C and B arrays are repeated for each region. The first region defines the largest rectangular region containing the problem and its boundary values must contain the largest K and L values in the mesh. The data for the second region redefines, or overwrites, the region constants for all mesh points belonging to that region. Data for each following region overwrites previously defined values in the same way. For example, suppose one wishes to define a coil region inside of an air region. If the coil region were given first in the input data and then the air region, the coil region would be overwritten and this coil would not exist in the problem.

Usually each region is closed, i.e., the data for the first and last boundary points of the region are identical. However, it is possible to specify data for a "point region" or a "line region". The purpose of this specification would be to define the physical coordinates of specific points, and perhaps also the total current at a point (a current filament) or a special fixed potential value at that point for an electrostatic problem. For point and line regions, the input values of C(2) and C(5) are not used in the program. Figure B.3.2.5 is an example of how the regional data is entered for the geometry shown in Fig. B.2.3.4. The logical mesh numbers were taken from a mesh slightly finer than that shown in Fig. B.3.2.2.

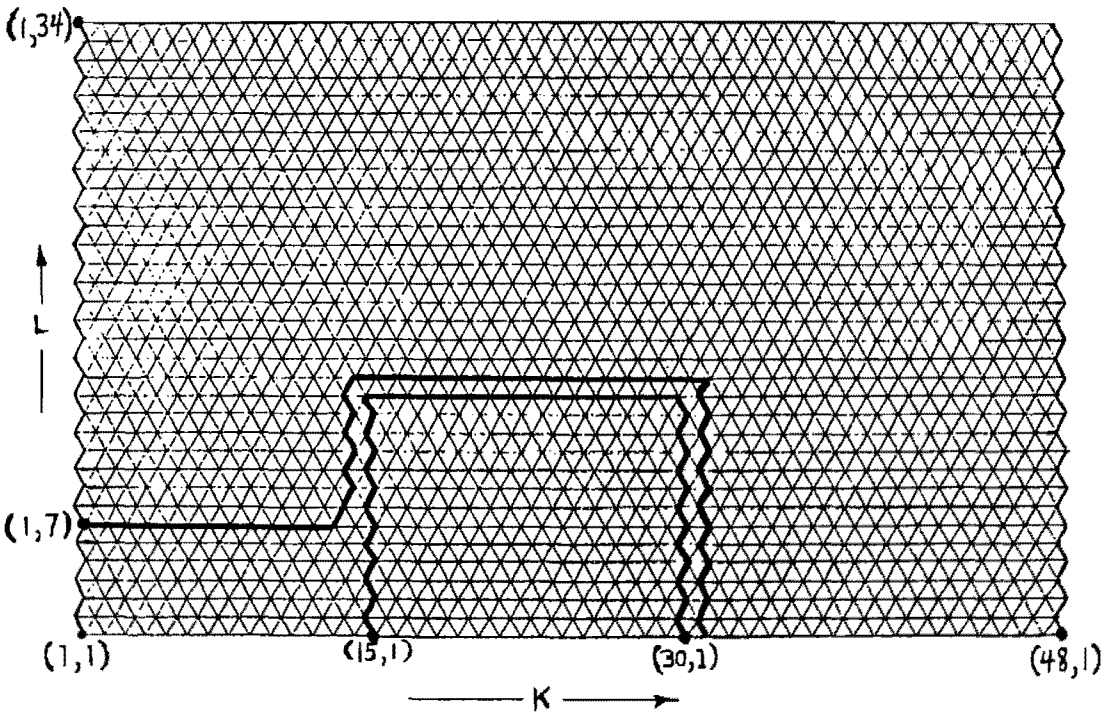


Figure B.3.2.2: A logical mesh for the H-shaped magnet.

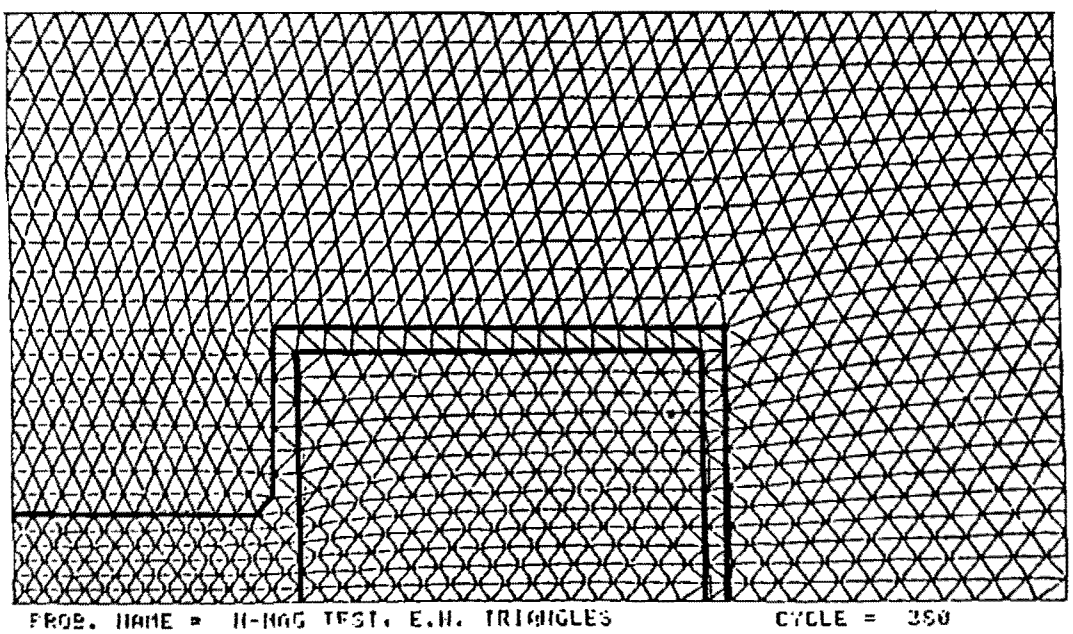


Figure B.3.2.3: The corresponding physical (relaxed) mesh for H-shaped magnet.

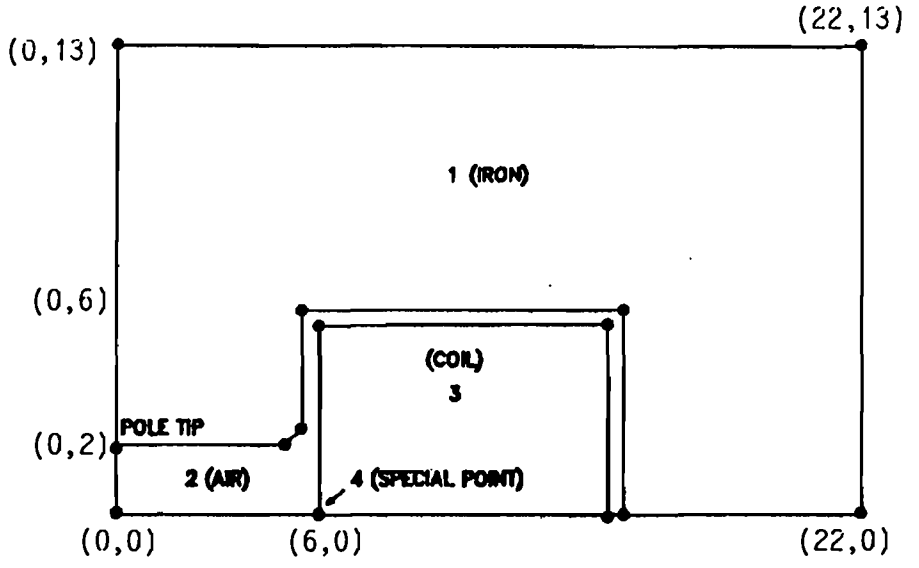


Figure B.3.2.4: Example of input geometry to LATTICE for H-shaped magnet.

```
H-MAGNET TEST, UNIFORM MESH  4/23/85
*2  3 *21  0 1 0 0 *9   1.0000 SKIP
  1  1      0.0000      0.0000  0  0 REGION
   1   1    0.0000     0.0000
  50   1   22.0000     0.0000
  50  34   22.0000    13.0000
   1  34    0.0000    13.0000
   1   1    0.0000     0.0000 COUN
  2  2      0.0000      0.0000  0  1 REGION
   1   6    0.0000     2.0000
  13   6    5.1000     2.0000
  13   7    5.5000     2.4000
  14   8    5.5000     2.8000
  13   9    5.5000     3.2000
  14  10    5.5000     3.6000
  13  11    5.5000     4.0000
  14  12    5.5000     4.4000
  13  13    5.5000     4.8000
  14  14    5.5000     5.2000
  13  15    5.5000     5.6000
  14  16    5.5000     6.0000
  35  16   15.0000     6.0000
  34  15   15.0000     5.6000
  35  14   15.0000     5.2000
  34  13   15.0000     4.8000
  35  12   15.0000     4.4000
  34  11   15.0000     4.0000
  35  10   15.0000     3.6000
```

```
34   9   15.0000    3.2000
35   8   15.0000    2.8000
34   7   15.0000    2.4000
35   6   15.0000    2.0000
34   5   15.0000    1.6000
35   4   15.0000    1.2000
34   3   15.0000    0.8000
35   2   15.0000    0.4000
34   1   15.0000    0.0000
50   1   22.0000    0.0000
50  34   22.0000   13.0000
 1  34    0.0000   13.0000
 1   6    0.0000    2.0000 COUN
3  1 -25455.7910      0.0000  0  1 REGION
14   1    6.0000    0.0000
33   1   14.5000    0.0000
34   2   14.5000    0.3929
33   3   14.5000    0.7857
34   4   14.5000    1.1786
33   5   14.5000    1.5714
34   6   14.5000    1.9643
33   7   14.5000    2.3571
34   8   14.5000    2.7500
33   9   14.5000    3.1429
34  10   14.5000    3.5357
33  11   14.5000    3.9286
34  12   14.5000    4.3214
33  13   14.5000    4.7143
34  14   14.5000    5.1071
33  15   14.5000    5.5000
14  15    6.0000    5.5000
15  14    6.0000    5.1071
14  13    6.0000    4.7143
15  12    6.0000    4.3214
14  11    6.0000    3.9286
15  10    6.0000    3.5357
14   9    6.0000    3.1429
15   8    6.0000    2.7500
14   7    6.0000    2.3571
15   6    6.0000    1.9643
14   5    6.0000    1.5714
15   4    6.0000    1.1786
14   3    6.0000    0.7857
15   2    6.0000    0.3929
14   1    6.0000    0.0000 COUN
```

Figure B.3.2.5: An example of a TAPE73 file (input to LATTICE) for the H-shaped magnet shown in Fig. B.3.2.4.

B.3.3 POISSON/PANDIRA Input to AUTOMESH

AUTOMESH prepares an input file, called TAPE73, for LATTICE. It constructs the logical mesh from triangles whose size and shape are specified by the user and from the physical region boundaries. It assigns logical (K, L) coordinates and physical (X, Y) coordinates to points on the boundaries of the region. Extra line regions can be added to the problem where requested. These lines form boundaries for changing the size of the triangles. AUTOMESH sets only six of the CONs to default values. AUTOMESH automatically assigns boundary conditions to the problem by setting CON(21) through CON(24). If these boundary conditions are not appropriate, they can be changed in LATTICE when that code asks for CON's changes or they can be changed directly by editing the file TAPE73 which is produced by AUTOMESH. In addition to the boundary conditions already mentioned, it sets CON(2) = NREG and CON(9) = CONV. The default for CON(9) is 1, that is, the length unit is centimeters; this value can be overwritten by including CONV in the REG namelist (see below).

The input to AUTOMESH is the same for either SUPERFISH runs or POISSON/PANDIRA runs with two exceptions. The first exception is the first data line, which is the title for the problem. If column 1 is blank, AUTOMESH assigns POISSON/PANDIRA defaults to some variables; if column 1 is not blank, the program assigns SUPERFISH defaults. The second exception occurs when the user elects to use cylindrical coordinates by later setting CON(19) = ICYLIN = 1. For POISSON/PANDIRA (X, Y) corresponds to (R, Z); for SUPERFISH it is opposite, namely, (X, Y) corresponds to (Z, R). Mathematically this may be confusing, but for most physical problems it is the natural choice. If the user is not satisfied with this convention, he can change it to some extent by setting NSWXY = 1 in TEKPLOT. Of course, this only changes the plots, not the expected inputs to AUTOMESH.

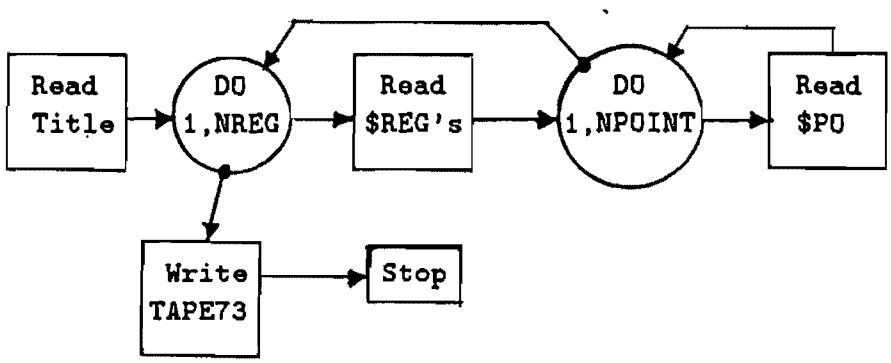


Figure B.3.3.1: Flow chart for read statements in AUTOMESH.

The first line of input to AUTOMESH is the title card. Positions 2 through 80 can be anything. Column 1 should be blank as mentioned above. The next 8 computer words (columns 2 through 65 on the CRAY, columns 2 through 33 on the VAX) are used for output identification.

Following the first line, AUTOMESH expects one or more groups of data. Each group consists of one REG NAMELIST input followed by one or more PO NAMELIST inputs. The first REG NAMELIST must include a value for NREG; NREG is the the number of REG NAMELISTS expected. The READ structure is shown in Fig. B.3.3.1.

NAMELIST is the standard FORTRAN input routine. Each such input starts with a blank in column 1 followed by $"name" where "name" is the name of the input. Here "name" is either REG or PO. Any item in the group may be entered in any order separated by commas. If there is any question about NAMELIST, see a FORTRAN manual. The following is a typical NAMELIST entry for the namelist REG:

```
$REG NREG=5,DX=0.08,XMAX=3.5,YMAX=2.85,IBOUND=1,NPOINT=8$
```

B.3.3.1 The REG NAMELIST.

Twenty-nine quantities can be entered in this NAMELIST for each region of the problem. Most quantities entered for the first region are used for all succeeding regions until changed by a subsequent REG input. This is called "successive region data overwrite". Certain quantities *must* be entered, while others have meaningful default values. The following is a list in alphabetical order describing the quantities as used by POISSON and PANDIRA. Some can only be entered in the first REG NAMELIST and must not be changed in subsequent regions. These are marked by ◇ before the variable.

Table B.3.3.I Region Namelist Variables.

Name	Default	Description
◇ CONV	1.0	Conversion factor for length units. If CONV = 1.0, units are centimeters. To use other units, set CONV to the number of centimeters per unit desired. CONV is the same as CON(9) in the input to LATTICE and should be entered for the first region only.
CUR	0.0	The total current in the region in amperes, or the fixed potential value on the boundary of the region for electrostatic problems. This is the constant C(3) in Sec. B.3.2, Input for LATTICE.
DEN	0.0	The current density in the region for magnet problems; the charge density for electrostatic problems. Units are amps/length^2 or Coulombs/length^2 for areas or amps/length or Coulombs/length for line regions. DEN is C(4) in Sec. B.3.2, Input for LATTICE.
◇ DX	none	The requested width of triangles in the mesh for the first region. It must not be changed for subsequent regions.
◇ DY	($\propto$ DX)	The requested height of triangles in the mesh for the first region. If DY is not specified the default value is either $\sqrt{3}*$ DX/2 if ITRI = 0, or 1, or DY = DX if ITRI = 2 (right triangle option). It must not be changed for subsequent regions.
IBOUND	−2	A special region boundary indicator. See discussion under the sixth region constant C(6) in the Input for LATTICE.

Table B.3.3.I (cont'd.) Region Namelist Variables.

Name	Default	Description
IPRINT	0	If IPRINT = 1, a special diagnostic printout to OUTAUT is provided by the subroutine LOGIC. If IPRINT $\neq$ 0 the mesh coordinates of mesh points are printed out to OUTAUT. If IPRINT = 0, there are no printouts. (IPRINT is not CON(32), but a local AUTOMESH variable.)
IREG	(n)	An arbitrary number identifying the region. The default value is 1 for the first entered REG NAMELIST and is incremented by 1 for each succeeding REG NAMELIST.
◇ ITRI	0	The type of triangle to be used for the mesh in the region. ITRI = 0 means equal weight; ITRI = 1 means isosceles; and ITRI = 2 means right. The distinction between equal weight and isosceles is the way that the relaxation is done from the logical mesh to the physical mesh.
◇ KMAX ◇ KREG1 ◇ KREG2 ◇ LMAX ◇ LREG1 ◇ LREG2	none* none* none* none* none* none*	Used to refine the mesh in a user-defined rectangle of the problem area. When these values are entered, they associate a logical mesh number with a physical position. Thus, KMAX corresponds to XMAX, LMAX corresponds to YMAX, KREG1 to XREG1, etc. This allows the user to force a smaller mesh step in a given region. For example, suppose XREG1 = 10.0, XREG2 = 20.0, KREG1 = 10, KREG2 = 110. This gives (KREG1 − 1) = 9 mesh steps in the X-direction of length 10.0/9 = 1.111 up to the location X = XREG = 10, giving a very coarse mesh. From X = 10.0 to X = 20.0 there will be (KREG2 − KREG1) = 100 mesh steps of length DX = 10/100 = 0.1, giving a finer mesh. The size of the mesh beyond X = 20.0 will depend on the difference between KMAX and KREG2, and between XMAX and XREG2. The same principle applies to the Y-direction. The values of these variables are global and hence should be entered for the first region and not changed in subsequent regions.

*If these values are not entered, the code assigns proper values (see under XREG1, XREG2, etc., below).

Table B.3.3.I (cont'd.) Region Namelist Variables.

Name	Default	Description
◇ LINX	0	A special indicator for vertical line regions. LINX = 0 produces vertical line regions at the locations where mesh size changes occur (XREG1 and XREG2). LINX = 1 produces no vertical line regions at the locations where mesh size changes occur. This parameter was introduced into the code in April of 1986. LINX = 1 can help LATTICE converge under some circumstances.
◇ LINY	0	A special indicator for horizontal line regions. It works the same way as LINX above, but for horizontal line regions at locations where mesh size changes occur (YREG1 and YREG2).
MAT	1	The material code for the region. MAT = 0 means that all points in the region are to be omitted from the problem and requires the use of the special boundary indicator IBOUND. The other possible values of this parameter are: MAT = 1, air or vacuum ($\kappa_m = 1$, $\kappa_e = 1$). = 2, iron with the internal permeability table, or constant reluctivity (dielectric constant) γ (κ_e) when CON(6) = −1. = 3, iron (dielectric) properties from user-defined Table 1, or a second constant $\gamma(\kappa_e)$ = 4, iron (dielectric) properties from user-defined Table 2, or a third constant $\gamma(\kappa_e)$ = 5, iron (dielectric) properties from user-defined Table 3, or a fourth constant $\gamma(\kappa_e)$ = 6 through 11, iron (dielectric) properties for permanent magnets (electrets) with straight line B(H) (D(E)) functions (PANDIRA ONLY).
NPOINT	none	The number of segment endpoints to be entered in the PO NAMELIST that follows this REG NAMELIST.
◇ NREG	none	The number of sets of REG NAMELIST data to be entered for this run. This must be entered in the first REG set and should not be changed in subsequent REG sets.

Table B.3.3.I (cont'd.) Region Namelist Variables.

Name	Default	Description
◇ XMAX	none	Maximum physical X value in the problem. For 3-dimensional problems with cylindrical symmetry where the coordinates are denoted (r, $\varphi = 0$, z) XMAX is the maximum value of r.
◇ XMIN	0.0	Minimum physical X value in the problem. For 3-dimensional problems with cylindrical symmetry where the coordinates are denoted (r, $\varphi = 0$, z) XMIN is the minimum value of r. When XMIN $\neq 0$ the user should consider entering values of XORG = CON(38) in POISSON, PANDIRA or MIRT when asked for CON changes. See Subsection B.5.3.6.
◇ XREG1	XMAX	A line region is added at XREG1. If KREG1 is not set, the width of the triangles will approximately double to the right of XREG1. If KREG1 is set, the triangle width will be determined as described under KMAX, etc. above.
◇ XREG2	XMAX	A line region is added at XREG2. If KREG2 is not set, the width of the triangles will approximately double to the right of XREG2. If KREG2 is set, the triangle width will be determined as described under KMAX, etc., above.
◇ YMAX	none	Maximum physical Y value in the problem. For 3-dimensional problems with cylindrical symmetry where the coordinates are denoted (r, $\varphi = 0$, z) YMAX is the maximum value of z.
◇ YMIN	0.0	Minimum physical Y value in the problem. For 3-dimensional problems with cylindrical symmetry where the coordinates are denoted (r, $\varphi = 0$, z) YMIN is the minimum value of z. When YMIN $\neq 0$ the user should consider entering values of YORG = CON(39) in POISSON, PANDIRA or MIRT when asked for CON changes. See Subsection B.5.3.6.

Table B.3.3.I (cont'd) Region Namelist Variables.

Name	Default	Description
◇ YREG1	YMAX	A line region is added at YREG1. If LREG1 is not set, the height of the triangles will approximately double above YREG1. If LREG1 is set, the triangle height will be determined as described under KMAX, etc., above.
◇ YREG2	YMAX	A line region is added at YREG2. If LREG2 is not set, the height of the triangles will approximately double above YREG2. If LREG2 is set, the triangle height will be determined as described under KMAX, above.

Chapter B.4

OUTPUT FROM LATTICE

The function of LATTICE is to find the physical mesh on which the problem is to be solved and to write the necessary mesh and problem information onto a file called TAPE35. LATTICE also produces an output file called OUTLAT.

The information contained in OUTLAT is usually not needed but may sometimes be helpful if something goes wrong in the solution process. OUTLAT contains, for each region, the region material number, the total current, the current density, the region boundary indicator, IBOUND, and a list of the region logical and physical boundary points. This is followed by a history of the mesh relaxation interation, which consists of the x-residual, η_x, ρ_x, y-residual, η_y and ρ_y. The quantities η_x and η_y are the x and y rates of convergence of the relaxation process. The quantities ρ_x and ρ_y are the over-relaxation factors.

After the iteration history, a table is printed giving the area of each region and the current density in each region. This is followed by a printout of the problem constants, that is, the CON array. Those CON's that have been changed in the input to LATTICE are flagged.

In addition, any error messages generated by running LATTICE are also recorded in this file. Finally, if CON(32) = IPRINT = −1, LATTICE prints a map of the x and y vectors, that is, it gives the coordinates of each mesh point. Figure B.4.1 illustrates an OUTLAT file.

```
beginning of lattice execution
dump 0 will be set up for poisson
h-magnet test, uniform mesh 4/23/85
region number = 1  material = 1  total current =0.0000
current density =0.0000  0-zoning
region boundary indicator= 0

 k   l       x          y
 1   1   0.00000    0.00000
50   1  22.00000    0.00000
50  34  22.00000   13.00000
 1  34   0.00000   13.00000
 1   1   0.00000    0.00000
  .
  .
  .
  .
relaxation parameters, 1433 unknown points.

elapsed time =0.5 sec.

 cycle    residx      etax     rhox     residy      etay     rhoy
   1    1.3783e-02  1.0000  1.6000  8.6575e-04  1.0000  1.6000
   2    1.8220e-02  0.6577  1.6000  1.1829e-03  0.6569  1.6000
   3    1.2090e-03  0.6636  1.6000  8.0609e-04  0.6815  1.6000
   4    8.1228e-03  0.6718  1.6000  5.5710e-04  0.6911  1.6000
   5    5.4555e-03  0.6716  1.6000  3.9792e-04  0.7143  1.6000
   6    3.6785e-03  0.6743  1.6000  2.8638e-04  0.7197  1.6000
   7    2.5302e-03  0.6878  1.6000  2.1345e-04  0.7454  1.6000
   .
   .
   .
  42    1.4031e-05  0.8287  1.7574  2.2786e-06  0.8412  1.7496
  43    1.1584e-05  0.8256  1.7574  1.9094e-06  0.8380  1.7496
  44    9.6645e-06  0.8343  1.7574  1.6023e-06  0.8392  1.7496
iteration converged
elapsed time =0.9 sec.
generation completed.

calculated current densities and areas

region      current densities    area
number        (amps/cm**2)      (cm**2)
   1             0.0000         21.3300
   2             0.0000        217.9200
   3          -544.5089         46.7500
```

```
dump number 0 has been written on tape35.
input or default value
problem constants and variables
   con( )  =            ,h-magnet test, uniform mesh 4/23/85
     ( 2)  =          3,nreg
  con( 6)  =          0,mode
  con( 7)  =  1.000e+00,stack
     ( 8)  =  1.000e+15,bdes
     ( 9)  =  1.000e+00,conv
    ( 10)  =  4.000e-03,fixgam
    ( 18)  =          0,nperm


    (113)  =  0.000e+00,angle
    (114)  =  0.000e+00,rnorm
    (115)  =  0.000e+00,anglz
    (123)  =  0.000e+00,tnegc
    (124)  =  0.000e+00,tposc
    (125)  =  1.000e+00,rzero

solution
problem constants and variables

  ( 3)  =         34,lmax
  ( 4)  =         50,kmax
  ( 5)  =         52,imax
 ( 11)  =        363,nair
 ( 12)  =       1163,nfe
 ( 13)  =         58,ninter


 ( 91)  =          0,numdmp
 (106)  =  1.000e+00,etaair
 (107)  =  1.000e+00,etafe
 (109)  =       1872,itot
 (118)  =      10000,maxdim
 (119)  =       5000,nwdim
```

Figure B.4.1 Sections of the file OUTLAT for the problem "hmagnet, test uniform mesh".

Chapter B.5

Input for POISSON and PANDIRA

B.5.1 Introduction

The structure of the input data for POISSON and PANDIRA is the same. Data may be entered either from a data file or directly from the terminal (TTY). After the code has asked for and received the name of the input source, the flow of data input is as shown in Figure B.5.1.1. The data is entered using the special free format described in section B.3.1.

From the diagram one can see that there are five basic read statements: 1. Dump number, 2. Changes in the CON array, 3. Permeability information, 4. Fixed potential values, and 5. Current filament data. Each of these will be discussed in a subsection below.

B.5.2 Dump numbers

The first line of input data contains only one number, NUM, which is the number of the TAPE35 dump to be read and processed. Dump 0 is written by LATTICE and Dumps numbered greater than zero are written by POISSON or PANDIRA. There are two main uses of the dump feature with NUM>0. The first is to continue a run that has not converged, and the second is to calculate and/or printout some auxiliary quantities that had not been done in the original run. For example, suppose that POISSON has run the maximum number of cycles (say 400) without converging. The code will automatically write "DUMP1" on TAPE35 and quit. The user may change the maximum number of cycles (to 850 say) and specify the dump number as NUM=1. The code will continue the problem where it left off and run an additional 450 iterations. After converging or after reaching the 850 limit, it will write "DUMP2" on TAPE35. If it did not converge, the user can increase the

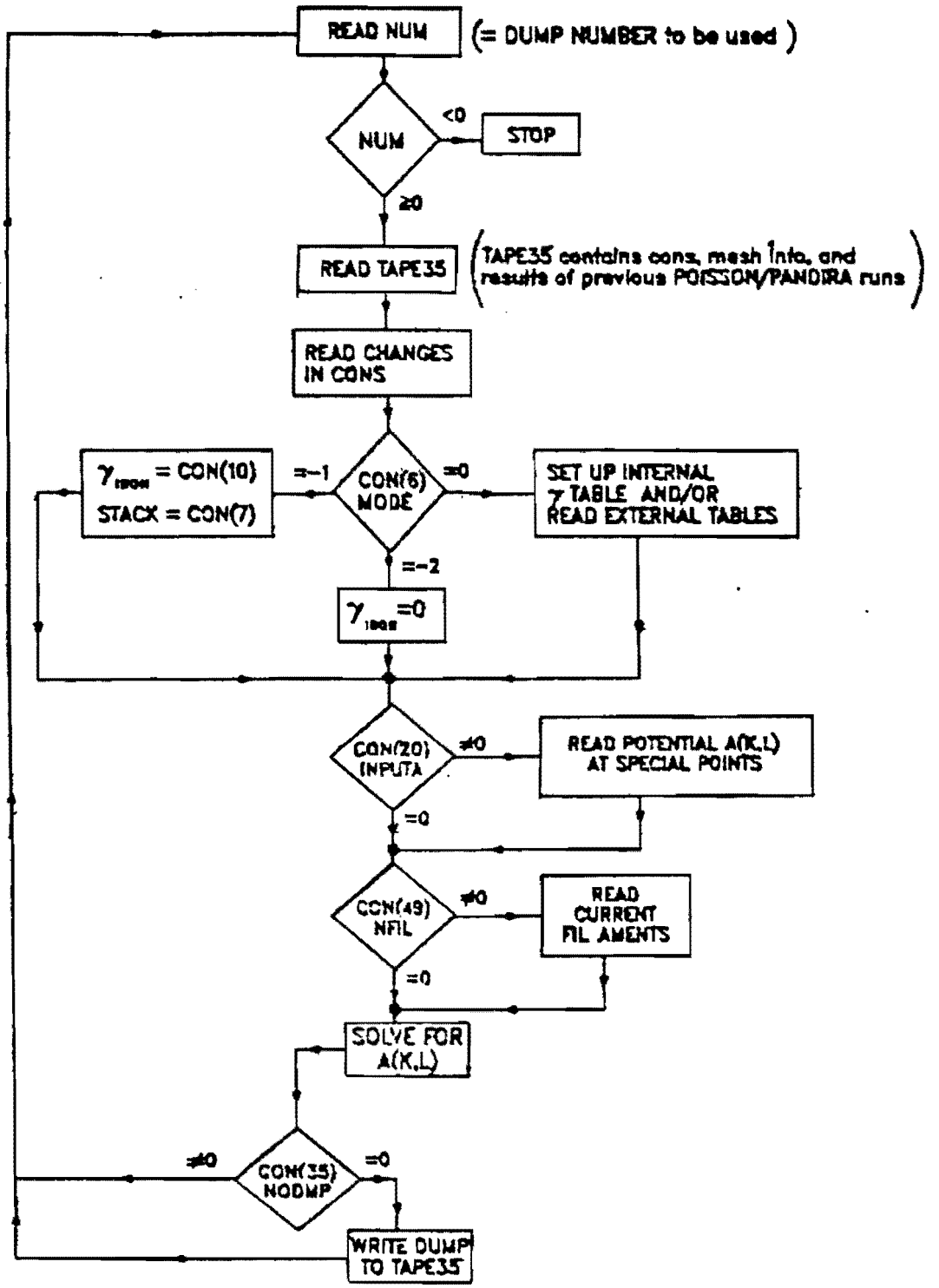


Figure B.5.1.1: Flow Diagram showing Inputs to POISSON or PANDIRA.

maximum number of cycles and run again. The maximum limit on the number of cycles in POISSON is 100,000.

The maximum number in PANDIRA is 20. Suppose the problem converged after 850 cycles, but you decide later to do an harmonic analysis on the field. You can start with DUMP2, enter CON values needed for the harmonic analysis and produce DUMP3, which will contain the required analysis. This will take much less time than running the cycles all over again.

B.5.3 Changes in the CON array

There are a large number of options that can be exercised by changing the values in the CON array. We have divided this section into 10 subsections.

B.5.3.1 Control of input.

Input of permeability data is controlled by CON's (6)=MODE, (7)=STACK, (10)=FIXGAM, and (18)=NPERM as discussed in Sec. B.5.4 below. CON(20) controls the number of special fixed potential values to be read in as discussed in Sec. B.5.5 below. CON(49) controls the number of current filaments to be read in as discussed in Sec. B.5.6 below.

B.5.3.2 Symmetry Options.

CON(19)=ICYLIN controls the choice of cartesian or cylindrical symmetry; CON(46)=ITYPE provides a way to make maximum use of the rotation and reflection symmetry of the magnet.

If ICYLIN = 0 (the default), the problem is assumed to have cartesian symmetry, that is, (x, y) coordinates and all functions independent of z. If ICYLIN = 1, the problem is assumed to have cylindrical symmetry, that is, (r, z) coordinates and all functions independent of the angle ϕ. In TEKPLOT outputs, r is the horizontal axis.

There are 9 standard rotation-reflection symmetry types (ITYPE) for cartesian systems, 3 standard types for cylindrical systems, and a way to construct special ITYPE-codes for symmetries not included in the standard types. Table B.5.3.I shows the standard ITYPE options.

Table B.5.3.I. Standard Symmetry Options for Magnets

ITYPE	Description
CARTESIAN SYMMETRY (ICYLIN=0)	
1	No Symmetry
2	Midplane symmetry*
3	Elliptical aperture quadrupole*
4	Symmetrical quadrupoles*
5	Skew elliptical aperture quads (x-axis is a field line)
6	Symmetrical "H" magnet or elliptical aperture sextupole*
7	Symmetrical sextupole*
8	Elliptical aperture octupole*
9	Symmetrical octupole*
CYLINDRICAL SYMMETRY (ICYLIN=1)	
1	No symmetry
2	Midplane symmetry (field lines are $\perp$ to r-axis for magnet problems; lines of constant potential are $\perp$ to r-axis for electrostatic problems.)
3	Midplane symmetry (r-axis is a line of constant scalar potential for electrostatic problems; does not apply to vector problems.)

*Field lines are perpendicular to the x-axis.

For a detailed description of these symmetry types, see Sec. B.13.2.

In general almost any symmetry desired may be specified by constructing ITYPE as a three digit code word. To understand how to construct this code, one must understand how the code constructs the vector and scaler potentials. These potentials are considered to be the real and imaginary parts of a complex function $F(z = x + iy)$ that is given by the expression

$$F(z) = A(x,y) + iV(x,y) = \sum_{n=1}^{\infty} c_n (z - z_0)^n \qquad \text{(B.5.1)}$$

POISSON computes the fields and gradients by taking the appropriate derivatives of $F(z)$. The symmetry of $F(z)$ determines which of the coefficients c_n are nonzero, pure real, or pure imaginary. There are two distinct types of symmetries: 1. reflection, and 2. rotation. Reflection symmetries determine whether the coefficients c_n are real, imaginary, or complex. Rotational symmetries imply that various terms in $F(z)$ vanish.

Reflection symmetry implies one of two conditions on the median plane (x-axis). Either $V = 0.$, $dA/dn = 0$ (a constant scalar potential boundary) or

$A = 0.$, $dV/dn = 0$ (a constant vector potential boundary) where dA/dn means the derivative in the direction normal to the x-axis. The units digit of CON(46) = ITYPE encodes this information. If ITYPE = htu, then:

$u = 0$ means $A = 0.$ and c_n is pure imaginary, $B_y(x, 0) = 0$,
$u = 1$ means $V = 0.$ and c_n is real, $B_x(x, 0) = 0$
$u = 2$ means no symmetry and hence c_n is complex.

There are two types of rotation symmetry. When the coordinate axes are rotated by an angle $2\pi/n$, either the potentials and fields are identical or are identical but with opposite signs. the tens and hundreds digits of ITYPE encode the rotation symmetry. For example, suppose the magnet is a symmetric quadrupole and has $dA/dn = 0$ on the median plane. Such a magnet has odd parity. Only one-eighth of the magnet need be described in AUTOMESH. It can be shown that the only nonzero coefficients c_n in the summation are those for $n = 2, 6, 10, \ldots$. To describe this sequence the lowest order term ($n = 2$) and the interval ($\Delta n = 4$) need to be given. This is encoded by letting the tens-digit be 4 and the hundreds-digit by 2. In summary then, since the symmetric quadrupole has reflection symmetry which makes the scalar potential V vanish, ITYPE would be 241, which is equivalent to the standard magnet type ITYPE = 4 in Table B.5.3.I above. Other special symmetry types are constructed similarly. For a more complete discussion, see Sec. B.13.2.

B.5.3.3 Electrostatic option.

The theory of electrostatics and magnetostatics is almost identical. See Sec. B.13.1. It is merely a matter of interpretation of the output. Relative reluctivity (1/relative permeability) is replaced by relative permittivity; the vector potential is replaced by the scalar potential. There is only one parameter that must be changed to switch between magnetostatics and electrostatics. One must set CON(66) = XJFACT = 0. The default value of XJFACT is 1. Furthermore, XJFACT has another use which will be described in Sec. B.5.3.4 below.

B.5.3.4 Fixed field option.

Many times one wishes to specify the field at a given point, e.g., at the center of the gap, and somehow determine the value of the current in the coil that will produce that field. This can be done in POISSON by specifying four parameters as summarized in Table B.5.3.II.

Table B.5.3.II. Parameters for Fixed Field Option

Parameter	Default	Definition
CON(8)=BDES	1.0E15	Absolute value of the fixed field. If BDES is not equal to its default value, XJFACT=CON(66) will be adjusted so that the field is BDES within a tolerance XJTOL=CON(67).
CON(40)=KBZERO CON(41)=LBZERO	1 1	The (K,L) coordinates specifying the location of BDES for adjusting the current factor.
CON(66)=XJFACT	1.0	The factor by which all current and current densities will be scaled in POISSON and PANDIRA. When used in this way, the default value is adequate for input. The program will adjust its value and print it out at the end. (As noted above, XJFACT = 0 indicates an electrostatic problem with no currents.)
CON(67)=XJTOL	1.0E-4	The tolerance on the determination of XJFACT from BDES.

The correct current is XJFACT times the initial guess for the current entered in the regional data back in AUTOMESH or LATTICE.

B.5.3.5 Permanent magnet potential initialization.

When solving permanent magnet problems with no currents (PANDIRA problem), the vector potential must be initialized by setting CON(101) = IPERM = 1 and defining a line region. The program automatically knows that IPERM is the "initializing" current in that line region. The value of IPERM is not important, so long as it is not zero. The location of the line region is also not too important.

B.5.3.6 Field calculation options.

As discussed in Sec. B.5.3.2 above, the vector and scaler potentials are assumed to be the real and imaginary parts of a complex potential $F(z)$ that is expanded as a power series in the variable $(z - z_0)$. The location of z_0 can be specified by entering new values for CON(38) = XORG and CON(39) = YORG. The default value of these parameters is zero. Note that for cylindrical symmetry, XORG = 0.0 is required.

The user can exercise some control over the accuracy of the calculation of the coefficients c_n by determining whether to use first nearest neighbor mesh points

only, or to use first and second nearest neighbor points in the determination of the coefficients. Also the weighting given to the second neighbors can be controlled. The two parameters involved are the following:

CON(48)=ISECND	= 0	Use first neighbors only
	= 1	Use first and second neighbors (the default)
CON(47)=W2ND	= 0.125	is the default weight factor for including second neighbors.

For a further discussion of the calculation of c_n's, see Sec. B.13.2.

B.5.3.7 Harmonic analysis.

Once having obtained an expression for the potential and the field as power series in the complex variable z, one can do an harmonic analysis over a region of the field to determine the size of the dipole, quadrupole, sextupole, octupole, etc. components in the field. The theory of this is summarized in Sec. B.13.3. There are six parameters that control the analysis; they are defined in Table B.5.3.III below.

Table B.5.3.III. Definition of Parameters Controlling Harmonic Analysis

Parameter	Default	Definition
CON(110)=NTERM	5	The number of harmonic coefficients to be obtained.
CON(111)=NPTC	none	The number of equidistant points on the arc of a circle with its center at the origin, at which points the vector potential is to be obtained by interpolation. Fourier analysis of the vector potential at these points yields the harmonic coefficients. NPTC should be approximately equal to the number of mesh points adjacent to the arc.
CON(112)=RINT	none	The radius of the arc used for the analysis. RINT should be less than, by at least one mesh space, the distance to the nearest interface between the given region and an iron or coil region. Choosing an arc too close to a pole face will give an erroneous result.

Table B.5.3.III. (cont.) Definition of Parameters Controlling Harmonic Analysis

Parameter	Default	Definition
CON(113)=ANGLE	none	The angular extent in degrees of the interpolation arc. The size of this arc depends on the symmetry of the magnet as specified by the ITYPE discussed above. For a symmetric quadrupole, only one-eighth of the problem is necessary, and hence ANGLE = 45 degrees is appropriate.
CON(114)=RNORM	none	The radius of the largest circle that will fit in the magnet aperture, or some other normalization radius of your choice.
CON(115)=ANGLZ	0.0	The angle in degrees from the x-axis to where integration arc begins

B.5.3.8 Over-relaxation factors.

The novice user probably will not want to make adjustments of the six over-relaxation factors unless he runs into difficulties with convergence of the problem. A discussion of the over-relaxation factors can be found in Sec. B.13.6. Table B.5.3.IV gives brief descriptions of these factors.

Table B.5.3.IV. Definitions of the Over-Relaxation Factor Parameters

Parameter	Default	Definition
CON(50)=IHDL	100000	An indicator used in POISSON only to determine the number of cycles between making quasi-integrals of $\mathbf{H} \cdot \mathbf{dl}$ around the Dirichlet boundary. Making corrections to the solution matrix based on the value of this integral sometimes speeds the convergence, particularly for non-symmetrical "II" magnets.
CON(74)=RHOPT1	1.90	(See CON(75)=RHOAIR below.)
CON(75)=RHOAIR	1.90	The over-relaxation factor in POISSON for air and interface points (and for iron points during a μ-finite-and-constant solution.)* This factor is automatically optimized during the iteration process if the initial value of RHOAIR is equal to RHOPT1.

Table B.5.3.IV. (cont.) Definitions of the Over-Relaxation Factor Parameters

Parameter	Default	Definition
CON(77)=RHOFE	1.0	The over-relaxation factor for iron points during a μ-finite-but-variable solution.*
CON(78)=RHOGAM	.08	The under-relaxation factor for the reluctivity γ during a μ-finite-but-variable solution.*
CON(80)=ISKIP	1	The number of cycles between recalculating the γ's during a μ-finite-but-variable solution.*

*See the description of CON(6)=MODE in Sec. B.5.4 below for a distinction between the μ-finite-and-constant and the μ-finite-but-variable cases.

B.5.3.9 Convergence criteria.

The problem is said to have converged if the potential has changed by less then a specified amount between tests of the convergence. There is a danger in using such a criterion for nonlinear problems (μ-finite-but-variable). The program cannot distinguish between slowly changing solutions and true minima. The potential is tested for convergence both in air regions and in the iron regions separately. One can set a separate convergence criterion for each region. The number of over-relaxation cycles between convergence tests can also be controlled. Table B.5.3.V defines the three parameters that control the convergence tests.

Table B.5.3.V. Definition of Parameters Controlling Convergence

Parameter	Default	Definition
CON(85)=EPSILA	5.0E-7	The number controling the convergence criterion for the potential solution in air and at interface points (and for iron points during a μ-finite-and-constant solution).
CON(86)=EPSILI	5.0E-7	The number controling the convergence criterion for the potential solution of iron points during a μ-finite-but-variable solution.
CON(87)=IVERG	10	The number of cycles between making the convergence tests during the iteration process. The default value of 10 should not be altered when using the option to optimize the over-relaxation factor RHOAIR=CON(75).

B.5.3.10 Output control.

One can choose or choose not to print out the (x, y) coordinates of the mesh points, to write a TAPE35 dump, and to calculate and print out the potentials and fields over a specified region of the mesh. One can also decide how frequently one prints out information on the iteration cycles. In some versions of the code one can check the remaining time in the run and write out a dump to TAPE35 5.0 seconds before the time limit. This latter feature does not work on the CRAY version because the CRAY automatically creates continuation files when a problem is terminated because of a time limit. Table B.5.3.VI. describes the parameters used in output control.

Table B.5.3.VI. Definition of Parameters used in Output Control

Parameter	Default	Definition
CON(29)=LIMTIM	0	The indicator to check the remaining time in the run. Execution will be terminated 5.0 seconds before the time limit and a TAPE35 dump will be written. Set LIMTIM=1 to exercise this option. (Note: This option does not work for the CRAY version of the code.)
CON(30)=MAXCY	100000*	The maximum number of iteration cycles. Most problems converge in a few hundred to a few thousand cycles for POISSON runs, and about 10 to 20 cycles for PANDIRA runs. It seldom makes sense to change this parameter, but if hitting a machine time limit, then set MAXCY to a lower number to get a dump.
CON(31)=IPRFQ	0	The cycle print frequency during iteration. The default value of 0 indicates that the iteration information will be printed only on the first and last cycles. Input values of IPRFQ must be integer multiples of IVERG in Table B.5.3.V above.
CON(32)=IPRINT	0	The mesh and field print option parameter PRINT=-1: Print (x, y) coordinates of mesh points from LATTICE. PRINT=0: No mesh and no field prints. PRINT=1: Print the vector potential array when CON(6)=MODE=0. PRINT= 2: Print $\|B\|$ in iron triangles. PRINT= 4: Print the (B_x, B_y) components in iron triangles.

Table B.5.3.VI. (cont.) Definition of Parameters used in Output Control

Parameter	Default	Definition
		Any combination of these options may be specified by a addition of IPRINT values. For example, IPRINT=7 prints the vector potential, $\mid B \mid$, and the components.
CON(34)=INACT	-1	An indicator to allow the user to interact with the iteration during POISSON or PANDIRA If INACT ≥ 1, the calculation is stopped at intervals and the user is asked to type: "GO", "NO", or "IN". If "GO", iteration continues; if "NO", iteration stops and final results are written; if "IN", user is asked for new values of CON's.
CON(35)=NODMP	0	The parameter controlling TAPE35 dump: NODMP=0: Write dump at end of run NODMP=1: Do not write dump
CON(42)=KMIN CON(43)=KTOP CON(44)=LMIN CON(45)=LTOP	1 KMAX 1 1	The (K,L) limits of the region in which the fields and gradients are to be calculated and printed. This print is independent of IPRINT. It prints fields only in non-iron regions even if KTOP and LTOP would include iron. An alternative way of defining the region is given below.
CON(54)=XMIN CON(55)=XMAX CON(56)=YMIN CON(57)=YMAX	0.0 0.0 0.0 0.0	The (x,y) limits of the region in which the fields and gradients are to be calculated and printed. Δx and Δy in this grid are specified by CON(43)=KTOP and CON(45)=LTOP in the following way: Δx = (XMAX−XMIN)/(KTOP−1) Δy = (YMAX−YMIN)/(LTOP−1)

*This is set to 20 for PANDIRA problems.

B.5.4 Entering Permeability Data

B.5.4.1 Introduction.

Basically there are four types of permeability inputs: 1. μ-infinite, 2. μ-finite-and-fixed, 3. μ-finite-but-variable, and 4. anisotropic-μ. Although we have labeled the types by μ, the code uses the reluctivity γ defined in general by the relation

$$\mathbf{H} = \vec{\vec{\gamma}}\,(|\mathbf{B}|) \cdot \mathbf{B}/\mu_0 + \mathbf{H}_c \tag{B.5.1}$$

The corresponding relation in electrostatics is

$$\mathbf{D} = \epsilon_0\, \vec{\vec{\kappa}}_e\,(|\mathbf{E}|) \cdot \mathbf{E} + \mathbf{D}_c. \tag{B.5.2}$$

For permanent magnets γ can be a tensor and $|H_c|$ is called the coercive force. This is consistent with the modern point of view that(**B**,**E**) are the primary quantities and (**H**,**D**) are derived from material properties. Henceforth it will be understood that the phrase dielectric constant κ_e can be substituted for the word reluctivity γ everywhere in the discussion.

In addition to the reluctivity the user has control of the stacking or fill factor for the material. The iron in most electromagnets is laminated hence part of the material in the iron region is not ferromagnetic. This is taken into account in the code by assigning a stacking factor between zero and one.

When there is more than one type of material (or stacking factor), the user must assign an identification number called MATER to the material. This number is compared with the regional number MAT in assigning properties to regions.

The next four subsections discuss the four types of input. The subsection after that will summarize the allowed input and give a flow diagram for the read statements.

B.5.4.2 μ-infinite case.

This is the default case and corresponds to CON(6) = −2. No other parameters are required. This input gives a good approximation to the field in the air region near soft iron if the field is less than about 500 gauss. It may be a useful first approximation at higher fields when used in the design of synchrotron magnets that start at low fields and go to higher fields.

B.5.4.3 μ-finite-and-constant case.

If CON(6) = MODE = −1, the code expects to read in from 1 to 4 values of

γ and stacking factor depending on the value of CON(18) = NPERM. The combination MODE = −1 and NPERM = −1, −2, −3, or −4 signals the μ-finite-and-constant case. Furthermore there is another way to enter data for the case of one fixed γ material by using just the CON's, namely, CON(6)=−1, CON(7)=STACK, CON(10)=FIXGAM, and CON(18)=NPERM=0.

B.5.4.4 μ-finite-but-variable case.

When μ is variable, it must be defined by a table. The code has one internal table corresponding to a very low carbon steel. The table is always printed out in the file OUTPOI. This case is signaled by MODE = 0. The code allows the use of the internal table plus up to 3 externally read in tables, depending on the value of NPERM = 0, 1, 2, or 3.

There is also an option that allows the user to have the internal table with up to 4 different stacking factors. This is signaled by MODE = 0 and NPERM = −1, −2, −3, or −4. Furthermore there is another way to enter data for the case of the internal table with one stacking factor. This is done using just the CON'S, namely, CON(6)=0, CON(7)=STACK, and NPERM = 0.

The table values defining the permeability may be entered in one of 3 forms: (B,γ), (B,μ), or (B,H) as specified by the parameter MTYPE, which is read in along with MATER and STACK. MTYPE = 1 implies (B,γ), which is the default; MTYPE = 2 implies (B,μ); and MTYPE = 3 implies (B,H). Each table can contain up to 50 points. If there are less than 50 points, the table must be terminated by "*s*". If MTYPE < 0, the material is a permanent magnet material.

B.5.4.5 Anisotropic μ and permanent magnet materials.

This section applies only to PANDIRA input. Anisotropic materials are characterized by an easy axis and a hard axis perpendicular to it. The values of μ are different along the two axes. The easy axis may be μ-finite-but-variable (table required), while the hard axis must be μ-finite-and-constant. Permanent magnet materials are usually anisotropic with the easy axis characterized by a μ-finite-and-constant relation, and a nonzero coercive force H_c. The code distinguishes between these cases by the value of the two parameters MTYPE and MATER. MTYPE = −1, −2, or −3 and MATER = 3, 4, or 5 implies anisotropic material; MTYPE < 0 and MATER > 5 implies permanent magnet material.

Definition of an anisotropic material requires up to 5 parameters, 4 identifying the easy axis and one defining the reluctivity along the hard axis. These are defined in Table B.5.4.I below.

Table B.5.4.I. Definition of Anisotropic Parameters[a]

Parameter	Definition
ANIN(1)=ANISO	The direction in degrees of the easy axis relative to the horizontal axis. See Fig. B.5.4.1.
ANIN(2)=GAMPER	Relative reluctivity γ perpendicular to the easy axis.
ANIN(3)=XOA ANIN(4)=YOA	The center of a circular arc for the easy axis direction. This optional data is used along with PHAXIS when the easy axis cannot be defined by ANISO above. See Fig. B.5.4.2.
ANIN(5)=PHAXIS	The angle in degrees between the radial vector R and the easy axis. See Fig. B.5.4.2. PHAXIS is usually zero or 90 degrees.

[a]These parameters must be entered in this order using the free format defined in Sec. B.3.1

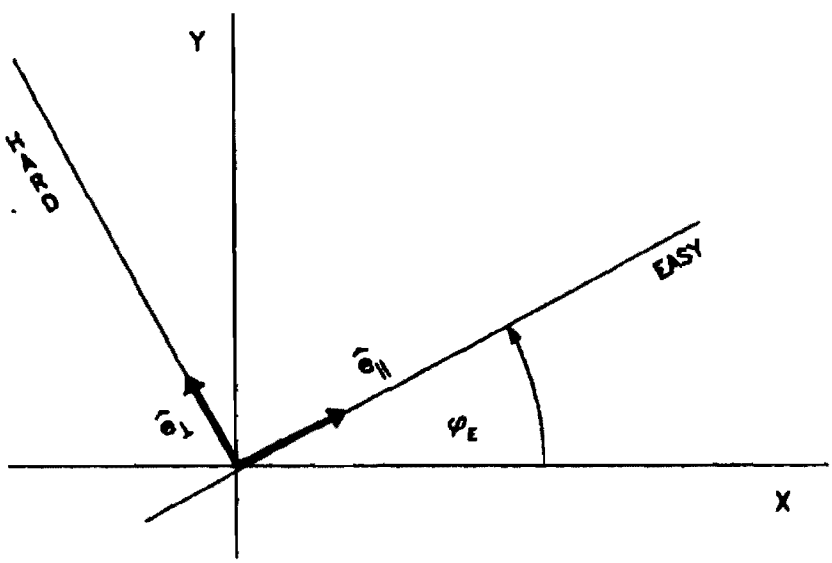


Figure B.5.4.1: Definition of angle φ_E = ANISO

Most anisotropic materials only require the first two parameters in this table. There are no natural materials where the easy axis direction changes with location in the material, but materials which approximate this behavior can be constructed with slabs of permanent magnet material, for which this option was created. If MATER ≤ 5, the easy axis permeability is specified by a table. If MATER ≥ 6 but

≤ 11, then the material is a permanent magnet and the easy axis is specified by two parameters (HCEPT, BCEPT) entered using the free format. HCEPT is the H-axis intercept in oersteds of the linear B-H relation; it is a negative number with the value of H_c. BCEPT is the B-axis intercept in gauss; it is called the residual induction B_r. This is illustrated in Fig. B.5.4.3. For permanent magnet problems with no electric currents remember to set CON(101) = IPERM = 1. (See Subsec. B.5.3.5 above.)

B.5.4.6 Summary of permeability Input.

There are a lot of options for entering permeability data. Figure B.5.1.1 above shows how CON(6)=MODE controls input. Figure B.5.4.4 shows schematically what happens when CON(6)=0 or -1, depending on the value of CON(18)=NPERM. (The coding is actually more complicated than this.) Table B.5.4.II illustrates which parameters control which options. The |NPERM| can always be less than the value used in the table, for example, 2 instead of 3 on line 3.c of the table. The variable "s" means stacking factor.

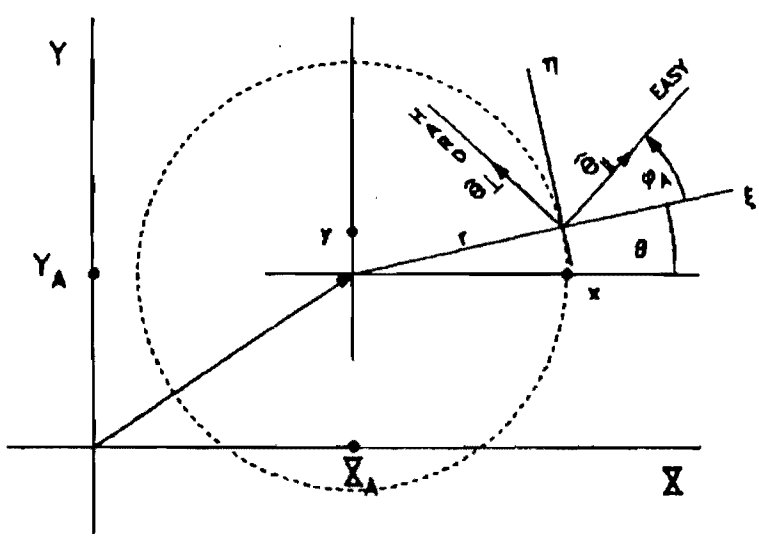


Figure B.5.4.2: Definition parameters XOA, YOA, and φ_A = PHAXIS when the easy axis direction varies over a circular arc.

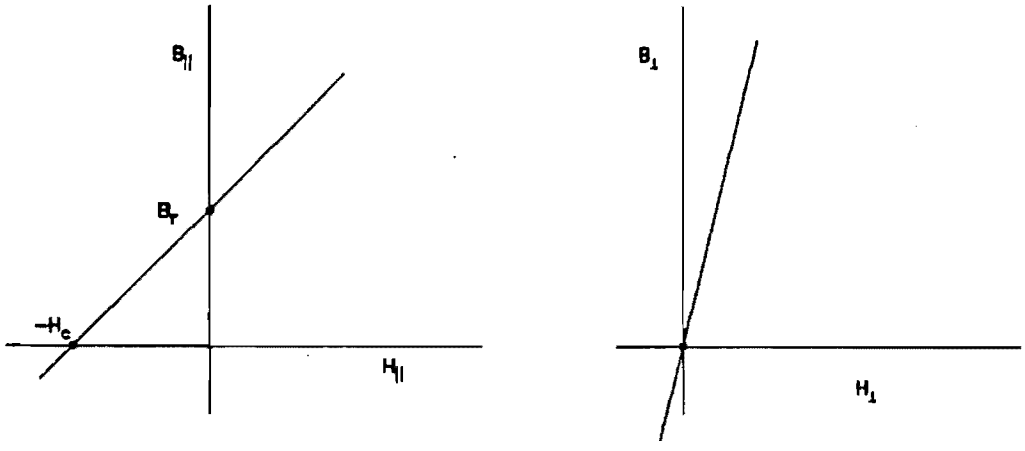


Figure B.5.4.3: Definition of parameters for permanent magnet materials: BCEPT=B_r, HCEPT=H_c and GAMPER=$H_\perp/B_\perp$.

B.5.5 Input of fixed potential points.

The ability to require that the potential be fixed at some point, along some line, or in some region can be very useful, especially in electrostatic problems. When CON(20)=INPUTA is greater than zero, that is, is equal to the number of fixed potential points, the code expects a list of points with their associated fixed potentials. The form of each line is "K L POT" where (K, L) are the logical coordinates of the point and *POT* is the fixed potential value, that is, the vector potential for magnetic problems and the scalar potential in volts for electrostatic problems. (To get (K, L) from (X, Y) set CON(32) = −1 in LATTICE and look in OUTLAT.) Please note that, if the above options are used with POISSON/PANDIRA programs received PRIOR to 9/28/86, for cylindrical coordinates (ICYLIN = CON(19) = 1), the POT values input must = $r*$ potential. For programs after 9/28/86, input just the potential values for both cylindrical or Cartesian coordinates.

≤ 11, then the material is a permanent magnet and the easy axis is specified by two parameters (HCEPT, BCEPT) entered using the free format. HCEPT is the H-axis intercept in oersteds of the linear B-H relation; it is a negative number with the value of H_c. BCEPT is the B-axis intercept in gauss; it is called the residual induction B_r. This is illustrated in Fig. B.5.4.3. For permanent magnet problems with no electric currents remember to set CON(101) = IPERM = 1. (See Subsec. B.5.3.5 above.)

B.5.4.6 Summary of permeability Input.

There are a lot of options for entering permeability data. Figure B.5.1.1 above shows how CON(6)=MODE controls input. Figure B.5.4.4 shows schematically what happens when CON(6)=0 or -1, depending on the value of CON(18)=NPERM. (The coding is actually more complicated than this.) Table B.5.4.11 illustrates which parameters control which options. The |NPERM| can always be less than the value used in the table, for example, 2 instead of 3 on line 3.c of the table. The variable "s" means stacking factor.

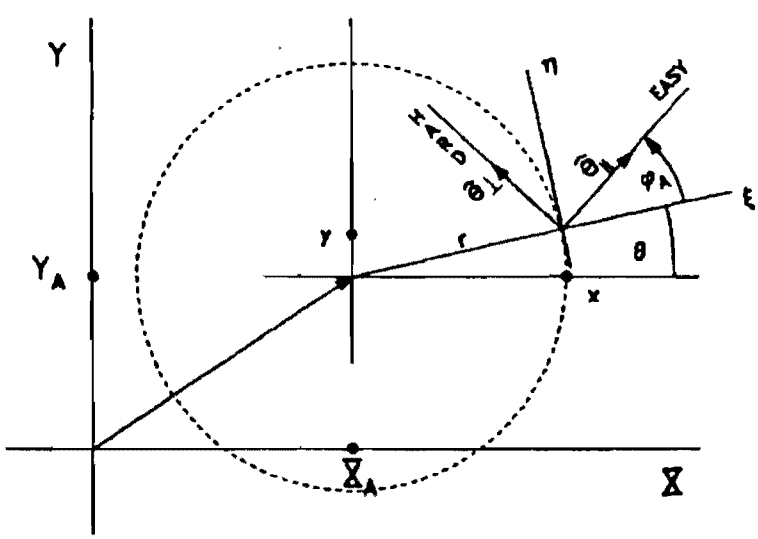


Figure B.5.4.2: Definition parameters XOA, YOA, and φ_A = PHAXIS when the easy axis direction varies over a circular arc.

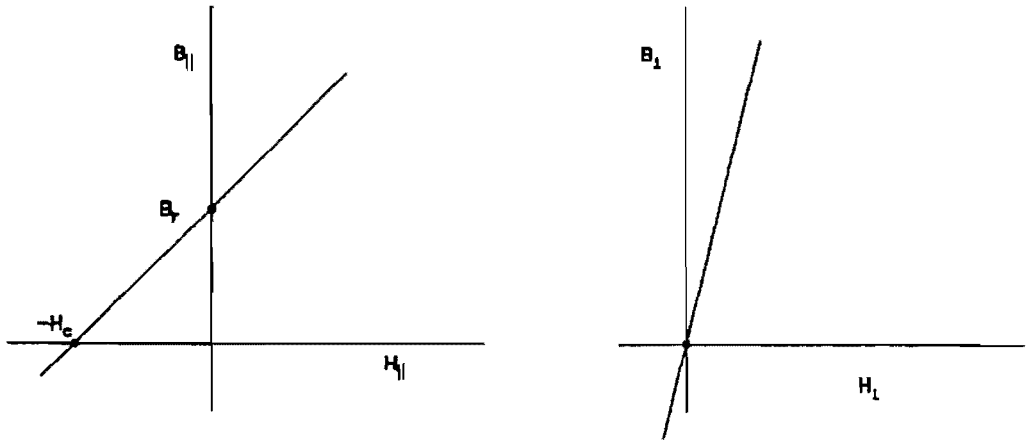


Figure B.5.4.3: Definition of parameters for permanent magnet materials: BCEPT=B_r, HCEPT=H_c and GAMPER=$H_\perp/B_\perp$.

B.5.5 Input of fixed potential points.

The ability to require that the potential be fixed at some point, along some line, or in some region can be very useful, especially in electrostatic problems. When CON(20)=INPUTA is greater than zero, that is, is equal to the number of fixed potential points, the code expects a list of points with their associated fixed potentials. The form of each line is "K L POT" where (K, L) are the logical coordinates of the point and *POT* is the fixed potential value, that is, the vector potential for magnetic problems and the scalar potential in volts for electrostatic problems. (To get (K, L) from (X, Y) set CON(32) = −1 in LATTICE and look in OUTLAT.) Please note that, if the above options are used with POISSON/PANDIRA programs received PRIOR to 9/28/86, for cylindrical coordinates (ICYLIN = CON(19) = 1), the POT values input must = $r*$ potential. For programs after 9/28/86, input just the potential values for both cylindrical or Cartesian coordinates.

TABLE B.5.4.II. Examples of Permeability Options

ISOTROPIC MATERIALS (POISSON/PANDIRA)	CON(6) MODE	CON(7) STACK	CON(10) FIXGAM	CON(18) NPERM	MATER	STACK[a]	MTYPE	ANIN(I) I=1,5	HCEPT BCEPT
1. $\mu = \infty$ (default)	−2	-[b]	-	-	-	-	-	-	-
2. μ finite & constant									
a) 1 set, using CON's	−1	s	7	0	-	-	-	-	-
b) 4 sets	−1	-	-	−4	2,3,4,5	4 value	-	-	-
3. μ finite but variable									
a) int. tab., 1 stack fac.	0	s	-	0	-	-	-	-	-
b) int. tab., 4 stack fac.	0	-	-	−4	2,3,4,5	4 value	-	-	-
c) int. tab. + 3 ext. tabs	0	s	-	3	3,4,5	3 value	3 val[c] (1,2,3)	-	-
d) int. tab. + 3 μ const. matls.	−1	s	-	−3	3,4,5	3 value	-	-	-
ANISOTROPIC MATERIALS (PANDIRA ONLY)									
1. not a permanent magnet									
a) (int. tab + 3 ext. tab-easy)	0	s	-	3	3,4,5	3 value	3 val[d]	3 set	-
2. Permanent magnets									
a) int. tab. +6 perm. mag.	0	s	-	6	6,7,8 9,10,11	6 value	6 val[e]	6 sets	6 sets
b) int. tab. + 3 ext. tab. + 6 perm. mag.	0	s	-	9	3–11	9 value	3 val+ 6 val[f]	9 sets	6 sets

[a]This stacking factors is not the same as CON(7), which only affects the internal table.
[b]The dash means that this parameter is not used and hence its value is not significant.
[c]Any 3 numbers (depending on input table type) from set (1, 2, 3).
[d]Any 3 numbers (depending on input table type) from the set (−1, −2, −3).
[e]Any 6 numbers as long as they are negative; the value of the number is not important.
[f]Any 3 numbers from the set(−1, −2, −3) and any 6 numbers <0. For MATER ≥ 6 the negative value of MTYPE is not significant.

B.5.6 Input for current filaments.

The main use of this feature is to include small trim windings in magnet problems. When CON(49)=NFIL is greater than zero, that is, is equal to the number of lines of current filament data to be entered, the code expects a list of points and corresponding current values. The form of each line is "K L CFIL" where (K, L) are the logical coordinates of the point and CFIL is the current in amperes.

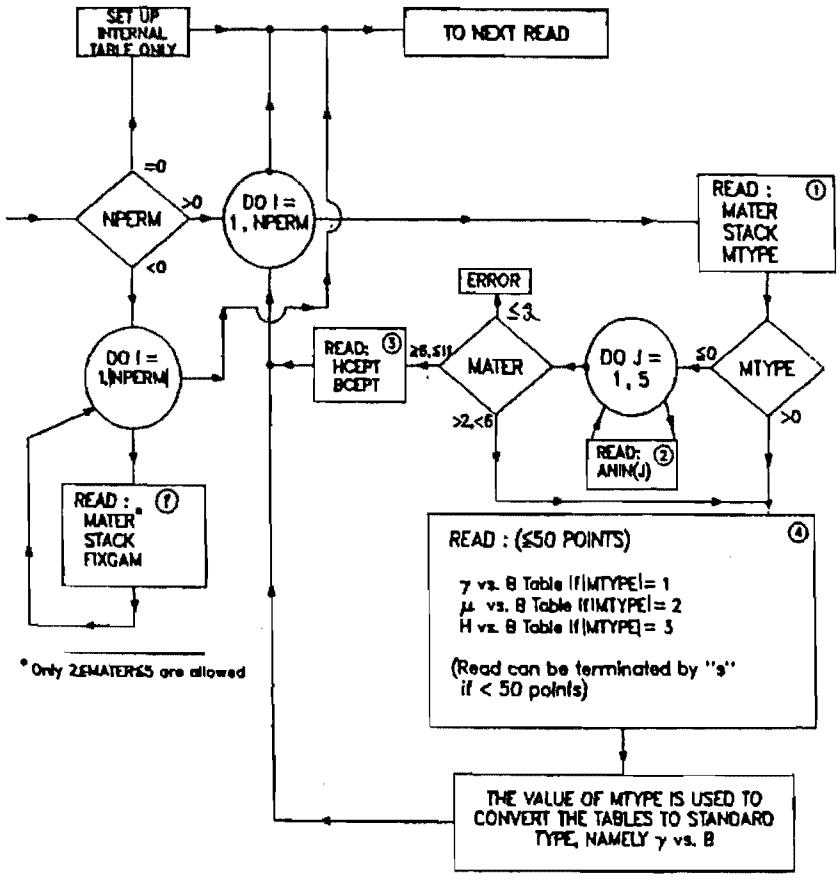


Fig. B.5.4.4: Flow diagram for entering permeability data when MODE=0 or −1.

Chapter B.6

Output of POISSON and PANDIRA

The information put out by POISSON and PANDIRA is almost identical, the principle difference being in the print of the iteration history. Because the two codes solve problems by different methods, different information on the iteration cycles is printed out.

Both codes write solution information to TAPE35 that can be used to plot flux lines via TEKPLOT. Both codes write information to output files as well as to the terminal. Examples of terminal output can be found in examples in Chap. B.12 below. The major portion of the information is written to the files OUTPOI for POISSON and OUTPAN for PANDIRA.

The material in these output files is of three types: 1. information used to solve the problem, 2. information on the iteration history, and 3. information on the solution. Information used to solve the problem includes a list of the CON values at the beginning of the run, and tables of B^2, γ, μ, and H as a function of B for the various magnetic materials appearing in the problem. The internal γ vs B table is printed out no matter whether it is used of not. Stacking factors are also printed out. The iteration history is a recapitulation of the terminal output, which is described in the examples of Chap. B.12 below. The amount and type of solution information is controlled by CON(32)=IPRINT as described in the previous section. Figure B.6.1 shows a portion of a potential map for the quadrupole problem discussed in Chap. B.3. The abscissa is the logical coordinate K and the ordinate is the logical coordinate L. Each logical point (K, L) has associated with it two values of the potential A (V for electrostatic problems). These are distinguished on the map by "u" and "l." The value A_u is the potential found in the upper triangle associated with the point and the value A_l is the potential in the lower triangle. Figure B.6.2 shows the relation between the point (K, L) and its upper and lower triangles. The association shown in the figure is unique in the sense that each triangle is associated with one and only one point in the mesh. Maps of $|B|$, B_x, and B_y are printed in a similar format.

dump number 1 has been written on tape35.
1 the following is a map of the potential array a, cycle 150

0 1

k . . . . . .

	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80
88 u																				
l																				
87 u																0.00				
l															0.27	0.19				
86 u															0.27	0.16	0.17			
l															0.42	0.34	0.31	0.42		
85 u														0.43	0.32	0.27	0.30	0.42		
l														0.56	0.49	0.45	0.48	0.55		
84 u														0.58	0.47	0.42	0.42	0.47	0.57	
l													0.85	0.71	0.64	0.60	0.59	0.62	0.68	
83 u												0.85	0.72	0.63	0.56	0.64	0.56	0.62	0.71	
l												1.01	0.90	0.81	0.75	0.72	0.73	0.76	0.82	0.94
82 u												1.03	0.90	0.79	0.72	0.68	0.68	0.71	0.76	0.83
l												1.17	1.06	0.98	0.91	0.87	0.86	0.87	0.92	0.99
81 u											1.20	1.07	0.97	0.89	0.83	0.81	0.82	0.86	0.92	1.00
l										1.51	1.34	1.24	1.15	1.08	1.03	1.00	1.00	1.02	1.06	1.13
80 u										1.51	1.35	1.24	1.14	1.06	1.00	0.96	0.95	0.97	1.01	1.07
l										1.68	1.55	1.43	1.33	1.25	1.19	1.15	1.14	1.14	1.17	1.21
79 u									1.71	1.56	1.43	1.32	1.23	1.16	1.12	1.10	1.10	1.12	1.16	1.22
l								2.06	1.86	1.74	1.62	1.52	1.43	1.36	1.31	1.29	1.28	1.29	1.31	1.35
78 u								2.06	1.88	1.75	1.62	1.51	1.41	1.34	1.28	1.25	1.24	1.24	1.27	1.31
l								2.25	2.10	1.95	1.82	1.71	1.62	1.54	1.49	1.45	1.42	1.42	1.43	1.46
77 u							2.28	2.11	1.96	1.82	1.71	1.60	1.52	1.46	1.41	1.39	1.38	1.39	1.42	1.46
l							2.45	2.31	2.16	2.03	1.92	1.82	1.73	1.66	1.61	1.58	1.56	1.56	1.57	1.60
76 u							2.50	2.32	2.17	2.03	1.91	1.80	1.71	1.64	1.58	1.55	1.53	1.53	1.54	1.57
l						2.90	2.67	2.52	2.38	2.25	2.13	2.02	1.93	1.85	1.79	1.75	1.72	1.70	1.70	1.72
75 u					2.90	2.69	2.54	2.39	2.25	2.12	2.01	1.91	1.83	1.76	1.72	1.69	1.67	1.67	1.69	1.71
l					3.12	2.94	2.77	2.61	2.47	2.35	2.23	2.13	2.05	1.98	1.92	1.88	1.86	1.85	1.85	1.86
74 u					3.16	2.96	2.78	2.62	2.47	2.34	2.22	2.11	2.02	1.95	1.89	1.85	1.83	1.82	1.82	1.83
l					3.36	3.19	3.02	2.86	2.71	2.58	2.45	2.35	2.25	2.17	2.11	2.06	2.02	2.00	1.99	1.99
73 u				3.42	3.21	3.03	2.86	2.71	2.56	2.44	2.33	2.23	2.15	2.08	2.03	1.99	1.97	1.96	1.97	1.98

Fig. B.6.1: Portion of the printed map giving magnetic field values in kilogauss in the iron region of a quadrupole magnet. (See Section B.3.3.)

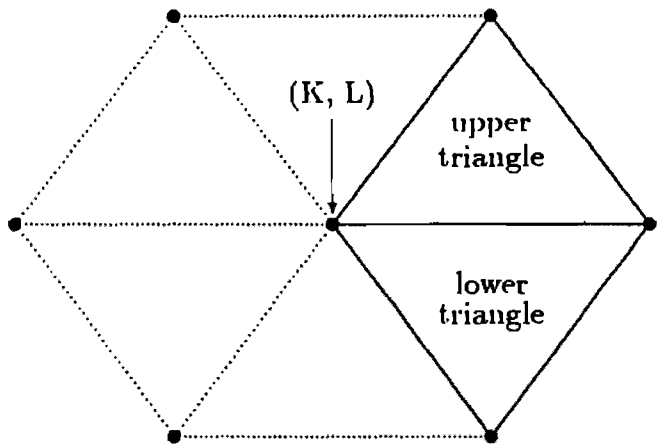


Fig. B.6.2: Each point (K, L) in the logical mesh has associated with it an upper and lower triangle.

In addition to these maps, the user may have the code print tables that assign values of potential and derived field quantities as a function of physical coordinates. The values are obtained by a least-squares interpolation scheme. The tables also give residuals (`afit`) for the physical point (x, y). The quantity `afit` is a measure of how well the polynomal fit to the potential matches the values of the potential at the (K, L) point. The polynomial is used to calculate the derivatives of the potential. The user can select the region covered in these tables by using CON's 42 through 45 or CON's 54 through 57 as described in Subsec. B.5.3.10 above. Figure B.6.3 shows a part of one of these tables.

When harmonic analysis is requested, by entering values for CON's 111 through 115, the code prints the results of the analysis in the output file. Figure B.6.4 shows an example table. The first portion is a table giving the coordinates and interpolated vector-potential values used in the analysis. The next section gives the real and imaginary parts of the coefficients in the expansion of the vector potential in harmonic polynomials. The absolute value of the coefficients is also printed. The third section of the harmonic analysis gives the real, imaginary and absolute values of the coefficients for the magnetic field expansion in harmonic polynomials. There is a direct correspondence between the harmonic polynomials and the multipole content of the field. For a more complete discussion see Sec. B.13.3.

Finally, for problems with special fixed vector potential points, there is a table giving the total current required to keep the potential fixed at these points.

```
least squares edit of problem.  cycle 9
Bymm qua symmetry type
Ostored energy = -5.0983e-09 joules / meter or radian
xjfact= 1.000000
```

0	k	1	a(vector)	x	y	bx(gauss)	by(gauss)	bt(gauss)	dby/dy(gauss/cm)	dby/dy(gauss/cm)	sfit
0	1	1	0.000000e+00	0.00000	0.00000	0.000	0.000	0.000	0.0000e+00	-2.7676e+03	2.3e-01
0	2	1	6.154677e+00	0.21269	0.00000	0.000	-589.685	589.685	0.0000e+00	-2.7469e+03	-1.1e+00
0	3	1	2.491422e+02	0.42537	0.00000	0.000	-1168.687	1168.687	0.0000e+00	-2.7443e+03	-2.3e-01
0	4	1	5.593174e+02	0.63806	0.00000	0.000	-1747.257	1747.257	0.0000e+00	-2.7461e+03	3.6e-01
0	5	1	9.926362e+02	0.85075	0.00000	0.000	-2330.915	2330.915	0.0000e+00	-2.7533e+03	9.5e-02
0	6	1	1.550469e+03	1.06343	0.00000	0.000	-2915.366	2915.366	0.0000e+00	-2.7476e+03	-2.8e-02
0	7	1	2.232639e+03	1.27612	0.00000	0.000	-3498.966	3498.966	0.0000e+00	-2.7409e+03	-3.4e-02
0	8	1	3.038839e+03	1.48881	0.00000	0.000	-4081.752	4081.752	0.0000e+00	-2.7386e+03	-5.3e-03
0	9	1	3.968923e+03	1.70149	0.00000	0.000	-4664.016	4664.016	0.0000e+00	-2.7369e+03	2.2e-02
0	10	1	5.022777e+03	1.91418	0.00000	0.000	-5245.626	5245.626	0.0000e+00	-2.7336e+03	4.4e-02
0	11	1	6.200222e+03	2.12687	0.00000	0.000	-5826.148	5826.148	0.0000e+00	-2.7274e+03	6.1e-02
0	12	1	7.500948e+03	2.33955	0.00000	0.000	-6404.752	6404.752	0.0000e+00	-2.7159e+03	7.3e-02
0	13	1	8.924402e+03	2.55224	0.00000	0.000	-6979.965	6979.965	0.0000e+00	-2.6952e+03	7.9e-02
0	14	1	1.046961e+04	2.76493	0.00000	0.000	-7549.193	7549.193	0.0000e+00	-2.6582e+03	7.4e-02
0	15	1	1.213486e+04	2.97761	0.00000	0.000	-8107.770	8107.770	0.0000e+00	-2.5916e+03	5.6e-02
0	16	1	1.391705e+04	3.19030	0.00000	0.000	-8647.202	8647.202	0.0000e+00	-2.4716e+03	2.0e-02
0	17	1	1.581066e+04	3.40299	0.00000	0.000	-9152.279	9152.279	0.0000e+00	-2.2578e+03	-3.5e-02
0	18	1	1.780585e+04	3.61567	0.00000	0.000	-9597.004	9597.004	0.0000e+00	-1.8894e+03	-9.7e-02
0	19	1	1.988579e+04	3.82836	0.00000	0.000	-9940.965	9940.965	0.0000e+00	-1.2974e+03	-1.2e-01
0	20	1	2.202365e+04	4.04104	0.00000	0.000	-10132.459	10132.459	0.0000e+00	-4.5903e+02	-7.7e-02
0	21	1	2.418188e+04	4.25373	0.00000	0.000	-10126.693	10126.693	0.0000e+00	5.2599e+02	2.7e-02
0	22	1	2.631657e+04	4.46642	0.00000	0.000	-9913.908	9913.908	0.0000e+00	1.4403e+03	1.1e-01
0	23	1	2.838703e+04	4.67910	0.00000	0.000	-9532.435	9532.435	0.0000e+00	2.0818e+03	1.2e-01
0	24	1	3.036432e+04	4.89179	0.00000	0.000	-9049.896	9049.896	0.0000e+00	2.3942e+03	7.1e-02
0	25	1	3.223408e+04	5.10448	0.00000	0.000	-8530.271	8530.271	0.0000e+00	2.4533e+03	2.9e-02
0	26	1	3.399330e+04	5.31716	0.00000	0.000	-8015.602	8015.602	0.0000e+00	2.3687e+03	1.9e-02
0	27	1	3.564557e+04	5.52985	0.00000	0.000	-7526.496	7526.496	0.0000e+00	2.2257e+03	4.4e-02
0	28	1	3.719717e+04	5.74254	0.00000	0.000	-7068.718	7068.718	0.0000e+00	2.0793e+03	1.2e-01
0	29	1	3.865448e+04	5.95522	0.00000	0.000	-6638.765	6638.765	0.0000e+00	1.9624e+03	2.1e-01
0	30	1	4.002263e+04	6.16791	0.00000	0.000	-6229.769	6229.769	0.0000e+00	1.8802e+03	2.5e-01
0	31	1	4.130556e+04	6.38060	0.00000	0.000	-5834.180	5834.180	0.0000e+00	1.8386e+03	3.4e-01
0	32	1	4.250506e+04	6.59328	0.00000	0.000	-5457.264	5457.264	0.0000e+00	1.7227e+03	-4.6e-01
0	33	1	4.362857e+04	6.80597	0.00000	0.000	-5116.341	5116.341	0.0000e+00	1.5341e+03	-9.8e-01
0	34	1	4.468414e+04	7.01866	0.00000	0.000	-4807.112	4807.112	0.0000e+00	1.4090e+03	-4.5e-01
0	35	1	4.567601e+04	7.23134	0.00000	0.000	-4521.072	4521.072	0.0000e+00	1.3083e+03	-1.9e-01
0	36	1	4.660867e+04	7.44403	0.00000	0.000	-4249.155	4249.155	0.0000e+00	1.2647e+03	-9.6e-03
0	37	1	4.748392e+04	7.65627	0.00000	0.000	-3980.541	3980.541	0.0000e+00	1.2606e+03	1.3e-01
0	38	1	4.830205e+04	7.86940	0.00000	0.000	-3714.201	3714.201	0.0000e+00	1.2301e+03	1.1e-01

Fig. B.6.3: Portion of least squares fit table from a PANDIRA run.

```
 1 harmonic analysis.
 0 integration radius = 1.86000.
 0 table for integrated points.
0
```

n	angle	x coord	y coord	kf	lf	vec.pot.
1	0.0000	1.8600	0.0000	10	1	4.74254e+03
2	4.5000	1.8543	0.1459	10	2	4.68423e+03
3	9.0000	1.8371	0.2910	10	2	4.51071e+03
4	13.5000	1.8086	0.4342	9	3	4.22621e+03
5	18.0000	1.7690	0.5748	10	4	3.83765e+03
6	22.5000	1.7184	0.7118	9	5	3.35455e+03
7	27.0000	1.6573	0.8444	9	6	2.78877e+03
8	31.5000	1.5859	0.9718	9	6	2.15424e+03
9	36.0000	1.5048	1.0933	8	7	1.46655e+03
10	40.5000	1.4144	1.2080	8	7	7.42774e+02
11	45.0000	1.3152	1.3152	8	8	3.54835e-01

```
 1table for vector potential coefficients
 0normalization radius = 2.92000
0        a(x,y) = re( sum (an + 1 bn) * (z/r) **n )
```

n	an	bn	abs(cn)
2	1.1690e+04	0.0000e+00	1.1690e+04
6	-1.2543e+01	0.0000e+00	1.2543e+01
10	9.3097e+00	0.0000e+00	9.3097e+00
14	-5.7175e+01	0.0000e+00	5.7175e+01
18	2.4653e+02	0.0000e+00	2.4653e+02

```
 1table for field coefficients
 0normalization radius = 2.92000
0        (bx - by) = 1 = sum n*(an + 1 bn)/r = (z/r)**(n-1)
```

n	n(an)/r	n(bn)/r	abs(n(cn)/r)
2	8.0070e+03	0.0000e+00	8.0070e+03
6	-2.5774e+01	0.0000e+00	2.5774e+01
10	3.1882e+01	0.0000e+00	3.1882e+01
14	-2.7413e+02	0.0000e+00	2.7413e+02
18	1.5197e+03	0.0000e+00	1.5197e+03

Figure B.6.4: Example of the harmonic analysis information written to the output file.

Chapter B.7

Input and Output for TEKPLOT

B.7.1 Input to TEKPLOT

TEKPLOT will plot the physical boundaries and mesh resulting from a LATTICE output. It will also plot the equipotentials from POISSON, PANDIRA, and MIRT output. More than one plot can be made in the same run by repeating the first two input data groups. The structure of the input is shown in Fig. B.7.1.1.

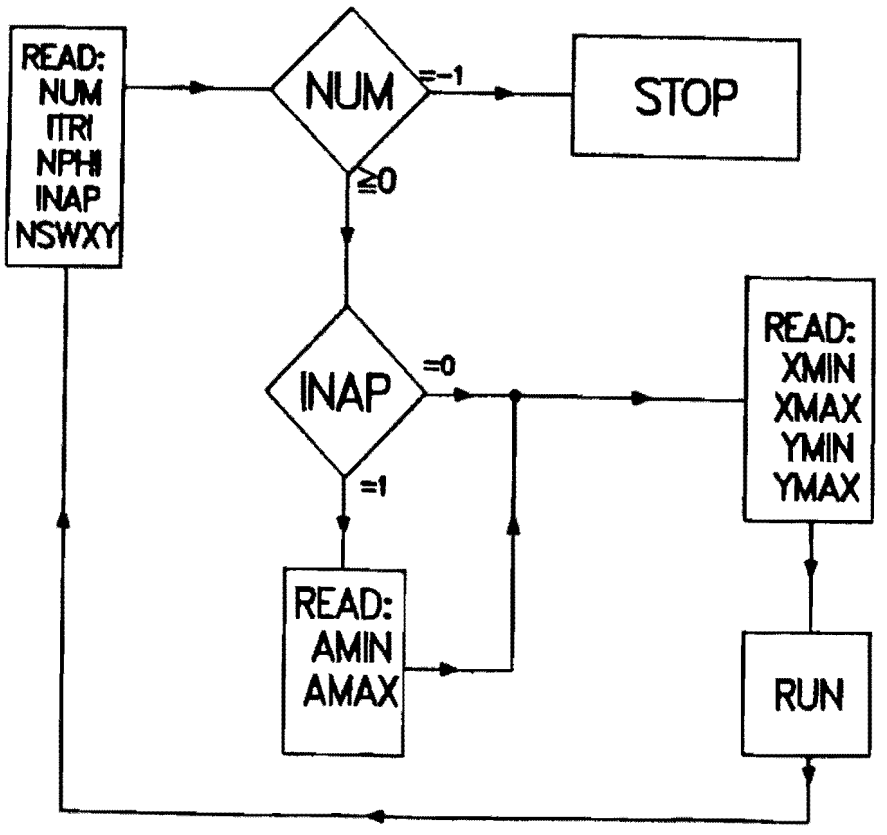


Figure B.7.1.1. Flow diagram for Read Statements in TEKPLOT.

This program uses the special free format described in Sec. B.3.1. The meaning of the parameters is given in Table B.7.1.I.

Table B.7.1.I Input Parameters to TEKPLOT.

Name	Default	Description
NUM	0	The TAPE35 "dump" number on which the (X,Y) coordinates of the mesh, and (if NUM > 0) the field values have been written.
ITRI	0	An indicator to specify whether the triangular mesh is to be plotted or only the physical boundary lines of regions. ITRI = 0 means do not plot the triangular mesh; ITRI = 1 means plot the triangular mesh.
NPHI	0	The number of equipotential lines to be plotted. The program does not plot lines for the smallest and largest potential values, one of which is usually just a point. For magnetostatic problems in two dimensions, the equipotential lines for the vector potential are magnetic fields in cartesian geometry. In cylindrical geometry the program plots "flux surfaces," which are $r * A(r, z)$ = constant. For most problems a good number for NPHI is between 20 and 50.
INAP	0	An indicator for an additional Read statement. INAP = 0 means do not read AMIN and AMAX, = 1 means read (on the next data line) the minimum and maximum values (AMIN and AMAX) of the equipotential lines to be plotted. The values plotted are $(AMAX - \Delta), (AMAX - 2 * \Delta), \ldots, (AMIN + \Delta)$, where Δ is (AMAX − AMIN)/(NPHI + 1).
NSWXY	0[a]	An indicator allowing an interchange of the X and Y axes NSWXY = 0 means no interchange; NSWXY = 1 means interchange.
XMIN XMAX YMIN YMAX	XMIN XMAX YMIN YMAX	The limits of the plot, which may be any part of the problem rectangle. The variables XMIN, XMAX, YMIN and YMAX should not be confused with variables of the same name that are entered in AUTOMESH and determine the size of the problem rectangle, however, if allowed to default, they will take on the values defined in AUTOMESH.

[a]Or last input value.

```
tekplot
 ?type input data- num, itri, nphi, inap, nswxy,
?  s
 input data
 num=   0   itri= 0    nphi=   0    inap= 0    nswxy= 0

 plotting prob. name =  full size cavity                    cycle = 0

 ?type input data- xmin, xmax, ymin, ymax
?  80.  125.  0.  25.
 input data
 xmin=  80.000  xmax= 125.000  ymin=   0.000  ymax=  25.000

 ?type go or no
?  go
```

Figure B.7.1.2: An example of interactive input to TEKPLOT.

After making the plot, TEKPLOT waits for a carriage return before prompting the user for more input. Upon receiving the carriage return, TEKPLOT asks for a dump number with accompanying input. To terminate the run the user enters -1S for the dump number.

An example of TEKPLOT input is given in Fig. B.7.1.2.

B.7.2 Output of TEKPLOT

In addition to providing the plots described above, TEKPLOT also makes an output file named OUTTEK, which contains a list of the contour values that were plotted. TEKPLOT will only plot closed regions.

B.7.3 System-dependent Plot Routines in TEKPLOT

TEKPLOT uses PLOT10 commands . If PLOT10 is not available at the user's installation, then the user will have to go into the FORTRAN code and substitute commands from his own graphics system. The calls to PLOT10 and their functions are listed at the beginning of the source code to facilitate substitutions.

Chapter B.8

Input and Output for FORCE

Presently the version of FORCE that we have is not compatible with the standard versions of POISSON, PANDIRA, and TEKPLOT. In the near future we will have it working, and at that time we will send out the documentation for FORCE and a worked example.

Chapter B.9

Input and Output for MIRT

B.9.1 Introduction

MIRT is a nonlinear optimization program that uses POISSON as a function generator. It optimizes a set of parameters defining a magnet by minimizing a weighted sum of squares of the deviations between the desired and the actual performance of a magnet, subject to restraining conditions. If $\{p_i, i = 1, ...n\}$ is a set of parameters, and $\{s_i, i = 1, ..., m > n\}$ is a set of performance numbers desired, for example, values of the magnetic field at given locations, then the sum to be minimized is

$$S = \sum_{i=1}^{m} w_i(s_i - \underline{s}_i)^2 + \sum_{i=1}^{l} v_i(p_i - \underline{p}_i)^2 \qquad \text{(B.9.1.1)}$$

where the w_i's are weights chosen either so that each term in the sum is approximately the same size, or in some other user-defined fashion. Choosing weights v_i and estimated values of $\left\{\underline{p}_i, i = 1, ..., l < n\right\}$ for some parameters, applies what are called soft restraints. The numbers s_i are calculated by POISSON. Hard restraints can also be applied. Hard restraints fix some values $\{r_i, i = 1, ..., h < n\}$ of performance numbers absolutely. In order to minimize S, one must calculate the change of s_i's as a function of the p_j's. This is contained in the "coefficient matrix"

$$M_{ij} = ds_i/dp_j \qquad \text{(B.9.1.2)}$$

which will be referred to later.

The principle use of MIRT is to trim the poleface of dipole magnets to produce a more uniform field in the midplane of the gap. MIRT has also been used to optimize a quadrupole magnet. However, to do so, the quadrupole was first transformed into a dipole through a conformal transformation. The dipole is then optimized and then transformed back to a quadrupole using the inverse conformal transformation. This use of conformal transformations is described in Sec. B.13.4.

Optimization means least squares minimization of the difference between the

field distribution desired by the user and the field produced by a given set of magnet parameters. The parameters that can be changed to produce the optimization are: 1. the poleface profile, 2. the current and current density in the coils, and 3. the coil position. The parameters describing the poleface profile have been discussed in Sec. B.1.6. The next section describes the input necessary to control the optimization, to define the optimal field, and to specify the magnet parameters to be adjusted. The final section describes the output produced by the code. A detailed example is given in Sec. B.12.4 below. (Note: Sec. B.12.4 has not been written yet.)

B.9.2 Input to MIRT

The input data for MIRT can be divided into seven groups. The first group is the variables controlling the overall optimization. These variables are called the "optimization constants." Most of them have default values that the user will seldom have need to change. The second group defines the "fitted points." Each fitted point is defined by a field value, a location, and a weight to be used in the least squares minimization. The third group specifies the parameters that will be adjusted to achieve the minimization of differences between the optimum field and the actual field. The fourth group is the set of numbers required to implement the soft restraints on the parameters. The fifth group controls the regeneration of the mesh when bumps are applied to polefaces and coils. The sixth group contains input for the coefficient matrix, if it has been saved from a previous run. The seventh group is identical to the normal POISSON input data as described in Chap. B.5 above. These seven groups are discussed in seven subsections below. All input is entered using the special free format discussed in Sec. B.3.1.

B.9.2.1 Optimization constants.

Nonlinear optimization requires special care. Predictions of desired parameter changes based on the linear coefficient matrix M_{ij} may not be valid. The program takes the following precautions. The change ΔT_i made by the program is the change suggested by the least squares process, multiplied by a relaxation parameter RLX. The relaxation parameter is modified from iteration to iteration depending on how nonlinear the problem is. The degree of nonlinearity is tested after every iteration with the help of a test summation

$$T^{last} = \sum_{i=1}^{m} w_i(s_i^{last} - \underline{s}_i)^2 + \sum_{i=1}^{l} v_i(p_i^{last} - \underline{p}_i)^2 + \sum_{i=1}^{h} u_i(r_i^{last} - \underline{r}_i)^2 \qquad \text{(B.9.2.1)}$$

where the u_i's are chosen so that the contribution from the r_i's is comparable to that from the s_i's. The change in T after each step

$$\Delta T = T^{last} - T^{previous}, \qquad \text{(B.9.2.2)}$$

is compared with the change ΔT_{lin} that would be expected if the problem were linear. It can be shown that the change ΔT_{lin} depends on the coefficient matrix

and on the relaxation parameter RLX. See Sec. B.13.6. According to the value of a function

$$TEST = (\Delta T/\Delta T_{lin} - 1), \qquad \text{(B.9.2.3)}$$

adjustment of RLX is made. This adjustment will be described below.

The program stops when one of three conditions is fulfilled: 1. S given by Eq.(B.9.1.1) or T given by Eq.(B.9.2.1) is less than the optimization parameter SRSUM described below, 2. RLX is greater than 1/4 and the difference between the last calculated values of either S or T is less than the optimization parameter PCNT, or 3. the recommended value of RLX is less than the optimization parameter RLXS.

Table B.9.2.I gives the meaning and default values of these optimization parameters, which are stored in an array called PCON.

Table B.9.2.I. Definition of the PCON array.

Constant	Default		Definition
PCON(1)=LOOP	0	If LOOP >0,	It is the maximum number of cycles allowed.
		=0,	The code runs POISSON once.
		<0,	The code only produces the coefficient matrix discussed below.
PCON(2)=MATRIX	0	If MATRIX= 1,	Save the coefficient matrix.
		= 2,	Read in the coeff. matrix.
		= 3,	Read and save coeff. matrix. The coeff. matrix is saved on file TAPE20.
PCON(3)=IPUNCH	0	If IPUNCH > 0,	Save the coordinates of the final optimized poleface boundary on the file TAPE10.
PCON(4)=MPRINT	1	If MPRINT= 1,	Write the coeff. matrix to the file OUTMIR.
		≠ 1,	Do not write coeff. matrix.
PCON(5)=MUSE	1		The number of solutions with each coefficient matrix. The coefficient matrix can be used for several iterations if desired.

Table B.9.2.I. (continued) Definition of the PCON array.

Constant	Default	Definition
PCON(6)=RLX	0.5	The starting relaxation factor for the parameters.
PCON(7)=RLXS	1/128	The minimum value that RLX may assume. The optimization will be terminated if a smaller relaxation factor is recommended.
PCON(8)=TEST1	0.1	A parameter that determines the options for changing RLX. The code calculates an auxiliary quantity TEST and compares (See Eq. B.9.2.3) it with TEST1 before changing RLX. If S is the sum of the weighted squares of differences at the fitted points, then TEST is the ratio of the improvement in T to the "as if" linear improvement in T minus 1.0. The prescription for changing RLX is as follows: If |TEST| > 4.*TEST1, halve RLX and if TEST>0., proceed; but if TEST<0, reject the previous solution. If 4.*TEST1> |TEST| >2.*TEST1, halve RLX and proceed. If 2.*TEST1> |TEST| > TEST1, proceed. If |TEST| <TEST1, double RLX and proceed.
PCON(9)=SRSUM	1.E-20	The optimization has converged when S, the sum of the weighted squares of differences, is less than SRSUM.
PCON(10)=PCNT	0.1	An alternative convergence criterion. When RLX > 0.25 and |S − AFS| < AFS*PCNT, where AFS is the anticipated final sum, then the code also considers the optimization complete.
PCON(11)=FEMAX	0.5	The maximum height or depth of bumps added to the poleface must be less than FEMAX times the starting half gap between the pole faces.

The first constant, LOOP, is usually the only one that has to be changed from its default value. A typical value for LOOP is 10.

B.9.2.2 Fitted points data.

There are three types of data entered here: 1. the number of points, 2. the fields and weights, and 3. the coordinates of the points. The first line of data consists of two numbers, NAIR and NIRN. NAIR is the number of "mu-infinite" fitted points; NIRN is the number of "mu-finite" fitted points. From these numbers the code calculates the number NFIT = NAIR + NIRN. The intention of having two types of data points is to allow one to optimize a given magnet simultaneously for both low field and high field operation. This is useful for instance in the design of dipole and quadrupole magnets for synchrotrons. At low fields, the permeability of the iron is essentially constant and nearly infinite. At high fields, the effect of the variability of the permeability is important. The importance of this type of optimization is illustrated in reference B.14.2.

Table B.9.2.II summarizes the second type of data, which defines the field and its weights.

Table B.9.2.II. Data defining the field at fitted points.

Variable	Definition
FFIT(I)	The desired value of the selected field quantity at point I. ITFIT selects the field quantity.
WFIT(I)	The weight factor at point I. This number depends on the type of points. If some points are fields and others are field gradients, then the weights must compensate for the difference in physical units as well as the intrinsic relative importance of the points. If WFIT < 0, the code treats the point as having a hard restraint. The code must fit this point exactly.
ITFIT(I)	This integer selects the type of quantity that can be fixed as follows: ITFIT $= 1$, fit B_y in gauss at point I. fit B_z for cylindrical problems. $= 2$, fit B_x in gauss at point I. fit B_r for cylindrical problems. $= 3$, fit dB_y/dx (gauss/cm) at point I. fit dB_z/dr cylindrical problems. $= 4$, fit $K = (1/B_y)dB_y/dx$ (1/cm). fit $n = (r/B_z)dB_z/dr$ for cylindrical problems. $= 5$, fit B_y/B_o at point I. fit B_z/B_o at point I. Where B_o is field at the center of the gap.

Table B.9.2.II. (continued) Data defining the field at fitted points.

Variable	Definition
	$= -1$ fit the $n = 1$ (dipole) harmonic coeff.
	$= -2$ fit the $n = 2$ (quad.) harmonic coeff.
	$= -3$ fit the $n = 3$ (sext.) harmonic coeff.
	$= -4$ fit the $n = 4$ (octu.) harmonic coeff.
	$= -5$ fit the $n = 5$ (decu.) harmonic coeff.
	etc.

The numbers FFIT, WFIT, and ITFIT are repeated in sequence until NFIT sets have been entered.

The third type of data is the coordinates of the fitted points, which are given by the variables XFIT(I) and YFIT(I). If the points fall on a line parallel to the horizontal axis, then one need not enter all the coordinates. Successive values of XFIT will be incremented by the distance between the first two values of XFIT and the first value of YFIT will be used for all points.

When entering data for a mixture of mu-infinite and mu-finite points, the NAIR mu-infinite points are entered first and in a block, $I = 1$ to NAIR.

B.9.2.3 Fitting parameters.

There are two tasks that must be performed to define the parameters to be used in the optimization. These are the following. *Firstly*, if the shape of a poleface or a boundary of a coil is to be changed, one must define the original poleface or current boundary. *Secondly*, one must define number and type of parameters being used, that is, total current, currents in specific regions, current filaments, or bumps on polefaces. Auxiliary information, like the location of the current region or the apex of the bump, along with limits on the size of current changes or bump heights, must be defined to keep the optimization process from going astray. The following paragraphs will describe the input parameters and options available for each of these two tasks.

Definition of interface or boundary to be optimized. The interface or boundary is described by giving the logical (K, L) coordinates of all points on the interface or boundary that are going to be affected by the parameters. For example, consider putting a bump on the edge of the poleface for a dipole magnet. If the poleface is 20 cm wide and you want the bump somewhere on the outer 5 cm of the poleface, than you must give the logical coordinates of the points on the outer 5 cm of the poleface. Another example is the location of a current region. Suppose that you want to move the coil up or down. The upper and lower boundaries of

the coil must be specified along with a few points on the sides of the coils that might be effected by the movement. If you want to change the total current or the current density in a given region, then one does not have to specify an interface or a boundary. The same is true of changing the current in a current filament or the value of the electrostatic potential on a given surface in the case when one has specified that the potential on the surface be held constant. The program does not allow one to change the location of current filaments or surfaces of constant potential.

Two lines of input are required. The first requires one number, NPOLE, which is the number of points on the optimization boundary. If NPOLE is zero, the second line is not needed. The second line is the list of logical coordinates from I = 1 to NPOLE. The format is KPOLE(1), KPOLE(2),..., KPOLE(NPOLE), LPOLE(1), LPOLE(2),..., LPOLE(NPOLE). If the points do not lie on a vertical line, then one can use a shortcut by entering "KPOLE(1) S" for the first array. The code will fill the rest of the array with the numbers KPOLE(1)+1, KPOLE(1)+2,..., KPOLE(1)+NPOLE−1. The boundary points may now be addressed by the index I = 1 to NPOLE instead of their logical coordinates. This will be important in defining the parameters.

Number, type, location, limits, and perturbation factors of the parameters. The first line of input for this task is the number of parameters NPAR. If NPAR is negative, the code expects to receive "soft restraint" data, which will be discussed below. The next NPAR lines of data contain three types of numbers associated with the arrays JTYPE, IND, and FAR. Figure B.9.2.1 schematically shows the relationship between the read statements used to fill these arrays

```
READ NPAR
DO J=1,NPAR
READ JTYPE(J)
     DO K=1,40
     READ IND(K,J)
     NEXT K
     DO K=1,5
     READ FAR(K,J)
     NEXT K
NEXT J
```

Fig. B.9.2.1: Schematic representation of the input to parameters JTYPE, IND, and FAR using BASIC language loops.

The meaning of the elements in the index array IND depends on the value of JTYPE. Basically IND has to do with the location of the bump, current region, or current filament being considered. The array FAR defines a rotated coordinate system, the limits on the allowed parameter variation, and the "perturbation factors",

which will be defined later.

Table B.9.2.III gives the definitions of IND when JTYPE = 0, that is, when currents are being varied. This is complicated because the current may be the total current, the current density in a region, or the current in a set of current filaments. Furthermore, in some cases, the current in one region may be the return current for a current in another region, hence the two currents must be equal.

Table B.9.2.III The meanings of IND when JTYPE(J) = 0.

THE TOTAL CURRENT (XJFACT) IS THE PARAMETER
Leave the array IND unchanged by skipping over this input.

THE INDIVIDUAL REGION CURRENT IS A PARAMETER
IND(1,J) = the region number entered in AUTOMESH or LATTICE
IND(2,J) = minus the region number if it is the return current.

THE CURRENT IN SET OF FILAMENTS IS THE PARAMETER
IND(1,J) = K coordinate of the 1st filament
IND(2,J) = L coordinate of the 1st filament
IND(3,J) = K coordinate of the 2nd filament
IND(4,J) = L coordinate of the 2nd filament
....
....
IND(2N−1,J) = K coordinate of the Nth filament
IND(2N,J) = L coordinate of the Nth filament
IND(2N+1,J) = −K coordinate of the 1st return filament
IND(2N+2,J) = −L coordinate of the 1st return filament
....
....
IND(4N−1,J) = −K coordinate of the Nth return filament
IND(4N,J) = −L coordinate of the Nth return filament

If there are no regions or filaments with return current, then of course there are no negative coordinates.

There are basically two types of bumps that can be put on polefaces: linear bumps and smooth bumps. The linear bump looks like a trapezoid (or a triangle when the left apex and right apex are the same point). The smooth bump is a quadratic spline and has three variations, which we have called the three interval, left side two interval, and right side two interval. See Fig. B.9.2.2. The shape of the bump is determined by the location of the apex points and the end points. The height of the bump is the parameter used in the optimization. The array IND contains the apex and end point locations.

Coil boundaries can be moved by bump mechanism also. For coils only linear bumps are allowed. To move a coil upward for example, one would use a negative rectangular bump on the bottom and positive rectangular bump on the top having the same width as the coil. One could then use the "soft restraint" parameters to make the height parameters of the two bumps nearly the same. As with polefaces,

the height is the parameter, and the apex and end-point information is put into the array IND.

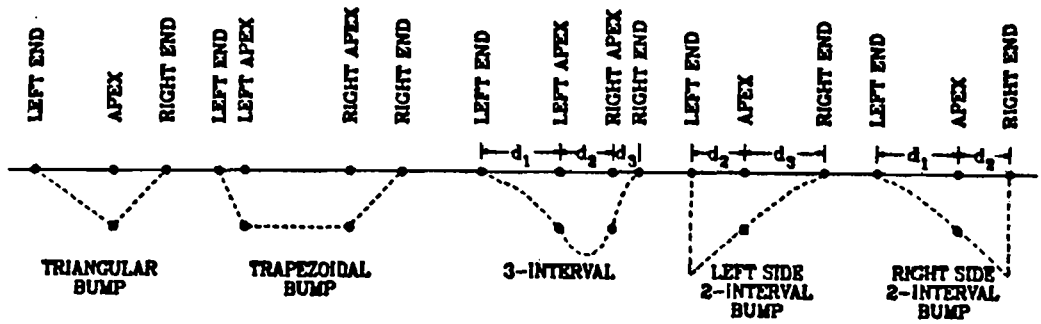


Figure B.9.2.2: Definition or types of bumps.

In those POISSON solutions where the potential on a given surface has been constrained to remain constant (See Sec. B.5.5), one can use MIRT to vary that potential value to optimize the field at some other location. The potential is the parameter and the points on the surface are stored in the array IND.

Table B.9.2.IV summarizes the values of JTYPE for the above types of optimization, and Table B.9.2.V gives the meaning of IND for these cases.

Table B.9.2.IV Allowed Values of JTYPE(J).

JTYPE(J)	Defintion
0	A current is the parameter, See Table B.9.2.III
1	Poletip linear bump height is the parameter
2	Poletip smooth bump height is the parameter
3	Coil linear bump height is the parameter
4,5	(Not used at present)
6	The constant potential is the parameter
negative	When JTYPE(J) is negative, the code expects a two-digit number. The leftmost digit of JTYPE is called ITYPE(J). Normally ITYPE(J) is given its default value: ITYPE=1, The parameter effects only the μ-infinite fitted points; this is the default when NIRN = 0. ITYPE=2, The parameter effects only the μ-finite fitted points; this is the default when NAIR =0. ITYPE=3, The parameter effects both the μ-infinite and μ-finite points; this is the default when both NAIR and NIRN are nonzero. Negative JTYPE allows the user to change ITYPE from its default value. The rightmost digit of JTYPE sets the parameter type as above.

The location of apex points, end points, and points on surfaces of constant potential are specified by giving the index I in the coordinate pair (KPOLE(I),LPOLE(I)) as discussed above in describing the optimization boundary.

Table B.9.2.V Interpertation of IND(K,J) for JTYPE(J) > 0

Type	Meaning of IND(K,J)
Triangular Bump[a] (JTYPE = 1 or 3)	IND(1,J) = left end point IND(2,J) = apex IND(3,J) = right end point
Trapezoidal Bump (JTYPE = 1 or 3)	IND(1,J) = left end point IND(2,J) = left apex IND(3,J) = right apex IND(4,J) = right end point
3 Interval Bump (JTYPE = 2)	IND(1,J) = left end point IND(2,J) = left apex IND(3,J) = right apex IND(4,J) = right end point
2 Interval left side bump (JTYPE = 2)	IND(1,J) = left end point IND(2,J) = left end point IND(3,J) = apex IND(4,J) = right end point
2 Interval right side bump (JTYPE = 2)	IND(1,J) = left end point IND(2,J) = apex IND(3,J) = right end point IND(4,J) = right end point
Constant Potential Surface (JTYPE = 6)	Skip the input. The code knows that the indices (KPOLE, LPOLE) for the surface are to be loaded into the IND array.

[a] Note that linear triangular bumps may be "leftsided" and "rightsided" by making the apex and one edge point the same. This is useful at the edge of a poletip.

The next set of entries on the same line with JTYPE and IND go into the FAR array. Table B.9.2.VI describes the five enteries into the FAR array.

Table B.9.2.VI Definition of the Constants in the FAR Array.

Constant	Default	Definition
FAR(1)=PHI(J)	0.0	The angle in degrees of rotation to the (x', y') coordinate system for bump parameter J.

Table B.9.2.VI (continued) Definition of the Constants in the FAR Array.

Constant	Default	Definition
FAR(2)=CUMIN(J)	none	The minimum allowed value of the total current, current density in a region, or current in a filament when JTYPE=0; the minimum bump height (usually a negative number) at the poleface or coil boundary when JTYPE = 1, 2, or 3; or the minimum allowed potential when JTYPE = 6.
FAR(3)=CUMAX(J)	none	The maximum allowed value of the total current, current density in a region, or current in a filament when JTYPE=0; the maximum bump height (usually a positive number) at poleface or coil boundary when the JTYPE = 1, 2, or 3; or the maximum allowed potential when JTYPE = 6.
FAR(4)=FUAIR(J)	.002	The mu-infinite perturbation factor for parameter J. If FUAIR < 0, then FUAIR is a perturbation amplitude; if FUAIR > 0, then the perturbation amplitude is FUAIR times the half gap at the bump apexes.
FAR(5)=FUIRN(J)	.002	The mu-finite perturbation factor for parameter J. (Comments under FUAIR(J) above are true for FUIRN(J) also.)

Initially bumps were viewed as iron added or subtracted to the poleface of a dipole magnet. In this case the apex stands vertically on a horizontal surface. When a bump is to be put on a surface that is not horizontal, it is necessary to rotate the coordinate system of the surface so that the bump is vertical in the new (x', y') system. To put bumps on non-horizontal surfaces, one should specify the rotation angle PHI(J) that carries the standard coordinate system (x, y) into the new rotated system.

The casual user will not want to change the perturbation factors FAR(4) and FAR(5) without acquiring an understanding of the least-squares procedure used to optimize the parameters. The perturbation factors are used in the calculation of the coefficient matrix, which is a derivative of the fitted points with respect to the parameters. These derivatives are calculated numerically by making small perturbations of the parameters and calculating the change of the fitted points. The perturbation factors determine the size of the perturbations used to calculate the derivatives. For more information see Sec. B.13.6, on numerical methods.

B.9.2.4 Input for soft restraints.

When not using soft restraints, just put an "S" on this line and proceed to the next data group. The program is currently set up for up to 10 soft restraints. Three types of numbers are required: 1. the index of the parameters to be restrained, 2. the desired values of the parameters, and 3. the weight factors used in the soft restraint. In Eq. (B.9.1.1.1), these are like the symbols "i", "p_i", and "v_i." The meanings are not exactly the same because the summation is sequential from 1 to l in Eq. B.9.1.1.1, but the soft parameter indices need not be sequential. Table B.9.2.VII summarizes the required input.

Table B.9.2.VII Summary of Input for Soft Restraints

Array	Meaning
JSOFT(I)	The array index J of the parameter to be restrained
SOFIT(I)	The desired value of the restrained parameter
WOFIT(I)	The weight factor for the restrained parameter

The code expects to read in 10 values of JSOFT, followed by 10 values of SOFIT, and followed by 10 values of WOFIT in the free format described in Sec. B.3.1.

B.9.2.5 Input for regeneration of the lattice.

After introducing bumps on polefaces or coils, it is necessary to regenerate the lattice so that the sides of the triangular mesh conform as closely as possible to the physical boundaries of the material. One must specify the number, type, and logical limits of the regions to be regenerated. The first line contains NGEN, the number of regeneration regions. The regeneration regions must enclose all mesh points that will be moved by "bump" parameters. The regeneration regions must not enclose any mesh points in coil regions, unless a bump is being put on a coil. This first line is followed by "NGEN" lines containing six numbers. Table B.9.2.VIII describes the numbers required.

B.9.2.6 Input data for the coefficient matrix.

This data set is optional. It allows the user to use the coefficient matrix from a previous run if it has been saved. See parameter PCON(2) in Table B.9.2.I above. If PCON(2) = 2, the coefficient matrix is read in automatically at this point in the input stream.

Table B.9.2.VIII Input to the Regeneration Array NAR for each value of M = 1,...,NGEN

Variable	Default	Definition
NAR(1)=MINK(M)	none	The regeneration region is a rectangle for which MINK is the minimum value of the logical coordinate K.
NAR(2)=MAXK(M)	none	Maximum value of the logical coordinate K for the regeneration region.
NAR(3)=MINL(M)	none	Minimum value of the logical coordinate L for the regeneration region.
NAR(4)=MAXL(M)	none	Maximum value of the logical coordinate L for the regeneration region
NAR(5)=ITRI(M)	0	The type of triangles to use in the regenerated region. ITRI = 0, equal weight triangles ITRI = 1, isosceles triangles ITRI = 2, right triangles
NAR(6)=MTYPE(M)	ITYPE	= 1, μ-infinite only regeneration region. = 2, μ-finite only regeneration region. = 3, μ-infinite and μ-finite regeneration region. The default is the same as ITYPE in Table B.9.IV above.

B.9.2.7 Input data for POISSON.

Since POISSON is used to generate the fitted points, the user must supply the required input for POISSON as defined in Chap. B.5 above. A minimum of three lines is required. The first two data lines have exactly the same meaning as in a normal POISSON run, namely, the "dump" number and any desired changes in the default CON values. The third data line terminates the MIRT run ("dump" = -1). If this third line contains the dump number of the just obtained solution, then one can follow with a fourth line containing some changed CON values. This will cause MIRT to run again with these new CON's. In any case, the final data line must set the dump number to -1 to terminate the run.

B.9.3 Output from MIRT.

MIRT directs output to both the terminal and to an output file called OUTMIR. The terminal output is described in the example given in Sec. B.12.4. It is an abbreviated version of the information given in OUTMIR, which will be described here. The output can be broken into four parts: a summary of the input, an initial POISSON run with analysis by MIRT, the main MIRT iteration cycle starting with a calculation of the coefficient matrix and ending with the convergence test, and finally a POISSON run giving the final optimized solution for the field. Each of these parts will be described below and illustrated with output from a simple H-shaped dipole magnet problem having two parameters, the total current and one smooth bump.

B.9.3.1 Summary of the input data.

This part begins with a list of CON'S, which are divided into input or default values and solution values. The later group contains parameters which can be changed by the code, although there are some which do not change. The distinction is not clear cut. The permeability data from TAPE35 is printed next. This is followed by a printout of the input data to MIRT, namely, the PCON array, the list of points to be fitted, the fitting parameters, the regeneration region data, and the optimization boundary data. Figure B.9.3.1 shows an example of the output. Note that the current version of the code lists a parameter called PCON(12) = SOFACT, which has no value. This should be ignored. It will be deleted or given a value in future versions of the program.

```
problem constants for optimization
(1) loop   = 5.  max.  no.  of optimization cycles
(2) matrix = 0
(3) ipunch = 0
(4) mprint = 2
(5) muse   = 1.  no.  of solutions with each coeff.  matrix
(6) rlx    = 0.5000000.  starting relaxation factor
(7) rlxs   = 0.0078125.  minimum relaxation factor
(8) test1  = 0.1000000.  relaxation factor test
(9) srsum  = 0.100e-19.  min.  sum (wt.  sqr.  of delta quantities 0(10) pcnt = 0.1000000
(11) femax = 0.500.  factor for parameter limits
(12) sofact=
 no.  of fitted points = 11
 specified values of by,mu=inf.  at y = 0.000      weight factors  kfit  lfit
     by,mu=inf.(  1) = 1.6000e+04    at x = 0.000   1.0000e+00       1     1
     by,mu=inf.(  2) = 1.6000e+04    at x = 0.500   1.0000e+00       2     1
     by,mu=inf.(  3) = 1.6000e+04    at x = 1.000   1.0000e+00       3     1
     by,mu=inf.(  4) = 1.6000e+04    at x = 1.500   1.0000e+00       5     1
     by,mu=inf.(  5) = 1.6000e+04    at x = 2.000   1.0000e+00       6     1
     by,mu=inf.(  6) = 1.6000e+04    at x = 2.500   1.0000e+00       7     1
     by,mu=inf.(  7) = 1.6000e+04    at x = 0.000   1.0000e+00       1     4
     by,mu=inf.(  8) = 1.6000e+04    at x = 0.500   1.0000e+00       3     4
     by,mu=inf.(  9) = 1.6000e+04    at x = 1.000   1.0000e+00       4     4
```

```
specified values of by.mu=inf.  at y = 0.000      weight factors  kfit  lfit
    by,mu=inf.( 10) = 1.6000e+04    at x = 1.500   1.0000e+00       5     4
    by,mu=inf.( 11) = 1.6000e+04    at x = 2.000   1.0000e+00       6     4
```

Figure B.9.3.1: Printout from OUTMIR of input data to MIRT for an H-shaped magnet containing infinitely permeable iron.

The fitted points are divided into sets with fixed values of Y. The fitting parameters are described by "type", that is, TYPE = 0 means a current, TYPE = 1, 2, or 3 is a bump, and TYPE = 6 is a constant potential. An example is shown in Fig. B.9.3.2 for two parameters.

```
no.  of parameters = 2
  1.  type 0 current (xjfact = 1.00000) - mu=infinit
      mu=i prt f.   = 2.000e-03
      mu=f prt f.   = 0.000e+00      min.  change = -5.000e-01
      cum.  change = 0.000e+00       max.  change =  5.000e-01

  2.  type 2 iron bump - smooth    - y direct.  phi = 0.00 - mu=infinit
            left        n =  1  k =  9  l = 7 x = 3.4000 y = 2.0000
            left apex   n =  2  k = 10  l = 7 x = 3.8250 y = 2.0000
            right apex  n =  5  k = 13  l = 7 x = 5.1000 y = 2.0000
            right       n =  6  k = 14  l = 8 x = 5.5000 y = 2.4000

   mu=i prt f.   = 2.000e-03
   mu=f prt f.   = 0.000e+00

                                  min.  change = -1.000e+00
   cum.  change = 0.000e+00       max.  change =  1.000e+00
```

Figure B.9.3.2: Printout from OUTMIR of parameter data entered into MIRT for an H-shaped magnet.

The core of the magnet is taken to be infinitely permeable iron. The phrase at the left (mu=i prt f.=2.000-e3) refers to FAR(4) = FUAIR = .002 as defined in Table B.9.2.VI above. The phrases "min change" and "max change" refer to FAR(2) and FAR(3). The phrase "cum. change" is an empty format at this point but will have meaning when the parameters are changed during the iteration cycles of MIRT. Under the heading of bump parameter, the phrase "phi = 0" means that the bump is on a horizontal surface, hence PHI=FAR(1) indicates that no rotated coordinate system is needed.

The next output summarizes the regeneration region data and the location of the original optimization boundaries, as shown for example in Fig. B.9.3.3 In this

example, the notation "2*" schematically shows the endpoints and apex points of the bump defined by parameter 2. If there had been more than one bump, then there would be a set of "3*"'s, "4*"'s, etc. One set for each bump. Note also that NPOLE = 6 for this example.

If the user had requested (PCON(2)=2) the use of a previously saved coefficient matrix or had entered soft restraint data, this input data would appear in OUTMIR after the regeneration data. This ends the summary of the input data.

B.9.3.2 Initial POISSON run with MIRT analysis.

MIRT calls POISSON for an evaluation of the least squares summation, which is to be minimized. The next entry in OUTMIR is a summary of the POISSON interation history as shown in Fig. B.9.3.4. The format may be a little confusing because it is folded in on itself to save space. Table B.9.3.1 gives the meanings of the numbers listed.

```
no.  of regeneration regions = 1
 1.  mink = 8  maxk = 15  minl = 3  maxl = 9  itri = 0  mgen = 1

optimization
boundary...  n   k   l   x       y
       2*    1   9   7  3.4000  2.0000
2*           2  10   7  3.8250  2.0000
             3  11   7  4.2500  2.0000
             4  12   7  4.6750  2.0000
2*           5  13   7  5.1000  2.0000
        *2   6  14   8  5.5000  2.4000
```

Figure B.9.3.3: Printout from OUTMIR showing the regeneration region and optimization boundary data.

```
elapsed time = 0.2 sec.

cycle      amin        amax      residual-air   eta-air   rhoair   xjfact
                       gmax      residual-iron  eta-iron  rhofe
    0   0.0000e+00  0.0000e+00   1.0000e+00    1.0000    1.9580   1.0000
                    4.0000e-03   1.0000e+00    1.0000    1.0000
  330  -1.1933e+05  0.0000e+00   4.5818e-07    0.9336    1.9580   1.0000
                    4.7133e-02   2.3388e-02    0.9251    1.0000

  poisson converged in 330 iterations
elapsed time = 1.7 sec.
```

Figure B.9.3.4: Printout from OUTMIR of the iteration history as POISSON solves Poisson's equation for the vector potential.

Table B.9.3.I. Defintions of the Parameters from POISSON.

Variable	Definition
AMIN	Minimum value of the vector (scaler) potential over the mesh.
AMAX	Maximum value of the vector (scaler) potential over the mesh.
Residual-air	CON(88), a parameter used to test the convergence of POISSON. If this parameter is less than 5.0e-07, the solution has converged for the air and interface points.
Eta-air	CON(106), a measure of the rate of convergence of the solution during the current cycle. It is used to calculate Rhoair = CON(75).
Rhoair	CON(75), the overrelaxation factor for air and interface points. This parameter is automatically optimized during the interation if the intitial value of Rhoair is set equal to the value of CON(74)=RHOPT1.
XJFACT	CON(66), the factor by which all current and current densities (but not current filaments) will be scaled. XJFACT = 0 indicates the use of a scalar potential for electrostatic problems.
Gmax	The maximum value of gamma, the inverse of the relative permeability over the mesh.
Residual-iron	CON(89), residual of iron points; used in testing convergence of the POISSON solution.
Eta-iron	CON(107), the rate of convergence of the solution at iron points during the current cycle.
Rhofe	CON(77), the overrelaxation factor for iron points in problems with finite but variable permeability.

This is followed by a table of fields and their derivatives at mesh points. It will be recalled that these fields are obtained by a least squares fitting process. The last column of the table is "afit". The ratio of "afit" to "a" is a measure of the goodness of fit by the edit polynomials used to approximate the field.

The next output gives a graph and table summary of the deviations of the specified (given in the input) and the achieved (calculated from POISSON) values of the fitted points. An example is given in Fig. B.9.3.5. The last line of this output is the sum of the weighted squares that is to be minimized.

```
            percent -5.493e+00                                          -7.105e-13
                     1.51212e+04                                         1.60000e+04
by.mu=inf.  at y=0.000 iiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiii
            0.000    x         *                                          0
            0.500    x         *                                          0
            1.000    x        *                                           0
            1.500    x       *                                            0
            2.000    x    *                                               0
            2.500    x*                                                   0
by.mu=inf.  at y=1.000 iiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiii
            0.000    x           *                                        0
            0.500    x           *                                        0
            1.000    x          *                                         0
            1.500    x          *                                         0
            2.000    x         *                                          0
                     iiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiii
                     1.51212e+04                                         1.60000e+04
            percent -5.493e+00                                          -7.105e-13

*  represents the achieved value
O  represents the specified value
+  represents both the achieved value and the specified value if they lie together

delta by,mu=inf.  at y = 0.000                          by,mu=inf.    percent deviation
      delta by,mu=inf.( 1) = 7.943e+02   at x =  0.000   15205.72115          -4.964
      delta by,mu=inf.( 2) = 7.960e+02   at x =  0.500   15203.99429          -4.975
      delta by,mu=inf.( 3) = 8.017e+02   at x =  1.000   15198.25744          -5.011
      delta by,mu=inf.( 4) = 8.138e+02   at x =  1.500   15186.39542          -5.085
      delta by,mu=inf.( 5) = 8.361e+02   at x =  2.000   15163.86665          -5.226
      delta by,mu=inf.( 6) = 8.788e+02   at x =  2.500   15121.17478          -5.493

delta by,mu=inf.  at y = 1.000                          by,mu=inf.    percent deviation
      delta by,mu=inf.( 7) = 7.884e+02   at x =  0.000   15211.64494          -4.927
      delta by,mu=inf.( 8) = 7.891e+02   at x =  0.500   15210.91492          -4.932
      delta by,mu=inf.( 9) = 7.914e+02   at x =  1.000   15208.58767          -4.946
      delta by,mu=inf.(10) = 7.958e+02   at x =  1.500   15204.16690          -4.974
      delta by,mu=inf.(11) = 8.031e+02   at x =  2.000   15196.86214          -5.020

sum no.  0  (wt.sqr.of delta quantities) = 7.190e+06
```

Figure B.9.3.5: Printout from OUTMIR of the specified and achieved values of the points to be fitted.

B.9.3.3 The MIRT iteration cycle.

MIRT now begins its first iteration cycle by calculating the coefficient matrix using the perturbation factors, ("Perturb. a." for mu-infinite specified points and "Perturb. f" for mu-finite points.) There is a POISSON run for each parameter. Fig. B.9.3.6 shows the output for the example we have been carrying along.

```
elapsed time = 2.0 sec.
calculating coefficient matrix

  1. type 0 current (xjfact = 1.00200) - mu=infinit                perturb.  a.  = 2.000e-03
                                                                   perturb.  f.  = 2.000e-03
     poisson converged in 160 iterations

  2. type 2 iron bump - smooth  -y direct.  phi =  0.00 - mu=infinit  perturb.  a.  = 4.000e-03
            left        n =  1 k = 9  l = 7  x = 3.4000  y=  2.0000   perturb.  f.  = 2.000e-03
```

```
          left apex   n =  2 k =10  l = 7  x = 3.8250  y=  1.9984
          right apex  n =  5 k =13  l = 7  x = 5.1000  y=  1.9985
          right       n =  6 k =14  l = 8  x = 5.5000  y=  2.4000

  poisson converged in 130 iterations
elapsed time = 3.5 sec.
coefficient matrix no.  1

            1           2
 1      1.039e+04   -5.309e+02
 2      1.039e+04   -5.085e+02
 3      1.037e+04   -4.671e+02
 4      1.034e+04   -3.946e+02
 5      1.028e+04   -2.142e+02
 6      1.019e+04    1.432e+02
 7      1.042e+04   -5.432e+02
 8      1.042e+04   -5.467e+02
 9      1.041e+04   -5.495e+02
10      1.038e+04   -5.497e+02
11      1.034e+04   -5.345e+02
```

Figure B.9.3.6: Printout from OUTMIR of the calculation for the coefficient matrix in the case of two parameters and 11 fitted points.

MIRT now uses the coefficient matrix to calculate changes in the parameters. The new parameters are displayed in the standard format used to display parameters as illustrated in Fig. B.9.3.7, which also shows the iteration history of a POISSON run using these parameters. MIRT then displays the new shape of the optimization boundary and the difference between the specified and achieved values of the fitted points. In addition MIRT also displays the new least squares summation and other terms needed to calculate a new relaxation factor. The anticipated final sum is compared with the actual sum to determine if convergence has occurred. In the example being used, convergence did not occur after the first iteration and hence MIRT starts a new iteration cycle starting with a recalculation of the coefficient matrix and ending with a convergence test.

The example we are following converged after three iterations. The important summations at the end of the third cycle are shown in Fig. B.9.3.8. Note that the anticipated least squares sum is very close to "sum no. (3)."

```
optimized solution no.  1, relaxation factor = 0.500000

  1. type 0 current (xjfact = 1.04213) - mu=infinit

          solution a.  = 4.213e-02    min.  change = -5.000e-01
          cum.  change = 4.213e-02    max.  change =  5.000e-01

  2. type 2 iron bump - smooth  -y direct.  phi =  0.00 - mu=infinit  perturb.  a.  = 4.000e-03
            left        n =  1 k = 9  l = 7  x = 3.4000  y=  2.0000   perturb.  f.  = 2.000e-03
            left apex   n =  2 k =10  l = 7  x = 3.8250  y=  1.9696
            right apex  n =  5 k =13  l = 7  x = 5.1000  y=  1.9709
            right       n =  6 k =14  l = 8  x = 5.5000  y=  2.4000

          solution a.  = 7.575e-02
```

```
                                    min.  change = -1.000e+00
          cum.  change = 7.575e-02   max.  change =  1.000e+00

elapsed time =     3.6 sec.
    cycle       amin          amax     residual-air   eta-air  rhoair   xjfact
                              gmax     residual-iron  eta-iron rhofe
        0   -1.2435e+05   0.0000e+00   4.5818e-07   0.9336   1.9580   1.0421
                          4.7133e-02   2.3388e-07   0.9251   1.0000
      230   -1.2303e+05   0.0000e+00   3.2147e-07   0.9062   1.9580   1.0421
                          5.5221e-02   1.1848e-07   0.8775   1.0000

    poisson converged in 230 iterations
elapsed time=    4.7 sec.
```

Figure B.9.3.7: Printout from OUTMIR showing the new parameters and the results of a POISSON run using these new parameters.

```
optimization converged

                (wt.sqr.  of delta quantities)

sum no.  0 =            7.190e+06

sum no.  1 =            1.951e+06

sum no.  2 =            4.331e+03

sum no.  3 =            3.819e+02

anticipated final sum (wt.sqr.of delta quantities) = 3.785e+02

dump number 1 has been written on tape35.
```

Fig. B.9.3.8: Printout from OUTMIR of the least square sum and related quantities for the final iteration of our example.

B.9.3.4 Final POISSON run.

After convergence, MIRT calls POISSON for a final run and writes the full table of field quantities at mesh points. The first part of this final run is illustrated in Fig. B.9.3.9

k	l	a(vector)	x	y	bx(gauss)	by(gauss)	bt(gauss)	dby/ dx(gauss/cm)	dby/dx(gauss/cm)	afit
1	1	0.000000e+00	0.00000	0.00000	0.000	16003.397	16003.397	0.0000e+00	0.0000e+00	-1.5e-04
2	1	-6.858502e+03	0.42857	0.00000	0.000	16002.729	16002.729	0.0000e+00	-3.0529e+00	6.1e-04
3	1	-1.371645e+04	0.85714	0.00000	0.000	16000.931	16000.931	0.0000e+00	-5.1825e+00	9.7e-03
4	1	-2.057351e+04	1.28571	0.00000	0.000	15998.621	15998.621	0.0000e+00	-5.2249e+00	4.7e-03
5	1	-2.742966e+04	1.71429	0.00000	0.000	15997.023	15997.023	0.0000e+00	-1.5546e+00	3.0e-03
6	1	-3.428560e+04	2.14286	0.00000	0.000	15998.002	15998.002	0.0000e+00	6.4812e+00	1.9e-02
7	1	-4.114275e+04	2.57143	0.00000	0.000	16002.359	16002.359	0.0000e+00	1.1787e+01	3.4e-03
8	1	-4.800163e+04	3.00000	0.00000	0.000	16003.557	16003.557	0.0000e+00	-1.5322e+01	-6.7e-02
9	1	-5.485603e+04	3.42857	0.00000	0.000	15974.202	15974.202	0.0000e+00	-1.4431e+02	-1.2e-01
10	1	-6.168051e+04	3.85714	0.00000	0.000	15850.199	15850.199	0.0000e+00	-4.7115e+02	-8.6e-02
1	2	0.000000e+00	0.00000	0.33333	0.000	16003.815	16003.815	2.5512e+00	0.0000e+00	-1.9e-05
2	2	-4.097282e+03	0.25602	0.33334	-0.643	16003.558	16003.558	2.4463e+00	-2.0115e+00	5.7e-04
3	2	-1.050700e+04	0.65655	0.33335	-1.514	16002.180	16002.180	1.8231e+00	-4.8065e+00	4.2e-03
4	2	-1.723347e+04	1.07693	0.33336	-1.968	15999.772	15999.772	2.1506e+01	-6.4759e+00	7.2e-03
5	2	-2.403050e+04	1.50179	0.33338	-1.483	15997.153	15997.153	- 2.7332e+00	-5.3818e+00	7.9e-03
6	2	-3.085423e+04	1.92837	0.33342	0.535	15995.984	15995.984	- 6.7020e+00	7.7012e-01	6.3e-03
7	2	-3.769382e+04	2.35592	0.33349	3.804	15998.823	15998.824	- 7.1140e+00	1.3066e+01	6.8e-03
8	2	-4.454824e+04	2.78426	0.33362	3.804	16006.895	16006.896	1.2637e+01	2.2155e+01	-2.3e-02
9	2	-5.142044e+04	3.21349	0.33385	-15.009	16011.027	16011.034	8.6834e+01	-1.5287e+01	-1.4e-01
10	2	-5.830898e+04	3.64407	0.33425	-83.877	15971.119	15971.340	2.4620e+02	-2.0012e+02	-3.8e-01

Figure B.9.3.9: Printout from OUTMIR of a part of the final POISSON run after convergence has been achieved.

Chapter B.10

Diagnostic and Error Messages

B.10.1 Messages from AUTOMESH

Diagnostic and Error messages printed from AUTOMESH can be broken into three categories: 1. those starting with the word "ERROR", 2. those starting with the word "TROUBLE", and 3. two additional messages. In the sections below, we list the messages, briefly define the problem and give a possible solution.

The following five conventions make the explanations simpler to write.

1. `CHANGE THE MESH SIZE` — usually the mesh is too coarse; user should rerun the problem with a finer mesh; sometimes a slight mesh size change will suffice.

2. `(X1,Y1)/(R1, THETA1)` — the Cartesian/polar coordinates of the previous point (from).
   `(X2,Y2)/(R2, THETA2)` — the Cartesian/polar coordinates of the present point (to).

3. `R` --- (printed as the value of a variable) means that this variable has been set out of range and not supplied by the user.

4. `(--) means computer prints out the value.`

5. `REGION (--)/O.K.` — AUTOMESH has successfully found paths for all boundary points in this, (--), region; if errors occur in one region, AUTOMESH proceeds to the next.

B.10.1.1 Messages containing "ERROR"

1. `"--- ERROR --- DATA FOR THIS CIRCLE FROM (X1,Y1)/(R1, THETA1) TO (X2,Y2)/(R2, THETA2) IS INCONSISTENT ..."`
   Either one or both coordinates are not given or the two points with center at $(X0, Y0)$ do not lie on the same circle to a relative accuracy of 10^{-3}. Correct the input data for the listed coordinates. The user should check that the coordinates are given RELATIVE to $(X0, Y0)$. Message from subroutine DATUPS.

2. `"--- ERROR --- DATA FOR THIS LINE ARE INSUFFICIENT ..."`
   Either one or both coordinates are not given. Correct the input data for the listed coordinates. Message from subroutine DATUPS.

3. `"--- ERROR --- DATA FOR THIS HYPERBOLA FROM (X1,Y1) TO (X2,Y2) IS INCONSISTENT ..."`
   Either one or both coordinates are not given, R is not given, or the two points do not lie on the same hyperbolic branch to a relative accuracy of 10^{-3}. Message from subroutine DATUPS.

4. `"--- ERROR --- X/Y IS OUT XMIN, XMAX/YMIN, YMAX LIMITS ..."`
   The X or Y point printed is less or greater than the given minimum or maximum value for X/Y in the first REG input line. Correct input. Message from DATUPS.

5. `"--- ERROR --- (KMAX + 2) * (LMAX + 2) = (--) IS GREATER THAN PROGRAM DIMENSIONS OF (--) ..."`
   The total number of mesh points have exceeded the maximum value dimensioned. Cut mesh size or increase parameter MXDIM and recompile as directed by the complete diagnostic message. (Note: Versions of the code received from us before June 1986 have a different diagnostic message and do not give directions for changing MXDIM.) Message from subroutine SETXY.

6. `"--- ERROR --- TROUBLE IN FINDING THE PATH OF A POINT ..."`
   AUTOMESH encountered trouble in both "forward" and or "backward" pass in subroutine LOGIC. To correct, decrease mesh size near the point and try again. Message from main program.

B.10.1.2 Messages containing "TROUBLE"

1. "--- TROUBLE --- DIMENSIONS FOR THE NSEG ARRAYS, EXCEEDED NSG OF (--) ..."
   AUTOMESH has exceeded the maximum number of boundary segments dimensioned in the program. Increase parameter NSG and recompile as directed. Message from subroutine FISHEG. (Note: Versions of the program received before June 1986 have no directions for increasing NSEG. The user can only decrease the number of segments in the input.)

2. "--- TROUBLE --- NPOINT = (--), EXCEEDS DIMENSION OF (--)"
   The number of PO entries for this region has exceeded the maximum number dimensioned. To correct, decrease the number of points or increase parameter NPTX and recompile as directed. (Note: Versions of the program received before June 1986 have no directions for increasing NPTX. The user can only decrease the number of points in the input.) Message from main program or subroutine INSERT.

3. "--- TROUBLE --- THE PROGRAM FOUND THE SAME (K, L) COORDINATES FOR THE FIRST AND LAST POINT OF THIS CURVE ..."
   The program has assigned the same mesh point in either vertical or horizontal direction for (X_1, Y_1) and (X_2, Y_2). This usually means mesh size is not fine enough.

   3a. Message is printed from subroutine LOGIC. The last line of the message prints the phrase "FORWARD PASS" or "BACKWORD PASS." AUTOMESH executes subroutine LOGIC twice—first in a "forward" search, and a second pass in a "backward" search—to find the path of the current segment. Then the program chooses the path with the smaller number of segments with no errors. A fatal error occurs if BOTH directions encounter "TROUBLE." To correct, change mesh size.

4. "--- TROUBLE --- PROGRAM DIMENSIONS 1000 FOR THE *KL* ARRAYS ARE INSUFFICIENT"
   The program has difficulty in finding the path for this segment and thus has exceeded the dimension allocated for storage of the path array. See 3a. above.

5. "--- TROUBLE --- LOGICAL PATH IS TRAPPED AT K = (--), L = (--)"
   The program cannot find the path for this current segment. See 3a above.

6. "--- TROUBLE --- TOO MANY END POINTS FOUND FOR THE LINE"
   The program has trouble adding a vertical/horizontal line region.

   6a. AUTOMESH could encounter a number of problems in subroutines XLINER/YLINER while attempting to add vertical/horizontal line regions. To correct, CHANGE MESH SIZE or in versions of the program received after April, 1986 set LINX/LINY = 1 in the first REG entry. (This latter option deletes the addition of all vertical/horizontal line regions at horizontal/vertical mesh change locations.)

7. "--- TROUBLE --- NO END POINTS FOUND FOR LINE"
   The program has trouble finding a mesh point for the end point of the added line region. See 6a. above.

8. "--- TROUBLE --- ONLY ONE END POINT FOR THE LINE"
   The program has trouble finding an end point for this added line region. See 6a. above.

9. "--- TROUBLE --- A POINT WITH (K = KREG) HAS X NOT = TO XREG"
   "--- TROUBLE --- A POINT WITH (L = LREG) HAS Y NOT = TO YREG"
   The program has difficulty adding a vertical/horizontal line region. See 6a. above.

B.10.1.3 Additional Diagnostic Messages

1. "DIMENSION OF 2000 FOR KR, LR ..."
   The program has run into difficulty and has exceeded the maximum number of points dimensioned for a region. CHANGE MESH SIZE and try again. Message from subroutine LOGSEG.

2. "DIMENSION OF 3000 INSUFFICIENT FOR KG, LG ..."
   The program has run into difficulty and has exceeded the total number of points dimensioned for *all* regions. CHANGE MESH SIZE and try again. Message from subroutine SAVAGE.

B.10.2 Messages from LATTICE

LATTICE writes all of diagnostic and error messages to the output file, OUTLAT, and some to the terminal if run is interactive. An explanation of the common terminology used in these messages is listed below.

1. k, l The mesh point numbering for the horizontal and vertical coordinates.
2. x, y The horizontal, vertical coordinates, respectively.
3. k', l' The mesh point numbering for the second of the two points.
4. (--) Means the computer prints out the value.

B.10.2.1 Messages Containing "ERROR EXIT"

1. `"--- ERROR EXIT --- TWO MESH DATA POINTS WITH A DIFFERENT K, L HAVE THE SAME X, Y COORDINATES"`
   followed by values of k, l, k', l', x, and y. This message is from the function ANGLF. The code has found the same physical coordinates assigned to two different logical points. Check input data; try reducing mesh spacing if input looks correct.

2. `"--- ERROR EXIT --- IN SUB. ANGLE COST = (--) AT KO = (--) LO = (--)"`
   Message from function ANGLF. The code has a cosine value greater than 1.0 at the logical point (KO, LO). Check input data, try reducing mesh spacing.

3. `"--- ERROR EXIT --- NWMAX EXCEEDS PROGRAM DIMENSIONS OF (--) ..."`
   Message from subroutine PRELIM. The storage for recalculating couplings has been exceeded. This storage has dimension of 1/2 of the parameter MXDIM. Increase MXDIM and recompile.

B.10.2.2 Messages Containing "INPUT DATA ERROR"

1. `"--- INPUT DATA ERROR --- ILLEGAL CHARACTER",`
   followed by a print of the input line. Message from subroutine FREE. The code has found a character it does not recognize in the line printed. Correct input.

2. `"--- INPUT DATA ERROR --- NO MANTISSA WITH EXPONENT",`
   followed by a print of the input line. Message from subroutine FREE. The code found an exponent standing alone in the line printed. Correct the input line.

B.10.2.3 Messages Containing "DATA ERROR"

These messages are issued whenever LATTICE encounters any errors in reading the input file. Mostly, such errors occur when a user creates his own input file for LATTICE. If the input file for LATTICE has been generated by a successful AUTOMESH run, it is unlikely there would be any errors of this type. In any case, the errors issued are self-explanatory. The user need only correct the identified error in the input file and rerun.

1. `"--- DATA ERROR --- THE NO. OF BOUND DATA VALUES (K, L, X, Y) = (--) FOR THIS REGION IS NOT A MULTIPLE OF 4"`
   Message from subroutine REREG. The code has found that the coordinate data on the input file is incomplete. Correct the input file (Usually TAPE 73). If generated by AUTOMESH, try changing mesh spacing.

2. `"--- DATA ERROR --- THE FIRST AND LAST POINTS OF REGION HAVE SAME K, L BUT DIFFERENT X, Y COORDINATES"`
   Message from subroutine REREG. Meshing has been done incorrectly. Correct input file. If generated by AUTOMESH, try changing mesh spacing.

3. `"--- DATA ERROR --- NEGATIVE OR ZERO L"`
   Message from subroutine REREG. The input file has an illegal value for the logical coordinate L. Correct input file.

4. `"--- DATA ERROR --- NEGATIVE OR ZERO K"`
   Message from subroutine REREG. The input file has an illegal value for the logical coordinate K. Correct input file.

5. `"--- DATA ERROR --- L, K AND LPRIME, KPRIME NOT ON SAME LOGICAL LINE"`
   Message from subroutine REREG. There is an error in the boundary input to LATTICE. Check input file.

6. `"--- DATA ERROR --- L EXCEEDS LMAX"`
   Message from subroutine REREG. The code has found a boundary point in the input file with a logical L coordinate greater than the maximum L coordinate. Correct input.

7. "--- DATA ERROR --- K EXCEEDS KMAX"
   Message from subroutine REREG. The code has found a boundary point in the input file with a logical K coordinate greater than the maximum K coordinate. Correct input.

8. "--- DATA ERROR --- YOU HAVE EXCEEDED THE MAXIMUM NUMBER OF REGIONS ALLOWED = (--)"
   Message from subroutine REREG. Too many regions. Increase parameter NRGN and recompile.

9. "--- DATA ERROR --- YOU HAVE EXCEEDED THE MAXIMUM NUMBER OF INPUT BOUNDARY POINTS PER REGION = (--)"
   Message from subroutine REREG. The storage for single region boundary points has been exceeded. Increase parameter NPMX and recompile.

10. "--- DATA ERROR --- TWO CONSECUTIVE DATA POINTS IN THIS REGION HAVE SAME K, L COORDINATES"
    Message from subroutine REREG. The code has found two consecutive boundary points assigned the same logical coordinates. Correct input data.

11. "--- DATA ERROR --- TWO CONSECUTIVE DATA POINTS IN THIS REGION HAVE SAME X, Y COORDINATES"
    Message from subroutine REREG. The code has found two consecutive boundary points assigned the same physical coordinates. Correct input data.

12. "--- DATA ERROR --- (KMAX+2)*(LMAX+2) EXCEEDS PROGRAM DIMENSIONS OF (--)"
    Message from subroutine REREG. There are too many mesh points in the problem. Increase the parameter MXDIM and recompile.

B.10.2.4 Messages Containing "TROUBLE" and "WARNING"

1. "--- TROUBLE --- DIMENSIONS FOR NO. OF SEGMENTS EXCEEDED NSG OF (--) ..."
   Prints to OUTLAT and terminal and immediately aborts. Message from main program; follow instructions given in the complete error message and recompile.

2. "--- WARNING ---THE MESH HAS NEGATIVE AND/OR ZERO AREA TRIANGLES"
   LATTICE writes to the file OUTLAT, a message whenever it encounters a negative or zero area in subroutine FILPOT, followed by the three coordinates that make up this triangle. The program processes the triangles of all regions before printing above message to OUTLAT and terminating. Message from main program; follow instructions or remesh the problem with a different mesh spacing.

3. "--- WARNING ---THE NUMBER OF INTERIOR POINTS = 0 ..."
   Message from subroutine SETTLE is self-explanatory in versions released after April 1986. For previous versions, the user has somehow set up the problem wrong. All points are boundary points and hence the potential is determined everywhere.

B.10.2.5 Miscellaneous Messages

1. "THE ABOVE REGION IS NOT CLOSED."
   This message is output to OUTLAT from subroutine REREG and is only a warning. User should check to see that the same values for the first and last coordinates for this region are specified if a closed region with interior points is desired.

2. "ITERATION TERMINATED---MAXIMUM NUMBER OF CYCLES."
   This message is output to OUTLAT from subroutine SETTLE and is only a warning. The mesh generation did not converge to the required accuracy after 100 iteration cycles. Run is continued with present mesh. User could try running the problem with this mesh or CHANGE MESH SIZE and rerun.

3. "THE LAST CORRECT POINT IS K = (--), L = (--)"
   Message from subroutine REREG. This message occurs after INPUT DATA ERROR messages numbers 3, 4, 5, 6, 7, 10, and 11. The logical coordinates are an aid in finding the error.

4. "ITERATION CONVERGED"
   Message from subroutine SETTLE. The mesh relaxation process was successful.

B.10.3 Messages from POISSON

B.10.3.1 Messages Starting with "ERROR EXIT"

1. `"--- ERROR EXIT --- (KMAX+2)(LMAX+2) IS GREATER THAN PROGRAM DIMENSIONS OF (---)"`
   Message from subroutine RDUMP. Too many points in problem. Increase parameter MAXDIM and recompile.
2. `"--- ERROR EXIT --- NWMAX EXCEEDS PROGRAM DIMENSIONS OF (---)"`
   Message from subroutine RDUMP. Not enough storage. Increase parameter MAXDIM and recompile.
3. `"--- ERROR EXIT --- THE MESH HAS NEGATIVE AND/OR ZERO AREA TRIANGLES"`
   Message from subroutine RDUMP. The mesh has a region where the triangles have collapsed or where logical lines have crossed. Remesh with a different mesh spacing.
4. `"--- ERROR EXIT --- MATERIAL CODE .GT.5"`
   Message from subroutine TABLE. The user has input CON(18) = NPERM negative and has read in a material code greater than 5. See instructions for use of NPERM = CON(18).

B.10.3.2 Messages Ending with "ERROR EXIT"

1. `"NAME OF MATERIAL IS LESS THAN OR EQUAL TO 1, OR IS GREATER THAN 11" "---ERROR EXIT---"`
   Message from subroutine TABIN. POISSON can handle up to 11 different materials. The first is reserved for air or current carrying coils. This message means that the user has input an illegal material number during the input of magnetic region data. Correct input.
2. `"THE NUMBER OF INPUT TABLES IS GREATER THAN FOUR" "---ERROR EXIT---"`
   Message from subroutine TABIN. The user has tried to input too many magnetic property tables. Only 4 tables can be input (1 internal and 3 external). Correct input.
3. `"GAMMA = H/B, AND B = 0.0" "---ERROR EXIT---"`
   Message from subroutine TABIN. The user is trying to input a MTYPE = 3 (H vs B) magnetic material table and has entered a value of B = 0. Since this would result in a divide by zero when converting to γ vs. B the run is stopped.
4. `"YOU HAVE EXCEEDED THE MAXIMUM DIMENSIONS ALLOWED FOR THE GAMMA VS B TABLES" "---ERROR EXIT---"`
   Message from subroutine TABIN. The number of data lines in a table of magnetic material properties (GAMMA vs B, MU vs B, or H vs B) is greater than 50. Edit the input table.

B.10.3.3 Messages with "DATA ERROR"

1. "--- DATA ERROR --- ITYPE IS NEGATIVE OR ZERO"
   Message from subroutine POWERS. CON(46) = ITYPE is negative or zero which is illegal. Consult information on CON(46).

B.10.3.4 Messages with "INPUT DATA ERROR"

1. "--- INPUT DATA ERROR --- ILLEGAL CHARACTER",
   followed by a print of the input line. Message from subroutine FREE. The code has found a character it does not recognize in the line printed. Correct line.

2. "--- INPUT DATA ERROR --- NO MANTISSA",
   followed by a print of the input line. Message from subroutine FREE. The code has found an exponent standing alone in the line printed. Correct input.

B.10.4 Messages from PANDIRA

B.10.4.1 Messages Starting with "ERROR EXIT"

1. "--- ERROR EXIT --- NOTE = CON(81) = 1"
   Message from main program. CON(81)=NOTE sets the point relaxation order and must be 0 for PANDIRA runs. Rerun LATTICE with CON(81) = 0.

2. "--- ERROR EXIT --- (KMAX+2)(LMAX+2) IS GREATER THAN PROGRAM DIMENSIONS OF (--")
   Message from subroutine PRDUMP. Too many points in the problem. Increase parameter MAXDIM and recompile.

3. "--- ERROR EXIT --- NWMAX EXCEEDS PROGRAM DIMENSIONS OF (--)"
   Message from subroutine PRDUMP. Not enough storage. Increase parameter MAXDIM and recompile.

4. "--- ERROR EXIT --- THE MESH HAS NEGATIVE AND/OR ZERO AREA TRIANGLES"
   Message from subroutine PRDUMP. The mesh has a region where the triangles have collapsed or where logical lines have crossed. Remesh with a different mesh spacing.

5. "--- ERROR EXIT --- NO. INTERFACE CURRENT POINTS .GT. DIMENSIONED ARRAY OF (--) ..."
   Message from subroutine RHANDS. The storage in COMMON/SINS/

is too small. Increase parameter value INMX in PANDIRA only and recompile.

6. "--- ERROR EXIT --- SUM OF INTER. & IRON A'S IS ZERO."
   Message from subroutine RHANDS. The code is calculating RESIDI = CON(89) and has found that, for the present iteration, the sum of the solution values at the iron and interface points is zero. This will result in a divide by zero so the run is stopped. Try cutting mesh size; check that the iron region is closed; or try running POISSON if applicable.

7. "--- ERROR EXIT --- NAMAX EXCEEDS PROGRAM DIMENSIONS OF (--)"
   Message from subroutine SWIND. Storage exceeded. Increase parameter MAXDIM and recompile.

8. "--- ERROR EXIT --- NROW = MINO(KMAX, LMAX) = (--) EXCEEDS MATRIX DIMENSIONS OF (--)"
   Message from subroutine TRIBES. The storage needed for the matrix inversions is greater than allowed. Change the larger dimension in COMMON/012/. Change the value of IMX in the DATA statement in subroutine TRIBES to new value. Change the appropriate dimension in the DIMENSION statements in subroutines MATINV and MATINP. In subroutine DINO, change the value of variable NDD (statement with label 10). Then recompile POILIB and PANDIRA.

9. "---ERROR EXIT--- MATERIAL CODE .GT. 5"
   Message from subroutine PTABLE. The user has input CON(18) = NPERM negative and has read in a material code greater than 5. See instructions for use of NPERM = CON(18).

B.10.4.2 Messages Ending with "ERROR EXIT"

1. "NAME OF MATERIAL IS LESS THAN OR EQUAL TO 1, OR IS GREATER THAN 11" "---ERROR EXIT---"
   Message from subroutine TABIN. PANDIRA can handle up to 11 different materials. The first is reserved for air and current carrying coils. This message means that the user has input an illegal material number during the input of magnetic region data. Correct input.

2. "THE NUMBER OF INPUT TABLES IS GREATER THAN 4" "---ERROR EXIT---"
   Message from subroutine TABIN. The user has tried to input too many magnetic property tables. Only 4 tables can be input (1 internal and 3 external). Correct input.

3. "GAMMA = H/B, AND B = 0.0" "---ERROR EXIT---"
   Message from subroutine TABIN. The user is trying to input a MTYPE =

3 (H vs B) magnetic material table and has entered a value of B = 0. Since this would result in a divide by zero when converting to γ vs. B, the run is stopped.

4. "YOU HAVE EXCEEDED THE MAXIMUM DIMENSIONS ALLOWED FOR THE GAMMA VS B TABLES" "---ERROR EXIT---"
   Message from subroutine TABIN. The number of data lines in a table of magnetic material properties (GAMMA vs B, MU vs B, or H vs B) is greater than 50. Edit the input table.

B.10.4.3 Messages with "DATA ERROR"

1. "--- DATA ERROR --- ITYPE IS NEGATIVE OR ZERO"
   Message from subroutine POWERS. CON(46) = ITYPE is negative or zero which is illegal. Consult information on CON(46).

B.10.4.4 Messages with "INPUT DATA ERROR"

1. "--- INPUT DATA ERROR --- ILLEGAL CHARACTER"
   followed by a print of the input line. Message from subroutine FREE. The code has found a character it does not recognize in the line printed. Correct line.

2. "--- INPUT DATA ERROR --- NO MANTISSA"
   followed by a print of the input line. Message from subroutine FREE. The code has found an exponent standing alone in the line printed. Correct input.

B.10.5 Messages from TEKPLOT

1. "--- INPUT DATA ERROR --- ILLEGAL CHARACTER"
   Followed by a print of the input line. Message from subroutine FREE. The code has found a character it does not recognize in the line printed. Correct input.

2. "--- INPUT DATA ERROR --- NO MANTISSA"
   Followed by a print of the input line. Message from subroutine FREE. The code has found an exponent standing alone in the line printed. Correct the input line.

3. "NUMBER OF REGIONS ON TAPE35 = (--) LARGER THAN DIMENSION NRGN.RUN STOPPED"
   Message from subroutine TRDUMP. The number of regions to be input exceeds storage. Increase parameter NRGN and recompile.

Chapter B.11

Convergence and Accuracy

The accuracy of POISSON is difficult to assess because POISSON calculates in two dimensions and assumes an infinite third dimension. Thus, the accuracy improves as the magnet gets longer. The accuracy is also dependent on the quality of the B vs.H iron table used. Several users have remarked that the internal table used by POISSON is a little too idealistic, especially at high values of H. We are seeking a better table to include in the program. If you have suggestions for one, please contact us. At low values of H, hystersis effects in real iron make it very difficult to correctly model the magnetic field. The authors of the code feel that POISSON is always accurate to at least 5% in absolute accuracy, but is more accurate when evaluating relative changes on the same magnet.

The less uniform the field distribution, the less accurate the solution. Conformal transformations can be used to improve the accuracy of the solution in cases of very non-uniform field distributions. This is described in Sec. B.13.4 below.

This chapter is not complete. In the future we hope to benchmark the code against some analytically solvabe problem, and to compare it with some other well-known codes.

Chapter B.12

EXAMPLES

B.12.1 QUADRUPOLE MAGNET

This example computes the magnetic fields and harmonics for a quadrupole magnet. By making use of symmetry, we need only calculate one-eighth of the magnet. Figure B.12.1.1 shows the magnet section calculated.

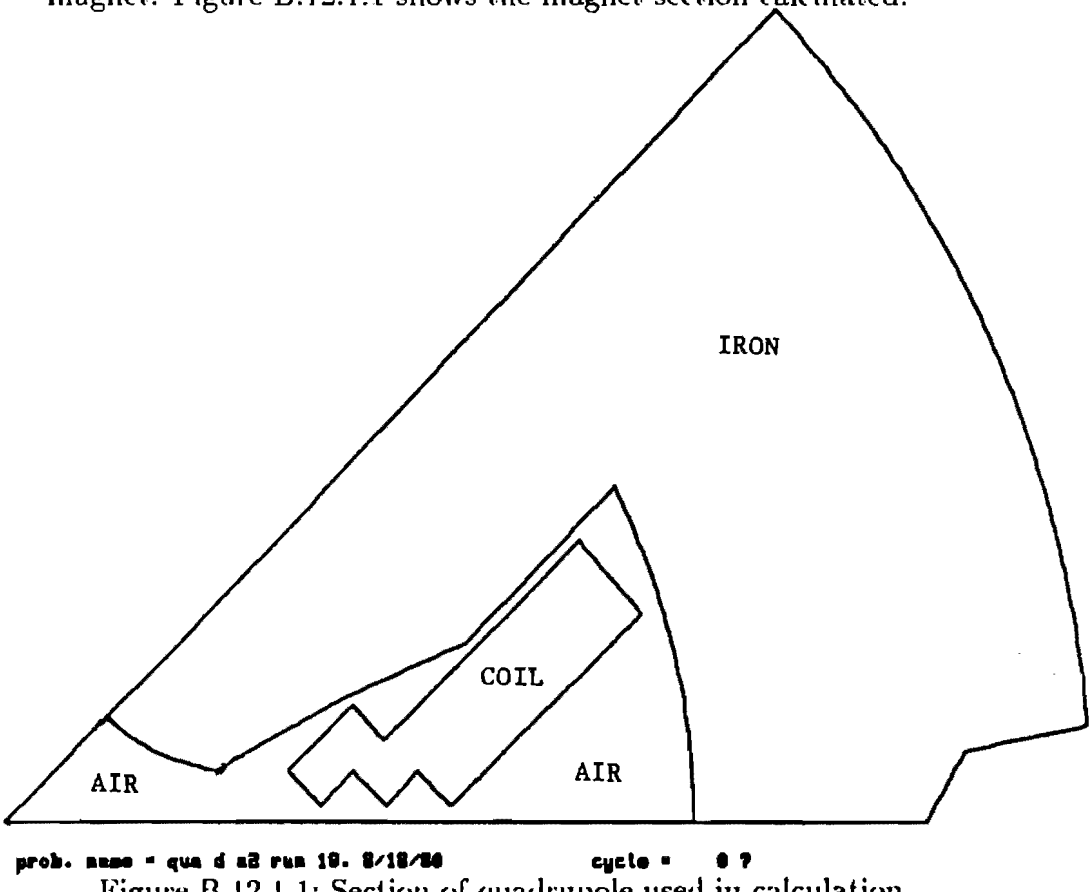


Figure B.12.1.1: Section of quadrupole used in calculation.

```
poisson-pandira test quad a2
$reg nreg=4, xmax=22.5228, ymax=15.92603, dx=.2100, mat=2, npoint=8 $
$po x=0.0, y=0.0 $
$po r=14.25, theta=0.0 $
$po r=19.1883, theta=0. $
$po x=19.9, y=1.4 $
$po x=22.4421, y=1.905 $
$po nt=2, r=22.5228, theta=45.0 $
$po r=2.92, theta=45. $
$po x=0.0, y=0.0 $
$reg mat=1, npoint=8 $
$po x=0.0, y=0.0 $
$po r=14.25, theta=0.0 $
$po x=12.59852, y=6.65882, nt=2 $
$po x=9.5, y=3.5603 $
$po x=7.02870, y=2.47058 $
$po x=4.28368, y=.99522 $
$po nt=3, r=2.92, x=2.0647518, y=2.0647518 $
$po x=0.0, y=0.0 $
$reg cur=9400., npoint =11 $
$po x=11.86, y=5.60 $
$po x=7.80, y=1.60 $
$po x=7.15, y=2.30 $
$po x=5.82, y=1.0 $
$po x=6.48, y=.32 $
$po x=7.15, y=1.0 $
$po x=7.85, y=.32 $
$po x=8.50, y=1.0 $
$po x=9.18, y=.32 $
$po x=13.18, y=4.14 $
$po x=11.86, y=5.60 $
$reg ibound=0, npoint=6, cur=0. $
$po x=0.0, y=0.0 $
$po x=2.0647518, y=2.0647518 $
$po r=22.5228, theta=45. $
$po x=22.4421, y=1.905, nt=2 $
$po x=19.9, y=1.4 $
$po x=19.1, y=0. $
```

Figure B.12.1.2: AUTOMESH input file for quadrupole in Fig. B.12.1.1.

The AUTOMESH input is shown in Fig. B.12.1.2. The file name is QTESTIN. In the input file all lines start in column 2. In the case of the title line, we need a blank to tell AUTOMESH that the input is for a POISSON or PANDIRA problem. All the rest start in column 2 because they are FORTRAN NAMELIST input lines.The first region defines the outer boundary of the problem. The REG NAMELIST tells AUTOMESH that there will be 4 regions (NREG=4), gives the extreme values of x and y (XMAX and YMAX), sets the rough triangle base size (DX), defines the entire problem region to be iron using the internal table (MAT=2), and tells the code there are 8 PO NAMELIST's to follow (NPOINT=8).

The second region input overwrites the first. The second REG NAMELIST sets MAT=1 which corresponds to $\kappa_m = 1$ used for both the air and coil regions. The 8 following PO NAMELISTs define the subregions. The third region input defines the coil inside the 2nd region. Here CUR is set to 9400 amps. The 4th region is a line region. This region is put in to get the proper boundary conditions, IBOUND=0 (DIRICHLET) on the 45° line and outer boundary. CUR is set to 0 because otherwise there would be a current equal to 9400 amps on the line because the previous region value was not overwritten.

```
automesh

 ?type input file name
?qtestin

 region no.  1
 ok

 region no.  2
 ok

 region no.  3
 ok

 region no.  4
 ok
 stop
automesh ctss time      .797 seconds
cpu=   .482     i/o=   .255     mem=   .060

all done
```

Figure B.12.1.3: AUTOMESH ouput to terminal for quadrupole example.

After preparing the input file, the next step is to run AUTOMESH. Figure B.12.1.3 shows the interactions with the terminal for a run made on a CRAY. The user types the executable file name AUTOMESH. The code asks for the input file name and the users reply tells it to use QTESTIN. No other input is necessary and AUTOMESH executes.

The next step is to run LATTICE. The user starts the run by typing the executable file name LATTICE. Figure B.12.1.4 shows what happens at the terminal. The code asks for an input file name and the user replies with TAPE73 which is the tape that AUTOMESH has set up for it. LATTICE prints messages and the

```
lattice

 ?type input file name
?  tape73

 beginning of lattice execution

 dump 0 will be set up for poisson
```

```
 poisson-pandira test quad a2

 type input values for con (?)
?*6 0 *32 2 *46 4 *111 11 1.86 45.  2.92 s

 elapsed time=   1.7 sec.
0iteration converged

 elapsed time=   3.3 sec.

 generation completed

 dump number 0 has been written on tape35.
 stop
lattice ctss time     4.231 seconds
cpu=   3.655     i/o=   .439     mem=   .137

all done
```

Figure B.12.1.4: User terminal instruction during LATTICE run of quadrupole example.

problem title followed by a request for CON inputs. At this point the user inputs seven changes. The user types:

*6 0 *32 2 *46 4 *111 11 1.86 45. 2.92 s

As it happens, none of these CON changes are necessary for the LATTICE run, but can be entered here for later use by POISSON. They could as well been entered after POISSON asks for CON input. The meaning of the CON values input is as follows:

*6 0 sets CON(6)=MODE equal to 0. This says that some iron in the problem has finite, nonconstant permeability.

*32 2 sets CON(32)=IPRINT to 2 which will cause POISSON to write the γ vs B tables used to the file OUTPOI and also to print a map of | B | in the iron.

*46 4 sets CON(46)=ITYPE equal to 4 which tells the code that the problem is a symmetrical quadrupole.

*111 11 1.86 45. 2.92 s changes CON's 111 thru 114. These values all give information needed by the code to do a harmonic analysis of the quadrupole field. CON(111)=NPTC=11 says use 11 points on a circular arc, CON(112)=RINT=1.86 says the radius of the arc is 1.86cm., CON(113)=ANGLE=45. indicates the length of the arc is to be 45°, and finally CON(114)=RNORM=2.92 tells the code to use a normalization radius of 2.92cm. The final s in the CON input line tells the code there is no more input. The code then continues execution and writes dump 0 to TAPE35.

At this point the user can use TEKPLOT to examine the mesh or to look at the region outlines. Figure B.12.1.5 shows how the user would use TEKPLOT to see the region outlines.

After the executable file name TEKPLOT is typed, the code asks for input for five parameters. The reply **s** indicates that the user wants to use the default values which are all zero. This will cause only the problem outline to be drawn. After printing the problem name the code asks for limits on the region to be drawn. The user reply, 0. 23., which says plot a square 23cm on a side with the lower right corner at the origin. No **s** is required for the input because all four values are given and, in this case, the code knows that it should continue. Next the user replies **GO**. After hitting the carriage return the code draws the outline shown in Fig. B.12.1.1, however, without the printing in the regions. To end TEKPLOT, the user hits the carriage return and, when the code asks for more input data, type **-1s**. The terminal output is shown in Fig. B.12.1.6.

```
tekplot
 ?type input data- num, itri, nphi, inap, nswxy,
?s
 input data
 num=   0     itri=   0     nphi=   0     inap=   0     nswxy=   0

 plotting prob.  name = poisson-pandira test   quad a2      cycle=   0

 type input data- xmin, xmax, ymin, ymax,
?0.  23.  0.  23.
 input data
xmin=   0.000    xmax=   23.000     ymin=   0.000     ymax=   23.000

 ?type go or no
?go
```

Figure B.12.1.5: Using TEKPLOT to examine the problem boundaries.

```
 type input data- num, itri, nphi, inap, nswxy,
?-1s
tekplot ctss time        1.084 seconds
cpu=   .216     i/o=   .793     mem=   .076

all done
```

Figure B.12.1.6: Terminal display after carriage return.

To run POISSON, the user types POISSON. (See Fig. B.12.1.7 for the terminal interaction.) The code asks where it should get input and the user replies TTY, indicating it will be entered at the terminal. The code asks for a dump number and is told 0. The code then reads dump 0 from TAPE35, writes the problem name and asks for CON input. Since all necessary changes were made in LATTICE and passed to POISSON through TAPE35, the user replies with an **s**. The code then solves the problem, writes the solution as dump number 1 to TAPE35, and prompts the user for another dump number. The user answers **-1s** to end the run.

```
poisson

 type ''tty'' or input file name
?  tty
```

```
 ?  type input value for num
?  0
 beginning of poisson execution from dump number 0

 prob.  name = poisson-pandira test quad a2

 ?type input values for con(?)
?  2

 elapsed time  =     1.6 sec.
0  cycle      amin          amax      residual-air   eta-air  rhoair  xjfact
                            gmax      residual-iron  eta-iron rhofe
0      0   0.0000e+00   0.0000e+00    1.0000e+00   1.0000   1.9000  1.0000
                        4.0000e-03    1.0000e+00   1.0000   1.0000
0    100                      rhoair optimized    0.9975   1.9763   lambda =   9.9993e-01
0    100   0.0000e+00   1.3931e+04    2.6722e-02   0.9975   1.9763  1.0000
                        9.1430e-04    1.4021e-02   1.0010   1.0000
0    200                      rhoair optimized    0.9873   1.9871   lambda =   9.9994e-01
0    200   0.0000e+00   4.5690e+04    1.4151e-02   0.9873   1.9781   1.0000
                        1.2039e-02    9.7234e-03   0.9911   1.0000
0    400                      rhoair optimized    0.9825   1.9775   lambda =   9.9994e-01
0    400   0.0000e+00   5.7708e+04    5.3831e-04   0.9825   1.9775   1.0000
                        6.1299e-03    1.2520e-03   0.9931   1.0000
0    800                      rhoair optimized    0.9923   1.9821   lambda =   9.9996e-01
0    800   0.0000e+00   5.8241e+04    3.3245e-06   0.9923   1.9821   1.0000
                        5.0758e-03    1.3139e-04   0.9947   1.0000
0   1600                      rhoair optimized    0.9947   1.9863   lambda =   9.9998e-01
0   1600   0.0000e+00   5.8251e+04    3.1264e-08   0.9947   1.9863   1.0000
                        4.9703e-03    1.9238e-06   0.9948   1.0000
0   1860   0.0000e+00   5.8251e+04    6.8252e-09   0.9953   1.9863   1.0000
                        4.9691e-03    4.8982e-07   0.9947   1.0000

 solution converged in 1860 iterations

 elapsed time =    53.0 sec.

dump number 1 has been written on tape35.

 type input value for num
?  -1s
stop
poisson ctss time   61.292   seconds
cpu=   55.365     i/o=   2.552     mem=   3.376

all done
```

Figure B.12.1.7: Terminal display for POISSON RUN.

During the running of the problem AUTOMESH has produced an output file called OUTAUT, LATTICE has produced OUTLAT, and POISSON has produced OUTPOI. These files give various information on the various runs. Usually, as in this case, OUTAUT and OUTLAT are of little interest because they give mainly information on the mesh. They are useful when there are difficulties with the meshing. However, the file OUTPOI contains information on the solution.

In this case the OUTPOI file contains 1376 136-character lines. OUTPOI contains a list of the CON values used for the run, a table giving the magnetic characteristic used for material 2, and copy of the terminal output. Since the user asked for it (CON(32)=2), OUTPOI also contains a printed map of the fields in the iron. A portion of the map is shown in Fig. B.12.1.8. The portion is recognizable as being the top right corner of the input problem geometry. The first column on the

	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80
88 u																				
1																				
87 u																0.00				
1															0.27	0.19				
86 u															0.27	0.16	0.17			
1															0.42	0.34	0.31	0.42		
85 u														0.43	0.32	0.27	0.30	0.42		
1														0.56	0.49	0.45	0.48	0.55		
84 u														0.58	0.47	0.42	0.42	0.47	0.57	
1													0.85	0.71	0.64	0.60	0.59	0.62	0.68	
83 u												0.85	0.72	0.63	0.56	0.54	0.56	0.62	0.71	
1												1.01	0.90	0.81	0.75	0.72	0.73	0.76	0.82	0.94
82 u												1.03	0.90	0.79	0.72	0.68	0.68	0.71	0.76	0.83
1												1.17	1.06	0.98	0.91	0.87	0.86	0.87	0.92	0.99
81 u											1.20	1.07	0.97	0.89	0.83	0.81	0.82	0.86	0.92	1.00
1										1.51	1.34	1.24	1.15	1.08	1.03	1.00	1.00	1.02	1.06	1.13
80 u										1.51	1.35	1.24	1.14	1.06	1.00	0.96	0.95	0.97	1.01	1.07
1										1.68	1.55	1.43	1.33	1.25	1.19	1.15	1.14	1.14	1.17	1.21
79 u									1.71	1.56	1.43	1.32	1.23	1.16	1.12	1.10	1.10	1.12	1.16	1.22
1								2.06	1.86	1.74	1.62	1.52	1.43	1.36	1.31	1.29	1.28	1.29	1.31	1.35
78 u								2.06	1.88	1.75	1.62	1.51	1.41	1.34	1.28	1.25	1.24	1.24	1.27	1.31
1								2.25	2.10	1.95	1.82	1.71	1.62	1.54	1.49	1.45	1.42	1.42	1.43	1.46
77 u							2.28	2.11	1.96	1.82	1.71	1.60	1.52	1.46	1.41	1.39	1.38	1.39	1.42	1.46
1							2.45	2.31	2.16	2.03	1.92	1.82	1.73	1.66	1.61	1.58	1.56	1.56	1.57	1.60
76 u							2.50	2.32	2.17	2.03	1.91	1.80	1.71	1.64	1.58	1.55	1.53	1.53	1.54	1.57
1						2.90	2.67	2.52	2.38	2.25	2.13	2.02	1.93	1.85	1.79	1.75	1.72	1.70	1.70	1.72
75 u					2.90	2.69	2.54	2.39	2.25	2.12	2.01	1.91	1.83	1.76	1.72	1.69	1.67	1.67	1.69	1.71
1					3.12	2.94	2.77	2.61	2.47	2.35	2.23	2.13	2.05	1.98	1.92	1.88	1.86	1.85	1.85	1.86
74 u					3.16	2.96	2.78	2.62	2.47	2.34	2.22	2.11	2.02	1.95	1.89	1.85	1.83	1.82	1.82	1.83
1					3.36	3.19	3.02	2.86	2.71	2.58	2.45	2.35	2.25	2.17	2.11	2.06	2.02	2.00	1.99	1.99
73 u				3.42	3.21	3.03	2.86	2.71	2.56	2.44	2.33	2.23	2.15	2.08	2.03	1.99	1.97	1.96	1.97	1.98

Figure B.12.1.8: Portion of the printed map giving magnetic field values in kilogauss in the iron of the quadrupole.

left gives a logical K coordinate at the intersection of a coordinate pair there are two values. These are the magnitudes of the field in kG in the upper (u) and lower (l) triangles associated with that (K, L) point.

The POISSON output file contains a table giving the values of the vector potential A, and related quantities B_x, B_y, $|B|$, dB_y/dy, dB_y/dx, along the x axis of the problem. The latter five quantities are obtained from the vector potential by a least square fit. The goodness of the fit is indicated by the value of AFIT given in the rightmost column of the table.

The final printout in OUTPOI gives information on the harmonic analysis. This information is shown in Fig. B.12.1.9. The angle, (x, y) coordinates, nearest (K, L) coordinates, and the interpolated vector potential are given for points on the arc. Also given are tables of coefficents for the harmonic expansion of the vector potential and the magnetic field.

TEKPLOT can be used to examine the field pattern in the quadrupole. Figure B.12.1.10 shows the terminal interactions. The user types TEKPLOT to initiate the run. When asked for input data, the reply sets NUM=1 to tell the code to use dump 1 from TAPE35, sets ITRI=0 since a drawing of the mesh is not wanted, and sets NPIII=35 to cause 35 field lines to be drawn. The final s in the input line leaves INAP and NSWXY with their 0 default values. Next the user is asked to enter the limits of the plotting region and replies: 0. 23. 0 or 23., defining the plot rectangle. On receiving go, the code plots Fig. B.12.1.11. TEKPLOT is terminated by hitting a carriage return followed by -1s as described in connection with Fig. B.12.1.6.

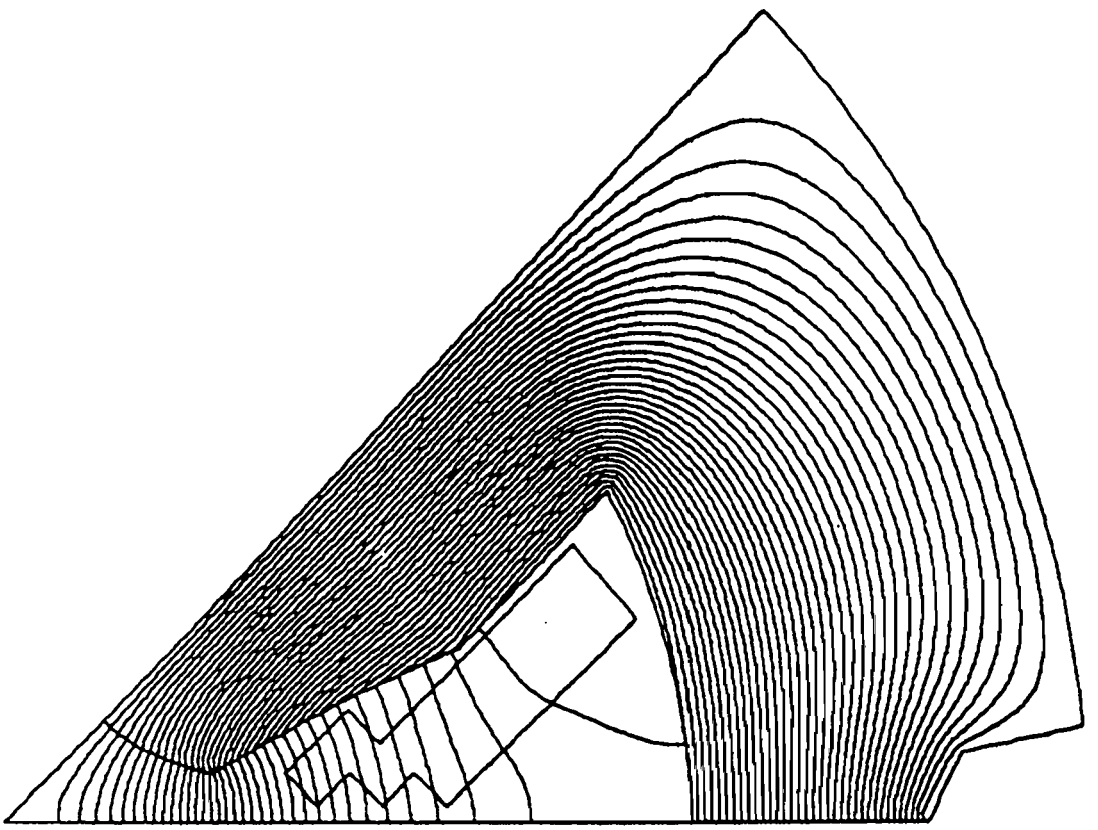
```
0table for interpolated points
0        n     angle    x coord    y coord    kf    lf      vec.pot.
         1    0.0000     1.8600     0.0000    10     1   4.74266e+03
         2    4.5000     1.8543     0.1459    10     2   4.68434e+03
         3    9.0000     1.8371     0.2910    10     2   4.51082e+03
         4   13.5000     1.8086     0.4342     9     3   4.22631e+03
         5   18.0000     1.7690     0.5748    10     4   3.83775e+03
         6   22.5000     1.7184     0.7118     9     5   3.35463e+03
         7   27.0000     1.6573     0.8444     9     6   2.78884e+03
         8   31.5000     1.5859     0.9718     9     6   2.15430e+03
         9   36.0000     1.5048     1.0933     8     7   1.46658e+03
        10   40.5000     1.4144     1.2080     8     7   7.42792e+02
        11   45.0000     1.3152     1.3152     8     8   3.54844e-01
1table for vector potential coefficients
0normalization radius =   2.92000
0        a(x,y) = re( sum (an + i bn) * (z/r)**n )
0         n        an              bn            abs(cn)
0         2     1.1691e+04     0.0000e+00     1.1691e+04
0         6    -1.2543e+01     0.0000e+00     1.2543e+01
0        10     9.3097e+00     0.0000e+00     9.3097e+00
0        14    -5.7177e+01     0.0000e+00     5.7177e+01
0        18     2.4654e+02     0.0000e+00     2.4654e+02

1table for field coefficients
0normalization radius =   2.92000
0        (bx - i by) = i * sum n*(an + i bn)/r * (z/r)**(n-1)
0         n     n(an)/r        n(bn)/r       abs(n(cn)/r)
```

```
0        2    8.0072e+03    0.0000e+00    8.0072e+03
0        6   -2.5772e+01    0.0000e+00    2.5772e+01
0       10    3.1882e+01    0.0000e+00    3.1882e+01
0       14   -2.7414e+02    0.0000e+00    2.7414e+02
0       18    1.5198e+03    0.0000e+00    1.5198e+03
```

Figure B.12.1.9: Harmonic analysis in file OUTPOI.

```
tekplot

 ?type input data- num, itri, nphi, inap, nswxy,
?1 0 35 s
 input data
 num=  1   itri=  0   nphi=  35   inap=  0   nswxy=  0

 plotting prob.  name = poisson-pandira test quad a2

 ?type input data- xmin, xmax, ymin, ymax
?0. 23. 0. 23.
 input data
 xmin=  0.000   xmax=  23.000   ymin=  0.000   ymax=  23.000

 type go or no
?go
```

Figure B.12.1.10: Terminal display for TEKPLOT run to get field lines.



prob. name = poisson-pandira test quad a2

Figure B.12.1.11: Magnetic field lines in quadrupole.

B.12.2 POISSON Example – Electrostatic Problem

This example computes the potential distribution inside one quadrant of a coaxial cylinder. The inner conductor has an odd-shaped bump on it to provide some spice to the calculation.

The AUTOMESH input file COAXCYL is shown in Fig. B.12.2.1. Since this problem will run using POISSON, the first column in the first line is blank. The first REG NAMELIST tells the codes that there are two regions, sets the triangle size, defines the maximum limits of the problem, and gives the number of PO cards to be read for the first region. The first region is one quadrant of a circle.

The second REG NAMELIST card sets MAT=0 to tell the code that the points inside the region are not in the problem. The variable CUR is set to 1000.; in an electrostatic problem, CUR corresponds to a fixed potential on the boundary of the region. To cause the problem to set this potential, IBOUND is set to −1. The eight PO NAMELIST cards following define the central excluded region.

```
 1  coaxial cylinder --- electrostatic example
 2  $reg nreg=2,dx=.1,dy=.1,xmax=5.,ymax=5.,npoint=4 $
 3  $po x=0., y=0. $
 4  $po x=0., y=5. $
 5  $po nt=2, r=5., theta=0. $
 6  $po x=0., y=0. $
 7  $reg mat=0, cur=1000., ibound=-1,npoint=8 $
 8  $po x=0., y=0. $
 9  $po x=0., y=2. $
10  $po nt=2, r=2., theta=60. $
11  $po r=3., theta=60. $
12  $po nt=2, r=3., theta=30. $
13  $po r=2., theta=30. $
14  $po nt=2, r=2., theta=0, $
15  $po x=0., y=0. $
```

Figure B.12.2.1: Input to AUTOMESH for electrostatic problem.

Figure B.12.2.2 shows the AUTOMESH run as seen from the terminal. We type the executable filename, **automesh**, followed by the input file name, **coaxcyl**. The code requests an input filename but does not wait for it because it was on the execute line.

```
automesh coaxcyl
 ?type input file name
region no.  1
```

```
ok
region no.  2
ok
stop
automesh ctss time      .219   seconds
cpu=      .146   sys=      .017   i/o+memory=      .056
all done
```

Figure B.12.2.2: AUTOMESH execution log on CRAY.

```
lattice tape73
 ?type input file name
 beginning of lattice execution
 dump 0 will be set up for poisson
  coaxial cylinder --- electrostat
 ?type input values for con(?)
?*21 0 1 0 1 *46 1 *66 0 s
 elapsed time=     0.5 sec.
0iteration converged
 elapsed time=     0.8 sec.
 generation completed
 dump number 0 has been written on tape35.
 stop
lattice ctss time     1.046   seconds
cpu=      .879   sys=      .026   i/o+memory=   .141
all done
```

Figure B.12.2.3: LATTICE execution log on CRAY.

The LATTICE run is shown in Fig. B.12.2.3. The execution filename, **lattice**, is typed followed by **tape73**, the AUTOMESH output file. The code requests an input filename but continues without pause because the needed name was on the execute line. It requests CON changes and is given six changes. The portion *21 0 1 0 1 sets the boundary conditions to be used in the problem. The CON(21) and CON(23) are set to 0, indicating Dirichlet conditions on the upper right curved boundary. In this case, the Dirichlet conditions mean that the surface is an equipotential. Con(22) and CON(24) are set to 1, indicating a Neumann condition on the bottom and left sides. The Neumann conditions means the equipotentials are perpendicular to the boundary. The input *46 1 sets CON(46)=ITYPE=1, indicating there is no symmetry. Setting CON(66)=XJFACT=0 is necessary because it tells the code that it is doing an electrostatic problem. After reading the "**s**" on the CON change line, LATTICE knows all changes have been made and proceeds to run.

```
tekplot
 ?type input data- num, itri, nphi, inap, nswxy,
?0 1 s
 input data
 num=  0   itri=  1   nphi=  0   inap=  0   nswxy=  0
 plotting prob.  name = coaxial cylinder --- electrostat cycle=  0
 ?type input data- xmin, amax, ymin, ymax,
?s
 imput data
 xmin=    0.000  xmax=    5.000  ymin=    0.000  ymax=  5.000
 type go or no
?go
```

Figure B.12.2.4: TEKPLOT execution log on CRAY.

Next, we use TEKPLOT to look at the mesh. See Fig.B.12.2.4. Type **tekplot** and the code asks for: a dump number (NUM), if a mesh plot is desired (ITRI), how many equipotential lines (NPHI) are required, if minimum and maximum values are to be entered for the equipotentials (INAP), and if the X and Y axes are to be interchanged (NSWXY). The reply shown says use dump 0 (NUM=0) and draw the mesh (ITRI=1). The "**s**", tells the code to set NPHI=0 (no lines), INAP=0 (no input), and NSWXY (leave the axes alone). The next request asks for the limits of the plotting area and the reply, "**s**", tells the code to use the problem limits. On receiving "**go**", TEKPLOT runs and produces Fig. B.12.2.5. TEKPLOT is ended by two carriage returns followed by a "**-1s**" in answer to the request for a dump number.

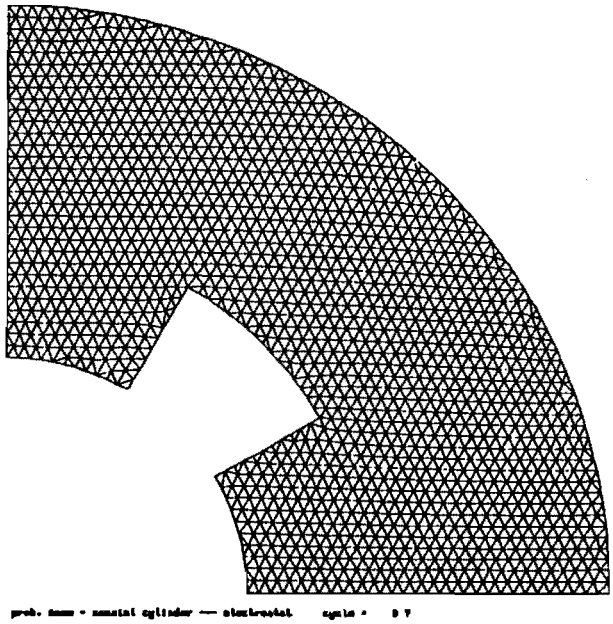


Figure B.12.2.5: TEKPLOT output for electrostatic problem showing mesh and regions.

The word **poisson** is typed to initialize execution of POISSON. The code asks for an input file name and (after receiving TTY) for a dump number (see Fig. B.12.2.6). After being told to use dump 0, the code requests CON changes. The reply *43 31 1 43 *s* sets limits on the region that the potential and its gradients are to be computed and written to the OUTPOI file. CON(43)=KTOP is set to 31, CON(44) = LMIN is set to 1 (its default value), and CON(45) = LTOP is set to 43. Since CON(42) = KMIN = 1 by default, the code will print out potentials and gradients for those points in the mesh with logical coordinates in the rectangle defined by (1, 1), (1, 43), (31, 43), 31, 1).

While executing, POISSON prints some information at the terminal: the cycle number, the current minimum and maximum values in the solutions matrix, the current residual, the current rate of convergence, and the current overrelaxation factor.

```
poisson
 ?type ''tty'' or input file name
?tty
 ?type input value for num
?0
 beginning of poisson execution from dump number 0
 prob.  name = coaxial cylinder --- electrostat
 type input values for con(?)
?*43 31 1 43 s
 elapsed time=     1.0 sec.
0  cycle     amin        amax    residual-air  eta-air rhoair  xjfact
0      0  0.0000e+00  0.0000e+00   1.0000e+00  1.0000  1.9000  0.0000
0    100                      rhoair optimized  0.9126  1.8998  lambda = 9.9864e-01
0    100  0.0000e+00  9.9322e+02   2.4187e-05  0.9126  1.8998  0.0000
0    150  0.0000e+00  9.9321e+02   2.7238e-07  0.9136  1.8998  0.0000
 solution coaverged in 150 iterations
 elapsed time=   1.5  sec.
 dump number 1 has been written on tape35
 ?type input value for num
?-1s
 stop
poisson ctss time=     3.317 seconds
cpu=     2.012   sys=     .028   i/o memory=     1.277
all done
```

Figure B.12.2.6: POISSON execution log on CRAY for electrostatic problem.

After POISSON has written dump 1 on TAPE35, it is terminated by typing "-1 s" in reply to the request for another dump number.

```
tekplot
 ?type input data- num, itri, nphi, inap, nswxy,
?1 0 50 s
 input data
```

```
 num=  1   itri=  0   nphi=  50   inap=  0   nswxy=  0
 plotting prob.  name = coaxial cylinder --- electrostat    cycle=  150
 ?type input data- xmin, xmax, ymin, ymax
?s
 input data
 xmin=  0.000   xmax=  5.000   ymin=  0.000   ymax=  5.000
 ?type go or no
?go
```

Figure B.12.2.7: TEKPLOT execution log on CRAY for equipotential plot.

To see if we solved the correct problem, we use TEKPLOT to look at the equipotentials. This time (Fig. B.12.2.7), we set NUM = 1 for dumpt, ITRI = 0 to avoid drawing the mesh, and NPHI = 50 to get 50 equipotential lines. The final "s" sets the remaining two values to default zeroes. The "s" in response to the request for maximum and minimum limits causes TEKPLOT to plot the whole problem area. After receiving "go" TEKPLOT plots, Fig. B.12.2.8, the equipotential lines are evenly spaced on the bottom and left boundaries as they should be.

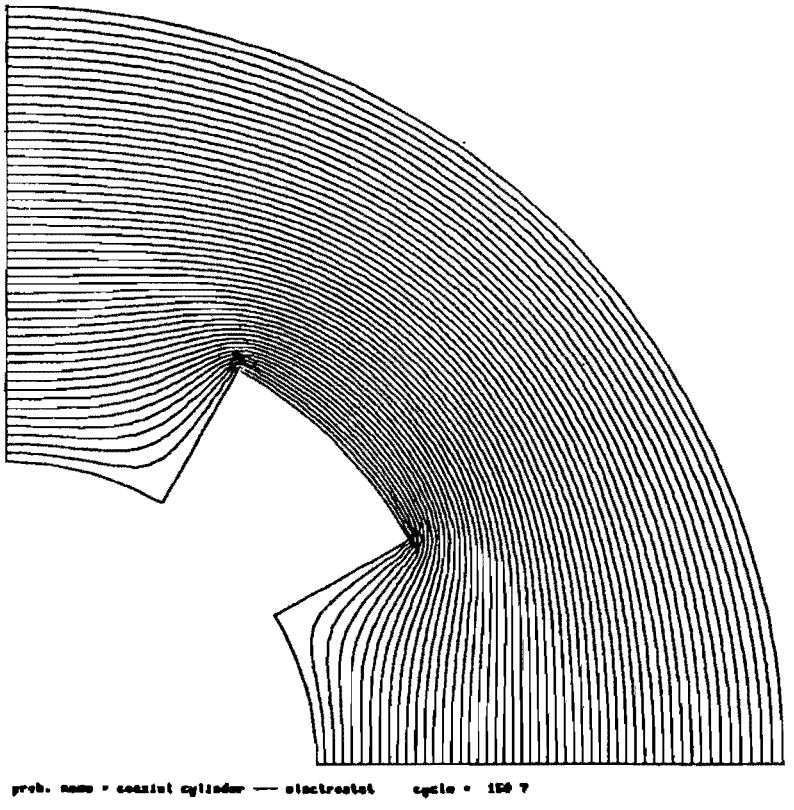


Fig. B.12.2.8: TEKPLOT output for electrostatic problem.

The file OUTPOI contains more information on the solution. OUTPOI contains a list of the CON values used in the solution, a copy of the iteration history, and a table of potential and gradients values for the problem (Points in the K–L region specified by using CONs 42 to 45). The initial part of this table is shown in Fig. B.12.2.9. The K and L columns are the logical coordinates of a point, while the X and Y columns give the physical position of the point. V is the scalar potential the code found for the point, and EX, EY, and ET are the X and Y components of the field and the total field. Vfit gives the difference between the potential calculated at the mesh point and the potential calculated by the code using a least square fit of solution values for the point and its neighbors.

```
1 least square edit of problem, cycle 150
  none   symmetry type
0   k  l   v(scalar)      x         y       ex(v/cm) ey(v/cm) et(v/cm)  vfit
0  21  1  1.000000e+03  2.00000  0.00000  357.848    3.302  357.863  -5.4e-02
0  22  1  9.649683e+02  2.10000  0.00000  343.879   -0.174  343.879   1.1e-02
0  23  1  9.310212e+02  2.20000  0.00000  335.778   -0.124  335.778   8.0e-03
0  24  1  8.976641e+02  2.30000  0.00000  331.978   -0.124  331.978   7.6e-03
0  25  1  8.645221e+02  2.40000  0.00000  331.328   -0.109  331.328   6.4e-03
0  26  1  8.313289e+02  2.50000  0.00000  332.854   -0.090  332.854   5.3e-03
0  27  1  7.979091e+02  2.60000  0.00000  335.730   -0.072  335.730   4.2e-03
0  28  1  7.641628e+02  2.70000  0.00000  339.289   -0.056  339.289   3.3e-03
0  29  1  7.300481e+02  2.80000  0.00000  343.009   -0.042  343.009   2.4e-03
0  30  1  6.955706e+02  2.90000  0.00000  346.490   -0.028  346.490   1.6e-03
0  31  1  6.607690e+02  3.00000  0.00000  349.449   -0.017  349.449   1.1e-03
0  21  2  1.000000e+03  1.99730  0.10470  351.841   18.521  352.328   1.1e-01
0  22  2  9.791931e+02  2.05694  0.10380  345.541   14.004  345.825   4.7e-02
0  23  2  9.465786e+02  2.15265  0.10276  336.397    8.125  336.495   8.1e-03
0  24  2  9.133672e+02  2.25222  0.10199  331.400    3.618  331.419   1.7e-03
0  25  2  8.803115e+02  2.35226  0.10139  329.952    0.414  329.952   1.0e-03
0  26  2  8.472403e+02  2.45238  0.10090  331.032   -1.735  331.036   8.0e-04
0  27  2  8.139690e+02  2.55250  0.10047  333.744   -3.038  333.758   6.0e-04
0  28  2  7.803771e+02  2.65262  0.10007  337.357   -3.670  337.377   4.7e-04
0  29  2  7.464058e+02  2.75273  0.09970  341.291   -3.779  341.312   2.9e-04
0  30  2  7.120465e+02  2.85284  0.09934  345.097   -3.499  345.115   2.6e-04
0  31  2  6.773290e+02  2.95294  0.09899  348.447   -2.994  348.460   2.9e-05
0  21  3  1.000000e+03  1.98900  0.20910  342.170   36.170  344.555  -1.0e-02
0  22  3  9.818349e+02  2.10270  0.20624  331.352   20.086  331.960  -9.9e-03
0  23  3  9.285665e+02  2.20404  0.20452  326.587    9.747  326.732  -3.7e-04
0  24  3  8.958943e+02  2.30435  0.20325  325.540    2.430  325.549   8.8e-04
0  25  3  8.631924e+02  2.40461  0.20222  327.118   -2.591  327.128   6.8e-04
0  26  3  8.302357e+02  2.50488  0.20133  330.417   -5.771  330.468   7.2e-04
0  27  3  7.988969e+02  2.60513  0.20052  334.686   -7.475  334.769   5.8e-04
0  28  3  7.631136e+02  2.70536  0.19975  339.307   -8.017  339.401   3.4e-04
0  29  3  7.288762e+02  2.80558  0.19902  343.803   -7.672  343.889   2.1e-04
0  30  3  6.942161e+02  2.90578  0.19832  347.805   -6.688  347.869   2.0e-04
0  31  3  6.591948e+02  3.00597  0.19765  351.071   -5.279  351.111  -2.3e-05
0  21  4  1.000000e+03  1.97540  0.31290  322.967   50.768  326.932   2.1e-02
0  22  4  9.738058e+02  2.05749  0.30989  318.497   33.267  320.229  -3.1e-04
0  23  4  9.428293e+02  2.15606  0.30740  316.031   17.204  316.499   3.4e-05
0  24  4  9.109878e+02  2.25627  0.30548  316.806    5.222  316.849   3.1e-04
```

Figure B.12.2.9: Part of OUTPOI file for electrostatic problem.

B.12.3 PANDIRA Example – Permanent Magnet Solenoid

The layout for this problem is shown in Fig. B.12.3.1 and the AUTOMESH input file, which we have chosen to call "**sol1**," is shown in Fig. B.12.3.2. Note that for POISSON and PANDIRA problems with cylindrical symmetry, the Y coordinate corresponds to the direction of the cylindrical axis, and the X-coordinate corresponds to the radial direction. The problem involves six regions. The first region defines the boundary of the problem. The material is assumed to be air.

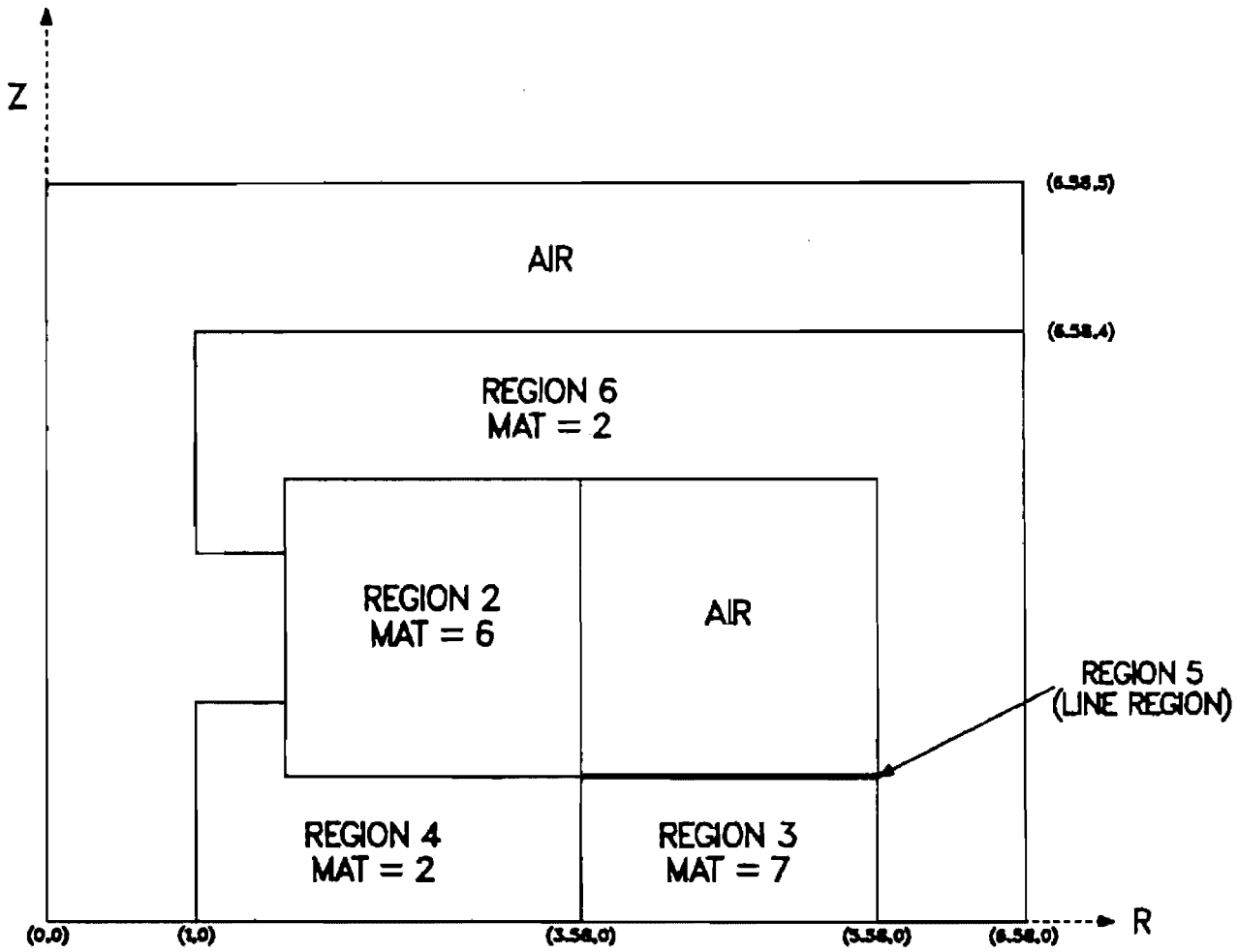


Figure B.12.3.1: Physical layout of regions for solenoidal permanent magnet.

The first REG NAMELIST line tells the codes that there are six regions (NREG=6), that triangle sizes are to be changed at XREG1=3.58 and at YREG1=3, and to define the triangle size (DX) and the maximum limits of the problem (XMAX, YMAX). The following PO NAMELIST lines define the boundary of the first region. In this case and also in the succeeding regions, extra points are defined on the boundary. These points ensure that AUTOMESH and LATTICE will place nodes at locations that will be needed later in the mesh assembly. The use of "sure points" is not always necessary, but they may often be used to get around mesh generation problems.

```
pm magnet solenoid
$reg nreg=6,npoint=10,dx=.102,xreg1=3.58,yreg1=3.,xmax=6.8,ymax=6. $
$po x=0.,y=0. $
$po x=1.,y=0. $
$po x=3.58, y=0. $
$po x=5.58, y=0. $
$po x=6.58, y=0. $
$po x=6.58, y=4. $
$po x=6.58, y=5. $
$po x=3.58, y=5. $
$po x=0., y=5. $
$po x=0., y=0. $
$reg mat=6,npoint=7 $
$po x=1.58, y=1. $
$po x=3.58, y=1. $
$po x=3.58, y=3. $
$po x=1.58, y=3. $
$po x=1.58, y=2.5 $
$po x=1.58, y=1.5 $
$po x=1.58, y=1. $
$reg mat=7, npoint=5 $
$po x=3.58, y=0. $
$po x=5.58, y=0. $
$po x=5.58, y=1. $
$po x=3.58, y=1. $
$po x=3.58, y=0. $
$reg mat=2, npoint=7 $
$po x=1., y=0. $
$po x=3.58, y=0. $
$po x=3.58, y=1. $
$po x=1.58, y=1.0 $
$po x=1.58, y=1.5 $
$po x=1.0, y=1.5 $
$po x=1., y=0. $
$reg mat=1, den=6400., npoint=2 $
```

```
36  $po x=3.58, y=1. $
37  $po x=5.58, y=1. $
38  $reg npoint=12, mat=2, den=0. $
39  $po x=5.58, y=0. $
40  $po x=6.58, y=0. $
41  $po x=6.48, y=4. $
42  $po x=3.58, y=4. $
43  $po x=1.0, y=4. $
44  $po x=1., y=2.5 $
45  $po x=1.58, y=2.5 $
46  $po x=1.58, y=3. $
47  $po x=3.58, y=3. $
48  $po x=5.58, y=3. $
49  $po x=5.58, y=1. $
50  $po x=5.58, y=0. $
```

Figure B.12.3.2: AUTOMESH input file **soll** for permanent manget solenoid.

The second region in the problem has a material number equal to 6; this indicates an anisotropic magnetic material. Region 3 has a material number equal to 7, another anisotropic material. In this problem, both materials will be the same, except that they have different easy axis directions. Regions 4 and 6 both have material number 2, indicating that the program is to assume iron corresponding to the internal table in PANDIRA. Region 5 is a line region, which has a current density of 6400 A/m and acts as the source of the magnetic field in the problem. Note that in the sixth REG NAMELIST, DEN is set to 0. Otherwise, the sixth region would have a 6400 A/m^2 current density.

<u>automesh soll</u>

```
?type input file name
```

Note: Since we entered the file name on the first line, we don't need to answer this question

```
region no.  1
ok
region no.  2
ok
region no.  3
ok
region no.  4
ok
region no.  5
ok
region no.  6
ok
stop
```

```
automesh ctss time      .346  seconds
cpu=    .266     sys=    .021     i/o+memory=    .059
all done
```

Figure B.12.3.3: Interaction with AUTOMESH on the CRAY for solenoid problem.

After making up the input file, the user types the AUTOMESH executable filename, **automesh**, followed by the input file name, **soll**; see Fig. B.12.3.3. The program asks the user to type the input file name and gets it from the execute line so no futher answer is needed. AUTOMESH executes producing the terminal output shown in Fig. B.12.3.3. The user now types the LATTICE executable filename, **lattice**, followed by **tape73**. See Fig. B.12.3.4. LATTICE starts to execute by asking for an input file name. However, it does not stop since it had its answer (TAPE73) from the execute line. LATTICE then asks the user for changes in the problem CONs. In the reply, the user changes eight CON values. The input *21 0 0 0 0 sets CON(21), CON(22), CON(23), and CON(24) to 0. These are the boundary conditions as the upper, lower, right, and left sides. The value 0 means that the boundary has a Dirichlet condition; that is, the boundary is a magnetic field line.

```
lattice tape73
 ?type input file name
 beginning of lattice execution
 dump 0 will be set up for poisson
  pm magnet solenoid
 ?type input values for con(?)
?*21 0 0 0 0 *6 0 *19 1 *101 1 *81 0 s
 elapsed time=     0.5 sec.
0 interation converged
 elapsed time=     0.7 sec.
 generation completed
 dump number 0 has been written on tape35.
 stop
lattice ctss time      .984 seconds
cpu=    .766     sys=    .023     i/o+memory=    .196
all done
```

Figure B.12.3.4: Interaction with LATTICE on CRAY for solenoid problem.

The entry *6 0 sets CON(6)=MODE=0. This tells the code that some materials in the problem will have finite and variable permeability (μ). Setting CON(19)=ICYLIN=1 tells the code it is dealing with a problem having cylindrical symmetry about the vertical axis. The statement *101 1 sets CON(101)=IPERM; this will tell PANDIRA to initialize the vector potential using currents in the SOURCE vector. In this case, the SOURCE vector is set by the surface current of region 5. The entry *81 0

sets CON(81)=NOTE=0. This sets the order of the point relaxation. NOTE=0 is required for all PANDIRA problems. The final "s" tells LATTICE there are no further changes.

```
tekplot
 type input data- num, itri, nphi, inap, nswxy,
?0 1 s
 num=   0     itri=   1     nphi=   0     inap=   0     nswxy=   0
 plotting prob.  name = pm magnet solinoid
 ?type input data- xmin, xmax, ymin, ymax,
?s
 input data
 xmin=   0.000     xmax=   6.580     ymin=   0.000     ymax=   5.000
 ?type go or no
?go
```

Figure B.12.3.5: Interaction with TEKPLOT on CRAY for solenoid problem.

At this point, TEKPLOT can be used to look at the mesh. Figure B.12.3.5 show the TEKPLOT session. The user types the executable filename, **tekplot**, and is asked for values for NUM, ITRI, NPHI, INAP, and NSWYY. The reply "**0 1 s**" sets NUM=0 and ITRI=1. The following three parameters are left by default with values of 0. The NUM=0 and ITRI=1 tells TEKPLOT to use dump 0 and to display the triangles in the mesh. The user is then asked for the minimum and maximum limits of the plotting region. The reply "**s**" tells the code to use the internal values for these variables. After being told to "**go**", the code produces Fig. B.12.3.6.

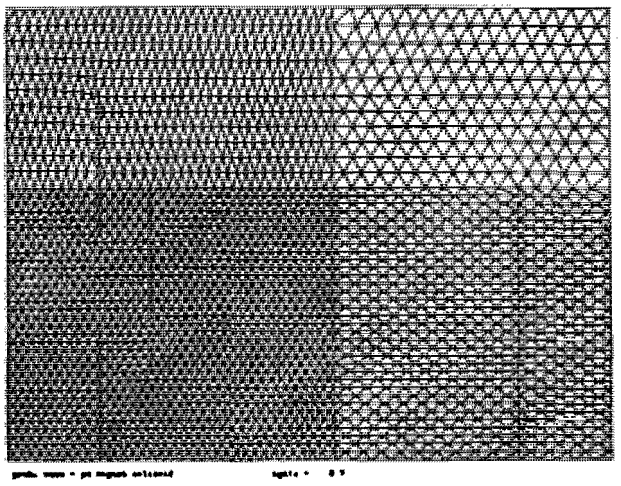


Figure B.12.3.6: TEKPLOT output for solenoid problem showing mesh and regional boundaries.

```
pandira
 ?type ''tty'' or input file name
?tty
 ?type input value for num
?0
 beginning of pandira execution from dump number 0

 prob.  name = pm magnet solenoid
 type input values for con(?)
?*30 30 *18 2 *43 1 1 46 s

 number of permeability tables to be read in = 2
 ?type input for table - mater, stack, mtype
?6 1. -1
 ?type input for table - aniso, gamper, x0, y0, phi
?270. 1. s
 ?type input for table - hcept, bcept
?-8500. 8500.
 ?type input for table - mater, stack, mtype
?7 1.  -1
 type input for table - aniso, gamper, x0, y0, phi
?180. 1. s
 ?type input for table - hcept, bcept
?-8500.  8500.
elapsed time=   2.4 sec.
cycle    amin        amax
                     bmax    residual-fe   eta-fe
   0  0.0000e+00 0.0000e+00
                 0.0000e+00   1.000e+00  1.0000
solution time=   1.2 sec.
   1  0.0000e+00 3.0600e+04
                 1.4561e+04   9.813e-03  1.0000
solution time=   2.0 sec.
   2 -9.7940e+03 2.3201e+04
                 2.7290e+04   4.219e-02  3.6058
solution time=   1.8 sec.
   3  -9.9147e+03   2.3157e+04
                    2.2828e+04  2.634e-02  0.6221
 solution time=     0.9 sec.
   4  -9.9336e+03   2.3150e+04
                    2.1476e+04  1.528e-02  0.5804
 solution time=     0.8 sec.
   5  -9.9473e+03   2.3145e+04
                    2.1474e+04  7.177e-03  0.4697
 solution time=     0.9 sec.
   6  -9.9516e+03   2.3143e+04
                    2.1542e+04  1.676e-03  0.2336
 solution time=     0.9 sec.
   7  -9.9520e+03   2.3143e+04
                    2.1561e+04  2.581e+04  0.1540
 solution time=     0.8 sec.
   8  -9.9520e+03   2.3143e+04
                    2.1563e+04  3.313e+04  0.0128
 solution time=     0.8 sec.
   9  -9.9520e+03   2.3143e+04
                    2.1563e+04  6.087e-10  0.0002

solution converged in 9 interations
elapsed time=   14.5 sec.
dump number 1 has written on tape35.
?type input value for num
```

Figure B.12.3.7: Interaction of PANDIRA on CRAY for solenoid magnet problem.

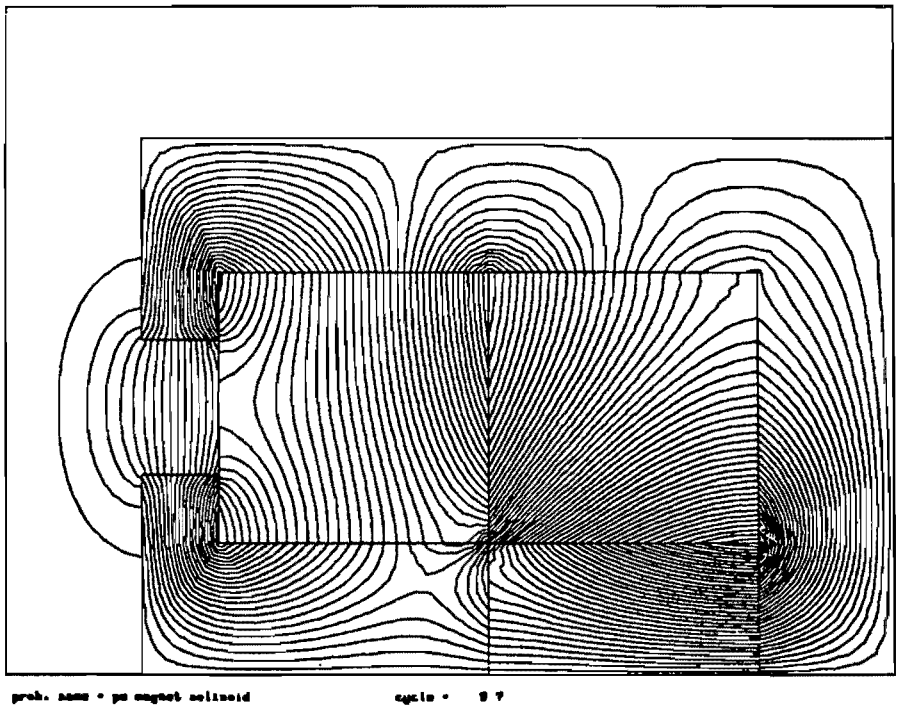


Figure B.12.3.8: TEKPLOT output showing field in solenoid magnet.

The next step is to run PANDIRA. The user types the executable file name, **pandira** (Fig. B.12.3.7). The program asks for the name of the source of input and is told TTY. It then asks for a dump number and is told 0. The next request is for CON changes. The reply is *30 30 *18 2 *43 1 1 46 S. The entry *30 30 changes CON(30)=MAXCY to 30. This is the maximum number of cycles allowed. The entry *18 2 sets CON(18)=NPERM=2; there are two permeability tables to be entered. The entry *43 1 1 46 gives values for CON(43), CON(44), and CON(45). These values set the limits in terms of (K, L) coordinates to the region in which fields and gradients are to be calculated. The two permeability tables are constructed from three input lines each. See Sec. B.5.4 for an explanation of the variables. After reading the final "s", PANDIRA executes and converges in nine cycles. The user can look at the field pattern using TEKPLOT. This time, he sets NUM=1, ITRI=0, and NPHI=50. On being told to "GO", TEKPLOT produces Fig. B.12.3.8.

The results of the calculation are in the file OUTPAN. This file contains a lists of the CON values in the solutions, a table giving the magnetic properties of material 2 (the internal iron table), tables giving the properties of materials 6 and 7, the history of the iteration, and an edit of the solution in the region defined bt CON(43) through CON(45). A portion of this edit is shown in Fig. B.12.3.9. The (K,L) and (R,Z) columns give the logical and physical coordinates of a point. The *ra(vector)* column gives the values of rA_θ found by the program at that point.

The br, bz, and bt columns are the radial, z, and total magnetic field at the point. The field index is given in the n-column. The final column, $rafit$, is the difference between the rA_θ value at the point found during the solution and the value found using the least squares fitting. The codes does a least squares fit with a polynomial and then gets the necessary derivatives of A_θ for the field from the polynomial.

Chapter B.13

Appendices

B.13.1 Theory of Electrostatics and Magnetostatics.

Let us begin with Maxwell's equations:[4]

$$\nabla \times \mathbf{E} + \partial \mathbf{B}/\partial t = 0 \tag{B.13.1.1}$$

$$\nabla \times \mathbf{H} - \partial \mathbf{D}/\partial t = \mathbf{J} \tag{B.13.1.2}$$

$$\nabla \cdot \mathbf{B} = 0 \tag{B.13.1.3}$$

$$\nabla \cdot \mathbf{D} = \rho. \tag{B.13.1.4}$$

Equations (B.13.1.2) and (B.13.1.4) imply the equation of continuity

$$\nabla \cdot \mathbf{J} + \partial \rho/\partial t = 0. \tag{B.13.1.5}$$

Maxwell's equations cannot be solved without assuming some relation between the vectors **E**, **B**, **D**, **H** and **J**. In vacuum the relation are

$$\mathbf{D} = \epsilon_0 \mathbf{E} \tag{B.13.1.6}$$

$$\mathbf{H} = \mathbf{B}/\mu_0 \tag{B.13.1.7}$$

$$\mathbf{J} = \sum_{i=1}^{N} \mathbf{J}_i \tag{B.13.1.8}$$

$$\rho = \sum_{i=1}^{N} \rho_i, \tag{B.13.1.9}$$

where

$$d\mathbf{J}_i/dt = \sum_{j \neq i} (e/m)_j [\rho_j \mathbf{E} + \mathbf{J}_j \times \mathbf{B}], \quad i = 1, \ldots, N, \tag{B.13.1.10}$$

and where the summation is over all ionic species present. Equation (B.13.1.10) is the Lorentz force equation under the assumption that charge and mass are quantized and that relativistic and radiation-damping effects can be ignored. This relation for $\mathbf{J}$ is just illustrative of the type of equation needed to close the system but will not be used here.

In isotropic solids, which are the only materials POISSON can handle, one can write with adequate generality

$$\mathbf{D} = \epsilon(\mathbf{x}, t, |\mathbf{E}|)|\mathbf{E}| = \kappa_e(\mathbf{x}, t, |\mathbf{E}|)\epsilon_0 \mathbf{E} \tag{B.13.1.11}$$

$$\mathbf{H} = \gamma(\mathbf{x}, t, |\mathbf{B}|)\mathbf{B}/\mu_0 \tag{B.13.1.12}$$

$$\mathbf{J} = \sigma(\mathbf{x}, t, |\mathbf{E}|)\mathbf{E}, \tag{B.13.1.13}$$

where the dielectric constant κ_e, the reluctivity γ and the conductivity σ are usually piecewise constant functions of $\mathbf{x}$. Note that the reluctivity γ is the reciprocal of the relative peameability κ_m. For fields slowly varying in time, the time-dependence of ϵ and γ can be ignored. The case of rapidly oscillating fields will be discussed later when examining the capabilities of SUPERFISH. Let us look at static problems first.

POISSON and PANDIRA solve a generalized intergal form of Poisson's equation in two-dimensional, cartesian coordinates or in cylindrically symmetric three-dimensional coordinates. This integral form of Poisson's equation works for both magnetostatics and electrostatics. It also handles both isotropic and anisotropic materials. In the following subsections we will start with the integral equation and show how it is related to Maxwell's and Poisson's equations.

B.13.1.1 Isotropic Magnetostatics in Cartesian Coordinates.

The integral form of Eq.(B.13.1.2) is

$$\oint_C \gamma(\mathbf{x}, |\mathbf{B}|)\nabla \times \mathbf{A} \cdot d\mathbf{l} = \mu_0 \int_A \mathbf{J} \cdot d\mathbf{a}, \tag{B.13.1.14}$$

where we have assumed that the displacement field $\mathbf{D}$ has no time dependence. The contour C encloses the area A. See Fig. B.13.1.1. The magnetic induction $\mathbf{B}$

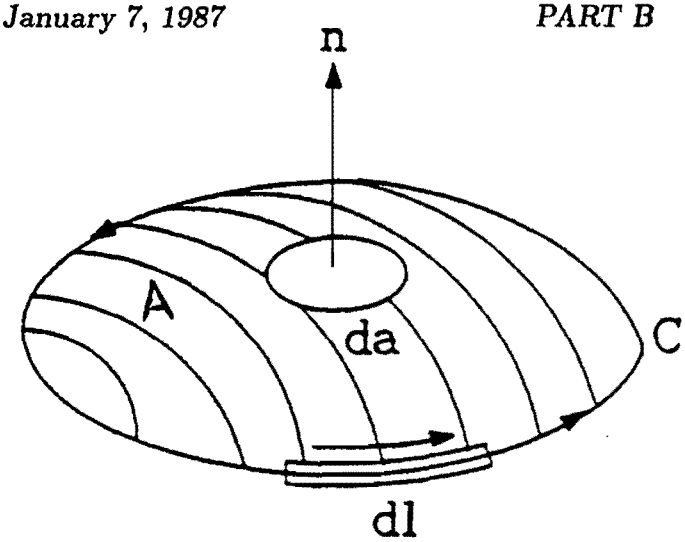


Fig.B.13.1.1 The geometry of the line and area integrals.

is related to the vector potential **A** by the equation

$$\mathbf{B} = \nabla \times \mathbf{A}. \tag{B.13.1.15}$$

One obtains the generalized Poisson equation by the following two steps:

$$\oint_C \gamma \nabla \times \mathbf{A} \cdot \mathbf{dl} = \int_A \nabla \times (\gamma \nabla \times \mathbf{A}) \cdot d\mathbf{a} = \int_A \mu_0 \mathbf{J} \cdot d\mathbf{a} \tag{B.13.1.16}$$

$$\nabla \times (\gamma \nabla \times \mathbf{A}) = \mu_0 \mathbf{J}. \tag{B.13.1.17}$$

Equation (B.13.1.17) does not look like Poisson's equation, but in the case of two-dimensional cartesian coordinates, it reduces to Poisson's equation in the following way. Let us assume that **A** is only a function of two coordinates (x,y) and that **B** has only two non-zero components, B_x and B_y. Equation (B.13.1.15) requires that

$$B_x = \frac{\partial A_z}{\partial y} - \frac{\partial A_y}{\partial z} = \frac{\partial A_z}{\partial y} \tag{B.13.1.18}$$

$$B_y = \frac{\partial A_x}{\partial z} - \frac{\partial A_z}{\partial x} = -\frac{\partial A_z}{\partial x} \tag{B.13.1.19}$$

$$B_z = \frac{\partial A_y}{\partial x} - \frac{\partial A_x}{\partial y} = 0. \tag{B.13.1.20}$$

Equation (B.13.1.20) is satisfied if A_x and A_y are required to be identically zero. This requirement also satisfies the guage relation

$$\nabla \cdot \mathbf{A} = \frac{\partial A_x}{\partial x} + \frac{\partial A_y}{\partial y} + \frac{\partial A_z}{\partial z} = 0. \tag{B.13.1.21}$$

It is easily seen that Eq.(B.13.1.17) becomes

$$\frac{\partial}{\partial x}\left(\gamma\frac{\partial A_z}{\partial x}\right)+\frac{\partial}{\partial y}\left(\gamma\frac{\partial A_z}{\partial y}\right)=-\mu_0 J_z \tag{B.13.1.22}$$

$$J_x = J_y = 0. \tag{B.13.1.23}$$

The latter equation forces the current to be parallel to **A** and Eq.(B.13.1.22) is the generalized form of Poisson's equation. Equation (B.13.1.14) in two dimensions reduces to

$$\oint_C \gamma(x,y,|\mathbf{B}|)\left[\frac{\partial A_z}{\partial y}dx-\frac{\partial A_z}{\partial x}dy\right]=\mu_0\int_A J_z dxdy. \tag{B.13.1.24}$$

For a small enough area A, we can assume thatγ and J_z are nearly constant and A_z is a linear function of x and y, which we can write as

$$A_z \cong -b_y x + b_x y + a_0. \tag{B.13.1.25}$$

The integrals become

$$\left[\gamma(b_x,b_y)\oint_C dx\right]b_x+\left[\gamma(b_x,b_y)\oint_C dy\right]b_y=\mu_0 J_z A. \tag{B.13.1.26}$$

This is a non-linear equation with two unknowns, b_x and b_y. It is the sort of equation that is created at each mesh point. Section B.13.6 explains how the set of mesh-point equations are solved to obtain numerically the vector potential A_z.

B.13.1.2 Isotropic Electrostatics in Cartesian Coordinates.

The integral form of Eq.(B.13.1.4) is

$$\oint_S \kappa_e(\mathbf{x},|\mathbf{E}|)\nabla V(\mathbf{x})\cdot d\mathbf{s}=-\frac{1}{\epsilon_0}\int_V \rho(\mathbf{x})dv, \tag{B.13.1.27}$$

where $V(\mathbf{x})$ is the scaler potential and the integral on the right is over a volume V whose surface is denoted by S. The electric field **E** is given by

$$\mathbf{E}=-\nabla V. \tag{B.13.1.28}$$

It is well known that Eq.(B.13.1.27) is equivalent to the differential equation

$$\nabla\cdot(\kappa_e\nabla V)=-\rho/\epsilon_0. \tag{B.13.1.29}$$

In two-dimensional cartesian coordinates this becomes

$$\frac{\partial}{\partial x}(\kappa_e V)+\frac{\partial}{\partial y}(\kappa_e V)=-\rho/\epsilon_0. \tag{B.13.1.30}$$

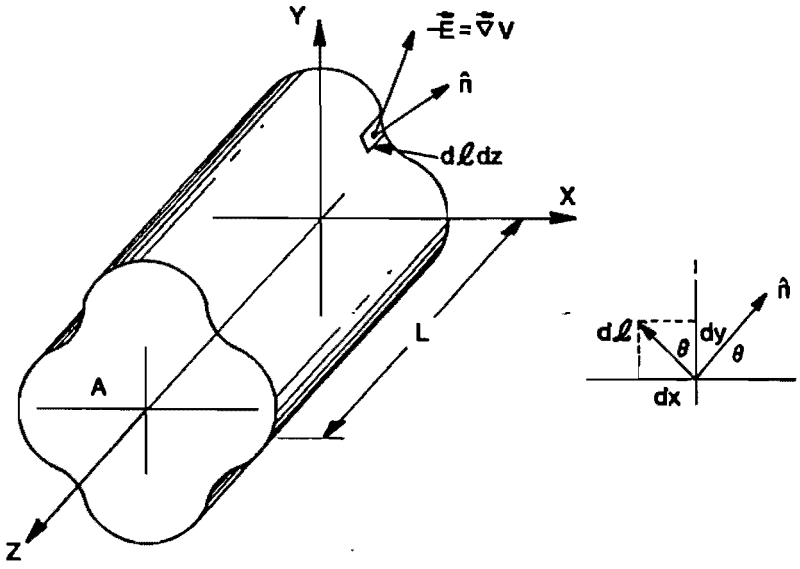


Fig.B.13.1.2: Cylindrical volume V of uniform cross section A and length L.

The similarity to Eq.(B.13.1.22) is obvious. The integral form of the equation can also be made to resemble Eq.(B.13.1.24). This is done as follows. Suppose that κ_e, $V(\mathbf{x})$ and $\rho(\mathbf{x})$ are functions of only (x, y). Take the volume V to be a generalized cylinder of cross section A and length L as shown in Fig.B.13.1.2. Since $V(\mathbf{x})$ is only a function of (x, y), ∇V has only x- and y-components. This immediately says that the surface integral over the flat ends of the volume cannot contribute because the direction of these areas is perpendicular to ∇V. The integral over the area of the cylindrical surface can be written as

$$\oint_S \kappa_e \nabla V \cdot d\mathbf{s} = \oint_C \int_o^L \kappa_e \nabla V \cdot \hat{\mathbf{n}} dz dl, \tag{B.13.1.31}$$

where $\hat{\mathbf{n}}$ is the outward normal to the surface and dl is an element of length on the contour C. The integral over z is just L and the quantity $\hat{\mathbf{n}}dl$ can be written in cartesian components as

$$\hat{\mathbf{n}}dl = dl\cos\theta\hat{\mathbf{e}}_\mathbf{x} + dl\sin\theta\hat{\mathbf{e}}_\mathbf{y}. \tag{B.13.1.32}$$

From Fig.B.13.1.2 we see that

$$dx = -dl\sin\theta, \tag{B.13.1.33}$$

and

$$dy = dl\cos\theta, \tag{B.13.1.34}$$

and hence

$$\hat{\mathbf{n}}dl = dy\hat{\mathbf{e}}_\mathbf{x} - dx\hat{\mathbf{e}}_\mathbf{y}. \tag{B.13.1.35}$$

When this relation is put into Eq.(B.13.1.31) the result is

$$\oint_S \kappa_e \nabla V \cdot d\mathbf{s} = -L \oint_C \kappa_e \left[\frac{\partial V}{\partial y} dx - \frac{\partial V}{\partial x} dy \right]. \tag{B.13.1.36}$$

The integral over the volume V can be written as

$$\frac{1}{\epsilon_0} \int_V \rho dV = \frac{L}{\epsilon_0} \int_A \rho dx dy. \tag{B.13.1.37}$$

This means that Eq.(B.13.1.27) can be written as

$$\oint_C \kappa_e(x, y, |\mathbf{E}|) \left[\frac{\partial V}{\partial y} dx - \frac{\partial V}{\partial x} dy \right] = \frac{1}{\epsilon_0} \int_A \rho dx dy. \tag{B.13.1.38}$$

Comparison with Eq.(B.13.1.24) makes it clear that there is a correspondance between the quantities

$$V \longrightarrow A_z \tag{B.13.1.39}$$

$$\kappa_e \longrightarrow \gamma \tag{B.13.1.40}$$

$$\frac{1}{\epsilon_0} \longrightarrow \mu_0 \tag{B.13.1.41}$$

$$\rho \longrightarrow J_z. \tag{B.13.1.42}$$

The same correspondances come out of a comparison between Eqs.(B.13.1.22) and (B.13.1.30).

B.13.1.3 Isotropic Magnetostatics in Cylindrical Coordinates.

We assume that **A** is only a function of (r, z) and not the cylindrical angle θ. The expression for **B** is

$$B_r = (\nabla \times \mathbf{A})_r = \frac{1}{r} \frac{\partial A_z}{\partial \theta} - \frac{\partial A_\theta}{\partial z} = -\frac{\partial A_\theta}{\partial z} \tag{B.13.1.43}$$

$$B_z = \frac{1}{r} \frac{\partial}{\partial r}(r A_\theta) - \frac{1}{r} \frac{\partial A_r}{\partial \theta} = \frac{1}{r} \frac{\partial}{\partial r}(r A_\theta) \tag{B.13.1.44}$$

$$B_\theta = \frac{\partial A_r}{\partial z} - \frac{\partial A_z}{\partial r} = 0. \tag{B.13.1.45}$$

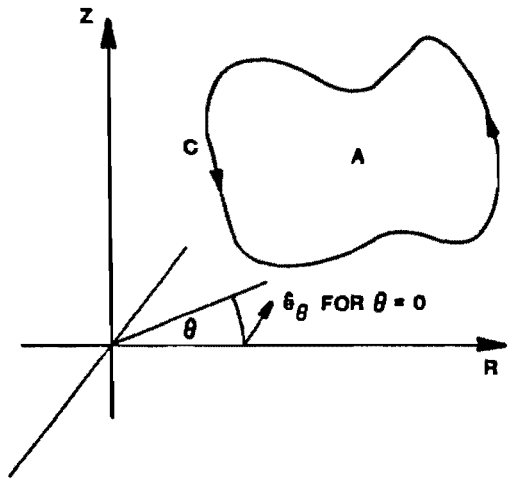


Fig.B.13.1.3: The geometry of the area and contour integrals in cylindrical coordinates. If the contour is taken in the counter-clockwise direction, then the area has a normal in the $-\hat{\mathbf{e}}_\theta$ direction. This is important in deriving Eq.(B.13.1.47).

Here we have assumed that **B** lies in the rz-plane. We choose the condition

$$A_r = A_z = 0, \tag{B.13.1.46}$$

which automatically satisfies the guage condition $\nabla \cdot A = 0$. Equation (B.13.1.14) written in cylindrical coordinates is

$$\oint_C \frac{\gamma}{r}\left[\frac{\partial}{\partial z}(rA_\theta)dr - \frac{\partial}{\partial r}(rA_\theta)dz\right] = \mu_0 \int_A J_\theta dr dz, \tag{B.13.1.47}$$

where we have artificially introduced a factor of r into the derivative with respect to z for symmetry. The geometry is shown in Fig.B.13.1.3.

There is a clear correspondance between quantities in cylindrical and cartesian coordinates given by the relations

$$r \longrightarrow x \tag{B.13.1.48}$$

$$z \longrightarrow y \tag{B.13.1.49}$$

$$rA_\theta \longrightarrow -A_z \tag{B.13.1.50}$$

$$J_\theta \longrightarrow -J_z \tag{B.13.1.51}$$

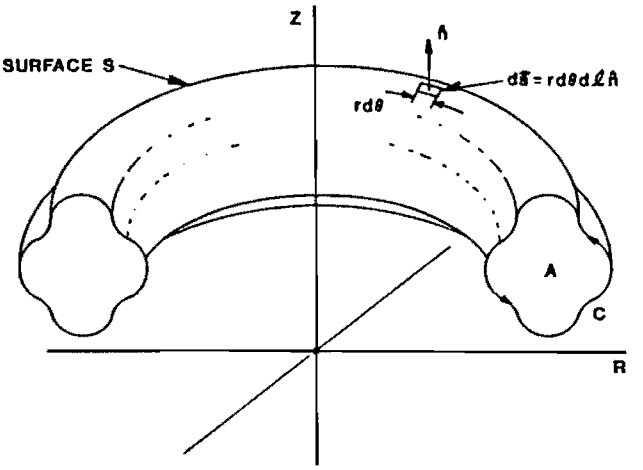


Fig.B.13.1.4: The geometry of the volume V, surface S, cross sectional area A, and contour C. Note that the normal to the area A is in the $-\hat{\mathbf{e}}_\theta$ direction.

$$\gamma/r \longrightarrow \gamma. \tag{B.13.1.52}$$

These relations are used in the codes.

B.13.1.4 Isotropic Electrostatics in Cylindrical Coordinates.

We assume that the scaler potential is only a function of (r, z). The electric field has only r- and z-components, which are given by

$$E_r = -\frac{\partial V}{\partial r} \tag{B.13.1.53}$$

$$E_z = -\frac{\partial V}{\partial z}. \tag{B.13.1.54}$$

The volume of integration in Eq.(B.13.1.27) is a torus of cross section A as illustrated in Fig.B.13.1.4. Integration over the angle θ can be done immediately and Eq.(B.13.1.27) can be expressed as

$$2\pi \oint_C \kappa_e \nabla V \cdot \hat{\mathbf{n}} r dl = -\frac{2\pi}{\epsilon_0} \int_A \rho r dr dz. \tag{B.13.1.55}$$

In analogy to Eq.(B.13.1.35), it can be shown that

$$\hat{\mathbf{n}} dl = dz \hat{\mathbf{e}}_\mathbf{r} - dr \hat{\mathbf{e}}_\mathbf{z}, \tag{B.13.1.56}$$

and hence Eq.(B.13.1.55) becomes

$$\oint_C \kappa_e r \left[\frac{\partial V}{\partial z} dr - \frac{\partial V}{\partial r} dz \right] = \frac{1}{\epsilon_0} \int_A \rho r dr dz. \tag{B.13.1.57}$$

Comparison with the magnetostatic equation in Cartesian coordinates, Eq.(B.13.1.24) give the correspondances

$$r \longrightarrow x \tag{B.13.1.58}$$

$$z \longrightarrow y \tag{B.13.1.59}$$

$$V \longrightarrow A_z \tag{B.13.1.60}$$

$$\kappa_e r \longrightarrow \gamma \tag{B.13.1.61}$$

$$\frac{1}{\epsilon_0} \longrightarrow \mu_0 \tag{B.13.1.62}$$

$$r\rho \longrightarrow J_z. \tag{B.13.1.63}$$

These correspondances are used in the codes. Comparison with Eqs.(B.13.1.48) through (B.13.1.52) shows that $A_\theta(r,z)$ is not analogous to $V(r,z)$ in cylindrical coordinates.

B.13.1.5 Anisotropic Magnetostatics.

Historically, there was an attempt to modify POISSON to handle two-dimensional anisotropic materials, but the convergence was extremely poor. It was decided to write a new program called PANDIRA, which uses the so-called direct method rather than the successive over-relaxation method for the numerical solution of Maxwell's equations. Maxwell's equations become

$$\nabla \times \mathbf{H} = \mathbf{J} \tag{B.13.1.64}$$

$$\nabla \cdot \mathbf{B} = 0 \tag{B.13.1.65}$$

$$\mathbf{B} = \nabla \times \mathbf{A} \tag{B.13.1.66}$$

$$\mathbf{H} = \overleftrightarrow{\gamma} \cdot \mathbf{B}/\mu_0 + \mathbf{H_c}. \tag{B.13.1.67}$$

where $\overleftrightarrow{\gamma}$ is the reluctivity tensor and $\mathbf{H_c}$ is the field when $\mathbf{B} = 0$, that is, the permanent magnetic field. PANDIRA handles anisotropic materials having an "easy-axis"

and a "hard-axis" perpendicular to the easy axis,

$$\mathbf{H} = \mathbf{H}_{\|} + \mathbf{H}_{\perp} \tag{B.13.1.68}$$

$$\mathbf{H}_{\|} = \gamma_{\|}\mathbf{B}_{\|}/\mu_0 + \mathbf{H}_c \tag{B.13.1.69}$$

$$\mathbf{H}_{\perp} = \gamma_{\perp}\mathbf{B}_{\perp}/\mu_0, \tag{B.13.1.70}$$

or one can write the inverse relations

$$\mathbf{B}_{\|} = \kappa_{m\|}\mu_0\mathbf{H}_{\|} + \mathbf{B}_r \tag{B.13.1.71}$$

$$\mathbf{B}_{\perp} = \kappa_{m\perp}\mu_0\mathbf{H}_{\perp}, \tag{B.13.1.72}$$

where

$$\gamma_{\|} = 1/\kappa_{m\|} \tag{B.13.1.73}$$

$$\gamma_{\perp} = 1/\kappa_{m\perp} \tag{B.13.1.74}$$

$$\mathbf{B}_r = -\kappa_{m\|}\mu_0\mathbf{H}_c, \tag{B.13.1.75}$$

and where $\mathbf{B}_{\|}$ is parallel to the easy axis and $\mathbf{B}_{\perp}$ is along the hard axis. The permeability relations are different in the two special directions. Figure B.13.1.5 shows typical relations. The easy axis is characterized by the coercive force H_c and a remanent field B_r.

PANDIRA allows two geometries for the easy axis. In the first geometry the easy axis is independent of position in the material. In the second geometry, the direction of the easy axis changes along the circumference of a circle that is not concentric with the origin of coordinates. These two geometries will be more fully explained below.

Easy axis in a fixed direction. Figure B.13.1.6 shows the directions of the easy and hard axes relative to the axes of the region of interest. The field $\mathbf{B}$ can be expressed in terms of the unit vectors for the two coordinate systems as

$$\mathbf{B} = B_{\mathbf{X}}\hat{\mathbf{e}}_{\mathbf{X}} + B_{\mathbf{Y}}\hat{\mathbf{e}}_{\mathbf{Y}} = B_{\|}\hat{\mathbf{e}}_{\|} + B_{\perp}\hat{\mathbf{e}}_{\perp}. \tag{B.13.1.76}$$

The unit vectors are related by the matrix transformation

$$\begin{pmatrix} \hat{\mathbf{e}}_{\|} \\ \hat{\mathbf{e}}_{\perp} \end{pmatrix} = \begin{pmatrix} \cos\phi_E & \sin\phi_E \\ -\sin\phi_E & \cos\phi_E \end{pmatrix} \begin{pmatrix} \hat{\mathbf{e}}_{\mathbf{X}} \\ \hat{\mathbf{e}}_{\mathbf{Y}} \end{pmatrix}. \tag{B.13.1.77}$$

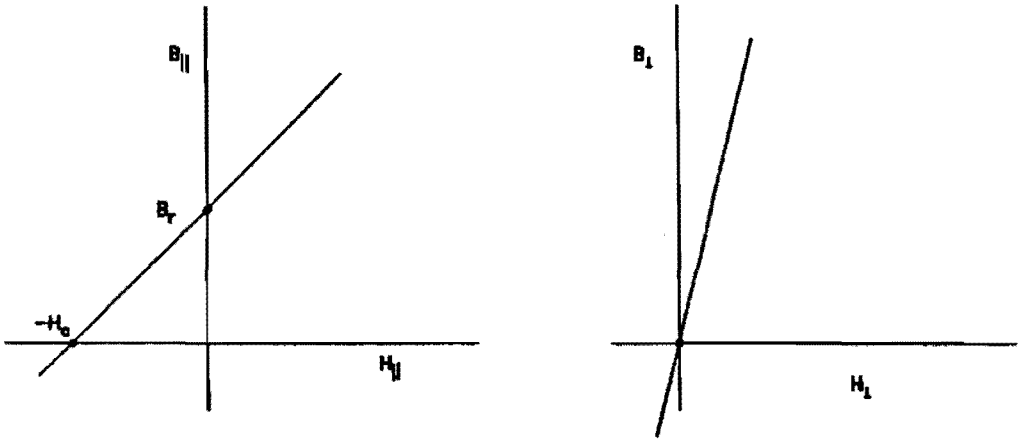


Fig.B.13.1.5: A plot of typical anisotropic B-H relations.

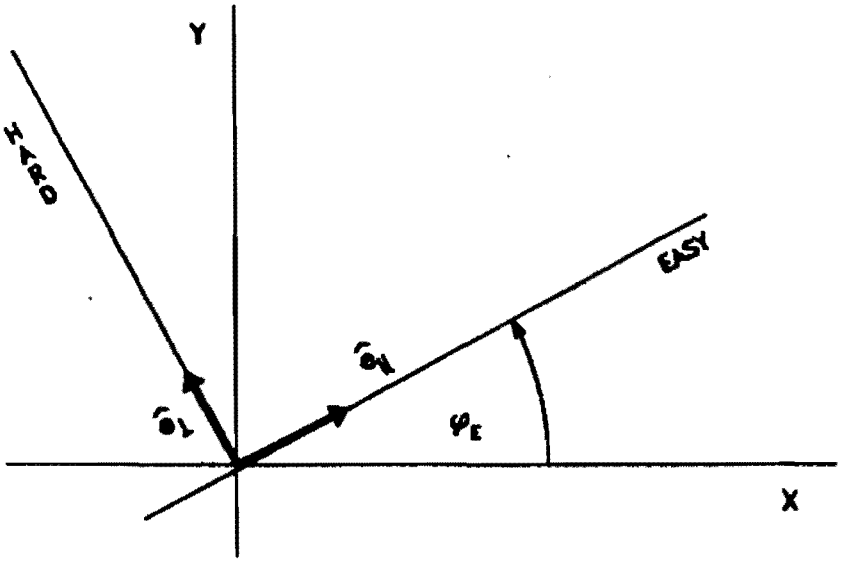


Fig.B.13.1.6: Definition of ϕ_E and unit vectors $\hat{\mathbf{e}}_{\|}$ and $\hat{\mathbf{e}}_{\perp}$.

The field **H** takes the form

$$\mathbf{H} = (\gamma_{\|} B_{\|}/\mu_0 + H_c)\hat{\mathbf{e}}_{\|} + (\gamma_{\perp} B_{\perp}/\mu_0)\hat{\mathbf{e}}_{\perp} = H_{\|}\hat{\mathbf{e}}_{\|} + H_{\perp}\hat{\mathbf{e}}_{\perp}, \tag{B.13.1.78}$$

from which one can deduce that

$$\mathbf{H} = (H_{\|}\cos\phi_E + H_{\perp}\sin\phi_E)\hat{\mathbf{e}}_{\mathbf{X}} + (H_{\|}\sin\phi_E - H_{\perp}\cos\phi_E)\hat{\mathbf{e}}_{\mathbf{Y}}. \tag{B.13.1.79}$$

By using the inverse transformation on the unit vectors, namely,

$$\begin{pmatrix} \hat{\mathbf{e}}_{\mathbf{X}} \\ \hat{\mathbf{e}}_{\mathbf{Y}} \end{pmatrix} = \begin{pmatrix} \cos\phi_E & -\sin\phi_E \\ \sin\phi_E & \cos\phi_E \end{pmatrix} \begin{pmatrix} \hat{\mathbf{e}}_{\|} \\ \hat{\mathbf{e}}_{\perp} \end{pmatrix}, \tag{B.13.1.80}$$

one can show that

$$\mathbf{B} = (B_{\mathbf{X}}\cos\phi_E + B_{\mathbf{Y}}\sin\phi_E)\hat{\mathbf{e}}_{\|} + (-B_X\sin\phi_E + B_Y\cos\phi_E)\hat{\mathbf{e}}_{\perp}. \tag{B.13.1.81}$$

This gives $B_{\|}$ and $B_{\perp}$ (and hence $H_{\|}$ and $H_{\perp}$ in terms of $B_{\mathbf{X}}$ and $B_{\mathbf{Y}}$.

After making the substitutions, one finds that

$$H_X = [(\gamma_{\|}\cos^2\phi_E + \gamma_{\perp}\sin^2\phi_E)B_{\mathbf{X}}/\mu_0 + \sin 2\phi_E(\gamma_{\|} - \gamma_{\perp})B_{\mathbf{Y}}/(2\mu_0) - H_c\cos\phi_E], \tag{B.13.1.82}$$

and

$$H_{\mathbf{Y}} = [\sin 2\phi_E(\gamma_{\|} - \gamma_{\perp})B_{\mathbf{X}}/(2\mu_0) + (\gamma_{\|}\sin^2\phi_E + \gamma_{\perp}\cos^2\phi_E)B_{\mathbf{Y}}/\mu_0 - H_c\sin\phi_E]. \tag{B.13.1.83}$$

This can be written as

$$\begin{pmatrix} H_{\mathbf{X}} \\ H_{\mathbf{Y}} \end{pmatrix} = \frac{1}{\mu_0}\begin{pmatrix} \gamma_{\mathbf{XX}} & \gamma_{\mathbf{XY}} \\ \gamma_{\mathbf{XY}} & \gamma_{\mathbf{YY}} \end{pmatrix}\begin{pmatrix} B_{\mathbf{X}} \\ B_{\mathbf{Y}} \end{pmatrix} - \begin{pmatrix} H_{c\mathbf{X}} \\ H_{c\mathbf{Y}} \end{pmatrix}, \tag{B.13.1.84}$$

where

$$\gamma_{XX} = \gamma_{\|}\cos^2\phi_E + \gamma_{\perp}\sin^2\phi_E \tag{B.13.1.85}$$

$$\gamma_{XY} = \sin 2\phi_E(\gamma_{\|} - \gamma_{\perp})/2 \tag{B.13.1.86}$$

$$\gamma_{YY} = \gamma_{\|}\sin^2\phi_E + \gamma_{\perp}\cos^2\phi_E \tag{B.13.1.87}$$

$$H_{c\mathbf{X}} = H_c\cos\phi_E \tag{B.13.1.88}$$

$$H_{c\mathbf{Y}} = H_c\sin\phi_E, \tag{B.13.1.89}$$

which defines the symmetric reluctivity tensor $\overset{\Rightarrow}{\gamma}$ and the coercive force $\mathbf{H}_c$.

Easy axis on an off-center circle. Although there is no natural material for which the direction of the easy axis is a function of position in the plane, one

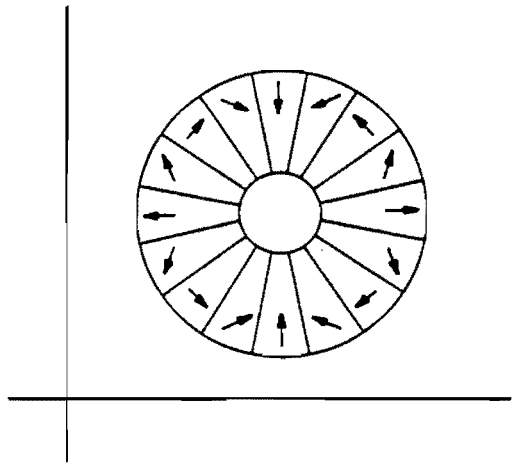


Fig.B.13.1.7: An example of a magnet constructed from wedge-shaped blocks of permanent magnet material with the easy axis in each block having a different orientation.

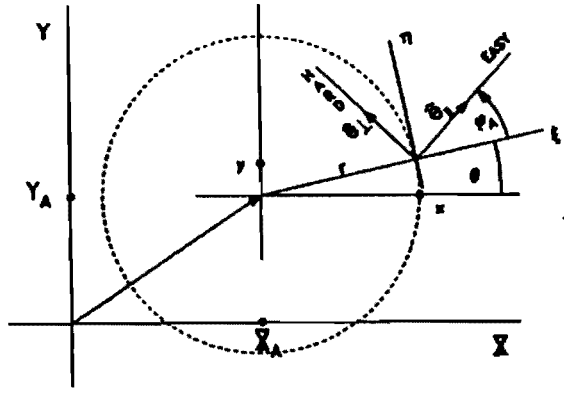


Fig. B.13.1.8: Definitions of parameters in off-center anisotropic materials.

can construct a magnet from blocks of permanent magnet material with each block having its own orientation. An example of this is shown in Fig. B.13.1.7. PANDIRA will handle the case when the direction is a function of angle around the center of a circle. The center of the circle need not coincide with the origin of coordinates. Figure B.13.1.8 shows the geometry of the easy axis in such a material. The orientation of the easy axis is determined by three parameters (X_A, Y_A, ϕ_A). Relating these parameters to the reluctivity tensor $\overset{\Rightarrow}{\gamma}$ and the coercive force vector $\mathbf{H_c}$ is a matter of coordinate transformations. There are three coordinate systems that enter into the formulation,

$$(\hat{\mathbf{e}}_{\|}, \hat{\mathbf{e}}_{\perp}) \rightarrow (\hat{\mathbf{e}}_{\xi}, \hat{\mathbf{e}}_{\eta}) \rightarrow (\hat{\mathbf{e}}_{\mathbf{X}}, \hat{\mathbf{e}}_{\mathbf{Y}}). \tag{B.13.1.90}$$

In the $(\hat{\mathbf{e}}_{\xi}, \hat{\mathbf{e}}_{\eta})$ system, the tensor $\overset{\Rightarrow}{\gamma}$ and the vector $\mathbf{H_c}$ are the same as described in Eqs.(B.13.1.85) through (B.13.1.89) above. The axes $(\hat{\mathbf{e}}_{\xi}, \hat{\mathbf{e}}_{\eta})$ change as a function of (x,y). The (x,y) coordinates are related to (X,Y) by a linear transformation. Let

$$\mathbf{X} = X\hat{\mathbf{e}}_{\mathbf{X}} + Y\hat{\mathbf{e}}_{\mathbf{Y}} = (X_A + x)\hat{\mathbf{e}}_{\mathbf{X}} + (Y_A + y)\hat{\mathbf{e}}_{\mathbf{Y}}, \tag{B.13.1.91}$$

and

$$\begin{pmatrix} \hat{\mathbf{e}}_{\xi} \\ \hat{\mathbf{e}}_{\eta} \end{pmatrix} = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} \hat{\mathbf{e}}_{\mathbf{X}} \\ \hat{\mathbf{e}}_{\mathbf{Y}} \end{pmatrix}, \tag{B.13.1.92}$$

where

$$\cos\theta = (X - X_A)/[(X - X_A)^2 + (Y - Y_A)^2]^{1/2}, \tag{B.13.1.93}$$

and

$$\sin\theta = (Y - Y_A)/[(X - X_A)^2 + (Y - Y_A)^2]^{1/2}. \tag{B.13.1.94}$$

From Fig. B.13.1.8 one sees that

$$\begin{pmatrix} \hat{\mathbf{e}}_{\|} \\ \hat{\mathbf{e}}_{\perp} \end{pmatrix} = \begin{pmatrix} \cos\phi_A & \sin\phi_A \\ -\sin\phi_A & \cos\phi_A \end{pmatrix} \begin{pmatrix} \hat{\mathbf{e}}_{\mathbf{X}} \\ \hat{\mathbf{e}}_{\mathbf{Y}} \end{pmatrix}. \tag{B.13.1.95}$$

Either by multiplication of the matrices or by looking at Fig.B.13.1.8 again, one can show that

$$\begin{pmatrix} \hat{\mathbf{e}}_{\|} \\ \hat{\mathbf{e}}_{\perp} \end{pmatrix} \begin{pmatrix} \cos(\phi_A + \theta) & \sin(\phi_A + \theta) \\ -\sin(\phi_A + \theta) & \cos(\phi_A + \theta) \end{pmatrix} \begin{pmatrix} \hat{\mathbf{e}}_{\mathbf{X}} \\ \hat{\mathbf{e}}_{\mathbf{Y}} \end{pmatrix}. \tag{B.13.1.96}$$

The derivation of the components of $\overset{\Rightarrow}{\gamma}$ can be completed as we did before for the fixed direction case. The only difference is that ϕ_E is replaced by $\phi_A + \theta(X, Y)$. The reluctivity tensor $\overset{\Rightarrow}{\gamma}$ and the vector $\mathbf{H_c}$ are now functions of position in the material.

In the input to PANDIRA, the parameters are defined by the equations

$$ANISO = \phi_E, GAMPER = \gamma_{\perp}, HCEPT = -H_c, \tag{B.13.1.97}$$

and $\gamma_{\|}$ is determined from B_r, that is, from Fig.B.13.1.5 we see that

$$\gamma_{\|} = B_r/(\mu_0 H_c). \tag{B.13.1.98}$$

With this as a background, we now write down the integral equation used in PANDIRA to find the vector potential, namely,

$$\oint_c [\vec{\vec{\gamma}} \cdot \nabla \times \mathbf{A} + \mu_0 \mathbf{H_c}] \cdot d\mathbf{l} = \mu_0 \int_A \mathbf{J} \cdot d\mathbf{a}. \tag{B.13.1.99}$$

In Cartesian coordinates we once more choose the gauge

$$\nabla \cdot \mathbf{A} = 0, \tag{B.13.1.100}$$

which requires that

$$\mathbf{A} = A_z \hat{\mathbf{e}}_z, \tag{B.13.1.101}$$

and

$$J_x = J_y = 0. \tag{B.13.1.102}$$

Equation (B.13.1.24) becomes

$$\begin{aligned} &\oint_e [\gamma_{xx} \frac{\partial A_z}{\partial y} - \gamma_{xy} \frac{\partial A_z}{\partial x} + \mu_0 H_{cx}] dx \\ &+ [\gamma_{xy} \frac{\partial A_z}{\partial y} - \gamma_{yy} \frac{\partial A_z}{\partial x} + \mu_0 H_{cy}] dy = \mu_0 \int_A J_z dx dy. \end{aligned} \tag{B.13.1.103}$$

This complication does not prevent one from applying the same numerical method discussed in Section B.13.6 below to find the solution. If A_z is a linear function of x and y as in Eq.(B.13.1.25), then Eq.(B.13.1.103) reduces to

$$\begin{aligned} &[\gamma_{xx} \oint_c dx + \gamma_{xy} \oint dy] b_x + [\gamma_{xy} \oint_c dx + \gamma_{yy} \oint_c dy] b_y = \\ &\mu_0 [J_z A - H_{cx} \oint_c dx - H_{cy} \oint dy]. \end{aligned} \tag{B.13.1.104}$$

This, like Eq.(B.13.1.26) is still a non-linear equation in b_x and b_y. Since the γ's are functions of b_x and b_y, the equation must be solved iteratively.

B.13.1.6 Anisotropic Electrostatics in Cartesian Coordinates.

The basic equations for ferroelectric materials are

$$\nabla \cdot \mathbf{D} = \rho \tag{B.13.1.105}$$

$$\nabla \times \mathbf{E} = 0 \tag{B.13.1.106}$$

$$\mathbf{E} = -\nabla V \tag{B.13.1.107}$$

$$\mathbf{D} = \epsilon_0 \vec{\kappa}_e \cdot \mathbf{E} + \mathbf{D_c} \tag{B.13.1.108}$$

As in the magnetostatic case, one can define an easy axis and a hard axis for the material. This results in relations analogous to Eqs.(B.13.1.68) through (B.13.1.89) with the correspondances

$$\mathbf{H} \longrightarrow \mathbf{D} \tag{B.13.1.109}$$

$$\mathbf{B} \longrightarrow \mathbf{E} \tag{B.13.1.110}$$

$$\gamma \longrightarrow \kappa_e \tag{B.13.1.111}$$

$$\mu_0 \longrightarrow 1/\epsilon_0. \tag{B.13.1.112}$$

The coordinate transformations (Eqs. (B.13.1.90) – (B.13.1.96)) are the same. The integral equation for the scalar potential is a straight forward generalization of Eq.(B.13.1.27), namely,

$$\oint_S (\kappa_e \cdot \nabla V + \mathbf{D_c}/\epsilon_0) \cdot d\mathbf{s} = \frac{1}{\epsilon_0} \int_V \rho dv. \tag{B.13.1.113}$$

The derivation leading to Eq.(B.13.1.36) is essentially unchanged. The final result is

$$\oint_S (\kappa_e \cdot \nabla V + \mathbf{D_c}/\epsilon_0) \cdot d\mathbf{s} = -L\left\{ \oint_c (\vec{\kappa}_c \cdot \nabla V + \mathbf{D_c}/\epsilon_0)_y dx - \oint_c (\kappa_e \cdot \nabla V + \mathbf{D_c}/\epsilon_0)_x dy \right\}, \tag{B.13.1.114}$$

and Eq. (B.13.1.38) now takes the form

$$\begin{aligned} &\oint_c [\kappa_{eyx} \frac{\partial V}{\partial x} + \kappa_{eyy} \frac{\partial V}{\partial y} + D_{cy}/\epsilon_0] dx \\ &- \oint_c [\kappa_{exx} \frac{\partial V}{\partial x} + \kappa_{exy} \frac{\partial V}{\partial y} + D_{cx}/\epsilon_0] dy = \frac{1}{\epsilon_0} \int_A \rho dx dy. \end{aligned} \tag{B.13.1.115}$$

A comparison with the magnetostatic result, Eq.(B.13.1.103) reveals distinct differences in interpretation. The two equations take the same form if we make the substitutions

$$V \longrightarrow A_z \tag{B.13.1.116}$$

$$\rho \longrightarrow J_z \tag{B.13.1.117}$$

$$\epsilon_0 \longrightarrow 1/\mu_0 \tag{B.13.1.118}$$

$$\kappa_{exx} \longrightarrow \gamma_{yy} \tag{B.13.1.119}$$

$$\kappa_{exy} = \kappa_{eyx} \longrightarrow -\gamma_{xy} \tag{B.13.1.120}$$

$$\kappa_{eyy} \longrightarrow \gamma_{xx} \tag{B.13.1.121}$$

$$D_{cx} \longrightarrow -H_{cx} \tag{B.13.1.122}$$

$$D_{cy} \longrightarrow H_{cy}. \tag{B.13.1.123}$$

The interchange of components and signs is equivalent to changing ϕ_E to $\phi_E + 90°$ when the easy axis has a fixed direction. This is not simply an interchange of hard and easy axes.

(Since no one has reported running an anisotropic electrostatic problem on PANDIRA, we are not sure if the above relations have been implemented in the code. Time constraints have not allowed us to check the coding itself.)

B.13.1.7 Anisotropic Magnetostatics in Cylindrical Coordinates.

To find the correspondances between the cylindrical and Cartesian cases, one must return to Eq. (B.13.1.99) and express $\nabla\times$, *etc.* in cylindrical coordinates. By following the steps taken in deriving Eq.(B.13.1.47) one finds that

$$\oint_C \left[\frac{\gamma_{rr}}{r}\frac{\partial}{\partial z}(-rA_\theta) - \frac{\gamma_{rz}}{r}\frac{\partial}{\partial r}(-rA_\theta) + \mu_0 H_{cr}\right] dr \tag{B.13.1.124}$$

$$+ \oint_C \left[\frac{\gamma_{rz}}{r}\frac{\partial}{\partial z}(-rA_\theta) - \frac{\gamma_{zz}}{r}\frac{\partial}{\partial r}(-rA_\theta) + \mu_0 H_{cz}\right] dz$$

$$= \mu_0 \int_A (-J_\theta) dr dz.$$

Comparison with Eq.(B.13.1.103) gives the correspondances

$$rA_\theta \longrightarrow -A_z \tag{B.13.1.125}$$

$$J_\theta \longrightarrow -J_z \tag{B.13.1.126}$$

$$\gamma_{rr}/r \longrightarrow \gamma_{xx} \tag{B.13.1.127}$$

$$\gamma_{rz}/r \longrightarrow \gamma_{xy} \tag{B.13.1.128}$$

$$\gamma_{zz}/r \longrightarrow \gamma_{yy} \tag{B.13.1.129}$$

$$H_{cr} \longrightarrow H_{cx} \tag{B.13.1.130}$$

$$H_{cz} \longrightarrow H_{cy}, \tag{B.13.1.131}$$

which are the analogue of Eqs. (B.13.1.50) – (B.13.1.52).

B.13.1.8 Anisotropic Electrostatics in Cylindrical Coordinates.

The starting point is Eq.(B.13.1.113), which must be expressed in cylindrical coordinates. The resulting equation is

$$\begin{aligned}\oint_C \left[(-r\kappa_{ezr})\frac{\partial V}{\partial r} + (-r\kappa_{ezz})\frac{\partial V}{\partial z} + \frac{1}{\epsilon_0}(-rD_{cz})\right]dr \\ + \oint_C \left[(r\kappa_{err})\frac{\partial V}{\partial r} + (r\kappa_{erz})\frac{\partial V}{\partial z} + \frac{1}{\epsilon_0}(rD_{cr})\right]dz \\ = \frac{1}{\epsilon_0}\int_A r\rho dr dz.\end{aligned} \tag{B.13.1.132}$$

Comparison with Eq.(B.13.1.103) leads to the correspondances

$$V \longrightarrow A_z \tag{B.13.1.133}$$

$$r\rho \longrightarrow J_z \tag{B.13.1.134}$$

$$\epsilon_0 \longrightarrow 1/\mu_0 \tag{B.13.1.135}$$

$$r\kappa_{err} \longrightarrow -\gamma_{yy} \tag{B.13.1.136}$$

$$r\kappa_{erz} = r\kappa_{ezr} \longrightarrow \gamma_{xy} \tag{B.13.1.137}$$

$$r\kappa_{ezz} \longrightarrow -\gamma_{xx} \tag{B.13.1.138}$$

$$rD_{cr} \longrightarrow H_{cy} \tag{B.13.1.139}$$

$$rD_{cz} \longrightarrow H_{cx}. \tag{B.13.1.140}$$

(Once again, we have not verified that this correspondence is actually in the code. Let the user beware!)

B.13.2 Auxiliary Properties of Static Magnetic and Electric Fields

The previous section showed how the static Maxwell's Equations give rise to the generalized Poisson equation for the static electric (scalar) and magnetic (vector) potentials. This section shows how the potentials are used to obtain the stored energy in the field, to obtain the fields and their derivatives, and to obtain the forces and torques on current-carrying coils and iron regions in magnetic fields. We will also obtain the forces and torques on charged plates and dielectric materials. These things will be done for both cartesian and cylindrical coordinates. In discussing the fields and their derivatives in cartesian coordinates, we will make use of complex variables and the theory of analytic functions. This will lay the basis for the discussion of harmonic analysis and the use of conformal transformations, in Sections B.13.3 and B.13.4.

B.13.2.1 Energy stored in the field.

The general expression for the energy in any volume V containing an electromagnetic field is

$$\mathcal{E} = \frac{1}{2}\int_V (\mathbf{E}\cdot\mathbf{D} + \mathbf{B}\cdot\mathbf{H})dv. \tag{B.13.2.1}$$

For electrostatic problems the $\mathbf{B}\cdot\mathbf{H}$−term is zero and for the magnetostatic problems the $\mathbf{E}\cdot\mathbf{D}$−term is zero. Our aim is to reduce this expression to area and contour integrals over potentials in two-dimensional cartesian coordinates and cylindrical coordinates. We begin by substituting

$$\mathbf{E} = -\nabla\phi_e \quad and \quad \mathbf{B} = \nabla\times\mathbf{A} \tag{B.13.2.2}$$

into Eq.(B.13.2.1) and using vector identities to get

$$\begin{aligned}\mathcal{E} &= \frac{1}{2}\int_V (-\nabla\phi_e\cdot\mathbf{D} + \nabla\times\mathbf{A}\cdot\mathbf{H})dv \qquad \text{(B.13.2.3)}\\ &= \frac{1}{2}\int_V \Big[-\nabla\cdot(\phi_e\mathbf{D}) + \phi_e\nabla\cdot\mathbf{D} + \nabla\cdot(\mathbf{A}\times\mathbf{H})\\ &\quad +\mathbf{A}\cdot\nabla\times\mathbf{H}\Big]dv.\end{aligned}$$

Green's theorem can be used to turn the integral over the volume of the divergence into an integral over the surface ∂V. The result is

$$\mathcal{E} = \frac{1}{2}\oint_{\partial V}(-\phi_e\mathbf{D} + \mathbf{A}\times\mathbf{H})\cdot d\mathbf{a} + \frac{1}{2}\int_V(\rho\phi_e + \mathbf{J}\cdot\mathbf{A})dv. \tag{B.13.2.4}$$

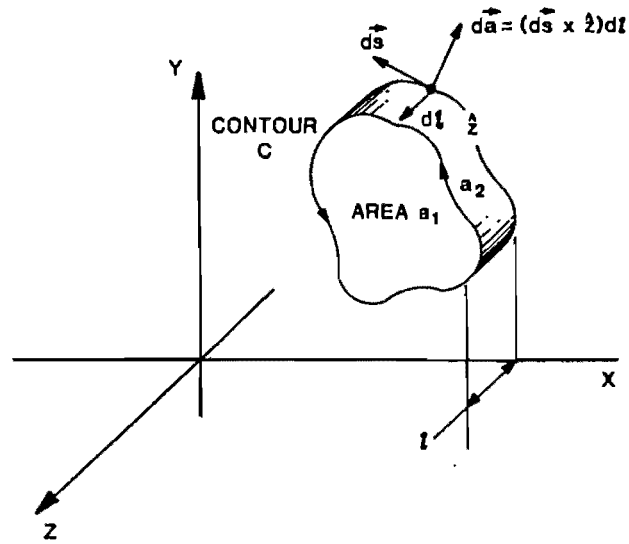


Figure B.13.2.1: The arbitrary volume with 2-D, cartesian symmetry. The front and back surfaces have area a_1. The edge has area a_2 and width l.

where we have used Maxwell's equations to express the volume integral in terms of the charge density ρ and the current density $\mathbf{J}$.

Let us consider cartesian coordinates first. We assume that all functions are independent of the z-coordinate. Consider an arbitrary volume as shown in Fig. B.13.2.1. The integration over z can be done immediately. The front and back planes have equal areas but opposite vector direction. This means that the surface integral over the front cancels the surface integral over the back. The result is

$$\mathcal{E} = \frac{1}{2}\int_{a_2}(-\phi_e\mathbf{D} + \mathbf{A}\times\mathbf{H})\cdot d\mathbf{a} + (l/2)\int_{a_1}(\rho\phi_e + \mathbf{J}\cdot\mathbf{A})da. \qquad \text{(B.13.2.5)}$$

The element of area on the ribbon edge can be written as

$$d\mathbf{a} = d\mathbf{s}\times dl\hat{\mathbf{z}}, \qquad \text{(B.13.2.6)}$$

where $\hat{\mathbf{z}}$ is the unit vector in the z-direction. The magnitude of the vector $d\mathbf{s}$ is the element of length along the counter-clockwise contour C enclosing the area a_1. See Fig. B.13.2.1. This means that we can turn the first integral into a contour integral around the area a_1. Furthermore, we note that both $\mathbf{J}$ and $\mathbf{A}$ must be in the z-direction. With theses simplifications we find that

$$\mathcal{E} = (l/2)\left[\oint_C(-\phi_e\mathbf{D} + \mathbf{A}\times\mathbf{H})\cdot d\mathbf{s}\times\hat{\mathbf{z}} + \int_{a_1}(\rho\phi_e + J_zA_z)da\right]. \qquad \text{(B.13.2.7)}$$

The first integral can be simplified further by using the vector idenities

$$\mathbf{D}\cdot d\mathbf{s}\times\hat{\mathbf{z}} = \mathbf{D}\times d\mathbf{s}\cdot\hat{\mathbf{z}} \equiv (\mathbf{D}\times d\mathbf{s})_z, \qquad \text{(B.13.2.8)}$$

and

$$\begin{aligned}\mathbf{A}\times\mathbf{H}\cdot d\mathbf{s}\times\hat{\mathbf{z}} &= (\mathbf{A}\times\mathbf{H})\times d\mathbf{s}\cdot\hat{\mathbf{z}} = (d\mathbf{s}\cdot\mathbf{A})(\mathbf{H}\cdot\hat{\mathbf{z}}) \\ &\quad -(\mathbf{H}\cdot d\mathbf{s})\mathbf{A}\cdot\hat{\mathbf{z}} = -A_z\mathbf{H}\cdot d\mathbf{s}. \qquad \text{(B.13.2.9)}\end{aligned}$$

The latter equality holds because $\mathbf{H}$ has no z-component.

For cartesian symmetry the code calculates the energy per unit length using the formula

$$\begin{aligned}\mathcal{E}/l &= \frac{1}{2}\Big\{\int_{a_1}(\rho\phi_e + J_zA_z)dxdy \qquad \text{(B.13.2.10)} \\ &\quad + \oint_C\Big[(\phi_eD_y - A_zH_x)dx - (\phi_eD_x + A_zH_y)dy\Big]\Big\}.\end{aligned}$$

In all cases handled by POISSON the contour integral vanishes on the boundary of the region because the boundary conditions are either pure Dirichlet ($A_z = 0$ or $\phi_e = 0$) or pure Neumann ($\mathbf{B}\cdot\hat{\mathbf{n}} = 0$ or $\mathbf{E}\cdot\hat{\mathbf{n}} = 0$). For example, consider the integral in Eq. (B.13.2.5),

$$\int_{a_2}\mathbf{A}\times\mathbf{H}\cdot d\mathbf{a} = \int_0^l dl\oint_C ds(\mathbf{A}\times H\cdot\hat{\mathbf{n}}), \qquad \text{(B.13.2.11)}$$

where $\hat{\mathbf{n}}$, being the normal to the ribbon area a_2, is also the normal to the contour C. The differential ds is still the element of distance along the contour. Figure B.13.2.2 illustrates the typical contour for one-quarter of an H-shaped magnet. As seen from the figure, the the contour integral vanishes. The energy per unit length reduces to

$$\mathcal{E}/l = (\frac{J}{2})\int_{coil}A_zda, \qquad \text{(B.13.2.12)}$$

which is $\frac{J}{2}$ times the integral of the magnetic potential over the area of the coil.

For permanent magnet (PANDIRA) problems, where there is no current density, the energy per unit length should be calculated from the contour integral where the contour goes around the permanent magnet material and not the boundary of the whole region. The code does not calculate the energy per unit length for the case of permanent magnets.

In the case of cylindrical symmetry, the volume of integration might be illustrated by Fig. B.13.2.3. The energy integral can be written as

$$\mathcal{E} = \frac{1}{2}\oint_{a_2}(-\phi_e\mathbf{D} + \mathbf{A}\times\mathbf{H})\cdot d\mathbf{a} + \pi\int_{a_1}(\rho\phi_e + \mathbf{J}\cdot\mathbf{A})rdrdz. \qquad \text{(B.13.2.13)}$$

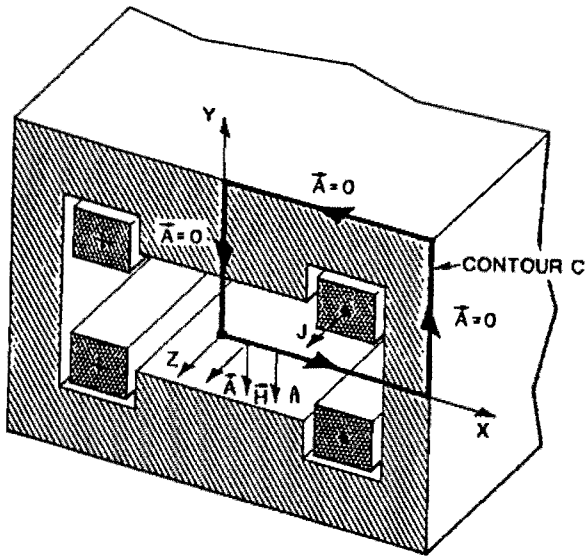


Figure B.13.2.2: The contour C for a typical H-shaped magnet. **A** vanishes on three sides because of Direchlet boundary conditions. $\mathbf{A} \times \mathbf{H} \cdot \hat{n}$ vanishes on the bottom because $\mathbf{A} \times \mathbf{H}$ is parallel to the contour.

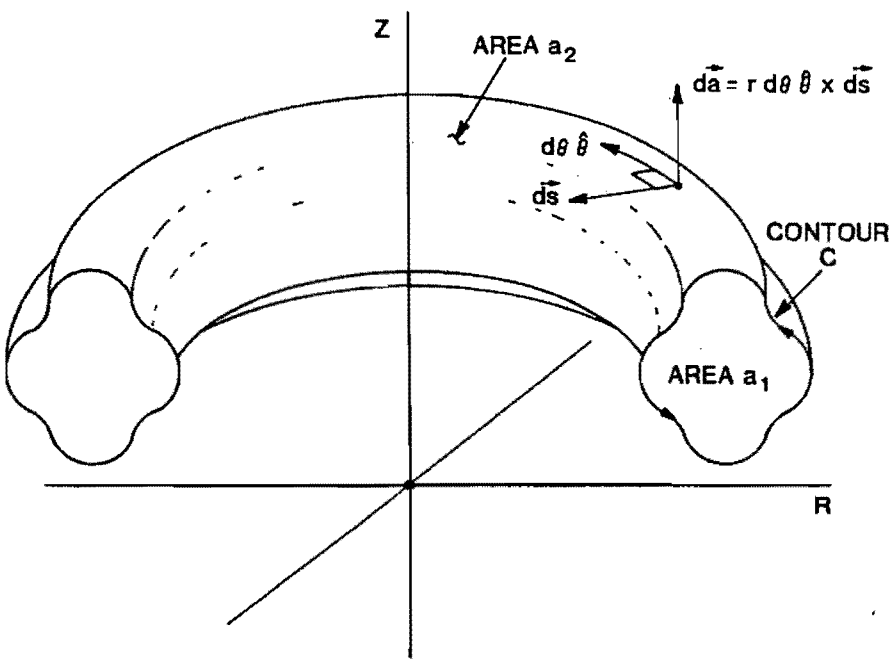


Figure B.13.2.3: Illustration of a general volune with cylindrical symmetry. The cross sectional area is a_1.

The integral over the volume of the torus has been simplified by an integration over the cylindrical angle θ. Once again the surface integral can be reduced to a contour integral by expressing the element of area as

$$d\mathbf{a} = rd\theta ds\hat{\mathbf{n}} = rd\theta\hat{\theta} \times d\mathbf{s}, \tag{B.13.2.14}$$

where $\hat{\mathbf{n}}$ is the unit normal and $\hat{\theta}$ is the unit vector in the θ direction. See Fig. B.13.2.3. After doing the integration on θ, noting that $\mathbf{A}$ and $\mathbf{J}$ only have θ-components, and $\mathbf{D}$ has no θ-component, we get

$$\begin{aligned} \mathcal{E} &= \pi\left\{\oint_C \left[(\phi_e D_z + A_\theta H_r)rdr + (-\phi_e D_r + A_\theta H_z)rdz\right]\right. \\ &\quad \left. + \int_{a_1} (\rho\phi_e + J_\theta A_\theta)rdrdz\right\}. \end{aligned} \tag{B.13.2.15}$$

The same arguments can be given to show that the contour integral vanishes on the boundary of the whole region, thus providing a simplification in POISSON calculations. Furthermore the authors of the code have chosen to ignore the integration over the angle θ. The printed results are in units of Joules/radian. Once again the energy is not calculated for permanent magnet problems.

B.13.2.2 Fields and their derivatives.

In principle one could obtain the fields by numerical diffrentiation of the potentials, but this is not a very accurate way of doing it. Furthermore, since the potentials are known only on points of the mesh, it would not be easy to calculate fields at points other than mesh points.

What POISSON and PANDIRA do, is fit a power series to the potential at a given point and then analytically take the derivatives of the series to get the field and its derivatives. This procedure is not only more accurate, but can also take advantage of the known symmetry of the magnet.

There is an important difference between the formulation of the problem in cartesian coordinates and cylindrical coordinates. In the cartesian case one can use the theory of complex variables, which has several advantages. The theory for cylindrical coordinates is slightly harder and will be treated separately at the end of this section.

In two-dimensional, cartesian coordinates, Maxwell's equations for the magnetic induction are

$$\frac{\partial}{\partial x}(\gamma B_y) - \frac{\partial}{\partial y}(\gamma B_x) = \mu_0 J, \tag{B.13.2.16}$$

and

$$\frac{\partial}{\partial x}(B_x) + \frac{\partial}{\partial y}(B_y) = 0. \tag{B.13.2.17}$$

Equation (B.13.2.17) can be satisfied by either

$$\mathbf{B} = \frac{\partial A_z}{\partial y}\hat{\mathbf{e}}_x - \frac{\partial A_z}{\partial x}\hat{\mathbf{e}}_y, \tag{B.13.2.18}$$

or

$$\mathbf{B} = -\frac{\partial V}{\partial x}\hat{\mathbf{e}}_x - \frac{\partial V}{\partial y}\hat{\mathbf{e}}_y, \tag{B.13.2.19}$$

where

$$\frac{\partial^2 V}{\partial x^2} + \frac{\partial^2 V}{\partial y^2} = 0. \tag{B.13.2.20}$$

The potential $V(x,y)$ is not a solution of Eq.(B.13.2.16) unless the current density $J(x,y)$ vanishes in the region under consideration. If we make one reasonable approximation, it is possible to reformulate the problem in terms of complex variables. Since we are interested in finding the magnetic potential $A_z(x,y)$ in the vicinity of a given point (x,y), we can assume that the reluctivity γ and the current density J are essentially constant in the vicinity of the given point. This means that we can move the reluctivity out of the differentiation and to the right-hand-side of Eq.(B.13.2.16). One obtains the equation

$$\frac{\partial^2 A_z}{\partial x^2} + \frac{\partial^2 A_z}{\partial y^2} = -\frac{\mu_0}{\gamma}J \tag{B.13.2.21}$$

for the potential $A_z(x,y)$. The solution to this equation can be written as the solution to the homogeneous equation plus a particular solution. It is easily verified that

$$A_z = A - \frac{\mu_0 J}{2\gamma}(x^2 + y^2), \tag{B.13.2.22}$$

where

$$\frac{\partial^2 A}{\partial x^2} + \frac{\partial^2 A}{\partial y^2} = 0. \tag{B.13.2.23}$$

We are now in a position to convert to the notation of complex variables.

Let $z = x+iy$ be a number in the complex plane. Mathematically speaking, there is an isomorphism between complex numbers z and points (x,y) in the cartesian plane. In current free regions the components of the magnetic induction $\mathbf{B}$ can be written in terms of either A or V as

$$B_x = \frac{\partial A}{\partial y} = -\frac{\partial V}{\partial x}, \tag{B.13.2.24}$$

and

$$B_y = -\frac{\partial A}{\partial x} = -\frac{\partial V}{\partial y}. \tag{B.13.2.25}$$

These relations betweeen A and V are the same as the Cauchy-Riemann conditions for a complex, analytic function

$$F(z) = A + iV. \tag{B.13.2.26}$$

This is easily seen by noting that

$$\frac{\partial F}{\partial x} = \frac{dF}{dz}\frac{\partial z}{\partial x} = \frac{dF}{dz}, \tag{B.13.2.27}$$

and

$$\frac{\partial F}{\partial y} = \frac{dF}{dz}\frac{\partial z}{\partial y} = i\frac{dF}{dz}, \tag{B.13.2.28}$$

and hence

$$\frac{dF}{dz} = \frac{\partial F}{\partial x} = i\frac{\partial F}{\partial y}. \tag{B.13.2.29}$$

If we define a complex, magnetic-induction function

$$B(z) = B_x + iB_y, \tag{B.13.2.30}$$

then

$$B(z)^* = i\frac{dF}{dz}. \tag{B.13.2.31}$$

where $*$ denotes complex conjugation. As the notation implies, $B(z)$ is also a complex, analytic function, which means that $B(z)$ can be expanded in a convergent power series in the variable z, provided that $B(z)$ is a single-valued function that does not have a singularity in the region of interest.

From Eqs.(B.13.2.19), (B.13.2.20), and (B.13.2.23) we see that both $A(x,y)$ and $V(x,y)$ satisfy Laplace's equation. In the theory of complex variables, functions with this property are called harmonic functions. Two other consequences of the Cauchy-Riemann conditions are: 1. given the vector potential $A(x,y)$, it is always possible to find the scalar potential $V(x,y)$ by integrating Eqs.(B.13.2.24) and (B.13.2.25); and 2. the curves $V(x,y) = v$ and $A(x,y) = a$ are orthogonal to one another for any constants v and a. Because the magnetic field lines are also orthogonal to the lines of constant potential, one can use the orthogonality of A and V to create plots of the field lines.

Let us turn now to the power series representation of the vector potential about a point z_o. This we write as

$$A(x_1, y_1) = Re\left\{\sum_{n=0}^{\infty} c_n(z_1 - z_0)^n\right\}, \tag{B.13.2.32}$$

or

$$A(x_1, y_1) = \sum_{n=0}^{\infty}\left[a_n u_{1,n} + (-b_n)v_{1,n}\right], \tag{B.13.2.33}$$

where

$$c_n = a_n + ib_n, \tag{B.13.2.34}$$

and

$$(z_1 - z_0)^n = u_{1,n} + iv_{1,n} \equiv u_n(x_i - x_0, y_i - y_0) + iv_n(x_i - x_0, y_i - y_0). \quad \text{(B.13.2.35)}$$

The polynomials u_n and v_n are called harmonic polynomials because they are also solutions of Laplace's equation. Table B.13.2.I shows the first 10 harmonic polynomials.

Table B.13.2.I
The First 10 Harmonic Polynomials[a]

n	$u_n(x,y)$	$v_n(x,y)$
1	x	y
2	$x^2 - y^2$	$2xy$
3	$x^3 - 3xy^2$	$3x^2y - y^3$
4	$x^4 - 6x^2y^2 + y^4$	$4x^3y - 4xy^3$
5	$x^5 - 10x^3y^2 + 5xy^4$	$5x^4y - 10x^2y^3 + y^5$
6	$x^6 - 15x^4y^2 + 15x^2y^4 - y^6$	$6x^5y - 20x^3y^3 + 6xy^5$
7	$x^7 - 21x^5y^2 + 35x^3y^4 - 7xy^6$	$7x^6y - 35x^4y^3 + 21x^2y^5 - y^7$
8	$x^8 - 28x^6y^2 + 7x^4y^4 - 28x^2y^6 + y^8$	$8x^7y - 56x^5y^3 + 56x^3y^5 - 8xy^7$
9	$x^9 - 36x^7y^2 + 126x^5y^4 - 84x^3y^6 + 9xy^8$	$9x^8y - 84x^6y^3 + 126x^4y5 - 36x^2y^7 + y^9$
10	$x^{10} - 45x^8y^2 + 210x^6y^4 - 210x^4y^6 + 45x^2y^8 - y^{10}$	$10x^9y - 120x^7y^3 + 252x^5y^5 - 120x^3y^7 + 10xy^9$

[a]The polynomials used in Eq.(B.13.2.33) are obtained from the polynomials in the table by replacing x by $(x_1 - x_0)$ and y by $(y_1 - y_o)$.

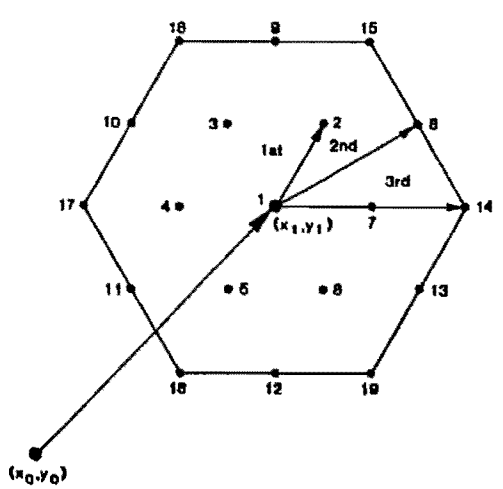


Figure B.13.2.4: First, second and third neighbors of a given point (x_1, y_1) on a regular triangular mesh. Points 2 through 7 are nearest neighbors; points 8 through 13 are second neighbors; and points 14 through 19 are third neighbors. The origin for the series expansion is (x_o, y_o).

The coefficients a_n and b_n are determined by truncating the power series at N terms and least-squares fitting the function $A(x,y)$ at neighboring points on the mesh. On a regular triangular mesh there are 6 first nearest neighbors, 6 second nearest neighbors and 6 third nearest neighbors, as illustrated in Fig. (B.13.2.4). Even after the mesh is distorted to conform to the boundaries of the physical regions, one can still identify these 18 points. Holsinger and Halbach lump the second and third neighbors together and call them second neighbors in the distorted mesh. Henceforth we will speak of six first neighbors and 12 second neighbors.

Suppose that we make a column matrix of the values of $A(x,y)$ evaluated at the point (x_1,y_1) and the first 18 neighbors of the point (x_1,y_1). We can make a column matrix out of the first $N=7$ pairs of coefficient (a_n,b_n). Next we construct an 19×14 matrix Z, whose rows are the coordinates of the points raised to the appropiate power and expressed in terms of the harmonic polynomials. Equation (B.13.2.33) can be written as

Equation (B.13.2.36)

$$\begin{bmatrix} 1 & 0 & u_{1,1} & v_{1,1} & u_{1,2} & \cdots & v_{1,6} \\ 1 & 0 & u_{2,1} & v_{2,1} & u_{2,2} & \cdots & v_{2,6} \\ 1 & 0 & u_{3,1} & v_{3,1} & u_{3,2} & \cdots & v_{3,6} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & 0 & u_{19,1} & v_{19,1} & u_{19,2} & \cdots & v_{19,6} \end{bmatrix} \begin{bmatrix} a_0 \\ b_0 \\ a_1 \\ \vdots \\ -b_6 \end{bmatrix} = \begin{bmatrix} A(x_1,y_1) \\ A(x_2,y_2) \\ A(x_3,y_3) \\ \vdots \\ A(x_{19},y_{19}) \end{bmatrix}$$

or in matrix form

$$ZC = A. \qquad \text{(B.13.2.36)}$$

Because the number of rows in Z is larger than the number of columns, this set of linear equations must be solved in the least-squares sense. There exist standard subroutines for doing this. One advantage of using least-squares is that the analytic function $A(x,y)$ will be smoother in the neighborhood of (x_1,y_1) than the values determined by POISSON in the neighborhood of this point. One disadvantage of this approach is the power series developed around the point (x_1,y_1) may not be consistent with the power series for the potential developed around the neighboring points (x_2,y_2), etc. In particular, artificial discontinuities are sometimes seen at points midway between neighboring points. This is a result of the truncation of the power series. The accuracy of the approximation depends on the distance between mesh points in the vicinity of the given point and the how rapidly the function $A(x,y)$ is changing. Let the user beware!

One way to make the power series expansion more accurate is to increase the number N, while still keeping the number of unknown coefficients a_n and b_n less than the 19 data points used in the least-squares fit. This can be done if there is any symmetry to the magnetic field. Suppose that the point (x_o,y_o) is a symmetry point of the magnet, that is, suppose that an axis of rotational symmetry passes

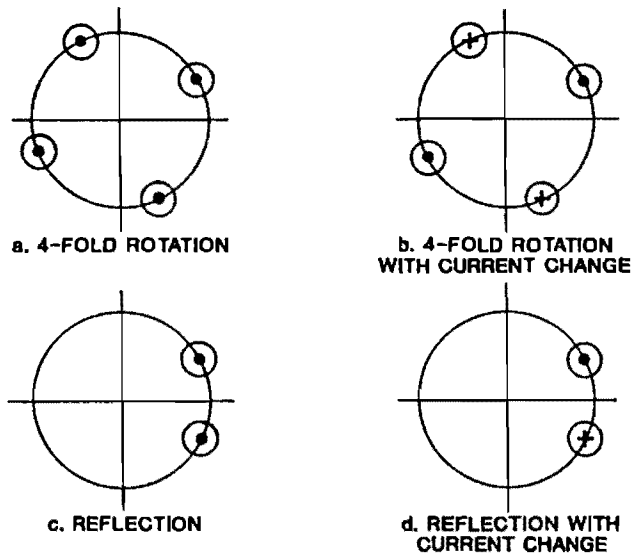


Figure B.13.2.5: Four basic types of symmetry that can occur in magnets: a. rotation, b. rotation and change of polarity, c. reflection, and d. reflection and change of polarity.

through this point or that this point lies in a plane of reflection symmetry. If there is a rotational axis, then some of the coefficients c_n must vanish. If there is a reflection plane, then c_n must be a real or imaginary number, depending on whether reflection changes the sign of the field or leaves it the same. Figure B.13.2.5 illustrate four types of symmetry that can occur in magnets. Figure B.13.2.5a shows a case of four-fold rotation symmetry for which no change in field polarity occurs with the rotation. Figure B.13.2.5b shows the same four-fold rotation symmetry, but this time, the field changes polarity. The change of polarity occurs because the current generating the field changes direction. This is like a reflection through the plane of the paper. Figure B.13.2.5c shows a case of reflection symmetry, where the reflection plane is perpendicular to the plane of the paper and passes through the x-axis. Finally, Fig. B.13.2.5d illustrates a reflection symmetry in which the polarity of the field changes sign. All other symmetries used in the codes are combinations of these basic symmetries. One can construct the set of all symmetry elements that leave the potential function $A(z, J)$ unchanged and use quite general group-theoretical methods to eliminate some of the coefficients c_n. The symmetry is simple enough that we will not introduce the full group-theoretical appartus.

Let us treat the rotational symmetry first. The effect of a rotation by an angle

α on a point (x, y) is

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} cos(\alpha) & -sin(\alpha) \\ sin(\alpha) & cos(\alpha) \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}. \qquad (B.13.2.37)$$

In complex notation this can be written

$$z' = \exp(i\alpha)z. \qquad (B.13.2.38)$$

The effect of this rotation on the vector potential is

$$A(e^{-i\alpha}z) = Re\left\{\sum_{n=0}^{\infty} c_n(e^{-in\alpha})z^n\right\}. \qquad (B.13.2.39)$$

Henceforth, for convenience we shall assume that the symmetry point z_o is the origin of coordinates. If this rotation is to leave the function invariant, then all the coefficients c_n must vanish except for n such that

$$e^{in\alpha} = 1 = e^{-2\pi iM}, \qquad (B.13.2.40)$$

where M is any integer. Furthermore, we know that the angle α must be some fraction of the number 2π. Let us write $\alpha = 2\pi/m$ and solve Eq.(B.13.2.40) for n. The result is

$$n = mM. \qquad (B.13.2.41)$$

For example, in the case illustrated in Fig. B.13.2.5a, $m = 4$ and the only nonvanishing coefficients are $c_o, c_4, ..., c_{4M}$.

If the rotation changes the polarity of the field, as illustrated in Fig. (B.13.2.5b), then the condition for nonvanishing coefficients is

$$A(e^{-2\pi i/m}z, -J) = Re\left\{\sum_{n=0}^{\infty} c_n(-J)e^{-2\pi in/m}z^n\right\} = A(z, J). \qquad (B.13.2.42)$$

Since changing the direction of the current must change the sign of the potential, we will assume that

$$c_n(-J) = -c_n(J) = e^{i\pi}c_n(J). \qquad (B.13.2.43)$$

It can be shown that this is equivalent to the condition that

$$cos(2\pi n/m) = -1, \qquad (B.13.2.44)$$

hence the argument of the cosine must be an odd multiple of π,

$$2\pi n/m = \pi(2M + 1), \qquad (B.13.2.45)$$

or

$$n = m(2M + 1)/2. \qquad (B.13.2.46)$$

Since n is an integer, m must be an even integer.

Let us now discuss the effects of reflection symmetry. This symmetry does not eliminate any of the coefficients, but instead tells us whether the c_n's are real or imaginary. The coordinate system can usually be arranged so that the x-axis is in a plane of reflection. This means that this reflection, which changes $(x+iy)$ to $(x-iy)$, is equivalent to complex conjugation. The effect of reflection on the vector potential is to replace z by z^* in its argument. The invariance condition becomes

$$A(z^*) = Re\left\{\sum_{n=0}^{\infty} c_n(J)(z^*)^n\right\} = A(z). \tag{B.13.2.47}$$

It can be shown that

$$(z^*)^n = (z^n)^* = u_n - iv_n, \tag{B.13.2.48}$$

and hence invariance requires that

$$Re\{c_n\}u_n + Img\{c_n\}v_n = Re\{c_n\}u_n - Img\{c_n\}v_n. \tag{B.13.2.49}$$

This implies that $Img\{c_n\}$ must be identically zero. In the same manner, it is easy to show that when $A(z,J)$ is invariant under reflection followed by a change in sign of the current, which changes $c_n(J)$ to $-c_n(J)$, that $Re\{c_n\}$ must vanish.

In summary then, symmetry reduces the number of unknown coefficients a_n and b_n. This means that one can include higher powers in the least-squares fit to the power series representation of the vector potential in the vicinity of a symmetry point. The use of symmetry also reduces the amount of information needed to generate the mesh for the problem. For example, only one-fourth of an H-shaped dipole magnet is required to describe the field for the full magnet. This will be discussed further in Sec. B.13.5 "Boundary Conditions and Meshes." Finally, it should be noted that the scalar potential

$$V(z,J) = Img\left\{\sum_{n=0} c_n(J)z^n\right\} \tag{B.13.2.50}$$

satisfies the same symmetry conditions and is easily generated.

Given the power series for either the vector potential or the scalar potential, it is easy to generate analytic expressions for the magnetic field and its derivatives. The magnetic field is given by

$$B(z) = -i\left(\frac{dF}{dz}\right)^* - \frac{\mu_0 J}{2\gamma}\left[\frac{\partial}{\partial y}(x^2+y^2) - i\frac{\partial}{\partial x}(x^2+y^2)\right]. \tag{B.13.2.51}$$

The second term comes from the particular solution to the inhomogenous equation. See Eqs.(B.13.2.21) and (B.13.2.22). Substitution of the power series for $F(z)$ gives the result

$$\begin{aligned} B(z) &= B_x + iB_y \qquad \text{(B.13.2.52)} \\ &= -\left\{\sum_{n=1}^{N} n(a_n v_{n-1} + b_n u_{n-1}) + \mu_0 J y/\gamma \right. \\ &\quad \left. + i\left[\sum_{n=1}^{N} n(a_n u_{n-1} - b_n v_{n-1}) + \mu_0 J x/\gamma\right]\right\}. \end{aligned}$$

The first derivatives of the field are found as follows. Special care must be taken regarding complex conjugation. The derivative of the complex potential $F(z)$ can be written

$$\frac{dF}{dz} = -i\left[B^* - \frac{\mu_0 J}{\gamma}(y + ix)\right] \equiv [-\overline{B}_y - i\overline{B}_x]. \qquad \text{(B.13.2.53)}$$

From this we can derive the Cauchy-Riemann conditions on the analytic function $dF(z)/dz$, namely,

$$\frac{d^2F}{dz^2} = -\frac{\partial \overline{B}_y}{\partial x} - i\frac{\partial \overline{B}_x}{\partial x} = i\frac{\partial \overline{B}_y}{\partial y} - \frac{\partial \overline{B}_x}{\partial y}, \qquad \text{(B.13.2.54)}$$

or

$$\frac{\partial \overline{B}_x}{\partial y} = \frac{\partial \overline{B}_y}{\partial x}, \qquad \text{(B.13.2.55)}$$

and

$$\frac{\partial \overline{B}_x}{\partial x} = -\frac{\partial \overline{B}_y}{\partial y}. \qquad \text{(B.13.2.56)}$$

Furthermore, since the power series for the second derivative can be written

$$\begin{aligned} \frac{d^2F}{dz^2} &= \sum_{n=2}^{N} n(n-1)c_n z^{n-2} \qquad \text{(B.13.2.57)} \\ &= \sum_{n=2}^{N}\left\{n(n-1)\left[(a_n u_{n-2} - b_n v_{n-2}) + i(a_n v_{n-2} + b_n u_{n-2})\right]\right\}, \end{aligned}$$

it follows that

$$\frac{\partial B_x}{\partial y} = -\sum_{n=2}^{N} n(n-1)(a_n u_{n-2} - b_n v_{n-2}) + \mu_0 J/\gamma \qquad \text{(B.13.2.58)}$$

and

$$\frac{\partial B_y}{\partial y} = \sum_{n=2}^{N} n(n-1)(a_n v_{n-2} + b_n u_{n-2}). \qquad \text{(B.13.2.59)}$$

As seen above, symmetry simplifies these general expression even further.

The electrostatic problem can be formulated in terms of the same complex potential $F(z)$. The only difference comes in the definition of $\mathbf{E}$ as the gradiant of A

instead of as the curl of A. This can be seen as follows. If we make the approximation that the dielectric constant ϵ and the charge density ρ are constant, then Maxwell's equations are

$$\nabla \times \mathbf{E} = 0 \tag{B.13.2.60}$$

and

$$\nabla \cdot \mathbf{E} = \rho/\epsilon. \tag{B.13.2.61}$$

Equation (B.13.2.60) has a solution of the form

$$\mathbf{E} = \frac{\partial V}{\partial y}\hat{\mathbf{e}_x} - \frac{\partial V}{\partial x}\hat{\mathbf{e}_y} \tag{B.13.2.62}$$

in two dimensions where the function $V(x,y)$ satisfies Laplace's equation. The solution to Eq.(B.13.2.61) can be written in the form

$$\mathbf{E} = -\frac{\partial \overline{A}}{\partial x}\hat{\mathbf{e}}_\mathbf{x} - \frac{\partial \overline{A}}{\partial y}\hat{\mathbf{e}}_\mathbf{y}, \tag{B.13.2.63}$$

where

$$\overline{A}(x,y) = A(x,y) - \rho(x^2+y^2)/(2\epsilon) \tag{B.13.2.64}$$

and $A(x,y)$ also satisfies Laplace's equation in two dimensions. The electrostatic potentials A and V are completely analogous to the magnetostatic potentials except for physical units (Volts as compared with Tesla-meters) and the fact the electric field is calculated as the gradiant of A as compared with calculating the magnetic induction as the curl of A.

In the discussion so far, we have assumed cartesian symmetry. The whole scheme will also work for cylindrical symmetry, but some changes have to be made in the formulas. The isomorphism to the complex plane is lost. The scalar and vector potentials must be treated separately. One wishes to express the potentials in the vicinity of a point (z_o, r_o) as a sum of polynomials. For convenience of writing we shall assume that the point (z_o, r_o) is the origin of coordinates. For the scalar potential we write

$$V(z,r) = \sum_{n=0}^{\infty} b_n v_n(z,r) - \frac{\rho}{\epsilon}(\frac{1}{8}r^2 + \frac{1}{4}z^2), \tag{B.13.2.65}$$

where

$$v_n(z,r) = \sum_{m=0}^{n} v_{mn} z^{n-m} r^m. \tag{B.13.2.66}$$

These polynomials must satisfy Laplace's equation in cylindrical coordinates, namely,

$$\frac{\partial^2 v_n}{\partial z^2} + \frac{1}{r}\frac{\partial}{\partial r}(r\frac{\partial v_n}{\partial r}) = 0. \tag{B.13.2.67}$$

It can be shown, by carrying out the above differentiations, that

$$v_{n1} = 0 \tag{B.13.2.68}$$

for all n and

$$v_{nm} = -\frac{(n-m+2)(n-m+1)}{m^2} v_{n,m-2}. \tag{B.13.2.69}$$

Taken together, these relations imply that $v_{nm} = 0$ for all odd m. Table B.13.2.II gives the first five polynomials.

Table B.13.2.II. Harmonic Polynomials for Cylindrical Coordinates

n	Scalar Potential v_n	Vector Potential u_n
1	z	r^2
2	$z^2 - r^2/2$	zr^2
3	$z^3 - 3zr^2/2$	$z^2r^2 - r^4/4$
4	$z^4 - 3z^2r^2 + 3r^4/8$	$z^3r^2 - 3zr^4/4$
5	$z^5 - 4z^3r^2/3 + z\,r^4/2$	$z^4r^2 - 3z^2r^4/2 + r^6/8$

The θ-component of the vector potential $A_\theta(z,r)$ also can be written as a sum of polynomials, but it is more convenient for computational purposes to multiply A_θ by r and write a polynomial series for the product, namely,

$$rA_\theta(r,z) = \sum_{n=1}^{\infty} a_n u_n(z,r) - \frac{\mu_0 J}{3\gamma} r^3, \tag{B.13.2.70}$$

where

$$u_n(z,r) = \sum_{m=0}^{n} u_{nm} z^{n-m} r^{m+1}, \tag{B.13.2.71}$$

and where the $u_n(z,r)$'s must satisfy the $\nabla \times \nabla \times \mathbf{A}$ equation

$$\frac{\partial^2}{\partial z^2}(rA_\theta) + r\frac{\partial}{\partial r}\left[\frac{1}{r}\frac{\partial}{\partial r}(rA_\theta)\right] = -\frac{\mu_0 J}{\gamma} r. \tag{B.13.2.72}$$

Once again, it can easily be shown that

$$u_{n0} = 0 \tag{B.13.2.73}$$

for all n, and

$$u_{nm} = -\frac{(n-m+2)(n-m+1)}{(m^2-1)} u_{nm-2}. \tag{B.13.2.74}$$

Taken together, these equations imply that $u_{nm} = 0$ for even n. The first five polynomials are given in Table B.13.II above.

The coefficients a_n and b_n are obtained by doing a least squares fit to the potentials at the nodes of the mesh as described above for cartesian coordinates.

For electrostatic problems the field components are

$$E_z = -\frac{\partial V}{\partial z} = -\sum_{n=1}^{\infty} \sum_{m=0}^{n}{}' (n-m) b_n v_{nm} z^{n-m-1} r^m + \frac{\rho}{2\epsilon} z \tag{B.13.2.75}$$

and

$$E_r = -\frac{\partial V}{\partial r} = -\sum_{n=1}^{\infty} \sum_{m=2}^{n}{}' m b_n v_{nm} z^{n-m} r^{m-1} + \frac{\rho}{4\epsilon} r. \tag{B.13.2.76}$$

where the prime on the summation symbol means a summation over even values of m. For magnetostatic problems the components of the induction are

$$B_z = \frac{1}{r}\frac{\partial}{\partial r}(rA_\theta) = \sum_{n=2}^{\infty} \sum_{m=1}^{n}{}'' (m+1) a_n u_{nm} z^{n-m} r^{m-1} - \frac{\mu_0}{\gamma} J r \tag{B.13.2.77}$$

and

$$B_r = -\frac{1}{r}\frac{\partial}{\partial z}(rA_\theta) = -\sum_{n=2}^{\infty} \sum_{m=1}^{n}{}'' (n-m) a_n u_{nm} z^{n-m-1} r^m. \tag{B.13.2.78}$$

where the double prime on the summation symbol means a summation over odd values of m. The program does not calculate any derivatives of the electrostatic field. It does calculate the r-derivative of B_z, which is given by the formula

$$\frac{dB_z}{dr} = \sum_{n=3}^{\infty} \sum_{m=3}^{n}{}'' (m+1)(m-1) a_n u_{nm} z^{n-m} r^{m-2} - \frac{\mu_0}{\gamma} J. \tag{B.13.2.79}$$

This ends the discussion of fields and their derivatives. The coding associated with this section is found in subroutines WORK and CWORK, which are contained in the program LIBSO.

B.13.2.3 Forces and torques.

Consider a conductor of cross sectional area S with the current flowing in the z-direction as shown in Fig. B.13.2.6. The force $\mathbf{F}$ on a section of thickness dz is given by the formula

$$\mathbf{F} = dz \int_S \mathbf{J} \times \mathbf{B} \; dxdy. \tag{B.13.2.80}$$

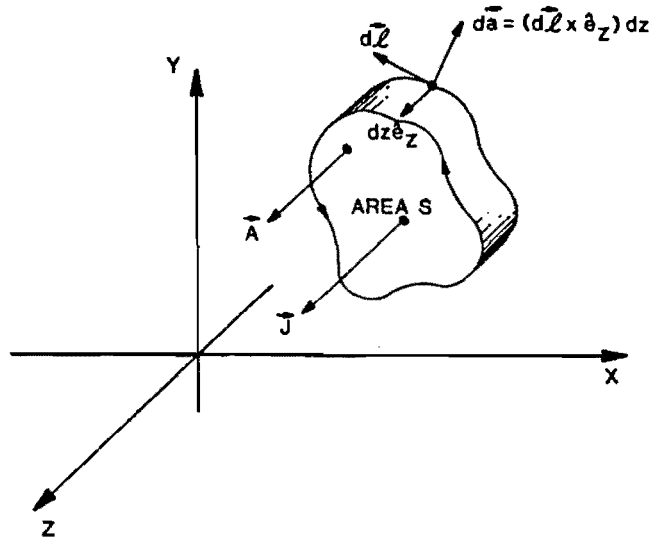


Figure B.13.2.6: Conductor of constant cross section with current flowing in the z-direction.

We will assume that the current density J is constant over the cross section, that is, independent of x and y. The magnetic induction $\mathbf{B}$ is independent of z and hence the vector potential $\mathbf{A}$ need only have a z-component; thus we can write

$$\mathbf{B} = \nabla \times \mathbf{A} = \frac{\partial A_z}{\partial y}\hat{\mathbf{e}}_\mathbf{x} - \frac{\partial A_z}{\partial x}\hat{\mathbf{e}}_\mathbf{y}. \tag{B.13.2.81}$$

It follows that we can write $\mathbf{J} \times \mathbf{B}$ in the form

$$\mathbf{J} \times \mathbf{B} = J_z\left(\frac{\partial A_z}{\partial x}\hat{\mathbf{e}}_\mathbf{x} + \frac{\partial A_z}{\partial y}\hat{\mathbf{e}}_\mathbf{y}\right). \tag{B.13.2.82}$$

We would like to use Stoke's Theorem to convert the integral in Eq. (B.13.2.80) from an area integral into a line integral. First we must convert the integral into vector form. Let the vector differential area be written as

$$d\mathbf{S} = dxdy\ \hat{\mathbf{e}}_\mathbf{z}. \tag{B.13.2.83}$$

Note that the following identities are true:

$$[\nabla \times (A_z\hat{\mathbf{e}}_\mathbf{y})]_z = \frac{\partial A_z}{\partial x} \tag{B.13.2.84}$$

$$[\nabla \times (A_z\hat{\mathbf{e}}_\mathbf{x})]_z = \frac{\partial A_z}{\partial y}. \tag{B.13.2.85}$$

Using these identities, we can write the force as

$$\mathbf{F} = J_z dz\left[\int \nabla \times (A_z\hat{\mathbf{e}}_\mathbf{y}) \cdot d\mathbf{S}\ \hat{\mathbf{e}}_\mathbf{x} - \int \nabla \times (A_z\hat{\mathbf{e}}_x) \cdot d\mathbf{S}\ \hat{\mathbf{e}}_\mathbf{y}\right]. \tag{B.13.2.86}$$

Stoke's Theorem can now be applied to each integral; the result is

$$\mathbf{F} = J_z dz \left(\oint_C A_z \hat{\mathbf{e}}_\mathbf{y} \cdot d\mathbf{l} \ \hat{\mathbf{e}}_\mathbf{x} - \oint_C A_z \hat{\mathbf{e}}_\mathbf{x} \cdot d\mathbf{l} \ \hat{\mathbf{e}}_\mathbf{y} \right). \tag{B.13.2.87}$$

This can also be written as

$$\mathbf{F} = J_z dz \left(\oint_C A_z dy \ \hat{\mathbf{e}}_\mathbf{x} - \oint_C A_z dx \ \hat{\mathbf{e}}_\mathbf{y} \right). \tag{B.13.2.88}$$

This is the form of the force on a current carrying element used in the code called FORCE.

The general formula for the torque on a thin slice of the conductor is given by the formula

$$\mathbf{T} = dz \int_S \mathbf{r} \times (\mathbf{J} \times \mathbf{B}) dx dy, \tag{B.13.2.89}$$

where

$$\mathbf{r} = x\hat{\mathbf{e}}_\mathbf{x} + y\hat{\mathbf{e}}_\mathbf{y}. \tag{B.13.2.90}$$

By using Eqs.(B.13.2.81) and (B.13.2.82) we find that

$$\mathbf{r} \times (\mathbf{J} \times \mathbf{B}) = J_z (x \frac{\partial A_z}{\partial y} - y \frac{\partial A_z}{\partial x}) \hat{\mathbf{e}}_\mathbf{z}. \tag{B.13.2.91}$$

Once again, we want to convert the area integral into a contour integral. The following identity is true:

$$[\nabla \times (-A_z \mathbf{r})]_z = x \frac{\partial A_z}{\partial y} - y \frac{\partial A_z}{\partial x}. \tag{B.13.2.92}$$

It follows that the torque can be written successively as

$$\mathbf{T} = -J_z \hat{\mathbf{e}}_\mathbf{z} dz \int_S \nabla \times (A_z \mathbf{r}) \cdot d\mathbf{S} \tag{B.13.2.93}$$

or

$$\mathbf{T} = -\mathbf{J} dz \oint_C A_z \mathbf{r} \cdot d\mathbf{l} \tag{B.13.2.94}$$

or

$$\mathbf{T} = -\mathbf{J} dz (\oint_C A_z x dx + \oint_C A_z y dy). \tag{B.13.2.95}$$

This is the form of the torque on a current carrying element used in FORCE. Note that the torque vector is parallel to the current density $\mathbf{J}$.

For problems with cylindrical symmetry, one calculates the force and torque on a thin wedge of materials. See Fig. B.13.2.7 for a picture of the geometry in the case of a toroidal conductor. For a general conductor, the cross section need not be a circle. The force on a wedge of angular width $d\theta$ is given by the formula

$$\mathbf{F} = d\theta \int_S \mathbf{J} \times \mathbf{B} \ r dr dz. \tag{B.13.2.96}$$

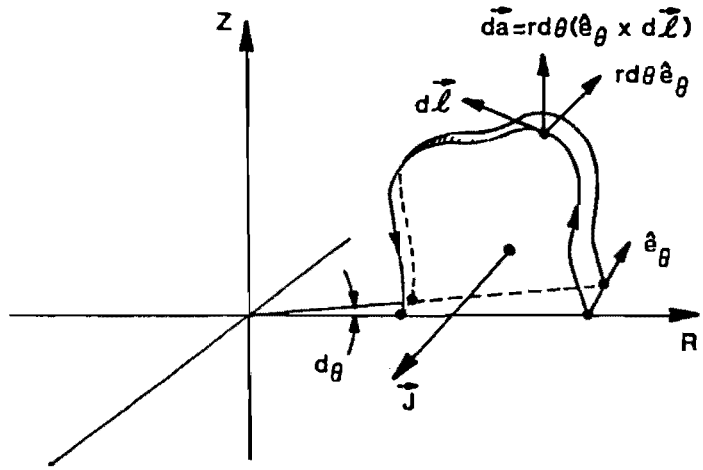


Figure B.13.2.7: Picture of a cylindrically symmetric conductor carrying a current density $\mathbf{J}$. Note that $\mathbf{J} = -J\hat{\mathbf{e}}_\theta$ and $\hat{\mathbf{e}}_\theta = \hat{\mathbf{e}}_z \times \hat{\mathbf{e}}_r$

Assuming that $\mathbf{J}$ is independent of r and z, and that $\mathbf{B}$ is independent of θ, one finds that

$$\mathbf{J} \times \mathbf{B} = -J\left[\frac{1}{r}\frac{\partial}{\partial r}(rA_\theta)\hat{\mathbf{e}}_\mathbf{r} + \frac{1}{r}\frac{\partial}{\partial z}(rA_\theta)\hat{\mathbf{e}}_\mathbf{z}\right] \tag{B.13.2.97}$$

and hence the force is

$$\mathbf{F} = -Jd\theta\left[\int_S \frac{\partial}{\partial r}(rA_\theta)drdz\hat{\mathbf{e}}_\mathbf{r} + \int_S \frac{\partial}{\partial z}(rA_\theta)drdz\hat{\mathbf{e}}_\mathbf{z}\right]. \tag{B.13.2.98}$$

We see that the function rA_θ in cylindrical coordinates is just like the function A_z in cartesian coordinates. The change from area integrals to line integrals is completely analogous. The final result is

$$\mathbf{F} = -Jd\theta\left[\oint_c rA_\theta dz\hat{\mathbf{e}}_\mathbf{r} - \oint_c rA_\theta dr\hat{\mathbf{e}}_\mathbf{z}\right]. \tag{B.13.2.99}$$

The final equation for the torque in cylindrical coordinates can be shown to be

$$\mathbf{T} = \mathbf{J}d\theta\left[\oint_c A_\theta r^2 dr + \oint_c A_\theta rzdz\right]. \tag{B.13.2.100}$$

The above treatment must be generalized when one wants to calculate the force and torque on the iron, which doesn't carry a current density $\mathbf{J}$. We will generalize the problem further by simultaneously treating electric as well as magnetic forces. The force density on a piece of ponderable matter is

$$\mathbf{f} = (\rho + \rho_p)\mathbf{E} + (\mathbf{J} + \mathbf{J}_m) \times \mathbf{B}, \tag{B.13.2.101}$$

where ρ_p is the polarization charge and J_m is the magnetization current density caused by aligning atomic magnetic dipoles. It is shown in the textbooks[5] that

$$\rho_p = -\nabla \cdot \mathbf{P} \tag{B.13.2.102}$$

and

$$\mathbf{J}_m = \nabla \times \mathbf{M}, \tag{B.13.2.103}$$

where

$$\mathbf{D} = \epsilon_0 \mathbf{E} + \mathbf{P} \tag{B.13.2.104}$$

and

$$\mathbf{H} = \gamma_0 B - \mathbf{M}. \tag{B.13.2.105}$$

Maxwell's equations in the static approximation are

$$\nabla \cdot \mathbf{D} = \rho \tag{B.13.2.106}$$

and

$$\nabla \times \mathbf{H} = \mathbf{J}. \tag{B.13.2.107}$$

Equations (B.13.2.101) through (B.13.2.107) give the following equation for the force density,

$$\mathbf{f} = \epsilon_0 (\nabla \cdot \mathbf{E})\mathbf{E} + \gamma_0 (\nabla \times \mathbf{B}) \times \mathbf{B}. \tag{B.13.2.108}$$

We next use two vector identities to convert this equation into one involving the electromagnetic stress tensor. It can be shown that

$$\nabla \cdot (\mathbf{EE}) = (\nabla \cdot \mathbf{E})\mathbf{E} + (\mathbf{E} \cdot \nabla)\mathbf{E} \tag{B.13.2.109}$$

and

$$\nabla(\mathbf{E} \cdot \mathbf{E}) = 2\mathbf{E} \times (\nabla \times \mathbf{E}) + 2(\mathbf{E} \cdot \nabla)\mathbf{E} \equiv \nabla \cdot (\mathbf{E} \cdot \mathbf{E} \overset{\Rightarrow}{I}). \tag{B.13.2.110}$$

Equation (B.13.2.109) is the divergence of a tensor formed from a product of the field vectors. Equation (B.13.2.110) is the gradiant of the scalar product. The gradiant can also be written as the divergence of a diagonal tensor. Combining these equations gives the relation

$$(\nabla \cdot \mathbf{E})\mathbf{E} = \nabla \cdot [\mathbf{EE} - \frac{1}{2}\mathbf{E} \cdot \mathbf{E} \overset{\Rightarrow}{I}] - \mathbf{E} \times (\nabla \times \mathbf{E}). \tag{B.13.2.111}$$

The last term in this equation is zero because of Maxwell's equations. The quantitiy in the square bracket is the electric part of the electromagnetic stress tensor,

$$\overset{\Rightarrow}{S}^{(E)} = \epsilon_0 [\mathbf{EE} - \frac{1}{2}\mathbf{E} \cdot \mathbf{E} \overset{\Rightarrow}{I}]. \tag{B.13.2.112}$$

The magnetic part of the force also can be expressed as the divergence of a tensor. We use the same vector identities written in terms of **B**,

$$\nabla \cdot (\mathbf{BB}) = (\nabla \cdot \mathbf{B})\mathbf{B} + (\mathbf{B} \cdot \nabla)\mathbf{B} \tag{B.13.2.113}$$

and

$$\nabla \cdot (\mathbf{B} \cdot \mathbf{B} \overset{\Rightarrow}{I}) = 2\mathbf{B} \times (\nabla \times \mathbf{B}) + 2(\mathbf{B} \cdot \nabla)\mathbf{B}. \tag{B.13.2.114}$$

Combining these equations gives

$$(\nabla \times \mathbf{B}) \times \mathbf{B} = \nabla \cdot (\mathbf{BB} - \frac{1}{2}\mathbf{B} \cdot \mathbf{B} \overset{\Rightarrow}{I}) - (\nabla \cdot \mathbf{B})\mathbf{B}. \tag{B.13.2.115}$$

The last term vanishes because of Maxwell's equations. The magnetic stress tensor is defined as

$$\overset{\Rightarrow}{S}^{(M)} = \gamma_0[\mathbf{BB} - \frac{1}{2}\mathbf{B} \cdot \mathbf{B} \overset{\Rightarrow}{I}]. \tag{B.13.2.116}$$

The total force on a volume V is

$$\mathbf{F} = \int_V \nabla \cdot (\overset{\Rightarrow}{S}^{(E)} + \overset{\Rightarrow}{S}^{(M)}) dv. \tag{B.13.2.117}$$

This can be transformed into a surface integral by Green's theorem

$$\mathbf{F} = \int_S (\overset{\Rightarrow}{S}^{(E)} + \overset{\Rightarrow}{S}^{(M)}) \cdot d\mathbf{a}. \tag{B.13.2.118}$$

Let us consider the volume shown in Fig. B.13.2.6. For two dimensional cartesian symmetry, we know that $B_z = 0$ and $\mathbf{B}$ is independent of the z-coordinate. This means that the integral over the front and back surfaces contribute nothing to the surface integral; $d\mathbf{a}$ is perpendicular to $\mathbf{B}$ and $\mathbf{E}$ and therefore

$$(\mathbf{EE} + \mathbf{BB}) \cdot d\mathbf{a} = \mathbf{E}(\mathbf{E} \cdot d\mathbf{a}) + \mathbf{B}(\mathbf{B} \cdot d\mathbf{a}) = 0. \tag{B.13.2.119}$$

For the $(\mathbf{E} \cdot \mathbf{E} + \mathbf{B} \cdot \mathbf{B})$ term, the contribution from the front face cancels the contribution from the back face. The integral on the ribbon edge of thickness dz is the whole integral. The differential area $d\mathbf{a}$ can be written

$$d\mathbf{a} = -d\mathbf{l} \times \hat{\mathbf{e}}_z dz = -(dy\hat{\mathbf{e}}_x - dx\hat{\mathbf{e}}_y)dz, \tag{B.13.2.120}$$

where the differential vector $d\mathbf{l}$ is pointing counterclockwise along the contour of the edge. The integrand is independent of z and therefore the surface integral becomes a contour integral in the xy-plane. After some tedious vector algebra the contour integral can be expressed in cartesian components. The resulting expressions for the components of the force are

$$\begin{aligned} F_x &= \tfrac{1}{2}dz \left\{ \epsilon_0 \oint \left[(E_x^2 - E_y^2)dy - 2E_xE_y dx \right] \right. \\ &\quad \left. + \gamma_0 \oint \left[(B_x^2 - B_y^2)dy - 2B_xB_y dx \right] \right\} \end{aligned} \tag{B.13.2.121}$$

and

$$F_y = \tfrac{1}{2}dz \left\{\epsilon_0 \oint \left[(E_x^2 - E_y^2)dx + 2E_xE_ydy\right] + \gamma_0 \oint \left[(B_x^2 - B_y^2)dx + 2B_xB_ydy\right]\right\}. \tag{B.13.2.122}$$

The beauty of these results is that in deriving them we had to make no assumption about the linearity of the relation between the fields **E** and **D** and between **B** and **H**. They are true with and without external charges and currents. Furthermore, there was no requirement that the charge or current densities be constant over the cross section of the material. The parallelism between the electric and magnetic parts of the force suggest that the coding will be the same for electrostatic and magnetostatic problems.

For completeness we give the formulas for the torque in cartesian coordinates, and for the force and torque in cylindrical coordinates. Because of the parallelism between electric and magnetic forces, only the magnetic part is recorded in these formulas:

$$T_z = dz\left(\gamma_0 \oint \left\{[xB_xB_y - \frac{1}{2}y(B_x^2 - B_y^2)]dy + [\frac{1}{2}x(B_x^2 - B_y^2) + yB_xB_y]dx\right\}\right) \tag{B.13.2.123}$$

$$\mathbf{F} = -\gamma d\theta\left\{\oint [r(\frac{1}{2}\left\{B_r^2 - B_z^2\right\}dz - B_rB_zdr)\hat{\mathbf{e}}_r + r(\frac{1}{2}\left\{B_r^2 - B_z^2\right\}dr + B_rB_zdz)\hat{\mathbf{e}}_z]\right\} \tag{B.13.2.124}$$

$$\mathbf{T} = \gamma_0 d\theta \hat{\mathbf{e}}_\theta(\theta) \oint \left\{\left[\frac{z}{2}(B_r^2 - B_z^2) - rB_rB_z\right]rdz - \left[\frac{r}{2}(B_r^2 - B_z^2) + zB_rB_z\right]rdr\right\}. \tag{B.13.2.125}$$

Note that the torque depends on the angle θ through the unit vector $\hat{\mathbf{e}}_\theta$. When integrated over the angle θ the result is identically zero.

In the case of cartesian summetry it is interesting to note that the magnetic force and torque can be written in complex variables as

$$F = F_x + iF_y = \frac{1}{2}L\gamma_0 i \oint B^2 dz^* \tag{B.13.2.126}$$

and

$$T_z = \frac{1}{2} L \gamma_0 R \epsilon \left\{ \oint z (B^*)^2 dz \right\}. \tag{B.13.2.127}$$

This concludes the discussion of auxiliary properties.

B.13.3 Harmonic Analysis

Harmonic analysis of the field gives a numerical estimate of the multipole content of the field in the gap of a magnet. Figure B.13.3.1 illustrates the parameters involved for an H-shaped dipole magnet. One generally chooses a point (usually the origin of coordinates for the problem) about which to do the analysis and a radius r of a circle about that point. The radius r should be chosen so that the circle does

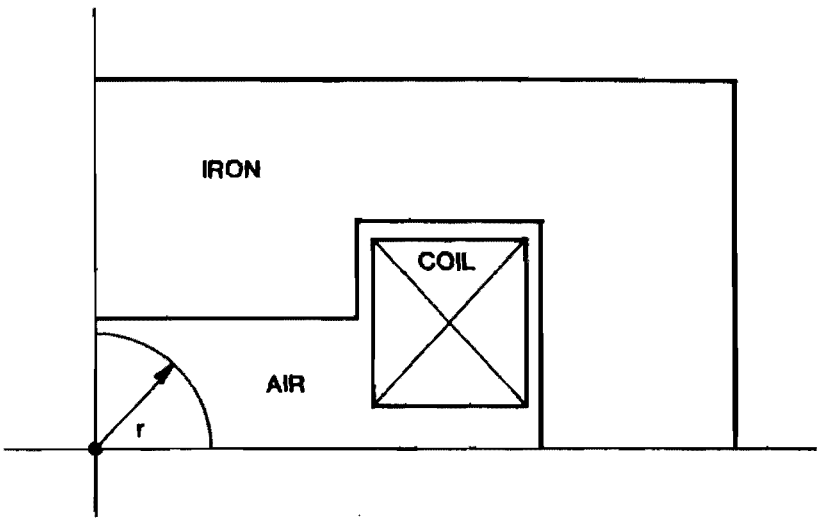


Figure B.13.3.1: One-quarter of a symmetric H-shaped magnet. The harmonic analysis will be done about the center of the gap on circular arc of radius r.

not enclose any iron or coil region. The distance between the circle and the pole piece should be larger than the mesh increment DY.

The mathematical theory is based on the idea that the vector potential $A(x, y)$ can be thought of as the real part of a complex function $F(z = x + iy)$. This idea is explained in Sec. B.13.2 above. The function $F(z)$ can be expanded in a power series

$$F(z) = A(x, y) + iV(x, y) = \sum_{n=0}^{\infty} (c_n / r_{norm}^n) z^n, \tag{B.13.3.1}$$

where r_{norm} is a normalization radius. With this definition, the c_n's all have the same units, e.g., gauss-cm. If we let

$$c_n = a_n + ib_n, \tag{B.13.3.2}$$

and

$$z^n = u_n + iv_n, \tag{B.13.3.3}$$

then the vector potential is given by

$$A(x,y) = \sum_{n=0}^{\infty}(a_n u_n - b_n v_n)/r_{norm}^n. \tag{B.13.3.4}$$

The quantities u_n and v_n are called harmonic polynomials; the first ten of them are found in Table B.13.2.I. Anyone familar with magnets will recognize that they generate the regular and skew multipole fields. This means that the representation of the potential in terms of this power series is equivalent to a multipole decomposition of the potential.

To do the decomposition, we must find the coefficients a_n and b_n. This is done by evaluating the potential on the circle. The simplest way to do this is to go to polar coordinate in the complex plane, expressing z by the formula

$$z = r\exp(i\varphi), \tag{B.13.3.5}$$

and the coefficients c_n by the formula

$$c_n = |c_n|\exp(i\alpha_n) = |c_n|\cos(\alpha_n) + i|c_n|\sin(\alpha_n). \tag{B.13.3.6}$$

When these relations are substituted into the formula for the vector potential, one obtains the sequence of equations

$$\begin{aligned} A(r,\varphi) &= Re\left\{\sum_{n=0}^{\infty}\left(\frac{r}{r_{norm}}\right)^n |c_n|e^{i\alpha_n}e^{in\varphi}\right\} \\ &= \sum_{n=0}^{\infty}\left(\frac{r}{r_{norm}}\right)^n |c_n|\cos(\alpha_n + n\varphi) \\ &= \sum_{n=0}^{\infty}\left(\frac{r}{r_{norm}}\right)^n \Big[|c_n|\cos(\alpha_n)\cos(n\varphi) - |c_n|\sin(\alpha_n)\sin(n\varphi)\Big] \\ &= \sum_{n=0}^{\infty}\left(\frac{r}{r_{norm}}\right)^n \Big[a_n\cos(n\varphi) - b_n\sin(n\varphi)\Big]. \end{aligned} \tag{B.13.3.7}$$

This is in the form of a Fourier series in the variable φ. Fourier analysis theory tells us that

$$a_0 = \frac{1}{2\pi}\int_o^{2\pi} A(r,\varphi)d\varphi,\ b_n = 0, \tag{B.13.3.8}$$

$$a_n = \frac{1}{\pi}\left(\frac{r_{norm}}{r}\right)^n \int_o^{2\pi} A(r,\varphi)\cos(n\varphi)d\varphi,\ n \geq 1 \tag{B.13.3.9}$$

and

$$b_n = -\frac{1}{\pi}\left(\frac{r_{norm}}{r}\right)^n \int_o^{2\pi} A(r,\varphi)\sin(n\varphi)d\varphi,\ n \geq 1. \tag{B.13.3.10}$$

The circle of integration should not include any part of an iron or coil region, because this is inconsistent with the assumptions underlying the use of the complex potential $F(z)$.

To take advantage of the magnet symmetry, Holsinger introduced two parameters into the integrations. Consider the magnet shown in Fig. B.13.3.1. Because of the field symmetry one does not need to integrate from $\varphi = 0$ to $\varphi = 2\pi$. In this case, it can be shown that the integrals from $\varphi = 0$ to $\varphi = 2\pi$ are four times the integrals from $\varphi = 0$ to $\varphi = \frac{\pi}{2}$.

We can generalize the limits of integration to be from $\varphi = \phi_z$ to $\varphi = \phi$, where φ is measured from the horizontal axis (x- or r-axis). The new definitions of the harmonic coefficients are:

$$a_0 = \frac{1}{\phi - \phi_z} \int_{\phi_z}^{\phi} A(r,\varphi) d\varphi, \quad b_0 = 0, \tag{B.13.3.11}$$

$$a_n = \frac{2}{\phi - \phi_z} \int_{\phi_z}^{\phi} \left(\frac{r_{norm}}{r}\right)^n A(r,\varphi) \cos(n\ \varphi), d\varphi, \tag{B.13.3.12}$$

$$b_n = -\frac{2}{\phi - \phi_z} \int_{\phi_z}^{\phi} \left(\frac{r_{norm}}{r}\right)^n A(r,\varphi) \sin(n\ \varphi), d\varphi. \tag{B.13.3.13}$$

The coefficients a_0 and b_0 are not needed to calculate the magnetic field and are not printed out by the program.

The integrals are done numerically by converting them to summations. To make this conversion we let

$$d\varphi \longrightarrow \Delta\varphi = \frac{\phi - \phi_z}{N_{ptc} - 1}\ , \tag{B.13.3.14}$$

where N_{ptc} is the number of equidistant points on the arc of radius r, at which points the vector potential $A(r,\varphi)$ is to be interpolated in carrying out the Fourier analysis.

One cannot, of course, obtain all the coefficients a_n and b_n, but must, for practical reasons, limit oneself to $n \leq N_{term}$, where the integer N_{term} is the number of coefficients to be obtained. Table B.13.3.I summarizes the input parameters for the problem.

Table B.13.3.I Input Parameters for Harmonic Analysis

Name	Default	Definition
CON(110) = NTERM	5	the number of coefficients to be obtained
CON(111) = NPTC	none	the number of equidistant points on the arc of a circle with radius RINT, with its center at the origin, (0, 0), at which points the vector potential is to be interpolated. Fourier analysis of the vector potential at these points yields the harmonic coefficients. NPTC should be approximately equal to the number of mesh points adjacent to the arc for best results.
CON(112) = RINT	none	the radius of the arc on which the vector potential is interpolated for the Fourier analysis. RINT should be less than, by at least one mesh space (triangle side), the shortest distance from the origin to the nearest singularity, i.e., a pole face or a coil.
CON(113) = ANGLE	none	the extent of the interpolation arc included in the problem, i.e., $(\phi - \phi_z)$ in Eqs. B.13.3.11 to B.13.3.14 above. ANGLE is measured in degrees.
CON(114) = RNORM	none	the normalization radius. It is often chosen to be either RINT or the radius of the magnet aperture.
CON(115) = ANGLZ	0.0	the angle in degrees from the horizontal axis to where the integration arc begins.
CON(108) = AROTAT	0.0	an additional rotation angle added to ANGLZ (This is apparently an artifact of a partially implemented modification to the code. The sophisticated user may want to look into subroutine MINT and finish the implementation.)

The harmonic analysis of the magnetic induction starts with the series relation

$$B_x - iB_y = i\frac{dF}{dz} = \sum_{n=1}^{\infty} in\left(\frac{r}{r_{norm}}\right)^{n-1}\frac{c_n}{r_{norm}}\exp[i(n-1)\varphi] \qquad \text{(B.13.3.15)}$$

This can be equated to a Fourier series for the induction, which we write in the form

$$B_x - iB_y = \sum_{m=0}^{\infty} B_m \exp(im\varphi) \qquad \text{(B.13.3.16)}$$

The strengths of the harmonic components of the induction are directly related to the a_n's and b_n's found in analyzing $A(x,y)$. The formula is

$$B_m = -\frac{(m+1)}{(r_{norm})}\left(\frac{r}{r_{norm}}\right)^m (b_{m+1} - ia_{m+1}) \qquad \text{(B.13.3.17)}$$

POISSON and PANDIRA print out the quantities a_n, b_n, na_n/r_{norm}, and nb_n/r_{norm}. The program also gives absolute values of the complex quantities c_n and nc_n/r_{norm}, which do not have a simple interpretation.

Figure B.13.3.2 is printout from a harmonic analysis done on a dipole magnet like the one shown in Fig. B.13.3.1. Note that the "R" in Fig. B.13.3.2 below is the normalization radius, not the radius of integration discussed above.

```
1HARMONIC ANALYSIS
0INTEGRATION RADIUS =   1.50000
0TABLE FOR INTERPOLATED POINTS.
0        N     ANGLE   XCOORD3   Y COORD   KF   LF       VEC.POT.

         1    0.0000    1.5000    0.0000    4    1   -2.39835E+04
         2    9.0000    1.4815    0.2347    5    2   -2.36889E+04
         3   18.0000    1.4266    0.4635    5    2   -2.28119E+04
         4   27.0000    1.3365    0.6810    4    3   -2.13736E+04
         5   36.0000    1.2135    0.8817    4    3   -1.94085E+04
         6   45.0000    1.0607    1.0607    4    4   -1.69649E+04
         7   54.0000    0.8817    1.2135    3    4   -1.41027E+04
         8   63.0000    0.6810    1.3365    3    4   -1.08929E+04
         9   72.0000    0.4635    1.4266    2    5   -7.41450E+03
        10   81.0000    0.2347    1.4815    1    5   -3.75347E+03
        11   90.0000    0.0000    1.5000    1    5    6.76941E-03
1TABLE FOR VECTOR POTENTIAL COEFFICIENTS
0NORMALIZATION RADIUS =   2.00000
0        A(X,Y) = RE( SUM (AN + I BN) * (Z/R)**N )
0         N        AN              BN            ABS(CN)
0         1    -3.1984E+04     0.0000E+00     3.1984E+04
0         3     7.8190E+00     0.0000E+00     7.8190E+00
0         5     3.8486E+00     0.0000E+00     3.8486E+00
0         7     8.9108E-01     0.0000E+00     8.9108E-01
0         9     9.6246E-02     0.0000E+00     9.6246E-02
1TABLE FOR FIELD COEFFICIENTS
0NORMALIZATION RADIUS =   2.00000
0        (BX - I BY) = I * SUM N*(AN + I BN)/R * (Z/R)**(N-1)
0         N     N(AN)/R        N(BN)/R      ABS(N(CN)/R)
0         1    -1.5992E+04     0.0000E+00     1.5992E+04
0         3     1.1728E+01     0.0000E+00     1.1728E+01
0         5     9.6214E+00     0.0000E+00     9.6214E+00
0         7     3.1188E+00     0.0000E+00     3.1188E+00
0         9     4.3310E-01     0.0000E+00     4.3310E-01
```

Figure B.13.3.2: Portion of printout from the POISSON output file for the H-shaped dipole magnet example from Chapter B.2., showing the harmonic analysis.

The user should be cautious in using the harmonic coefficients in any practical design, unless he is sure of the symmetry implications. Strictly speaking, the integrals in Eqs. B.13.3.9 and B.13.3.10 are over the range from zero to 2π. Holsinger has used symmetry to decrease the range of integration. If one does not have the correct symmetry type (ITYPE = CON(46)), some of the a_n's and b_n's will be wrong. Furthermore there are some symmetry types that do not fit into the Halbach-Holsinger scheme, and the only way that they can be treated is to put in the full magnet geometry and set ITYPE = 1. In particular, be careful of the case where the only symmetry is a reflection through the y-axis along with a change in current. See Fig. B.13.3.3.

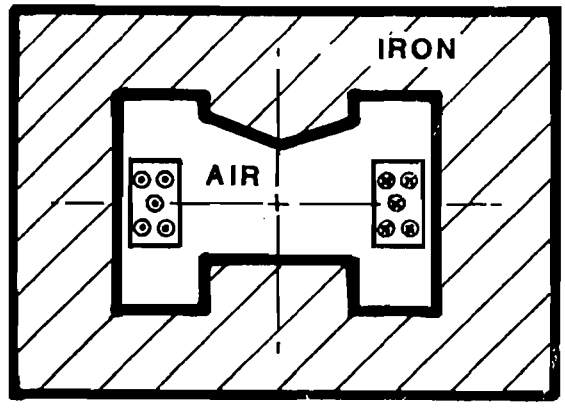


Figure B.13.3.3: Example of a magnet with reflection plus current change.

B.13.4 Conformal Transformations

Conformal transformations can be quite useful in trimming multipole magnets. Presently, the program MIRT can only be used to trim dipole magnets, that is, to make the field in a given region as uniform as possible by adjusting current densities and boundaries of iron regions. Through the use of conformal transformations MIRT can be used to remove higher multipole components of the field in quadrupole magnets. This is done by conformally transforming the quadrupole field to a dipole field, making the dipole field more uniform, and transforming back to the quadrupole geometry.

The theory behind this method has been explained quite well by K. Halbach.[6,7] There is very little that can be added to what he has said. We have reproduced these articles here. More recently Robert Lari of Argonne National Lab. has tested the procedure using our standard version of POISSON. We have included here his Light Source Note, LS-32, which may clarify further the practical details of carrying out the procedure.

NUCLEAR INSTRUMENTS AND METHODS 64 (1968) 278–284; © NORTH-HOLLAND PUBLISHING CO.

APPLICATION OF CONFORMAL MAPPING TO EVALUATION AND DESIGN OF MAGNETS CONTAINING IRON WITH NONLINEAR $B(H)$ CHARACTERISTICS*

K HALBACH

Lawrence Radiation Laboratory, Berkeley, California, U.S.A.

Received 10 May 1968

It is shown that for evaluation of two-dimensional magnets with nonlinear iron in a conformally transformed geometry, only minor changes of the magnetostatic equations are required. The advantages resulting from evaluation or design of a magnet in a suitably transformed geometry are discussed in detail.

1. Introduction

Conformal mapping is a powerful technique for finding solutions, or for simplifying the process of finding solutions, to Laplace's differential equation in two dimensions, and a large number of applications to many fields can be found in any textbook dealing with this subject. This method has also been applied successfully to the design of long magnets with infinite permeability of the iron that, far away from the ends, can be described with sufficient accuracy by the two dimensional Laplace equation[1,2]. However, it does not seem to be generally known that application of a conformal transformation to a two dimensional multipole can greatly simplify the evaluation or design of that type of magnet even when the iron has nonlinear $B(H)$ characteristics. It is the purpose of this paper to point out some of the advantages that result if such a magnet is conformally transformed and then, in the new geometry, evaluated with a digital computer that solves Poisson's equation numerically. To this end, we write first the magnetostatic equations in the original coordinate system in such a way that it will be easy to transform them to the new coordinate system. From the transformed equations we then deduce the modifications that have to be incorporated in the computer code that numerically integrates the normal magnetostatic equations with nonlinear $B(H)$ characteristics. In the discussion of the application of conformal transformations to two dimensional magnets with nonlinear iron, emphasis is on the description of the advantages that result when the magnetostatic equations are solved numerically. However, to give a complete picture, we will also point out some of the generally known benefits associated with conformal transformations when applied to this kind of problem.

2. Magnetostatic differential equations in the original and conformally transformed coordinates

2.1. Notation

Units used throughout are mks. Complex numbers and operators are identified by underlining; their complex conjugate by an asterisk. The absolute value of a complex number is indicated by two vertical bars, and its real part by Re. The Cartesian coordinates of the original problem are x and y; the Cartesian coordinates of the transformed problem are u and v, and they are related to x and y through a suitably chosen conformal transformation

$$u+\mathrm{i}v \equiv \underline{w} = \underline{w}(x+\mathrm{i}y) \equiv \underline{w}(\underline{z}). \tag{1}$$

Quantities that depend directly on x and y, or are of special significance in the x, y coordinate system, carry the subscript z, and similarly carry the subscript w when they depend directly on u and v, or are of special significance in the u, v coordinate system.

The reason to consider only conformal transformations is the well known fact that the structure of the magnetostatic equations is destroyed under any other than conformal transformations.

2.2. Magnetostatic differential equation in original coordinate system

Without loss of generality, we can derive the two components B_x and B_y of the magnetic flux density from a vector potential which has only a component (A_z) perpendicular to the x-y plane. Introducing the complex field quantity

$$\underline{B}_z = B_x+\mathrm{i}B_y = \frac{\mathrm{d}}{\mathrm{d}y}A_z - \mathrm{i}\frac{\mathrm{d}}{\mathrm{d}x}A_z = -\mathrm{i}\left(\frac{\mathrm{d}}{\mathrm{d}x}A_z + \mathrm{i}\frac{\mathrm{d}}{\mathrm{d}y}A_z\right) \tag{2}$$

and the complex operator

$$\underline{D}_z = -\mathrm{i}\left(\frac{\mathrm{d}}{\mathrm{d}x} + \mathrm{i}\frac{\mathrm{d}}{\mathrm{d}y}\right), \tag{3a}$$

we obtain

$$\underline{B}_z = \underline{D}_z A_z. \tag{4}$$

* Work performed under the auspices of the U.S. Atomic Energy Commission. AEC Contract no. W-405-eng-48.

It should be noted that $\underline{D}_z$ acting on any analytical function $\underline{K}(\underline{z})$ is zero:

$$\underline{D}_z\underline{K}(\underline{z}) = 0. \tag{5}$$

Assuming an isotropic medium, and introducing γ, the reciprocal of the relative permeability μ (which may depend both on location and flux density),

$$\gamma_z(x,y,|\underline{B}_z|) = 1/\mu_z(x,y,|\underline{B}_z|),$$

we obtain for the field components H_x, H_y, and the complex field quantity

$$\underline{H}_z = H_x + \mathrm{i}H_y,$$
$$\underline{H}_z = (1/\mu_0)\gamma_z\underline{B}_z = (1/\mu_0)\gamma_z\underline{D}_zA_z. \tag{6}$$

The magnetostatic equation relating H_x, H_y to the current density j_z in the direction perpendicular to the x-y plane is:

$$j_z(x,y) = \frac{\mathrm{d}}{\mathrm{d}x}H_y - \frac{\mathrm{d}}{\mathrm{d}y}H_x = -\mathrm{Re}\left\{\mathrm{i}\left(\frac{\mathrm{d}}{\mathrm{d}x} - \mathrm{i}\frac{\mathrm{d}}{\mathrm{d}y}\right)\right\}(H_x + \mathrm{i}H_y).$$

With eqs. (3a) and (6), we obtain therefore for the magnetostatic differential equation in the x-y coordinate system:

$$\mathrm{Re}\,\underline{D}_z^*\gamma_z(x,y,|\underline{B}_z(x,y)|)\underline{D}_zA_z(x,y) = -\mu_0 j_z(x,y). \tag{7}$$

It should be noted that when $\underline{D}_z^*$ acts on γ_z, it acts not only on the explicit dependence of γ_z on x, y, but also on the x, y dependence that results from the dependence of γ_z on $|\underline{B}_z(x,y)|$. This is, of course, the reason why total and not partial derivatives are used in defining $\underline{D}_z$.

2.3. Magnetostatic differential equation in transformed coordinate system

Introducing new coordinates u,v through the conformal transformation, eq. (1), we can express x and y in $A_z(x,y)$ through u and v, obtaining a new function $A_w(u,v)$. The implicit dependence of A_w on x,y is of course the same as the direct dependence of A_z on x,y:

$$A_z(x,y) = A_w\{u(x,y), v(x,y)\}. \tag{8}$$

To obtain the magnetostatic equation in u, v, we express $\underline{D}_z$ through derivatives with respect to u, v. From eq. (3a) we get:

$$\underline{D}_z = -\mathrm{i}\left\{\frac{\mathrm{d}u}{\mathrm{d}x}\frac{\mathrm{d}}{\mathrm{d}u} + \frac{\mathrm{d}v}{\mathrm{d}x}\frac{\mathrm{d}}{\mathrm{d}v} + \mathrm{i}\left(\frac{\mathrm{d}u}{\mathrm{d}y}\frac{\mathrm{d}}{\mathrm{d}u} + \frac{\mathrm{d}v}{\mathrm{d}y}\frac{\mathrm{d}}{\mathrm{d}v}\right)\right\}.$$

Using the Cauchy-Riemann relations:

$$\frac{\mathrm{d}u}{\mathrm{d}y} = -\frac{\mathrm{d}v}{\mathrm{d}x}; \quad \frac{\mathrm{d}v}{\mathrm{d}y} = \frac{\mathrm{d}u}{\mathrm{d}x},$$

we obtain, with

$$\underline{D}_w = -\mathrm{i}\left(\frac{\mathrm{d}}{\mathrm{d}u} + \mathrm{i}\frac{\mathrm{d}}{\mathrm{d}v}\right) \tag{3b}$$

and

$$\underline{w}' = \mathrm{d}\underline{w}/\mathrm{d}\underline{z} = 1/\underline{z}':$$
$$\underline{D}_z = \left(\frac{\mathrm{d}u}{\mathrm{d}x} - \mathrm{i}\frac{\mathrm{d}v}{\mathrm{d}x}\right)\underline{D}_w = \underline{w}'^*\underline{D}_w. \tag{9}$$

For the relation between $\underline{B}_w = \underline{D}_wA_w$ and $\underline{B}_z$ we obtain from eqs. (4), (8) and (9):

$$\underline{B}_z = \underline{w}'^*\underline{B}_w = \underline{B}_w/\underline{z}'^*. \tag{10}$$

It should be noted that eq. (10) is identical to the transformation formulas that one obtains if one assumes that the fields can be derived from a complex potential function, although that has obviously not been assumed in the derivation of eq. (10).

Using, similarly to eq. (8), and with eq. (10)

$$\gamma_z(x,y,|\underline{B}_z|) = \gamma_w\{u(x,y), v(x,y), |\underline{B}_w|/|\underline{z}'|\}, \tag{11}$$

we obtain from eqs. (7), (8) and (9):

$$\mathrm{Re}\,\underline{D}_z^*\underline{w}'^*\gamma_w\underline{D}_wA_w = -\mu_0 j_z.$$

Because of eq. (5), we can write $\underline{w}'^*$ to the left of $\underline{D}_z^*$. We thus obtain, using eq. (9) again:

$$|\underline{w}'|^2\cdot\mathrm{Re}\,\underline{D}_w^*\gamma_w\underline{D}_wA_w = -\mu_0 j_z.$$

Introducing

$$j_z\{x(u,v), y(u,v)\}\cdot|\underline{z}'|^2 = j_w(u,v), \tag{12a}$$

we get for the magnetostatic differential equation in the u, v coordinate system:

$$\mathrm{Re}\,\underline{D}_w^*\gamma_w(u,v,|\underline{B}_w|/|\underline{z}'|)\underline{D}_wA_w = -\mu_0 j_w(u,v). \tag{12b}$$

Of the many quantities that are of interest in the design or evaluation of magnets, we want to discuss only the transformation properties of two more quantities. Although both transformation properties are trivial, they are of such practical importance that it is worthwhile to state them.

Since x, y and u, v are related through a conformal transformation, infinitesimal areas $\mathrm{d}a_w$ and $\mathrm{d}a_z$ are related through

$$\mathrm{d}a_z = |\underline{z}'|^2\mathrm{d}a_w. \tag{13}$$

With eq. (12a) we obtain therefore for the total current I_w passing through any given area in the u, v coordinate system:

$$I_w = \int j_w\mathrm{d}a_w = \int j_z|\underline{z}'|^2\mathrm{d}a_w = \int j_z\mathrm{d}a_z = I_z,$$

i.e. total currents passing through conformally mapped areas are identical.

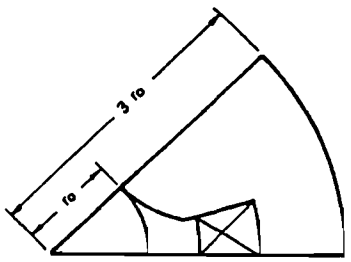


Fig. 1. $\frac{1}{8}$ of quadrupole in original geometry.

For the field energy E stored in any given area per unit length of magnet, we obtain from

$$E_w = \int |\underline{B}_w|^2 \gamma_w \mathrm{d}a_w/(2\mu_0)$$

and eqs. (10), (11) and (13)

$$2\mu_0 E_w = \int |\underline{B}_z|^2 \gamma_z |\underline{z}'|^2 \mathrm{d}a_w = \int |\underline{B}_z|^2 \gamma_z \mathrm{d}a_z = 2\mu_0 E_z,$$

i.e. conformally mapped areas store equal field energy per unit magnet length.

2.4. Magnetostatic computer code modifications for integration of eq. (12)

Comparing eqs. (7) and (12), one notices two differences:

a. The current density j_w appearing in eq. (12b) is related to the current density j_z through eq. (12a). The proper current density j_w can obviously be obtained by modifying the input data according to eq. (12a). Since in most practical magnets, the current density j_z is constant within the boundary of each conductor, it is convenient to add a small subroutine that allows to input $\underline{w}(\underline{z})$ and $|\underline{z}'|^2$, and the boundary and current density j_z for each conductor, and that then prepares the input data to give the correct j_w. That same routine can of course also be used to transform all other boundaries from the x, y coordinates to the u, v coordinates. Since this routine does not interact with the integration routine, it is clearly a simple task to add this routine to the program.

b. The major discrepancy between eqs. (7) and (12b) is that γ_z depends on $|\underline{B}_z|$, whereas γ_w depends on $|\underline{B}_w|/|\underline{z}'|$. It would be an easy task to store $1/|\underline{z}'|$ for each iron mesh zone and then to multiply each flux density value by $1/|\underline{z}'|$ before finding the value of γ_w when integrating eq. (12b). However, since in most integration routines, the value for the flux density in a mesh zone is derived by appropriate numerical procedures from potentials at mesh points surrounding that zone, it will in general be easy to modify the algorithm so that it gives $|\underline{B}_w|/|\underline{z}'|$ instead of $|\underline{B}_w|$. Both this and the above mentioned modifications were incorporated into POISSON[3]) with very little effort and without increasing the storage requirements or the execution time for evaluation of magnets.

Although we used vector potentials for the derivation of eq. (12), the resulting conclusions concerning the necessary modifications of j and γ are, of course, independent of the method that was used to derive them, and are valid no matter what algorithm is used to actually integrate the magnetostatic equations.

3. Consequences of application of conformal transformation to evaluation of iron magnets

3.1. Introductory remarks

When discussing the advantages resulting from using conformal transformations in conjunction with a magnet analysis program, we will talk especially about the analysis program POISSON[3]). Although some comments apply only to POISSON, or a program similar to it, most remarks are valid no matter what analysis program is used. Also, we will talk mostly about evaluation or design of quadrupoles, since their discussion is representative for all higher multipoles.

3.2. Conformal transformation of a quadrupole

To provide a good quadrupole field in a circular aperture, it is desirable to build a magnet with the highest degree of symmetry possible. Fig. 1 shows the schematic outline of $\frac{1}{8}$ of such a magnet, with the 0° and 45° lines being lines of constant scalar and vector potentials respectively. Within the aperture, the field $\underline{B}_z$ can be derived from a complex potential

$$\underline{F}(\underline{z}) = A_z + \mathrm{i}V_z$$

and because of the symmetry of the magnet shown in fig. 1, the power series for $\underline{F}(\underline{z})$ has to have the form

$$\underline{F}(\underline{z}) = \sum_{n=0}^{\infty} a_{2(2n+1)} \underline{z}^{2(2n+1)}. \tag{14a}$$

From this follows for the fields:

$$\underline{B}_z^* = \mathrm{i}\underline{F}'(\underline{z}) = 2\mathrm{i} \sum_{n=0}^{\infty} (2n+1) a_{2(2n+1)} \underline{z}^{4n+1}. \tag{14b}$$

The transformation

$$\underline{w} = \underline{k}\underline{z}^2 \tag{15a}$$

leads to

$$\underline{F}_w(\underline{w}) = \sum_{n=0}^{\infty} a_{2(2n+1)} \cdot (\underline{w}/\underline{k})^{2n+1} \tag{16a}$$

and

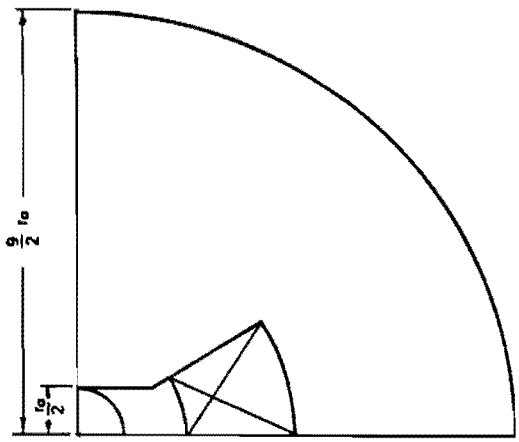


Fig. 2. ¼ of quadrupole in transformed geometry.

$$\underline{B}_w^* = (i/k) \sum_{n=0}^{\infty} (2n+1) \cdot a_{2(2n+1)} (\underline{w}/\underline{k})^{2n}. \quad (16b)$$

When the magnet is a good quadrupole magnet, in the aperture the term proportional to a_2 dominates in eq. (14), and therefore dominates also in eq. (16). But eq. (16) then describes an essentially homogeneous field in the aperture, and this is of course highly desirable for evaluation as well as poleface design of a magnet. Before discussing the resulting advantages in detail, it is convenient to introduce a particular value for the scale factor $\underline{k}$ in eq. (15a).

To get simple relations for saturation considerations, we introduce r_0 as the distance from the center of the original magnet to the iron nearest the center and require that for $|\underline{z}| = r_0$, $|\underline{w}'| = 1$, giving $|\underline{B}_z| = |\underline{B}_w|$ there.

Using for simplicity a real k, we thus get from eq. (15a) for $|\underline{z}| = r_0$: $|\underline{w}'| = 2kr_0 = 1$.

Using this in eq. (15a) we obtain

$$\underline{w} = \rho_0(\underline{z}/r_0)^2; \quad \rho_0 = \tfrac{1}{2}r_0 \quad (15b)$$

and

$$\underline{w}' = (\underline{z}/r_0) = (\underline{w}/\rho_0)^{\frac{1}{2}}. \quad (15c)$$

For a $2n$-pole, one would use similarly:

$$\underline{w} = \rho_0(\underline{z}/r_0)^n; \quad \rho_0 = r_0/n,$$
$$\underline{w}' = (\underline{z}/r_0)^{n-1} = (\underline{w}/\rho_0)^{1-1/n}. \quad (15d)$$

Applying the transformation described in eq. (15b) to the magnet shown in fig. 1 leads to the configuration shown in fig. 2, which is drawn to the same scale as fig. 1.

When evaluating a quadrupole magnet, the quantity of interest is usually the gradient of the field. From eqs. (10) and (15c) we obtain

$$\underline{B}_z^* = \underline{B}_w^* \underline{z}/r_0 = \underline{B}_w^* (\underline{w}/\rho_0)^{\frac{1}{2}}, \quad (17a)$$
$$\underline{B}_w^* = r_0 \underline{B}_z^*/\underline{z}. \quad (17b)$$

From eq. (17b) follows the $\underline{B}_w^*$ is directly a measure for the gradient in the real magnet even if the quadrupole is not perfect. To obtain the local gradient in the aperture region, we can differentiate eq. (17a) with respect to $\underline{z}$†:

$$\mathrm{d}\underline{B}_z^*/\mathrm{d}\underline{z} = \{\underline{B}_w^* + 2\underline{w} \cdot (\mathrm{d}\underline{B}_w^*/\mathrm{d}\underline{w})\}/r_0. \quad (17c)$$

It is clear that if the magnet is a good quadrupole magnet, the derivative on the right side of eq. (17c) contributes very little to the local field gradient inside the good field aperture.

3.3. Advantages of transformed magnets

The most obvious reason for gaining advantages through conformally transforming a magnet is, of course, the same reason why conformal mapping has been used advantageously for a long time in many fields: The simplifications of the geometry make many aspects of a problem so transparent that they become outright trivial, whereas they are often quite obscure in the original geometry. For instance, it is qualitatively much easier to see what kind of an effect a modification of the magnet near the useful field aperture has on the field of an essentially homogeneous-field magnet than it is to see what the effect of the equivalent modification is on the gradient of a quadrupole magnet. Or, to take a specific drastic case: if a sextupole magnet has a circular useful field aperture r_0, and a significant modification of the magnet is made at the distance $3r_0$ from the center, it is not entirely obvious what its effect is on the second derivatives of the field. However, if the circular aperture of the transformed magnet is ρ_0, the modification in this geometry is then at the distance $27 \cdot \rho_0$ and it is obvious that this will have very little effect on the homogeneity of the field inside ρ_0, allowing the conclusion that the modification will also have very little effect on the second derivatives of the field in the original magnet. From these considerations follows that the task of designing a multi-

† This is, of course, correct only where $\underline{B}^*_z$ can be derived from a complex potential, i.e. where γ = const. and $j = 0$.

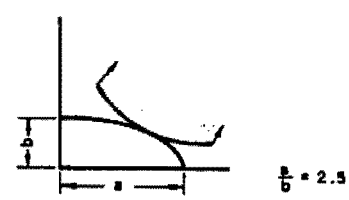


Fig. 3. ¼ of aperture ellipse and poleface in original geometry.

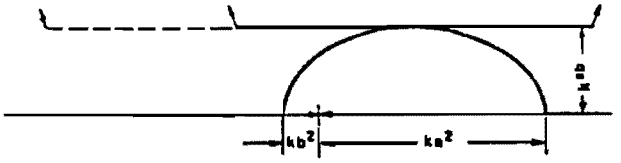


Fig. 4. ¼ of aperture ellipse and poleface in transformed geometry.

pole magnet with a reasonably pure multipole field in the useful field aperture becomes greatly simplified through a conformal transformation, particularly since one can apply many of the fairly simple and well understood rules that one has for the design of homogeneous field magnets.

To demonstrate this in more detail we consider the design of a quadrupole magnet that is required to have a good quadrupole field within an elliptical aperture. Fig. 3 shows one quarter of an aperture ellipse with a ratio of major to minor axis of 2.5, with a rough outline of one quarter of the magnet poleface also indicated. To see how far the poleface has to be carefully designed, we apply the transformation $\underline{w} = k\underline{z}^2$, mapping the ¼-ellipse of fig. 3 into the ¼-ellipse in fig. 4. Since we know that in order to obtain a homogeneous dipole field, the poleface should extend at least one quarter of the magnet gap beyond the ends of the aperture region, we also indicate the required poleface width. It is also shown how far the poleface would have to go on the left side in order to get a quadrupole that is symmetrical with respect to the 45°-line (in the original geometry), with the resulting better field quality because of the higher degree of symmetry. Since conformally mapped areas store the same energy per unit magnet length, it is directly evident how much one pays in terms of stored energy for the magnet with the higher symmetry. After fixing the ends of the polefaces in the described manner, one would then transform these endpoints into the original geometry and design the rest of the magnet structure (coils, yokes, etc.) in the original geometry. Then, as the last step, after transforming the whole magnet and generating a mesh in the new geometry, one would evaluate the magnet in the transformed geometry and optimize the poleface to give a highly homogeneous field in the transformed geometry, leading to a high quality quadrupole field in the original geometry.

There are, of course, some basic differences between true homogeneous field magnets and homogeneous field magnets that are obtained through conformal transformation of a multipole magnet. In most magnets the coils have a uniform current density and the air-coil and most air-iron interfaces are straight lines (the magnet shown in fig. 1 is an exception in this respect). This is of course no longer true after a multipole magnet has been transformed. Although this has generally very little effect on the design of the aperture region, it means for instance that one can not obtain a practical multipole magnet by transformation of a window frame magnet.

A more significant difference arises when one considers saturation effects. When one designs a homogeneous field magnet and other considerations, such as stored field energy, do not preclude such a conservative design, one can get very good field homogeneity over a wide field range by extending the flat poleface significantly beyond the aperture limits. Doing the same in the case of a transformed multipole would lead to a badly saturating magnet because, according to eqs. (12b) and (15d), the quantity determining the saturation in iron is

$$|\underline{B}_w| \cdot |\underline{w}/\rho_0|^{1-1/n}.$$

With $|\underline{B}_w|$ in the poleface region essentially constant, the factor

$$|\underline{w}/\rho_0|^{1-1/n}$$

will lead to stronger saturation effects the more the poleface is extended beyond the aperture limit. Numerically, this effect can be quite significant: if the total width of the symmetrical poleface of a transformed quadrupole is $3\rho_0$ in one case, and $4\rho_0$ in another, the values for $|\underline{w}/\rho_0|^{\frac{1}{2}}$ at the ends of the poleface are 1.344 and $1.5 = 1.344 \times 1.11$. This points out that in order to design a multipole magnet with a good field distribution at low as well as at high fields, it is exceedingly important to be able to achieve good field distributions with as little iron beyond the aperture limits as possible. For this reason, a performance optimization procedure that allows one to optimize the field distribution simultaneously at low and high fields[4]) is even more important for the design of multipole magnets than it is for the design of dipole magnets.

While the presence of the above mentioned saturation effects is obvious without application of a conformal transformation, their qualitative and quantitative discussion and evaluation is considerably easier in the transformed geometry.

Again with respect to this subject, one gains a better understanding through considering the transformed magnet. The resulting simplifications of the design process will of course in many cases lead to improved design and performance of magnets.

While the advantages discussed so far are to a large degree of a qualitative nature, evaluation of multipole magnets in the transformed geometry can also lead to

a very significant increase in accuracy. Magnet evaluation codes usually compute potentials at a large number of discrete mesh points that cover the geometry of the magnet. In the algorithm for the calculation of the potentials, the behaviour of the potential in the region around mesh points is generally approximated by polynomials in the coordinates. These polynomials are, to cite two examples, of first order in POISSON, and second order in SYBIL[5]). This basic difference is due to the meshes employed in these two types of programs, SIBYL using a uniform rectangular mesh and POISSON a variable triangular mesh. This gives POISSON the advantage of being more flexible and therefore having virtually no restrictions on the boundaries of the problem and between different materials, at the expense of being less accurate. While these inaccuracies are of very little significance far away from the useful field aperture, they can be of importance in the aperture if the field there is highly inhomogeneous. By evaluating a multipole magnet in the transformed geometry, the aperture field will be very homogeneous for a well designed magnet. Consequently, the potentials are nearly exactly linear functions of the coordinates, thus practically eliminating this source of error. Furthermore, the local field gradients in the original geometry are essentially given by the field in the transformed geometry. This means that in order to obtain the local field gradient in the aperture of a quadrupole, one has to calculate second derivatives of relatively inaccurate potentials if the evaluation is done in the original geometry, whereas one has to take essentially only first derivatives of very accurate potentials if the evaluation is done in the transformed geometry. The cascading of these two main accuracy-improving properties leads to a very significant improvement of overall accuracy: Evaluating a magnet that has an analytical quadrupole field distribution with POISSON gave the gradients in the aperture region with about 1% errors when the evaluation was done in the original geometry, whereas the error was only 0.01% when evaluated in the transformed geometry. While 0.01% accuracy is better than normally needed, 1% errors are more than tolerable in many cases. It is clear that the accuracy improvement is even more urgently needed (and obtainable with this procedure) for higher multipoles, where even an intrinsically more accurate program like SIBYL could not be expected to be quite as accurate as one would need under some circumstances. A minor advantage results when a multipole magnet is to be evaluated with an irregular variable mesh program like POISSON and the evaluation is done in the transformed geometry: Because of the curvature of the poleface in the original geometry, it is very difficult to generate a good mesh point distribution, while it is very easy to generate a practically perfect mesh in the aperture region of the transformed magnet.

Finally, it might be worthwhile to remark that it is possible to check internal consistency and accuracy of a program by computing potentials and fields of a magnet in the original and a conformally transformed geometry, and then comparing the results.

4. Limitations and drawbacks

Although it is possible to evaluate a transformed magnet geometry that covers more than one leaf of a Riemann surface, this is clearly neither desirable nor practical. Therefore a magnet geometry should be sufficiently symmetrical so that the transformed magnet covers only 360° or less of a plane. While most multipole magnets satisfy this condition, one has to realize that for all practical purposes this makes it impossible to evaluate in the transformed geometry the effects of slight assembly-asymmetries of a basically symmetric magnet.

It is clear that by transforming a $2n$-pole with $\underline{w} \sim \underline{z}^n$, the ratio of the aperture area to total magnet area is much smaller in the transformed geometry than it is in the original geometry, leading to a reduced mesh point density in the aperture of the transformed magnet. When using a variable mesh code, this can be partly corrected by adjusting the mesh spacing accordingly; with a fixed mesh code, one gains a small advantage because the fraction of the magnet that has to be evaluated is generally larger in the original geometry than it is in the transformed geometry. (For instance, one has to evaluate $\frac{1}{4}$ of a symmetrical sextupole in the original geometry, but only $\frac{1}{12}$ in the transformed geometry.) Depending on the details of the magnet under consideration, sometimes neither one of these gains is enough to compensate sufficiently the reduced mesh point density in the aperture region of the transformed magnet. We want to discuss briefly two methods that can be used to improve the mesh point density in the aperture.

Instead of using the transformation that produces exactly the desired mapping, one can use one that gives the desired mapping in the aperture region in very good approximation, and compresses the transformed magnet far away from the aperture.

For instance, instead of using the really desired transformation

$$\underline{w} = (r_0/n)(\underline{z}/r_0)^n$$

for a $2n$-pole, one can use the transformation

$$\underline{w} = \{r_0/(n\varepsilon)\} \ln \{1 + \varepsilon(\underline{z}/r_0)^n\} \text{ with } \varepsilon \ll 1.$$

In the aperture region ($|\underline{z}|/r_0 \leqq 1$) the latter transformation gives approximately the same mapping as the former, whereas for $\varepsilon(|\underline{z}|/r_0)^n \geqq 1$ they differ markedly, making the overall size of the second transformed magnet relative to its aperture smaller than the first. The slight deviation from the transformation

$$\underline{w} = (r_0/n)(\underline{z}/r_0)^n$$

should not decrease the evaluation accuracy in the aperture if ε is not too large; also all the gains of qualitative nature described at the beginning of this section are preserved. Since the exact form of the used transformation is known, it is of course very easy to take the difference between it and the transformation $\underline{w} \sim \underline{z}^n$ quantitatively into account. In magnets with extreme dimensions one could even think of using the transformation:

$$\underline{w} = \{r_0/(\varepsilon n)\} \ln [1 + \ln \{1 + \varepsilon(\underline{z}/r_0)^n\}].$$

A somewhat simpler procedure would be to evaluate the magnet in two steps: First in the original geometry, giving the overall potential distribution and all the gross-saturation characteristics, but very poor accuracy in the aperture region. In the second step one evaluates, in the "ideally" transformed geometry, only a part of the magnet, extending, in the transformed geometry, from the center to about 5–20 times the aperture dimension. Depending upon whether or not this region contains coils, one can then evaluate this partial magnet with boundaries parallel or perpendicular to field lines calculated in the first step, or with boundary values of the potentials obtained in the first step. Since the boundaries are far removed from the aperture region, the accuracy of the evaluation in the aperture region will be practically independent of the choice of the boundaries or the boundary values. If the saturation behaviour is of no interest, it will in most cases not even be necessary to make the first evaluation since one can guess in general with sufficient accuracy what one has to do at the boundaries.

Referring to eqs. (10) and (12a), it is evident that $\underline{z}' \neq 0$; $\underline{w}' \neq 0$ has to hold in all regions containing iron and $\underline{w}' \neq 0$ has to hold in coil regions of magnets. In the rare cases where one would like to use a transformation that violates these conditions, it is usually possible to modify the ideally wanted transformation such that it still gives essentially the desired mapping in the area of interest, but avoids the violation of the above mentioned conditions, just as was suggested to compress the outside portions of a mapped magnet.

References

[1]) CERN Report, PS/MM 35 (April, 1958).

[2]) W. Hardt, Electrotechn. Z. A84 (1963) 892.

[3]) A. M. Winslow, J. Comp. Phys. 1 (1967) 149 (this particular program was written by J. R. Spoerl).

[4]) K. Halbach, Proc. Intern. Conf. *Magnet technology* (1967); UCRL-17436 (1967).

[5]) Conceived and developed by R. Christian, written by R. Christian and J. H. Dorst, UCRL-16389 (1965).

NUCLEAR INSTRUMENTS AND METHODS 66 (1968) 154–156; © NORTH-HOLLAND PUBLISHING CO.

CALCULATION OF THE STRAY FIELD OF MAGNETS WITH POISSON*

K. HALBACH

Lawrence Radiation Laboratory, University of California, Berkeley, California, U.S.A.

Received 9 September 1968

It is shown that application of conformal mapping to two dimensional magnets, containing iron with nonlinear $B(H)$ characteristics, allows calculation of magnetic fields in all 2D space. For some magnets of finite length, this information can be used to get a good approximation for the stray fields in all of 3D space. It is furthermore shown that only minor modifications to POISSON are necessary to allow application of the same techniques to magnets with axial symmetry, leading in this case to solutions that genuinely and accurately describe the fields in all of 3D space.

1. Introduction

It is sometimes necessary to have information about the stray fields produced by two dimensional magnets. It is clear that the following consideration, being purely two dimensional, is valid at most up to a distance of the order of the physical length of the magnet. We will later describe a procedure that can give approximate information about three dimensional stray fields and will finally discuss application of the basic procedure to axially symmetric magnets. To simplify the description of the procedure, we discuss its application to a specific magnet that is typical for the kind of problem that arises in practice (and also leads to simple figures that are easily drawn). Although of general validity, the description of the method is tailored to the use of the magnetostatic analysis program POISSON[1]).

2. Stray fields produced by a window frame magnet

Fig. 1 represents the cross section of ¼ of a window frame magnet. The field lines are perpendicular to the midplane 0-8, and the symmetry plane 0-6 has a constant vector potential. For calculation of the field with a digital computer, one obviously has to limit artificially the grid that is used for the description of the problem. Even when saturation effects are of importance, it is a reasonably good approximation to limit the grid along the line 6-7-8 and put that line onto the same vector potential as the line 0-6. Whether one limits the grid in this way, or limits it farther outside, with air between 6-7-8 and the grid limitation, is immaterial for the method used to compute the stray fields.

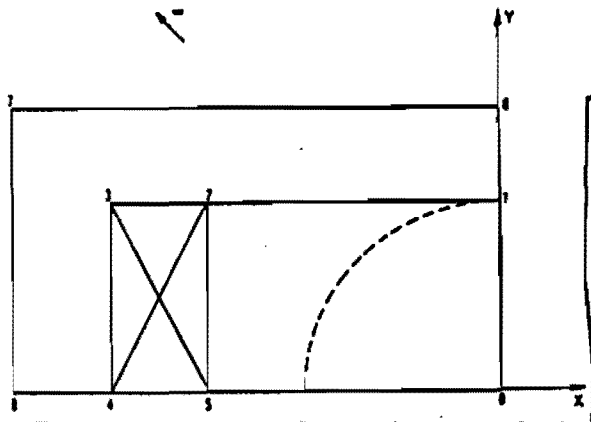


Fig. 1. Original geometry of magnet (complex z-plane).

When one wants to calculate the stray field at some point outside the magnet, the grid does not only have to include that point, but should go significantly beyond it in order to avoid falsification of the stray field by the artificial grid limitation. This leads to an impractically large total number of mesh points, since the magnet itself should still contain a reasonable number of grid points in order to describe the stray field-producing saturation of the iron adequately. The large number of grid points and the errors resulting from the artificial grid limitation are avoided with the following procedure.

One first solves the magnetostatic problem, without regard for stray fields, in the configuration shown in fig. 1, with the artificial grid limitation along line 6-7-8. One then solves the same magnet again, but in the geometry obtained by applying the conformal transformation

$$w = -R^2/z, \qquad (w = u + iv;\ z = x + iy), \qquad (1)$$

to the original magnet. R is a suitably chosen scaling length, and fig. 2 represents the transformed magnet, drawn to the same scale as the magnet in fig. 1, with R equal to the distance 0-5.

The minor program modifications necessary to analyse a magnet with nonlinear iron in a conformally transformed geometry have been described elsewhere[2]) and are incorporated into POISSON.

* This work was done under the auspices of the U.S. Atomic Energy Commission.

see also last page.

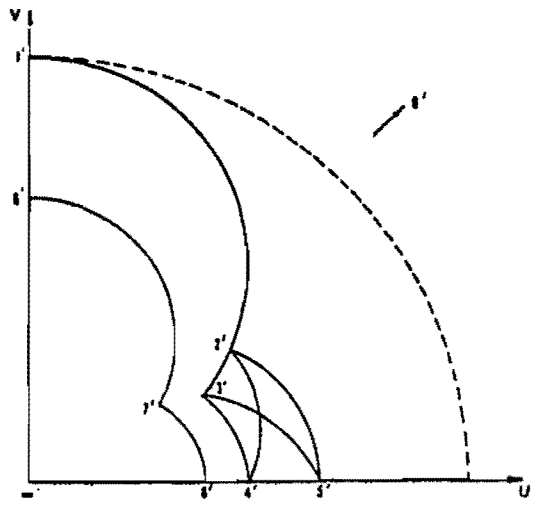


Fig. 2. Conformally mapped geometry of magnet (complex w-plane).

To solve the magnetostatic problem in the transformed geometry, we limit the grid in the w-plane along the map of a suitably chosen contour *inside* the magnet in the original geometry, for instance line 1-3-4, or alternatively the dashed circle. We obtain the vector potentials at the grid points of that mapped contour from the analysis in the original geometry, and solve the resulting boundary value problem, or boundary value problem with currents, in the transformed geometry. The field components B_u, B_v inside the contour ∞'-6'-7'-8'-∞' are obtained from the vector potentials by standard numerical differentation, and the field components B_x, B_y in the original geometry are obtained by application of eq. (10) of [2]) and yield with eq. (1):

$$(B_x - iB_y) = (B_u - iB_v)(dw/dz) = \\ = (B_u - iB_v)(R/z)^2 = (B_u - iB_v)(w/R)^2. \quad (2)$$

It can also be practical to calculate from the vector potentials the multipole coefficients a_n of the complex potential describing the fields in the w-plane:

$$F_w(w) = \sum_n^{\infty} a_n w^n. \quad (3a)$$

This gives for the Laurent-expansion of the stray field potential:

$$F_z(z) = \sum_n^{\infty} a_n(-R^2/z)^n. \quad (3b)$$

The only one step described above that is not routinely performed by POISSON is the transfer of the vector potentials from the contour 1-3-4 to the mapped contour 1'-3'-4' in the w-plane. The simplest way to accomplish this is to map the grid points on the outer contour in the w-plane into the z-plane and calculate the potentials there by interpolation of the vector potential field. Linear interpolation should in general be sufficient, since minute details of the vector potential distribution along the outer contour in the w-plane should have only a small effect on the stray fields. An alternate method to solve for the stray fields would be to compute the scalar potentials along the contour 6-7-8 and use its map as the outer problem boundary in the w-plane. The first method is preferable, since scalar potentials are usually not computed by POISSON and the stray fields would be more sensitive to errors in the scalar potentials along 6'-7'-8' then they are to errors in the vector potentials along 1'-3'-4'.

It is clear that the computation of the stray fields in the w-plane allows the presence of ferromagnetic bodies in the stray field region as long as they satisfy the conditions implied by the two dimensional approximation.

When one is dealing with symmetrical multipoles ($2n$-poles), it has been shown[2]), that it is advantageous to analyse their fields in the geometry obtained by the conformal transformation $w = k^{n-1}z^n$. Similarly, the stray fields of such a magnet should be computed in the geometry obtained by transforming the original geometry with $w = -R^{n+1}/z^n$, and the stray fields in the original geometry are obtained from the fields in the transformed geometry with appropriately modified equivalents to eq. (2).

When one is evaluating magnets that saturate badly, or magnets with an open iron core (for instance C-magnets), it is not always obvious how much the solution in the magnet is influenced by the artificial grid limitation. To obtain a better solution, one can compute the stray fields as described above, then transfer the vector potentials obtained along the map of the grid-limiting-contour of the original magnet (6'-7'-8') to that contour (6-7-8) in the original geometry, and solve the problem again in the original geometry with these new values at that boundary. With this iterative process, which in general will not have to be repeated, one clearly obtains a very accurate all 2D-space solution for the magnet. There are obviously other configurations where these all 2D-space solutions might be useful. One might, for example, want to know how the field in an iron-free magnet is influenced by the presence of iron at a distance outside the magnet.

It is worthwhile to point out that it is possible to obtain reasonable approximations for the three dimensional stray fields at virtually any distance produced by magnets that have small end effects in the sense that

they do not contribute significantly to the stray fields. One could derive the three dimensional stray fields from a vector potential that is obtained by superposition of finite length filamentary multipoles with strengths giving the "near field" multipole strengths described by eq. (3b).

3. Extension to magnets with axial symmetry

Magnetostatic fields with axial symmetry can be derived from a vector potential that has only an azimuthal component which is, of course, independent of the azimuth. If we introduce the axial and radial coordinates respectively as the x and y coordinates of a Cartesian coordinate system and furthermore introduce a pseudo vector potential V with only a component in the $x \times y$ direction equal to y times the vector potential, then the magnetostatic equations for the axially symmetric problem can be represented by:

$$B = (y)^{-1}\nabla_c \times V, \tag{4a}$$

$$H = \gamma(|B|,x,y)\cdot B/\mu_0, \qquad (\gamma = 1/\mu_{rel.}), \tag{4b}$$

$$\nabla_c \times [(\gamma/y)\nabla_c \times V)] = \mu_0 j. \tag{4c}$$

In these equations, j has only a component in the $x \times y$ direction, equal to the current-density in the original problem, and ∇_c is the Cartesian form of the del operator.

The only difference between eqs. (4) and the equations describing a genuine two dimensional problem in Cartesian coordinates is the extra factor $1/y$, and the integration routine is easily modified to take this into account. POISSON, just as its predecessor TRIM (as well as other programs), has this modification incorporated for the solution of problems with axial symmetry. It is clear from eqs. (4) and [2]) that in order to solve eqs. (4) in a conformally mapped geometry, one needs to incorporate into POISSON the same modifications that are needed to solve genuine two dimensional problems in a transformed geometry, as well as the modification necessary to take the factor $1/y$ into account. Although this has not been done yet, it is clearly a simple matter to do so. To find the stray fields for an axially symmetric problem, the same transformation can be used that was employed in the 2D case, and the rest of the procedure is also identical, with only two modifications: the power expansion, eq. (3), is not applicable, and the field components in the x-$y(z$-$r)$ coordinate system are obtained from

$$B_x - iB_y = \{(\partial V/\partial v) + i(\partial V/\partial u)\}(\mathrm{d}w/\mathrm{d}z)/y.$$

In contrast to the two dimensional case, application of this procedure to an axially symmetric magnetostatic problem gives a solution that genuinely and accurately covers all space.

References

[1]) POISSON is an improved version of TRIM [originally written by A. M. Winslow, J. Computer Phys. 1 (1967) 149] and was developed by J. R. Spoerl, R. F. Holsinger and K. Halbach. POISSON uses, like TRIM, the vector potential in an irregular triangular mess.

[2]) K. Halbach, Nucl. Instr. and Meth. 64 (1968) 278.

A more convenient transformation is $W = \dfrac{Z}{1 + Z/ia}$. It maps the $y \geq 0$ 1/2 plane onto a circular disc of radius $a/2$, with its center at $W = ia/2$. With this transformation, only one computer run is needed

ARGONNE NATIONAL LABORATORY
9700 South Cass Avenue, Argonne, Illinois 60439

September 12, 1985

Mary Menzel
LANL AT-6 MS-829
P. O. BOX 1663
Los Alamos, NM 87545

Dear Mary,

I wish to thank you for your help in tracking down the problems I encountered using POISSON and, in particular, the conformal mapping feature. My understanding of it is as follows:

LATTICE

1. The user must transform the geometry from the x-y plane to the u-v plane.

2. CON(37) = MAP, CON (123) = TNEGC, CON (124) = TPOSC, AND CON (125) = RZERO must be specified so that LATTICE can map the currents to the u-v plane and correct them slightly so the total current is the same in both the x-y and u-v planes.

3. Cannot use line regions with specified potential.

POISSON

1. The field and gradient are printed out at the air points for both the x-y and u-v plane.

2. The field at the iron points is printed out for the x-y plane only.

3. The stored energy is the same in both planes.

4. The flux lines plotted are lines of constant vector potential in the u-v plane. Caution, these do not indicate the flux density in the x-y plane.

I am enclosing a copy of a Light Source Note, LS-32, describing the testing I did of the Harmonic Analysis feature of POISSON. Please give John Warren and Martyn Foss a copy and express my thanks to them.

Sincerely,

Bob Lari

Robert J. Lari

RJL:ehr

Enclosure

Harmonic Analysis Errors in Calculating Dipole, Quadrupole, and Sextupole Magnets using POISSON

Robert J. Lari
September 10, 1985

Introduction

The computer program POISSON was used to calculate the dipole, quadrupole, and sextupole magnets of the 6 GeV electron storage ring. A trinagular mesh must first be generated by LATTICE. The triangle size is varied over the "universe" at the discretion of the user. This note describes a series of test calculations that were made to help the user decide on the size of the mesh to reduce the harmonic field calculation errors. A conformal transformation of a multipole magnet into a dipole reduces these errors.

Dipole Magnet Calculations

A triangular mesh used to calculate a "perfect" dipole magnet is shown in Fig. 1. Both the physical (x-y) and logical (K-L) mesh coordinates are shown. The lower boundary of this "universe" is a flux normal boundary and can be considered the mid-plane of the magnet. The top boundary is also a flux normal boundary and can be considered as an infinite permeable pole tip. The left boundary is a flux line of vector potential 0.0 G-cm and the right boundary is also a flux line of vector potential 140,000 G-cm. Since the distance between these boundaries is 14 cm, the flux density in the universe will be uniformly 10000 gauss. A mesh 8 units high by 29 units wide was used. The total number of mesh points is (8+2) (29+2) or 310 mesh points. This includes the four phantom mesh lines surrounding that shown.

The harmonics are calculated by integrating the vector potential on a circular arc and doing a Fourier analysis of it. Hence, the program requests an integration radius, RINT, a starting angle, ANGLZ, a change in angle, ANGLE, and a normalization radius, RNORM. The number of terms to calculate, NTERM, and the number of equidistant points on the arc of circle, NPTC must also be specified. The integration radius and normalization radius were both 3.0 and 19 points were used on the circular arc from 0 to 180 degrees. These

19 mesh points are shown circled in Fig. 1. Table I gives the results of the calculation. All harmonics have units of gauss. The maximum error for a mesh this size is less than .05 gauss in 10000 gauss at a radius of 3 cm!

A smaller mesh size might be possible, but with this size, 0.5 by 0.5 cm, the mesh is not too distorted at RINT when 0.17 cm by 1.5 cm shims are attached to the pole tip at the sides. Equal weight triangles were used.

Quadrupole Magnet Calculations

It can be shown [1] that the pole shape for a perfect p-pole magnet satisfies equation (1).

$$r^{p/2} \sin (p/2)\theta = R_B^{\,p/2} \tag{1}$$

Likewise the coil shape satisfies equation (2).

$$r^{p/2} \cos (p/2)\theta = R_c^{\,p/2} \tag{2}$$

R_B is the distance to the pole and R_c the distance to the coil. In rectangular coordinates for a quadrupole magnet, $p = 4$, these become:

$$xy = R_B^2/2 \text{ (pole shape)} \tag{3}$$

and

$$x^2 - y^2 = R_c^2 \text{ (coil shape)} \tag{4}$$

A perfect quadrupole is shown in Fig. 2. As in the case of the dipole, the lower and upper boundaries are flux normal boundaries. The left side is a flux line at 45 degrees and the right side is a flux line of vector potential A_1 where

$$A_1 = \int_o^{R_c} B_y dx = B' \frac{R_c^2}{2} = (1000) \frac{G}{cm} \frac{(6.945)^2}{2} cm^2 = 24116.51 \text{ G-cm.}$$

It is best to distribute the triangles uniformly along the x axis and the 45 degree line, since the field varies linearly. This makes the change of the flux density the same across each triangle and makes the errors equal. POISSON assumes that the vector potential varies linearly across each triangle. This assumption conflicts with the quadrupole field which varies linearly with radius. This effect is illustrated in Fig. 3.

The distribution of mesh points along the pole tip can be found by solving equations 3 and 4 simultaneously for a fixed R_B and for the 15 values of R_c which are the x coordinates of the mesh points on the x axis. Similarly, the distribution of mesh points along the coil can be found by solving equations 3 and 4 simultaneously for a fixed R_c and for the 8 values of R_B along the 45 degree line. This method of distribution has been used to calculate the harmonics for four different mesh sizes. The results are given in Table II. A flux plot is shown in Fig. 4.

Using the same number of mesh points, 170, as were used in the dipole case results in field errors of 15 gauss in 3000 or 0.5 percent. Doubling the mesh in each direction results in 527 points and reduces the field errors to 3.2 gauss or 0.1 percent. Again, doubling the mesh for a total of 1829 points gives 1.9 gauss error or 0.06 percent. With 6785 mesh points, the error is 0.3 gauss or 0.01 percent. These results clearly demonstrate the conflict between the basic assumption of field uniformity in POISSON and the linear field of a quadrupole magnet. A method to circumvent this problem is described in the last section.

Sextupole Magnet Calculations

In rectangular coordinates for a sextupole magnet, p = 6, equations 1 and 2 become:

$$3x^2y - y^3 = R_B^3 \text{ (pole shape)} \tag{5}$$

$$x^3 - 3y^2x = R_c^3 \text{ (coil shape)} \tag{6}$$

A perfect sextupole is shown in Fig. 5. The lower and upper boundaries are flux normal boundaries. The left side is a flux line at 30 degrees of vector potential zero. The right side is a flux line of vector potential A_2 where

$$A_2 = \int_o^{R_c} B_y dx = B'' \frac{R_c^3}{3} = \frac{(100)}{3} (7.2569)^3 = 12738.9 \text{ G-cm.}$$

To distribute the mesh points along the x axis so that the change in field between successive points is the same requires that

$$x_N = (N/NPTS)^{1/2} (R_c)$$

where N is the nth point and NPTS is the total number of points along the x axis. A similar distribution can be made along the 30 degree line. Using these x_N values as R_c in equation 2, the two equations, 1 and 2, can be solved simultaneously to find the nth point on the pole tip. The points on the right boundary can be found in a similar way using the R_N as R_B.

The results of the calculations for two mesh sizes is shown in Table III. Using 432 mesh points results in errors of 1.5 gauss out of 3025 gauss or 0.05 percent. With 1364 mesh points, the error is 0.6 gauss or 0.02 percent. A flux plot is shown in Fig. 6.

Conformal Transformation

It has been shown[2] that higher pole magnets can be transformed into a dipole magnet by the transformation:

$$W = \frac{Z^{p/2}}{(\frac{p}{2}) R_o^{(p/2)-1}} \tag{7}$$

where

$W = u + iv$

$Z = x + iy$

p = number of poles

R_o = the magnet bore radius.

for p = 4, a quadrupole magnet, these become

$$u = \frac{x^2 - y^2}{2R_o}$$
$$v = \frac{xy}{R_o} \qquad (8)$$

The pole tip and coil shown in Fig. 7a of a quadrupole magnet is transformed by equations 8 into the pole tip and coil of a dipole magnet of Fig. 7b. The program LATTICE transforms the current from the x-y plane into the u-v plane. The user must first transform the x-y geometry into the u-v geometry and use this as input to LATTICE. The total current is the same in both planes[2].

POISSON transforms the permeability and prints out the fields, gradients, etc. in both the x-y and u-v planes for the air points. The fields in the steel are printed out as they would be in the x-y plane. The stored energy is the same in both planes. A discussion of the advantages, limitations and drawbacks is given in reference 2 and will not be repeated here.

References

1. Robert J. Lari and Gerald J. Bellendir, "Calculation of the Harmonic Content of Asymmetrical Sextupole and Octupole Magnets," Particle Accelerator Division Internal Report GJB/RJL-3 June 1, 1967.

2. K. Halbach, "Application of Conformal Mapping to Evaluation and Design of Magnets Containing Iron with Nonlinear B(H) Characteristics," Instruments and Methods, 64 (1968) pp 278-284.

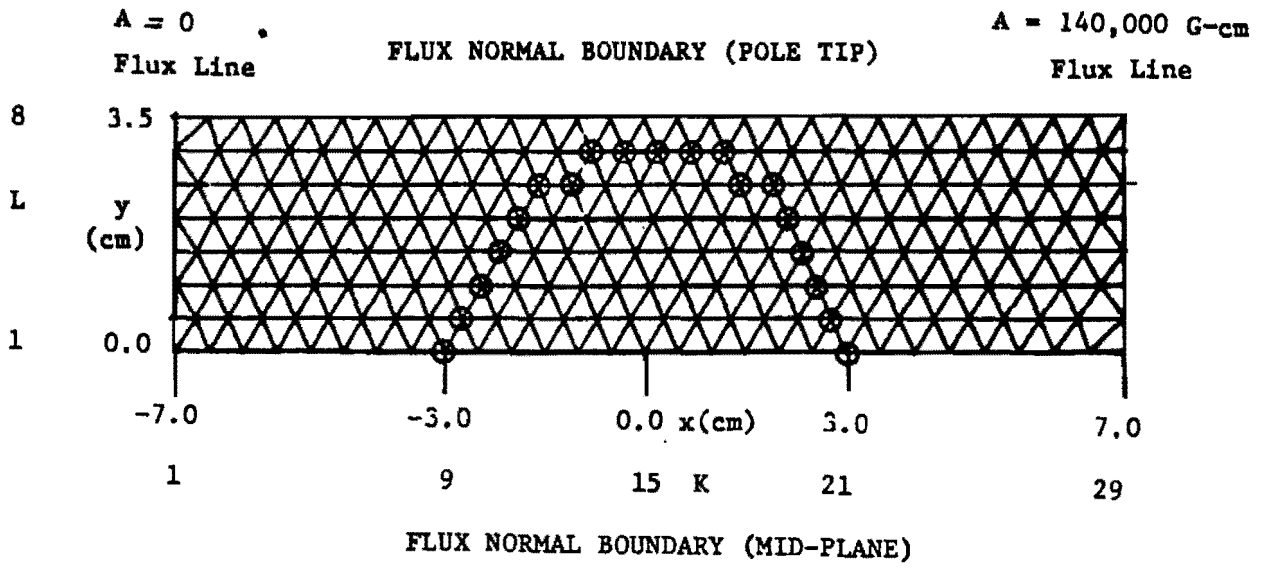
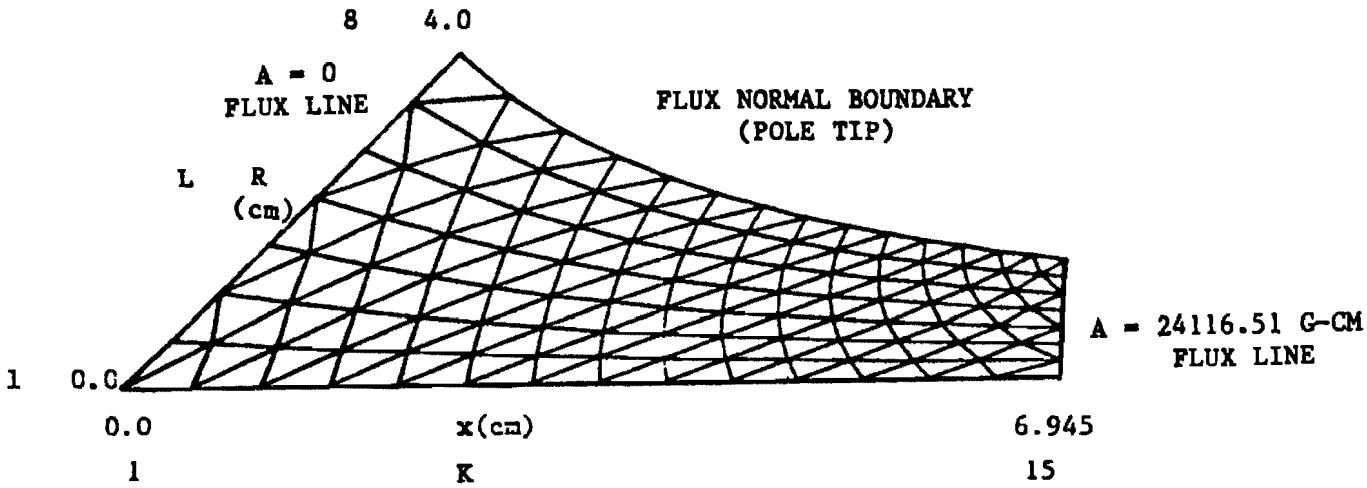


FIGURE 1. PERFECT DIPOLE MAGNET

n	B_n (gauss)
1	10000.000
2	.002
3	- .032
4	.034
5	- .027
6	.042
7	- .047

TABLE I



FLUX NORMAL BOUNDARY (MID-PLANE)

FIGURE 2. PERFECT QUADRUPOLE MAGNET

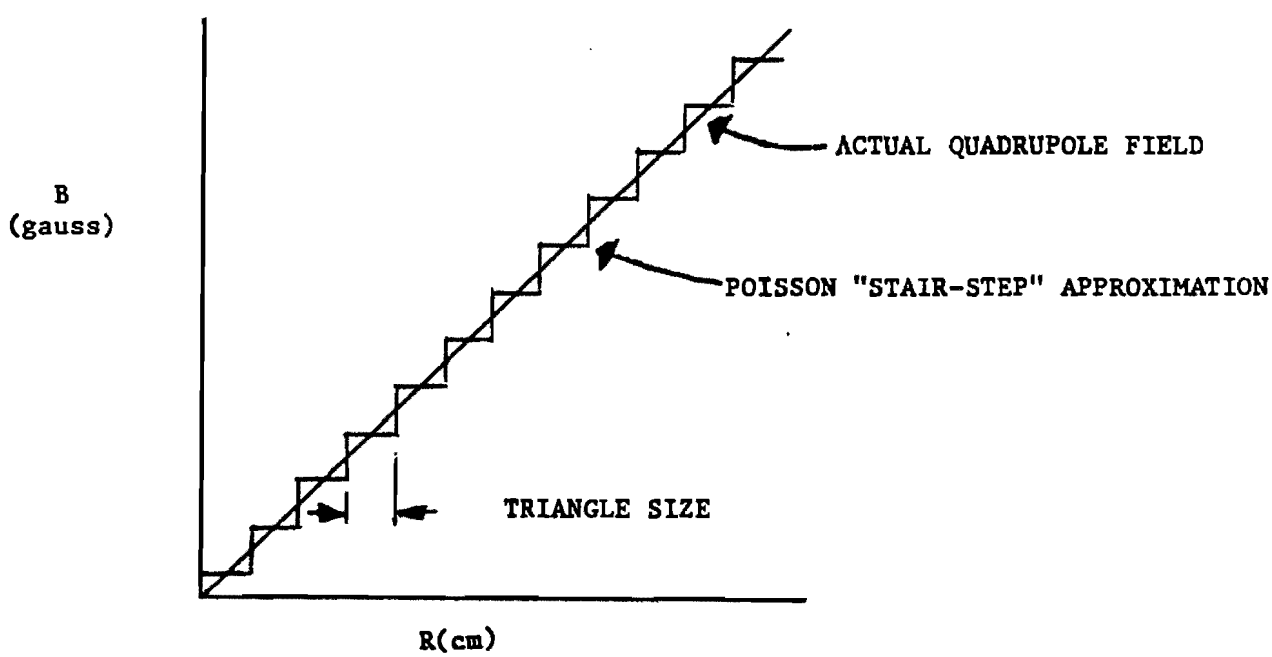


FIGURE 3. POISSON APPROXIMATION TO A QUADRUPOLE FIELD

n	B_n (gauss) (at RNORM = 3.0 cm)			
2	2985.7	2996.8	2998.1	2999.7
6	14.54	2.45	1.20	0.12
10	−6.00	-0.96	-0.48	-0.21
14	3.67	0.35	-0.03	-0.08
18	-2.29	-0.32	0.05	-0.12
22	0.47	0.15	-0.02	-0.27
26	0.36	0.01	0.02	-0.04
NO. OF MESH PTS.	170	527	1829	6785

TABLE II. A PERFECT QUADRUPOLE MAGNET HARMONIC CALCULATIONS

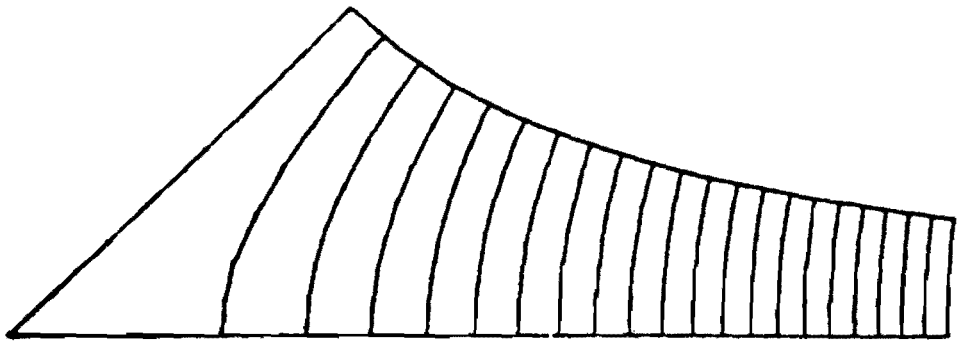


FIGURE 4. PERFECT QUADRUPOLE FLUX LINES

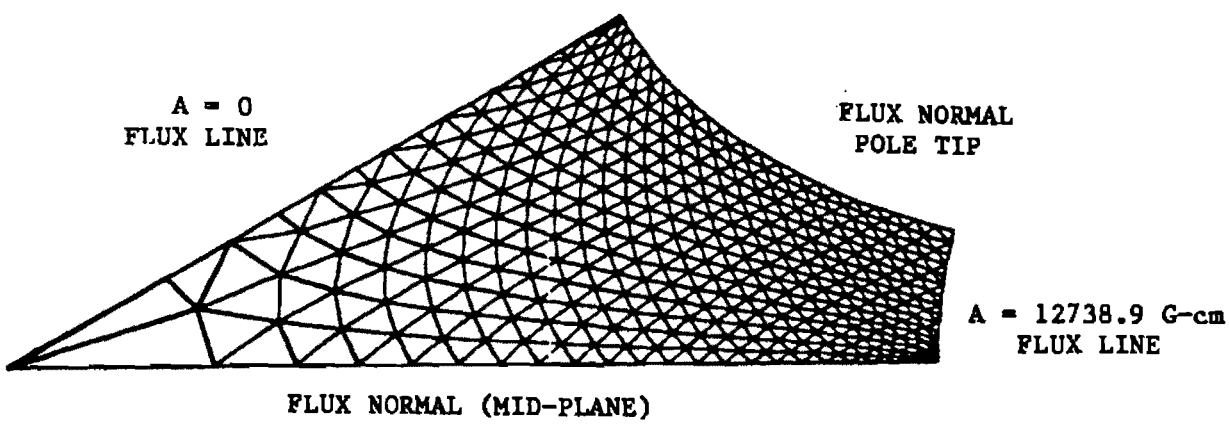


FIGURE 5. PERFECT SEXTUPOLE MAGNET

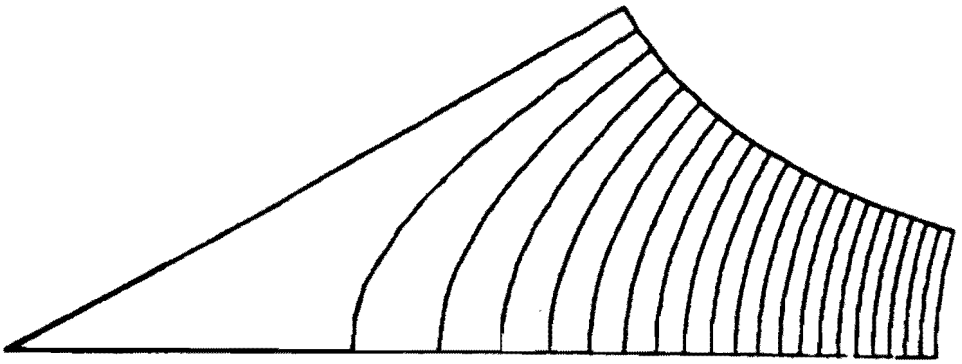


FIGURE 6. PERFECT SEXTUPOLE MAGNET FLUX LINES

n	(RNORM = 5.5 cm) B_n (gauss)	
3	3023.4	3024.5
9	-0.28	-0.02
15	-1.59	-0.66
21	-0.40	-0.02
No. of Mesh Pts.	432	1364

TABLE III. A PERFECT SEXTUPOLE HARMONIC CALCULATIONS

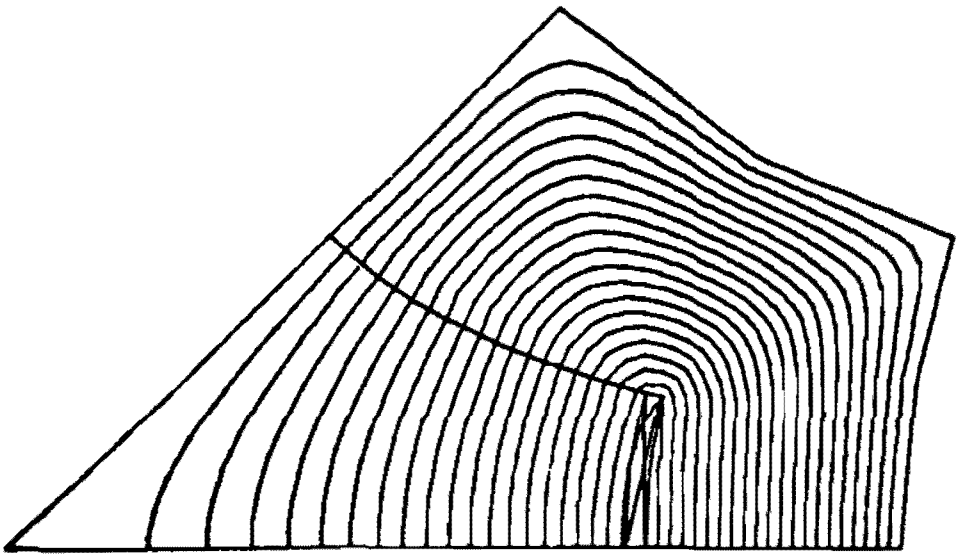


FIGURE 7a. QUADRUPOLE IN THE x-y PLANE

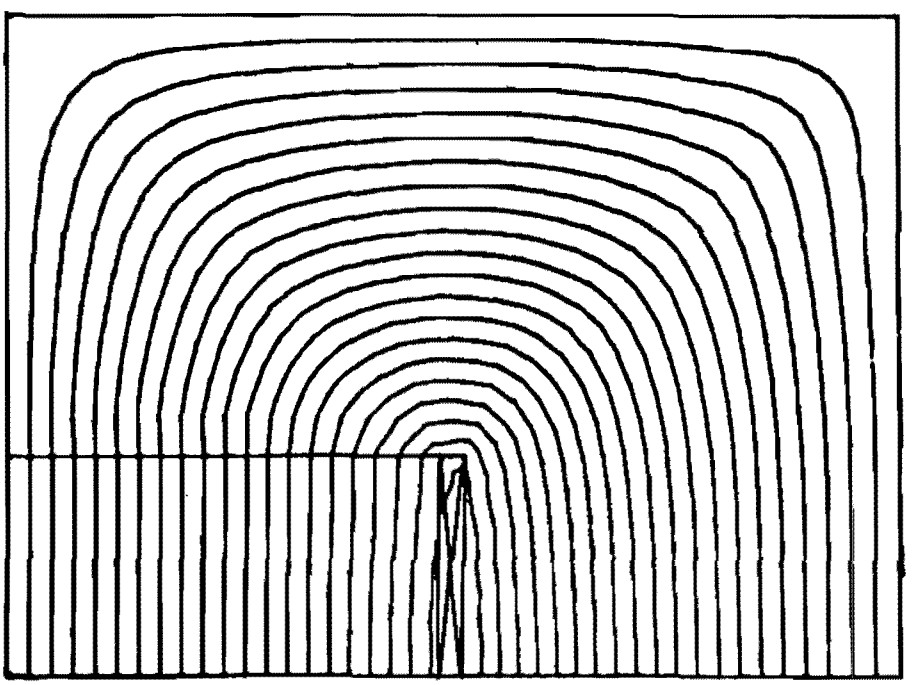


FIGURE 7b. QUADRUPOLE TRANSFORMED INTO A DIPOLE MAGNET IN THE u-v PLANE

B.13.5 Boundaries and Meshes

This section is not complete. When it is finished, it will be sent to persons who have received this manual from the Los Alamos Accelerator Code Group. The material given below is a sample of what will be included.

B.13.5.1 Boundary conditions

The solution to a two-dimensional, second-order, partial differential equation like Poisson's equation is not uniquely determined by the equation itself. One needs to place a constraint on the solution by specifying the value of the solution, and/or its derivative, along some closed boundary line.

Boundary conditions cannot be imposed arbitrarily. The most general, allowed boundary condition depends on the type of differential equation (hyperbolic, elliptic, or parabolic). Poisson's equation for the vector potential $A(x,y)$ and reluctivity $\gamma(A(x,y))$ takes the form

$$\frac{\partial}{\partial x}[\gamma(A)\frac{\partial A}{\partial x}] + \frac{\partial}{\partial y}[\gamma(A)\frac{\partial A}{\partial y}] + J(x,y) = 0, \qquad \text{(B.13.5.1)}$$

and is an elliptic differential equation. It can be show that the most general boundary condition for elliptic equations is of the form

$$aA + b[n_x\frac{\partial A}{\partial x} + n_y\frac{\partial A}{\partial y}] = c, \qquad \text{(B.13.5.2)}$$

where a, b and c are functions of position on the boundary curve. The quantities n_x and n_y are components of a unit, inward-normal vector to the boundary curve.

For purposes of the computer program POISSON the boundary curve is the perimeter of the area meshed by LATTICE. The boundaries between regions inside the meshed areas are not boundaries on which boundary conditions must be imposed. The functions a, b and c are piecewise constant on portions of the boundary. In fact, c is normally zero and either a is zero or b is zero on a given portion of the boundary. This specialized form of boundary condition is rarely absolutely correct on the boundary of the meshed region. The error resulting from using incorrect boundary conditions is usually of little practical importance when one is concerned about the magnetic field at a location far from the boundary. An excellent discussion of this point can be found in the book by D.A. Lowther and P.P. Silvester.[8] When a portion of the boundary is a line of symmetry, then the boundary condition on that portion of the boundary can be exact. On other portions of the boundary it may not even be obvious how to impose a boundary condition. The magnet designer must rely on previous experience and his expectations for the final field.

In what follows we will define the nomenclature, derive some rules of thumb for choosing boundary conditions, discuss successive region overwrite as it applies to boundary conditions, and say a few words about a new version of the code available from Lawrence Berkeley Laboratory that has more general boundary conditons.

The nomenclature used in the general theory of PDE's is:[9]

Dirichlet – $A(x,y)$ specified *everywhere* on the boundary,
Neumann – $\hat{\mathbf{n}} \cdot \nabla A$ is specified *everywhere* on the boundary,
Intermediate – Linear combination (See Eq.(B.13.5.2) is specified on the boundary,
Homogeneous – The specified value (c in Eq.(B.13.5.2)) on the boundary is zero,
Inhomogeneous – The specified value on the boundary is not zero.

The boundary conditions allowed by POISSON are a special form of the inhomogeneous, intermediate case. The authors of POISSON use the following nomenclature:

Dirichlet – $A(x,y) = c$ on *some portion* of the boundary,
Neumann – $\hat{\mathbf{n}} \cdot \nabla A$ on *some portion* of the boundary.

Usually the constant c is zero, but by using CON(20)=INPUTA (See Sec. B.5.5) or C(6) = −1 (See Sec. B.3.2) one can set the potential on some portion of the boundary to a non-zero constant value. The full closed boundary has been divided into four pieces corresponding to the four sides of the most general defining rectangle for the problem (LEFT, UPPER, RIGHT, LOWER). The elements of the CON array, CON(21), CON(22), CON(23), and CON(24), are used to specify pure Dirichlet or pure Neumann conditions on the separate sides of the problem rectangle.

Most users have little or no intuition regarding the behavior of $A(x,y)$ and are more comfortable with the direction of the magnetic induction $\mathbf{B}$ or the electric field $\mathbf{E}$. The following derivation shows how one relates Dirichlet and Neumann conditions to the field direction at the boundary. Let us write the unit inward, normal to the boundary in the form

$$\hat{\mathbf{n}} = n_x \hat{\mathbf{e}}_\mathbf{x} + n_y \hat{\mathbf{e}}_\mathbf{y}. \tag{B.13.5.3}$$

The unit tangent to the boundary at this point is perpendicular to $\hat{\mathbf{n}}$ and can be shown to be

$$\hat{\mathbf{t}} = t_x \hat{\mathbf{e}}_\mathbf{x} + t_y \hat{\mathbf{e}}_\mathbf{y} = n_y \hat{\mathbf{e}}_\mathbf{x} - n_x \hat{\mathbf{e}}_\mathbf{y}. \tag{B.13.5.4}$$

For the Dirichlet condition, since $A(x,y)$ is constant on the boundary, the component of the gradient of $A(x,y)$ parallel to the boundary is zero. This gives the relation

$$\hat{\mathbf{t}} \cdot \nabla A = n_y \frac{\partial A}{\partial x} - n_x \frac{\partial A}{\partial y} = 0. \tag{B.13.5.5}$$

But we know that $B_x = \partial A/\partial y$ and $B_y = -\partial A/\partial x$. This gives us the relation

$$\hat{\mathbf{t}} \cdot \nabla A = -(n_x B_x + n_y B_y) = -\hat{\mathbf{n}} \cdot \mathbf{B} = 0 \tag{B.13.5.6}$$

This implies that the magnetic field must be parallel to the boundary.

For electrostatic problem, it is easily seen that $V(x,y)$ constant on the boundary leads to the equation

$$\hat{\mathbf{t}} \cdot \nabla V = -(t_x E_x + t_y E_y) = -\hat{\mathbf{t}} \cdot \mathbf{E} = 0 \qquad \text{(B.13.5.7)}$$

This implies that the electric field must be perpendicular to the boundary.

For the Neumann condition, the normal derivative of $A(x,y)$ vanishes. This gives the relation

$$\hat{\mathbf{n}} \cdot \nabla A = n_x \frac{\partial A}{\partial x} + n_y \frac{\partial A}{\partial y} = 0. \qquad \text{(B.13.5.8)}$$

But we know that $B_x = \partial A/\partial y$ and $B_y = -\partial A/\partial x$, and $n_x = -t_y$ and $n_y = t_x$. This gives us the relation

$$\hat{\mathbf{n}} \cdot \nabla A = t_x B_x + t_y B_y = \hat{\mathbf{t}} \cdot \mathbf{B} = 0 \qquad \text{(B.13.5.9)}$$

This implies that the magnetic field must be perpendicular to the boundary.

For electrostatic problem, it is easily seen that $\hat{\mathbf{n}} \cdot \nabla V(x,y) = 0$ leads to the sequence of equations

$$\hat{\mathbf{n}} \cdot \nabla V = n_x \frac{\partial V}{\partial x} + n_y \frac{\partial V}{\partial y} = 0, \qquad \text{(B.13.5.10)}$$

and

$$\hat{\mathbf{n}} \cdot \nabla V = -(n_x E_x + n_y E_y) = -\hat{\mathbf{n}} \cdot \mathbf{E} = 0 \qquad \text{(B.13.5.11)}$$

This implies that the electric field must be parallel to the boundary. Table B.13.5.I summarizes the results.

Table B.13.5.I. Implications of Dirichlet and Neumann Conditions

Field	Dirichlet	Neumann
Magnetic	parallel to boundary	perpendicular to boundary
Electrostatic	perpendicular to boundary	parallel to boundary

In cases of high symmetry these conditions can hold exactly. There are two main types of two-dimensional symmetry — reflection and rotation. Under each of these types there are two subtypes — electric currents change sign or they do not.

Figure B.13.5.1 shows the two types of reflection symmetry and the directions of the field lines. The plane between two equal and opposite currents, which is called a separatrix, has $\hat{\mathbf{n}} \cdot \mathbf{B} = 0$ hence implies Dirichlet boundary conditions. The plane

between currents of the same sign, which is called an "even line", is a Neumann boundary. This assumes of course that the line coincides with one of the boundary lines for the problem.

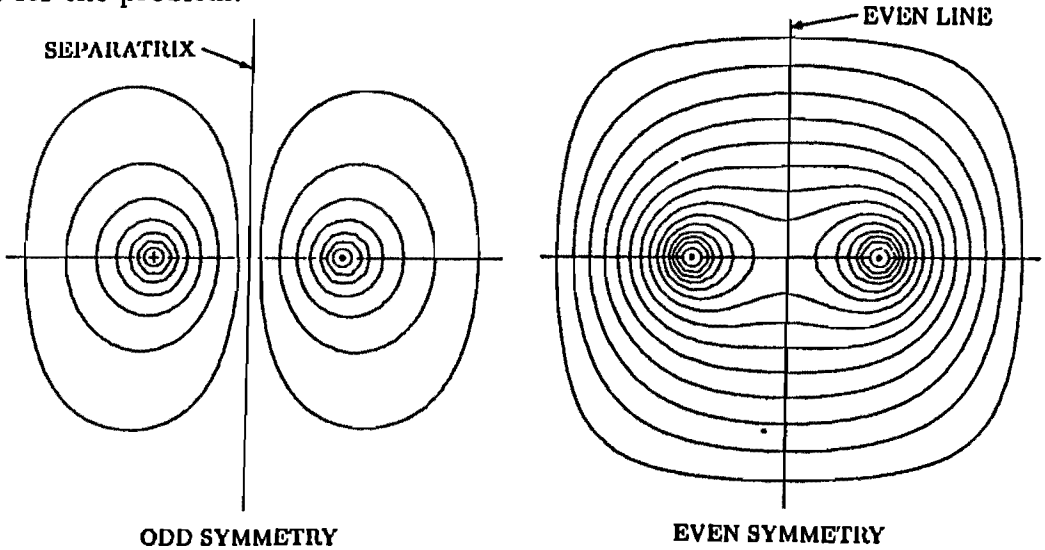


Figure B.13.5.1: The direction of magnetic field lines for two types of reflection symmetry: separatrix and even line.

If there is rotational symmetry, this has the effect of introducing angular arrays of separatrices and/or even lines as illustrated in Fig. B.13.5.2. If any of these special lines corresponds to a boundary for the problem, then one can use the rule:

Separatrix — Dirichlet
Even line — Neumann.

When there is iron in the problem, it is generallly true that the field inside the iron near the surface is parallel to the surface. This means that if an iron boundary coincides with a problem boundary, then the boundary condition is Dirichlet.

When one cannot use symmetry or iron boundaries, then the allowed boundary conditions will only be approximate. It is still useful to look for separatrices between currents of different sign. When a separatrix hits a problem boundary, that boundary tends to be nearly a Dirichlet boundary. See Fig. B.13.5.3. Another useful fact is that the field outside an iron surface tends to be normal to that surface. This may help in guessing the appropriate, approximate boundary conditions.

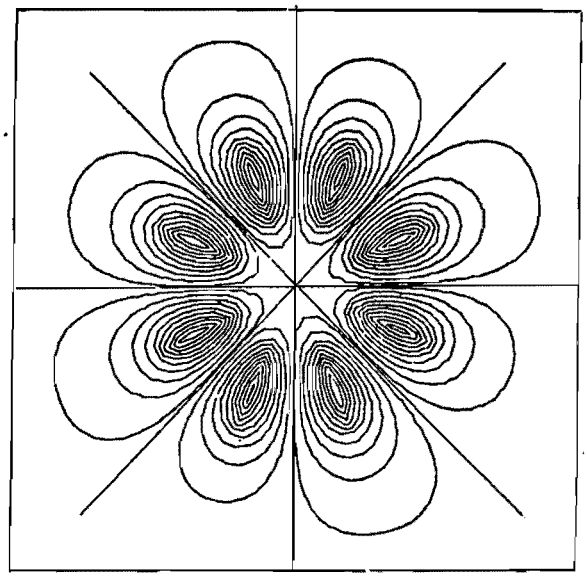


Figure B.13.5.2: Rotational symmetry introduces arrays of reflection lines. In this example only one half of the upper right rectangle is required to define the probelm, and hence the 45° line will become a problem boundary.

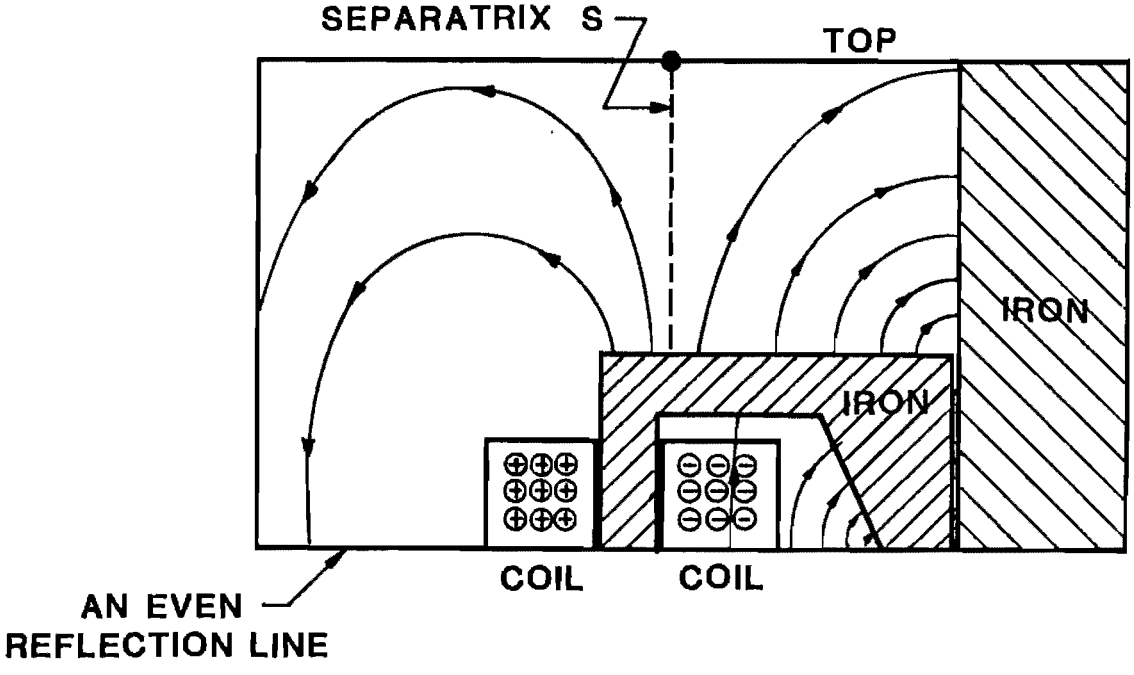


Figure B.13.5.3: Use of a separtrix to decide on an approximate boundary condition. The separatrix S intersects the boundary TOP, hence the field will be approximately parallel to TOP, and hence Dirichlet.

B.13.5.2 Meshes

(This subsection will discuss the numbering of the mesh points and discuss the order in which the iteration scheme sweeps through the lattice.)

B.13.6 Numerical methods used in POISSON and PANDIRA

The customary way to solve the magnetostatic problem would be to map the interior region with a mesh and then solve for the vector potential at these mesh points using a discretized form of Poisson's Equation and the necessary boundary conditions. The procedure used in this program does not directly solve Poisson's Equation; rather, it solves a form of Ampere's Law over a closed region made up of triangular plates. The procedure is iterative using either a successive over-relaxation algorithm (POISSON) or a direct method (PANDIRA) combined with an iterative correction scheme for the reluctivity function to obtain an estimate of the solution. The solution is the vector potential at each mesh point (vertices of the triangles).

The description of the procedure will be divided into the following sections:

B.13.6.1 Mesh formulation;
B.13.6.2 Numerical procedure for calculating the vector potential at a mesh point;
B.13.6.3 Boundary conditions;
B.13.6.4 Calculations of fields;
B.13.6.5 Computer algorithm using SOR method;
B.13.6.6 Computer algorithm using the direct method.

B.13.6.1 Mesh formulation.

An irregular triangular mesh is generated over the region, within the boundaries conforming to the following rules:

1. The triangulation will form a regular topology; that is, it is topologically equivalent to an equilateral triangle array in which six triangles meet at every interior mesh point;

2. Any part of the interior that is unique or has the same attributes such as current density, permeability, etc., must be set-up as an interior subregion;

3. Any triangle along an external boundary or boundary of a closed interior region will have two of its vertices lying on these boundary defining lines. These boundaries will then be described by piece-wise linear segments which are the sides of the interpolating triangles.

Since any polygonal region can be triangulated, the method can be applied to regions of any shape and will produce a mesh in which boundaries and interfaces lie entirely on mesh lines. In those areas where the gradient of the vector potential is expected to be large the triangles should be made more dense.

B.13.6.2 Numerical procedure for calculating vector potentials at a mesh point.

The procedure solves for the vector potential, $\mathbf{A}(\mathbf{r})$, at each mesh point using a relationship derived from Ampere's Law,

$$\oint_C \mathbf{H}(\mathbf{r}) \cdot d\mathbf{l} = \int_S \mathbf{J}(\mathbf{r}) \cdot \hat{\mathbf{n}}\, da. \qquad \text{(B.13.6.1)}$$

where $\mathbf{H}(\mathbf{r})$ is the magnetic field; $\mathbf{J}(\mathbf{r})$ is the current density; $\mathbf{r}$ is $r_x\,\hat{\mathbf{e}}_x + r_y\,\hat{\mathbf{e}}_y + r_z\,\hat{\mathbf{e}}_z$; $\hat{\mathbf{n}}$ is a unit vector normal to the surface S; C is the contour of the line integral enclosing the surface S; and S is the surface enclosed by the contour C. The

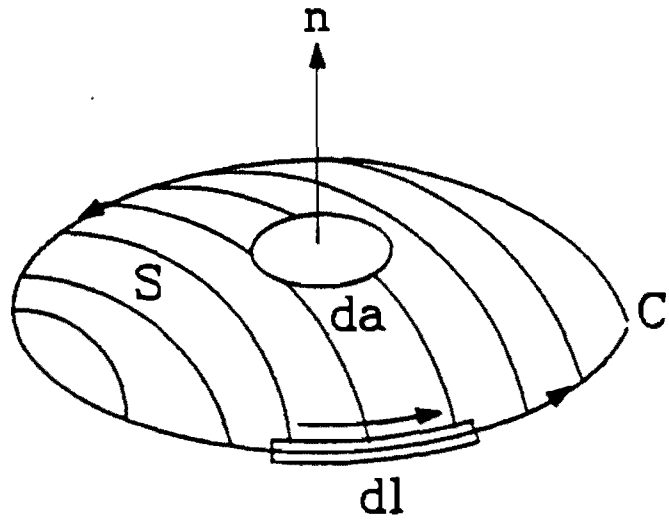


Figure B.13.6.1: Relation between the surface and contour elements in Eq.(B.13.6.1)

solution is iterative in nature in that the calculated value of the vector potential at a mesh point is a function of previous estimates and arbitrary initializations. The following approximations are made for each triangular area:

1. The vector potential is linear over a triangle;
2. A reluctivity is associated with each triangle and is constant over it;
3. A current density is associated with each triangle and is constant over it.

Since Eq. (B.13.6.1) must be expressed in terms of the vector potential $\mathbf{A}(\mathbf{r})$, the first step is to express the magnetic field $\mathbf{H}(\mathbf{r})$ in terms of the reluctivity $\gamma(|\mathbf{B}(\mathbf{r})|)$ and the magnetic induction $\mathbf{B}(\mathbf{r})$. This relationship is

$$\gamma(|\mathbf{B}(\mathbf{r})|)\,\mathbf{B}(\mathbf{r}) = \mu_0\mathbf{H}(\mathbf{r}). \qquad \text{(B.13.6.2)}$$

By substituting this into Eq.(B.13.6.1), Ampere's Law becomes

$$\oint_C \gamma(|\mathbf{B}(\mathbf{r})|)\,\mathbf{B}(\mathbf{r}) \cdot d\mathbf{l} = \mu_0 \int_S \mathbf{J}(\mathbf{r}) \cdot \hat{\mathbf{n}}\, da. \tag{B.13.6.3}$$

The final step is to use the fact that the magnetic induction $\mathbf{B}(\mathbf{r})$ is the curl of the vector potential, namely,

$$\mathbf{B}(\mathbf{r}) = \nabla \times \mathbf{A}(\mathbf{r}). \tag{B.13.6.4}$$

Substituting this into Eq.(B.13.6.3) give the required form

$$\oint_C \gamma(|\mathbf{B}(\mathbf{r})|)\,\nabla \times \mathbf{A}(\mathbf{r}) \cdot d\mathbf{l} = \mu_0 \int_S \mathbf{J}(\mathbf{r}) \cdot \hat{\mathbf{n}}\, da. \tag{B.13.6.5}$$

Next the curl is expanded in terms of its components, giving

$$\nabla \times \mathbf{A}(\mathbf{r}) = \left(\frac{\partial A_z}{\partial y} - \frac{\partial A_y}{\partial z}\right)\hat{\mathbf{e}}_x + \left(\frac{\partial A_x}{\partial z} - \frac{\partial A_z}{\partial x}\right)\hat{\mathbf{e}}_y + \left(\frac{\partial A_y}{\partial x} - \frac{\partial A_x}{\partial y}\right)\hat{\mathbf{e}}_z. \tag{B.13.6.6}$$

The induction $\mathbf{B}(\mathbf{r})$ is by definition in the simulation a 2-dimensional vector lying in the x-y plane. The vector potential $\mathbf{A}(\mathbf{r})$ can then be chosen normal to this plane with only the A_z-component being non-zero,

$$A_x = A_y = 0. \tag{B.13.6.7}$$

A necessary condition to be satisfied for Poisson's equation, in addition to Ampere's Law, is that of the Coulomb gauge choice,

$$\nabla \cdot \mathbf{A}(\mathbf{r}) = 0, \tag{B.13.6.8}$$

which can be expanded in the following form,

$$\frac{\partial A_x}{\partial x} + \frac{\partial A_y}{\partial y} + \frac{\partial A_z}{\partial z} = 0. \tag{B.13.6.9}$$

Since Eq.(B.13.6.7) implies

$$\frac{\partial A_x}{\partial x} = \frac{\partial A_y}{\partial y} = 0, \tag{B.13.6.10}$$

the only remaining condition from Eq.(B.13.6.9) is

$$\frac{\partial A_z}{\partial z} = 0. \tag{B.13.6.11}$$

This is equivalent to the statement that the A_z is not a function of z. By making the position vector $\mathbf{r}$ a 2-dimensional vector in the x-y plane, the Coulomb gauge condition has been satisfied. Henceforth it will be understood that

$$\mathbf{r} = r_x\,\hat{\mathbf{e}}_x + r_y\,\hat{\mathbf{e}}_y. \tag{B.13.6.12}$$

It follows that

$$\mathbf{B} = \nabla \times \mathbf{A}(\mathbf{r}) = \frac{\partial A_z}{\partial y}\,\hat{\mathbf{e}}_x - \frac{\partial A_z}{\partial x}\,\hat{\mathbf{e}}_y, \tag{B.13.6.13}$$

and Eq. (B.13.6.5) becomes

$$\oint_C \gamma(|\mathbf{B}(\mathbf{r})|)\left[\frac{\partial A_z}{\partial y}\,\hat{\mathbf{e}}_x - \frac{\partial A_z}{\partial x}\,\hat{\mathbf{e}}_y\right] \cdot d\mathbf{l} = \mu_0 \int_S \mathbf{J}(\mathbf{r}) \cdot \hat{\mathbf{n}}\, da. \tag{B.13.6.14}$$

This section will only handle the development in Cartesian coordinates; the theory in cylindrical coordinates is analogous.

Before continuing with this derivation, a brief discussion of the complex notation used in place of vector notation by the authors of POISSON would be helpful. A two-dimensional vector

$$\mathbf{a} = a_x\,\hat{\mathbf{e}}_x + a_y\,\hat{\mathbf{e}}_y \tag{B.13.6.15}$$

can be represented as a complex number

$$a = a_x + i\,a_y. \tag{B.13.6.16}$$

The complex conjugate of a is defined by the relation

$$a^* = a_x - i\,a_y. \tag{B.13.6.17}$$

Given two vectors

$$\mathbf{a} = a_x\,\hat{\mathbf{e}}_x + a_y\,\hat{\mathbf{e}}_y \tag{B.13.6.18}$$

$$\mathbf{b} = b_x\,\hat{\mathbf{e}}_x + b_y\,\hat{\mathbf{e}}_y, \tag{B.13.6.19}$$

and their complex representation

$$a = a_x + i\,a_y \tag{B.13.6.20}$$

$$b = b_x + i\,b_y, \tag{B.13.6.21}$$

then the product a^*b can be written as

$$a^*b = (a_x - ia_y)(b_x + ib_y) = a_xb_x + a_yb_y + i\,(a_xb_y - a_yb_x) \tag{B.13.6.22}$$

and contains both the scalar and vector products of the vectors, namely,

$$\mathbf{a} \cdot \mathbf{b} = a_xb_x + a_yb_y \tag{B.13.6.23}$$

$$\mathbf{a} \times \mathbf{b} = (a_xb_y - a_yb_x)\,\hat{\mathbf{e}}_z. \tag{B.13.6.24}$$

Hence we can write

$$a^*b = \mathbf{a} \cdot \mathbf{b} + i\,\hat{\mathbf{e}}_z \cdot (\mathbf{a} \times \mathbf{b}), \tag{B.13.6.25}$$

and it follows that

$$\mathbf{a} \cdot \mathbf{b} = Re\{a^* b\}. \tag{B.13.6.26}$$

The above equation can be used to get the integrand on the left-hand side of Eq.(B.13.6.14) into a tractable form.

Corresponding to the relationship between the vector $\mathbf{B}(\mathbf{r})$ and the vector potential,

$$\mathbf{B}(\mathbf{r}) = \frac{\partial A_z}{\partial y}\,\hat{\mathbf{e}}_x - \frac{\partial A_z}{\partial x}\,\hat{\mathbf{e}}_y, \tag{B.13.6.27}$$

one can define a complex function

$$B(x,y) = \frac{\partial A_z}{\partial y} + i\left(-\frac{\partial A_z}{\partial x}\right), \tag{B.13.6.28}$$

and its complex conjugate

$$B^*(x,y) = \frac{\partial A_z}{\partial y} + i\,\frac{\partial A_z}{\partial x}. \tag{B.13.6.29}$$

The vector $d\mathbf{l}$ will not be represented as a sum of its components but rather as a complex number dl. The logic for this will be seen later in the analysis.

The integrand, excluding the $\gamma(|\mathbf{B}(\mathbf{r})|)$ term, in this complex notation is

$$\mathbf{B}(\mathbf{r}) \cdot d\mathbf{l} = Re\{B^* dl\} = Re\left\{\left[\frac{\partial A_z}{\partial y} + i\,\frac{\partial A_z}{\partial x}\right] dl\right\}. \tag{B.13.6.30}$$

The potential A_z can be viewed as a scalar function of the complex number z, namely,

$$A_z(x,y) = A_z(z) = A_z(x + i\,y). \tag{B.13.6.31}$$

It is also useful to think of A_z as a function of both z and z^* since

$$x = \frac{1}{2}(z + z^*) \;\; and \;\; y = \frac{i}{2}(z^* - z). \tag{B.13.6.32}$$

Now evaluate the partial derivatives $\partial A_z(z,z^*)/\partial x$ and $\partial A_z(z,z^*)/\partial y$.
It is easy to see that

$$\frac{\partial A_z(z,z^*)}{\partial x} = \frac{\partial A_z(z,z^*)}{\partial z}\frac{\partial z}{\partial x} + \frac{\partial A_z(z,z^*)}{\partial z^*}\frac{\partial z^*}{\partial x}, \tag{B.13.6.33}$$

and

$$\frac{\partial A_z(z,z^*)}{\partial y} = \frac{\partial A_z(z,z^*)}{\partial z}\frac{\partial z}{\partial y} + \frac{\partial A_z(z,z^*)}{\partial z^*}\frac{\partial z^*}{\partial y}. \tag{B.13.6.34}$$

Because of the relations

$$\frac{\partial z}{\partial x} = 1\,, \quad \frac{\partial z^*}{\partial x} = 1\,, \tag{B.13.6.35}$$

$$\frac{\partial z}{\partial y} = i\,, \; and \; \frac{\partial z^*}{\partial y} = -i, \tag{B.13.6.36}$$

it follows that the partial derivatives can be expressed as

$$\frac{\partial A_z(z,z^*)}{\partial x} = \left(\frac{\partial}{\partial z} + \frac{\partial}{\partial z^*}\right) A_z(z,z^*) \tag{B.13.6.37}$$

and

$$\frac{\partial A_z(z,z^*)}{\partial y} = i\left(\frac{\partial}{\partial z} - \frac{\partial}{\partial z^*}\right) A_z(z,z^*). \tag{B.13.6.38}$$

Substituting these relations into Eq.(B.13.6.29) gives

$$B^*(x,y) = i\left[\frac{\partial}{\partial z} + \frac{\partial}{\partial z^*} - i^2\left(\frac{\partial}{\partial z} - \frac{\partial}{\partial z^*}\right)\right] A_z(z,z^*)$$

or

$$B^*(x,y) = 2i\,\frac{\partial A_z(z,z^*)}{\partial z}. \tag{B.13.6.39}$$

Equation (B.13.6.30), with this new notation, becomes

$$\mathbf{B}(\mathbf{r}) \cdot d\mathbf{l} = Re\left\{2i\,\frac{\partial A_z(z,z^*)}{\partial z} dl\right\}. \tag{B.13.6.40}$$

A few words about the intended numerical scheme are appropriate at this point. The vector potential at each mesh point, including the boundary points, is calculated as a function of the following attributes of the six surrounding triangles:

1. The magnitude and position of the vector potentials at each of the surrounding six mesh points,
2. The area of each of these triangles,
3. The current density of each of these triangles,
4. The reluctivity of each of these triangles.

The geometry used to calculated the vector potential A_0 is shown in Fig. B.13.6.2.

For notational convenience the following conventions are adopted for the remainder of this section:

$$A_i \equiv A_z\ (z_i, z^*_i)\ \textit{and}\ J_i \equiv J_z\,(z_i, z^*_i)\,.$$

Ampere's Law is now applied to a contour around the point $i = 0$. The contour is not on the circumference of this hexagon but rather on a dodecagon whose circumference passes through the midpoints of each side shared by two triangles and the respective centroids of each triangle. The path is shown in Fig. B.13.6.3.

The procedure for evaluating the integrals will be developed for one of the triangles shown in Fig.B.13.6.4, designated "triangle 2", and then generalized for all six triangles forming the dodecagon shown in Fig.B.13.6.3.

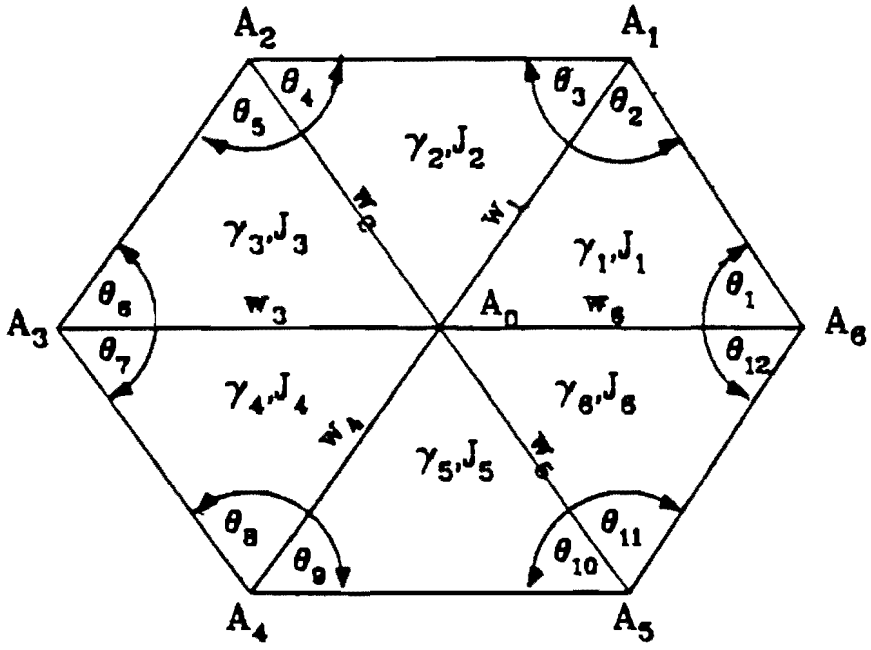


Figure B.13.6.2: Definition of physical quantities relative to the mesh geometry. A_i is the vector potential at mesh point i; γ_i's are reluctivities; J_i's are current densitites; and w_i's are coupling parameters.

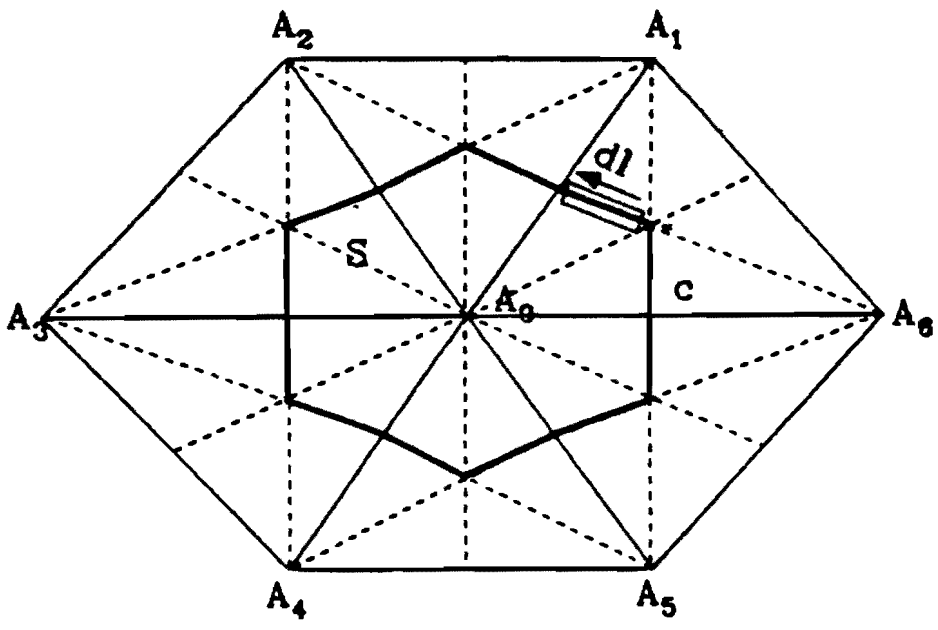


Figure B.13.6.3: Path of contour integration for Ampere's Law.

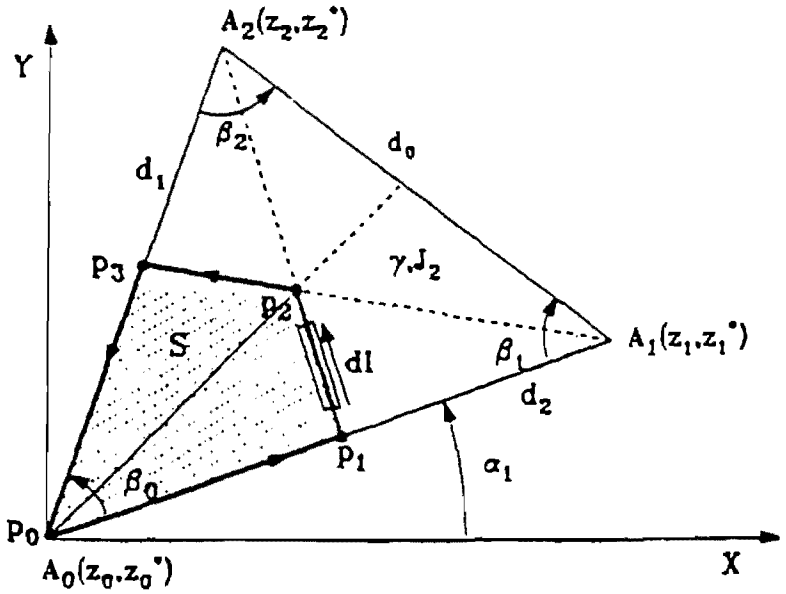


Figure B.13.6.4: The contour integration over the dodecahedron can be broken into integrations over six quadrangles.

Ampere's equation for the shaded portion of Fig. B.13.6.4 using the complex notation, is given by

$$\oint_C \gamma_2(|\mathbf{B}(\mathbf{r})|)\, Re\left\{2i\frac{\partial A}{\partial z}\, dl\right\} = \mu_0\, J_2 \frac{a_2}{3}, \tag{B.13.6.41}$$

where C is the path $(p_0, p_1, p_2, p_3, p_0)$; a_2 is the area of "triangle 2"; J_2 is the normal component of current density for "triangle 2"; γ_2 is the reluctivity for "triangle 2"; and μ_0 is 4×10^{-7} in rationalized MKS units.

To evaluate the left-hand side of Eq.(B.13.6.41), a functional relationship of the vector potential over the triangle must be developed. The assumption of the program is that the vector potential varies linearly over any triangle in the mesh. Using the three-point form for the equation of a plane, an expression for the vector potential at any point can be developed. The positions of the three vertices and their respective vector potentials are used to determine the coefficients of the expression. The three-point form can be written as the determinant,

$$\begin{vmatrix} z - z_0 & z^* - z_0^* & A - A_0 \\ z_1 - z_0 & z_1^* - z_0^* & A_1 - A_0 \\ z_2 - z_0 & z_2^* - z_0^* & A_2 - A_0 \end{vmatrix} = 0. \tag{B.13.6.42}$$

The plane in function space is pictured in Fig. B.13.6.5.

Expand this determinant and solve for A. Introduce the notation

$$\Delta A_1 = A_1 - A_0 \tag{B.13.6.43}$$

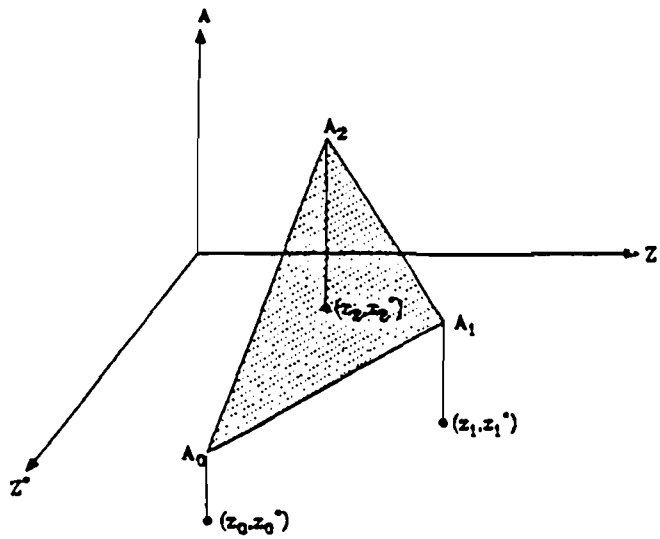


Figure B.13.6.5: The vector potential $A(z, z^*)$ is taken to be a linear function over each triangle. The functional dependence is completely determined by the values of A at the corners.

and

$$\Delta A_2 = A_2 - A_0. \tag{B.13.6.44}$$

It can be shown that the determinant reduces to

$$(z - z_0)(z_1^* - z_0^*)\Delta A_2 + (z_2 - z_0)(z^* - z_0^*)\Delta A_1 + (z_1 - z_0)(z_2^* - z_0^*)(A - A_0)$$
$$- (z_2 - z_0)(z_1^* - z_0^*)(A - A_0) - (z - z_0)(z_2^* - z_0^*)\Delta A_1 - (z_1 - z_0)(z^* - z_0^*)\Delta A_2 = 0, \tag{B.13.6.45}$$

which can be solved for A, giving

$$A = A_0 + \frac{[\Delta A_1]\,[(z - z_0)\,((z_2^* - z_0^*) - (z_2 - z_0)\,(z^* - z_0^*)]}{(z_1 - z_0)\,(z_2^* - z_0^*) - (z_2 - z_0)\,(z_1^* - z^*{}_0)}$$
$$+ \frac{[\Delta A_2]\,[(z_1 - z_0)\,((z^* - z_0^*) - (z - z_0)\,(z_1^* - z_0^*)]}{(z_1 - z_0)\,(z_2^* - z_0^*) - (z_2 - z_0)\,(z_1^* - z_0^*)}. \tag{B.13.6.46}$$

Taking the origin at the coordinates (z_0, z_0^*) reduces Eq. (B.13.6.46) to

$$A = A_0 + \frac{\Delta A_1 z_2^* - \Delta A_2 z_1^*}{z_1 z_2^* - z_2 z_1^*}\, z + \frac{\Delta A_2 z_1 - \Delta A_1 z_2}{z_1 z_2^* - z_2 z_1^*}\, z^*, \tag{B.13.6.47}$$

or

$$A = A_0 + Cz - C^* z^* \tag{B.13.6.48}$$

where

$$C = \frac{\Delta A_1 z_2^* - \Delta A_2 z_1^*}{z_1 z_2^* - z_2 z_1^*}. \tag{B.13.6.49}$$

Using Eq.(B.13.6.47) it is an easy task to calculate

$$\frac{\partial A}{\partial z} = C = \frac{\Delta A_1 z_2^* - \Delta A_2 z_1^*}{z_1 z_2^* - z_2 z_1^*}. \tag{B.13.6.50}$$

Substituting this expression into Eq.(B.13.6.41) and making the assumption that the reluctivity is constant, namely

$$\gamma_2(|\mathbf{B}(\mathbf{r})|) \simeq \gamma_2 = constant \tag{B.13.6.51}$$

and is not a function of $|\mathbf{B}(\mathbf{r})|$, the results are

$$\oint_C \gamma_2 Re\left[\left(2i\frac{\Delta A_1 z_2^* - \Delta A_2 z_1^*}{z_1 z_2^* - z_2 z_1^*}\right)(dl)\right] = \mu_0 J_2 \frac{a_2}{3}. \tag{B.13.6.52}$$

Since the integrand is not a function of z it may be taken outside the integral sign. The result is

$$\gamma_2 Re\left[2i\frac{\Delta A_1 z^*{}_2 - \Delta A_2 z^*{}_1}{z_1 z^*{}_2 - z_2 z^*{}_1}\oint_C dl\right] = \mu_0 J_2 \frac{a_2}{3}. \tag{B.13.6.53}$$

Before evaluating the line integral it should be recalled that all six triangles will be used in the final equation for A_0. The assumption is made that the effective $\mathbf{B}(\mathbf{r})$ field on the common side of any two adjacent triangles is the average of the $\mathbf{B}(\mathbf{r})$ fields in the two triangles. Therefore the line integrals, in the opposite directions, along any of these common sides will cancel each other. This is illustrated in Fig. B.13.6.6.

Triangles on the boundary will not have their path integrals canceled along the boundary. Hence the advantage of this type of configuration implies that the sum, over the entire problem area, of Ampere's Law applied locally around each mesh point is equal to Ampere's Law applied around the boundary of the complete problem. See Fig. B.13.6.7.

Returning to the geometry of Fig. B.13.6.4, this leaves the integration over the open path $C = (p_1,\ p_2,\ p_3)$ where the points p_1 and p_3 have coordinates given by the complex numbers

$$p_1 = \frac{z_1 + z_0}{2} \tag{B.13.6.54}$$

and

$$p_3 = \frac{z_2 + z_0}{2}. \tag{B.13.6.55}$$

The noncanceling part of the contour integral is

$$\oint_C dl = \int_{p_1}^{p_2} dl + \int_{p_2}^{p_3} dl$$

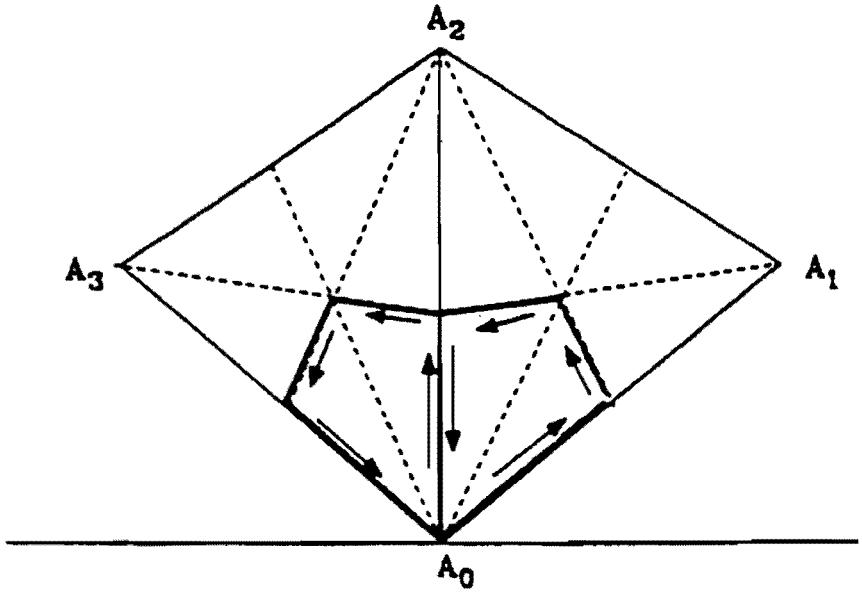


Figure B.13.6.6: It is assumed that the effective $\mathbf{B}(\mathbf{r})$ on common sides of of adjacent triangles are the same and therefore the line integrals along these common sides will cancel one another.

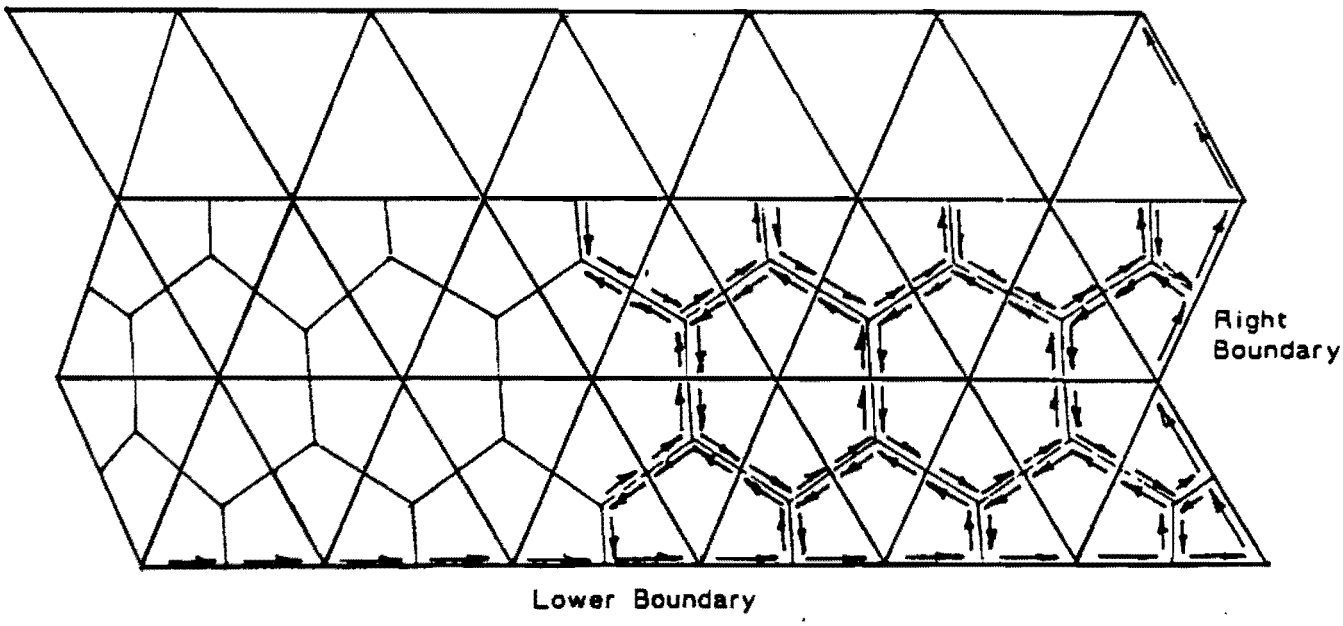


Figure B.13.6.7: The sum of contour integrals around each point adds up to a contour integral around the whole region because of cancelation of interior contours.

$$
\begin{aligned}
&= (p_2 - p_1) + (p_3 - p_2) \\
&= p_3 - p_1 \\
&= \frac{z_2 + z_0}{2} - \frac{z_1 + z_0}{2} \\
&= \frac{z_2 - z_1}{2}.
\end{aligned} \tag{B.13.6.56}
$$

Substituting this result into equation (13.6.53) gives the equation

$$
\gamma_2 Re\left[i\frac{(\Delta A_1 z_2^* - \Delta A_2 z_1^*)(z_2 - z_1)}{z_1 z_2^* - z_2 z_1^*}\right] = \mu_0 J_2 \frac{a_2}{3}. \tag{B.13.6.57}
$$

In order to obtain the real part of the factor on the left-hand side, it is necessary to express the positional variables z_i and z_i^* in terms of the angles $(\beta_0, \beta_1, \beta_2)$ and the side dimensions (d_0, d_1, d_2) of the triangle shown in Fig. B.13.6.4. Begin by evaluating the denominator, namely

$$
z_1\, z_2^* - z_2\, z_1^*. \tag{B.13.6.58}
$$

The z's can be expressed in exponential form as follows:

$$
z_1 = d_2 e^{i\alpha_1} \tag{B.13.6.59}
$$

$$
z_1^* = d_2 e^{-i\alpha_1} \tag{B.13.6.60}
$$

$$
z_2 = d_1 e^{i(\alpha_1 + \beta_0)} \tag{B.13.6.61}
$$

$$
z_1^* = d_1 e^{-i(\alpha_1 + \beta_0)}. \tag{B.13.6.62}
$$

When these expressions are substituted into Eq.(B.13.6.58) the result is

$$
\begin{aligned}
z_1\, z^*{}_2 - z_2\, z^*{}_1 &= d_2 e^{i\alpha_1} d_1 e^{-i(\alpha_1+\beta_0)} - d_1 e^{i(\alpha_1+\beta_0)} d_2 e^{-i\alpha_1} \\
&= d_1 d_2 \left(e^{-i\beta_0} - e^{i\beta_0}\right) \\
&= -i2 d_1 d_2 \sin(\beta_0).
\end{aligned} \tag{B.13.6.63}
$$

Next evaluate the numerator,

$$
i\,(\Delta A_1\, z_2^* - \Delta A_2\, z_1^*)\,(z_2 - z_1), \tag{B.13.6.64}
$$

using the relations

$$
\begin{aligned}
z_2 - z_1 &= -d_0 \cos(\beta_0 - \alpha_1) + i d_0 \sin(\beta_0 - \alpha_1) \\
&= d_0 e^{i(\pi + \alpha_1 - \beta_1)},
\end{aligned} \tag{B.13.6.65}
$$

$$
\begin{aligned}
z^*{}_1\,(z_2 - z_1) &= d_2 e^{-i\alpha_1} \cdot d_0 e^{i(\pi+\alpha_1-\beta_1)} \\
&= -d_0 d_2 e^{-i\beta_1},
\end{aligned} \tag{B.13.6.66}
$$

$$z_2^*(z_2 - z_1) = d_1 e^{-i(\alpha_1+\beta_0)} \cdot d_0 e^{i}(\pi + \alpha_1 - \beta_1)$$
$$= -d_0 d_1 e^{i\beta_2}. \qquad \text{(B.13.6.67)}$$

We find that Eq.(B.13.6.64) can be written,

$$i(\Delta A_1 z_2^* - \Delta A_2 z_1^*)(z_2 - z_1) = i\left(\Delta A_1 d_0 d_1 e^{i\beta_2} + \Delta A_2 d_0 d_2 e^{-i\beta_1}\right). \qquad \text{(B.13.6.68)}$$

Having evaluated both the numerator and denominator of Eq. (B.13.6.57), we can combine our results to obtain the following,

$$Re\left[\frac{i(\Delta A_1 z_2^* - \Delta A_2 z_1^*)(z_2 - z_1)}{z_1 z_2^* - z_2 z_1^*}\right] = Re\left[\frac{i\left(\Delta A_1 d_0 d_1 e^{i\beta_2} + \Delta A_2 d_0 d_2 e^{-i\beta_1}\right)}{-i2d_1 d_2 \sin(\beta_0)}\right]$$

$$= Re\left\{\frac{[\Delta A_1 d_0 d_1][\cos(\beta_2) + i\sin(\beta_2)] + [\Delta A_2 d_0 d_2][\cos(-\beta_1) + i\sin(-\beta_1)]}{-2d_1 d_2 \sin(\beta_0)}\right\}$$

$$= -\frac{\Delta A_1 d_0 d_1 \cos(\beta_2)}{2d_1 d_2 \sin(\beta_0)} - \frac{\Delta A_2 d_0 d_2 \cos(\beta_1)}{2d_1 d_2 \sin(\beta_0)}$$

$$= -\frac{\Delta A_1 \cos(\beta_2)\sin(\beta_0)}{2\sin(\beta_0)\sin(\beta_2)} - \frac{\Delta A_2 \cos(\beta_1)\sin(\beta_0)}{2\sin(\beta_0)\sin(\beta_1)}, \qquad \text{(B.13.6.69)}$$

where

$$\frac{d_0}{d_1} = \frac{\sin(\beta_0)}{\sin(\beta_1)}, \qquad \text{(B.13.6.70)}$$

and

$$\frac{d_0}{d_2} = \frac{\sin(\beta_0)}{\sin(\beta_2)}. \qquad \text{(B.13.6.71)}$$

The final expression is in terms of the cotangents of the interior angles and the vector potentials at the vertices, namely,

$$Re\left[\frac{i(\Delta A_1 z_2^* - \Delta A_2 z_1^*)(z_2 - z_1)}{z_1 z_2^* - z_2 z_1^*}\right] = -\frac{1}{2}[\Delta A_1 \cot(\beta_2) + \Delta A_2 \cot(\beta_1)]$$

$$= -\frac{1}{2}\{[A_1 - A_0][\cot(\beta_2)] + [A_2 - A_0][\cot(\beta_1)]\}$$

$$= \frac{1}{2}\{[A_0][\cot(\beta_1) + \cot(\beta_2)] - A_1\cot(\beta_2) - A_2\cot(\beta_1)\}. \qquad \text{(B.13.6.72)}$$

Substituting this result into Eq.(B.13.6.57) gives the relation

$$\frac{\gamma_2}{2}\left\{[A_0]\left[\cot(\beta_1)+\cot(\beta_2)\right]-A_1\cot(\beta_2)-A_2\left[\cot(\beta_1)\right]\right\}=\mu_0 J_2\frac{a_2}{3}. \quad \text{(B.13.6.73)}$$

Change the angle representation from β's to θ's as shown in Fig. B.13.6.2. The result is

$$\frac{\gamma_2}{2}\left\{[A_0]\left[\cot(\theta_3)+\cot(\theta_4)\right]-A_1\cot(\theta_4)-A_2\left[\cot(\theta_3)\right]\right\}=\mu_0 J_2\frac{a_2}{3}. \quad \text{(B.13.6.74)}$$

The complete line integral around the dodecagon is the sum of the partial paths over each triangle. See Fig. B.13.6.3. The result is

$$\begin{aligned}
&\frac{\gamma_1}{2}\left\{[A_0]\left[\cot(\theta_1)+\cot(\theta_2)\right]-A_1\cot(\theta_1)-A_2\cot(\theta_2)\right\}\\
&+\frac{\gamma_2}{2}\left\{[A_0]\left[\cot(\theta_3)+\cot(\theta_4)\right]-A_2\cot(\theta_3)-A_1\cot(\theta_4)\right\}\\
&+\frac{\gamma_3}{2}\left\{[A_0]\left[\cot(\theta_5)+\cot(\theta_6)\right]-A_3\cot(\theta_5)-A_2\cot(\theta_6)\right\}\\
&\qquad\qquad +\\
&\qquad\qquad \vdots\\
&+\frac{\gamma_6}{2}\left\{[A_0]\left[\cot(\theta_{11})+\cot(\theta_{12})\right]-A_6\cot(\theta_{11})-A_5\cot(\theta_{12})\right\}\\
&\qquad\qquad =\frac{\mu_0}{3}\sum_{i=1}^{6}J_i a_i. \qquad \text{(B.13.6.75)}
\end{aligned}$$

Solve for A_0 using the following "coupling" coefficient notations,

$$\begin{aligned}
w_1&=\frac{1}{2}\left[\gamma_1\cot(\theta_1)+\gamma_2\cot(\theta_4)\right]\\
w_2&=\frac{1}{2}\left[\gamma_2\cot(\theta_3)+\gamma_3\cot(\theta_6)\right]\\
w_3&=\frac{1}{2}\left[\gamma_3\cot(\theta_5)+\gamma_4\cot(\theta_8)\right]\\
&\vdots\\
w_6&=\frac{1}{2}\left[\gamma_6\cot(\theta_{11})+\gamma_1\cot(\theta_2)\right].
\end{aligned}$$

The final result is

$$A_0=\frac{\sum_{i=1}^{6}A_i w_i+\frac{\mu_0}{3}\sum_{i=1}^{6}J_i a_i}{\sum_{i=1}^{6}w_i}. \quad \text{(B.13.6.76)}$$

B.13.6.3 Boundary conditions.

Boundary conditions are divided into two classes:

1) Dirichlet - the value of the solution variable is specified at a boundary;
2) Neumann - the value of the normal derivative of the solution variable is specified at a boundary.

Only one of these boundary conditions can be qualified over a specified portion of the boundary.

Dirichlet boundary: Any boundary or portion of a boundary that is specified as Dirichlet has a value of the solution variable, A_z, at each mesh point on the boundary. The default value at these mesh points is zero. The user may input non-zero values, if necessary, for the problem definition.

Neumann boundary: In the case of the Neumann boundary condition, the only allowed value, at the mesh points along this boundary, for the normal derivative of the potential A_z is zero, which can be written

$$\frac{\partial A_z}{\partial n}\,\hat{\mathbf{e}}_{\mathbf{n}} = 0, \tag{B.13.6.77}$$

where $\hat{\mathbf{e}}_{\mathbf{n}}$ is the unit vector normal to the boundary.

Since the program, computationally speaking, uses only Dirichlet values at mesh points on the boundary, any boundary or boundary segment that is designated as Neumann must have an appropriate set of Dirichlet values calculated for these points. The procedure is the same one that is used for calculating the vector potentials A_z over the interior mesh points with the exception that the values of the reluctivities associated with those triangles lying outside the boundary are set equal to zero. Each Neumann boundary point must be surrounded by 6 triangles as in Fig. B.13.6.8. This is identical to the geometrical structure in Fig. B.13.6.3 with the exception that the path integral will not lie in the triangles outside the boundary nor along the boundary but will follow the contour defined in Fig. B.13.6.8.

It is required to show that the path integral along this Neumann boundary is zero. First divide the closed path of integration into three parts. This will include the boundary even though it will be later shown that the path integral along this segment is zero.

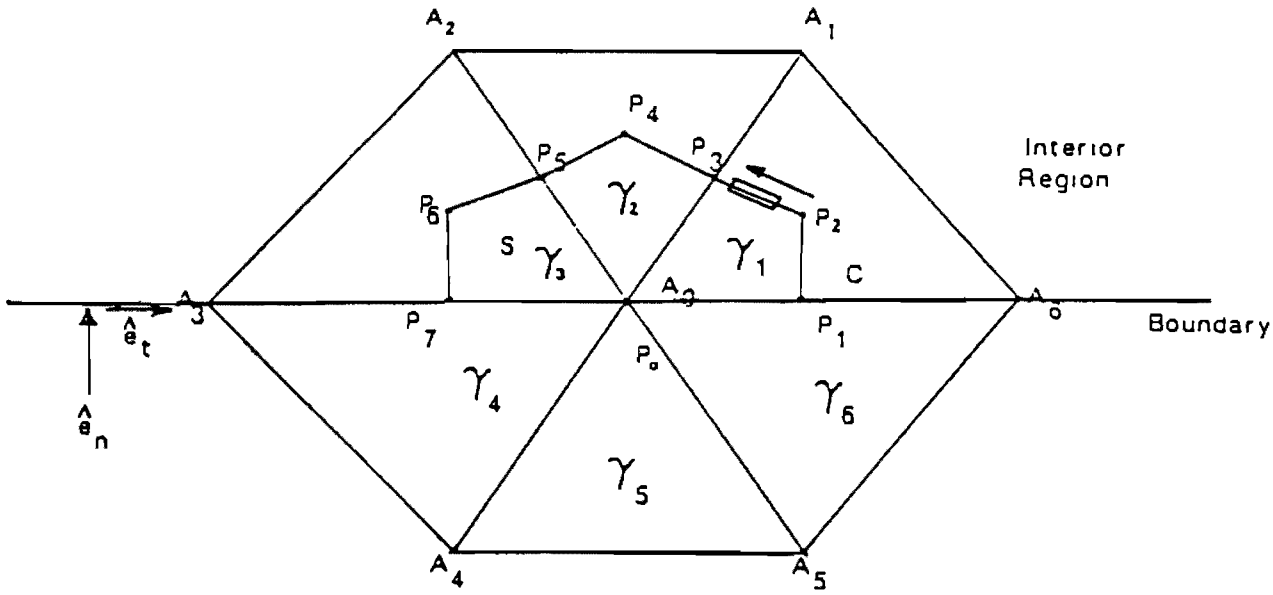


Figure B.13.6.8: Path of contour integration on a Neumann boundary is $C \equiv$ path $(p_1, p_2, \ldots, p_6, p_7)$.

$$\oint_{C_1} \gamma(|\mathbf{B}(\mathbf{r})|)\,\mathbf{B}(\mathbf{r}) \cdot d\mathbf{l} + \oint_{C_2} \gamma(|\mathbf{B}(\mathbf{r})|)\,\mathbf{B}(\mathbf{r}) \cdot d\mathbf{l} + \oint_{C_3} \gamma(|\mathbf{B}(\mathbf{r})|)\,\mathbf{B}(\mathbf{r}) \cdot d\mathbf{l} =$$

$$\mu_0 \int_S \mathbf{J}(\mathbf{r}) \cdot \hat{\mathbf{n}}\, da \qquad \text{(B.13.6.78)}$$

where C_1 is the path (p_0, p_1); C_2 is the path $(p_1, p_2, \ldots, p_6, p_7)$; and C_3 is the path (p_7, p_0).

Recalling that the vector functions $\mathbf{B}(\mathbf{r})$ and $\mathbf{A}(\mathbf{r})$ are 2-dimensional, we write

$$\mathbf{B}(\mathbf{r}) = B_n\,\hat{\mathbf{e}}_n + B_t\,\hat{\mathbf{e}}_t \qquad \text{(B.13.6.79)}$$

and

$$\mathbf{A}(\mathbf{r}) = A_n\,\hat{\mathbf{e}}_n + A_t\,\hat{\mathbf{e}}_t + A_z\,\hat{\mathbf{e}}_z \qquad \text{(B.13.6.80)}$$

where $\hat{\mathbf{e}}_n$ is the unit vector normal to the boundary, and $\hat{\mathbf{e}}_t$ is the unit vector tangent to the boundary.

Using Eq.(B.13.6.4), namely,

$$\mathbf{B}(\mathbf{r}) = \nabla \times \mathbf{A}(\mathbf{r}), \qquad \text{(B.13.6.81)}$$

and expanding the curl in this reference system, one obtains the result

$$\mathbf{B}(\mathbf{r}) = \left(\frac{\partial A_z}{\partial t} - \frac{\partial A_t}{\partial z}\right)\hat{\mathbf{e}}_n + \left(\frac{\partial A_n}{\partial z} - \frac{\partial A_z}{\partial n}\right)\hat{\mathbf{e}}_t + \left(\frac{\partial A_t}{\partial n} - \frac{\partial A_n}{\partial t}\right)\hat{\mathbf{e}}_z. \quad \text{(B.13.6.82)}$$

The reasoning is still valid, as in Eq. (B.13.6.7), that the components of $\mathbf{A}(\mathbf{r})$ in the two-dimensional plane are equal to zero, hence

$$A_n = A_t = 0 \quad \text{(B.13.6.83)}$$

The result of equating the coefficients of like unit vectors is

$$B_n = \frac{\partial A_z}{\partial t} \quad \text{(B.13.6.84)}$$

and

$$B_t = -\frac{\partial A_z}{\partial n}. \quad \text{(B.13.6.85)}$$

Since at a Neumann boundary

$$\frac{\partial A_z}{\partial n} = 0, \quad \text{(B.13.6.86)}$$

then

$$B_t = 0. \quad \text{(B.13.6.87)}$$

Hence along the Neumann boundary

$$\mathbf{B}(\mathbf{r}) \cdot d\mathbf{l} = B_t\, dl = 0. \quad \text{(B.13.6.88)}$$

The path integrals along a Neumann boundary segments will then be zero, that is,

$$\oint_{C_1} \gamma(|\mathbf{B}(\mathbf{r})|)\,\mathbf{B}(\mathbf{r}) \cdot d\mathbf{l} = \oint_{C_3} \gamma(|\mathbf{B}(\mathbf{r})|)\,\mathbf{B}(\mathbf{r}) \cdot d\mathbf{l} = 0.$$

B.13.6.4 Calculation of fields.

The magnetic induction, $\mathbf{B}(\mathbf{r})$, by definition is a 2-dimensional vector function lying in the xy-plane.

$$\mathbf{B}(\mathbf{r}) = B_x\,\hat{\mathbf{e}}_x + B_y\,\hat{\mathbf{e}}_y, \tag{B.13.6.89}$$

and is equal to the curl of the vector potential $\mathbf{A}(\mathbf{r})$

$$\mathbf{B}(\mathbf{r}) = \nabla \times \mathbf{A}(\mathbf{r}) = \frac{\partial A_z}{\partial y}\,\hat{\mathbf{e}}_x - \frac{\partial A_z}{\partial x}\,\hat{\mathbf{e}}_y. \tag{B.13.6.90}$$

Equating the coefficients of like unit vectors of the above equation with those of the magnetic induction $\mathbf{B}(\mathbf{r})$ from Eq.(B.13.6.89) give the required equations describing the components of the magnetic induction over each triangle in terms of the only non-zero component of the vector potential, A_z. Since the vector potential is linear over any triangle the magnetic induction, $\mathbf{B}(\mathbf{r})$ will be constant over that triangle. The partial derivatives

$$B_x = \frac{\partial A_z}{\partial y} \tag{B.13.6.91}$$

and

$$B_y = -\frac{\partial A_z}{\partial x} \tag{B.13.6.92}$$

can then be evaluated, analytically, for any triangle in the mesh by using the positions, (x, y), of the three vertices and their corresponding values for the vector potential component, A_z, at these points. The following equations are the results of evaluating Eq. B.13.6.37 and Eq. B.13.6.38 using Eq. B.13.6.47:

$$B_x = \frac{\partial A_z}{\partial y} = \frac{(A_2 - A_0)x_1 - (A_1 - A_0)x_2}{x_1 y_2 - x_2 y_1}, \tag{B.13.6.93}$$

$$B_y = -\frac{\partial A_z}{\partial x} = \frac{(A_2 - A_0)y_1 - (A_1 - A_0)y_2}{x_1 y_2 - x_2 y_1}. \tag{B.13.6.94}$$

The origin for each triangle in the above equations is taken at the coordinates $(x_0, y_0) = (0, 0)$ for convenience.

B.13.6.5 Computer algorithm using SOR method.

In the development of the algorithm that solves for the vector potential, $\mathbf{A}(\mathbf{r})$, the assumption was made that the reluctivity γ is constant over a triangle and not a function of the magnetic induction $\mathbf{B}(\mathbf{r})$, namely

$$\gamma_i(|\mathbf{B}(\mathbf{r})|) \simeq \gamma_i = constant. \tag{B.13.6.95}$$

This was an expedient which now must be dealt with. An iterative procedure is used by the program that handles this problem. It solves for the vector potential

one mesh point at a time while also modifying the reluctivities of the surrounding six triangles. Use Fig. B.13.6.2 as a reference for the procedure steps. These steps are as follows:

1. Given the estimates, both initialized and calculated, for the vector potential to be calculated, A_0, as well as those on the six surrounding mesh points, A_1, A_2, ..., A_5, A_6, calculate new estimates of the reluctivities γ_1, γ_2, ..., γ_5, γ_6 associated with these six triangles . Use these to the calculate new estimates of the coupling coefficients, ${w_1}^{new}$, ${w_2}^{new}$, ..., ${w_5}^{new}$, ${w_6}^{new}$. Now underrelax these values,

$$ {w_i}^{n+1} = q\,{w_i}^{new} + (1-q){w_i}^{n} \qquad \text{(B.13.6.96)} $$

   where $0 < q < 1$.

2. Calculate a new estimate of the vector potential, A_0, using a successive overrelaxation (SOR) technique,

$$ A_0^{k+1} = A_0^k + \omega \left[\frac{\sum_{i=1}^{3} A_i^k w_i^{n+1} + \sum_{i=4}^{6} A_i^{k+1} w_i^{n+1} + \frac{\mu_0}{3} \sum_{i=1}^{6} J_i a_i}{\sum_{i=1}^{6} w_i^{n+1}} - A_0^k \right] \qquad \text{(B.13.6.97)} $$

   where $0 < \omega < 2$. The superscript index k in the above equation is a sweep counter of estimates and is not incremented until a convergence criterion for the vector potential is met.

3. Repeat steps 1 and 2 until a convergence criterion is met.

This procedure is followed for each mesh point as it sequentially sweeps across the mesh.

B.13.6.6 Computer algorithm using the direct method.

A direct method was developed for the program PANDIRA. It is direct in the sense that estimates of the vector potential for all the interior mesh points are calculated simultaneously. The procedure is still overall iterative in that the reluctivities of each triangle must be first estimated before the vector potentials can be calculated directly and then these two steps repeated again until a convergence criterion is met. The procedure is as follows:

1. Given the estimates, both initialized and calculated, for the vector potentials on the boundaries and well as all the interior mesh points, calculate new estimates of the reluctivities, γ_i, for all the interior triangles. Use these to the calculate new estimates of the coupling coefficients, ${w_i}^{new}$. Now underrelax these values, using the relation

$$ {w_i}^{n+1} = q\,{w_i}^{new} + (1-q){w_i}^{n} \qquad \text{(B.13.6.98)} $$

where $0 < q < 1$.

2. Calculate new estimates of the vector potentials using a direct method that takes advantage of the block tridiagonal form of the matrix. The number of equations is equal to the number of interior points to be solved. The following equation, referenced to Fig. B.13.6.2, can be used to setup these linear algebraic equations which then can be solved simultaneously. The resulting equation is

$$A_0 = \frac{\sum_{i=1}^{6} A_i w_i^{n+1} + \frac{\mu_0}{3} \sum_{i=1}^{6} J_i a_i}{\sum_{i=1}^{6} w_i^{n+1}}. \tag{B.13.6.99}$$

3. Repeat steps 1 and 2 until a convergence criterion is met.

B.13.7 Table of Problem Constants in Numerical Order

Constant	Default	Symbol	Function
CON(1)	0	KPROB	KPROB is used by LATTICE to differentiate between a SUPERFISH and a POISSON-PANDIRA run. It is set to 0 or 1 in LATTICE, depending on the absence or presence of a character in the first position of the title line. KPROB = 0 means POISSON or PANDIRA. KPROB $\neq$ 0 means SUPERFISH.
CON(2)	none	NREG	Number of regions in the problem. Passed by AUTOMESH to LATTICE and from LATTICE to POISSON or PANDIRA. Note: NREG must be less than 32.
CON(3)	none	LMAX	Number of points in the L (vertical) direction in the logical mesh. Determined in LATTICE.
CON(4)	none	KMAX	Number of points in the K (horizontal) directon in the logical mesh. Determined in LATTICE.
CON(5)	none	IMAX	IMAX = KMAX + 2
CON(6)	-2	MODE	Option indicator for permeability in matter. MODE = -2 indicates "infinite" permeability (iron); MODE = -1 indicates permeability is finite, constant and defined by CON(10); MODE = 0 indicates permeability is finite, not constant, and the values will be taken from the internal table or tables supplied by the user. Mode = 0 can also be used to read in up to 4 values of constant $\gamma = 1./\mu$. See Chap. B.5 for more information.
CON(7)	1.0	STACK	Stacking or fill factor for iron regions. The default value of STACK will be overwritten for regions with MAT > 1 when tables are entered by changing CON(18). See Chap. B.5
CON(8)	1.0E+15	BDES	A flag for changing the value of the magnetic field B at location (KBZERO = CON(40), LBZERO = CON(41)). If BDES is not equal to its default value, the electric current factor XJFACT = CON(66) will be adjusted so that $\|B\|$ = BDES within a tolerance XJTOL = CON(67). If BDES is left at its default value, no adjustment is made.
CON(9)	1.0	CONV	Conversion factor for length units. Default units are centimeters. Set CONV equal to the number of centimeters per unit desired. CONV must be be changed in LATTICE.
CON(10)	0.004	FIXGAM	The value of γ = 1/(relative permeability) when the user has set CON(6) = MODE = -1. FIXGAM is also used to initialize a table for MODE = 0.

Table of Problem Constants in Numerical Order (cont.)

Constant	Default	Symbol	Function
CON(11)	0	NAIR	Number of "air" points. The default is an initial value. LATTICE counts the number of mesh points in the air regions of the magnet and records the value in CON(11). The user has no control of this CON.
CON(12)	0	NFE	Number of "iron" points. The default is an initial value. LATTICE counts the number of mesh points in the iron regions of the magnet and records the value in CON(12). The user has no control of this CON.
CON(13)	0	NINTER	Number of interface points. The default is an initial value. An interface point is a point whose nearest neighbors are a mixture of air points and iron points. See CON(11) and CON(12). The user has no control of this CON.
CON(14)	none	none	Not used in POISSON or PANDIRA problems.
CON(15)	none	NPINP	Total number of points in the problem. NPINP = NAIR + NFE + NINTER + NBND + NSPL. The user has no control of this CON.
CON(16)	0	NBND	Number of Dirichlet boundary points. Default is an initial value. LATTICE counts these points and stores the number in NBND. The user has no control of this CON.
CON(17)	0	NSPL	Number of points held at special fixed potential values. Default is an initial value. POISSON counts these points and stores the number in NSPL. The user has no control of this CON.
CON(18)	0	NPERM	If positive, this is the number of permeability tables to be read in as data by POISSON or PANDIRA. If negative, it is the number of stacking factors to be used with the internal permeability table or up to 4 different γ's if CON(6) = MODE = −1. After entering a nonzero value of NPERM, the action of the code is complex. See Sec. B.5.4 for details.
CON(19)	0	ICYLIN	A flag indicating the coordinate system to be used. ICYLIN = 1 indicates cylindrical coordinates using (horizontal, vertical) = (R, Z). Note that these axes are interchanged relative to those used in SUPERFISH. ICYLIN = 0 indicates two-dimensional (X,Y) coordinates. CON(19) must be changed in POISSON, PANDIRA, MIRT, or earlier.

Table of Problem Constants in Numerical Order (cont.)

Constant	Default	Symbol	Function
CON(20)	0	INPUTA	The number of special fixed potential values to be read in as data by POISSON or PANDIRA. When INPUTA is set, the code will write "NUMBER OF FIXED POTENTIAL VALUES TO BE READ IN (INPUTA)." The user is now expected to enter INPUTA number of lines of the form: "K L POT" where (K,L) are the logical mesh coordinates of the point and POT is the value of the potential in the appropriate units for the problem.
CON(21)	0	NBSUP	An indicator for the type of boundary conditions on the upper boundary. NBSUP = 0 indicates a Dirichlet boundary condition, which means magnetic field lines are parallel to the boundary line. NBSUP = 1 indicates a Neumann boundary condition, which means that the magnetic field lines are perpendicular to the boundary line. The default value that was passed by AUTOMESH is shown. AUTOMESH will pass the other value if IBOUND on the REG input line is used. See Sec. B.3.3.
CON(22)	1	NBSLO	An indicator for the type of boundary condition on the lower boundary. See CON(21) for description. The AUTOMESH default is shown. (The default from LATTICE is 0.)
CON(23)	0	NBSRT	An indicator for the type of boundary condition on the right boundary. See CON(21) for description.
CON(24)	0	NBSLF	An indicator for the type of boundary condition on the left boundary. See CON(21) for description.
CON(25)	0	NAMAX	Number of elements in the GTU and GTL vectors. The user has no control of this CON.
CON(26)	0	NWMAX	Number of points for recalculating couplings. The user has no control of this CON.
CON(27)	0	NGMAX	Number of points for recalculating gammas. User has no control of this CON.
CON(28)	0	NGSAM	Number of points for recalculating gammas with NM6 = NM1. The user has no control of this CON.
CON(29)	0	LIMTIM	An indicator used to check the remaining time in the run. Deactivated by comment in both POISSON and PANDIRA.

Table of Problem Constants in Numerical Order (cont.)

Constant	Default	Symbol	Function
CON(30)	100 000	MAXCY	Maximum number of iteration cycles. In POISSON the default is as shown; in PANDIRA the default is 20.
CON(31)	0	IPRFQ	The print frequency during POISSON interation cycles. IPRFQ = 0 indicates that the iteration information will be printed only on the first and last cycle. Input values of IPRFQ must be integer multiples of IVERG = CON(87).
CON(32)	0	IPRINT	An indicator for print options: IPRINT = −1 in LATTICE writes the (X, Y) coordinates of mesh points to OUTLAT. IPRINT = 1 (or any odd integer) writes a map of the solution matrix A into OUTPOI or OUTPAN. If IPRINT = 2, POISSON constructs and writes the gamma vs B table to OUTPOI; POISSON and PANDIRA write a map of $\|B\|$ in the iron triangles. IF IPRINT = 4, POISSON and PANDIRA write field components (B_x, B_y) in the iron region triangles into OUTPOI or OUTPAN. Any combination of these three write options is available by summing the IPRINT values. For example, IPRINT = 7 (1 + 2 + 4) will give all three options.
CON(33)	none	none	Not used in POISSON or PANDIRA,
CON(34)	−1	INACT	An indicator to allow the user to interact with the iteration during POISSON or PANDIRA. If INACT $\geq$ 1, the calculation is stopped at intervals and the user is asked to type: "GO", "NO", or "IN". If "GO", iteration continues; if "NO", iteration stops and final results are written; if "IN", user is asked for new values of CON's.
CON(35)	0	NODMP	An indicator controlling the write to TAPE35. If NODMP = 0, a dump is written; if NODMP $\neq$ 0, no dump is written.
CON(36)	0	IRNDMP	Not used in POISSON or PANDIRA. In MIRT IRNDMP = 1 causes the code to read from two dumps on TAPE35 when doing either gamma = 0 or gamma finite optimizations.
CON(37)	1	MAP	A parameter in the conformal transformation W = Z**MAP/(MAP*RZERO **(MAP-1). In LATTICE this will cause a transformation of the current density; in POISSON or PANDIRA the permeability is transformed and the fields are calculated in both the original and transformed geometries. For more information on the use of conformal transformations, see Sec. B.13.4.

Table of Problem Constants in Numerical Order (cont.)

Constant	Default	Symbol	Function
CON(38)	0.0	XORG	The real part of the complex number Z0 used to specify the origin in the polynomial expansion for the magnetic potential A(X,Y) = Re[$\Sigma C_n * (Z - Z0) * *n$]. The dimension of CON(38) is centimeters; values are automatically multiplied by CON(9) = CONV to get XORG and YORG. Note: For cylindrical coordinates, CON(38) is always zero. For more information on this, see Secs. B.5.3 and B.13.3.
CON(39)	0.0	YORG	The imaginary part of the complex number Z0. See CON(38) for further description.
CON(40)	1	KBZERO	The K logical coordinate of the point at which BDES = CON(8) is specified.
CON(41)	1	LBZERO	The L logical coordinate of the point at which BDES = CON(8) is specified.
CON(42)	1	KMIN	The lower K bound of the region in which the field and its gradient are to be calculated at each mesh point.
CON(43)	KMAX	KTOP	The upper K bound of the region in which the field and its gradient are to be calculated at each mesh point. If the user prefers to specify physical limits to the region, then KTOP is used to determine DX. See CON(54) through CON(57).
CON(44)	1	LMIN	The lower L bound of the region in which the field and its gradient are to be calculated at each mesh point.
CON(45)	1	LTOP	The upper L bound of the region in which the field and its gradient are to be calculated at each mesh point. If the user prefers to specify physical limits to the region, then LTOP is used to determine DY. See CON(54) through CON(57).
CON(46)	2	ITYPE	A constant specifying the symmetry of the problem and used to determine which C_n terms appear in the harmonic analysis of the complex potential function and whether C_n is real, imaginary or complex. For further discussion see Secs. B.5.3 and B.13.3. For problems with cartesian coordinates: ITYPE = 1 means no symmetry, ITYPE = 2 means midplane symmetry, ITYPE = 3 means elliptical aperture quad, ITYPE = 4 means symmetrical quad, ITYPE = 5 means skew elliptical aperture quad, ITYPE = 6 means

Table of Problem Constants in Numerical Order (cont.)

Constant	Default	Symbol	Function
CON(46)	(cont.)		symmetrical "H" dipole or elliptical aperture sexupole, ITYPE = 7 means symmetrical sextupole, ITYPE = 8 means elliptical aperture octupole, ITYPE = 9 means symmetrical octupole. For all of the above symmetry codes except ITYPE = 1 or 5, field lines are perpendicular to the X-axis. For ITYPE = 5, the X-axis is a field line. For vector potentials in cylindrical coordinates: ITYPE = 1 means no symmetry, ITYPE = 2 means midplane symmetry and the magnetic field lines are perpendicular to the R-axis. For scalar potentials in cylindrical coordinates: ITYPE = 1 means no symmetry, ITYPE = 2 means midplane symmetry and the lines of constant potential are perpendicular to the R-axis, ITYPE = 3 means midplane symmetry and the R-axis is a line of constant potential.
CON(47)	0.125	W2	The weight factor for the second nearest neighbors to a given lattice point in the mesh; W2ND = $\sqrt{W2}$. It is used in the determination of the C_n's in the harmonic expansion of the potentials. See Secs. B.5.3 and B.13.2.
CON(48)	1	ISECND	An indicator telling the program to use second nearest neighbors in determining the C_n's in the harmonic expansion of the potentials. ISECND = 0 means use first neighbors only; ISECND = 1 means use 1st and 2nd nearest neighbors.
CON(49)	0	NFIL	The number of current filament values to be read in. If NFIL $\neq 0$ the program will print "NUMBER OF CURRENT FILAMENT VALUES TO BE READ IN = (NFIL)". The user is expected to type NFIL lines of the form: K L CFIL. K and L are the logical mesh coordinates of the filament and CFIL is the current in the filament.
CON(50)	100000	IHDL	An indicator used in POISSON only to determine the number of cycles between making quasi-integrals of $\mathbf{H} \cdot d\mathbf{l}$ around the Dirichlet boundary. Making corrections to the solution matrix based on the value of this integral sometimes speeds the convergence, particularly for non-symmetrical "H" magnets.
CON(51)	0	NPONTS	In LATTICE this is the number of unknown relaxation points in the mesh. In POISSON and PANDIRA: NPONTS = NAIR + NINTER if MODE ≤ -2; NPONTS = NAIR + NINTER + NFE if MODE > -2

Table of Problem Constants in Numerical Order (cont.)

Constant	Default	Symbol	Function
CON(52)	0.001	OMEGA0 or OMEGA	A parameter used in calculating overrelaxation factors in LATTICE and POISSON.
CON(53)	25	IRMAX	An index for checking the progress of the relaxation process in LATTICE; not used in POISSON or PANDIRA.
CON(54)	0.0	XMIN	CON(54) through CON(57) are used to redefine the region over which the field and its gradient are calculated. If CON(54) through CON(57) and CON(42) through CON(45) are left at their default values, then the field and its gradient are calculated only along the horizontal axis. The user can define a larger region. XMIN is the lower X bound of the redefined region.
CON(55)	0.0	XMAX	The upper X bound of the region over which the field and its gradient are calculated. The quantities are calculated on a grid. The grid step DX is specified by DX = (XMAX − XMIN)/(KTOP − 1), where KTOP = CON(43).
CON(56)	0.0	YMIN	The lower Y bound of the region over which the field and its gradient are calculated. See CON(54).
CON(57)	0.0	YMAX	The upper Y bound of the region over which the field and its gradient are calculated. The quantities are calculated on a grid. The grid step DY is specified by DY = (YMAX − YMIN)/(LTOP − 1), where LTOP = CON(45). See CON(54).
CON(58)	0	IBOUT	A parameter controlling a phantom output file for POISSON or PANDIRA. At present, IBOUT = 1 will cause your run to abort because of a write to an unassigned file. IBOUT has been deactivated by a "c" in column 1 of statements involving IBOUT.
CON(59)	π	PI	PI is given to machine accuracy, namely, PI = 4. * ATAN(1.).
CON(60)	0.0	SPOSG	Total positive current at generation; used only in LATTICE. The user has no control of this CON.
CON(61)	0.0	SNEGG	Total negative current at generation; used only in LATTICE. The user has no control of this CON.
CON(62)	0.0	STOTG	Total current in problem at generation; used only in LATTICE. The user has no control of this CON.

Table of Problem Constants in Numerical Order (cont.)

Constant	Default	Symbol	Function
CON(63)	0.0	SPOSA	Total positive current in problem at solution. The user has no control of this CON.
CON(64)	0.0	SNEGA	Total negative current in problem at solution. The user has no control of this CON.
CON(65)	0.0	STOTA	Total current in problem at solution. The user has no control of this CON.
CON(66)	1.0	XJFACT	The factor by which all current and current densities (but not current filaments) will be scaled in POISSON or PANDIRA; XJFACT = 0.0 indicates the use of a scalar potential (no currents) for electrostatic problems.
CON(67)	1.0E-04	XJTOL	The tolerance allowed in the determination of XJFACT = CON(66) when given BDES = CON(8), the field at a specified point.
CON(68)	1.0	AFACT	The factor in MIRT by which scalar potentials are scaled when XJFACT = CON(66) = 0.0.
CON(69)	0.0	RATIO	The ratio of |BZERO|/(XJFACT = CON(66)) in POISSON for the solution in the air portion of the problem.
CON(70)	0	ICAL	An indicator for the type of formula to use in calculating the current associated with a point near the boundary of a coil. Can only be changed in LATTICE. ICAL = 0 means use normal area formula; ICAL = 1 means use angle formula. This latter formula is more accurate near coil boundaries.
CON(71)	0	NEGAT	A flag indicating a zero or negative area triangle in the mesh. This may occur in the relaxation of the mesh in LATTICE and NEGAT $\neq$ 0 will generate a diagnostic message.
CON(72)	0.0	SNOLDA	The old sum of delta squared's for air points. This number is calculated in POISSON at the end of IVERG = CON(87) cycles.
CON(73)	0.0	SNOLDI	The old sum of delta squared's for iron points. This number is calculated in POISSON at the end of IVERG = CON(87) cycles.

Table of Problem Constants in Numerical Order (cont.)

Constant	Default	Symbol	Function
CON(74)	1.9	RHOPT1	A flag to cause optimization of the parameter CON(75) = RHOAIR. If RHOPT1 is equal to the initial value of RHOAIR, then RHOAIR is automatically optimized during the iterations.
CON(75)	1.9	RHOAIR	The overrelaxation factor in POISSON for air and interface points and for iron points in problems with the permeability finite and constant. This factor is automatically optimized during the iteration if the initial value of CON(75) = CON(74) = RHOPT1.
CON(76)	none	RHOM1	RHOM1 = RHOGAM − 1., where RHOGAM = CON(78). The user has no control over this CON.
CON(77)	1.0	RHOFE	The over relaxation factor for iron points in problems with finite but variable permeability. Used in POISSON.
CON(78)	0.08	RHOGAM	The overrelaxation factor for the inverse permeability in problems having finite but variable permeability. Used in POISSON.
CON(79)	1.6	RHOXY	The initial value of the X and Y mesh over-relaxation factors in LATTICE.
CON(80)	1	ISKIP	The number of cycles between recalculation of inverse permeabilities during a problem with finite and variable permeability. Used in POISSON.
CON(81)	1	NOTE	A flag for determining the order in which the mesh points are relaxed. NOTE = 0 gives the order: air points, interface points, then iron points. <u>NOTE = 0 must be used for PANDIRA.</u> NOTE = 1 gives the order: (air + interface) points, then iron points.
CON(82)	none	BMAX	The maximum value of B in the triangles. Used in PANDIRA. Note: Subroutine PTABLE has an internal variable with the same name.
CON(83)	0	IABORT	An abort flag in LATTICE, POISSON and PANDIRA. If IABORT = 1, the run is stopped.
CON(84)	1.0E-05	EPSO	A parameter to test for convergence in the mesh generation. Used in LATTICE only.

Table of Problem Constants in Numerical Order (cont.)

Constant	Default	Symbol	Function
CON(85)	5.0E-07	EPSILA	A parameter to test for convergence of the potential solution of air and interface points and also iron points if the problem has finite and constant permeability. Used in POISSON.
CON(86)	5.0E-07	EPSILI	A parameter to test for convergence of the potential solution of iron points in a problem with finite but variable permeability.
CON(87)	10	IVERG	The number of cycles between convergence tests. The default value of 10 should not be altered when using the option to optimize the over-relaxation factor RHOAIR = CON(75). Used in POISSON.
CON(88)	1.0	RESIDA	The residual of the air points at each IVERG = CON(87) cycles. The user has no control of this CON. Used in POISSON.
CON(89)	1.0	RESIDI	The residual of the iron points at each IVERG = CON(87) cycles. The user has no control of this CON. Used in POISSON.
CON(90)	0	ICYCLE	The present iteration number; used in LATTICE, POISSON and PANDIRA. The user has no control of this CON.
CON(91)	0	NUMDMP	Present dump number for writing to TAPE35.
CON(92)–(99)	none	none	This set of eight words stores the title of the problem, which was read by LATTICE.
CON(100)	none	ITERM	A print control number for OUTPOI or OUTPAN. If ITERM = 0 and XJFACT = CON(66) = 0, the writing of DBZDR, XN, and AFIT to files OUTPOI or OUTPAN is suppressed.
CON(101)	0	IPERM	A flag in PANDIRA for the initialization of the vector potential. If IPERM = 0, the vector potential for a permanent magnet is initialized with the current vector called SOURCE, which is given a value when NFIL = CON(49) is not zero or when a coil region with current is input. If there are no real current sources in the problem, then set IPERM = 1 or any nonzero number. This will allow the code to initialize the potential.
CON(102)	$600\ 000\ 000_8$	IAMASK	A mask used in LATTICE, POISSON, and PANDIRA to isolate bits in certain words.
CON(103)	$2\ 000\ 000\ 000_8$	ISCAT	A mask used in LATTICE to isolate bits in certain words.

Table of Problem Constants in Numerical Order (cont.)

Constant	Default	Symbol	Function
CON(104)	$4\ 000\ 000\ 000_8$	IFILT	A mask used in LATTICE, POISSON and PANDIRA to isolate bits in certain words.
CON(105)	$100\ 000_8$	IDIRT	A mask used in LATTICE to isolate bits in certain words.
CON(106)	1.0	ETHAIR	A measure of the rate of convergence of the solution during the current cycle. Used in POISSON to calculate RHOAIR = CON(75).
CON(107)	1.0	ETAFE	The current rate of convergence in iron.
CON(108)	0	AROTAT or	AROTAT was an additional rotational angle added to ANGLZ = CON(115) in harmonic analysis. It has been deactivated in subroutine MINT.
		ICYSEN	IF ICYSEN = 0 in POISSON, the boundary integrals are not printed.
CON(109)	none	ITOT	ITOT = (KMAX + 2) * (LMAX + 2).
CON(110)	0	NTERM	The number of coefficients to be obtained in the harmonic analysis of the potential. See Section B.13.3.
CON(111)	0	NPTC	The number of equidistantly spaced points on the arc of a circle with its center at the origin, at which the vector potential is to be calculated by interpolation. The Fourier analysis of the vector potential at these points yields the coefficients in the harmonic analysis. The input value of NPTC should be approximately the number of mesh points adjacent to the arc.
CON(112)	0.0	RINT	The radius of the arc of a circle on which the vector potential is to be calculated by interpolation for use in the harmonic analysis. RINT should be less than the distance to the nearest singularity (the pole or coil) by at least one mesh space, i.e., the size of the side of a triangle.
CON(113)	0.0	ANGLE	The angle in degrees that defines the extent of the arc of a circle on which the vector potential is to be calculated by interpolation for use in the harmonic analysis. See CON(111).
CON(114)	0.0	RNORM	The aperture radius or other normalization radius used in the harmonic analysis of the vector potential. See Sec. B.13.3 for a discussion of the harmonic analysis.

Table of Problem Constants in Numerical Order (cont.)

Constant	Default	Symbol	Function
CON(115)	0.0	ANGLZ	The angle in degrees that defines the initial point of the arc of the circle on which the vector potential is to be calculated by interpolation for use in the harmonic analysis. ANGLZ is measured from the X-axis. See Sec. B.13.3 for complete discussion of harmonic analysis.
CON(116)	37_8	MASK37	A mask used in LATTICE, POISSON and PANDIRA to isolate bits in certain words.
CON(117)	$77\ 777_8$	MASK5	A mask used in POISSON and PANDIRA to isolate bits in certain words.
CON(118)	none	MAXDIM	The maximum allowed value of ITOT = CON(109).
CON(119)	none	NWDIM	NWDIM = MAXDIM/2, where MAXDIM = CON(118).
CON(120)	377_8	MASKC1	A mask for the eighth character in a word.
CON(121)	117400_8	MASKC2	A mask for the seventh character in a word.
CON(122)	none	TSTART	The wall clock starting time for the codes that contain TSTART.
CON(123)	0.0	TNEGC	The total negative current in the geometry obtained after a conformal transformation. It is equal to the total negative current in the original geometry. This CON must be entered in LATTICE.
CON(124)	0.0	TPOSC	The total positive current in the geometry obtained after a conformal transformation. It is equal to the total positive current in the original geometry. This CON must be entered in LATTICE.
CON(125)	1.0	RZERO	The scaling factor in the conformal transformation W = Z**MAP/ (MAP*RZERO** (MAP-1)), where MAP = CON(37). Normally RZERO is the aperture radius. This CON must be entered in LATTICE.

B.13.8 Table of Probcons in Alphabetical Order

AFACT	CON(68)
ANGLE	CON(113)
ANGLZ	CON(115)
BDES	CON(8)
BMAX	CON(82)
CONV	CON(9)
EPSILA	CON(85)
EPSILI	CON(86)
EPSO	CON(84)
ETAFE	CON(107)
ETHAIR	CON(106)
FIXGAM	CON(10)
IABORT	CON(83)
IAMASK	CON(102)
IBOUT	CON(58)
ICAL	CON(70)
ICYCLE	CON(90)
ICYLIN	CON(19)
ICYSEN or AROTAT	CON(108)
IDIRT	CON(105)
IFILT	CON(104)
IHDL	CON(50)
IMAX	CON(5)
INACT	CON(34)
INPUTA	CON(20)
IPERM	CON(101)
IPRFQ	CON(31)
IPRINT	CON(32)
IRMAX	CON(53)
IRNDMP	CON(36)
ISCAT	CON(103)
ISECND	CON(48)

Table of Probcons in Alphabetical Order (cont.)

ISKIP	CON(80)
ITERM	CON(100)
ITOT	CON(109)
ITYPE	CON(46)
IVERG	CON(87)
KBZERO	CON(40)
KMAX	CON(4)
KMIN	CON(42)
KPROB	CON(1)
KTOP	CON(43)
LBZERO	CON(41)
LIMTIM	CON(29)
LMAX	CON(3)
LMIN	CON(44)
LTOP	CON(45)
MAP	CON(37)
MASK37	CON(116)
MASK5	CON(117)
MASKC1	CON(120)
MASKC2	CON(121)
MAXCY	CON(30)
MAXDIM	CON(118)
MODE	CON(6)
NAIR	CON(11)
NAMAX	CON(25)
NBND	CON(16)
NBSLF	CON(24)
NBSLO	CON(22)
NBSRT	CON(23)
NBSUP	CON(21)
NEGAT	CON(71)
NFE	CON(12)
NFIL	CON(49)
NGMAX	CON(27)
NGSAM	CON(28)

Table of Probcons in Alphabetical Order (cont.)

NINTER	CON(13)
NODMP	CON(35)
NOTE	CON(81)
NPERM	CON(18)
NPINP	CON(15)
NPONTS	CON(51)
NPTC	CON(111)
NREG	CON(2)
NSPL	CON(17)
NTERM	CON(110)
NUMDMP	CON(91)
NWDIM	CON(119)
NWMAX	CON(26)
OMEGA0 or OMEGA	CON(52)
PI	CON(59)
RATIO	CON(69)
RESIDA	CON(88)
RESIDI	CON(89)
RHOAIR	CON(75)
RHOFE	CON(77)
RHOGAM	CON(78)
RHOM1	CON(76)
RHOPT1	CON(74)
RHOXY	CON(79)
RINT	CON(112)
RNORM	CON(114)
RZERO	CON(125)
SNEGA	CON(64)
SNEGG	CON(61)
SNOLDA	CON(72)
SNOLDI	CON(73)
SPOSA	CON(63)
SPOSG	CON(60)
STACK	CON(7)
STOTA	CON(65)

Table of Probcons in Alphabetical Order (cont.)

STOTG	CON(62)
TNEGC	CON(123)
TPOSC	CON(124)
TSTART	CON(122)
W2	CON(47)
XJFACT	CON(66)
XJTOL	CON(67)
XMAX	CON(55)
XMIN	CON(54)
XORG	CON(38)
YMAX	CON(57)
YMIN	CON(56)
YORG	CON(39)

Chapter B.14

References For Part B

1. A. M. Winslow, "Numerical Solution of the Quasilinear Poisson Equation in a Nonuniform Triangular Mesh," J. Comp. Phys., pp. 2149–72 (1967).

2. K. Halbach, "A Program for Inversion of System Analysis and Its Application to the Design of Magnets," Proc. 2nd Conf. on Magnet Technology, Oxford, England, July 10th – 14th, 1967, pub. by Rutherford Laboratory, p. 47.

3. K. Halbach, "Design of Permanent Multipole Magnets with Oriented Rare Earth Cobalt Materials," Nucl. Inst. and Meth., 169 (1980) 1.

4. J. A Stratton, Electromagnetic Theory, McGraw-Hill, New York (1941), p. 1.

5. J. A. Stratton, op cit., p. 11.

6. K. Halbach, "Application of Conformal Mapping to Evaluation and Design of Magnets Containing Iron with Nonlinear B(H) Characteristics," Nucl. Inst. and Meth., 64 (1968) 278.

7. K. Halbach, "Calculations Stray Field of Magnets with POISSON," Nucl. Inst. and Meth., 66 (1968) 154.

8. D. A. Lowther and P. P. Silvester, Computer-Aided Design in Magnetics, Springer-Verlag, Berlin (1986).

9. P. M. Morse and H. Feshbach, Methods of Theoretical Physics, McGraw Book Co. (1953) p. 676.

Chapter B.15

Index

AMAX 9–17
AMIN 9–17
Ampere's Law 1–3
ANGLE 5–8
ANGLZ 5–8
ANISO 5–13, 13–14
Anisotropic Electrostatics 13–15, 18
Anisotropic Magnetostatics 13–9, 17
artificial discontinuities 13–27
BCEPT 5–14
boundary conditions 1–7, 13–21
boundary, optimized 9–6
bump height 9–9
bumps 1–11
CFIL 5–17
change of polarity 13–28
coefficient matrix 9–1, 12
coercive force 1–4, 13–10
complex variables 13–19, 23
conformal transformation 9–1, 13–45
CONV 3–14
Convergence 5–9
CUMAX(J) 9–11
CUMIN(J) 9–11
CUR 3–14
CWORK 13–34
cylindrical symmetry 13–32
degree of nonlinearity 9–2
DEN 3–14
dielectric constant 1–4
Dirichlet 1–7
Dump numbers 5–1
DX 3–14
DY 3–14
easy-axis 13–10
electret 1–4
electrostatics 1–6, 5–5
electrostatic problem 13–32
electrostatic problems 13–34
energy 1–9
energy density 13–19
energy per unit length 13–21
energy/radian 13–23
EPSILA 5–9
EPSILI 5–9
Eta-air 9–17
Eta-iron 9–17
FAR Array 9–10
FAR(K,J) 9–7
FEMAX 9–4
FFIT 9–5
field derivatives 13–30
FIXGAM 5–12
force 1–9, 13–19, 34, 37, 39, 40
Fourier series 13–43
FUAIR(J) 9–11
FUIRN(J) 9–11
GAMPER 5–13, 13–14
Gauss's Law 1–3
geometry, cylindrical 1–5, 1–7
geometry, cartesian 1–4, 6
Gmax 9–17

hard-axis 13-10
Hard restraints 9-1
Harmonic analysis 13-42
harmonic functions 13-25
harmonic polynomials 13-26, 33, 43
HCEPT 5-14, 13-14,
IBOUND 3-14
ICYLIN 5-3
IHDL 5-8
INACT 5-11
IND(K,J) 9-7
INPUTA 5-17
internal table 5-12
IPRFQ 5-10
IPRINT 3-15, 4-1, 5-10
IPUNCH 9-3
IREG 3-15
ISECND 5-7
ISKIP 5-9
Isotropic Electrostatics 13-4, 8
Isotropic Magnetostatics 13-2, 6
ITFIT 9-5
ITRI 3-15
ITRI(M) 9-13
ITYPE 5-3, 9-9
ITYPE, code 5-4
IVERG 5-9
JSOFT(I) 9-12
JTYPE(J) 9-7
KBZERO 5-6
KMAX 3-15
KMIN 5-11
KPOLE 9-7
KREG1 3-15
KREG2 3-15
KTOP 5-11
LBZERO 5-6
least squares 9-1
LIMTIM 5-10
linear bump 9-8
LINX 3-16
LINY 3-16
LMAX 3-15
LMIN 5-11
logical mesh 1-8
LOOP 9-3
LPOLE 9-7
LREG 3-15
LREG1 3-15
LTOP 5-11
MAT 3-16
materials, anisotropic 1-4
materials, anisotrophy 1-5
materials, isotropic 1-4, 5
materials, permanent magnet 1-5, 4
MATRIX 9-3
MAXCY 5-10
MAXK(M) 9-13
MAXL(M) 9-13
Maxwell's equations 1-3
mesh, topologically 1-7
MINK(M) 9-13
MINL(M) 9-13
MODE 5-12
MPRINT 9-3
MSE 1-3
MTYPE 5-13
MTYPE(M) 9-13
μ-infinite 5-12
μ-finite-and-constant 5-12
μ-finite-but-variable 5-12
multipole fields 13-43
MUSE 9-3
NAIR 9-5
NCELL 3-16
NDRIVE 3-16
Neumann 1-7
NEW 3-18
NFIL 5-17
NFIT 9-5
NIRN 9-5
NODMP 5-11
NPAR 9-7
NPERM 5-12

NPOINT 3-16
NPOLE 9-7
NPTC 5-7
NREG 3-17
NT 3-18
NTERM 5-7
NUM 5-1
optimization 9-1
Optimization constants 9-2
OUTLAT 4-1
OUTMIR 9-14
Over-relaxation 1-8, 5-8
particular solution 13-24
PCNT 9-4
PCON 9-3
permanent magnet 13-21
permeability 1-4
PHAXIS 5-13
PHI(J) 9-10
physical mesh 1-8
polynomial, harmonic 1-9
reflection symmetry 13-28, 13-30
regeneration 9-12
relaxation parameter 9-3
reluctivity 1-4
remanent 13-10
Residual-air 9-17
Residual-iron 9-17
RHOAIR 5-8, 9-17
RHOFE 5-8, 9-17
RHOGAM 5-9
RHOPT1 5-8
RINT 5-7
RLX 9-4
RLXS 9-4
RNORM 5-8
rotational symmetry 13-27
smooth bump 9-8
SOFIT(I) 9-12
soft restraints 9-1, 12
SRSUM 9-4
STACK 5-12
stored energy 13-19
stress tensor 13-38
symmetry 13-23
symmetry, rotation 5-5
symmetry, reflection 5-4
TEST1 9-4
THETA 3-18
torques 1-9, 13-19, 36, 37, 40,
trimming 1-10
types of bumps 9-8
units 1-4, 1-9
W2ND 5-7
WFIT 9-5
WOFIT(I) 9-12
WORK 13-34
XJFACT 5-5, 6, 9-8, 17
XJTOL 5-6
XMAX 3-17, 5-11
XMIN 3-17, 5-11
XOA 5-13
XORG 5-6
XREG1 3-17
XREG2 3-17
YMAX 3-17, 5-11
YMIN 3-17, 5-11
YOA 5-13
YORG 5-6
YREG1 3-17
YREG2 3-17

Part C

RF CAVITY CODES — THE SUPERFISH GROUP

In 1976 Holsinger and Halbach[2] published a paper describing a new way of solving the generalized Helmholtz equation

$$\nabla^2\phi + k^2\phi = S$$

for the eigenvalues k and the eigenfunctions $\phi(x)$. The quantity S is called the source term. The applications were to rf waveguides in two dimensions and cylindrically symmetric rf cavities in three dimensions. The Helmholz equation is an elliptic partial differential equation, hence the same type of boundary conditions are required for its solution as for the generalized Poisson equation, namely, $\phi(x)$ or its derivative must be specified on all portions of the mesh boundary, but not both on the same portion.

The first step in solving the problem is to define physical regions internal to the cavity and overlay these regions with a logical triangular mesh, which is then deformed into the so-called physical mesh. The sides of the triangles in the physical mesh conform as closely as possible to the physical boundaries of the regions. These steps are accomplished using the programs AUTOMESH and LATTICE.

The next step is to solve the Helmholtz eigenvalue problem. This is done in the code called SUPERFISH. The solution gives either the TM or TE modes of the cavity depending on the boundary conditions. Given the solutions, the user usually wants auxiliary properties such as plots of electric field, transit time factors, power losses on the cavity walls, and sensitivity of the eigenfrequencies to small perturbations of the cavity structure. These auxiliary calculations are done in the postprocessors TEKPLOT and SFO. Each of these codes will be described below in some detail, but to begin with, we will summarize the basic theory.

Chapter C.1

SUMMARY OF THE BASIC THEORY

In the subsections that follow we will write down the basic forms of the Helmholtz equation for cylindrical and cartesian coordinates, list the definitions of some auxiliary quantities calculated by the post-processors and discuss the system of units used in the codes. This is followed by a short discussion of the numerical method used to solve the eigenvalue problem. In particular we discuss the significance of the drive point in obtaining a solution.

C.1.1 Equations for the TM and TE modes

The most common application of SUPERFISH is for finding the accelerating (TM) modes of a cylindrically symmetric accelerating cavity. It can be shown (see Sec. C.13.1) that Maxwell's equations take the form

$$-\frac{\partial H_\theta}{\partial z} - \frac{1}{c}\frac{\partial E_z}{\partial t} = 0 \tag{C.1.1.1}$$

$$\frac{1}{r}\frac{\partial}{\partial r}(rH_\theta) - \frac{1}{c}\frac{\partial E_r}{\partial t} = 0 \tag{C.1.1.2}$$

$$\frac{\partial E_r}{\partial z} - \frac{\partial E_z}{\partial r} + \frac{1}{c}\frac{\partial H_\theta}{\partial t} = 0 \tag{C.1.1.3}$$

$$\frac{1}{r}\frac{\partial}{\partial r}(rE_r) + \frac{\partial E_z}{\partial z} = 0 \tag{C.1.1.4}$$

where H_θ is the component of the magnetic field in the cylindrical direction, expressed in electric field units. The true magnetic field is $\sqrt{\epsilon/\mu}$ times H_θ .

and

$$c = \frac{1}{\sqrt{\epsilon\mu}} \tag{C.1.1.5}$$

The relative permittivity κ_e and the relative permeability κ_m are input parameters for each region, and they are used when the true magnetic field is required. See for example Table C.1.2.I below. The direction of the fields are illustrated in Fig. C.1.1.1.

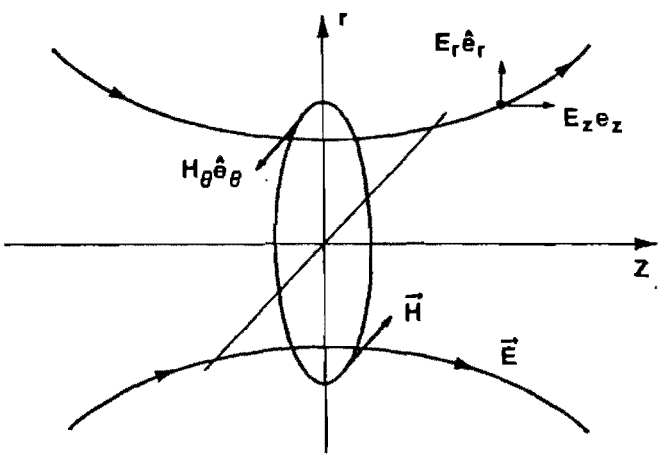


Figure C.1.1.1: Illustration of electric and magnetic field lines for the TM mode in a cylindrically symmetric rf cavity. The magnetic field lines are coming out of the plane of the paper above the z-axis and going into the paper below the z-axis.

In the TM mode r- and z-components of the magnetic field and the θ-component of the electric field vanish. It can also be easily shown that the equations for the deflecting (TE) modes of an rf cavity are very similar to those for the TM modes, namely,

$$-\frac{\partial E_\theta}{\partial z} - \frac{1}{c}\frac{\partial}{\partial t}(-H_r) = 0 \tag{C.1.1.6}$$

$$\frac{1}{r}\frac{\partial}{\partial r}(rE_\theta) - \frac{1}{c}\frac{\partial}{\partial t}(-H_r) = 0 \tag{C.1.1.7}$$

$$\frac{\partial}{\partial z}(-H_r) - \frac{\partial}{\partial r}(-H_z) + \frac{1}{c}\frac{\partial E_\theta}{\partial t} = 0 \tag{C.1.1.8}$$

$$\frac{1}{r}\frac{\partial}{\partial r}(-rH_r) + \frac{\partial}{\partial z}(-H_z) = 0 \tag{C.1.1.9}$$

The only differences are the interchange of E for H and the minus sign appearing before H_r and H_z. The output of SUPERFISH has no flag to tell the user which type of mode is being calculated. The column labels always display field components for the TM modes. It is up to the user to reinterpret these column headings when TE modes are being calculated. Generally speaking, the only difference between solving a problem for TE modes instead of TM modes is the boundary conditions. This is discussed further in Secs. C.3 and C.13.3 below.

We will work with the equations for the TM mode. If we assume that

$$H_\theta(r,z,t) = H_\theta(r,z)\cos\omega t \tag{C.1.1.10}$$

and define

$$k = \omega/c \tag{C.1.1.11}$$

Then Eqs. (C.1.1.1) through (C.1.1.4) can be reduced to the form

$$\frac{\partial}{\partial r}\left[\frac{1}{r}\frac{\partial}{\partial r}\left(rH_\theta(z,r)\right)\right] + \frac{\partial^2 H_\theta(z,r)}{\partial z^2} + k^2 H_\theta(z,r) = 0 \tag{C.1.1.12}$$

$$E_r(z,r,t) = -\frac{1}{k}\frac{\partial H_\theta(z,r)}{\partial z}\sin\omega t \tag{C.1.1.13}$$

$$E_z(z,r,t) = \frac{1}{kr}\frac{\partial}{\partial r}\left(rH_\theta(z,r)\right)\sin\omega t \tag{C.1.1.14}$$

Equation (C.1.1.12) is the Helmholtz eigenvalue equation, and the other two define the electric field components.

SUPERFISH also solves Maxwell's equations in two-dimensional cartesian coordinates. The main applications are to waveguides and cross sections of an RFQ accelerating cavity. It is assumed that the waveguide is infinitely long in the z-direction and the fields are independent of z. That is, the program solves for the cut-off wavenumber and fields. For Cartesian symmetry it is usually the TE mode that is of interest. The geometry of the fields is illustrated in Fig. C.1.1.2. For the mode shown, the magnetic field is coming out of the end of the cavity.

The equations solved by the code are

$$\frac{\partial^2 H_z(x,y)}{\partial x^2} + \frac{\partial^2 H_z(x,y)}{\partial y^2} + k^2 H_z(x,y) = 0 \tag{C.1.1.15}$$

$$E_x(x,y,t) = \frac{1}{k}\frac{\partial H_z(x,y)}{\partial y}\cos\omega t \tag{C.1.1.16}$$

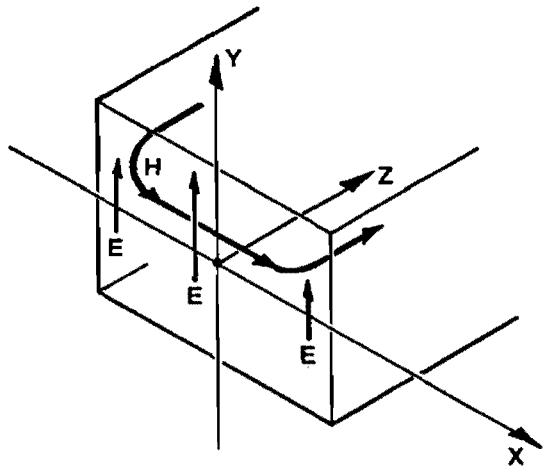


Figure C.1.1.2: Illustration of electric and magnetic field lines for the TE mode in a rf cavity having cartesian symmetry.

$$E_y(x,y,t) = -\frac{1}{k}\frac{\partial H_z(x,y)}{\partial x}\cos\omega t \qquad \text{(C.1.1.17)}$$

The eigenvalue k is still given by Eq. (C.1.1.11) and H_z is assumed to be proportional to cos ωt.

C.1.2 Auxiliary Quantities Calculated in SFO1.

Table C.1.2.I is a summary of formulas for auxiliary quantities calculated from the fields generated by SUPERFISH. The discussion following the table defines the symbols and describes the assumptions used in deriving the equations. Full derivations and references are given in Sec. C.13.2.

Table C.1.2.I Formulas for Auxiliary Quantities in Cylindrical Coordinates

1. Energy/Volume,

$$U = \frac{\epsilon}{2}\frac{\int_s [\ H_\theta(z,r)]^2\, r dr dz}{\int_s r dr dz}$$

2. Power Loss on Walls,

$$P_w = \pi\sqrt{\frac{k\rho}{2\mu^3 c^3}}\oint_c [H_\theta\,(z,r)]^2\, r dl$$

3. Power Loss on Stems,

$$P_s = 2\pi R_s\sqrt{\frac{k\rho}{2\mu^3 c^3}}\int_{r_1}^{r_2} [H_\theta\,(z_s,r)]^2\, dr$$

4. Average Accelerating Field,

$$E_0 = \frac{1}{L}\int_{-L/2}^{L/2} E_z(z, r=0) dz$$

5. Shunt Impedance,

$$Z_s = {E_0}^2 L/\,(P_w + P_s)$$

6. Quality Factor,

$$Q = \frac{\pi k\sqrt{\frac{\epsilon}{\mu}}\int_s [H_\theta(z,r)]^2\, r\ dr\ dz}{(P_w + P_s)}$$

7. Maximum Electric Field, E_{max} is found by searching.

8. Frequency Perturbation,

$$\frac{\Delta k}{k} = \frac{\int_{\delta v}\left\{[H_\theta(z,r)]^2 - [E_z(z,r)]^2 - [E_r(z,r)]^2\right\} dv}{2\int_v [H_\theta(z,r)]^2\, dv}$$

9. Transit Time Factors $(K = 2\pi/L)$:

$$T(K) = \frac{1}{E_0 L}\int_{-L/2}^{L/2} E_z(z, r=0)\cos Kz\ dz$$

$$TP(K) = -\frac{K}{2\pi}\frac{dT}{dK} = \frac{K}{2\pi E_0 L}\int_{-L/2}^{L/2} z E_z(z,0)\sin Kz\ dz$$

Table C.1.2.I (cont'd.) Formulas for Auxiliary Quantities in Cylindrical Coordinates

$$TPP(K) = \left(\frac{K}{2\pi}\right)^2 \frac{d^2T}{dK^2} = \left(\frac{K}{2\pi}\right)^2 \frac{1}{E_0 L} \int_{-L/2}^{L/2} z^2 E_z(z,0) \cos Kz \, dz$$

10. Coupling Coefficients ($K = 2\pi/L$) :

$$S(K) = \frac{2}{E_0 L} \int_0^{L/2} E_z(z,0) \sin Kz \, dz$$

$$SP(K) = \frac{K}{\pi E_0 L} \int_0^{L/2} z E_z(z,0) \cos Kz \, dz$$

$$SPP(K) = \frac{K^2}{2\pi^2 E_0 L} \int_0^{L/2} z^2 E_z(z,0) \sin Kz \, dz$$

The electric and magnetic fields are very nearly 90° out of phase. The energy stored in the field shifts sinusoidally back and forth between the magnetic and electric field, but remains a constant, independent of time. Therefore, the energy per unit volume U can be evaluated when the magnetic field is maximum and the electric field is zero, without loss of generality. The integral is over the zr-cross section of the cavity. The integration over the cylindrical angle ϕ has been carried out analytically.

If the electrical resistivity ρ of the cavity walls were zero, there would be no power loss and the electric field amplitude would go to zero at the wall. For walls with finite resistivity, the electric field penetrates the wall and causes an ohmic current to flow. The derivation of the equation for P_w is not trivial and involves some approximation.[12] The main approximation is that the field energy in the wall is much less than the field energy in the cavity. In cylindrical coordinates, the integral over the surface of the cavity is easily changed to a line integral around the cross section of the cavity in the zr-plane. The line element is called dl and the contour around the cavity cross section is called C.

When there are small cylindrical stems holding the drift tubes in the middle of the cavity, there is an additional power loss given by P_s. Figure C.1.2.1 illustrates a typical stem arrangement.

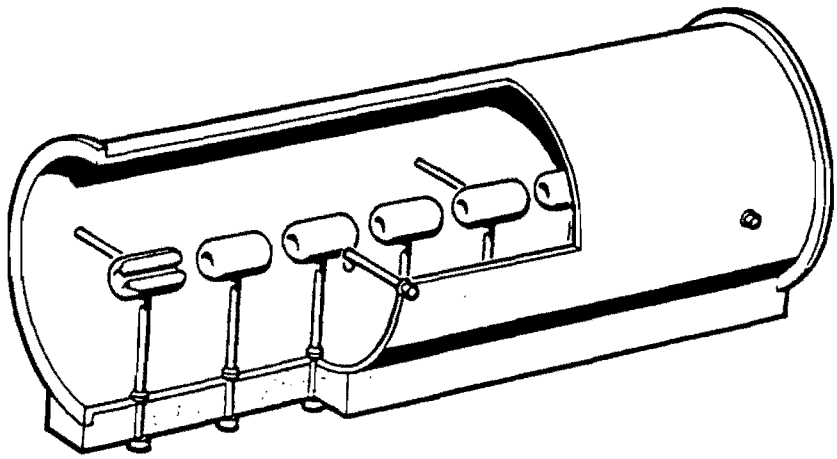


Figure C.1.2.1: Cutaway of a DTL tank showing stems holding the drift tubes in place.

In the formula for P_s, R_s is the radius of the stem; r_1 is the outer radius of the drift tube; r_2 is the radius of the tank holding the drift tubes; and z_s is the z-coordinate of the stem. The other parameters in the formula have been defined above.

The average accelerating field E_0 is defined as the integral of the z-component of the field along the beam direction. The rf cell has length L and is assumed to be symmetric with the center of the gap between drift tubes located at $z = 0$. Real cavities are slightly asymmetric but since the electric field is vanishingly small at the ends of the drift tubes, the approximation that the cavity is symmetric is a good one.

Shunt impedance Z_s has dimensions of Ohms/meter. It is a measure of excellence. The larger the accelerating field for a given power loss/unit-length, the better the accelerator.

The quality factor Q is a ratio of the energy stored in the cavity to the energy dissipated per radian of rf. A high Q is desirable if it means low power dissipation, but if it means large stored energy, then it is not desirable because it implies sensitivity to frequency errors. For pulsed systems high Q also implies a long time-constant for filling the cavity with rf field.[13]

The maximum electric field on a metal boundary E_{max} is important because this determines whether and where electrical breakdown will occur. The code must do a search since there is no way to calculate this quantity from a formula.

It is also useful to know how sensitive the resonant frequency is to errors in the

size of the cavity. The frequency perturbation is determined by using Slater's Perturbation Theorem,[14] which states that, for small perturbations, the relative change in resonant frequency caused by a perturbation that *decreases* the volume of the cavity by an amount δV is given by Formula 8, in Table C.1.2.I. In particular when the perturbation is a stem, the formula reduces to

$$\frac{\Delta k}{k} = \frac{\pi R_s^2 \int_{r_1}^{r_2} \left\{ E_z^2(z,r) + E_r^2(z,r) - [H_\theta(z,r)]^2 \right\} dr}{\int_V [H_\theta(z,r)]^2 \, dv} \tag{C.1.2.1}$$

The transit time factors T and TP come into the calculation of the transit time TT, which is defined by the relation

$$TT = \int_{-L/2}^{L/2} \frac{dz}{c\beta_z(z)} \tag{C.1.2.2}$$

where $c\beta_z(z)$ is the velocity of the synchronous particle going through the cavity. It can be shown that

$$TT \simeq \frac{L}{c\beta_{in}} \left\{ 1 - \frac{\delta}{\beta_{in}{}^3 \gamma_{in}{}^3} \left[\frac{1}{2} T\left(\frac{k}{\beta_{in}}\right) \sin\phi_s + TP\left(\frac{k}{\beta_{in}}\right) \cos\phi_s \right] \right\} \tag{C.1.2.3}$$

where $c\beta_{in}$ is the velocity of the synchronous particle at the entrance to the cavity; k is related to the resonant frequency of the cavity by Eq.(C.1.1.10); and ϕ_s is the rf phase when the particle is at the center of the gap. The quantity δ is given by the equation

$$\delta = \frac{eE_0 L}{mc^2} \tag{C.1.2.4}$$

where m is the rest mass of the particle. The transit time factor TPP and the coupling coefficients S, SP, and SPP are needed to describe the radial motion of the particle as it goes through the accelerating gap. For more details see Sec. C.13.2 or reference C.14.15.

C.1.3 Units

Before running SUPERFISH the user must define the cavity and give a starting frequency in AUTOMESH or LATTICE. The units of length are assumed to be centimeters but this unit can be changed to almost anything the user desires by the use of CONV in AUTOMESH = CON(9) in LATTICE. The quantity CONV is the number of centimeters per unit length desired. Angles entered into AUTOMESH or LATTICE are assumed to be in degrees. The estimated frequency ω is assumed to be in the units of megahertz (MHz).

All quantities appearing in the output of SUPERFISH will have properly identified units. It should be noted that quantities such as power loss P_ω, electric fields, etc., which are calculated in SFO1, are normalized by assuming that the average axial electric field E_0, defined in Table C.1.2.1 above, is 1 MV/meter.

C.1.4 Method of Solution Using a Drive Point

The method of solution is based on Stoke's Theorem in vector analysis. Equations (C.1.1.12) and (C.1.1.15) are special cases of the equation

$$\nabla \times (\nabla \times \mathbf{H}) - k^2\mathbf{H} = 0 \qquad \text{(C.1.4.1)}$$

where $\mathbf{H}$ is a function of (z, r) for cylindrical symmetry or (x, y) for cartesian symmetry. To solve this equation in the region of interest, we introduce an irregular, triangular mesh and derive a linear difference equation at each mesh point. Figure C.1.4.1 shows the neighborhood of a given mesh point.

We introduce a secondary mesh by drawing connecting lines between the "center of mass" of every triangle and the center of each of the six lines connecting "0" to its nearest neighbors. The mesh point is now surrounded by a unique 12-sided polygon. This secondary mesh of dodecagons covers completely the whole region of the problem. The difference equations for $\mathbf{H}$ are now obtained by integrating Eq. (C.1.4.1) over the area, one dodecagon at a time. This yields

$$\int_A \nabla \times (\nabla \times \mathbf{H}) \cdot d\mathbf{a} = \oint_C \nabla \times \mathbf{H} \cdot d\mathbf{l} = k^2 \int_A \mathbf{H} \cdot d\mathbf{a} \qquad \text{(C.1.4.2)}$$

If we assume that $\mathbf{H}$ can be approximated by a linear function of the variables (z, r) or (x, y) within every triangle, then $\mathbf{H}$ inside every triangle is uniquely determined by the values of $\mathbf{H}$ at the three corner-mesh points of the triangle. The integrals in Eq. (C.1.4.2) can be done analytically and the results expressed in terms of the value of $\mathbf{H}$ at the mesh point n and its six nearest neighbors, giving a

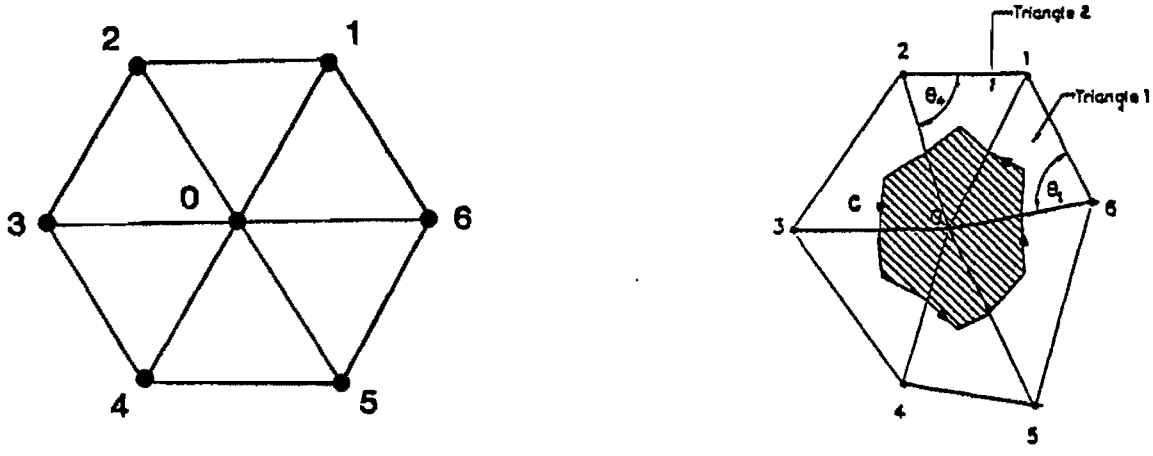


Figure C.1.4.1: Six nearest neighbors of a given point called "0" in a triangular mesh with secondary dodecagon. The contour C is the path of integration in Stoke's Theorem.

set of homogeneous equations of the form

$$\sum_{m=0}^{6} H_{nm}\left(V_m + k^2 W_m\right) = 0, \quad n = 1, \ldots, N \tag{C.1.4.3}$$

where N is the number *of mesh point.* This set of equations can be thought of as an N by N matrix of coefficients multiplying a column matrix containing the values of H at the mesh points. This is a very sparse matrix with at most six entries in each row. The equations then take the form

$$\sum_{m=1}^{N} A_{nm} H_m = 0, \quad n = 1, \ldots, N \tag{C.1.4.4}$$

where the single index m on H now runs now runs from 1 to N.

We would like to use the so-called direct method for solving this set of equations, but this cannot be done unless we can convert this set of homogeneous equations artificially into a set of inhomogeneous equations. This is accomplished by replacing one of the equations (say the equation for the p-th mesh point) by the simple equation

$$H_p = 1 \tag{C.1.4.5}$$

The prescription for making the set inhomogeneous is as follows: Every matrix element in row p and in column p is set equal to zero, except the (p,p) diagonal element is set equal to one. Column p is moved to the right side of the equation and its sign is changed except for the element in row p, which is set equal to one.

The direct solution then gives values of H at all other mesh points. If we take the original difference equation for the mesh point p, we can solve for H_p. This value will in general differ from one, except at resonance. This difference can be interpreted as being proportional to the current $I(k)$ necessary at that point to drive the cavity to the prescribed amplitude of $H = 1$ at the mesh point p. For this reason we refer to this mesh point as the "driving point." In essence, SUPERFISH finds the eigenvalues k by numerically finding the zeros of a functional proportional to $I(k)$.

In principle, the location of the drive point is arbitrary. In practice, the drive point should not be placed at a point where the value of the magnetic field in TM modes is nearly zero. This will result in a "current" $I(k)$ that is very flat and this makes it difficult to find the zero being sought. The algorithm for choosing the default location of the drive point usually leads to quick convergence, but for cavities with unusual shapes the default drive point may not be chosen optimally. If the user is having trouble getting the code to converge, he should use the option of moving the drive point to another location in the cavity.

Chapter C.2

Simple Example — Drift Tube Linac (DTL) Cavity

A drift-tube linac, in essence, is a long tube with a series of smaller tubes inside it. The smaller tubes are drift tubes. These are attached to the wall of the outer tube by stems. The spaces between adjacent drift tubes are the accelerating gaps. A section of this assembly consisting of one-half of a drift tube followed by a gap followed by another half drift tube forms an rf cell of the linac. When all cells are tuned to the same resonant frequency the entire assembly will also resonate at the same frequency. SUPERFISH is used to tune individual cells although it can also be used to study multiple cell structures.

The portions of the rf cell boundaries have special names. The inner radius of the enclosing tank is the cavity radius; the outer radius of the drift tube is the drift tube radius; and the angle the drift tube face makes with the axis of the drift tube is called the face angle. Figure C.2.1 identifies some of these features.

The cavity radius, the drift tube diameter, the nose radius, the corner radius, and the bore radius are all defined by the design of the linac. The main problem for any particular cavity is to choose the cavity length, face angle, and accelerating gap length in such a way that the accelerated particle gains just the right energy, surface electric fields are not too high, and the cavity has the design frequency of the linac. This usually requires several SUPERFISH runs, but in this example we will only give the results of the final run. Once we have found the final design, there are a number of other things that one wants to know about the cavity. For beam dynamics one needs the transit time factor and other related integrals. For the electrical design it is useful to know the power dissipated on the metal surfaces of the cavity, the shunt impedance, the quality factor Q, the stored energy in the cavity and the surface electric fields. From a perturbation analysis we can learn how small changes in the positions of surfaces will affect the resonant frequency.

The first step in running the problem is to set up an input file for AUTOMESH.

We shall assume that the cell is symmetric and therefore it is only necessary to calculate half of the cell, thus reducing the amount of input to AUTOMESH. Figure C.2.1 shows the outline of one-half of the cavity. By default SUPERFISH assumes that the cavity is a figure of revolution about the line labelled 1–10.

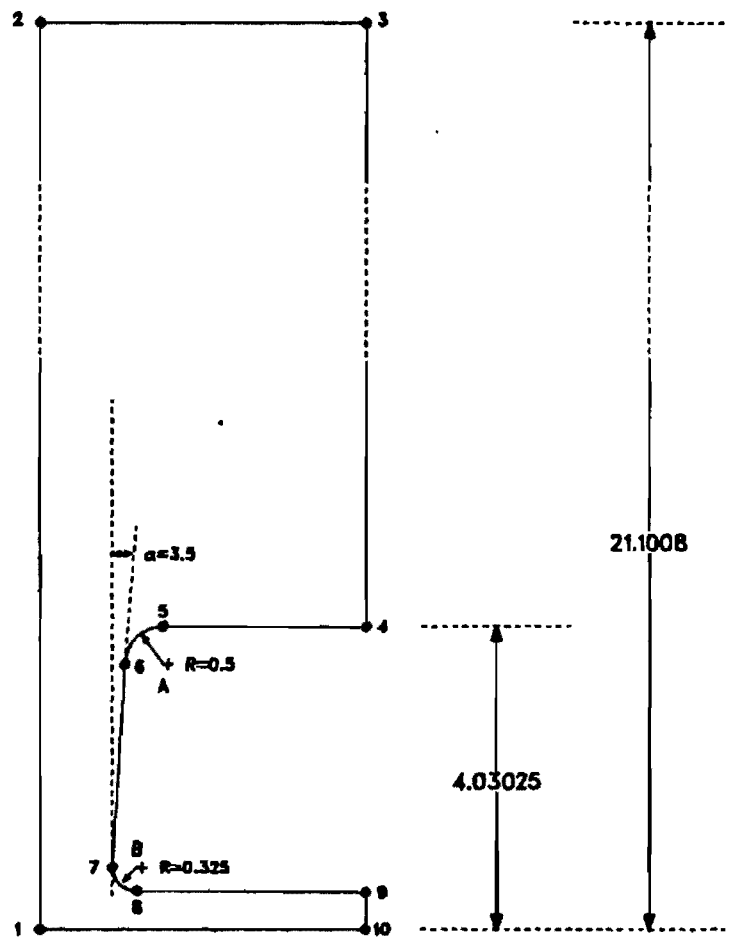


Figure C.2.1: Geometry of drift tube linac cell. The cavity radius is 21.1; the drift tube radius is 4.03; the nose radius is 0.325; the bore radius is the length of the line 9–10. The face angle is $\alpha = 3.5^0$.

The picture of the full cavity can be visualized by reflecting Figure C.2.1 in the vertical R-axis and rotating the whole figure around the horizontal Z-axis. The numbered dots indicate the boundary segment endpoints that enter into the data file. The points are numbered in the order given in the AUTOMESH input file shown in Figure C.2.2.

```
ssuperfish dtl test problem
 $reg nreg=1,dx=0.075,xmax=4.2843,ymax=21.1008,yreg1=5.,
 yreg2=7.,npoint=11$
 $po x=0.0      , y=0.0      $
 $po x=0.0      , y=21.1008  $
 $po x=4.2843   , y=21.1008  $
 $po x=4.2843   , y= 4.03025 $
 $po x=1.60454  , y= 4.03025 $
 $po nt=2 x0= 1.60454, y0= 3.53025, r=0.5, theta=176.5$
 $po x=0.93936  , y= 0.84484 $
 $po nt=2, x0=1.26375, y0= 0.825, r=0.325, theta=270.0$
 $po x=4.2843   , y= 0.5     $
 $po x=4.2843   , y= 0.0     $
 $po x=0.0      , y= 0.0     $
```

Figure C.2.2: Input to AUTOMESH for DTL cell.

The cavity dimensions are in centimeters. Figure C.2.1 is not drawn to scale and the dotted segments on the vertical lines indicate that the vertical scale has been foreshortened.

The first line of the input file is the title, which can be whatever you choose, except that the first column of the title must <u>not</u> be a blank. (A blank in column one signals AUTOMESH that the input file specifies a magnet problem input for POISSON or PANDIRA.) The lines following the first line are in standard FORTRAN namelist format with a blank space in the first column and the namelist items delimited by $. The second and third lines of the file make up the REG (region) namelist. The variables to be entered are the number of regions, NREG, and the approximate longitudinal step of the logical mesh, DX in centimeters. One can also enter the vertical step DY, but we have chosen to use the default value, which is about $0.87 * DX$. XMAX is the longitudinal extent. The YREG1 and YREG2 variables allow one to change the fineness of the vertical dimension of the mesh to reduce the number of mesh points in the problem. From $R = 5$ cm to $R = 7$ cm, the mesh is to be twice as coarse as between $R = 0$ and $R = 5$ cm. Between $R = 7$ cm and $R =$ YMAX the mesh is coarsened again by another factor of two over that between 5 and 7 cm. Since 5 cm is larger than the radius of the drift tube, this should not affect the accuracy of the solution very much. The variable NPOINT is the number of endpoint lines that are to follow. If there are n segments, there must be NPOINT $= n + 1$ endpoint data lines to close the boundary. There are other variables that can be set in the REG namelist, but we do not need them for this example and hence they will have their default values. (See Chaps. C.3 — C.8 for a full description of these variables.)

The PO input data lines describe the endpoints of curves that form the physical boundary of the region. The variable NT specifies the type of curve to be drawn from the previous point to the next point. Its default value is 1 (a straight line), which is why NT is not specified on the first 5 PO lines. The first PO line gives the (Z, R) coordinates of the starting point; the next four lines give the coordinates of the succeeding four points, each connected to the preceding point by a straight line. The sixth PO line gives the endpoint of a circular arc that starts at point 5 and ends at point 6. The line gives the coordinates of the center of the circle, (X_0, Y_0), the radius of the circle, $R = 0.5$, and the angular position of the endpoint, $THETA = 176.5^0$; NT is set to 2 to tell the program that it should draw a circle. The seventh line again gives the (Z, R) coordinates of a point reached by a straight line. The eighth PO line describes a point reached by a circular arc, and the remaining lines describe endpoints of straight line segments. The last PO line is the same as the first, and thus closes the boundary.

With the input data file prepared, we are ready to start the AUTOMESH run. This is done on the CTSS system by entering the name of the executable AUTOMESH file. Figure C.2.3 shows what happens at the terminal. Throughout the text when displaying terminal sessions, the information entered by the user will always be <u>underlined</u>. AUTOMESH asks for the name of the input file, which in this example is SFT1. AUTOMESH then executes and prints out some information on the logical boundary segment endpoints, and produces a file called TAPE73, which is used for input to LATTICE. AUTOMESH also produces a summary file OUTAUT. Usually there is no reason to consult OUTAUT, but it can be useful if something is wrong with the input file or if you want to know the parameters, either input or default, used in the run. It also contains a copy of the contents of TAPE73.

```
  automesh
 ?type input file name
?  sft1
 region no.  1
 logical boundary segment  end points
 iseg       kb    lb       kd    ld       ko    lo
    1        1     1        0    1        1    78
    2        1    78        0    1        1    93
    3        1    93        0    1        1   146
    4        1   146        1    0       58   146
    5       58   146        0   -1       58    93
    6       58    93        0   -1       58    78
    7       58    78        0   -1       58    63
    8       58    63       -1    0       22    63
    9       22    63       -1    0       16    56
   10       16    56       -1   -1       14    14
   11       14    14       -1   -1       18     9
   12       18     9        1    0       58     9
   13       58     9        0   -1       58     1
   14       58     1       -1    0        1     1
  stop
 automes  ctss time     .594    seconds
 cpu=     .211   sys=    .619  i/o+memory=    .364
  all done
```

Figure C.2.3: Log of interaction with AUTOMESH. The logical coordinates (kb, lb) correspond to the beginning points and the (ko, lo) to the end points of the segment. The long segment from (0, 0) to (0, 21.008) has been divided in 3 parts because of the mesh size changes (YREG1=5 and YREG2=7). The quantities (kd, ld) represent the incremental steps in going from (kb, lb) to (ko, lo). Essentially all they give are the direction of the step.

The next step is to execute LATTICE, which on CTSS is done by giving the executable file name LATTICE. Figure C.2.4 shows what happens. LATTICE asks

```
lattice
?type input file name
?  tape73
beginning of lattice execution
dump 0 will be set up for superfis
ssuperfish dtl test problem
?type input values for con(?)
?  s
 elapsed time  =    1.0 sec.
 0iteration converged
 elapsed time  =    3.8 sec.
 generation completed
 dump number 0 has been written on tape35.
 stop
lattice   ctss  time    4.337   seconds
cpu=    3.659     sys=    .034 i/o+memory=       .644

  all done
```

Figure C.2.4: Log of interaction with LATTICE.

for the input file name; TAPE73 was entered. The program then asks if the user wants to change any of the constants (CON(?)). Sometimes one wants to change the boundary conditions. Let us review these conditions. The upper boundary is a metal wall, which requires that electric field lines be perpendicular to the boundary. This is called a Neumann boundary condition, and is the upper boundary default value for SUPERFISH. The two side boundaries are vacuum, hence not conducting, but by symmetry the electric field must be normal to these boundaries; again these are default values for the side walls. The lower boundary is the axis and the field must be parallel to the boundary in the TM-mode. This is called a Dirichlet boundary condition, which is the SUPERFISH default for the lower boundary. Therefore there is no need to change the boundary constants. A review of the other possible input constants (See Chapters C.3 and C.5.) shows that the default values are appropriate for this run. The user types "S" (for "skip") to tell LATTICE to proceed. LATTICE completes execution by printing that it has written a block called DUMP 0 on file TAPE35.

At this point we could call SUPERFISH, but it would be wise to see if the half cavity looks the way we want it to. To do this the user executes TEKPLOT by typing TEKPLOT. Figure C.2.5 shows how TEKPLOT responds.

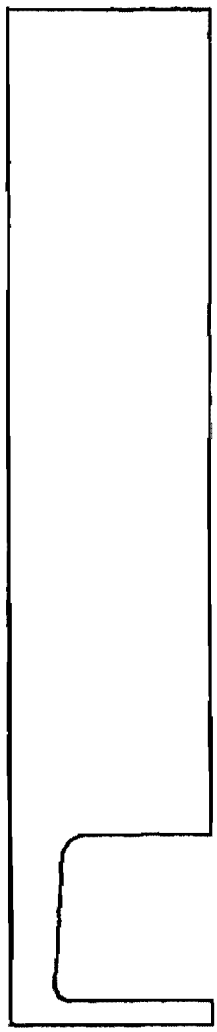
```
tekplot
 ?type input data- num, itri, nphi, inap, nswxy,
?  0,s
 input data
 num= 0   itri= 0   nphi=  0   inap= 0    nswxy= 0
 plotting prob. name = superfish dtl test problem  cycle = 0
 ?type input data-- xmin, xmax, ymin, ymax,
?  0.  4.5 0.  22.
 input data
 xmin=  0.000  xmax=  4.500  ymin=  0.000 ymax= 22.000
 ?type go or no
?  go
```

Figure C.2.5: Log of interaction with TEKPLOT.

TEKPLOT asks for five pieces of data, whose default values are all zero. To get just the cavity outline, the defaults are sufficient. The format of the reply is numbers separated by spaces or commas. If one wanted to enter only the first number, one could type the number and an "S" as shown in Figure C.2.5. If on the other hand one wants to change the fifth number, the first four numbers have to be entered first. Variations on this format are hopefully obvious.

The meaning of the five variables is as follows. NUM is the DUMP number. ITRI = 1 means that we want the triangular mesh plotted. NPHI is the number of equipotential lines to be plotted; a non-zero value only makes sense if NUM is not equal to zero. INAP has to do with the minimum and maximum values of the equipotential lines to be plotted and hence also is inappropiate for NUM = 0. The parameter NSWXY = 1 causes the x- and y-axes to be interchanged in the plot.

Line 3 on Figure C.2.5 indicates that we have set NUM = 0. TEKPLOT answers by typing out the input data it proposes to use and the problem title. It then asks for the boundary limits wanted on the graphical display. All that is needed are the rough values of the boundaries of the cavity. TEKPLOT echos this input and asks permission to proceed. If we answer no, the program goes back to the first input line and the user may change any of the previous variables in turn. On the CTSS system, if one answers: go, TEKPLOT erases the CRT screen and draws the cavity (See Figure C.2.6). To end TEKPLOT, press the carriage return key once. TEKPLOT asks for input data; type -1 s to exit TEKPLOT.



```
(?type input data - num, itri, nphi, inap, nswxy)
?  -1 s
tekplot ctss time 1.043 seconds
cpu= .209 sys= .027 i/O+memory= .808
```

```
prob.  name = superfish dtl test problem          freq = 0.000   ?
```

Figure C.2.6: Output of TEKPLOT showing the DTL cavity boundary.

We are now ready to run SUPERFISH; the executable file name on CTSS is FISH (See Figure C.2.7). SUPERFISH asks for an input file; we answer "**tty**" if we wish to enter data interactively from the keyboard. If one has prepared a set of input data in advance and put it in a file, then the name of this file should be entered at this time. In the interactive mode, SUPERFISH then asks for a DUMP number NUM. Then the program SUPERFISH types a couple of lines and asks for any changes in the CON array. This is the appropriate point to specify a guesstimate of the resonant frequency of the cavity. The next line tells the program to change CON(65) to 425. MHz and skip to the end of the array without making any other change.

```
fish
 ?type "tty" or input file name
?  tty
 ?type input value for dump number
?  0
 beginning of superfish execution from dump number 0
 prob.  name = superfish dtl test problem
 ?type input values for con(?)
?  *65 425.  s
 elapsed time = 2.7 sec.
cycle     hmin          hmax          residual
    0    0.0000e+00  0.0000e+00     1.0000e+00
    ----------------------------------------------------
                                          k**a = 7.9341e-03
                                           freq = 4.2500e+02
solution time =  20.993 sec.
    1   0.0000e+00  2.3276e+00   1.0000e+00
kfix = 58  lfix =146 delta1= -1.0127e-05 d1(k**2)= -2.7635e-07
    ----------------------------------------------------
                   using slope = -1 formula with rlx =1.0000
del k**2 = -2.7635e-07  k**2 = 7.9338e-03  freq = 4.2499e+02
solution converged in   1 iterations
elapsed time  =  24.0 sec.
dump number  1 has been written.
?type input value for dump num
?  -1s
stop
fish     ctss time    25.175     seconds
cpu=    4.186   sys=     .388   i/o+memory=   20.601
```

Figure C.2.7: Log of interaction with SUPERFISH on CRAY.

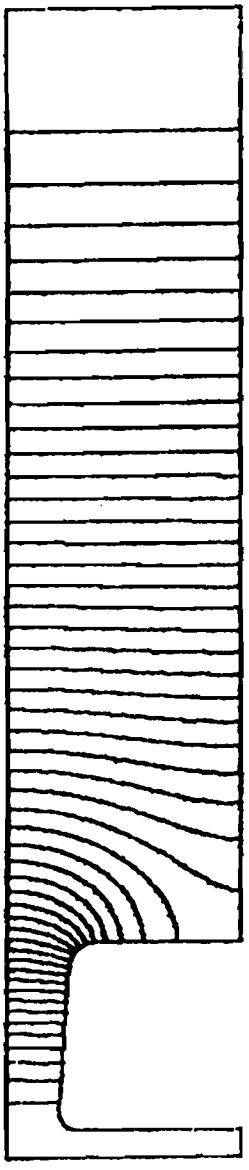
SUPERFISH then executes with the results shown in Figure C.2.7. The resonant frequency found by the program is very close to the guesstimate.

A useful check on the solution is to look at some of the field lines to be sure that we have the mode that we are looking for. This is done by exiting SUPERFISH and entering TEKPLOT as shown at the end of Figure C.2.7 and the beginning of Figure C.2.8. This time, when TEKPLOT asks for input to NUM, etc.,

```
tekplot
 ?type input data- num, itri, nphi, inap, nswxy
?  1 0 50 s
 input data
 num=  1   itri= 0   nphi=  50   inap= 0   nswxy= 0
 plotting prob. name = superfish dtl test problem    cycle = 1
 ?type input data-xmin, xmax, ymin, ymax,
?  0.  4.5 0.  22.
 input data
 xmin=   0.000  xmax=   4.500  ymin=  0.000  ymax= 22.000
 ?type go or no
?  go
```

Figure C.2.8: Log of interaction with TEKPLOT for field plot.

we type "1 0 50 s" to tell the program to use DUMP 1, not plot the triangular mesh, draw 50 equipotentials of rH_ϕ, which are parallel to the electric field lines, and to skip to the end of the input. Again, the rough boundaries are the same as before. TEKPLOT then draws Figure C.2.9.



```
(?type input data - num, itri, nphi, inap, nswxy)
?  -1 s
tekplot ctss time 5.188 seconds
cpu= 2.913 sys= .050 i/0+memory= 1.225
 all done
```

```
prob.  name = superfish dtl test problem freq = 424.993
```

Figure C.2.9: Output of TEKPLOT for electric field lines.

TEKPLOT is exited as before by typing "-1 s." The next step is to execute the SUPERFISH Output routine, SFO1. Figure C.2.10 shows the results of typing the

```
sfo1
 ?type "tty" or input file name
?  tty
 ?type input value for num
?  1
 beginning of sfodtl execution from dump number  1

 prob. name = superfish dtl test problem

 ?type input values for con(?)
?  *50 9 s
 ?type input values for iseg's(?)
?  4 -5 -6 -7 8 9 10 11 12 s
1
 superfish output summary    09:01:31   84/06/17
 problem name =
 cavity:length =  8.569 cm,    diameter     =    42.202 cm
 frequency (starting value = 426.000)       =   424.993 mhz
 beta= 0.1215            proton energy      =     6.999 mev
 normalize factor  (e0 = 1.0 mv/m) ascale   =     4827.3
 stored energy (for problem geometry)       =     0.0094 joules
 stored energy    (full cavity)             =     0.0188 joules
 power dissipation (for problem geometry)   =     521.12 watts
 power dissipation  (full cavity)           =    1042.24 watts
 t,tp,tpp,s,sp,spp = 0.874  0.038  0.005  0.394  0.053  0.004
 q = 48231       shunt impedance            =    82.213 mohm/m
 product z*t**2                     ztt     =    62.816 mohm/m
 magnetic field on outer wall               =      1281 amps
 maximum electric field on boundary         =     5.741 mv/s


iseg  zbeg    rbeg  zend   rend    emax       power      d-freq   d-freq
      (cm)    (cm)  (cm)   (cm)    (mv)        (w)       (delz)   (delr)
  4  0.000  21.101  4.284  21.101  0.00061   249.1 wall  0.0000  -1.3230
  8  4.284   4.030  1.604   4.030  2.73689   144.2 wall  0.0000   0.1057
  9  1.604   4.030  1.105   3.561  5.60369    22.0 wall  2.9432   2.5156
 10  1.105   3.561  0.939   0.845  5.30568    17.4 wall  6.8319   0.4179
 11  0.939   0.845  1.264   0.500  5.74078     0.0 wall  0.3529   0.1434
 12  1.264   0.500  4.284   0.500  0.72178     0.0 wall  0.0000   0.0021
                                  total     432.7 wall
 -5  4.284  21.101  4.284   7.000  0.63067    61.0 stem  0.9211   0.0000
 -6  4.284   7.000  4.284   5.000  0.61812    15.7 stem  0.2423   0.0000
 -7  4.284   5.000  4.284   4.030  0.32672    11.8 stem  0.3001   0.0000
                                  total      88.4 stem   1.4635
 ?type input value for num
?  -1s
 stop
 sfo1     ctss  time    1.904    seconds
 cpu=     .311    sys=    .052  i/o+memory=    1.541

  all done
```

Figure C.2.10: Log of interaction with SFO1 for DTL cavity.

executable file name SFO1. This program can be run from a prepared input file or interactively from the terminal as we have done in Figure C.2.10. After telling SFO1, when it asks, that we will input from the terminal and to use DUMP 1, we change CON(50) to 9, indicating that we want power and frequency shifts calculated for 9 sections of the cavity boundary. SFO1 then asks for the section numbers, ISEG's. The reply as shown in Figure C.2.10 has 9 numbers, some of which are negative. The reason for this is as follows. Remember that the REG line in the input to AUTOMESH asked for mesh changes at $R = 5$ and 7 cm. Inside the code, this caused the left and right boundaries to be divided at $R = 5$ and 7, thus adding two boundary segments on each side. There are now a total of 14 boundary pieces. The numbering of the segments (numbers in circles) is illustrated in Figure C.2.11.

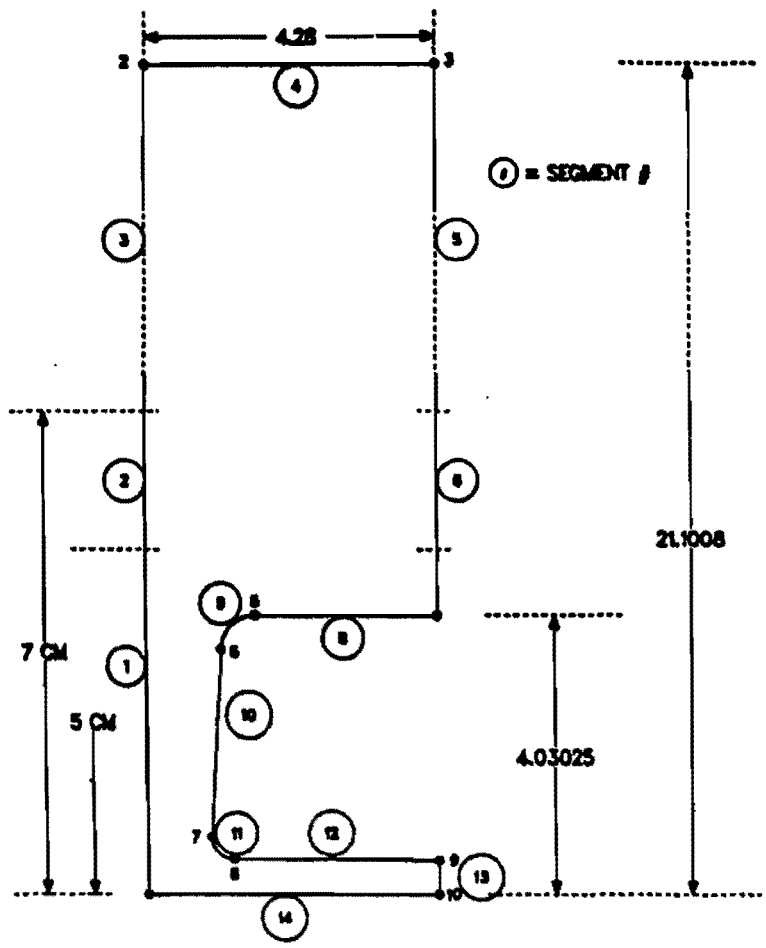


Figure C.2.11: Numbering of line segments for DTL problem.

The ISEG's input has the following interpretation. Section 4 is the outer wall; sections 5, 6, and 7 form the cavity boundary between the wall and the drift tube; sections 8, 9, 10, 11, and 12 are the drift tube. The minus signs before sections 5, 6, and 7 mean that we want SFO1 to calculate power and frequency shifts for these portions as if a drift tube stem were supporting the drift tube. If these segment numbers were positive, SFO1 would assume a solid wall.

After accepting this input, SFO1 executes, printing information about the properties of the cavity. Much of the information in Figs. C.2.10 and C.2.12 is self-explanatory. The quantity beta is the relativistic velocity ratio for a proton of the energy shown. SFO1 assumes that the total cell length corresponds to beta times the wavelength lambda of the electromagnetic mode for a proton and calculates beta from this information. The normalization factor tells us that SFO1 assumed that the drift tube linac was operating with an average axial field of 1 MV/m. If actual operation is to be at a different gradient, the stored energy and power must be adjusted accordingly. The normalization factor could also be changed by changing CON(100) at the beginning of the SUPERFISH run. The values in the line labeled "T, TP, TPP, S, SP, SPP" are the transit time factor and related quantities, which are explained in Section C.13.2. The magnetic field on the outer wall is constant for a TM(01) mode and is given in units of Amp/m. The maximum electric field (E_{max}) at the boundary is given as a quick check. If one wants more detailed values along the boundaries on which power was calculated, one should look into the file OUTSFO.

The final table printed by SFO1 gives the power dissipated on the segments and the frequency shifts that would result if the boundary in question were shifted in R or Z by 1 mm in a direction that would result in an increase in the volume of the cavity. The values given for the segments representing the stem correspond to a 1 cm radius rod. This radius can be changed by changing CON(81) at the beginning of the SFO1 run. The frequency shift (delz) for the stem is the number of MHz change for a 1 mm change in the radius of the stem.

The SFO1 output file OUTSFO contains a list of the problem CON's used, the value of E_Z at points on the axis, the values of several moment integrals of E_Z along the axis, and a copy of the terminal output. It also contains tables for each segment on which power and frequency shift calculations were requested. Each segment table consists of the K, L (logical mesh coordinates) and Z, R coordinates of the points on the path, the value of H_θ at the point and the value of $\mid E_Z \mid$ midway between points. At the end of the table appear the maximum electric field on the path, the power, and the estimated frequency shifts per millimeter shift in Z and in R. A portion of this output is shown in Figure C.2.12.

```
power to be calculated on the following segment
iseg    kb    lb      kd    ld      ke    le
  12    18     9       1     0      58     9

field along a specified path

   m   kpath  lpath      z          r            h-phi          abs(e)
                                                  (a/m)          (v/m)

   1      18      9     1.264     0.500      3.756e+01
                                                                 7.218E+05
   2      19      9     1.339     0.500      2.467e+01
                                                                 4.463e+05
   3      20      9     1.415     0.500      1.670e+01
                                                                 2.946e+05
   4      21      9     1.490     0.500      1.145e+01
                                                                 1.991e+05
   5      22      9     1.566     0.500      7.890e+00
                                                                 1.362e+05
   6      23      9     1.641     0.500      5.458e+00
                                                                 9.380e+04
   7      24      9     1.717     0.500      3.783e+00
                                                                 6.484e+04
   8      25      9     1.792     0.500      2.625e+00
                                                                 4.492e+04
   9      26      9     1.868     0.500      1.823e+00
                                                                 3.116e+04
  10      27      9     1.943     0.500      1.267e+00
                                                                 2.164e+04
  11      28      9     2.019     0.500      8.804e-01

  12  1.2637e+00  5.0000e-01  4.2843e+00  5.0000e-01  7.2178e-01  1.1383e-04
wall     0.0000e+00  4.1813e-03

   .
   .
   .
                                                                 1.300e-01
  41      58      9     4.284     0.500      3.240e-05
 emax on the above path =  7.2178e-01
 power on the above path =  1.1383e-04
             total power =  5.2112e+02
 freq= 424.993 mhz       delta-freq =  0.00418 mhz
                   delta-freq/freq  = 9.8386e-06
                   per mm. for delta r perturbation (volume added)
```

Figure C.2.12: A portion of the output of SF01 for the DTL.

Chapter C.3

SUPERFISH Input for LATTICE and AUTOMESH

Before the code AUTOMESH, the only way to enter data describing the physical geometry of the problem was through LATTICE. Although most users will use AUTOMESH to enter input, it is worthwhile understanding the structure of LATTICE input so that the structure of AUTOMESH makes sense. Furthermore there may be occasions when the user wishes to modify the output of AUTOMESH, which becomes the input to LATTICE. In this case, the user needs an understanding of the input to LATTICE.

C.3.1 Format-Free Input Routine (FREE I, RAY1, N1,..., RAYI, NI)

The authors of the Poisson Group programs developed their own format-free input routine to make it easier to enter data into all programs except AUTOMESH. The input into AUTOMESH is via the standard FORTRAN NAMELIST method.

The other Poisson Group programs expect most of the input file to be in a format that can be read by one of the following CALL statements:

```
CALL FREE(1,RAY1,N1)
CALL FREE(2,RAY1,N1,RAY2,N2)
CALL FREE(3,RAY1,N1,RAY2,N2,RAY3,N3)
```

where RAY1, RAY2, and RAY3 are arrays of length at least equal to N1, N2, and N3 respectively. In some cases the array length is variable. The CALL statements and dimensions are part of the program and hence not under the user's control, except for arrays of variable length. Section C.3.2 spells out in detail which of the three CALL statements are being used to enter data and what freedom the the user has.

The FREE format uses special characters to shorten input and to save array space. These characters and their functions are described below, and an example is given which uses all of them. When the forms in the left-hand column below are used for input, FREE interprets them as explained on the right. The case (upper or lower) for the letters R, S and C may be important on some computers.

*I X — This notation means store the number X, in the location (I) of the current array. If there are numbers following X, they are stored in locations (I+1), (I+2), etc.

XRN — This notation means store the number X, in N successive locations in the current array. A blank between X and R is optional. (Think of RN as being shorthand for "repeat N times.")

S — This symbol means skip the rest of the input to the current array and go to the next array, or end the read if the current array is the last array in the CALL FREE statement.

C — This symbol means count the number of values read into the current array and save the number as N1, N2 or N3 as appropriate. It also acts like S above. The purpose of this feature is to read in arrays of variable length.

Numbers may be either integers or floating point numbers. The latter can be in simple decimal format ±XX.XX or in scientific format ±X.XXXE±XX. The exponent must contain a plus or a minus sign. The plus sign in front of the mantissa is optional. The only other non-numeric characters allowed in the input field are the blank and the comma. Either the blank or the comma can be used to separate input values. Comments may follow the last S or C or required numbers on any input line.

The example below illustrates all the above features. A and B are dimensioned arrays, and K is a single variable.

```
Calling sequence:   N=100; CALL FREE(3,A,5,B,N,K,1)

input line:
   -3,4. +5.3E-2 R2 S *20 .1R10 C 13 THIS IS AN EXAMPLE.

This input produces the array values:
   A(1) = -3
   A(2) = 4.0
   A(3) = 0.053
   A(4) = 0.053
   A(5) = unchanged
```

```
B(1) thru B(19) = unchanged
B(20) thru B(29) = 0.1
N = 10
K = 13
```

One final important note. The FREE entry format requires all floating point numbers to have a decimal point. For example, ANGLE = 90 degrees must be entered as "90." in order to be recognized correctly. Leaving out the decimal point is a common beginner's mistake.

C.3.2 SUPERFISH Inputs to LATTICE

Logically the user would begin by making an input file to AUTOMESH, but to understand the reasoning behind AUTOMESH it is important to understand the structure of the input file to LATTICE. The input file for LATTICE is called TAPE73 for historical reasons.

The structure of the read statements in LATTICE is shown in Fig. C.3.2.1.

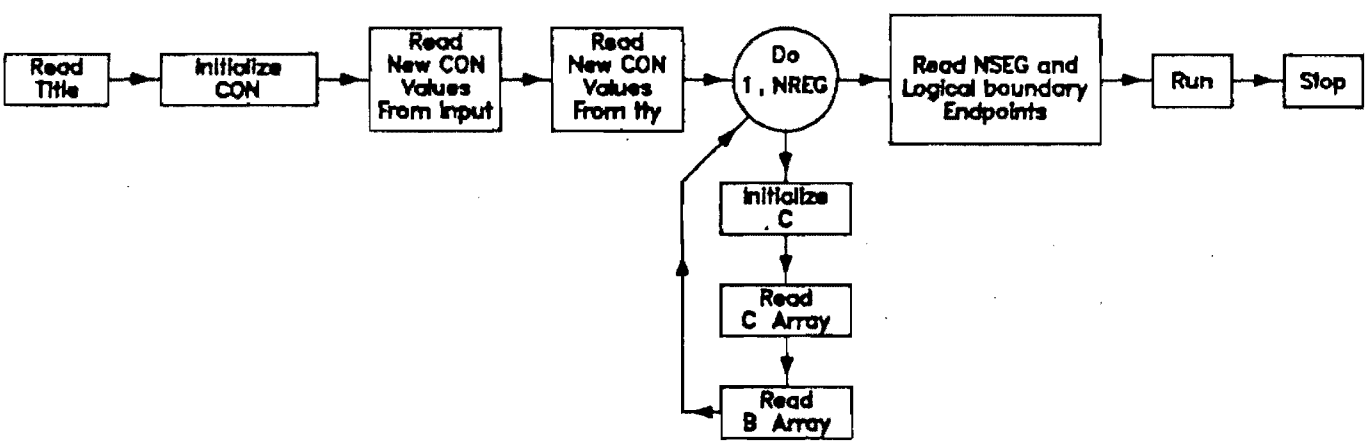


Figure C.3.2.1: Flow Diagram for Read statements in LATTICE for SUPERFISH.

The first data line can have anything in columns 2 through 80. If the first column is non-blank, then this data set is for a SUPERFISH problem. If the first column is blank, then this data set is for a POISSON or PANDIRA problem. The characters in columns 2 through 65 are stored and used in printouts for run identification. Because of the smaller word size on the VAX as compared with the CRAY, only columns 2 through 33 are available to the user for run identification when running on the VAX.

The reason that LATTICE distinguishes between SUPERFISH and POISSON runs is because LATTICE initializes the elements of the CON and C arrays with their default values. Some of the CON's and C's have different meanings for SUPERFISH as compared to POISSON or PANDIRA, and therefore the default values are different.

The next line (or several lines if needed) is reserved for making changes in the default values of the CON's (elements in the CON array). Only one CON absolutely must be changed; the user must tell the program the number of regions, NREG = CON(2). NREG is the upper limit of the DO-loop for the next read statements.

There are several other CON's that should be examined at this point. Many of these can *only* be changed in LATTICE if they are to have any effect at all on the problem. A list and brief description are given in Table C.3.2.I.

Other CON's can be changed from their default values at this time even though they have no effect in LATTICE. The changes will carry through to the output file TAPE35 and be available to the other programs when needed.

The format for entering the changes in the CON's is the special free format written for this program and described in Sec. C.3.1. The following example will illustrate the power of this format. Suppose we want to change CON(2), CON(9), and CON(21) through CON(24). The input line might read as follows:

```
*2 10 *9 2.54 *21 1 1 0 0 S
```

The * occurs before the number of the element to be changed. When several elements in a row are to be changed, only the first one need be indicated by a star. The final notation S means skip the rest of the elements in the array. This same free format is used to enter elements into the C and B arrays that come next.

Table C.3.2.I CON's that can only be changed in LATTICE

Number	Name	SUPERFISH Default	Description
CON(2)	NREG	None	Number of Regions in the problem geometry. Presently, NREG $\leq$ 31.
CON(9)	CONV	1.0	Conversion factor for the units of length in the problem CONV = 1.0 for centimeters CONV = 0.1 for millimeters CONV = 2.54 for inches.
CON(21) CON(22) CON(23) CON(24)	NBSUP NBSLO NBSRT NBSLF	1 0 1 1	Indicator for boundary conditions on the UPper, LOwer, RighT, and LeFt boundaries of the rectangular region defining the problem. A default value of 0 indicates a Dirichlet boundary condition, which means electric field lines in the TM mode are parallel to the boundary line; a default value of 1 indicates a Neumann boundary condition which means electric field lines are perpendicular to the boundary line in the TM mode.
CON(32)	IPRINT	0	An indicator for print options: IPRINT = 0 gives no printout; IPRINT = −1 causes LATTICE to write the (X, Y) coordinates of mesh points to OUTLAT on the CRAY or to LATTICE.OUT on the VAX. It is not often that this write to OUTLAT is of much use, but the option exists. This parameter can be changed again when running SUPERFISH. An odd value results in a print of the solution matrix on the OUTFIS file.
CON(36)	NSEG	none	The number of boundary segment endpoints for all regions. This number is normally computed in AUTOMESH and passed to LATTICE by the file TAPE73.
CON(79)	RHOXY	1.6	The starting over-relaxation factor for the irregular mesh generation. There is seldom a reason to change this number.

Table C.3.2.I (continued)
CON's that can only be changed in LATTICE

Number	Name	SUPERFISH Default	Description
CON(84)	EPSO	1.0E-5	The convergence criterion for mesh generation. There is seldom a reason to change this number, but if LATTICE has trouble converging, increasing EPSO might help.

There are six elements in the C array for each region; they are called "region constants". The first of these is an arbitrary region identification number. These numbers need not be in numerical order.

The second region constant is a material code that tells the program whether the region being defined is air or a material with different dielectric and/or magnetic properties. For SUPERFISH problems C(2) can take on the values 0 through 5. The value 0 means that the region is not in the problem. When C(2) = 0 for a region, no mesh is set up in this region. The treatment of the boundary points of such a region are determined by the region constant C(6) discussed below. The value C(2) = 1 indicates that the region is in the problem and is to be treated as an air or vacuum region. (This air value can be overwritten in SUPERFISH if two or more materials are used, by entering a value of MATER equal to 1. See NPERM=CON(18)) The values C(2) = 2 through 5 indicate regions with different values of relative permittivity κ_e and relative permeability κ_m. These values are read into SUPERFISH when NPERM=CON(18) is not equal to 0.

The third region constant, C(3), should be 0 in SUPERFISH problems unless the region being entered is a drive point. Then C(3) should be set to 1 and C(6) should be set to −1.

In SUPERFISH, the fourth region constant C(4) is used to define the length of cells in multicell cavities. Its use is a little complicated because the numbers being entered usually have nothing to do with the region being described. Figure C.3.2.2 shows a three cell cavity.

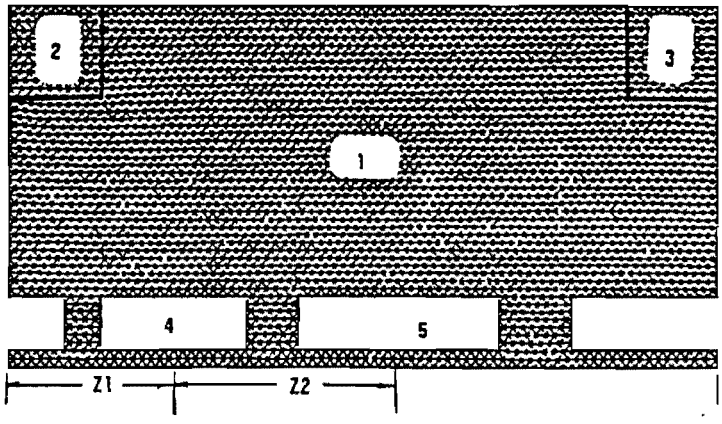


Figure C.3.2.2: Multicell rf cavity; Regions 4 and 5 have C(2) = 0; the Z1 and Z2 are C(4)'s.

Region 1 was originally the whole area. Out of this area we have defined several other regions using what is called "successive region data overwriting." The lengths of the first two cells are indicated on the figure by the symbols Z1 and Z2. The input value for C(4) associated with each region is given in Table C.3.2.II.

Table C.3.2.II.
Example of input for C(4) in Multicell Cavity Problems

Region	C(4)
1	-
2	Z1
3	Z2
4	-
5	-

The Z values are entered starting with the second region and continued until lengths of the first (N − 1) cells have been entered. The program knows the length of the whole multicell problem and therefore it calculates the length of the last cell. The default value of C(4) is 0.

The fifth region constant is an integer indicating the type of triangle to be used in defining the logical mesh for the region. There are three choices:

C(5) = 0 equal weight triangles (the default)
C(5) = 1 isosceles triangles
C(5) = 2 right triangles.

Equal weight and isosceles triangles are geometrically the same in a strictly uniform mesh. The difference comes in the relaxation process by which the *logical* mesh is deformed to the *physical* mesh. The distinction between the logical and physical mesh is discussed below when we discuss input to the B-array. The default is probably the best choice.

The sixth region constant is called a special boundary indicator. This constant is used for two purposes. It is used to indicate a user-defined drive point (C(6) = −1), and it is used to indicate the boundary condition on a boundary of the problem that does not coincide with the extreme rectangular logical boundary of the problem. When it is used for the latter purpose, it can have a value of 0 or 1. C(6) = 1 indicates a Neumann boundary condition, and C(6) = 0 indicates a Dirichlet boundary condition. The LATTICE default values for C(6) are C(6) = 0 for the first region, and C(6) = 1 for all other regions. This is not suitable for a SUPERFISH problem which usually needs C(6) = 1 for all regions. When AUTOMESH is used, it sets up C(6) = 1 for all regions by default. When the problem area does not fill the full rectangle defined by the first region, that is, when some region has C(2) = 0, then C(6) must be used to specify the desired boundary condition. Figure C.3.2.2 shows an example of a problem where the special boundary condition is required. The electric field must be perpendicular to the boundaries of regions 4 and 5 which are drift tubes. This requires Neumann boundary conditions, or C(6) = 1, which is the default value. This means that the user can skip the entry of this parameter in this case.

To avoid singularities in the solution of cavity modes, there must be a "drive point" in the cavity. This is a point region that can be put anywhere in the cavity by the user. There is a default location in the code, but sometimes it is in an inappropriate position, and the user must specify his own location. C(6) = −1 for this point region and C(3) = 1 will do the job. The code knows that the point region defined in the B-array is a drive point.

The B-array requires a list of logical and physical coordinates for the boundary of each region. The first stage of any problem using LATTICE is to set up a physical picture of the geometry and assign physical coordinates (X, Y) to points on the boundaries of the various regions. The second stage is to superimpose a regular triangular mesh on the whole area. Each mesh point can be assigned logical coordinates (K, L) as illustrated in Fig. C.3.2.3. One can now associate logical coordinates (K, L) with physical coordinates (X, Y) by matching the physical point with the closest logical point. LATTICE will use this association to distort the logical mesh into the so-called physical mesh, consisting of irregular triangles as illustrated in Fig. C.3.2.4.

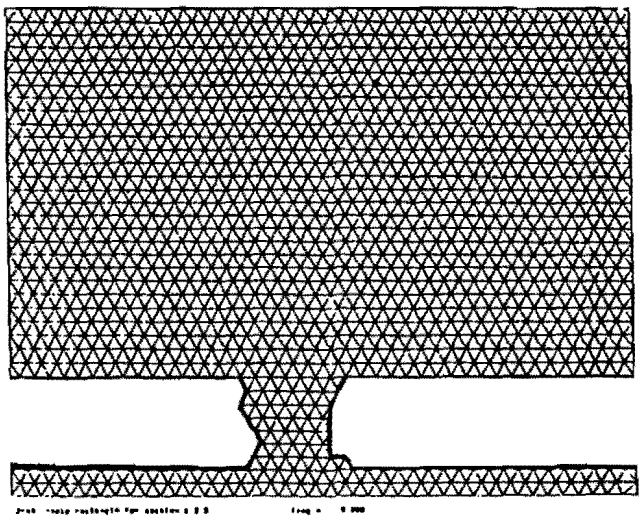


Figure C.3.2.3: A logical mesh for a drift tube linac.

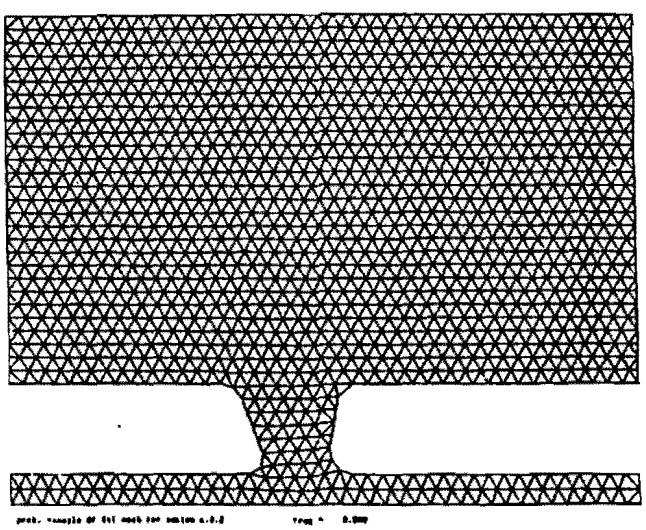


Figure C.3.2.4: The corresponding physical (relaxed) mesh for a drift tube linac.

The tediousness of constructing the logical mesh and making the association of coordinates is the main reason AUTOMESH was created. Since LATTICE connects points on boundaries with straight lines, one need only specify points at the ends of long straight segments, but approximating a circular arc with straight lines requires many points.

Once again the special free format is used to enter the values in the B array. The order of the input is K(1), L(1), X(1), Y(1), K(2), L(2), X(2), (Y2), etc. Note that the origin of coordinates in the logical mesh is (1, 1), not (0, 0). This input list is terminated with the free format character C, which stands for "Count the number of input values".

As indicated in Fig. C.3.2.1, the data groups for the C and B arrays are repeated for each region. The first region defines the largest rectangular region containing the problem and its boundary values must contain the largest K and L values in the mesh. The data for the second region redefines, or overwrites, the region constants for all mesh points belonging to that region. Data for each following region overwrites previously defined values in the same way. For example, suppose one wishes to define a drift tube region such as region 4 in Fig. C.3.2.2. If this region were given first in the input data and then followed by the larger region 1, then region 4 would be overwritten and this region would not exist in the problem.

Usually each region is closed, i.e., the data for the first and last boundary points of the region are identical. However, it is possible to specify data for a "point region" or a "line region". One purpose of this would be to define the physical coordinates of specific points, for example a drive point in a cavity problem. For point and line regions, the input values of C(2) and C(5) are not used in the program.

The final set of data LATTICE needs for a SUPERFISH problem is a table giving, for each boundary segment, the logical starting point, the change in K and in L from the starting point to the second point on the segment and the logical ending point of this segment. However, the first line of this data has a "special coded value" for the starting K logical coordinate. This special value of K is the number of boundary segments times 1000 plus the proper K coordinate.

Figure C.3.2.5 shows the LATTICE input for the cavity of Fig. C.3.2.2.

```
1 lattice input example
 *2  6 *21  1 0 1 1 *9    1.0000
 *36 12 *37  1 skip
   1  1       0.0000       0.0000  0  1 region
    1   1     0.0000     0.0000
    1   3     0.0000     1.0000
   11   3     5.0000     1.0000
   11  10     5.0000     4.0000
    1  10     0.0000     4.0000
    1  47     0.0000    20.0000
   79  47    39.0000    20.0000
   79  10    39.0000     4.0000
   67  10    33.0000     4.0000
   67   3    33.0000     1.0000
   79   3    39.0000     1.0000
   79   1    39.0000     0.0000
    1   1     0.0000     0.0000 coun
   2  2       0.0000      12.0000  0  1 region
    1  42     0.0000    18.0000
    1  47     0.0000    20.0000
    5  47     2.0000    20.0000
    5  42     2.0000    18.0000
    1  42     0.0000    18.0000 coun
   3  2       0.0000      24.5000  0  1 region
   75  47    37.0000    20.0000
   79  47    39.0000    20.0000
   79  42    39.0000    18.0000
   75  42    37.0000    18.0000
   75  47    37.0000    20.0000 coun
   4  0       0.0000       0.0000  0  1 region
   15   3     7.0000     1.0000
   15  10     7.0000     4.0000
   35  10    17.0000     4.0000
   35   3    17.0000     1.0000
   15   3     7.0000     1.0000 coun
   5  0       0.0000       0.0000  0  1 region
   39   3    19.0000     1.0000
   39  10    19.0000     4.0000
   61  10    30.0000     4.0000
   61   3    30.0000     1.0000
   39   3    19.0000     1.0000 coun
   6  1       1.0000       0.0000  0 -1 region
   79  47    39.0000    20.0000 coun
  28001       1      0      1      1      3
      1       3      1      0     11      3
     11       3      0      1     11     10
     11      10     -1      0      1     10
      1      10      0      1      1     47
      1      47      1      0     79     47
     79      47      0     -1     79     10
     79      10     -1      0     67     10
     67      10      0     -1     67      3
     67       3      1      0     79      3
     79       3      0     -1     79      1
     79       1     -1      0      1      1
      1      42      0      1      1     47
      1      47      1      0      5     47
      5      47      0     -1      5     42
      5      42     -1      0      1     42
     75      47      1      0     79     47
     79      47      0     -1     79     42
     79      42     -1      0     75     42
     75      42      0      1     75     47
     15       3      0      1     15     10
     15      10      1      0     35     10
     35      10      0     -1     35      3
     35       3     -1      0     15      3
     39       3      0      1     39     10
     39      10      1      0     61     10
     61      10      0     -1     61      3
     61       3     -1      0     39      3
```

Figure C.3.2.5: LATTICE input for the cavity of Fig. C.3.2.2. The data in the figure was generated using AUTOMESH and this method of generation is recommended.

C.3.3 SUPERFISH Input for AUTOMESH

AUTOMESH prepares an input file, called TAPE73, for LATTICE. It constructs the logical mesh from triangles whose size and shape are specified by the user and from the physical region boundaries. It assigns logical (K, L) coordinates and physical (X, Y) coordinates to points on the boundaries of the region. Extra line regions can be added to the problem where requested. These lines form boundaries for changing the size of the triangles. AUTOMESH automatically assigns boundary conditions (CON(21) through CON(24)) to the problem and writes TAPE73. CON values that AUTOMESH passes to LATTICE are CON(2) = NREG, CON(9) = CONV, CON(21) = 1, CON(22) = 0, CON(23) = 1, CON(24) = 1, CON(36) = NSEG, CON(37) = NCELL = 1.

Occasionally the default values that AUTOMESH gives for some CON values may not be proper for the problem. In this case the user can change CON values when asked for CON changes by LATTICE or he may enter TAPE73 and make changes directly.

The input to AUTOMESH is the same for either SUPERFISH runs or POISSON/PANDIRA runs with two exceptions. The first exception is the first data line, which is the title for the problem. If column 1 is blank, AUTOMESH assigns POISSON/PANDIRA defaults to some variables; if column 1 is not blank, the program assigns SUPERFISH defaults. The second exception occurs when the user elects to use cylindrical coordinates by later setting CON(19) = ICYLIN. For POISSON/PANDIRA (X, Y) corresponds to (R, Z); for SUPERFISH it is the opposite, namely, (X, Y) corresponds to (Z, R). Mathematically this may be confusing, but for most physical problems it is the natural choice. If the user is not satisfied with this convention, he can change it to some extent by setting NSWXY = 1 in TEKPLOT. Of course, this only changes the plots but not the expected inputs to AUTOMESH.

The first line of input to AUTOMESH is the title card. Positions 2 through 80 can be anything. Column 1 should be some nonblank character as mentioned above. The next 8 computer words (columns 2 through 65 on the CRAY, columns 2 through 33 on the VAX) are used for output identification.

Following the first line, AUTOMESH expects one or more groups of data. Each group consists of one REG NAMELIST input followed by one or more PO NAMELIST inputs. The first REG NAMELIST must include a value for NREG; the NREG is the number of REG NAMELISTS expected. The READ structure is shown in Fig. C.3.3.1.

NAMELIST is the standard FORTRAN input routine. Each such input starts with a blank in column 1 followed by $"name" where "name" is the name of the input. Here "name" is either REG or PO. Any item in the group may be entered in

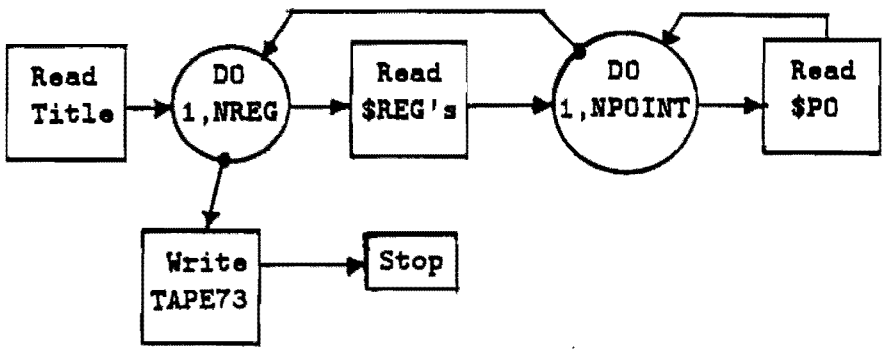


Figure C.3.3.1: Flow chart for read statements in AUTOMESH.

any order separated by commas. If there is any question about NAMELIST, see a FORTRAN manual. The following is a typical NAMELIST entry for the namelist REG:

```
$ REG NREG=5,DX=0.08,XMAX=3.5,YMAX=2.85,IBOUND=1,NPOINT=8$
```

C.3.3.1 The REG NAMELIST.

Twenty-nine quantities can be entered in this NAMELIST for each region of the problem. Most quantities entered for the first region are used for all succeeding regions until changed by a subsequent REG input. This is called "successive region data overwrite". Certain quantities *must* be entered, while others have meaningful default values. The following is a list in alphabetical order describing the quantities as used by SUPERFISH. Some can only be entered in the first REG NAMELIST and must not be changed in sebsequent regions. These are marked by ◇ before the variable.

Table C.3.3.I Region Namelist Variables.

Name	Default	Description
◇ CONV	1.0	Conversion factor for length units. If CONV = 1.0, units are centimeters. To use other units, set CONV to the number of centimeters per unit desired. CONV is the same as CON(9) in the input to LATTICE and should be entered for the first region only.
CUR	0.0	If the region is a drive point, CUR should be set to 1. If another region follows, reset CUR to 0 in that next region. This is the constant C(3) in the Sec. C.3.2 Input for LATTICE.
DEN	0.0	Length of cell for multicell problems. This is C(4) in LATTICE region input. See Section C.3.2.
◇ DX	none	The requested width of triangles in the mesh for the first region. It must not be changed for subsequent regions.
◇ DY	($\propto$ DX)	The requested height of triangles in the mesh for the first region. If DY is not specified the default value is either $\sqrt{3}*$ DX/2 if ITRI = 0, or 1, or DY = DX if ITRI = 2 (right triangle option). It must not be changed for subsequent regions.
IBOUND	−2	A special region boundary indicator. See discussion under the sixth region constant C(6) in the Input for LATTICE. If the present region is to be a new drive point, the user must set IBOUND = −1.

Table C.3.3.I (cont'd.) Region Namelist Variables.

Name	Default	Description
IPRINT	0	If IPRINT = 1, special diagnostic printout is provided by the "logical path-finding" routine. Logical and physical coordinates are also printed for any non-zero value of IPRINT. This information is in file OUTAUT. (IPRINT here is not the same as CON(32) in Table C.3.2.I.)
IREG	(n)	An arbitrary number identifying the region. The default value is 1 for the first entered REG NAMELIST and is incremented by 1 for each succeeding REG NAMELIST.
◇ ITRI	0	The type of triangle to be used for the mesh in the region. ITRI = 0 means equal weight; ITRI = 1 means isosceles; and ITRI = 2 means right. The distinction between equal weight and isosceles is the way that the relaxation is done from the logical mesh to the physical mesh.
◇ KMAX ◇ KREG1 ◇ KREG2 ◇ LMAX ◇ LREG1 ◇ LREG2	none* none* none* none* none* none*	Used to refine the mesh in a user-defined rectangle of the problem area. When these values are entered, they associate a logical mesh number with a physical position. Thus, KMAX corresponds to XMAX, LMAX corresponds to YMAX, KREG1 to XREG1, etc. This allows the user to force a smaller mesh step in a given region. For example, suppose XREG1 = 10.0, XREG2 = 20.0, KREG1 = 10, KREG2 = 110. This gives (KREG1 - 1) = 9 mesh steps in the X-direction of length 10.0/9 = 1.111 up to the location X = XREG = 10, giving a very coarse mesh. From X = 10.0 to X = 20.0 there will be (KREG2 - KREG1) = 100 mesh steps of length DX = 10/100 = 0.1, giving a finer mesh. The size of the mesh beyond X = 20.0 will depend on the difference between KMAX and KREG2, and between XMAX and XREG2. The same principle applies to the Y-direction. The values of these variables are global and hence should be entered for the first region and not changed in subsequent regions.

*If these values are not entered, the code assigns proper values (see under XREG1, XREG2, etc., below).

Table C.3.3.I (cont'd.) Region Namelist Variables.

Name	Default	Description
◇ LINX	0	A special indicator for vertical line regions. LINX = 0 produces vertical line regions at the locations where mesh size changes occur (XREG1 and XREG2). LINX = 1 produces no vertical line regions at the locations where mesh size changes occur. This parameter was introduced into the code in April of 1986. LINX = 1 can help LATTICE converge under some circumstances.
◇ LINY	0	A special indicator for horizontal line regions. It works the same way as LINX above, but for horizontal line regions at locations where mesh size changes occur (YREG1 and YREG2).
MAT	1	The material code for the region. MAT = 0 means that all points in the region are to be omitted from the problem and requires the use of the special boundary indicator IBOUND. The other possible values of this parameter are: MAT = 1, air or vacuum. = 2, 3, 4, 5; Regions with different values of dielectric constant κ_e and relative permeability κ_m.
◇ NCELL	1	The number of cells in a multicell SUPERFISH problem. It is not needed in AUTOMESH or LATTICE, but is passed through the CON array to SF01. It corresponds to CON(37).
◇ NDRIVE	0	Special indicator for drive points. If NDRIVE = 0, AUTOMESH is to assign the drive point. If NDRIVE = 1, AUTOMESH expects the user to eventually supply a one point (NPOINT = 1) region. When AUTOMESH detects NPOINT = 1, it automatically sets CUR = 1 and IBOUND = −1 for that region. This special indicator was installed in the code in July of 1986 to correct a problem. Previously the code might assign two drive points and use its own in place of the user-supplied point.
NPOINT	none	The number of segment endpoints to be entered in the PO NAMELIST that follows this REG NAMELIST.

Table C.3.3.I (cont'd) Region Namelist Variables.

Name	Default	Description
◇ NREG	none	The number of sets of REG NAMELIST data to be entered for this run. This must be entered in the first REG set and should not be changed in subsequent REG sets.
◇ XMAX	none	Maximum physical X value in the problem. For 3-dimensional problems with cylindrical symmetry where the coordinates are denoted by $(r, \varphi = 0, z)$ XMAX is the maximum value of z.
◇ XMIN	none	Minimum physical X value in the problem. For 3-dimensional problems with cylindrical symmetry where the coordinates are denoted by $(r, \varphi = 0, z)$ XMIN is the minimum value of z.
◇ XREG1	XMAX	A line region is added at XREG1. If KREG1 is not set, the width of the triangle will approximately double to the right of XREG1. If KREG1 is set, the triangle width will be determined as described under KMAX, etc. above.
◇ XREG2	XMAX	A line region is added at XREG2. If KREG2 is not set, the width of the triangles will approximately double to the right of XREG2. If KREG2 is set, the triangle width will be determined as described under KMAX, etc., above.
◇ YMAX	none	Maximum physical Y value in the problem. For 3-dimensional problems with cylindrical symmetry where the coordinates are denoted by $(r, \varphi = 0, z)$ YMAX is the maximum value of r.
◇ YMIN	none	Minimum physical Y value in the problem. For 3-dimensional problems with cylindrical symmetry where the coordinates are denoted by $(r, \varphi = 0, z)$ YMIN is the minimum value of r.
◇ YREG1	YMAX	A line region is added at YREG1. If LREG1 is not set, the height of the triangle will approximately double above YREG1. If LREG1 is set, the triangle height will be determined as described under KMAX, etc., above.
◇ YREG2	YMAX	A line region is added at YREG2. If LREG2 is not set, the height of the triangles will approximately double above YREG2. If LREG2 is set, the triangle height will be determined as described under KMAX, above.

C.3.3.2 The PO NAMELIST.

The PO data sets specify the physical boundary of the present region. The first PO data set specifies the initial point. Each succeeding set specifies the endpoint of a curve starting from the previous point, and the type of curve to be drawn from the previous point to this endpoint. The NAMELIST variables are described as follows:

Table C.3.3.II Endpoint Namelist Variables

Name	Default	Description
NT	1	The type of curve to be drawn from the previous point. NT = 1 means a straight line = 2 means an arc of a circle = 3 means a sector of an hyperbola
X0 Y0	0.0 0.0	The cartesian coordinates in the mesh rectangle of the present origin of the coordinates (X, Y) (R, or THETA).
X Y	none none	The cartesian coordinates of the point relative to the present origin point (X0, Y0). For 3-dimensional problems with cylindrical symmetry the coordinates are usually denoted (z, φ, r). AUTOMESH assumes $\varphi = 0$, z $\longrightarrow$ X and r $\longrightarrow$ Y. Naturally Y $\geq$ 0 is assumed.
R THETA	none none	The polar coordinates of the point relative to the present origin point, (see above) UNLESS drawing an hyperbolic segment. In this case, R is the minimum distance from the origin (0, 0) to the hyperbola $R^2 = 2xy$, and THETA has no meaning. For circular arcs THETA is measured in degrees from the displaced horizontal axis. See Figs. C.3.3.2 and C.3.3.3.
NEW	0	This variable is used to force separation between regions. When NEW = 1, the points on the logical path for this segment are not allowed to coincide with points on the path around any previous region. When NEW = −1, the points on the logical path for this segment are not allowed to coincide with the points on the path around any previous region, except for the starting and ending point.

AUTOMESH sets up a cartesian coordinate system with the point (XMIN, YMIN) in the lower left corner and the point (XMAX, YMAX) in the upper right corner.

A point is specified by giving its coordinates relative to the origin point $(X0, Y0)$. The default value of $(X0, Y0)$ is $(0, 0)$. A point can be specified either in cartesian coordinates (X, Y) relative to $(X0, Y0)$ or in polar coordinates relative to $(X0, Y0)$. When defining an arc of a circle between the beginning and ending points of a boundary segment, the origin point $(X0, Y0)$ must be moved to the location of the center of the circle. See Fig. C.3.3.2. If the endpoint is specified by cartesian coordinates, the radius of the circle is calculated by the program to be

$$R = \sqrt{(X - X0)^2 + (Y - Y0)^2} \tag{C.3.3.1}$$

Alternatively, the endpoint may be specified by polar coordinates. The user enters the radius R and the angle THETA in degrees relative to the X-axis.

The only hyperbolas defined by the program are symmetric about the line $X = Y$, i.e., are defined by the equation

$$R^2 = 2XY \tag{C.3.3.2}$$

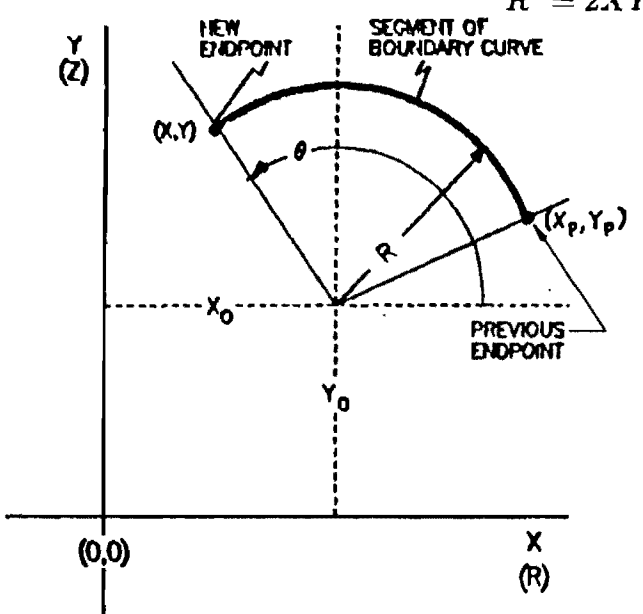


Figure C.3.3.2: Meaning of X0, Y0, R, and θ for circular segment of regional boundary. $(X - X0)^2 + (Y - Y0)^2 = R^2$; θ is in degrees. Note that X and Y are relative to the origin (X0, Y0). X0, Y0, and R must be calculated self-consistently with X_p, Y_p using Eq.(C.3.3.1) to an accuracy of one part in a thousand when the endpoint is in polar coordinates R, θ.

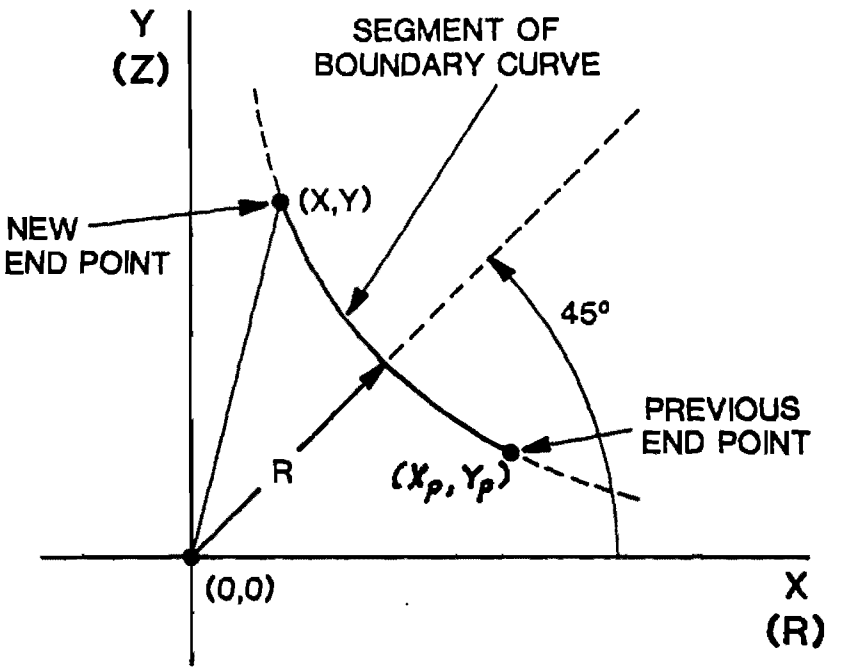


Figure C.3.3.3: Meaning of X, Y, R for hyperbolic segment of regional boundary $R^2 = 2XY$; the choice of X and Y must keep the hyperbola in the first quadrant. The quantities R, X_p and Y_p must be calculated self-consistently to one part in a thousand, and likewise R, X and Y must be self-consistent to the same accuracy.

Figure C.3.3.4 gives an example of AUTOMESH input. It is the file used to generate the three cell cavity LATTICE input in Figs. C.3.2.3. and C.3.2.4

```
1 lattice input example
 $reg xreg=5,dx=.5,xmax=39.,ymax=29.,npoint=13,ncell=3$
 $po x=0.,y=0.$
 $po x=0.,y=1.$
 $po x=3.,y=1.$
 $po x=3.,y=4.$
 $po x=0.,y=4.$
 $po x=0.,y=29.$
 $po x=39.,y=29.$
 $po x=39.,y=4.$
 $po x=31.,y=4.$
 $po x=31.,y=1.$
 $po x=39.,y=1.$
 $po x=39.,y=0.$
 $po x=0.,y=0.$
 $reg npoint=5,mat=2,den=9. $
 $po x=0.,y=15.$
 $po x=0.,y=29.$
 $po x=5.,y=29.$
 $po x=5.,y=15.$
 $po x=0.,y=15.$
 $reg npoint=5,mat=2,den=21.5$
 $po x=34.,y=29.$
 $po x=39.,y=29.$
 $po x=39.,y=15.$
 $po x=34.,y=15.$
 $po x=34.,y=29.$
 $reg npoint=5,mat=0, den=0.$
 $po x=5.,y=1.$
 $po x=5.,y=4.$
 $po x=13.,y=4.$
 $po x=13.,y=1.$
 $po x=5.,y=1.$
 $reg npoint=5,mat=0$
 $po x=16.,y=1.$
 $po x=16.,y=4.$
 $po x=27.,y=4.$
 $po x=27.,y=1.$
 $po x=16.,y=1.$
```

Figure C.3.3.4: Example of AUTOMESH input for three-cell cavity shown in Fig. C.3.2.2.

The input variable NEW was introduced to fix small glitches in the logical mesh caused by an inherent limitation of AUTOMESH. Unless corrected, these glitches can affect LATTICE and cause inaccuracies in SUPERFISH. If the glitch occurs where two regions abut or where a region comes close the the mathematical boundaries of the problem, then sometimes the use of NEW will help.

C.3.3.3 Boundary Condition Data Set Up by AUTOMESH.

The ordinary boundary condition indicators, CON(21) through CON(24) are set to their default values for SUPERFISH problems:

CON(21)	=	NBSUP	=	1
CON(22)	=	NBSLO	=	0
CON(23)	=	NBSRT	=	1
CON(24)	=	NBSLF	=	1

The region special boundary indicators C(6) = IBOUND have been discussed in Sec. C.3.2 above.

Chapter C.4

OUTPUT FROM LATTICE

The function of LATTICE is to find the physical mesh on which the problem is to be solved and to write the necessary mesh and problem information onto a file called TAPE35. LATTICE also produces an output file called OUTLAT.

The information contained in OUTLAT is usually not needed but may sometimes be helpful if something goes wrong in the solution process. OUTLAT contains for each region, the region material number, the total current, the current density, the region boundary indicator IBOUND, and a list of the region's logical and physical boundary points. This is followed by a history of the mesh relaxation interation, which consists of the x-residual, η_x, ρ_x, y-residual, η_y and ρ_y. The quantities η_x and η_y are the x and y rates of convergence of the relaxation process. The quantities ρ_x and ρ_y are the over-relaxation factors.

After the iteration history, a table is printed giving the area of each region and the current density in each region. This is followed by a printout of the problem constants, that is, the CON array. Those CON's that have been changed in the input to LATTICE are flagged.

In addition, any error messages generated by running LATTICE are also recorded in this file. Finally, if CON(32) = IPRINT = −1, LATTICE prints a map of the x and y vectors, that is, it gives the coordinates of each mesh point. Figure C.4.1 illustrates an OUTLAT file.

```
 beginning of lattice execution
 dump 0 will be set up for superfis
 ssuperfish dtl test problem
 region number = 1  material = 1
 total current = 0.0000 current density = 0.0000            0-zoning
 region boundary indicator=  1

     k    l      x           y
     1    1      0.00000     0.00000
     1   78      0.00000     5.00000
     1   93      0.00000     7.00000
     1  146      0.00000    21.10080
    58  146      4.28430    21.10080
    58   93      4.28430     7.00000
    58   78      4.28430     5.00000
    58   63      4.28430     4.03020
    .
    .
    .
   13       58    9        0   -1       58    1
   14       58    1       -1    0        1    1
relaxation parameters,  5582 unknown points.

 elapsed time  =    0.7 sec.
cycle      residx       etax     rhox       residy       etay     rhoy
     1     1.3897e-02   1.0000   1.6000     1.7218e-04   1.0000   1.6000
     2     1.6932e-02   0.6651   1.6000     2.0059e-04   0.6456   1.6000
     3     1.1466e-02   0.6772   1.6000     1.3414e-04   0.6687   1.6000
     4     7.6306e-03   0.6656   1.6000     9.1041e-05   0.6787   1.6000
     5     5.1851e-03   0.6795   1.6000     6.2746e-05   0.6892   1.6000
     6     3.4353e-03   0.6626   1.6000     4.3637e-05   0.6955   1.6000
     7     2.3603e-03   0.6871   1.6000     3.0565e-05   0.7004   1.6000
     .
     .
     .
    81     1.0057e-05   0.9437   1.9003     3.0921e-11   0.7928   1.7676
    82     9.4727e-06   0.9419   1.9003     2.4441e-11   0.7904   1.7676
iteration converged
 elapsed time  =    2.6 sec.
 generation completed

calculated current densities and areas

region   current density            area
 number      (amps/cm**2)          (cm**2)
     1           0.0000            78.9743
     2           0.0000             0.0000
     3           0.0000             0.0000
     4           0.0000             0.0000
```

```
dump number 0 has been written on tape35.

input or default value
problem constants and variables
con(   ) =            , superfish dtl test problem
   (  2) =          4, nreg
con(  6) =          0, mode
   (  9) =  1.000e+00, conv
   ( 18) =          0, nperm
   ( 19) =          1, icylin
   ( 20) =  1.000e+00, xm
   ( 21) =          1, nbsup
con( 22) =          0, nbslo
   ( 23) =          1, nbsrt
con( 24) =          1, nbslf
con( 29) =          0, limtim
   ( 30) =         10, maxcy
   ( 32) =          0, iprint
   ( 34) =         -1, inact
   ( 35) =          0, nodmp
con( 36) =         14, nseg
   :
   :
   (107) =  0.000e+00, zctr
   (108) =  1.800e+02, dphi

solution
problem constants and variables

   (  3) =        146, lmax
   (  4) =         58, kmax
   (  5) =         60, imax
   ( 11) =       6122, nair
   ( 12) =          0, nfe
   ( 13) =          0, ninter
   :
   :
   (109) =       8880, itot
   (118) =      10000, maxdim
   (119) =       5000, nwdim
```

Figure C.4.1: Sections of the file OUTLAT for the problem "superfish dtl test problem."

Chapter C.5

Input to SUPERFISH

Since SUPERFISH gets most of its required input from TAPE35, the user has little to input except CON's. SUPERFISH asks for a dump number and for CON changes.

The following list gives the CON's that affect the SUPERFISH calculation.

CON(18) = NPERM	Number of sets of relative permittivity and permeability (κ_e and κ_m) data to be read in. NPERM is used when regions have material codes (MAT) not equal to 0 or 1 in the REG NAMELIST. After entry of NPERM, the code will ask NPERM times for an input line of the form: "MATER EPSIL FLOMU", where MATER is the material code number in the region having relative permittivity EPSIL and relative permeability FLOMU.
CON(19) = ICYLIN	A flag to indicate the symmetry of the problem. ICYLIN = 1 (default) means the problem has cylindrical symmetry. ICYLIN = 0 means the problem has two-dimensional cartesian symmetry.
CON(30) = MAXCY	Maximum number of iterations to find resonant frequency. Default value is 10; it can be changed if convergence is slow.
CON(32) = IPRINT	If IPRINT is odd, a map of the solution array A is written on OUTFIS file. Default value is 0. This may be useful in debugging.

CON(34) = INACT — A flag to allow interactive control during SUPERFISH frequency iterations. If INACT $\neq -1$, calculation is stopped at end of each iteration and user is asked to type "go", "no", or "in". If "go", iteration continues; if "no", iteration is ended; if "in", user is asked for new iteration values of numbers in the CON array and iteration continues with the new values. The number of iterations is still limited by CON(30). The default value of INACT is -1.

CON(35) = NODMP — A flag which indicates whether the output of the SUPERFISH run is to be written on TAPE35.
If NODMP = 0 (default value), a new dump is written to TAPE35.
If NODMP = 1, no dump is written.

CON(62) = NSTEP — Number of steps in k^2. If NSTEP $\neq 0$, SUPERFISH makes a search in k^2 to aid the user in locating possible resonant frequencies. The program makes NSTEP steps through a range of $k^2 = \epsilon\mu\omega^2$ determined by CON(63) and CON(65) or CON(66) defined below. The default value of NSTEP is 0.

CON(63) = DELKSQ — The size of the steps to be taken in k^2 during the search for a resonant frequency. This number is used only when CON(62) $\neq 0$.

CON(65) = FREQ — An estimate of the resonant frequency to start an iterative convergence to a final resonant frequency or a starting frequency for a step search in k^2 . If making a step search and CON(65) is not zero, then CON(66) is ignored.

CON(66) = XKSQ — Starting value of k^2 to be used in the search for a resonant frequency. This number is used only when CON(62) $\neq 0$ and CON(65) is zero.

CON(73) = IPIVOT — A flag to indicate the type of pivoting desired in the numerical procedure to find the fields. The default is 0, no pivoting.
IPIVOT = 1 gives partial pivoting, and
IPIVOT = 2 gives complete pivoting.
See Sec. 13.4 for an explanation of the pivoting procedure.

CON(86) = EPSIK — Convergence criterion for SUPERFISH iteration. When $|\Delta k^2| \ / \ k^2 <$ EPSIK iteration stops. See also CON(30). Default value is 1.0E-04.

CON(87) = IRESID — If IRESID = 1, the code calculates the residual of the solution matrix A and writes the result to the OUTFIS file. The default is 0.

Chapter C.6

Output from SUPERFISH

SUPERFISH writes information to TAPE35 and an output file called OUTFIS as well as the information it writes to the terminal. Examples of terminal output are given in Chapter 12. The principal information on the solution is written to TAPE35 as dump 1 (or higher). The analysis of the solution information is done by the output routine SFO1, which reads the proper dump information from TAPE35.

The output file OUTFIS contains a list of the CON's values used for the solution, a list of the material properties κ_e and κ_m for each region, and an iteration history. The history is the same as that sent to the terminal.

The dump that SUPERFISH writes to TAPE35 can be read by TEKPLOT and used to plot field lines.

Chapter C.7

Input and Output for TEKPLOT

C.7.1 Input for TEKPLOT

TEKPLOT will plot the physical boundaries and mesh resulting from a LATTICE output. It will also plot the field lines from SUPERFISH output. More than one plot can be made in the same run by repeating the first two input data groups. The structure of the input is shown in Fig. C.7.1.1.

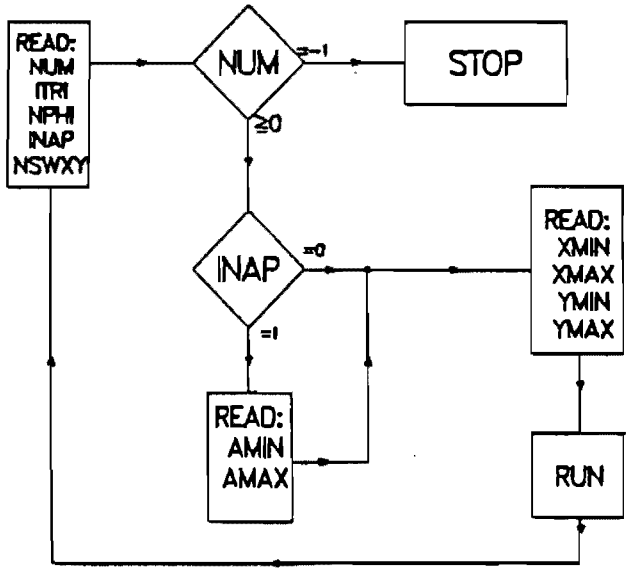


Figure C.7.1.1: Flow diagram for Read Statements in TEKPLOT.

This program uses the special free format described in Sec. C.3.1. The meaning of the parameters is given in the table below.

Name	Default	Description
NUM	0	The TAPE35 "dump" number on which the (X, Y) coordinates of the mesh, and (if NUM > 0) the field values have been written.
ITRI	0	An indicator to specify whether the triangular mesh is to be plotted or only the physical boundary lines of regions. ITRI = 0 means do not plot the triangular mesh. = 1 means plot the triangular mesh.
NPHI	0	The number of field lines to be plotted. The program does not plot lines for the smallest and largest field values, one of which is usually just a point. For TM modes with cylindrical geometry the program plots lines of constant $r * H_\phi(z,r)$, which are parallel to electric field lines. For cartesian geometry it plots contours of constnt H_z. For most problems a good number for NPHI is between 20 and 50.
INAP	0	An indicator for an additional Read statement. INAP = 0 means do not read AMIN and AMAX, = 1 means read (on the next data line) the minimum and maximum values (AMIN and AMAX) of the equipotential lines to be plotted. The values plotted are (AMAX-Δ),($AMAX - 2 * \Delta$),..., ($AMIN + \Delta$), where Δ is (AMAX–AMIN)/ (NPHI+1).
NSWXY	0*	An indicator allowing an interchange of the X and Y axes. NSWXY = 0 means no interchange; NSWXY = 1 means interchange.
XMIN XMAX YMIN YMAX	XMIN XMAX YMIN YMAX	The limits of the plot, which may be any part of the problem rectangle. The variables XMIN, XMAX, YMIN and YMAX should not be confused with variables of the same name that are entered in AUTOMESH and determine the size of the problem rectangle, however, if allowed to default, they will take on the values defined in AUTOMESH.

*Or last input value.

After making the plot, TEKPLOT waits for a carriage return before prompting the user for more input. Upon receiving the carriage return, TEKPLOT asks for a dump number with accompanying input. To terminate the run the user enters -1 S for the dump number. An example of TEKPLOT input is given in Fig. C.7.1.2.

```
tekplot
 ?type input data- num, itri, nphi, inap, nswxy,
?  s
 input data
 num=   0   itri= 0    nphi=   0    inap= 0    nswxy= 0

 plotting prob. name =  full size cavity                    cycle = 0

 ?type input data- xmin, xmax, ymin, ymax
?  80.  125.  0.  25.
 input data
 xmin=  80.000  xmax= 125.000  ymin=   0.000  ymax=  25.000

 ?type go or no
?  go
```

Figure C.7.1.2: An example of interactive input to TEKPLOT.

C.7.2 Output of TEKPLOT

In addition to providing the plots described above, TEKPLOT also makes an output file named OUTTEK, which contains a list of the contour values that were plotted. TEKPLOT will only plot closed regions.

C.7.3 System-dependent Plot Routines in TEKPLOT

TEKPLOT uses PLOT10 commands. If PLOT10 is not available at the user's installation, then the user will have to go into the FORTRAN code and substitute commands from his own graphics system. The calls to PLOT10 and their functions are listed at the beginning of the source code to facilitate substitutions.

Chapter C.8

Further SUPERFISH Output Routines — SFOn

Presently there is no standard program to display the auxiliary properties of an rf cavity. The reason for this is that one is interested in different things for different designs. The useful quantities for a drift-tube linac (DTL) and a radio-frequency quadrupole (RFQ) are not the same. Even in a DTL, the desired output would be different for single cell and multiple cell problems. We are working on a universal SFO routine, but at present there are only SFO's for particular problems. Below we give an example of an SFO that works for one half of a DTL cell and for multicell problems.

SFO1 is a routine used to extract information from the solution produced by SUPERFISH for a DTL. Thus, the inputs to SFO1 tell the code what sort of answers are wanted. The code asks the user to specify the SUPERFISH output "dump number" and then asks for changes in the problem constants. There are eight problem constants that can be changed meaningfully at this stage. These are listed below.

The auxiliary quantities printed out by SFO1 have been described in Sec. C.1.2. There is no universal agreement on these definitions and therefore comparison with the output of other codes such as URMEL, CAVIT, etc., should be done with care.

There exists another postprocessor called SHY which calculates the value of the electric field in the TM mode over an area in the XY-plane. It has been deactivated but will be activated again in the near future.

CON(37) = NCELL	Number of cells in multicell problems. See Secs. C.3.2. or C.12.4
CON(50) = NPEG	Number of boundary segments on which power and frequency perturbations are to be calculated. Default value is 0. If the user enters a nonzero number into CON(50), the program will ask for a list of segment numbers. The segment numbers are separated by spaces. A negative segment number indicates that it is to be calculated as if it were a drift tube stem of radius RSTEM = CON(81).
CON(78) = LINT	The logical L-coordinate of the line along which the normalization integral $\int E_z dz$ is calculated. The default is LINT = 1, the bottom line of the logical mesh. If the z-component of the electric field vanishes on this line, the normalization is ill-defined, and the user must choose some other line. If LINT $\neq 1$, or the line L = 1 is not the horizontal axis (Y = 0) the transit-time factor and related quantities are not calculated.
CON(81) = RSTEM	Radius in centimeters of the assumed drift tube stem. Default value is 1.0. It is used in the calculation of the power dissipation and frequency perturbation.
CON(100) = VSCALE	Normalization factor for average axial electric field. VSCALE is the user-desired, average electric field on the cylindrical (z) axis in volts/meter. The default value is 1.0E+06 V/m.
CON(106) = BETA	The particle velocity divided by the velocity of light. If no value is entered, BETA will be calculated from ZCTR = CON(107) and DPHI = CON(108) on the assumption that the accelerated particle is a proton.
CON(107) = ZCTR	The z-coordinate (cylindrical coordinates) of the "synchronous particle" when the electric field is maximum. Usually, this is the same as the geometric center of the gap between two drift tubes in an Alvarez linac. The default value is zero.
CON(108) = DPHI	The change in the rf-phase in degrees as the "synchronous particle" crosses the portion of the cavity defined by LATTICE. The default value is 180 degrees.

Chapter C.9

PAN-T

PAN-T is a code that calculates the temperature distribution in the walls of an rf-cavity given the electric field at the walls, the thermal conductivity of the wall materials, and the temperature at the outer surface of the wall. The code was written around 1982 and has been used both at Los Alamos and at Chalk River Nuclear Laboratories in Canada. Presently it is not compatible with the standard versions of AUTOMESH, LATTICE, and SUPERFISH, which are required to produce the input for this program. As soon at this code is available for use with the standard versions of the POISSON Group Codes, users will be notified and a more detailed description of its usage will be given.

C.9.1 The Basic Physics of PAN-T

Let us assume that the heat flow in the walls of an rf-cavity has come to equilibrium, so that the temperature $T(x, y, z)$ and the heat flow vector $\mathbf{H}(x, y, z)$, measured in units of (watts/cm^2) are independent of time. It is well known that the heat flow is proportional to the gradient of the temperature, namely,

$$\mathbf{H} = -K(x, y, z)\nabla T, \tag{C.9.1}$$

where K is the thermal conductivity of the wall material, which will be allowed to depend on position in space.

The source of heat is the dissipation of the rf-field in the walls of the cavity. This can be calculated from the field at the walls and the electrical resistivity of the wall material. Let this source be called $Q(x, y, z)$. Its units are watts/cm^3. Energy conservation requires that the amount of heat crossing the surface S of a closed volume V must be equal to the amount of heat generated in the volume. This gives the integral relation

$$\oint_S \mathbf{H} \cdot d\mathbf{S} = \int Q dV. \tag{C.9.2}$$

Using Green's Theorem, we immediately get the differential relation

$$\nabla \cdot \mathbf{H} = Q. \tag{C.9.3}$$

When Eq.(C.9.1.1) is inserted into this equation, one obtains the generalized Poisson equation,

$$\nabla \cdot (K \nabla T) = -Q. \tag{C.9.4}$$

It is easily seen why the solution of this problem is an obvious extension of the POISSON Group Codes. In fact, the author of the codes has used a modification of PANDIRA to obtain the solution. This is probably the source of the name for the program (PANdira for Temperature.) The thermal conductivity function K in PAN-T corresponds to the reluctivity function γ of the magnetic problem. In PANDIRA γ can be a function of the magnetic field. This makes the generalized Poisson equation nonlinear. In PAN-T, the author of the code has restricted the function K to be a linear function of position. We are further restricted to two cartesian dimensions or to cylindrical symmetry in three dimensions. This means that either

$$K(x, y) = A + Bx + Cy \tag{C.9.5}$$

in cartesian coordinates, or

$$K(z, r) = A + Bz + Cr \tag{C.9.6}$$

in cylindrical coordinates. In most applications, the wall is metallic and we can set $B = C = 0$ in the input data.

The specialization of Eq.(C.9.1.4) to cartesian, two-dimensional space requires a slight redefinition of the source term and the thermal conductivity. One must define the source $Q(x, y)$ as having dimensions of watts/cm^2, and the conductivity K as having dimensions of watts/deg C. The heat equation is written

$$\frac{\partial}{\partial x}(K\frac{\partial T}{\partial x}) + \frac{\partial}{\partial y}(K\frac{\partial T}{\partial y}) = Q \tag{C.9.7}$$

In cylindrical coordinates, with all functions independent of the cylindrical angle θ, the heat equation can be written

$$\frac{\partial}{\partial r}\left(rK\frac{\partial T}{\partial r}\right) + \frac{\partial}{\partial z}\left(rK\frac{\partial T}{\partial z}\right) = rQ. \tag{C.9.8}$$

The code replaces K and Q by rK and rQ when finding the solution. These quantities likewise have reduced dimensions of watts/deg C and watts/cm^2.

C.9.2 Preparation of Input File

PAN-T requires as input either file TAPE37 created by LATTICE, or file TAPE38 prepared by SUPERFISH. If the user is going to work from TAPE37, then he must somehow generate and include in this file the heat source term Q. If working from the output of SUPERFISH, the source term is generated automatically and included in TAPE38.

PAN-T recognizes cavity walls by the material numbers assigned to the regions containing the wall material. A region with material number MAT having values between 6 and 10 is assumed to be wall material. The material number for the vacuum or air portion of the cavity is given as MAT = 1. The normal way of entering MAT numbers is in the $REG NAMELIST$ input to AUTOMESH. Figure C.9.2.1 shows an example of an input to AUTOMESH for a drift-tube linac that specifies two wall regions.

When AUTOMESH is executed with this as an input file, it generates an output file called TAPE36 that becomes the input file to LATTICE. LATTICE in turn produces a file called TAPE37 that becomes the input for SUPERFISH. The user runs SUPERFISH in the normal way and this generates an output file called TAPE38. TAPE38 contains most of the information needed by PAN-T, except for the boundary conditions for the thermal problem, the thermal conductivities of the wall materials, and the fixed temperatures on the outer walls of the cavity. The format for entering this information is described in the next section.

```
xcell 31 lampf tank 1
 $reg nreg=3, mat=1, dx=0.5, xmax=7.845, ymax=50.0,
yreg1=12.0,yreg2=24.0, npoint=9 $
 $po x=0.0, y=0.0 $
 $po x=0.0, y=47.0 $
 $po x=0.0, y=50.0 $
 $po x=7.845, y=50.0 $
 $po x=7.845, y=47.0 $
 $po x=7.845, y=9.0 $
 $po x=7.845, y=0.75 $
 $po x=7.845, y=0.0 $
 $po x=0.0, y=0.0 $
 $reg mat=6, npoint=5 $
 $po x=0.0, y=47.0 $
 $po x=0.0, y=50.0 $
 $po x=7.845, y=50.0 $
 $po x=7.845, y=47.0 $
 $po x=0.0, y=47.0 $
 $reg mat=7, npoint=7 $
 $po x=7.845, y=9.0 $
 $po x=4.135, y=9.0 $
 $po nt=2, x0=4.135, y0=7.0, r=2.0, theta=180.0 $
 $po x=2.135, y=1.25 $
 $po nt=2, x0=2.635, y0=1.25, r=0.5, theta=290.0 $
 $po x=7.845, y=0.75 $
 $po x=7.845, y=9.0 $
```

Figure C.9.2.1: Input to AUTOMESH for a DTL cell containing two regions with different wall materials denoted by MAT= 6 and 7.

C.9.3 Special Input to PAN-T

PAN-T uses the special FREE format described in Section C.3.1 above. When running interactively from the terminal, the program prompts for all data with self-explanatory messages. Figure C.9.3.1 shows an example of an interactive session. The computer response numbers are fictitious, but this should give the idea.

The printed output is saved in a file called OUTPANT. The user can get a visual idea of the temperature isotherms by running TEKPLOT just as he would for any SUPERFISH problem. Since we do not have a recent run of PAN-T, we cannot show a plot of the problem: "xcell 31 lampf tank 1."

```
  ?type "tty" or input file name
?  tty
?type input dump number
?  1
 beginning of pant execution from dump number 1
 problem name= xcell 31 lampf tank 1
 ?type input values for con(?)
*21 0 1 0 1 s

  input for conductivity, mater = 6      k=a+b*z+c*r(w/cm-degc)
 ?type a b c
?  4.0 0.0 0.0

  input for conductivity, mater = 7      k=a+b*z+c*r(w/cm-degc)
 ?type a b c
?  4.0 0.0 0.0

 material properties
 region no.   material no.          a          b          c
 (w/cm-degc)
      2            7              4.000      0.000      0.000
      3            6              4.000      0.000      0.000
 ?type input for mat. no. and upper bound. fixed temp.
?  7 30.0

 ?type input for mat. no. and right bound. fixed temp.
?  6 40.0

 elapsed time =  1.0   sec.
  cycle          amin          amax
    0        +1.0000e-06    +1.2345e-02
    1        +1.0000e-06    +1.2345e-02
  solution converged in    1 iterations.
  elapsed tim= 2.0      sec.
 ?type input dump number.
?  -1

 all done
```

Figure 9.3.1: An example of an interactive session with PAN-T.

Chapter C.10

Diagnostic and Error Messages

C.10.1 Messages from AUTOMESH

Diagnostic and Error messages printed from AUTOMESH can be broken into three categories: 1. those starting with the word "ERROR", 2. those starting with the word "TROUBLE", and 3. two additional messages. In the sections below, we list the messages, briefly define the problem and give a possible solution.

The following five conventions make the explanations simpler to write.

1. `CHANGE THE MESH SIZE` — usually the mesh is too coarse; user should rerun the problem with a finer mesh; sometimes a slight mesh size change will suffice.

2. `(X1,Y1)/(R1, THETA1)` — the Cartesian/polar coordinates of the previous point (from).
   `(X2,Y2)/(R2, THETA2)` — the Cartesian/polar coordinates of the present point (to).

3. `R` --- (printed as the value of a variable) means that this variable has been set out of range and not supplied by the user.

4. `(--) means computer prints out the value.`

5. `REGION (--)/O.K.` — AUTOMESH has successfully found paths for all boundary points in this, (--), region; if errors occur in one region, AUTOMESH proceeds to the next.

C.10.1.1 Messages containing "ERROR"

1. `"--- ERROR --- DATA FOR THIS CIRCLE FROM (X1,Y1)/(R1, THETA1)" TO (X2,Y2)/(R2, THETA2) IS INCONSISTENT ...`
   Either one or both coordinates are not given or the two points with center at $(X0, Y0)$ do not lie on the same circle to a relative accuracy of 10^{-3}. Correct the input data for the listed coordinates. The user should check that the coordinates are given RELATIVE to $(X0, Y0)$. Message from subroutine DATUPS.

2. `"--- ERROR --- DATA FOR THIS LINE ARE INSUFFICIENT ..."`
   Either one or both coordinates are not given. Correct the input data for the listed coordinates. Message from subroutine DATUPS.

3. `"--- ERROR --- DATA FOR THIS HYPERBOLA FROM (X1,Y1) TO (X2,Y2) IS INCONSISTENT" ...`
   Either one or both coordinates are not given, R is not given, or the two points do not lie on the same hyperbolic branch to a relative accuracy of 10^{-3}. Message from subroutine DATUPS.

4. `"--- ERROR --- X/Y IS OUT XMIN, XMAX/YMIN, YMAX LIMITS ..."`
   The X or Y point printed is less or greater than the given minimum or maximum value for X/Y in the first REG input line. Correct input. Message from DATUPS.

5. `"--- ERROR --- (KMAX + 2) * (LMAX + 2) = (--) IS GREATER THAN PROGRAM DIMENSIONS OF (--) ..."`
   The total number of mesh points have exceeded the maximum value dimensioned. Cut mesh size or increase parameter MXDIM and recompile as directed by the complete diagnostic message. (Note: Versions of the code received from us before June 1986 have a different diagnostic message and do not give directions for changing MXDIM.) Message from subroutine SETXY.

6. `"--- ERROR --- TROUBLE IN FINDING THE PATH OF A POINT ..."`
   AUTOMESH encountered trouble in both "forward" and or "backward" pass in subroutine LOGIC. To correct, decrease mesh size near the point and try again. Message from main program.

C.10.1.2 Messages containing "TROUBLE"

1. "--- TROUBLE --- DIMENSIONS FOR THE NSEG ARRAYS, EXCEEDED NSG OF (--) ..."
   AUTOMESH has exceeded the maximum number of boundary segments dimensioned in the program. Increase parameter NSG and recompile as directed. Message from subroutine FISHEG. (Note: Versions of the program received before June 1986 have no directions for increasing NSEG. The user can only decrease the number of segments in the input.)

2. "--- TROUBLE --- NPOINT = (--), EXCEEDS DIMENSION OF (--)"
   The number of PO entries for this region has exceeded the maximum number dimensioned. To correct, decrease the number of points or increase parameter NPTX and recompile as directed. (Note: Versions of the program received before June 1986 have no directions for increasing NPTX. The user can only decrease the number of points in the input.) Message from main program or subroutine INSERT.

3. "--- TROUBLE --- THE PROGRAM FOUND THE SAME (K, L) COORDINATES FOR THE FIRST AND LAST POINT OF THIS CURVE ..."
   The program has assigned the same mesh point in either vertical or horizontal direction for $(X_1,\ Y_1)$ and $(X_2,\ Y_2)$. This usually means mesh size is not fine enough.

   3a. Message is printed from subroutine LOGIC. The last line of the message prints the phrase "FORWARD PASS" or "BACKWORD PASS." AUTOMESH executes subroutine LOGIC twice—first in a "forward" search, and a second pass in a "backward" search—to find the path of the current segment. Then the program chooses the path with the smaller number of segments with no errors. A fatal error occurs if BOTH directions encounter "TROUBLE." To correct, change mesh size.

4. "--- TROUBLE --- PROGRAM DIMENSIONS 1000 FOR THE *KL* ARRAYS ARE INSUFFICIENT"
   The program has difficulty in finding the path for this segment and thus has exceeded the dimension allocated for storage of the path array. See 3a. above.

5. "--- TROUBLE --- LOGICAL PATH IS TRAPPED AT K = (--), L = (--)"
   The program cannot find the path for this current segment. See 3a above.

6. "--- TROUBLE --- CANNOT FIND A FIXED H-PHI POINT"
   Message from subroutine POTREG. The code cannot find a point in the problem region with coordinates that agree with the assigned magnetic drive point coordinates. In principle you should not get this message unless you assign your own drive point and make an error in the input. In practice, we suspect a logic error in older versions of the code. Please call us if you cannot find the source of the error.
   In versions of the code received after Septemeber, 1986, the message means that AUTOMESH had difficulty assigning the drive point at the upper lefthand corner of the cavity. To correct, input your own drive point region by setting NDRIVE = 1 in the first region and later defining a point region where you want the drive point. AUTOMESH will automatically set CUR = 1 and IBOUND = −1 as required.

7. "--- TROUBLE --- TOO MANY END POINTS FOUND FOR THE LINE"
   The program has trouble adding a vertical/horizontal line region.

   7a. AUTOMESH could encounter a number of problems in subroutines XLINER/YLINER while attempting to add vertical/horizontal line regions. To correct, CHANGE MESH SIZE or in versions of the program received after April, 1986 set LINX/LINY = 1 in the first REG entry. (This latter option deletes the addition of all vertical/horizontal line regions at horizontal/vertical mesh change locations.)

8. "--- TROUBLE --- NO END POINTS FOUND FOR LINE"
   The program has trouble finding a mesh point for the end point of the added line region. See 7a. above.

9. "--- TROUBLE --- ONLY ONE END POINT FOR THE LINE"
   The program has trouble finding an end point for this added line region. See 7a. above.

10. "--- TROUBLE --- A POINT WITH (K = KREG) HAS X NOT = TO XREG"
    "--- TROUBLE --- A POINT WITH (L = LREG) HAS Y NOT = TO YREG"
    The program has difficulty adding a vertical/horizontal line region. See 7a. above.

C.10.1.3 Additional Diagnostic Messages

1. `"DIMENSION OF 2000 FOR KR, LR ..."`
   The program has run into difficulty and has exceeded the maximum number of points dimensioned for a region. `CHANGE MESH SIZE` and try again. Message from subroutine LOGSEG.

2. `"DIMENSION OF 3000 INSUFFICIENT FOR KG, LG ..."`
   The program has run into difficulty and has exceeded the total number of points dimensioned for *all* regions. `CHANGE MESH SIZE` and try again. Message from subroutine SAVAGE.

C.10.2 Messages from LATTICE

LATTICE writes all of diagnostic and error messages to the output file, OUTLAT, and some to the terminal if run is interactive. An explanation of the common terminology used in these messages is listed below.

1. k, l The mesh point numbering for the horizontal and vertical coordinates.
2. x, y The horizontal, vertical coordinates, respectively.
3. k', l' The mesh point numbering for the second of the two points.
4. (--) Means the computer prints out the value.

C.10.2.1 Messages Containing "ERROR EXIT"

1. `"--- ERROR EXIT --- TWO MESH DATA POINTS WITH A DIFFERENT K, L HAVE THE SAME X, Y COORDINATES"`
followed by values of k, l, k', l', x, and y. This message is from the function ANGLF. The code has found the same physical coordinates assigned to two different logical points. Check input data; try reducing mesh spacing if input looks correct.

2. `"--- ERROR EXIT --- IN SUB. ANGLE COST = (--) AT KO = (--) LO = (--)"`
Message from function ANGLF. The code has a cosine value greater than 1.0 at the logical point (KO, LO). Check input data, try reducing mesh spacing.

3. `"--- ERROR EXIT --- NWMAX EXCEEDS PROGRAM DIMENSIONS OF (--) ..."`
Message from subroutine PRELIM. The storage for recalculating couplings has been exceeded. This storage has dimension of 1/2 of the parameter MXDIM. Increase MXDIM and recompile.

C.10.2.2 Messages Containing "INPUT DATA ERROR"

1. `"--- INPUT DATA ERROR --- ILLEGAL CHARACTER",`
followed by a print of the input line. Message from subroutine FREE. The code has found a character it does not recognize in the line printed. Correct input.

2. `"--- INPUT DATA ERROR --- NO MANTISSA WITH EXPONENT",`
followed by a print of the input line. Message from subroutine FREE. The code found an exponent standing alone in the line printed. Correct the input line.

C.10.2.3 Messages Containing "DATA ERROR"

These messages are issued whenever LATTICE encounters any errors in reading the input file. Mostly, such errors occur when a user creates his own input file for LATTICE. If the input file for LATTICE has been generated by a successful AUTOMESH run, it is unlikely there would be any errors of this type. In any case, the errors issued are self-explanatory. The user need only correct the identified error in the input file and rerun.

1. `"--- DATA ERROR --- THE NO. OF BOUND DATA VALUES (K, L, X, Y) = (--) FOR THIS REGION IS NOT A MULTIPLE OF 4"`
   Message from subroutine REREG. The code has found that the coordinate data on the input file is incomplete. Correct the input file (Usually TAPE 73). If generated by AUTOMESH, try changing mesh spacing.

2. `"--- DATA ERROR --- THE FIRST AND LAST POINTS OF REGION HAVE SAME K, L BUT DIFFERENT X, Y COORDINATES"`
   Message from subroutine REREG. Meshing has been done incorrectly. Correct input file. If generated by AUTOMESH, try changing mesh spacing.

3. `"--- DATA ERROR --- NEGATIVE OR ZERO L"`
   Message from subroutine REREG. The input file has an illegal value for the logical coordinate L. Correct input file.

4. `"--- DATA ERROR --- NEGATIVE OR ZERO K"`
   Message from subroutine REREG. The input file has an illegal value for the logical coordinate K. Correct input file.

5. `"--- DATA ERROR --- L, K AND LPRIME, KPRIME NOT ON SAME LOGICAL LINE"`
   Message from subroutine REREG. There is an error in the boundary input to LATTICE. Check input file.

6. `"--- DATA ERROR --- L EXCEEDS LMAX"`
   Message from subroutine REREG. The code has found a boundary point in the input file with a logical L coordinate greater than the maximum L coordinate. Correct input.

7. "--- DATA ERROR --- K EXCEEDS KMAX"
   Message from subroutine REREG. The code has found a boundary point in the input file with a logical K coordinate greater than the maximum K coordinate. Correct input.

8. "--- DATA ERROR --- YOU HAVE EXCEEDED THE MAXIMUM NUMBER OF REGIONS ALLOWED = (--)"
   Message from subroutine REREG. Too many regions. Increase parameter NRGN and recompile.

9. "--- DATA ERROR --- YOU HAVE EXCEEDED THE MAXIMUM NUMBER OF INPUT BOUNDARY POINTS PER REGION = (--)"
   Message from subroutine REREG. The storage for single region boundary points has been exceeded. Increase parameter NPMX and recompile.

10. "--- DATA ERROR --- TWO CONSECUTIVE DATA POINTS IN THIS REGION HAVE SAME K, L COORDINATES"
    Message from subroutine REREG. The code has found two consecutive boundary points assigned the same logical coordinates. Correct input data.

11. "--- DATA ERROR --- TWO CONSECUTIVE DATA POINTS IN THIS REGION HAVE SAME X, Y COORDINATES"
    Message from subroutine REREG. The code has found two consecutive boundary points assigned the same physical coordinates. Correct input data.

12. "--- DATA ERROR --- (KMAX+2)*(LMAX+2) EXCEEDS PROGRAM DIMENSIONS OF (--)"
    Message from subroutine REREG. There are too many mesh points in the problem. Increase the parameter MXDIM and recompile.

C.10.2.4 Messages Containing "TROUBLE" and "WARNING"

1. "--- TROUBLE --- DIMENSIONS FOR NO. OF SEGMENTS EXCEEDED NSG OF (--) ..."
   Prints to OUTLAT and terminal and immediately aborts. Message from main program; follow instructions given in the complete error message and recompile.

2. "--- WARNING ---THE MESH HAS NEGATIVE AND/OR ZERO AREA TRIANGLES"
   LATTICE writes to the file OUTLAT, a message whenever it encounters negative or zero area in subroutine FILPOT, followed by the three coordinates that make up this triangle. The program processes the triangles of all regions before printing above message to OUTLAT and terminating. Message from main program; follow instructions or remesh the problem with a different mesh spacing.

3. "--- WARNING ---THE NUMBER OF INTERIOR POINTS = 0 ..."
   Message from subroutine SETTLE is self-explanatory in versions released after April 1986. For previous versions, the user has somehow set up the problem wrong. All points are boundary points and hence the potential is determined everywhere.

C.10.2.5 Miscellaneous Messages

1. "THE ABOVE REGION IS NOT CLOSED."
   This message is output to OUTLAT from subroutine REREG and is only a warning. User should check to see that the same values for the first and last coordinates for this region are specified if a closed region with interior points is desired.

2. "ITERATION TERMINATED---MAXIMUM NUMBER OF CYCLES."
   This message is output to OUTLAT from subroutine SETTLE and is only a warning. The mesh generation did not converge to the required accuracy after 100 iteration cycles. Run is continued with present mesh. User could try running the problem with this mesh or CHANGE MESH SIZE and rerun.

3. "THE LAST CORRECT POINT IS K = (--), L = (--)"
   Message from subroutine REREG. This message occurs after INPUT DATA ERROR messages numbers 3, 4, 5, 6, 7, 10, and 11. The logical coordinates are an aid in finding the error.

4. "ITERATION CONVERGED"
   Message from subroutine SETTLE. The mesh relaxation process was successful.

C.10.3 Messages from SUPERFISH

C.10.3.1 Messages with "ERROR EXIT"

1. `"--- ERROR EXIT --- NO DATA FOR EPSILON/MU MATERIAL"`
   Message from subroutine EPSIN. The code found a region with MAT $\geq$ 2 but found NPERM = CON(18) equal to zero. Rerun with CON(18) set equal to the number of different set of κ_e/κ_m lines to be input. See CON(18).

2. `"--- ERROR EXIT --- EPSILON = 0.0 FOR MATERIAL NO. (--)"`
   Message from subroutine EPSIN. The code found an error in the κ_e/κ_m input for one of the materials with material number > 1. See CON(18) for proper input format.

3. `"--- ERROR EXIT --- 2ND TRY --- IMPROVEMENT FAILED"`
   Message from subroutine FROOT. The root finder rejected an improvement for the 2nd time and ended the run. Try rerunning with a different initial frequency.

4. `"--- ERROR EXIT --- DIMENSIONS OF 10 FOR FREQ. ITERATION EXCEEDED"`
   Message from subroutine FROOT. The code did not find a resonance after 10 tries and has stopped the run. Rerun with a better guess for the resonant frequency.

5. `"--- ERROR EXIT --- (KMAX+2)(LMAX+2) EXCEEDS PROGRAM DIMENSIONS"`
   Message from subroutine SRDUMP. Too many points in the problem mesh. Increase the MAXDIM parameter and recompile.

6. `"--- ERROR EXIT --- THE MESH HAS NEGATIVE AND/OR ZERO AREA TRIANGLES"`
   Message from subroutine SRDUMP. The mesh has a region where the triangles have collapsed or where logical lines have crossed. Remesh the problem. If using LATTICE try changing the mesh spacing.

7. `"--- ERROR EXIT --- NROW = MINO(KMAX, LMAX) = (--) EXCEEDS MATRIX DIMENSIONS OF (--)"`
   Message from subroutine STRIBES. The storage needed for the matrix inversion is greater than allowed. Increase parameter IMX in all places it occurs and recompile SUUPERFISH and POILIB. (Note: there may be other dimensions statements to change in early versions of the code.)

C.10.3.2 Messages with "INPUT DATA ERROR"

1. `"--- INPUT DATA ERROR --- ILLEGAL CHARACTER"`
   Followed by a print of the input data line. Message from subroutine FREE. The code has found a character it does not recognize in the line printed. Correct input.

2. `"--- INPUT DATA ERROR --- NO MANTISSA WITH EXPONENT"`
   Followed by print of the input data line. Message from subroutine FREE. The code has found an exponent standing alone in the line printed. Correct input line.

C.10.3.3 Miscellaneous Messages

1. `"--- SOLUTION TERMINATED --- MAXIMUM NUMBER OF ITERATIONS"`
   Message from main program. The code has performed the number of iterations requested by CON(30) = MAXCY and stopped the iteration. Restart with better guess for resonant frequency.

2. `"--- WARNING --- CC = (--) SET = 0.0 IN FROOT"`
   Message from subroutine FROOT. This is a warning that the root finder is having some trouble. If the run continues and converges, it may be ignored. If the iteration does not converge try a better guess for the resonant frequency.

3. `"--- TROUBLE WITH THE LAST IMPROVEMENT"`
   Message from subroutine FROOT. The root finder doesn't like its answer. This print will be followed by message number 4 (below) or by ERROR EXIT message number 3.

4. "THE PREVIOUS $D(K**2)$ WILL BE DISCARDED"
   Message from subroutine FROOT. The root finder has discarded the results of the last iteration. If the iteration converges this can be ignored. If the run is stopped, try a better guess for the resonant frequency.

5. `"NO. OF REGIONS INPUT, (--) GREATER THAN NRGN."`
   Message from subroutine SRDUMP. The region storage is too small. Increase parameter NRGN and recompile.

6. `"NO. OF SEGMENTS, (--) GREATER THAN NSG."`
   Message from subroutine SRDUMP. The number of segments input exceeds storage. Increase parameter NSG and recompile.

C.10.4 Messages from SFO1

C.10.4.1 Messages with "ERROR EXIT"

1. `"--- ERROR EXIT --- RSUM = 0.0 IN SUB. HELINE"`
   Message from subroutine HELINE. The code is doing an integral and has found two adjacent points with average radius (or average y coordinate) equal to 0. In a cylindrical geometry this will result in a divide by 0 so the run is stopped. Check to see if segment numbers requested for power and frequency shifts are correct. Don't ask for power and frequency shifts on the offending segment.

2. `"--- ERROR EXIT --- KPATH, LPATH DIMENSIONS OF 500 EXCEEDED"`
   Message from PATH. The storage in COMMON/PATB/ has been exceeded. In most cases this means that the code cannot find the path for this segment. For a very long segment an increase in storage may help but this is rare. Sometimes a change in mesh spacing will help. The rest of the problem can often be done by not requesting power on this segment.

3. `"--- ERROR EXIT --- (KMAX+2)(LMAX+2) EXCEEDS PROGRAM DIMENSIONS OF (--)"`
   Message from subroutine ZRDUMP. Too many points in problem. Increase parameter MAXDIM.

4. `"--- ERROR EXIT --- THE MESH HAS NEGATIVE AND/OR ZERO AREA TRIANGLES"`
   Message from subroutine ZRDUMP. The mesh has a region where the triangles have collapsed or where the logical lines have crossed. Remesh the problem. If using LATTICE, try changing the mesh spacing.

C.10.4.2 Message with "INPUT DATA ERROR"

1. `"--- INPUT DATA ERROR --- ILLEGAL CHARACTER"`
   Followed by a print of the input line. Message from subroutine FREE. The code has found a character it does not recognize in the line printed. Correct input.

2. `"--- INPUT DATA ERROR --- NO MANTISSA WITH EXPONENT"`
   Followed by a print of the input line. Message from subroutine FREE. The code has found an exponent standing alone in the line printed. Correct input line.

C.10.4.3 Messages with "DATA ERROR"

1. "--- DATA ERROR --- THE STARTING K,L MESH POINT IS NOT A REGION BOUNDARY POINT"
   Message from subroutine PATH. The code has found that an interior point has been passed as a segment end print. Probably a code bug.

C.10.4.4 Messages with "ERROR RETURN"

1. "--- ERROR RETURN --- CAVITY LENGTH IS NOT CORRECTLY DEFINED"
   Message from subroutine EZAXIS. The code found no boundary or interior points along logical line given by CON(78) = LINT or the value of ZLONG was 0. LINT should be 1 on entry to EZAXIS.

C.10.4.5 Messages with "WARNING"

1. "--- WARNING --- RO = 0.0 in SUB. TRASIT"
   Message from subroutine TRASIT. The code is calculating transit time integrals and has found a point whose right upper neighboring point is not in the problem. This would result in a divide by zero so the contribution of the point to the integral is ignored.

2. "--- WARNING --- YO = 0.0 IN SUB. TRASIT"
   Message from subroutine TRASIT. The code is calculating transit time integrals and has found a point whose right upper neighboring point is not in the problem. This would result in a divide by zero so the contribution of the point is ignored.

C.10.4.6 Miscellaneous Messages

1. "INTEGRATION ON THE Z-AXIS IS NOT ALLOWED"
   Message from subroutine HEPOW. The user has asked for power on a boundary segment that lies on the Z axis in cylindrical geometry. This will result in a divide by zero so the code stops. Remove the request for power on the offending segment.

2. "NO. OF REGIONS INPUT, (--) GREATER THAN NRGN"
   Message from subroutine ZRDUMP. The number of regions to be input exceeds the storage. Increase parameter NRGN and recompile.

3. "NO. OF SEGMENTS (--) GREATER THAN NSG"
   Message from subroutine ZRDUMP. The number of segments being input exceeds the storage. Increase parameter NSG and recompile.

C.10.5 Messages from TEKPLOT

1. `"--- INPUT DATA ERROR --- ILLEGAL CHARACTER"`
   Followed by a print of the input line. Message from subroutine FREE. The code has found a character it does not recognize in the line printed. Correct input.

2. `"--- INPUT DATA ERROR --- NO MANTISSA"`
   Followed by a print of the input line. Message from subroutine FREE. The code has found an exponent standing alone in the line printed. Correct the input line.

3. `"NUMBER OF REGIONS ON TAPE35 = (--) LARGER THAN DIMENSION NRGN.RUN STOPPED"`
   Message from subroutine TRDUMP. The number of regions to be input exceeds storage. Increase parameter NRGN and recompile.

Chapter C.11

Convergence and Accuracy

C.11.1 Convergence

When searching for a resonance, SUPERFISH checks the value of $\Delta k^2/k^2$ ($k = 2\pi f/c$) to see if it is less than CON(86) = EPSIK. Here, Δk^2 is the change in k^2 from the previous iteration. The default value of EPSIK is 10^{-4}. The allowed number of iterations is set by CON(30), which has a default value of 10. This number should be enough to find the resonance in most cases. If it is not enough, the code root finder may be lost. The user should check the printed values of $D1(K**2)$ to determine what is going on. If these values are decreasing monotonically, the code is probably converging and the user should start over with a new starting frequency near to the iteration's final frequency. If the $D1(K**2)$ values are jumping or the printed frequencies are far apart, it is likely that the code is having trouble and it might be profitable for the user to run SUPERFISH in the step mode (INACT $\neq -1$) to get a better feel for the location of the zeros of $D1(K**2)$.

C.11.2 Accuracy

In an early article, Halbach and Holsinger[2] report errors of 1 part in 10^4 for the frequency and 1 part in 3000 for the stored energy when calculating the fundamental mode of a pill box cavity modeled with 1395 points. Accuracy was less for higher modes.

Gluckstern, Ryne, and Holsinger[5] have shown that 1 part in 10^4 accuracy in frequency is obtainable in a pill box cavity with equal length and radius using a mesh as coarse as 15x15. For a spherical cavity they obtain 1 part in 10^4 using a 20x20 mesh. Gluckstern, Ryne, and Holsinger[5] and Gluckstern[6] have shown that the convergence goes as N^{-2} where N is the number of points in the mesh and have developed a method for improving SUPERFISH results using a postprocessor.

Several years ago we compared the results of SUPERFISH with the results of URMEL on standard problems like the pill box and the spherical cavity. URMEL gave frequencies of the higher modes which were consistently 0.3 to 0.5 % lower than those of SUPERFISH. We have not benchmarked the more recent version of URMEL against SUPERFISH and therefore all we can say is that the user should be cautious when using the frequencies of higher-order modes.

Another more annoying problem has to do with the values of power loss, Q, and shunt impedance. We find that different cavity codes use different definitions. Also different versions of SFO1 out in the user community use different definitions. Some, including our current standard version, do not scale properly when the normalization constant VSCALE is changed. Once again, the user should be cautious until we can clarify this matter.

Chapter C.12

SUPERFISH Examples

C.12.1 Spherical Cavity

This example is a SUPERFISH run of a spherical cavity. The input file SPHI is shown in Fig. C.12.1.1. A 1 appears in column one of the first card to denote a SUPERFISH problem. The file is very simple. The calculation requires only half the sphere and since the code assumes it is working with a figure of revolution about the z-axis if ICYLIN = CON(19) = 1 (default for SUPERFISH problems), only the upper half of the half sphere need be modeled.

```
1 1 spherical cavity
2  $reg nreg=1, dx=2., xmax=100., ymax=100., npoint=4 $
3  $po x=0., y=0. $
4  $po x=0., y=100. $
5  $po nt=2. r=100., theta=0. $
6  $po x=0., y=0. $
```

Figure C.12.1.1: Input file for AUTOMESH.

AUTOMESH is run by typing the executable file name, **automesh**. Figure C.12.1.2 shows the session at the terminal. AUTOMESH asks for an input file name; it is given **sph1** and executes quickly. Typing the LATTICE executible file name **lattice**, results in the action shown in Fig. C.12.1.3. On asking, the code is told that tape73 is the input file. Then LATTICE asks for CON's; there are no changes so the reply is "s". The code then executes.

To run SUPERFISH the executable file name, **fish** is typed (See Fig. C.12.1.4). When it asks, the code is told to take input from TTY and to use dump 0. The only change necessary to the CON's is an initial frequency value; CON(65) is set to 130.MHz. SUPERFISH finds the resonant frequency in three iterations.

```
?  automesh
 ?  type input file name
?  sphi

 region no.   1
 logical boundary segment end points
 iseg       kb   lb       kd   ld       ke    le
    1        1    1        0    1        1    59
    2        1   59        1    0       51     1
    3       51    1       -1    0        1     1
  stop
 automes   ctss time     .287     seconds
 cpu=      .102   sys=    .020   i/o+memory=        .165

  all done
```

Figure C.12.1.2: Log of interaction with AUTOMESH.

```
lattice
 ?  type input file name
?  tape73

 beginning of lattice execution

 dump 0 will be set up for superfis

 1 spherical cavity

 ?type input values for con(?)
?  s

 elapsed time  =   0.6 sec.
0iteration converged

 elapsed time  =   1.4 sec.

 generation completed

 dump number 0 has been written on tape35
 stop
lattice   ctss  time      1.615      seconds
cpu=     1.205   sys=       .022     i/o+memory=       .388

  all done
```

Figure C.12.1.3: Log of interaction with LATTICE.

```
fish
 ?  type "tty" or input file name
?  tty
 ?  type input value for dump num
?  0
 beginning of superfish execution from dump number  0

 prob. name  =   spherical cavity

 ?  type input values for con(?)
?  *65 130.  s

 elapsed time  =    1.3 sec.

 cycle      hmin          hmax              residual

     0   0.0000e+00  0.0000e+00          1.000e+00

        ------------------------------------------------------------
                                                        k**2 = 7.4234e-04
                                                        freq = 1.3000e+02
 solution time =    2.297 sec.

      1   0.0000e+00   1.0525e+00   1.000e+00
 kfix =   1   lfix = 59 delta1 =   2.7168e-02 d1(k**2)=  1.1284e-05
 ------------------------------------------------------------------
                      using slope = -1 formula with rix =1.000
 del k**2 =  1.1284e-05   k**2  =  7.5363e-04    freq =  1.3098e+02
 solution time =    2.208 sec.

     2   0.0000e+00   1.1319e+00   1.000e+00
 kfix  =    1  lfix = 59 delta1 = -2.2771e-03 d1(k**2)= -8.0543e-07
 ------------------------------------------------------------------
                                          delta1(k**2)     d1(k**2)
                                 1st deriv.  -2.6094e+03  -1.0714e+00
                            using two point secant formula
 del k**2 = -7.5177e-07   k**2 =  7.5288e-04    freq =   1.3092e+02
 solution time =    2.419 sec.

     3    0.0000e+00   1.1261e+00   1.000e+00
 kfix =  1   lfix = 59 delta1 = -1.6309e-04 d1(k**2)=   -5.8330e-08
 ------------------------------------------------------------------
                                         delta1(k**2)      d1(k**2)
                                 1st deriv.  -2.8121e+03   -9.9379e-01
                                 2nd deriv.  -9.6241e+06    3.6833e+03
                           using three point parabola formula
 del k**2  = -5.8344e-08   k**2 =  7.5282e-04    freq =  1.3091e+02

 solution converged in     3 iterations

 dump number    1 has been written
```

Figure C.12.1.4: Log of interaction with SUPERFISH.

To get information on the performance of the cavity we run SFO1, Fig. C.12.1.5. SFO1 is told to take input from the TTY and to use dump 1, which holds the SUPERFISH solution. When asked, CON(50) is set to 1, indicating that power and frequency shifts are desired on only one segment. SFO1 asks for the number of that segment and we give it "2". This segment is the only one on the boundary of the sphere. SFO1 asks if a summary is wanted at the terminal and prints it when answered "go". Some of the information printed doesn't apply, for example, STEM radius, but the user should have no problem separating the wheat from the chaff.

```
 sfo1
?type "tty" or input file name
 ?  "tty"
 ?type input value for num
?  1
 begining of sfo1 execution from dump number 1
 prob.  name = spherical cavity
 ?type input values for con(?)
?  * 50 1 s
 ?type input values for iseg's
?  2
 ?type go for output summary at terminal
?  go
superfish dtl output summary    10:13:53   84/09/25
problem name = spherical cavity
cavity length  =  200.000 cm    cavity diameter  =       0.000 cm
     d.t. gap  =    0.000 cm        stem radius  =       1.000 cm
frequency (starting value = 130.000)             =     130.914 mhz
beta =  0.8734                     proton energy  =     988.050 mev
normalization factor   (e0=1 mv/m)        ascale  =     7398.1
stored energy (mesh problem only)                =     3.7281 joules
power dissipation  (mesh problem only)           =   23955.43 watts
t,tp,tpp,s,sp,spp  =   0.158   0.150   -0.034   0.650   -0.058   0.042
q  =  128012                     shunt impedance  =       41.74 mohm/m
product z*t**2                               ztt  =        1.04 mohm/m
magnetic field on outer wall                     =        1952 amp/m
maximum electric field on boundary               =       0.543 mv/m

iseg   zbeg   rbeg   zend   rend   emax   power          d-freq   d-freq
       (cm)   (cm)   (cm)   (cm)  (mv/m)   (w)           (delz)   (delr)
 2  0.00010  0.00010  0.000  0.000 0.5430  2.40e+04 wall  -0.0313  -0.1296
 ?type input value for num
?
```

Figure C.12.1.5: Log of interaction with SFO1.

TEKPLOT can be used to display the field pattern. Fig. C.12.1.6 shows how it's done. We enter the executable file name TEKPLOT. The program asks for NUM, ITR1, NPHI, INAP, NSWXY. The answer 1 0 75 s tells the code to set NUM = 1 (use dump 1), ITR1 = 0 (no mesh drawn), NPHI = 75 (draw 75 lines). The final "s" tells the code to use defaults for the remaining values. The code asks for x and y limits on the region to be plotted and we answer "s" to tell it to use the internal values of XMIN, XMAX, YMIN and YMAX. A "go" when asked, causes the program to produce Fig. C.12.1.7. TEKPLOT is terminated by hitting the carriage return and entering -1 s when asked for a new value for NUM.

Additional information on the run can be found in the OUTAUT, OUTLAT, OUTFIS, and OUTSFO files produced by the codes used.

```
tekplot
 ?type input data- num, intri, nphi, inap, nwxy,
?  1 0 75 s
 input data
 num= 1 itri= 0 nphi= 75 inap= 0 nswxy= 0

 plotting prob.  name = spherical cavity

 ?type input data- xmin, xmax, ymin, ymax
?  s
 input data
 xmin= 0.000 xmax= 100.000 ymin=0.0000 ymax = 100.000

 ?  type go or no
?  go
```

Figure C.12.1.6: Log of interaction with TEKPLOT.

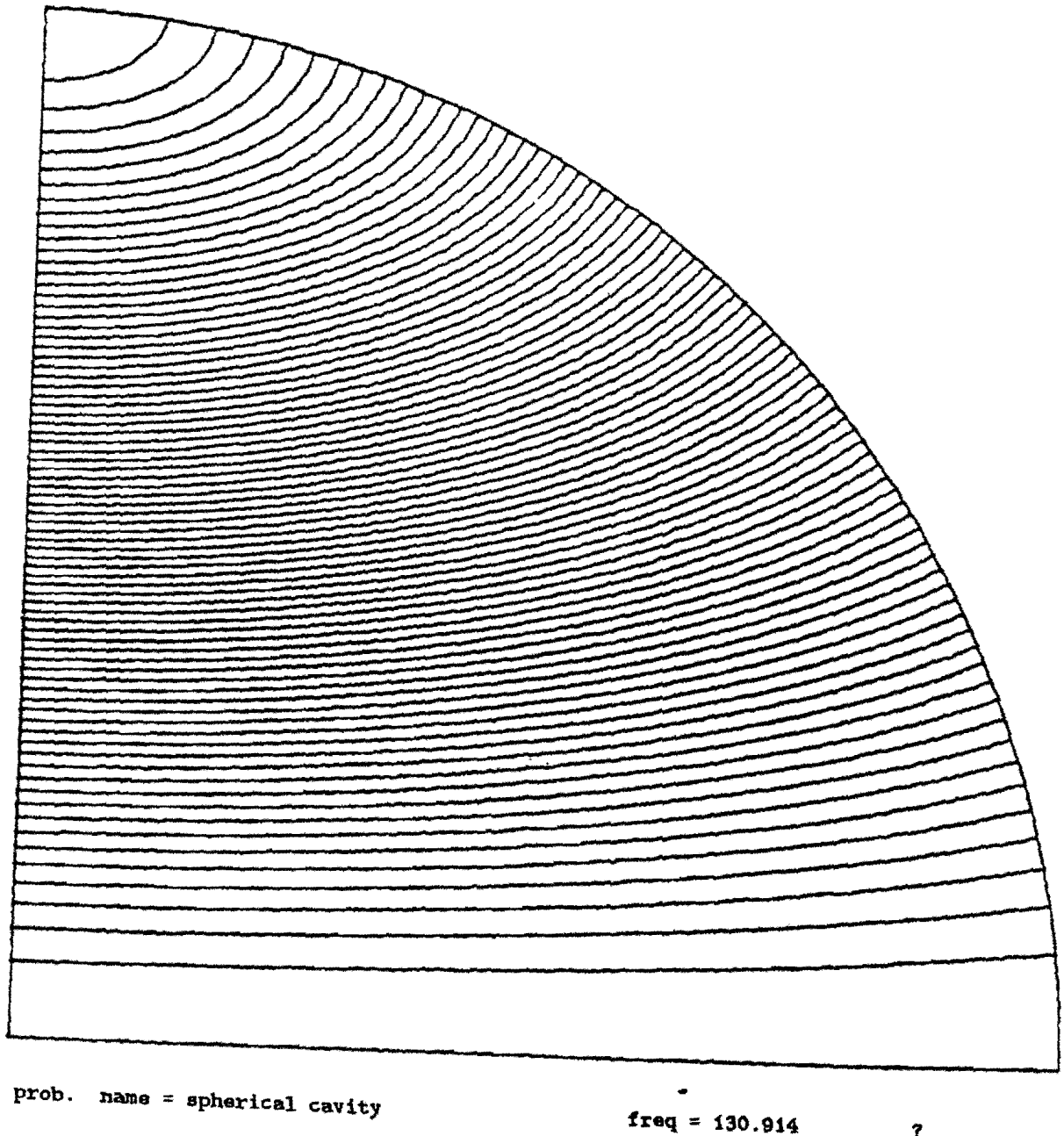


Figure C.12.1.7: TEKPLOT output of electric field lines.

C.12.2 Synchrotron Cavity with Dielectrics and Ferrites

This example illustrates a SUPERFISH run of a cavity loaded with several different dielectric and ferrite materials.

Figure C.12.2.1 shows the input file, FULLCAV, for the problem. There are 13 regions. The first region defines the extreme boundaries and the remaining regions define areas inside the extreme boundaries each of which has its own constant but individual value of permeability μ and permittivity ϵ. In each region a material number is set in the REG NAMELIST. Figure C.12.2.2 shows the terminal output resulting from the AUTOMESH run. The LATTICE run is shown in Fig. C.12.2.3.

```
full size cavity
$reg mat=1,nreg=13,npoint=15,xmin=0.,xmax=125.00,ymin=0.,ymax=25.00,
  dx= .50,dy=1.00$
$po x=0.0,y=0.0$
$po x=116.88,y=0.0$
$po x=116.88,y=8.0$
$po x=5.0,y=8.0$
$po x=5.0,y=9.0$
$po x=96.64,y=9.0$
$po x=116.88,y=13.0$
$po x=116.88,y=25.0$
$po x=96.64,y=25.0$
$po x=96.64,y=14.0$
$po x=95.64,y=13.0$
$po x=93.0,y=13.0$
$po x=85.0,y=17.5$
$po x=0.0,y=17.5$
$po x=0.0,y=0.0$
$reg mat=2,npoint=5$
$po x=96.64,y=16.0$
$po x=96.64,y=25.0$
$po x=99.18,y=25.0$
$po x=99.18,y=16.0$
$po x=96.64,y=16.0$
$reg mat=2,npoint=5$
$po x=100.18,y=16.00$
$po x=100.18,y=25.0$
$po x=102.72,y=25.0$
$po x=102.72,y=16.0$
$po x=100.18,y=16.0$
$reg mat=2,npoint=5$
$po x=103.72,y=16.00$
$po x=103.72,y=25.0$
$po x=106.26,y=25.0$
$po x=106.26,y=16.00$
$po x=103.72,y=16.00$
$reg mat=2,npoint=5$
$po x=107.26,y=16.00$
$po x=107.26,y=25.0$
$po x=109.80,y=25.0$
$po x=109.80,y=16.00$
$po x=107.26,y=16.00$
$reg mat=2,npoint=5$
$po x=110.80,y=16.00$
$po x=110.80,y=25.0$
$po x=113.34,y=25.0$
$po x=113.34,y=16.00$
$po x=110.80,y=16.00$
$reg mat=2,npoint=5$
$po x=114.34,y=16.00$
$po x=114.34,y=25.00$
$po x=116.88,y=25.00$
$po x=116.88,y=16.00$
$po x=114.34,y=16.00$
$reg mat=3,npoint=5$
$po x=96.64,y=14.0$
$po x=116.88,y=14.0$
$po x=116.88,y=14.75$
$po x=96.64,y=14.75$
$po x=96.64,y=14.0$
$reg mat=4,npoint=5$
$po x=99.18,y=16.$
$po x=100.18,y=16.$
$po x=100.18,y=25.$
$po x=99.18,y=25.$
$po x=99.18,y=16.$
$reg mat=4,npoint=5$
$po x=102.72,y=16.$
$po x=103.72,y=16.$
$po x=103.72,y=25.$
$po x=102.72,y=25.$
$po x=102.72,y=16.$
$reg mat=4,npoint=5$
$po x=106.26,y=16.$
$po x=107.26,y=16.$
$po x=107.26,y=25.$
$po x=106.26,y=25.$
$po x=106.26,y=16.$
$reg mat=4,npoint=5$
$po x=109.80,y=16.$
$po x=110.80,y=16.$
$po x=110.80,y=25.$
$po x=109.80,y=25.$
$po x=109.80,y=16.$
$reg mat=4,npoint=5$
$po x=113.34,y=16.$
$po x=114.34,y=16.$
$po x=114.34,y=25.$
$po x=113.34,y=25.$
$po x=113.34,y=16.$
```

Figure C.12.2.1: Input file for AUTOMESH.

```
automesh
  ?type input file name
?  fullcav

region no.  1
logical boundary segment end points
iseg    kb   lb    kd   ld    ke  le
   1     1    1     1    0   235   1
   2   235    1     0    1   235   9
   3   235    9    -1    0    11   9
   4    11    9     0    1    11  10
   5    11   10     1    0   195  10
   6   195   10     1    0   235  14
   7   235   14     0    1   235  26
   8   235   26    -1    0   195  26
   9   195   26    -1   -1   194  15
  10   194   15    -1    0   193  14
  11   193   14    -1    0   187  14
  12   187   14    -1    0   171  19
  13   171    9    -1    0     1  19
  14     1    9     0   -1     1   1

region no.  2
logical boundary segment end points
iseg    kb   lb    kd   ld    ke  le
  15   194   17     1    1   195  26
  16   195   26     1    0   200  26
  17   200   26    -1   -1   199  17
  18   199   17    -1    0   194  17

region no.  3
logical boundary segment end points
iseg    kb   lb    kd   ld    ke  le
  19   201   17     1    1   202  26
  20   202   26     1    0   207  26
  21   207   26    -1   -1   206  17
  22   206   17    -1    0   201  17

region no.  4
logical boundary segment end points
iseg    kb   lb    kd   ld    ke  le
  23   208   17     1    1   209  26
  24   209   26     1    0   214  26
  25   214   26    -1   -1   213  17
  26   213   17    -1    0   208  17

region no.  5
logical boundary segment end points
iseg    kb   lb    kd   ld    ke  le
  27   215   17     1    1   216  26
  28   216   26     1    0   221  26
  29   221   26    -1   -1   220  17
  30   220   17    -1    0   215  17

region no.  6
logical boundary segment end points
iseg    kb   lb    kd   ld    ke  le
  31   222   17     1    1   223  26
  32   223   26     1    0   228  26
  33   228   26    -1   -1   227  17
  34   227   17    -1    0   222  17

region no.  7
logical boundary segment end points
iseg    kb   lb    kd   ld    ke  le
  36   229   17     1    1   230  26
  36   230   26     1    0   235  26
  37   235   26     0   -1   235  17
  38   235   17    -1    0   239  17

region no.  8
logical boundary segment end points
iseg    kb   lb    kd   ld    ke  le
  39   194   15     1    0   235  15
  40   235   15     0    1   235  16
  41   235   16    -1    0   195  16
  42   195   16    -1   -1   194  15

region no.  9
logical boundary segment end points
iseg    kb   lb    kd   ld    ke  le
  43   199   17     1    0   201  17
  44   201   17     1    1   202  26
  45   202   26    -1    0   200  26
  46   200   26    -1   -1   199  17

region no. 10
logical boundary segment end points
iseg    kb   lb    kd   ld    ke  le
  47   206   17     1    0   208  17
  48   208   17     1    1   209  26
  49   209   26    -1    0   207  26
  50   207   26    -1   -1   206  17

region no. 11
logical boundary segment end points
iseg    kb   lb    kd   ld    ke  le
  51   213   17     1    0   215  17
  52   215   17     1    1   216  26
  53   216   26    -1    0   214  26
  54   214   26    -1   -1   213  17

region no. 12
logical boundary segment end points
iseg    kb   lb    kd   ld    ke  le
  55   220   17     1    0   222  17
  56   222   17     1    1   223  26
  57   223   26    -1    0   221  26
  58   221   26    -1   -1   220  17

region no. 13
logical boundary segment end points
iseg    kb   lb    kd   ld    ke  le
  59   227   17     1    0   229  17
  60   229   17     1    1   230  26
  61     1    1     0  235     1   1
stop
xauto  ctss time   1.636   seconds
cpu=  .735   sys=   .033  i/o+memory= . 868

  all done
```

Figure C.12.2.2: Log of interaction with AUTOMESII.

```
lattice

 ?type input file name
?  tape73

 beginning of lattice execution

 dump 0 will be set up for superfis

 1 full size cavity

 ?type input values for con(?)
?  s
 elapsed time  =    1.9 sec
0 iteration converged

 elapsed time  =    2.2 sec

 generation completed

 dump number 0 has been written on tape35.
 stop
lattice   ctss time     2.417     seconds
cpu=     1.895   sys=     .031   i/o+memory=      .491

  all done
```

Figure C.12.2.3: Log of interaction with LATTICE.

No CON's are changed since the only changes needed can be postponed until the SUPERFISH run. Figure C.12.2.4 shows the right end of the cavity where all the regions with different μ and ϵ are situated. This figure was obtained by using TEKPLOT and setting XMIN to 95. and XMAX to 125. so that only part of the region would be plotted. The full cavity is shown in Fig. C.12.2.5.

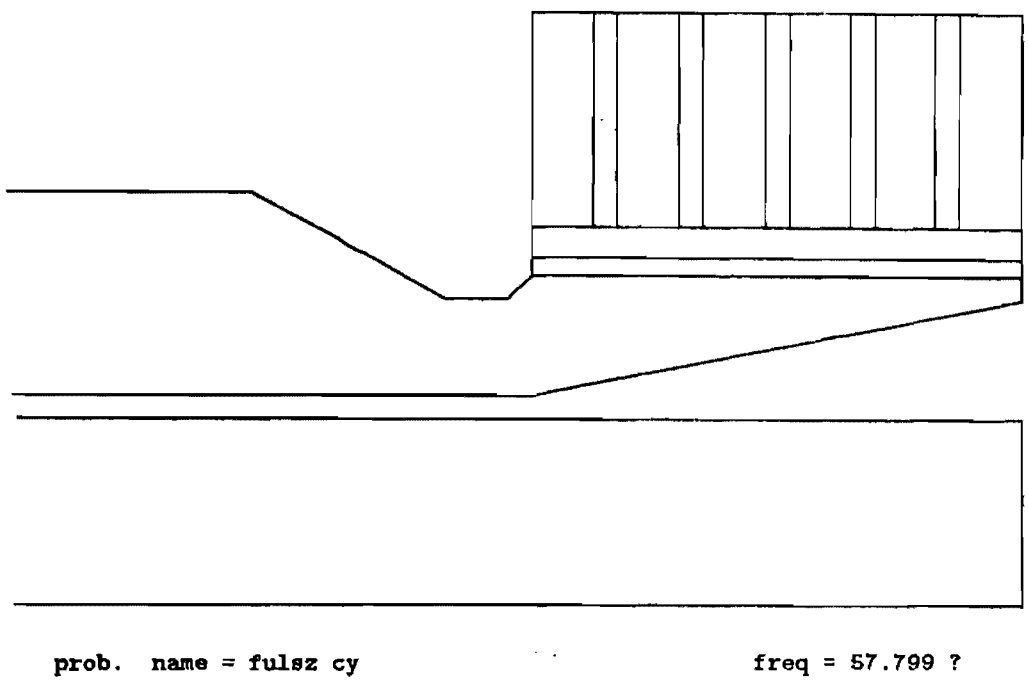
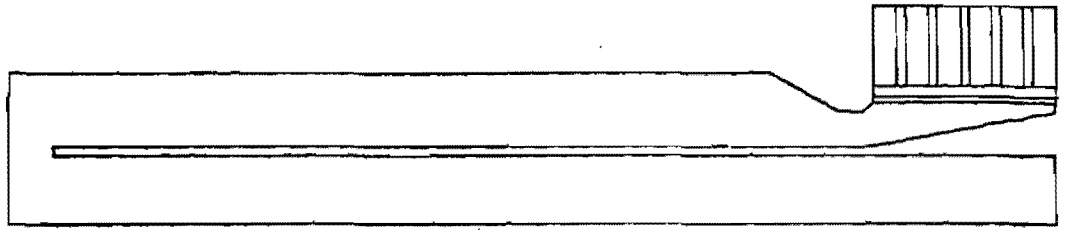


Figure C.12.2.4: TEKPLOT output for right end of cavity.

Figure C.12.2.6 shows the result of running SUPERFISH. As usual, we answer `tty` for the input and `0` for the dump number. For the CON changes we input `*18 3 *65 57.77 s`. The change in CON(18) tells the program there are to be 3 sets of relative ϵ and relative μ values to be input. CON(65) is, of course, the starting frequency. Because CON (18) was set to 3, the program asks for MATER, EPSILON, MU three times. Each time the reply consists of the material number, the relative dielectric constant κ_e and the relative permeability κ_m. After the third line, the code runs the problem. Using TEKPLOT, we can look at the field pattern in the cavity. Figure C.12.2.7 shows a blowup of the right end of the cavity and C.12.2.8 shows the whole cavity.

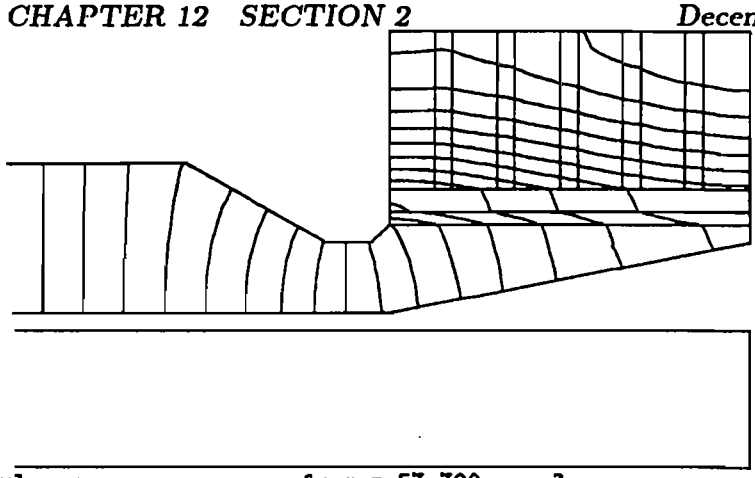


prob. name = fulsz cy freq = 57.799 ?

Figure C.12.2.5: TEKPLOT output for full length of cavity.

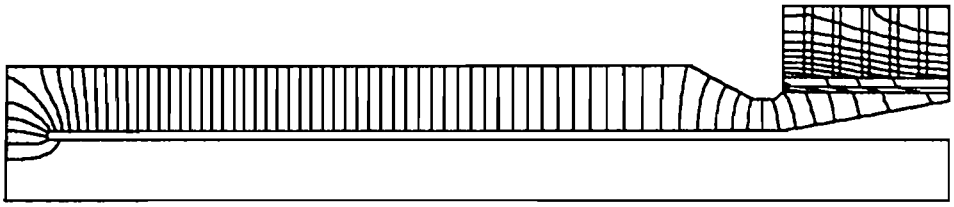
```
fish
 ?type "tty" or input filename
?  tty
 ?type input value for dump num
?  0
 beginning of superfish execution from dump number 0

 prob.  name = full size cavity

 ?type input values for con(?)
?  *18 3 *65 57.77 s

 ?type input for mater, epsilon, mu
?  2 14.5 1.5

 ?type input for mater, epsilon, mu
?  3 10.0 1.0

 ?type input for mater, epsilon, mu
?  4 10.0 1.0

 elapsed time = 3.7 sec.
 cycle       hmin          hmax          residual
    0     0.0000e+00  0.0000e+00     1.0000e+00

          ------------------------------------------------------------
                                                   k**2 =  1.4660e-04
                                                    freq =  5.7770e+01
 solution time =   9.312 sec.

    1   0.0000e+00  2.2981e+00     1.0000e+00
 kfix =235   lfix = 26 delta1 =  1.5310e-03 d1(k**)=  1.4885e-07
 ------------------------------------------------------------------
                         using slope = -1 formula with rix =1.0000
 del k**2 =  1.4885e-07   k**2 =    1.4675e-04   freq =  5.7799e+01
 solution time =   9.876 sec.

    1    0.0000e+00   2.3089e+00   1.0000e+00
 kfix =235   lfix = 26  delta1 = -7.0304e-06  d1(k**)2= -6.7742e-10
 ------------------------------------------------------------------
                                          delta1(k**2)     d1(k**2)
                              1st deriv.    -1.0332e+04   -1.0045e+00
                         using two point secant formula
 del k**2 =   -6.7417e-10    k**2 =  1.467e-04    freq =  5.7799e+01

 solution converged in       2 iterations

 dump number  1 has been written.

 ?type input value for dump num
?
```

Figure C.12.2.6: Log of interaction with SUPERFISH.



prob. name = fulsz cy freq = 57.790 ?

Figure C.12.2.7: TEKPLOT output for field in right end of cavity.



prob. name = fulsz cy freq = 57.790 ?

Figure C.12.2.8: TEKPLOT output of field in full cavity.

Figure C.12.2.9 shows the terminal output resulting from running SFO1. We tell the program to get input from "tty" and SUPERFISH output data from dump 1. CON(50) is set to 10 to get power and frequency shifts on 10 segments in the problem. The program requests the numbers of the segments and is given 10 segment numbers. The program then executes.

More information on the results of running the various codes can be found on the output files OUTAUT for AUTOMESH, OUTLAT for LATTICE, OUTFIS for SUPERFISH and OUTSFO for SFO1.

```
sfo1
 ?type "tty" or input file name
?  tty
 ?type input value for dump num
?  1
 ?beginning of superfisch execution from dump number 1

 ?prob.  name = full size cavity
 ?type input value for con(?)
?  *50 10 s

 ?type input value for iseg's
?  3,4,5,6,7,8,9,10,11,12,13

1
 superfish output summary      10:23:39   84/09/24

 problem name =

 cavity length = 233.760 cm,      diameter        =       50.000 cm

 frequency (starting value= 57.770)               =       57.799 mhz

 beta=  0.4507                  proton energy     =      112.792 mev

 normalize factor  (e0=1.0 mv/m)  ascale          =       81167.6

 stored energy  (for problem geometry)            =       38.7417 joules

 stored energy    (full cavity)                   =       77.4834 joules

 power dissipation  (for problem geometry)        =    1353630.48 watts

 power dissipation  (full cavity)                 =    2707260.95 watts

 t,tp,tpp,s,sp,spp  =   0.984    0.005    0.001    0.137    0.021    0.000

 q=    10393                 shunt impedance     =         0.863 mohm/m

 product =*t**2                           *tt     =         0.837 mohm/m

 magnetic field on outer wall                     =         21545 amp/m

 maximum electric field on boundary               =        54.844 mv/m
```

Figure C.12.2.9: Log of interaction with SFO1.

C.12.3 Electron Linac Cavity

This example calculates the resonant frequency and other properties of a cavity proposed for an electron linac. Figure C.12.3.1 shows the AUTOMESH input file named ECELL. In this problem, the units are inches so the variable CONV is set to 2.54, the number of centimeters per inch, in the REG NAMELIST. The height of the mesh triangles is approximately doubled at 1.75 inches by setting YREG1 = 1.75. The eleven PO NAMELIST cards give the successive endpoints of the boundary segments.

```
1 1 electron linac cell
2  $reg nreg=1,dx=.035,xmax=2.3,ymax=3.35,npoint=11,
3   conv=2.54,yreg1=1.75 $
4  $po x=0., y=0. $
5  $po x=0.0, y=3.2384 $
6  $po x=.4967, y=3.2365 $
7  $po nt=2,x0=.375,y0=1.5542,x=1.675,y=-0.1937 $
8  $po nt=2,x0=1.8,y0=1.3505,x=0.128,y=-0.2147 $
9  $po x=1.006,y=0.6 $
10 $po nt=2,x0=1.006,y0=.55,r=.05,theta=270. $
11 $po x=2.269, y=0.5 $
12 $po x=2.269, y=0.0 $
13 $po x=1.006, y=0.0 $
14 $po x=0.0, y=0.0 $
```

Figure C.12.3.1: Input file for AUTOMESH.

The next step in solving the problem is to run AUTOMESH. We type **automesh**, the name of the AUTOMESH executable file. The code asks for the input file name and is given **ecell**. AUTOMESH runs producing Fig. C.12.3.2 at the terminal.

```
automesh

 ?type input file name
ecell

 region no.  1
 logical boundary segment end points
 iseg     kb    lb       kd    ld       ke    le
    1      1     1        0     1        1    59
    2      1    59        0     1        1    84
    3      1    84        1     0       16    84
    4     16    84        1     0       60    59
    5     60    59        0    -1       60    46
    6     60    46        0    -1       56    39
    7     56    39        0    -1       30    21
    8     30    21       -1     0       30    18
    9     30    18        1     0       66    17
   10     66    17        0    -1       66     1
   11     66     1       -1     0       30     1
   12     30     1       -1     0        1     1
 stop
automesh  ctss time          .365     seconds
cpu=     .172    sys=        .023    i/o+memory=        .169

 all done
```

Figure C.12.3.2: Log of interaction with AUTOMESH.

To finish setting up the mesh, we run LATTICE. Figure C.12.3.3 shows the LATTICE execution listing.

```
lattice
 ?type input file name
tape73

beginning of lattice execution

dump 0 will be set up for superfis

1 electron linac cell

 ?type input values for con(?)
?  s

 elapsed time  =    0.8 sec.
0iteration converged

  elapsed time  =    2.3 sec.

 generation completed

 dump number 0 has been written on tape35
 stop
lattice  ctss time     2.597     seconds
cpu=    2.150    sys=    .028   i/o+memory=              .419

  all done
```

Figure C.12.3.3: Log of interaction with LATTICE.

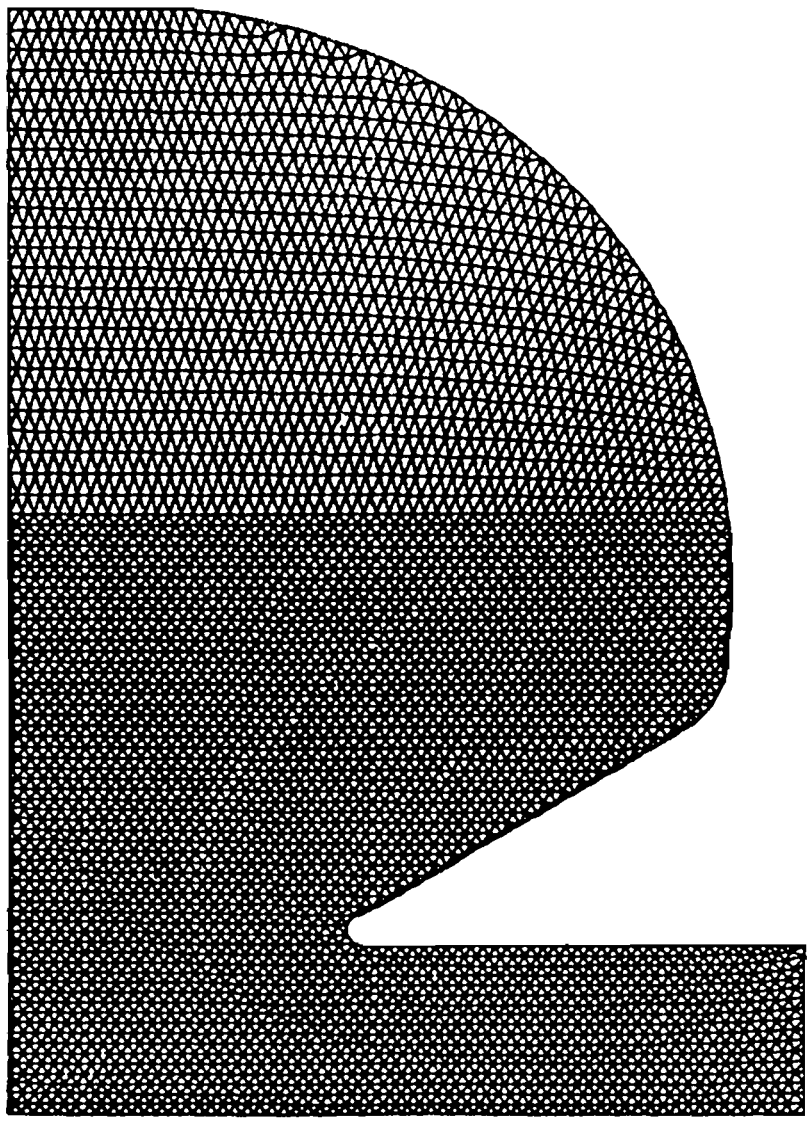
When asked, the code is told that the input is from TAPE73. When it asks for new values for the CON's, no changes are necessary so the code is told to proceed by typing an "s" followed by a carriage return. LATTICE executes with no problems.

We decide to look at the mesh to make sure we have set up the proper problem. We type the TEKPLOT executable file name **tekplot** (See Fig. C.12.3.4). The code asks for input data and we reply that we want dump NUM = 0 and to show the mesh (ITR1 = 1). The line is terminated by an "s" to tell the code to use the default values for the remaining input. The code then shows the values it is going to use and asks for the $x - y$ limits on the plotting area. The reply "s" tells the

```
tekplot

 ?type input data- num, itri, nphi, inap, nswxy,
?  0 1 s

 input data
 num=  0    itri= 1    nphi=    0    inap= 0    nswxy= 0

 plotting prob. name =  electron linac cell

 ?type input data- xmin, xmax, ymin, ymax,
?  s
 input data
 xmin=   0.0000  xmax=    2.269  ymin=    0.0000  ymax=    3.238

 ?type go or no
?  go
```

Figure C.12.3.4: Log of interaction with TEKPLOT for mesh.

code to use the minimum and maximum values in the mesh. The code prints these values and asks if it should proceed. We reply "go;" the code blanks the screen and produces Fig. C.12.3.5. To end TEKPLOT hit the carriage return and type **-1 s**.



```
prob.  name = electron linac cell              freq =       0.000
```

Figure C.12.3.5: TEKPLOT output showing mesh.

To proceed with the solution, we type **fish** (See Fig. C.12.3.6), which is the executable file name of SUPERFISH. The program asks for "tty" or the name of an input file and is told **tty**. The designation "tty" is short for "teletype", which means typing input from the terminal.

```
fish

?type "tty" or input file name
?  tty

?type input value for dump num
?  0

 beginning of superfish execution from dump number   0

 prob. name  =   electron linac cell

 ?type input values for con(?)
?  *65 1240.  s

 elapsed time  =    1.7 sec.

 cycle      hmin         hmax       residual

     0   0.0000e+00   0.0000e+00   1.000e+00
         ------------------------------------------------------------------
                                                          k**2 =  6.7540e-02
                                                          freq =  1.2400e+03
 solution time =    3.804 sec.

    1    0.0000e+00    1.3929e+00     1.000e+00
 kfix =  60    lfix = 59   delta1 =  1.0906e-03   d1(k**2)=   9.6243e-05
         ------------------------------------------------------------------
                              using slope = -1 formula with rlx = 1.000
 del k**2  =  9.6243e-05    k**2 =   6.7636e-02     freq  =   1.2409e+03
 solution time =     3.623 sec.

    2     0.0000e+00   1.3973e+00    1.000e+00
 kfix  =  60   lfix  =  59  delta1 = -4.1411e-06  d1(k**2)=  -3.6272e-07
 ------------------------------------------------------------------------
                                              delta1(k**2)      d1(k**2)
                                     1st deriv.   -1.1375e+01   -1.0038e+00
                              using two point secant formula
 del k**2  =  -3.6136e-07     k**2 =   6.7636e-02    freq =   1.2409e+03
solution converged in      2 iterations

elapsed time = 9.4 sec.

dump number    1 has been written.

?type input value for dump num
?
```

Figure C.12.3.6: Log of interaction with SUPERFISH.

It asks for the dump number and is told 0. The next request is for CON-changes and we type *65 1240. s to tell it to start its iteration at 1240 MHz. The program requires two iterations to find the resonant frequency at 1240.9 MHz. We exit SUPERFISH by typing -1 s.

To use dump 1 to determine properties of the cavity we type **sfo1**. SFO1 asks for either "tty" or an input file name. Given "**tty**," the program asks for a dump number; we enter 1. When the program asks for CON's we change CON(50) to 9 and indicate this is all by typing a **s**.

CON(50) is the number of boundary segments (See **iseg** in Fig. C.12.3.2) on which we want power and frequency shifts calculated. SFO1 prints a summary of the results at the terminal when we answer its question with a "**go**" (See Fig. C.12.3.7). More details of the results can be found in OUTSFO.

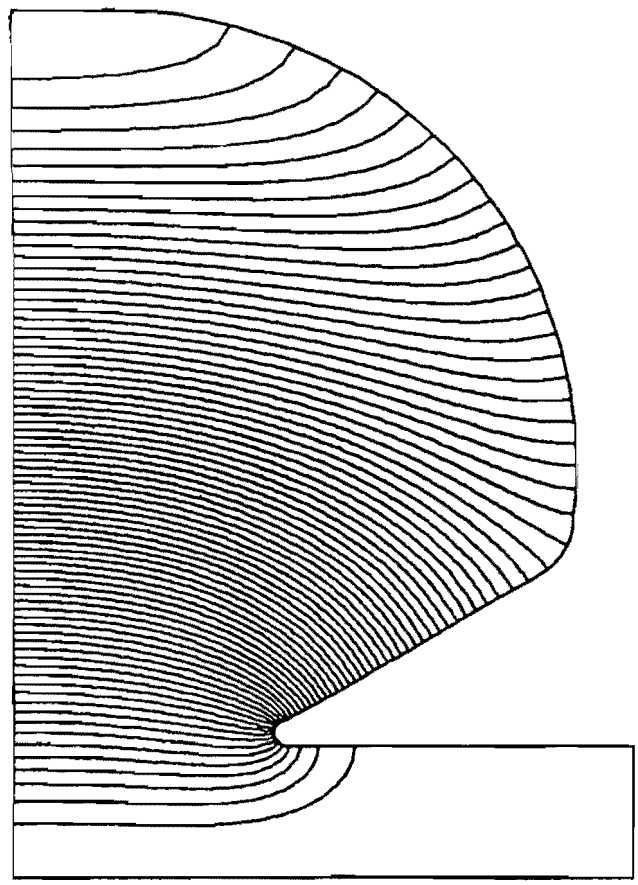
```
sfo1
?type "tty" or inpuit file name
?  tty
?type input value for num
?  1

beginning of sfo1 execution from dump number 1

 prob. name = electron linac cell

?type input values for con(?)
?  *50 9 s

?type input values for iseg's
?  2 3 4 5 6 7 8 9 10

?type go for output summary at terminal
?  go

 superfish dtl output summary          10:03:09   84/09/25
 problem name = electron linac cell
 cavity length  =  11.527 cm       cavity diameter     =     0.000 cm
       d.t. gap =  10.414 cm           stem radius     =     1.000 cm
 frequency (starting value =1240.000)                  =  1240.880 mhz
 beta =  0.4771                       proton energy    =   129.335 mev
 normalization factor    (e0=1 mv/m)       ascale      =    8555.7
 stored energy (mesh problem only)                     =    0.0021 joules
 power dissipation (mesh problem only)                 =    950.41 watts
 t,tp,tpp,s,sp,spp  =   0.583    0.109    0.004    0.640    0.045    0.022
 q  =   17080                      shunt impedance     =     60.64 mohm/m
 product z*t**2                                 ztt   =     20.60 mohm/m
 magnetic field on outer wall                          =      1623 amp/m
 maximum electric field on boundary                    =     7.156 mv/m

 iseg  zbeg   rbeg   zend   rend   emax     power          d-freq   d-freq
       (cm)   (cm)   (cm)   (cm)   (mv/m)    (w)           (delz)   (delr)
   2  0.000  4.445  0.000  8.226  0.7396  2.30e+02 wall    -6.7080  0.0000
   3  0.000  8.226  1.262  8.219  0.0158  7.82e+01 wall    -0.0155 -3.2060
   4  1.262  8.219  5.206  4.445  0.2704  3.68e+02 wall    -9.1618 -9.7270
   5  5.206  4.445  5.207  3.456  0.3011  6.77e+01 wall    -2.4970 -0.1441
   6  5.207  3.456  4.897  2.885  0.3682  5.16e+01 wall    -1.6527 -0.9099
   7  4.897  2.885  2.555  1.524  3.6447  1.52e+02 wall     3.6095  6.2115
   8  2.555  1.524  2.555  1.270  7.1559  2.81e+00 wall    12.1550  0.0000
   9  2.555  1.270  5.763  1.270  2.3364  2.38e-01 wall     0.0000  0.9841
  10  5.763  1.270  5.763  0.000  0.0107  9.38e-06 wall     0.0000  0.0000
 ?type input value for num
?
 ---input data error--- illegal character

 retype line
-1s
 stop
sfo1     ctss time      .786     seconds
cpu=      .209   sys=     .037  i/o+memory=      .540

 all done
```

Figure C.12.3.7: Output of SFO1 for electron linac problem.

TEKPLOT can be used to look at the field pattern in the cavity. This time dump 1 is used and NPHI is set to 75. The results are shown in Figs. C.12.3.8 and C.12.3.9.

```
tekplot
 ?type input data- num, itri, nphi, inap, nswxy,
?  1 0 75 s
 input data
 num= 1 itri= 0 nphi= 75 inap= 0 nswxy= 0

 plotting prob.  name = electron linac cell

 ?type input data- xmin, xmax, ymin, ymax,
?  s
 input data
 xmin= 0.000 xmax= 2.269 ymin= 0.000 ymax= 3.238

 ?type go or no
?  go
```

Figure C.12.3.8: Log of interaction with TEKPLOT.



prob. name = electron linac cell freq = 1240.880 ?

Figure C.12.3.9: TEKPLOT output for field distribution.

Chapter C.13

APPENDICES

C.13.1 RF Cavity Theory

This section summarizes the theory behind SUPERFISH. The problem is to find the electromagnetic resonance frequencies and evaluate the field components in a cavity surrounded by perfectly conducting walls. As with POISSON, there are two geometries that can be handled: three-dimensional with cylindrical symmetry and two-dimensional cartesian symmetry. The theory will be presented for cylindrical symmetry; at the end we will indicate the modifications for cartesian coordinates.

Although there are no real currents or charges in the cavity, we are going to introduce a fictitious magnetic current density $\mathbf{K}$, and magnetic charge density σ which will "drive" the fields in the cavity. At resonance, the amount of current needed to drive the cavity should approach zero. Something like this is used to determine the resonance frequency in the iteration scheme.

We shall assume that the medium in the cavity is homogeneous, isotropic, non-conducting, with piecewise constant permittivity and permeability so that

$$\mathbf{D} = \epsilon \mathbf{E}, \tag{C.13.1.1}$$

$$\mathbf{B} = \mu \mathbf{H}. \tag{C.13.1.2}$$

With cylindrical symmetry $\mathbf{E}, \mathbf{H}, \mathbf{K}$, and σ must be independent of θ.

Maxwell's equations can be written in two sets, which are

$$\left[\nabla \times \mathbf{E} + \frac{\partial \mathbf{B}}{\partial t}\right]_r = K_r \text{ or } -\frac{\partial E_\theta}{\partial z} + \mu \frac{\partial H_r}{\partial t} = K_r, \tag{C.13.1.3}$$

$$\left[\nabla \times \mathbf{E} + \frac{\partial \mathbf{B}}{\partial t}\right]_z = K_z \text{ or } \frac{1}{r}\frac{\partial (rE_\theta)}{\partial r} + \mu\frac{\partial H_z}{\partial t} = K_z, \tag{C.13.1.4}$$

$$\left[\nabla \times \mathbf{H} - \frac{\partial \mathbf{D}}{\partial t}\right]_\theta = 0 \text{ or } \frac{\partial H_r}{\partial z} - \frac{\partial H_z}{\partial r} - \epsilon\frac{\partial E_\theta}{\partial t} = 0, \tag{C.13.1.5}$$

$$\nabla \cdot \mathbf{B} = \sigma \text{ or } \frac{1}{r}\frac{\partial (rH_r)}{\partial r} + \frac{\partial H_z}{\partial z} = \frac{\sigma}{\mu}, \tag{C.13.1.6}$$

and

$$\left[\nabla \times \mathbf{H} - \frac{\partial \mathbf{D}}{\partial t}\right]_r = 0 \text{ or } -\frac{\partial H_\theta}{\partial z} - \epsilon\frac{\partial E_r}{\partial t} = 0, \tag{C.13.1.7}$$

$$\left[\nabla \times \mathbf{H} - \frac{\partial \mathbf{D}}{\partial t}\right]_z = 0 \text{ or } \frac{1}{r}\frac{\partial (rH_\theta)}{\partial r} - \epsilon\frac{\partial E_z}{\partial t} = 0, \tag{C.13.1.8}$$

$$\left[\nabla \times \mathbf{E} + \frac{\partial \mathbf{B}}{\partial t}\right]_\theta = K_\theta \text{ or } \frac{\partial E_r}{\partial z} - \frac{\partial E_z}{\partial r} + \mu\frac{\partial H_\theta}{\partial t} = K_\theta, \tag{C.13.1.9}$$

$$\nabla \cdot \mathbf{D} = 0 \text{ or } \frac{1}{r}\frac{\partial (rE_r)}{\partial r} + \frac{\partial E_z}{\partial z} = 0. \tag{C.13.1.10}$$

Note that the first four equations involve the field components (H_r, E_θ, H_z), while the last four are nearly identical but involve (E_r, H_θ, E_z). This corresponds to a separation into transverse electric (TE) modes for which $E_\theta \neq 0$ and transverse magnetic (TM) modes for which $H_\theta \neq 0$. It is usually the TM modes of the cavity that are of interest to accelerator designers, because they have $E_z \neq 0$ on the cylindrical axis.

Equations (C.13.1.3) through (C.13.1.5) can be combined to give a second-order partial differential equation for E_θ alone. Differentiate Eq. (C.13.1.3) by z; differentiate Eq. (C.13.1.4) by r; subtract the two results; and make use of Eq. (C.13.1.5) to eliminate the combination $(\partial H_r/\partial z - \partial H_z/\partial r)$.

The result is

$$-\frac{\partial}{\partial r}\left[\frac{1}{r}\frac{\partial}{\partial r}(rE_\theta)\right] - \frac{\partial^2 E_\theta}{\partial z^2} + \mu\epsilon\frac{\partial^2 E_\theta}{\partial t^2} = \left(\frac{\partial K_r}{\partial z} - \frac{\partial K_z}{\partial r}\right) = [\nabla \times \mathbf{K}]_\theta. \tag{C.13.1.11}$$

Similarly we can obtain an equation for H_θ by using Eqs. (C.13.1.7) through (C.13.1.9); the result is

$$-\frac{\partial}{\partial r}\left[\frac{1}{r}\frac{\partial}{\partial r}(rH_\theta)\right] - \frac{\partial^2 H_\theta}{\partial z^2} + \mu\epsilon\frac{\partial^2 H_\theta}{\partial t^2} = \epsilon\frac{\partial K_\theta}{\partial t}. \tag{C.13.1.12}$$

We are interested in solutions that are periodic in time. Let us arbitrarily assume that

$$\mathbf{K}(r,z,t) = \overline{\mathbf{K}}(r,z)\sin\omega t. \tag{C.13.1.13}$$

It then must follow from Eqs. (C.13.1.11) and (C.13.1.12) that we can write

$$E_\theta(r,z,t) = \overline{E}_\theta(r,z)\sin\omega t, \tag{C.13.1.14}$$

and

$$H_\theta(r,z,t) = \sqrt{\frac{\epsilon}{\mu}}\overline{H}_\theta(r,z)\cos\omega t. \tag{C.13.1.15}$$

This definition of $\overline{H}_\theta$ makes $\overline{E}_\theta$ and $\overline{H}_\theta$ have the same dimensions. As a result, the same coding can be used for both the TE and TM modes in SUPERFISH.

When these assumptions are put into Eqs.(C.13.1.11) and (C.13.1.12), the results are

$$\nabla^2\overline{E}_\theta - \frac{1}{r^2}\overline{E}_\theta + k^2\overline{E}_\theta = -\left[\nabla\times\overline{\mathbf{K}}\right]_\theta, \tag{C.13.1.16}$$

$$\nabla^2\overline{H}_\theta - \frac{1}{r^2}\overline{H}_\theta + k^2\overline{H}_\theta = -\epsilon\omega\overline{K}_\theta = -\sqrt{\frac{\epsilon}{\mu}}k\overline{K}_\theta, \tag{C.13.1.17}$$

where

$$k = \sqrt{\mu\epsilon}\omega \tag{C.13.1.18}$$

is called the eigenvalue and

$$\nabla^2 f = \frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial f}{\partial r}\right) + \frac{\partial^2 f}{\partial z^2} \tag{C.13.1.19}$$

is the two-dimensional Laplacian in cylindrical coordinates. Given $\overline{E}_\theta$ and $\overline{H}_\theta$ from these equations, one can use Eqs. (C.13.1.3) through (C.13.1.8) to find H_r, H_z, E_r, and E_z. The integration over time is trivial. The constants of integration just determine the initial phase of the fields at $t = 0$ and can be set equal to zero for our purposes. The results are

$$H_r = -\sqrt{\frac{\epsilon}{\mu}}\frac{1}{k}\left(\frac{\partial\overline{E}_\theta}{\partial z} + \overline{K}_r\right)\cos\omega t, \tag{C.13.1.20}$$

$$H_z = \sqrt{\frac{\epsilon}{\mu}}\frac{1}{k}\left[\frac{1}{r}\frac{\partial}{\partial r}\left(r\overline{E}_\theta\right) - \overline{K}_z\right]\cos\omega t, \tag{C.13.1.21}$$

$$E_r = -\sqrt{\frac{\epsilon}{\mu}}\frac{1}{k}\frac{\partial\overline{H}_\theta}{\partial z}\sin\omega t, \tag{C.13.1.22}$$

$$E_z = -\sqrt{\frac{\epsilon}{\mu}}\frac{1}{k}\left[\frac{1}{r}\frac{\partial}{\partial r}(r\overline{H}_\theta)\right]\sin\omega t. \tag{C.13.1.23}$$

It is easily seen that Eq. (C.13.1.10) is identically satisfied and that Eq. (C.13.1.6) is satisfied if

$$\nabla \cdot \overline{\mathbf{K}} \sin \omega t = -\frac{\partial \sigma(r,z,t)}{\partial t}. \tag{C.13.1.24}$$

This is just the equation of continuity for magnetic current with the magnetic charge given by

$$\sigma(r,z,t) = \overline{\sigma}(r,z) \cos \omega t. \tag{C.13.1.25}$$

Note that in the TM mode, the electric field lines are parallel to the lines of constant rH_θ, which can be seen as follows. A field line is a curve $r(z)$ whose tangent is proportional to the ratio of the electric field components, thus

$$\frac{dr}{dz} = \frac{E_r}{E_z} = \frac{\frac{-\partial H_\theta}{\partial z}}{\frac{1}{r}\frac{\partial}{\partial r}(rH_\theta)}. \tag{C.13.1.26}$$

This implies that

$$\frac{1}{r}\frac{\partial}{\partial r}(rH_\theta)\, dr + \frac{\partial H_\theta}{\partial z} dz = 0, \tag{C.13.1.27}$$

or, multiplying through by r,

$$\nabla (rH_\theta) \cdot d\mathbf{r} = 0, \tag{C.13.1.28}$$

which implies that rH_θ is a constant along an electric field line. This result is used in TEKPLOT.

It is helpful in understanding the Halbach and Holsinger paper[2] to apply the Poynting theorem[16] to the cavity fields. Poynting's theorem in this case can be written as

$$\oint \mathbf{E} \times \mathbf{H} \cdot d\mathbf{a} + \frac{\partial}{\partial t} \int \frac{1}{2}\left[\epsilon E^2 + \mu H^2\right] dv = \int \mathbf{H} \cdot \mathbf{K} dv, \tag{C.13.1.29}$$

where the first term on the left is interpreted as the flow of energy out across the cavity surface **a**, enclosing the volume v. The second term on the left is the change in energy of electromagnetic fields in the enclosed volume. The term on the right is the rate of work being done on the field by the magnetic current. Let us introduce the time-dependence and carry out the time derivative. Since the cavity is closed, the surface integral must vanish. The result is

$$\int \sqrt{\frac{\epsilon}{\mu}} \overline{\mathbf{H}} \cdot \overline{\mathbf{K}} = \omega \int \epsilon \left(\overline{E}^2 - \overline{H}^2\right) dv, \tag{C.13.1.30}$$

where

$$\overline{E}^2 = \overline{E}_r^2 + \overline{E}_\theta^2 + \overline{E}_z^2, \tag{C.13.1.31}$$

and similarly for $\overline{H}^2$. This is the generalized version of Eq. (8) in Ref. 2 for cavities containing dielectrics or permeable material. This differs from the result of Ref. 2 in that it is no longer possible to take k out of the integral because ϵ and μ need not be constant; only ω can come out of the integral.

Equation (C.13.1.30) gives us an ω-dependent quantity proportional to the fictitious magnetic current that can be used to determine when resonance occurs. The program uses the normalized quantity

$$D\left(\omega^2\right) = \omega \int \overline{\mathbf{H}} \cdot \overline{\mathbf{K}} dv / \int \epsilon \overline{H}^2 dv \equiv R\left(\omega^2\right) - \omega^2. \tag{C.13.1.32}$$

From Eq.(C.13.1.30), it is easily seen that

$$R\left(\omega^2\right) = \omega^2 \int \epsilon \overline{E}^2 dv / \int \epsilon \overline{H}^2 dv. \tag{C.13.1.33}$$

Resonance occurs at a value of ω for which $D(\omega^2) = 0$, which implies that no magnetic current is required to maintain fields in the cavity. It also means that $R(\omega^2) = \omega^2$, which implies that the energy stored in the electric field is equal to the energy in the magnetic field.

It turns out that this criterion is not sufficient to determine the resonances. It is also necessary that $dD(\omega^2)/d\omega^2 = -1$. Between each true root of $D(\omega^2)$ there is a false root where the slope is + 1. This can be seen as follows. Let the derivative with respect to ω^2 be denoted by a prime,

$$\frac{df}{d\omega^2} = f'. \tag{C.13.1.34}$$

In vector form, after the time dependence has been removed, Maxwell's equations become

$$\nabla \times \overline{\mathbf{E}} - \sqrt{\epsilon\mu}\, \omega \overline{\mathbf{H}} = \overline{\mathbf{K}}, \tag{C.13.1.35}$$

$$\nabla \times \overline{\mathbf{H}} - \sqrt{\epsilon\mu}\, \omega \overline{\mathbf{E}} = \mathbf{0}. \tag{C.13.1.36}$$

This implies that

$$\nabla \times \overline{\mathbf{E}}' - \sqrt{\epsilon\mu} \left(\frac{\overline{\mathbf{H}}}{2\omega} + \omega \overline{\mathbf{H}}' \right) = \overline{\mathbf{K}}', \tag{C.13.1.37}$$

$$\nabla \times \overline{\mathbf{H}}' - \sqrt{\epsilon\mu} \left(\frac{\overline{\mathbf{E}}}{2\omega} + \omega \overline{\mathbf{E}}' \right) = \mathbf{0}. \tag{C.13.1.38}$$

If now we calculate

$$\nabla \cdot \left(\overline{\mathbf{E}} \times \overline{\mathbf{H}}' - \overline{\mathbf{E}}' \times \overline{\mathbf{H}} \right) = \overline{\mathbf{H}}' \cdot \nabla \times \overline{\mathbf{E}} - \overline{\mathbf{E}} \cdot \nabla \times \overline{\mathbf{H}}' - \overline{\mathbf{H}} \cdot \nabla \times \overline{\mathbf{E}}' + \overline{\mathbf{E}}' \cdot \nabla \times \overline{\mathbf{H}}, \tag{C.13.1.39}$$

one finds that

$$\nabla \cdot \left(\overline{\mathbf{E}} \times \overline{\mathbf{H}}' - \overline{\mathbf{E}}' \times \overline{\mathbf{H}}\right) = \overline{\mathbf{H}}' \cdot \overline{\mathbf{K}} - \overline{\mathbf{H}} \cdot \overline{\mathbf{K}}' - \frac{\sqrt{\epsilon\mu}}{2\omega}\left(\overline{E}^2 + \overline{H}^2\right). \qquad \text{(C.13.1.40)}$$

If now we integrate over the volume of the cavity and note that

$$\int \nabla \cdot \left(\overline{\mathbf{E}} \times \overline{\mathbf{H}}' - \overline{\mathbf{E}}' \times \overline{\mathbf{H}}\right) dv = \oint \left(\overline{\mathbf{E}} \times \overline{\mathbf{H}}' - \overline{\mathbf{E}}' \times \overline{\mathbf{H}}\right) \cdot d\mathbf{a} \qquad \text{(C.13.1.41)}$$

can be made to vanish with the proper boundary conditions, then

$$\int \sqrt{\frac{\epsilon}{\mu}} \left(\overline{\mathbf{H}} \cdot \overline{\mathbf{K}}' - \overline{\mathbf{H}}' \cdot \overline{\mathbf{K}}\right) dv = -\int \frac{\epsilon}{2\omega}\left(\overline{E}^2 + \overline{H}^2\right) dv. \qquad \text{(C.13.1.42)}$$

If we let the fictitious current $\overline{\mathbf{K}}$ vary with ω^2 in such a way that $\overline{\mathbf{H}}^2$ is not changed as we approach resonance, then $\overline{H}' = 0$ and we get the formula

$$\int \sqrt{\frac{\epsilon}{\mu}}\, \overline{\mathbf{H}} \cdot \overline{\mathbf{K}}' dv = -\frac{1}{2\omega}\int \epsilon \left(\overline{E}^2 + \overline{H}^2\right) dv. \qquad \text{(C.13.1.43)}$$

With the same assumption, take the derivative of $D(\omega^2)$ in Eq. (C.13.1.32); the result is

$$D'(\omega) = \frac{1}{2\omega^2} D + \omega \int \sqrt{\frac{\epsilon}{\mu}} \overline{\mathbf{H}} \cdot \overline{\mathbf{K}}' dv \Big/ \int \epsilon \overline{H}^2 dv, \qquad \text{(C.13.1.44)}$$

or

$$D'(\omega) = \frac{1}{2\omega^2} D - \frac{1}{2}\int \epsilon \left(\overline{E}^2 + \overline{H}^2\right) dv \Big/ \int \epsilon \overline{H}^2 dv. \qquad \text{(C.13.1.45)}$$

At resonance, $D = 0$ and the electric energy equals the magnetic energy so that

$$D'(\omega_{res}) = -1. \qquad \text{(C.13.1.46)}$$

The program uses the fact that $D' < 0$ to improve convergence and to discriminate between real and false resonances.

Suppose ω_1 and ω_2 are two adjacent resonant frequencies. At these frequencies, $D = 0$ and $D' = -1$. This is illustrated in Fig. C.13.1.1. If D is a continuous function, somewhere between ω_1 and ω_2 there must be a place where $D = 0$ but $D' > 0$, which is the false resonance root.

The modifications to the above theory for application to waveguides and cross sections of a Radio-Frequency Quadrupole (RFQ) are straightforward. All that must be done is to replace the Laplacian given in cylindrical coordinates by the Laplacian given in cartesian coordinates. Figure C.13.1.2 shows a waveguide of arbitrary but

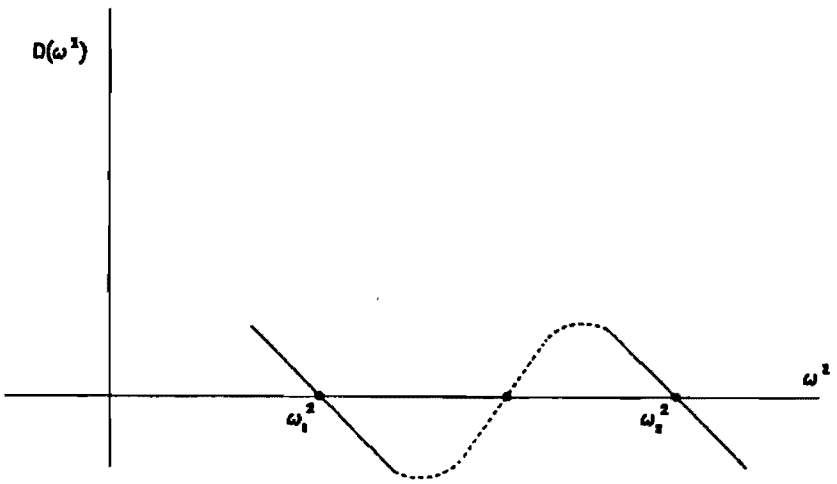


Figure C.13.1.1: Demonstration that $D(\omega^2)$ must have false zero.

uniform cross section. The directions of the coordinate axes are indicated on the figure.

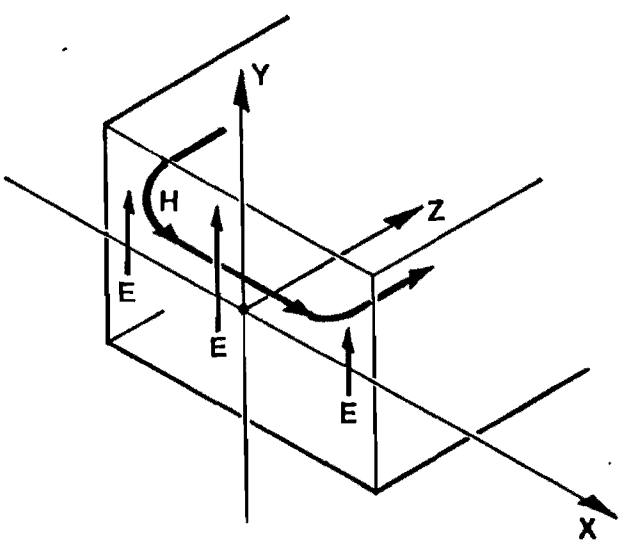


Figure C.13.1.2: The coordinate system for a waveguide with arbitrary but uniform cross section.

The only waveguide modes that can be calculated by SUPERFISH are the TE and TM cutoff modes, namely, those modes that have zero propagation vector along the z-axis. Fortunately these are the modes of interest in the design of an RFQ. Let us repeat the derivation given at the beginning of the section with a slight variation. Once again, we start with Maxwell's equations

$$\nabla \times \mathbf{H} - \frac{\partial \mathbf{D}}{\partial t} = 0, \qquad \nabla \cdot \mathbf{D} = 0, \tag{C.13.1.47}$$

$$\nabla \times \mathbf{E} + \frac{\partial \mathbf{B}}{\partial t} = \mathbf{K}, \qquad \nabla \cdot \mathbf{B} = \sigma. \tag{C.13.1.48}$$

To see more clearly what Holsinger and Halbach have done in the code, we write the material relations in the following form

$$\mathbf{H} = \frac{1}{\mu}\mathbf{B} = \sqrt{\frac{\epsilon}{\mu}}\left(\frac{1}{\sqrt{\epsilon\mu}}\mathbf{B}\right) = \sqrt{\frac{\epsilon}{\mu}}(c\mathbf{B}) = \sqrt{\frac{\epsilon}{\mu}}\mathbf{F}, \tag{C.13.1.49}$$

$$\mathbf{D} = \epsilon\mathbf{E}. \tag{C.13.1.50}$$

The field $\mathbf{F}$ has the same physical dimensions as those of the electric field $\mathbf{E}$. In terms of $\mathbf{E}$ and $\mathbf{F}$, Maxwell's equations take the form

$$\nabla \times \mathbf{F} - \frac{1}{c}\frac{\partial \mathbf{E}}{\partial t} = 0, \qquad \nabla \cdot \mathbf{E} = 0, \tag{C.13.1.51}$$

$$\nabla \times \mathbf{E} + \frac{1}{c}\frac{\partial \mathbf{F}}{\partial t} = \mathbf{K}, \qquad \nabla \cdot \mathbf{F} = c\sigma. \tag{C.13.1.52}$$

In the usual fashion, we can derive the wave equation for $\mathbf{F}$ by taking the curl of first equation and substituting for the curl of $\mathbf{E}$ from the third equation. This leads to the following sequence of equations

$$\nabla \times \nabla \times \mathbf{F} - \frac{1}{c}\frac{\partial}{\partial t}(\mathbf{K} - \frac{1}{c}\frac{\partial \mathbf{F}}{\partial t}) = 0, \tag{C.13.1.53}$$

$$\nabla(\nabla \cdot \mathbf{F}) - \nabla^2\mathbf{F} - \frac{1}{c}\frac{\partial \mathbf{K}}{\partial t} + \frac{1}{c^2}\frac{\partial^2 \mathbf{F}}{\partial t^2} = 0, \tag{C.13.1.54}$$

$$\nabla(c\sigma) - \nabla^2\mathbf{F} - \frac{1}{c}\frac{\partial \mathbf{K}}{\partial t} + \frac{1}{c^2}\frac{\partial^2 \mathbf{F}}{\partial t^2} = 0, \tag{C.13.1.55}$$

$$\nabla^2\mathbf{F} - \frac{1}{c^2}\frac{\partial^2 \mathbf{F}}{\partial t^2} = \nabla(c\sigma) - \frac{1}{c}\frac{\partial \mathbf{K}}{\partial t} \equiv \mathbf{T}. \tag{C.13.1.56}$$

By taking the curl of the third equation and substituting for the curl of $\mathbf{F}$ from the first equation, one can derive a wave equation for the electric field $\mathbf{E}$, which takes the form

$$\nabla^2\mathbf{E} - \frac{1}{c^2}\frac{\partial^2 \mathbf{E}}{\partial t^2} = -\nabla \times \mathbf{K} \equiv -\mathbf{S}. \tag{C.13.1.57}$$

A trial solution for the TE propagating wave mode in cartesian coordinates takes the form

$$F_z(x,y,z,t) = \overline{F}_z(x,y)\cos(k_z z - \omega t). \tag{C.13.1.58}$$

When this is substituted into the z-component of the wave equation, one obtains the equation

$$\left[\frac{\partial^2 \overline{F}_z}{\partial x^2} + \frac{\partial^2 \overline{F}_z}{\partial y^2} + (\frac{\omega^2}{c^2} - k_z^2)\overline{F}_z\right]\cos(k_z z - \omega t) = T_z(x,y,z,t). \tag{C.13.1.59}$$

The z- and t- dependence can be removed from this equation by assuming the fictitious driving magnetic current and charge take the following form

$$K_z(x,y,z,t) = \overline{K}_z(x,y)\sin(k_z z - \omega t), \tag{C.13.1.60}$$

$$\sigma(x,y,z,t) = \overline{\sigma}(x,y)\sin(k_z z - \omega t). \tag{C.13.1.61}$$

The final equation for $\overline{F}_z$ is

$$\frac{\partial^2 \overline{F}_z}{\partial x^2} + \frac{\partial^2 \overline{F}_z}{\partial y^2} + \left(\frac{\omega^2}{c^2} - k_z^2\right)\overline{F}_z = ck_z\overline{\sigma} + \frac{\omega}{c}\overline{K}_z. \tag{C.13.1.62}$$

It can be shown that, if we assume the following relations for E_z, K_x and K_y,

$$E_z(x,y,z,t) = \overline{E}_z(x,y)\cos(k_z z - \omega t), \tag{C.13.1.63}$$

$$K_x(x,y,z,t) = \overline{K}_x(x,y)\cos(k_z z - \omega t), \tag{C.13.1.64}$$

$$K_y(x,y,z,t) = \overline{K}_y(x,y)\cos(k_z z - \omega t), \tag{C.13.1.65}$$

then the wave equation for E_z reduces to

$$\frac{\partial^2 \overline{E}_z}{\partial x^2} + \frac{\partial^2 \overline{E}_z}{\partial y^2} + \left(\frac{\omega^2}{c^2} - k_z^2\right)\overline{E}_z = -(\frac{\partial \overline{K}_y}{\partial x} - \frac{\partial \overline{K}_x}{\partial y}). \tag{C.13.1.66}$$

One can get a self-consistent set of equations by assuming further the following relations

$$E_x(x,y,z,t) = \overline{E}_x(x,y)\sin(k_z z - \omega t), \tag{C.13.1.67}$$

$$E_y(x,y,z,t) = \overline{E}_y(x,y)\sin(k_z z - \omega t), \tag{C.13.1.68}$$

$$F_x(x,y,z,t) = \overline{F}_x(x,y)\sin(k_z z - \omega t), \tag{C.13.1.69}$$

$$F_y(x,y,z,t) = \overline{F}_y(x,y)\sin(k_z z - \omega t). \tag{C.13.1.70}$$

When these relations are put into the first and third Maxwell equations, the result is

$$\overline{F}_x = \frac{c}{\omega}\left(\frac{\partial \overline{E}_z}{\partial y} - k_z\overline{E}_y - \overline{K}_x\right), \tag{C.13.1.71}$$

$$\overline{F}_y = -\frac{c}{\omega}\left(\frac{\partial \overline{E}_z}{\partial x} - k_z\overline{E}_x - \overline{K}_y\right), \tag{C.13.1.72}$$

$$\overline{E}_x = -\frac{c}{\omega}\left(\frac{\partial \overline{F}_z}{\partial y} - k_z\overline{F}_y\right), \tag{C.13.1.73}$$

$$\overline{E}_y = \frac{c}{\omega}\left(\frac{\partial \overline{F}_z}{\partial x} - k_z\overline{F}_x\right). \tag{C.13.1.74}$$

Since the quantities $\overline{F}_x, \overline{F}_y, \overline{E}_x$, and $\overline{E}_y$ occur on both sides of the above four equations, one can further simplify the equations by proper substitutions. The final result is

$$\frac{\partial^2 \overline{E}_z}{\partial x^2} + \frac{\partial^2 \overline{E}_z}{\partial y^2} + (k_0^2 - k_z^2)\overline{E}_z = -\left(\frac{\partial \overline{K}_y}{\partial x} - \frac{\partial \overline{K}_x}{\partial y}\right), \tag{C.13.1.75}$$

$$\frac{\partial^2 \overline{F}_z}{\partial x^2} + \frac{\partial^2 \overline{F}_z}{\partial y^2} + (k_0^2 - k_z^2)\overline{F}_z = ck_z\overline{\sigma} + k_0\overline{K}_z, \tag{C.13.1.76}$$

$$\overline{F}_x = \frac{k_0}{k^2}\left(\frac{\partial \overline{E}_z}{\partial y} - \frac{k_z}{k_0}\frac{\partial \overline{F}_z}{\partial x}\right), \tag{C.13.1.77}$$

$$\overline{F}_y = -\frac{k_0}{k^2}\left(\frac{\partial \overline{E}_z}{\partial x} - \frac{k_z}{k_0}\frac{\partial \overline{F}_z}{\partial y}\right), \tag{C.13.1.78}$$

$$\overline{E}_x = -\frac{k_0}{k^2}\left(\frac{\partial \overline{F}_z}{\partial y} + \frac{k_z}{k_0}\frac{\partial \overline{E}_z}{\partial x} + \overline{K}_y\right), \tag{C.13.1.79}$$

$$\overline{E}_y = \frac{k_0}{k^2}\left(\frac{\partial \overline{F}_z}{\partial x} - \frac{k_z}{k_0}\frac{\partial \overline{E}_z}{\partial y} - \overline{K}_x\right), \tag{C.13.1.80}$$

where the quantities k and k_0 are defined by the relations

$$k = \sqrt{k_0^2 - k_z^2}, \tag{C.13.1.81}$$

$$k_0 = \frac{\omega}{c}. \tag{C.13.1.82}$$

These equations cannot be further separated unless we assume that we are dealing with the cutoff mode for which the wavevector k_z is zero. Under these circumstances, the equations above break into two sets. The first set involves the field components $\overline{F}_z$, $\overline{E}_x$, and $\overline{E}_y$. This is the TE mode. The second set of equations involves the field components $\overline{E}_z$, $\overline{F}_x$ and $\overline{F}_y$, which corresponds to the TM mode. There is no need to be concerned about the magnetic current and charge densities in these equations, because they will vanish at resonance.

The theory for finding the resonant modes in cartesian coordinates follows very closely the theory presented above in cylindrical coordinates.

C.13.2 Auxiliary Properties of RF-Cavities

The formulas used in SFO1 to calculate auxiliary properties have been displayed in Sec.C.1.2. These formulas are to some extent based on long-established conventions. Some of them were derived using assumptions appropiate only to proton, drift-tube linacs and should not be assumed to apply to electron linacs or to RFQ proton linacs. The derivation of some of these formulas is not readily available in the accelerator literature.

The intention of this section is to give the derivations and point out the assumptions inherent in the formulas. The expert user will probably find nothing new here, but the novice may profit from this discussion.

(This section is presently in rough draft stage and will be sent to persons on the mailing list for the Los Alamos Accelerator Code Group when it is finished.)

C.13.3 Boundaries and Meshes

This section has not been written. It will closely parallel Sec. B.13.5, which is in rough draft stage. The intention of this section is to give the user some guidance on choosing proper boundary conditions for SUPERFISH problems. This is not a trivial problem because the boundary conditions allowed by the code are only approximate for some geometries.

The subsection on "meshes" will describe the numbering of mesh points on the logical mesh and describe ordering used in the code for setting up the equations to be solved. It will also describe the "virtual" mesh points beyond the physical boundary of the problem.

C.13.4 Numerical Methods

This section is only partially written. Persons who have received this manual directly from the Los Alamos Accelerator Code Group will receive the completed section when it is finished. The numerical methods have been partially described in Chapter C.1, and in Sec. C.13.1. The fundamental paper describing the methods was written by Holsinger.[2] For the convenience of the reader, it has been reproduced here and must serve as a substitute until a more complete description is available.

Particle Accelerators
1976, Vol. 7, pp. 213–222

© Gordon and Breach, Science Publishers Ltd.
Printed in the United Kingdom

SUPERFISH—A COMPUTER PROGRAM FOR EVALUATION OF RF CAVITIES WITH CYLINDRICAL SYMMETRY

K. HALBACH

Nuclear Science Division, Lawrence Berkeley Laboratory, University of California, Berkeley, California 94720, USA

and

R. F. HOLSINGER

Los Alamos Scientific Laboratory, University of California, Los Alamos, New Mexico 87545, USA

(*Received June 17, 1976*)

The difference equations for axisymmetric fields are formulated in an irregular triangular mesh, and solved with a direct, noniterative method. This allows evaluation of resonance frequencies, fields, and secondary quantities in extreme geometries, and for the fundamental as well as higher modes. Finding and evaluating one mode for a 2000 point problem takes of the order of 10 sec on the CDC 7600.

1 INTRODUCTION

Over the last 10 to 15 years, a number of computer programs have been developed that find the electromagnetic resonance frequency and evaluate the axisymmetric fields in rf cavities with axisymmetric symmetry. The codes that allow this analysis to be made in an essentially arbitrary axisymmetric geometry (see for instance Refs. 1–3) have the following in common: For some geometries, like cavities that have a large diameter compared to their length, and/or for modes higher than the fundamental mode, the convergence rate can be extremely small, or convergence may not be achieved at all. Stated very briefly, the reason for these problems is the fact that in all these codes, an overrelaxation method is used to solve a set of homogeneous linear field equations. The properties of these equations are such that some well developed methods for overrelaxation-factor optimization are not applicable, and it might well be true that the eigenvalues of the matrices for some problems are located in such a way that even an optimized overrelaxation scheme would still result in unacceptably low convergence rates.

To eliminate these problems, we developed the code SUPERFISH that uses a direct, noniterative method to solve a set of inhomogeneous field equations. This code is a combination of some parts of the code RFISH,[4] some new ideas, and the direct solution method used by Iselin in his magnet code FATIMA.[5] In order to give a good overall understanding of SUPERFISH in a limited space, we do not present all detailed formulas, but do include the description of all parts that are conceptually significant, even at the expense of reformulating and/or condensing parts of the cited literature.

In Sections 2 and 3 we discuss separately the structure of the difference equations, and the direct, noniterative method used to solve a set of inhomogeneous linear equations. In Section 4 the basic structure of SUPERFISH is described, and the remaining sections give some details of the theory and of the program as it exists today, and an outline of contemplated future developments.

2 STRUCTURE OF THE DIFFERENCE EQUATIONS IN AN IRREGULAR TRIANGULAR MESH

Inspection of Maxwell's equations shows that for $\partial \mathbf{E}/\partial\phi = 0$, $\partial \mathbf{H}/\partial\phi = 0$, i.e., axisymmetric fields, two independent sets of solutions can exist: one having as only nonzero field components E_ϕ, H_z, H_r; the other, H_ϕ, E_z, E_r. These two solutions are, for equivalent boundary conditions, identical;

we therefore talk only about the latter set. Assuming, without loss of generality, that the magnetic field is proportional to $\cos \omega t$, and the electric field is proportional to $\sin \omega t$, and using suitable units, Maxwell's equations can be written as

$$\text{curl } \mathbf{H} = k\mathbf{E}, \tag{1a}$$

$$\text{curl } \mathbf{E} = k\mathbf{H}, \tag{1b}$$

with $k = \omega/c$, and $\mathbf{H}$ and $\mathbf{E}$ representing the electric and magnetic fields divided by their respective time dependence.

We seek to find numerical solutions for some of the eigenvalues k and associated fields of Eqs. (1a) and (1b) in cylindrical cavities of essentially arbitrary shapes, with $\mathbf{H} = 0$ on the axis and possibly some other parts of the boundary (Dirichlet boundaries), and the electric field perpendicular to the remaining boundaries (Neumann boundaries), implying infinitely conducting walls there.

2.1 *The Mesh*

To solve the differential Eqs. (1a) and (1b), we introduce an irregular triangular mesh[6] in the z-r plane. Figure 1 shows the logical mesh, with mesh points identified by labels K and L, assuming the integer values 1 through $K_{max} = K_2$, and 1 through L_2. To establish a mesh that can be used to solve the field equations for a particular geometry, defined by its boundaries, the user first assigns boundary coordinates z, r to an arbitrary but reasonable selection of logical points K, L. The mesh generator, described in Ref. 6, then generates a mesh of triangles that is topologically identical to the logical mesh, but has all boundaries defined by mesh lines. The mesh generator finds the z, r coordinates of interior points, for a given set of boundary points, with an iterative process that is similar to a numerical method used to solve Laplace's equation. Figure 2a shows a logical mesh, and Figure 2b a physical mesh, for one half of an Alvarez cavity. In Figure 2a, points on heavily drawn logical lines represent those with assigned z, r coordinates. The two heavily drawn interior lines are used to delineate zones with different mesh point densities. Exterior mesh points, i.e., points inside the drift tube, are not shown since they do not affect the field calculations.

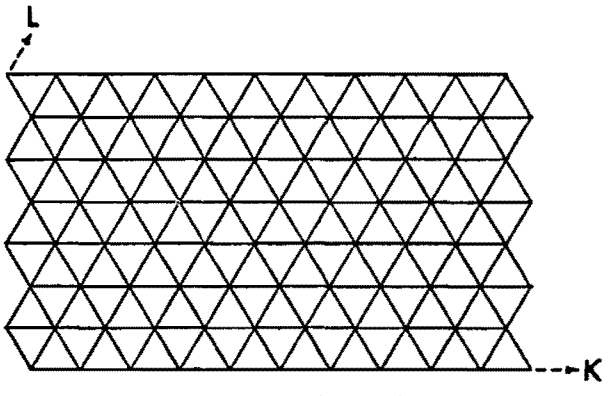


FIGURE 1 Logical triangular mesh.

2.2 *The Difference Equations for Interior Points*

We use the quantity $H = H_\phi$ to describe the rf fields. This somewhat unconventional choice (usually $r \cdot H_\phi$ is used) has the advantage of not requiring any special treatment of the region close to the axis, since H will be proportional to r there, whereas $rH_\phi \sim r^2$ for small r. From Eqs. (1a) and

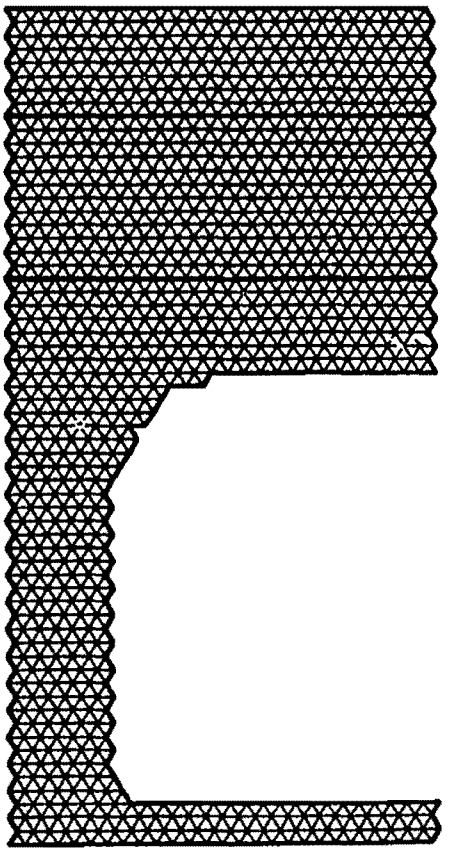


FIGURE 2a Logical mesh for 1/2-Alvarez cavity.

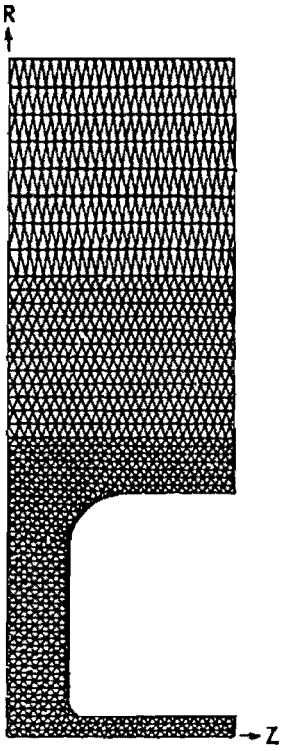


FIGURE 2b Physical mesh for 1/2-Alvarez cavity.

(1b) we obtain as the differential equation for H:

$$\text{curl}(\text{curl } \mathbf{H}) = k^2\mathbf{H}. \tag{2}$$

To derive difference equations for H, we use the procedure described by Winslow[6]: we first introduce a secondary mesh by drawing connecting lines between the "center of mass" of every triangle and the center of each of the three sides of the triangle. As a consequence, every mesh point is now surrounded by a unique twelve-sided polygon. This secondary mesh of dodecagons covers completely the whole problem area, and Figure 3 shows the dodecagon surrounding just one mesh point. The difference equations for H are now obtained by integrating Eq. (2) over the area (in the z-r plane) of one dodecagon at a time. This yields

$$\int \text{curl}(\text{curl } \mathbf{H}) \cdot d\mathbf{a} = \oint \text{curl } \mathbf{H} \cdot d\mathbf{s} = k^2 \int \mathbf{H} \cdot d\mathbf{s}. \tag{3}$$

Assuming that H behaves like a linear function of z and r within every triangle, H inside every triangle is uniquely determined by the values of H at the three corner-mesh points of the triangle.

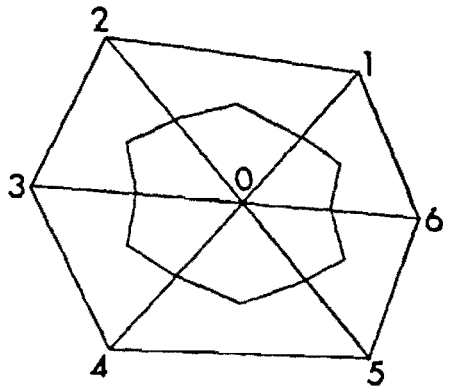


FIGURE 3 Irregular triangular mesh with secondary dodecagon.

The integrals in Eq. (3) can therefore be expressed in terms of the value of H at the "center-mesh point" of the dodecagon and its six nearest logical neighbors, giving a relationship of the following kind

$$\sum_{0}^{6} H_n(V_n + k^2 W_n) = 0, \tag{4}$$

with V_n and W_n depending only on the coordinates z, r of the seven mesh points involved.

Identifying each difference equation with its "center-point," we therefore get one difference equation for H at every interior mesh point.

2.3 *The Treatment of Boundary Points*

Turning now to mesh points on the boundaries of the problem, it is clear that no difference equations are needed for H at boundary points when the boundary conditions require $H \equiv 0$ there. Nevertheless, we have to explore whether or not the difference equations for such points are satisfied. To this end, we consider first Dirichlet-boundary points that are not on the problem axis. This kind of boundary condition can obviously only be imposed as a symmetry condition along a plane defined by z = const. This implies that in the real world a point on one side of this line has an H-value of the same magnitude, but opposite sign, as the symmetrically located point, and the difference equation, Eq. (4), is clearly satisfied for every such boundary point.

This argument cannot be applied without elaboration for points on axis ($r = 0$), and the difference equations resulting from Eq. (3) are in fact not satisfied for those points. To see how this can be interpreted, we can introduce on the

right-hand side of Eq. (1b) a (magnetic) current density term j that has only an azimuthal component. This gives on the right-hand side of Eq. (3) the additional term kI, where I represents the total current associated with the point under consideration, and assumed to be concentrated there. In our case of axis points, an azimuthal current on the axis is, of course, without consequences, and this whole argument could also be used to lend legitimacy to the application of the symmetry consideration to axis points.

When **E** is required to be perpendicular to boundary-mesh lines, we consider that part of the dodecagon surrounding a boundary point that goes through inside problem triangles, i.e., the polygon 0–1′–2′–3′–4′–0 in Figure 4. Since $\mathbf{E} = \text{curl}\,\mathbf{H}/k$ is required to be perpendicular to lines 4–0 and 0–1,

$$\int_{4'-0-1'} \text{curl}\,\mathbf{H} \cdot d\mathbf{s} = 0.$$

This means that the difference equation for H at such a boundary point is identical to that for interior points except that only contributions from interior triangles are taken into account.

A very important property of the difference equation, Eq. (4), is the fact that if we use the logical coordinates (indices) K, L to identify mesh points, and if K_0, L_0 are the coordinates of any specific mesh point for which we write down Eq. (4), then the coordinates of the other mesh points contributing to Eq. (4) differ from K_0 and L_0 by not more than ± 1.

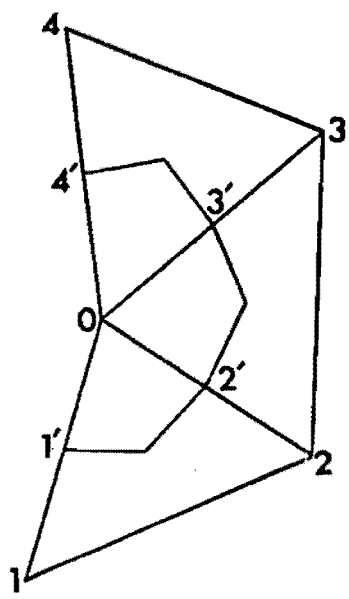


FIGURE 4 Boundary mesh point with neighboring interior mesh points, and secondary polygon.

3 DIRECT, NONITERATIVE SOLUTION OF A SET OF INHOMOGENEOUS LINEAR EQUATIONS

If one wrote down difference Eq. (4) for *all* (i.e., including all boundary and exterior mesh points) H_{KL} of the logical mesh by rows from left to right; and if one also had some inhomogeneous terms, one could write the resulting system of equations in the following form:

$$\begin{pmatrix} a_{11} & a_{12} & & & & \\ a_{21} & a_{22} & a_{23} & & & \\ & a_{32} & a_{33} & a_{34} & & \\ & & & \vdots & & \\ & & & a_{L_2-1,L_2-2} & a_{L_2-1,L_2-1} & a_{L_2-1,L_2} \\ & & & & a_{L_2,L_2-1} & a_{L_2,L_2} \end{pmatrix} \times \begin{pmatrix} \mathscr{H}_1 \\ \mathscr{H}_2 \\ \mathscr{H}_3 \\ \vdots \\ \mathscr{H}_{L_2-1} \\ \mathscr{H}_{L_2} \end{pmatrix} = \begin{pmatrix} G_1 \\ G_2 \\ G_3 \\ \vdots \\ G_{L_2-1} \\ G_{L_2} \end{pmatrix} \tag{5}$$

In this matrix equation, $\mathscr{H}_1$ represents a column vector with the components H_{K1}, $K = 1 - K_2$; $\mathscr{H}_2$, a vector with components H_{K2}, $K = 1 - K_2$, etc, and the G_n represent the corresponding inhomogeneous terms. The matrix on the left side of Eq. (5) has all zeroes except for the block matrices a_{ij} of size $K_2 \times K_2$. These blocks are also sparse, the diagonal blocks containing only three nonzero elements in every row, and the off-diagonal blocks not having more than three in any row. Deferring to Section 4 the discussion of how we can cast our field evaluation problem in the form of Eq. (5), involving all points of the logical mesh as well as inhomogeneous terms on the right side of Eq. (5), we discuss now the method used to solve Eq. (5).

We first transform all diagonal blocks into unity matrices, and remove all blocks to the left of the diagonal blocks, with the Gaussian block elimination process: we multiply the equations represented by the first row of blocks from the left by a_{11}^{-1}, and

then subtract from the second row the new first row after multiplication from the left by a_{21}. The new set of equations is then the same as the original one, except that $a_{21} = 0$; $a_{11} = I$; and a_{12}, G_1, a_{22}, and G_2 are now modified. This process is repeated, involving rows 2 and 3, then 3 and 4, etc. The very last step in this process is the multiplication of the last row (modified by the previous step) from the left with the modified block matrix $a^{-1}_{L_2, L_2}$.

Having Eq. (5) rewritten in this form, the last row now represents directly the solution for $\mathscr{H}_{L_2}$. Using this now numerically known vector in row $L_2 - 1$ yields directly the solution for $\mathscr{H}_{L_2-1}$, and continuing this back-substitution process yields the numerical values of all components of all block vectors $\mathscr{H}_n$.

It is important to recognize the fact that this particular fast direct method to solve inhomogeneous linear equations can be used only if they can be cast in the form of Eq. (5), and if the matrix on the left side of Eq. (5) is nonsingular.

4 CALCULATION OF FIELDS AND RESONANCE FREQUENCIES IN SUPERFISH

In order to allow application of the direct linear equation solution described in Section 3, we have to include in an artificial way in the system of equations also those points that are part of the logical mesh, but are external to the actual field solution problem. How this is done, and the treatment of points on Dirichlet boundaries, is discussed in Section 4.1; the creation of the inhomogeneous terms is discussed in Section 4.2, and the resonance frequency determination is discussed in Section 4.3.

4.1 *Treatment of Exterior Points and Points on Dirichlet Boundaries*

The simplest and most practical way to include exterior points without affecting the actual field equations, and without causing the matrix on the left side of Eq. (5) to become singular, is to let the equation for every exterior point read $H_{\text{exterior}} = 0$, and to make all couplings to other equations zero by setting the corresponding coefficients equal to zero also. In other words, if n_0 is the index identifying an exterior point (not a block!) in the overall H-vector on the left side of Eq. (5), one simply sets all elements of row n_0 and column n_0 of the matrix in Eq. (5) equal to zero, with the exception of the n_0, n_0 diagonal element, which is set equal to one. The n_0-element of the inhomogeneous contribution vector on the right side of Eq. (5) is set equal to zero also. The logic of the equation-solving routine is arranged in such a way that the thus-introduced zeroes are actually never used in multiplications, just as the other zeroes in the sparse matrices are never used as multipliers either. Points on Dirichlet boundaries are treated in exactly the same way. However, in contrast to exterior points, their z-r coordinates do enter into the expressions for V_n, W_n in Eq. (4) involving the other point(s) of the triangles that have one or more Dirichlet-boundary points at their corners.

4.2 *Generation of Inhomogeneous Terms for Eq. (5)*

In order to turn the set of homogeneous difference equations [Eq. (4)] into a well-posed set of inhomogeneous field equations, one could be tempted to introduce at one mesh point a driving (magnetic) current, as discussed in Section 2.3. That would be an unwise procedure when one is close to a resonance, since the matrix in Eq. (5) is singular for every resonance frequency, leading, as it must, to infinite fields. Instead, we prescribe that an appropriately chosen off-axis mesh point has the field value one and in effect remove the difference equation for that point from the system of difference equations. To do this without destroying the structure of the field equations, we can proceed in one of the following two ways:

1) If the chosen point is identified by its index n_1 in the overall H vector, we set all matrix elements in row n_1 of the matrix in Eq. (5) equal to zero, except for the diagonal element n_1, n_1, which is set equal to 1. In the vector G on the right-hand side of Eq. (5), all elements are set equal to zero, except the n_1-element is set equal to one. Column n_1 of the matrix is left unchanged.

2) Every matrix element in row n_1 and in column n_1 is set equal to zero, except the n_1, n_1-diagaonal element is set equal to one. The vector on the right-hand side of Eq. (5) is set to equal minus the original column n_1 of the matrix, except for element n_1, which is set equal to one.

The second procedure treats the point with the fixed field value in the same way as exterior points and points on Dirichlet boundaries, and in

addition nonzero terms are generated on the right-hand side of Eq. (5). We therefore use that procedure in the code.

In contrast to the explicit introduction of a driving current, with procedures (1) and (2) the matrix on the left side of Eq. (5) is well conditioned even for resonance frequencies.

If we take the original difference equation for the point with the prescribed field value and solve for the field value at that point, using the solution values of the field at the neighbor points, we will get a value different from the prescribed value, except at resonance. This difference can be interpreted as being proportional to the current I_1 necessary at that point to drive the cavity to the prescribed amplitude at the point with the prescribed field value. For this reason we will refer to this point as the driving point.

4.3 *Resonance Frequency Determination*

The driving current I_1 introduced above depends on k^2 through the coupling coefficients in the difference Eq. (4), and the resonance condition is characterized by

$$I_1(k^2) = 0, \tag{6}$$

since then there is no difference between the value of H_{n_1} as calculated from the difference equation for that point, and the prescribed value used there to solve for the fields: i.e., the difference equations are satisfied for all points of consequence. To find the value(s) of k^2 for which Eq. (6) is satisfied,

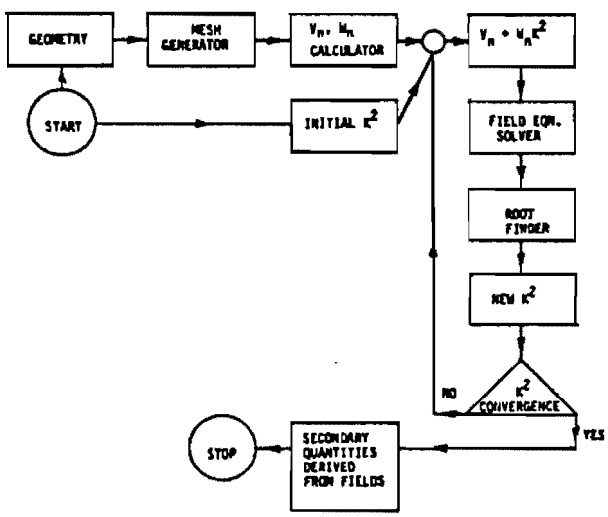


FIGURE 5 Flow diagram of SUPERFISH.

we can combine the above-described "function generator" for $I_1(k^2)$ with a numerical root-finding algorithm, such as the secant method, or a parabola fit method. The latter method is used in the present stage of code development. But we expect that it will be useful to use a root-finding algorithm that takes into account some of the properties of $I_1(k^2)$ that are described in Section 5. Figure 5 depicts a flow diagram of the major parts of SUPERFISH.

5 PROPERTIES OF $I_1(k^2)$ AND INTRODUCTION AND PROPERTIES OF $D(k^2)$

To obtain an understanding of some of the properties of the function $I_1(k^2)$, and later $D(k^2)$, we will go back to the differential equations, Eqs. (1a) and (1b), with Eq. (1b) amended on the right side by the magnetic current density $\mathbf{j}$, assumed to be constant over a small area surrounding the driving point. In the process of deriving some formulas, we have to evaluate integrals like $\int \mathbf{H} \cdot \mathbf{j}_1 \cdot dv$, and set this equal $2\pi r_1 \cdot h_1 \cdot I_1$ where I_1 is the total driving current; r_1, the distance of the driving point from the problem axis; and h_1, the magnetic field averaged over the region where $j_1 \neq 0$. The association between h_1 resulting from this continuum theory and the value of H at a mesh point in the representation by the difference equations is complicated by the fact that h_1 has a logarithmic singularity when the area where $j \neq 0$ is reduced to zero (for fixed I_1). While it seems reasonable to set h_1 equal to H at the driving point, or the value resulting from averaging H over the dodecagon associated with the driving point, it is clear that the quantitative relationships developed below will describe only approximately the relationships between the quantities derived from the difference equations. However, it is also clear that the general behavior of the functions of interest is correctly described by the results derived from the continuum theory below.

Adding the term $\mathbf{j}_1$ to the right side of Eq. (1b) gives

$$\text{curl}\,\mathbf{E} = k\mathbf{H} + \mathbf{j}_1. \tag{7}$$

Forming the scalar product of both sides of this equation with $\mathbf{H}$, and subtracting from that Eq. (1a), after being multiplied by $\mathbf{E}$, yields:

$$\begin{aligned}\mathbf{H}\,\text{curl}\,\mathbf{E} - \mathbf{E}\,\text{curl}\,\mathbf{H} &\equiv \text{div}(\mathbf{E} \times \mathbf{H}) \\ &= kH^2 + \mathbf{j}_1\mathbf{H} - kE^2.\end{aligned}$$

Integrating this over the whole problem volume gives

$$\int \mathrm{div}(\mathbf{E} \times \mathbf{H})\mathrm{d}v \equiv \int (\mathbf{E} \times \mathbf{H}) \cdot \mathrm{d}\mathbf{a}$$
$$= 2\pi r_1 h_1 I_1 - k \int (E^2 - H^2)\mathrm{d}v. \quad (8)$$

Since $\mathbf{E} \times \mathbf{H}$ is either zero on the problem boundary, or perpendicular to the boundary normal, $\int (\mathbf{E} \times \mathbf{H})\mathrm{d}\mathbf{a} = 0$, and we get

$$D(k^2) \equiv \frac{2\pi r_1 h_1 k I_1}{\int H^2 \,\mathrm{d}v} = R(k^2) - k^2, \quad (9)$$

$$R(k^2) = \frac{\int k^2 E^2 \mathrm{d}v}{\int H^2 \,\mathrm{d}v} = \frac{\int (\mathrm{curl}\,\mathbf{H})^2 \,\mathrm{d}v}{\int H^2 \,\mathrm{d}v}. \quad (10)$$

The new function $D(k^2)$ has the property that its value does not depend on the scaling of h_1, or I_1 if that is the quantity that one wants to consider as the primary variable.

To obtain more information about the behavior of $I_1(k^2)$, we now calculate $\mathrm{d}I_1/\mathrm{d}(k^2) = I_1'$. To this end, we take the derivatives with respect to k^2 of Eqs. (1a) and (7). Indicating derivatives with respect to k^2 by primes, we get

$$\mathrm{curl}\,\mathbf{H}' = k\mathbf{E}' + \mathbf{E}/2k \quad (11)$$

$$\mathrm{curl}\,\mathbf{E}' = k\mathbf{H}' + \mathbf{H}/2k + \mathbf{j}_1'. \quad (12)$$

It should be noted that for our procedure of field evaluation, $\mathbf{H}' = 0$ on Dirichlet boundaries, because $\mathbf{H} = 0$ there for all k^2. Similarly $\mathbf{E}'$ is perpendicular to Neumann boundaries since the component of $\mathbf{E}$ parallel to a Neumann boundary is zero for all k^2. We now consider

$$\mathrm{div}(\mathbf{E} \times \mathbf{H}' - \mathbf{E}' \times \mathbf{H}) \equiv \mathbf{H}' \cdot \mathrm{curl}\,\mathbf{E} - \mathbf{E} \cdot \mathrm{curl}\,\mathbf{H}' - \mathbf{H} \cdot \mathrm{curl}\,\mathbf{E}' + \mathbf{E}' \,\mathrm{curl}\,\mathbf{H}.$$

Using for the curl expressions the appropriate right sides of Eqs. (1a), (7), (11) and (12) yields

$$\mathrm{div}(\mathbf{E} \times \mathbf{H}' - \mathbf{E}' \times \mathbf{H}) = \mathbf{H}' \cdot \mathbf{j}_1 - \mathbf{H} \cdot \mathbf{j}_1' - (E^2 + H^2)/2k.$$

Integrating this over the problem volume gives, as in Eq. (8), zero on the left side, yielding

$$2\pi r_1 k(h_1' I_1 - h_1 I_1') = \int (E^2 + H^2)\mathrm{d}v/2. \quad (13)$$

We intentionally made no *a priori* assumptions whether we consider h_1 or I_1 fixed when k^2 is changed. However, for the case considered so far, $h_1' = 0$, and we can immediately deduce the following conclusions from Eq. (13):

$$h_1 I_1' < 0 \quad \text{(Foster's theorem)}. \quad (14)$$

This means that for fixed h_1, between every two resonances ($I_1(k^2) = 0$) $I_1(k^2)$ must have a singularity such that the sign of $I_1(k^2)$, and therefore also of $D(k^2)$, changes.

At a resonance, $\int E^2 \,\mathrm{d}v = \int H^2 \,\mathrm{d}v$ [see Eqs. (9) and (10)], giving $\int H^2 \,\mathrm{d}v$ on the right side of Eq. (13). We therefore get from Eqs. (13), (9), and (10) at a resonance ($I_1 = 0$):

$$\frac{2\pi r_1 h_1 k I_1'}{\int H^2 \,\mathrm{d}v} = D'(k^2) = R'(k^2) - 1 = -1. \quad (15)$$

Since $I_1(k^2)$ has a singularity between resonances, it is more convenient to study $D(k^2)$ in the vicinity of these singularities. To this end, we first consider $R(k^2)$. According to Eq. (9), $R(k^2) = k^2$ at every resonance, and $R' = 0$ at resonance follows from Eq. (15). Since R cannot be negative, $R(k^2)$ must look qualitatively as indicated in Figure 6 and $R - k^2 = D(k^2)$ as shown in Figure 7. An important consequence is that between resonances, $D(k^2)$ goes through zero, and this sign change must take place where $I_1(k^2)$ has a singularity.

To study $D(k^2)$ in the vicinity of this root of $D(k^2)$ that does *not* represent a resonance, we take advantage of the fact that $D(k^2)$ is independent of the scaling of the field and current quantities. We can therefore consider I_1 as given and kept constant, and consider h_1 as the k^2-dependent

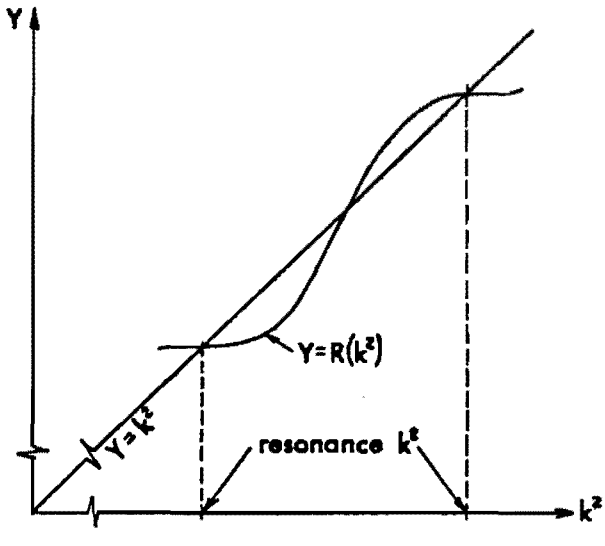


FIGURE 6 Graphical representation of properties of $R(k^2)$.

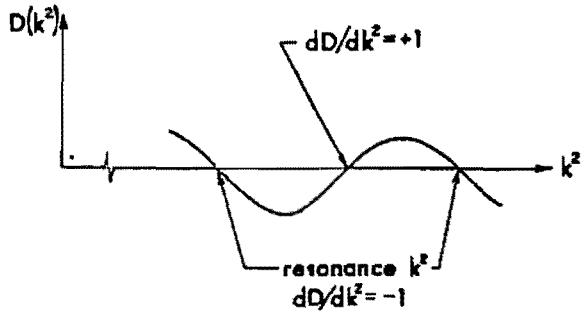


FIGURE 7 Graphical representation of properties of $D(k^2)$.

quantity that causes $D(k^2) = 0$. At this "between-resonance root," it follows from Eqs. (9) and (10) that $\int E^2\,dv = \int H^2\,dv$, giving again $\int H^2\,dv$ as the right side of Eq. (13). We therefore get from that equation

$$\frac{2\pi r_1 k I_1 h_1'}{\int H^2\,dv} = D'(k^2) = R'(k^2) - 1 = 1. \quad (16)$$

A possible use of Eqs. (15) and (16) will be briefly described at the end of Section 6.3.

6 PROGRAM STATUS AND FUTURE DEVELOPMENT

The computer program was originally written for the CDC 7600 operating under the Livermore Time-Sharing System (LTSS), and we describe here that particular version.

6.1 *Computer Time and Storage Requirements*

The CPU time required for a field evaluation is dominated by the time required to invert the block matrices. For the system of equations described at the beginning of Section 3, the time used for inversion of the block matrices is proportional to $K_2^3 \cdot L_2$. When $K_2 > L_2$, the difference equations are arranged along columns of the logical mesh, leading to this expression for the CPU time

$$T = T_1 N^2 \varepsilon, \quad (17)$$

with N representing the total number of logical mesh points, and ε the smaller of the two numbers K_2/L_2, L_2/K_2. For the CDC 7600 under LTSS, $T_1 \approx 0.75\ \mu$sec.

With the present system to find the roots of $I_1(k^2)$, it takes 3 to 6 field iterations to determine a resonance frequency accurately.

The storage requirements for the program are approximately $11 \cdot N$ exclusive of the memory required for the modified off-diagonal block matrices, needed for the back-substitution. These matrices represent $N^{3/2} \cdot \varepsilon^{1/2}$ words, too much to be accommodated in core for $N > 1500$. For larger problems, the disk has to be used. However, S. B. Magyary (LBL) has pointed out that one needs to store only two such matrices when one has to calculate only $I_1(k^2)$ (and not the complete field map), provided the driving point is associated with the last row of block matrices on the left side of Eq. (5). In that case, the large amount of storage is not needed until one has a converged resonance frequency.

6.2 *Accuracy*

Since we know from our experience with the RFISH code and the magnet code POISSON that the program is unlikely to have problems related to curved boundaries, we have made analytically testable runs so far only for empty pill-box cavities.

To see whether this code has any problems with extreme geometries, we ran an empty box of 5 cm length and 150 cm radius with 1267 points. Without any difficulty, the code returned the fundamental frequency correct to all five printed digits.

Much more extensive runs were made for an empty box 60 cm long and a radius of 88 cm. The mesh point separation was 2 cm in both the axial and radial direction, giving a total of 1395 points. The fundamental frequency of this cavity is 130.389 MHz and is reproduced by the code with an error of 1 part in 10,000, while the stored energy is reproduced to an accuracy of 1 part in 3000. A much more severe test is the evaluation of higher modes. Resonance frequency number eight is 582.44 MHz, and is returned by the code as 583.59 MHz; the stored energy calculated by the code is 3% smaller than the correct value. Figure 8 shows the pattern of electrical field lines (rH = const) for this mode. It should be noted that the distance between an extreme value of rH and the next axial node is only 7.5 mesh spacings. Modes 29 and 30 represent an even more extreme test: The analytical frequencies are 1179.9 MHz and 1186.3 MHz, and the code-produced frequencies are 1183.0 MHz and 1196.6 MHz, while the energy of these modes is off by approximately 10% Considering the fact that mode 29 has six radia and one axial nodes, and mode 30 has three radia

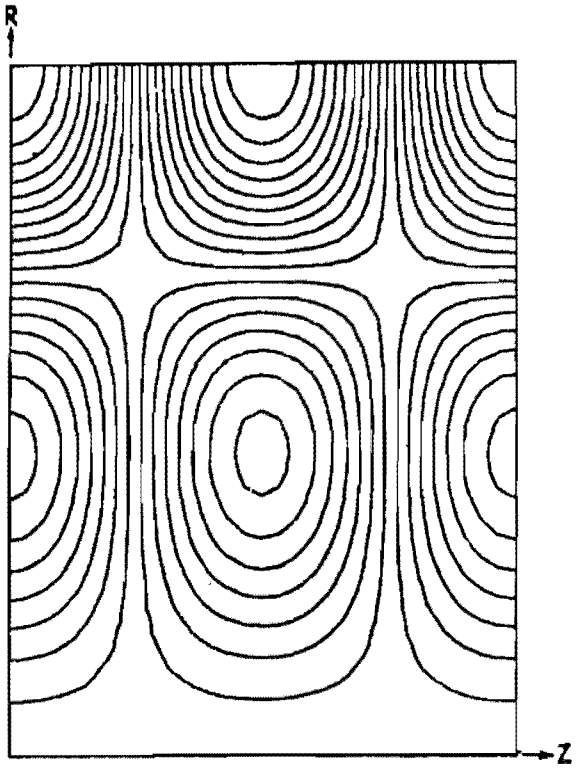


FIGURE 8 Electric field lines (rH = const) for mode No. 8 in test cavity.

and four axial nodes, these numbers are surprisingly good. The closeness of the two resonances did not cause any problems. "Turning on" the partial and complete pivoting of the matrix inversion routines, or iterating on the field residuals of the solution of the difference equations, did not change any of these numbers. However, increasing the number of mesh points caused a marked improvement of the accuracy of the frequency and the stored energy, indicating that the numerical errors are due to mesh size, and not round-off errors.

6.3 *Secondary Quantities, and Near Future Developments*

The program calculates now, or will calculate in the very near future, the following secondary quantities: stored energy; transit time factors; energy dissipated on designated surfaces; $|H|_{max}$, $|E|_{max}$ on designated surfaces; shunt impedance; Q; and frequency perturbation by drift tube stems.

We also plan to calculate and print out coefficients that indicate how the movement of designated surfaces perturbs the resonance frequency. These quantities were calculated by RFISH and proved extremely valuable.

To simplify the work on high-order modes, we intend to generate printout plots of node lines (i.e., H = 0-lines) and/or plots of points with local extrema of rH, and $I_1(k^2)$ and $D(k^2)$ plots.

To simplify the design of cavities that have to have a predetermined resonance frequency, we intend to run the code with that fixed frequency (possibly modified by drift tube stems) and to accomplish $I_1 = 0$ by moving or deforming a designated boundary. The techniques necessary to do this are already used in the magnet design code MIRT[7] and can easily be incorporated in SUPERFISH.

To reduce the number of iterations necessary to find a resonance frequency, we plan to employ a root-finding routine that uses the properties of $D(k^2)$ expressed by Eqs. (15) and (16). If this code is used extensively to find high-order modes, it might also be profitable to attempt to develop a mode pattern analysis and prediction routine.†

6.4 *Advantages of SUPERFISH*

The main advantage of the code is the capability to solve problems that other codes cannot solve at all, or only with great expenditure of computer time. In addition, the code is quite fast, requiring only about 1 sec per iteration on the frequency for the test problem discussed above. With five iterations and the time used to calculate miscellaneous other quantities, one has a complete solution in 6 sec. The irregular triangular mesh, while not allowing as many mesh points as a square mesh, has the advantage of allowing the definition of boundaries by mesh lines, and to produce a mesh with a large density of mesh points in regions where the problem requires high resolution.

6.5 *Disadvantages of SUPERFISH*

The drawback associated with the irregular triangular mesh is the fact that one has to generate such a mesh. This extra step can slow down the

† Since submission of this report for printing, the more sophisticated root-finding routine has been developed and is working very well. Work on the mode prediction algorithm has started and looks very promising.

total process of receiving the desired answers. This problem has been partly reduced by the creation of the code AUTOMESH, developed by one of us (R.F.H.) while at CERN. This code optimizes automatically the coordination between space-boundary coordinates and logical coordinates, provided that one is satisfied with a uniform mesh point density in a limited number of distinct regions. At the time of writing this paper, an effort is being undertaken at LASL by D. Swenson, W. Jule, and one of us (R.F.H.) to improve the whole process of data input and mesh generation.

There is one basic drawback associated with the necessity of having a driving point in the problem: if one happend to choose its location such that it is on a $H = 0$ line for the problem under consideration computational problems would result. For that reason, it is advisable to put the driving point on a Dirichlet boundary. When the code detects the computational difficulty, it can switch the driving point to a more favorable neighboring point on the boundary, thus eliminating the problem.

ACKNOWLEDGEMENTS

We thank Dr. T. Elioff, Mr. A. Faltens, Dr. L. J. Laslett, Mr. S. B. Magyary at LBL, and Dr. R. A. Jameson, Ms. S. Johnson, Dr. W. E. Jule; Dr. E. A. Knapp, and Dr. D. A. Swenson at LASL, for discussions and/or support of this work. This work was done with support from the U.S. Energy Research and Development Administration. Any conclusions or opinions expressed in this report represent solely those of the authors and not necessarily those of The Regents of the University of California, the Lawrence Berkeley Laboratory or the U.S. Energy Research and Development Administration.

REFERENCES

1. T. Edwards, MURA Report 622 (1961), unpublished (MESSYMESH).
2. H. C. Hoyt, *Rev. Sci. Instrum.* 37, 755 (1966) (LALA).
3. M. Bell, G. Dôme, *Proc. 1970 Proton Lin. Acc. Conf.*, p. 329.
4. Developed in 1969–70 by K. Halbach, R. F. Holsinger, R. B. Yourd specifically for analysis of cylindrical rf cavities with large diameter-to-length ratio.
5. C. Iselin, to be published.
6. A. Winslow, *J. Comp. Phys.* 1, 149 (1966).
7. K. Halbach, *Proc. of 2nd Intern. Conf. on Magnet Technology*, Oxford 1967, p. 47.

C.13.5 Table of Problem Constants in Numerical Order for SUPERFISH

Constant	Default	Symbol	Function
CON(1)	1	KPROB	KPROB is used by LATTICE to differentiate between a SUPERFISH and a POISSON-PANDIRA run. It is set to 0 or 1 in LATTICE, depending on the absence or presence of a character in the first position of the title line. KPROB = 0 means a POISSON or PANDIRA run; KPROB $\neq$ 0 means a SUPERFISH run.
CON(2)	none	NREG	Number of regions in the problem. Passed by AUTOMESH to LATTICE and from LATTICE to SUPERFISH via TAPE73. Note NREG must be less than 32
CON(3)	none	LMAX	Number of points in the L (vertical) direction in the logical mesh. Determined in LATTICE.
CON(4)	none	KMAX	Number of points in the K (horizontal) direction in the logical mesh. Determined in LATTICE.
CON(5)	none	IMAX	IMAX = KMAX + 2
CON(6)	0	MODE	Not used by the present code. It is being reserved for later inclusion of dielectric materials.
CON(7)	none	none	Not used in SUPERFISH problems.
CON(8)	none	none	Fraction of RFQ cross section actually calculated.
CON(9)	1.0	CONV	Conversion factor for length units. Default units are centimeters. Set CONV equal to the number of centimeters per unit desired. CONV must be changed in LATTICE.
CON(10)	0.004	none	Not used in SUPERFISH problems.

Constant	Default	Symbol	Function
CON(11)	0	NAIR	Number of "air" points. The default is an initial value. LATTICE counts the number of mesh points in the air regions of the cavity and records the value in CON(11). The user has no control of this CON.
CON(12)	0	NFE	Number of "iron" points. The default is an initial value. LATTICE counts the number of mesh points in the iron regions of the cavity and records the value in CON(12). The user has no control of this CON.
CON(13)	0	NINTER	Number of interface points. The default is an initial value. An interface point is a point whose nearest neighbors are a mixture of air points and iron points. See CON(11) and CON(12). The user has no control of this CON.
CON(14)	none	none	Not used in SUPERFISH problems.
CON(15)	none	NPINP	Total number of points in the problem. NPINP = NAIR + NFE + NINTER + NBND + NSPL. The user has no control of this CON.
CON(16)	0	NBND	Number of Dirichlet boundary points. Default is an initial value. LATTICE counts these points and stores the number in NBND. The user has no control of this CON.
CON(17)	0	NSPL	Number of points held at special fixed potential values. Default is an initial value. SUPERFISH counts these points and stores the number in NSPL. The user has no control of this CON.
CON(18)	0	NPERM	The number of sets of data defining the relative permittivity and permeability in regions with material code MATER > 1. The program will ask NPERM times for an input line of the form "MATER EPSIL FLOMU", where MATER is the material code number in the region having relative permittivity EPSIL and permeability FLOMU.

Constant	Default	Symbol	Function
CON(19)	1	ICYLIN	A flag indicating coordinate system to be used. ICYLIN = 1 indicates cylindrical coordinates using (horizontal, vertical) = (Z, R). Note that these axes are interchanged relative to those used in POISSON. ICYLIN = 0 indicates two-dimensional (X, Y) coordinates. CON(19) must be changed in SUPERFISH or earlier.
CON(20)	1	none	Not used in SUPERFISH problems.
CON(21)	1	NBSUP	An indicator for the type of boundary condition on the upper boundary. NBSUP = 0 indicates a Dirichlet boundary condition, which means electric field lines are parallel to the boundary line. NBSUP = 1 indicates a Neumann boundary condition, which means that the electric field lines are perpendicular to the boundary line. The default value passed by AUTOMESH is shown. AUTOMESH will pass the other value if IBOUND on the REG input line is used. See Sec. B.3.3. The default value if LATTICE is used alone is zero.
CON(22)	0	NBSLO	An indicator for the type of boundary condition on the lower boundary. See CON(21) for description.
CON(23)	1	NBSRT	An indicator for the type of boundary condition on the right boundary. See CON(21) for description.
CON(24)	1	NBSLF	An indicator for the type of boundary condition on the left boundary. See CON(21) for description.
CON(25)	none	NAMAX	Not used in SUPERFISH problems.
CON(26)	none	NWMAX	Not used in SUPERFISH problems.
CON(27)	none	NGMAX	Not used in SUPERFISH problems.
CON(28)	none	NGSAM	Not used in SUPERFISH problems.

Constant	Default	Symbol	Function
CON(29)	0	LIMTIM	Not used in SUPERFISH problems.
CON(30)	10	MAXCY	Maximum number of iteration cycles to find the resonance.
CON(31)	none	none	Not used in SUPERFISH problems.
CON(32)	0	IPRINT	An indicator for print options: IPRINT = −1 in LATTICE writes the (X, Y) coordinates of mesh points to OUTLAT. IPRINT = 1 (or any odd integer) writes a map of the solution matrix A into OUTFIS.
CON(33)	none	NOT	A parameter used in SFO1 to select an output tape number. The user has no control of this CON.
CON(34)	-1	INACT	An indicator to allow the user to interact with the frequency iteration in SUPERFISH. If INACT $\neq$ −1, the calculation is stopped at intervals and the user is asked to type: "GO", "NO", or "IN". If "GO", iteration continues; if "NO", iteration stops and final results are written; if "IN", user is asked for new values of CON's.
CON(35)	0	NODMP	An indicator controlling the write to TAPE 35. If NODMP = 0, a dump is written; if NODMP $\neq$ 0, no dump is written.
CON(36)	none	NSEG	The number of boundary segments passed to LATTICE from AUTOMESH by TAPE73. This CON is used in LATTICE only.
CON(37)	1	NCELL	The number of cells in multicell problems. It is used in SFO1 to calculate the transit time factor.

Constant	Default	Symbol	Function
CON(38)	none	KSTART	The value of the horizontal logical-mesh coordinate associated with the starting point of the present line segment on which power dissipation is to be calculated. This CON is used in SFO1; the user has no control of this CON.
CON(39)	none	LSTART	The value of the vertical logical-mesh coordinate associated with the starting point of the present line segment on which power dissipation is to be calculated. This CON is used in SFO1; the user has no control of this CON.
CON(40)	none	KEND	The value of the horizontal logical-mesh coordinate associated with the end point of the present line segment on which power dissipation is to be calculated. This CON is used in SFO1; the user has no control of this CON.
CON(41)	none	LEND	The value of the vertical logical-mesh coordinate associated with the end point of the present line segment on which power dissipation is to be calculated. This CON is used in SFO1; the user has no control of this CON.
CON(42)	1	none	Not used in SUPERFISH problems.
CON(43)	0	KTOP	A number set in LATTICE but not used in LATTICE, SUPERFISH, TEKPLOT or SFO1. It was probably used in a postprocessor called SHY.
CON(44)	1	none	Not used in SUPERFISH problems.
CON(45)	1	none	Not used in SUPERFISH problems.
CON(46)	none	ITYPE	Not used in SUPERFISH problems.
CON(47)	0.125	none	Not used in SUPERFISH problems.

Constant	Default	Symbol	Function
CON(48)	0	none	Not used in SUPERFISH problems.
CON(49)	0	none	Not used in SUPERFISH problems.
CON(50)	0	NPEG	The number of boundary segments on which power dissipation and frequency perturbations are to be calculated. If entered interactively, the program will ask for segment numbers.
CON(51)	0	NPONTS	In LATTICE this is the number of unknown relaxation points in the mesh. In SUPERFISH, NPONTS = NAIR + NINTER. NPONTS is used as the end-of-a-loop index.
CON(52)	0.001	OMEGA0	A parameter used in calculating over-relaxation factors in LATTICE. It is not used in the remainder of the SUPERFISH problem.
CON(53)	25	IRMAX	An index for checking the progress of the relaxation process in LATTICE; not used in the remainder of the SUPERFISH problem.
CON(54)	0.0	none	Not used in SUPERFISH problems.
CON(55)	0.0	none	Not used in SUPERFISH problems.
CON(56)	0.0	none	Not used in SUPERFISH problems.
CON(57)	0.0	none	Not used in SUPERFISH problems.
CON(58)	2.997925E+10	CLIGHT	The speed of light in vacuum in cm/sec.
CON(59)	π	PI	PI is given to machine accuracy, namely, $\pi = 4. * ATAN(1.)$.
CON(60)	none	KDEL	A parameter used in SFO1 in subroutines PATH and RFOUT; the user has no control of this CON.

Constant	Default	Symbol	Function
CON(61)	none	LDEL	A parameter used in SFO1 in subroutines PATH and RFOUT; the user has no control of this CON.
CON(62)	0.0	NSTEP	The number of steps in the variable k^2 used in a search for new resonances. SUPERFISH makes NSTEP steps through a range of k^2 values determined by CON(63), CON(65) and CON(66).
CON(63)	0.0	DELKSQ	The size of the steps taken in the k^2 search described in CON(62) above.
CON(64)	none	FREQS	The starting value of FREQ = CON(65). It is printed in SFO1. The user has control of this CON.
CON(65)	0.0	FREQ	When entered interactively, it is the starting value for the iteration to find a resonant frequency or the starting value of the frequency used in the k^2 search described in CON(62). During the iteration or search, it is the value of the frequency for the present step.
CON(66)	0.0	XKSQ	Initially it is the starting value of k^2, namely, $XKSQ = (2. * \Pi * FREQ/CLIGHT) **2$. During the run, it is the value of k^2.
CON(67)	none	DKSQ	The change in k^2 at the present step of the search described under CON(62).
CON(68)	none	XKO or SMALLK	XKO = SQRT(XKSQ); only used in SFO1.
CON(69)	none	none	Not used in SUPERFISH problems.
CON(70)	none	none	Not used in SUPERFISH problems.

Constant	Default	Symbol	Function
CON(71)	0	NEGAT	A flag indicating a zero or negative area triangle in the mesh. This may occur in the relaxation of the mesh in LATTICE and NEGAT $\neq 0$ will generate a diagnostic message.
CON(72)	0.0	ERG	A number proportional to the energy in the rf field at the present iteration. If ICYLIN = 1, ERG = $\int H^2 R dR dZ$; if ICYLIN = 0, ERG = $\int H^2 dX dY$.
CON(73)	0	IPIVOT	A control parameter for pivoting during the block matrix inversion process. If IPIVOT = 0, no pivoting; if IPIVOT = 1, partial pivoting; and IPIVOT = 2; complete pivoting. See Sec. C.13.4.
CON(74)	none	ASCALE	A scaling factor for the electric field chosen in such a way that $\int E_z dZ/L =$ VSCALE, where the integral is along a path parallel to the Z-axis, L is the length of the path, and VSCALE = CON(100).
CON(75)	none	POWER	The power dissipated on the conducting boundaries defined after entering a value for CON(50). CON(75) is calculated in SFO1.
CON(76)	0.0	ERGY or ENERGY	The energy stored in the rf field contained in the volume defined by the boundaries of the problem.
CON(77)	0.0	EMAX	The maximum value of the electric field found on any boundary segment entered after entering a value for CON(50).

Constant	Default	Symbol	Function
CON(78)	1	LINT	The logical L coordinate of the line along which the the integral $\int E_z dZ$ is calculated for the normalization of the electric field. See CON(74). Note that if LINT $\neq$ 1 or if no vertical coordinate of a point along the logical line LINT = 1 is zero, then the transit time factors are not calculated in SFO1.
CON(79)	1.6	RHOXY	The initial value of the X and Y mesh over-relaxation factors in LATTICE.
CON(80)	none	none	Not used in SUPERFISH problems.
CON(81)	1.0	RSTEM	The radius in centimeters of the stem assumed to be sticking into the cavity for the purpose of holding the drift tube in place. Used for the power dissipation and frequency perturbation calculations.
CON(82)	none	none	Not used for SUPERFISH problems.
CON(83)	0	IABORT	An abort flag in LATTICE, SUPERFISH, and SFO1. If IABORT = 1, the run is stopped.
CON(84)	1.0E-05	EPSO	A parameter to test for convergence in the mesh generation. Used in LATTICE only.
CON(85)	5.0E-07	none	Not used in SUPERFISH problems.
CON(86)	1.0E-04	EPSIK	A parameter to test for convergence of the frequency solution; the convergence is satisfied when $\lvert\Delta k^2\rvert/k^2 < EPSIK$.
CON(87)	0	IRESID	A flag to indicate whether the residual of the solution matrix A should be calculated. If IRESID = 1, the residual is calculated.
CON(88)	1.0	RESIDA	The residual of the solution matrix A.

Constant	Default	Symbol	Function
CON(89)	1.0	RESIK	The value of $\Delta k^2/k^2$ for the present iteration.
CON(90)	0	ICYCLE	The present iteration number; used in LATTICE and SUPERFISH.
CON(91)	0	NUMDMP	Present dump number for writing to TAPE35.
CON(92)–(99)	none	none	This set of eight words stores the title of the program, which was read by LATTICE
CON(100)	1.0E+06	VSCALE	A normalization factor for the average electric field. See CON(74). VSCALE is to be entered in V/m.
CON(101)	none	none	Not used in SUPERFISH problems.
CON(102)	600000000_8	IAMASK	A mask used in LATTICE to isolate bits in certain words.
CON(103)	200000000_8	ISCAT	A mask used in LATTICE to isolate bits in certain words.
CON(104)	4000000000_8	IFILT	A mask used in LATTICE to isolate bits in certain words.
CON(105)	100000_8	IDIRT	A mask used in LATTICE to isolate bits in certain words.
CON(106)	0.0	BETA	The velocity of the particles traversing the cavity divided by the velocity of light. If BETA is not entered interactively, it will be calculated from ZCTR = CON(107) and DPHI = CON(108), assuming the particles are protons. BETA is used in SFO1.

Constant	Default	Symbol	Function
CON(107)	0.0	ZCTR	The longitudinal coordinate of the "synchronous particle" when the electric field is a maximum. Usually this is the geometric center of the gap between two drift tubes in an Alverez linac. ZCTR is used in SFO1.
CON(108)	180.0	DPIII	The change in the phase of the rf field as the "synchronous particle" crosses the portions of the cavity defined by the boundaries put into SUPERFISH. The units are degrees; DPIII is used in SFO1.
CON(109)	none	ITOT	ITOT = (KMAX + 2)*(LMAX + 2).
CON(110)	none	T	The transit time factor; $T = \frac{1}{E_0 L} \int_{-L/frac}^{L/2} E(z) \cos \frac{2\pi z}{L} dz$
CON(111)	none	TP	$TP = \frac{1}{E_0 L^2} \int_{-L/frac}^{L/2} z E(z) \sin \frac{2\pi z}{L} dz$
CON(112)	none	TPP	$TPP = \frac{1}{E_0 L^3} \int_{-L/frac}^{L/2} z^2 E(z) \cos \frac{2\pi z}{L} dz$
CON(113)	none	S	$S = \frac{1}{E_0 L} \int_{-L/frac}^{L/2} E(z) \sin \frac{2\pi z}{L} dz$
CON(114)	none	SP	$SP = \frac{1}{E_0 L^2} \int_{-L/frac}^{L/2} z E(z) \cos \frac{2\pi z}{L} dz$
CON(115)	none	SPP	$SPP = \frac{1}{E_0 L^3} \int_{-L/frac}^{L/2} z^2 E(z) \sin \frac{2\pi z}{L} dz$
CON(116)	37_8	MASK37	A mask used in LATTICE to isolate bits in certain words.
CON(117)	77777_8	MASK5	A mask used in SUPERFISH to isolate bits in certain words.
CON(118)	none	MAXDIM	The maximum allowed value of ITOT = CON(109).
CON(119)	none	NWDIM	NWDIM = MAXDIM/2, where MAXDIM = CON(118).

Constant	Default	Symbol	Function
CON(120)	377_8	MASKC1	A mask for the eighth character in a word.
CON(121)	117400_8	MASKC2	A mask for the seventh character in a word.
CON(122)	none	TSTART	The wall clock starting time for the codes that contains this variable.
CON(123)	2.6544E-3	SQEM	The quantity SQRT(EPSO/FMUO), which is one over the impedance of empty space. In spite of being specifically defined in SUPERFISH, it is not used in the code.
CON(124)	8.8542E-14	EPS0	The permittivity of free space in "code units", i.e., depends on CONV = CON(9); the default code units are rationalized CKS. This CON appears not to be used in SUPERFISH problems. It is calculated from FMUO and CLIGHT.
CON(125)	$4 * \Pi * 1.0E - 09$	FMUO	The permeability of free space in "code units." Not used in SUPERFISH problems. See CON(124).

C.13.6 Table of Probcons in Alphabetical Order For SUPERFISH

ASCALE	CON(74)
BETA	CON(106)
CLIGHT	CON(58)
CONV	CON(9)
DELKSQ	CON(63)
DKSQ	CON(67)
DPHI	CON(108)
EMAX	CON(77)
EPSIK	CON(86)
EPSOH	CON(84)
EPS0	CON(124)
ERG	CON(72)
ERGY or ENERGY	CON(76)
FMUO	CON(125)
FREQ	CON(65)
FREQS	CON(64)
IABORT	CON(83)
IAMASK	CON(102)
ICYCLE	CON(90)
ICYLIN	CON(19)
IDIRT	CON(105)
IFILT	CON(104)
IMAX	CON(5)
INACT	CON(34)
IPIVOT	CON(73)
IPRINT	CON(32)
IRESID	CON(87)
IRMAX	CON(53)
ISCAT	CON(103)
ITOT	CON(109)
ITYPE	CON(46)
KDEL	CON(60)
KEND	CON(40)
KMAX	CON(4)
KPROB	CON(1)
KSTART	CON(38)
KTOP	CON(43)

LDEL	CON(61)
LEND	CON(41)
LIMTIM	CON(29)
LINT	CON(78)
LMAX	CON(3)
LSTART	CON(39)
MASK37	CON(116)
MASK5	CON(117)
MASKC1	CON(120)
MASKC2	CON(121)
MAXCY	CON(30)
MAXDIM	CON(118)
MODE	CON(6)
NAIR	CON(11)
NAMAX	CON(25)
NBND	CON(16)
NBSLF	CON(24)
NBSLO	CON(22)
NBSRT	CON(23)
NBSUP	CON(21)
NCELL	CON(37)
NEGAT	CON(71)
NFE	CON(12)
NGMAX	CON(27)
NGSAM	CON(28)
NINTER	CON(13)
NODMP	CON(35)
NOT	CON(33)
NPEG	CON(50)
NPERM	CON(18)
NPINP	CON(15)
NPONTS	CON(51)
NREG	CON(2)
NSEG	CON(36)
NSPL	CON(17)
NSTEP	CON(62)
NUMDMP	CON(91)
NWDIM	CON(119)
NWMAX	CON(26)
OMEGA0	CON(52)
PI	CON(59)
POWER	CON(75)

RESIDA	CON(88)
RFQ	CON(8)
RESIK	CON(89)
RHOXY	CON(79)
RSTEM	CON(81)
S	CON(113)
SP	CON(114)
SPP	CON(115)
SQEM	CON(123)
T	CON(110)
TP	CON(111)
TPP	CON(112)
TSTART	CON(122)
VSCALE	CON(100)
XKO or SMALLK	CON(68)
XKSQ	CON(66)
ZCTR	CON(107)

Chapter C.14

References for Part C

1. A. M. Winslow, "Numerical Solution of the Quasilinear Poisson Equation in a Nonuniform Triangular Mesh," J. Comp. Phys., (1967) pp. 2149–72.

2. K. Halbach and R. F. Holsinger, "SUPERFISH, a Computer Program for the Evaluation of RF Cavities with Cylindrical Symmetries," Part. Accel., 7 (1976) 213–22.

3. K. Halbach, R. F. Holsinger, W. E. Jule, and D. A. Swenson, "Properties of the Cylindrical RF Cavity Evaluation Code SUPERFISH," Proc. 1976 Proton Linear Accelerator Conf., Chalk River, Canada, September 14th – 17th, 1976.

4. R. L. Gluckstern, R. F. Holsinger, K. Halbach, and G. N. Minerbo, "ULTRAFISH — Generalization of SUPERFISH to $m > 1$," Proc. 1981 Linear Accelerator Conference, Santa Fe, New Mexico, USA, October 19th – 23rd, 1981.

5. R. L. Gluckstern, R. D. Ryne, and R. F. Holsinger, "Numerical Programs for Obtaining Accurate Resonant Frequencies of Modes in Azimuthally Symmetric Electromagnetic Cavities," Proceedings 1983 COMPUMAG Conference, Genoa, Italy.

6. R. L. Gluckstern, "RF Systems: Electromagnetic Fields in RF Cavities and Cavity Chains," Physics of High Energy Particle Accelerators (Stony Brook Summer School, 1983) AIP Conference Proceedings No. 87, New York, 1984.

7. G. A. Loew, et al, "Computer Calculations of Traveling-Wave Periodic Structure Properties," IEEE Trans. NS26 (1979) 3701.

8. A. G. Daikovsky, Yu. I. Portugalov, and A. D. Ryabov, "PRUD — Code for Calculation of the Nonsymmetric Modes in Axial Symmetric Cavities," Part. Accel. 12, (1982) pp. 59–64.

9. W. Wilhelm, "CAVIT and CAV3D — Computer Programs for RF Cavities with Constant Cross Section or any Three Dimensional Form," Part. Accel. 12, (1982) pp. 139–45.

10. B. M. Fomel, V. P. Jackowlev, M. M. Karliner, and P. B. Lysyansky, "LANS — A New Code for the Evaluation of the Electromagnetic Fields and Resonance Frequencies of Axisymmetrical Cavities," Part. Accel. 11, (1981) pp. 173–9.

11. T. Weiland, "TBCI and URMEL — New Computer Codes for Wake Field and Cavity Mode Calculations," IEEE Trans. NS–30, pp. 2489–91 (1983).

12. J. D. Jackson, Classical Electrodynamics, John Wiley, New York, (1962) pp. 235–41.

13. H. G. Herewood in Linear Accelerators, edited by P. M. Lapostolle and A. L. Septier, North Holland Pub. Co., New York (1970), p. 28.

14. E. L. Ginzton, Microwave Measurements, McGraw-Hill Book Co. (1957), p. 438. See Also L.C. Maier Jr. and J. C. Slater, J. App. Phys. 23 (1952) 68.

15. A. Carne, et al, Linear Accelerators, edited by P. M. Lapostolle and A. L. Septier, North Holland Pub. Co., New York (1970), pp. 747–83.

16. J. A. Stratton, Electromagnetic Theory, McGraw-Hill Book Co. (1941) pp. 131–7.

Chapter C.15

Index

accelerating field 1.9, 1.11
AMAX 7.2
arc of a circle 3.19
BETA 8.2
boundary conditions 1.7
boundary indicator 3.8
C array 3.6
CON 3.12
CON's 3.4
CONV 3.5 3.14
Convergence criterion 5.3
coordinates, logical 3.9
coordinates, physical 3.9
coupling coefficients 1.10, 1.12
CUR 3.14
DELKSQ 5.2
DEN 3.14
direct method 1.14
Dirichlet 3.8
DPHI 8.2
drift tube stem 8.2
drive point 1.13, 1.15, 3.7, 3.8
DTL 2.1
DX 3.14
DY 3.14
energy density 1.9, 1.10
EPSIK 5.3
EPSO 3.6
equipotential lines 7.2
field lines 7.2, 13.4
format-free 3.1
FREQ 5.2
frequency ω 1.7
frequency perturbation 1.12
Helmholtz eigenvalue 1.7
Helmholz equation 1.3
hyperbolas 3.19
IBOUND 3.14
ICYLIN 3.12, 5.1
INACT 5.2,
INAP 7.2
interactive control 5.2
interchange axes 7.2
IPIVOT 3.15, 5.1, 5.2
IPRINT 3.5, 4.1
IREG 3.15
IRESID 5.3
ITRI 7.2, 3.15
KMAX 3.15
KREG1 ..... 3.15
KREG2 ..... 3.15
LINT 8.2
LINX 3.16
LINY 3.16
LMAX 3.15
LREG 3.15
LREG1 ..... 3.15
magnetic charge 13.1
magnetic current 13.1
MAT 3.16
MATER EPSIL FLOMU 5.1
material code 3.6
MAXCY 5.1
maximum field 1.9, 1.11

mesh, secondary 1.13
modes 1.5
modes, cartesian 1.7
modes, cylindrical 1.5
modes, cylindrical, TE 1.6
modes, cylindrical, TM 1.5
multicell 8.2
multicell cavities 3.7
NAMELIST 3.1
NBSLF 3.5, 3.22
NBSLO 3.5, 3.5, 3.22
NBSRT 3.22
NBSUP 3.5, 3.22
NCELL 3.16, 8.2
NDRIVE 3.16
Neumann 3.8
NEW 3.18
NODMP 5.2
normalization 1.13, 8.2
NPERM 5.1
NPHI 7.2
NPOINT 3.16
NREG 3.5, 3.16
NSEG 3.5, 8.2
NSTEP 5.2
NSWXY 3.12
NSWXY 7.2
NT 3.18
NUM 7.2
OUTAUT 2.4
OUTLAT 4.1
OUTTEK 7.3
particle velocity 8.2
permeability 1.6
permittivity 1.6
perturbation analysis 2.1
perturbation, stem 1.12
perturbations 8.2
pivoting 5.2
power loss 1.10
power loss, walls 1.9
powerloss, stems 1.9
Poynting theorem 13.4
quality factor 1.9, 1.11
REG NAMELIST 3.12
region number 3.6
region, line 3.10
region, overwrite 3.10
region, point 3.10
reluctivity 1.6
resonance conditions 13.5
resonance, false 13.6
resonant frequency 5.2
rf-phase 8.2
RHOXY 3.5
RSTEM 8.2
search in k^2 ..... 5.2
shunt impedance 1.9, 1.11
Slater's theorem 1.12
special K 3.10
Stoke's Theorem 1.13
TAPE35 ..... 2.6
TAPE73 ..... 2.4, 3.3
(TE) modes 13.2
THETA 3.18
(TM) modes 13.2
transit time factors 1.9, 1.12
triangles, irregular 3.9
type of triangle 3.7
units 1.13
VSCALE 8.2
wavevector k 1.7
wavevector k, perturbation 1.9
X0 ..... 3.18
XKSQ 5.2
XMAX 3.16
XMIN 3.16
XMIN 7.2
XREG1 ..... 3.16
XREG2 ..... 3.16
Y0 ..... 3.18
YMAX 3.16
YMIN 3.17, 7.2
YREG1 ..... 3.17

YREG2 ..... 3.17
ZCTR 8.2

*U.S. Government Printing Office: 1992-673-036/67044