

UNCLASSIFIED

AD NUMBER

AD884920

LIMITATION CHANGES

TO:

Approved for public release; distribution is unlimited.

FROM:

Distribution authorized to U.S. Gov't. agencies only; Test and Evaluation; JUN 1971. Other requests shall be referred to Naval Research Lab., Washington, DC 20390.

AUTHORITY

NRL ltr 28 Jan 1986

THIS PAGE IS UNCLASSIFIED

AD 884920

AD NO. _____
DDC FILE COPY



SYSTEMS, SCIENCE AND SOFTWARE

20
ma

3SR-405-A

THE ZAP LASER ANALYSIS PROGRAM

Report Prepared By:

J.H. Alexander

M. Troost

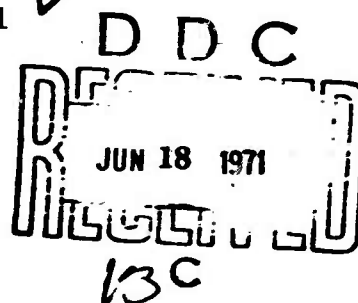
J.E. Welch

This effort was sponsored by the
Advanced Research Projects Agency

ARPA Order No. 660

Contract No. N00014-70-C-0341

June 1971



The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the Advanced Research Projects Agency or the U.S. Government.

346

DISCLAIMER NOTICE

THIS DOCUMENT IS THE BEST
QUALITY AVAILABLE.

COPY FURNISHED CONTAINED
A SIGNIFICANT NUMBER OF
PAGES WHICH DO NOT
REPRODUCE LEGIBLY.

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author)

Systems, Science and Software
P.O. Box 1620
La Jolla, California

2a. REPORT SECURITY CLASSIFICATION

Unclassified

2b. GROUP

3. REPORT TITLE

THE ZAP LASER ANALYSIS PROGRAM

4. DESCRIPTIVE NOTES (Type of report and inclusive dates)

Final Report

5. AUTHOR(S) (First name, middle initial, last name)

James H. Alexander
M. Troost
J.E. Welch

6. REPORT DATE

June 1971

7a. TOTAL NO. OF PAGES

7b. NO. OF REFS

0

8a. CONTRACT OR GRANT NO.

ARPA Order No. 660

8b. PROJECT NO.

c.

d.

9a. ORIGINATOR'S REPORT NUMBER(S)

3SR-405-A

9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)

10. DISTRIBUTION STATEMENT

11. SUPPLEMENTARY NOTES

12. SPONSORING MILITARY ACTIVITY

Advanced Research Projects Agency

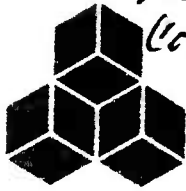
13. ABSTRACT

This report presents a detailed description of the ZAP computer program. It is used to determine the distribution of energy deposition within a laser assembly. The program is written in FORTRAN IV. The calculation uses ray tracing and Monte Carlo techniques to obtain a solution. The program is applicable to a wide range of geometries and a spectrum of wave lengths.

DD FORM 1473
1 NOV 65

Security Classification

Distribution limited to U.S. Gov't. agencies only;
Test and Evaluation; *June 71* . Other requests
for this document must be referred to



*Naval Research Laboratory
Code 1265, wach DC 20390*

SYSTEMS, SCIENCE AND SOFTWARE

3SR-405-A

THE ZAP LASER ANALYSIS PROGRAM

by

**J.H. Alexander
M. Troost
J.E. Welch**

Final Report

**Principal Investigator: Dr. Marius Troost (714) 453-0060
Project Scientist: Mr. C. W. Hartley (ONR)**

**Sponsored by
Advanced Research Projects Agency
ARPA Order No. 660**

**Program Code Number: 6 23 01 D
Contract No: N00014-70-C-0341
Effective Date of Contract: 1 May 1970
Contract Expiration Date: 31 December 1970
Amount of Contract: \$44,000**

June 1971

P. O. BOX 1620, LA JOLLA, CALIFORNIA 92037, TELEPHONE (714) 453-0060

PREFACE

The ZAP laser analysis program was written for the Naval Research Laboratory, Washington, D.C., under contract N00014-70-C-0341. The Principal Investigator was Dr. M. Troost. The code was programmed and debugged by J.E. Welch. Dr. J.H. Alexander contributed strongly to the geometrical analysis, in particular the Helix. Dr. A.G. Petschek defined the statistical analysis. Dr. J. Trenholme of NRL made many worthwhile suggestions for improving the program and contributed the second sample problem. The excellent cooperation by NRL personnel throughout the contract period contributed significantly to the success of this contract.

ABSTRACT

This report presents a detailed description of the ZAP computer program. It is used to determine the distribution of energy deposition within a laser assembly. The program is written in FORTRAN IV. The calculation uses ray tracing and Monte Carlo techniques to obtain a solution. The program is applicable to a wide range of geometries and a spectrum of wave lengths.

TABLE OF CONTENTS

	Page
PREFACE	ii
ABSTRACT.	iii
1. METHODS USED.	1
1.1 Introduction	1
1.2 Ray Tracing.	1
1.3 Monte Carlo and Russian Roulette	2
1.4 Accuracy	3
1.5 Multiple Wavelengths	3
1.6 Geometry Specifications.	4
1.7 Segments	4
1.8 Surface Definitions.	6
1.8.1 Plane	7
1.8.2 Cylinder.	7
1.8.3 Conic Cylinder.	7
1.8.4 Sphere.	8
1.8.5 Half-cone	8
1.8.6 Paraboloid of Revolution.	8
1.8.7 Prolate Elipsoid of Revolution.	8
1.8.8 Hyperboloid of Revolution	8
1.8.9 Helical Tube.	8
1.8.10 Boundary Intersections.	9
1.9 New Segment Determination.	10
1.10 Normal to the Surface.	11
1.11 Angle of Reflection.	11
1.12 Angle of Refraction.	11
1.13 Percent of Reflection.	11
1.13.1 Fresnel's Reflection for Dielec- trics	12
1.13.2 No Reflection	12
1.13.3 Total Reflection.	12

TABLE OF CONTENTS, Contd.

	Page
1.13.4 Constant Reflection	12
1.13.5 Table Reflection.	12
1.13.6 Function Reflection	13
1.13.7 Diffusion Reflection.	13
1.13.8 Wavelength Dependent Reflection .	13
1.14 Source Definition.	13
1.14.1 Surface Source.	13
1.14.2 Volume Source	14
1.15 Energy Deposition.	14
1.16 Logic Flow	15
1.17 Other Uses	16
2. THE PROGRAM	17
2.1 Programming Philosophy	17
2.2 Sources.	18
2.2.1 Surface Source.	18
2.2.2 Volume Source	20
2.2.3 Ray Input	21
2.3 Reflection	22
2.3.1 Specular Reflection	22
2.3.2 Diffuse Reflection.	23
2.3.3 Percent of Reflection for Dielec- trics	25
2.4 Refraction	26
2.5 Debugging.	28
2.5.1 Printer Plots	28
2.5.2 Random Errors	31

TABLE OF CONTENTS, Contd.

	Page
2.6 Geometry	32
2.6.1 Equations for a Plane	32
2.6.2 Equations for a Cylindrical Sur- face.	35
2.6.3 Equations for a Sphere.	39
2.6.4 Equations for a Conic Cylinder. .	43
2.6.5 Equations for a Cone.	50
2.6.6 Equations for an Ellipsoid of Revolution.	54
2.6.7 Equations for a Paraboloid of Revolution.	58
2.6.8 Equations for a Hyperboloid of Revolution.	60
2.6.9 Equations for a Helix	64
3. INPUT — OUTPUT.	66
3.1 Setup Considerations	66
3.1.1 Miscellaneous Rules	67
3.1.2 User Routines	68
3.2 Input.	72
3.2.1 General Input (three cards) . . .	73
3.2.2 Reflection Data	75
3.2.3 Material Data	76
3.2.4 Boundary Data	77
3.2.5 Segment Data.	81
3.2.6 Structure Data.	81
3.2.7 Lamp Data	82
3.2.8 Source Data	83
3.2.9 Ray Data.	84
3.2.10 Test Data	84
3.2.11 Plot Data	86

TABLE OF CONTENTS, Contd.

	Page
3.3 Output	87
3.4 Sample Problem #1.	89
3.5 Sample Problem #2.	102
3.5.1 Introduction.	102
3.5.2 Initial Analysis of the Geometry.	102
3.5.3 Input Preparation	105
3.5.4 Discussion of Output.	108
3.5.5 Conclusions	111
APPENDIX A	A1
A.1 Helix.	A1
A.1.1 Introduction.	A1
A.1.2 The Helix	A2
A.1.3 The Helical Pipe.	A4
A.1.4 The Ray and the Helical Pipe.	A5
A.1.5 The Turns of the Helix.	A6
A.1.6 The Sequence of the Turns	A7
A.1.7 Numbering the Turns	A12
A.1.8 Seven Special Points on a Ray	A14
A.1.9 The Distance from a Point to a Line.	A15
A.1.10 The Distance from a Point to a Turn.	A16
A.1.11 The Inside of the Pipe.	A17
A.1.12 The Distance from the Ray to a Turn.	A19
A.1.13 Finding the Intersections	A20
A.1.14 Finding the Roots of $B(\theta) = a^2/2$	A23
A.1.15 Assembling the Pieces	A26

TABLE OF CONTENTS, Contd.

	Page
APPENDIX B	B1
B.1 Automatic Segmentation	B1
B.2 Input for LAZY	B2
APPENDIX C — ZAP FLOW DIAGRAM AND PROGRAM LISTING. . .	C1
APPENDIX D — ZAP DATA SHEETS	D1

LIST OF ILLUSTRATIONS

Number		Page
2.1	A plane whose normal is not parallel to any of the major axis	41
2.2	Cylinder whose axis is not parallel to any of the major axis	42
2.3	Sphere	44
2.4	Conic cylinder with axis parallel to Z-axis.	44
2.5	Cone with axis parallel to Z-axis.	51
2.6	Ellipsoid of Revolution with axis parallel to Z-axis	55
2.7	Paraboloid of revolution with axis parallel to Z-axis	55
2.8	Hyperboloid of Revolution.	61
3.1	A simple laser cavity used for sample problem #1	90
3.2	Top view of laser system used in sample problem #1	91

LIST OF ILLUSTRATIONS, Contd.

Number		Page
3.3	Side view of laser system used in sample problem #1	92
3.4	Top view of right half of laser system used in sample problem #1.	93
3.5	Top view of left half of laser system used in sample problem #1.	94
3.6	Top view of lower half of laser system used in sample problem #1.	95
3.7	General view of laser analyzed. A 1 cm diameter xenon flashlamp pumps a 1 cm diameter ND^{3+} - doped glass rod. Both are 10 cm long, and are enclosed in an 11 cm diameter spherical reflector. Cylindrical end pieces hold the rod and lamp	103
3.8	The decomposition of the laser assembly for ZAP input. Boundaries (indicated by numbers starting with B) are chosen so that all needed segments (indicated by numbers starting with S) can be outlined. The mother-daughter relationships are shown by arrows pointing from mother to daughter(s), or by inclusion in the mother. The laser rod is segmented so that the energy deposition pattern can be found . .	104
A.1	A view of the helix and ray looking down the axis of the helix.	A9
A.2	Another top view of the helix and ray.	A10
A.3	Traces of typical rays: 1) $\kappa\mu/\gamma < 0$, 2) $0 < \kappa\mu/\gamma < b$, 3) $b < \kappa\mu/\gamma$, 4) ray with 8 intersections	A13

1. METHODS USED

1.1 INTRODUCTION

Several methods might be employed to calculate the distribution of the energy deposition within a laser assembly. Analytical calculations are impossible due to the complexity of the geometry. Numerical solutions of the relevant wave equations should be possible, but the computer time cost for three-dimensional calculations is prohibitive. Ray tracing, which is frequently used in optical problems, seems to be the most practical method available.

1.2 RAY TRACING

In ray tracing, rays may be selected systematically or at random. In simple geometries, systematic selection of rays is advantageous, as only a few are needed for a good sample. In complicated 3-dimensional geometries, the number of rays required for an adequate systematic sampling becomes prohibitively large, and randomly selected rays from known distributions will give more reliable results for the same amount of computer time. As many laser designs are of considerable complexity, the ZAP program uses randomly selected rays from specified distributions. Each ray is defined by the three directional cosines, the coordinates of the starting point, and an intensity distribution as a function of wavelength.

A problem for any method of ray tracing in a design with a large amount of internal reflection and no black absorbers, is to decide when to stop following a particular ray. As each ray loses its energy very gradually via partial absorption, it is impractical to follow the ray until it is completely absorbed; without a black absorber this would take

a very large number of reflections. If a ray is ignored only after its energy falls below a negligible level (e.g., 1% of its original strength), the paradoxical situation exists where most of the tracing effort is used for rays that contribute very little to the final result. If, on the other hand, each ray is cut off when its energy is still significant (e.g., 25% of its original strength), larger errors will be made. A method is needed that does not ignore weak rays, but that does not waste too much effort on them.

1.3 MONTE CARLO AND RUSSIAN ROULETTE

By randomly selecting the rays to be traced, the problem is transformed into one that can be solved by Monte Carlo methods. A technique frequently used in Monte Carlo solutions is the Russian Roulette Technique. This technique allows each ray to be cut off while its energy is still significant without ignoring the weak rays. Basically this method involves statistically eliminating some weak rays, while strengthening other weak rays for further tracing. In practice, the Russian Roulette method works as follows.

A ray is followed until its intensity falls below a certain predetermined level (e.g., 50% of original intensity). At that point, it is decided randomly whether this ray will survive. If it dies, it is ignored and no record is kept. If it survives, its intensity is multiplied by a factor which is the inverse of the probability of survival in the random test. The intensity of the surviving ray is therefore always relatively high (e.g., above 50% of the original intensity). As each ray may several times pass through this procedure before being killed, even very weak rays are statistically represented in the sample. Partial reflections may be treated by creating a new ray of appropriate intensity. A pass through the Russian Roulette routine will then decide whether the new ray survives and with what intensity.

Energy deposition is determined by calculating the exponential attenuation along the path of a ray. The deposition is assigned to the segment through which the ray passed. The intensity is checked at every intersection of the ray with a boundary, and a game of Russian Roulette is played when necessary. Eventually each ray will therefore be killed in a game of Russian Roulette.

1.4 ACCURACY

The accuracy of the energy absorbed in the various regions of the system is dependent on the number of rays traced. Several runs may be necessary to determine the number of rays needed for the accuracy required.

1.5 MULTIPLE WAVELENGTHS

Multiple wavelengths are handled in the ZAP program by assigning intensities for each wavelength to a single ray. For this technique to be applied it must be assumed that the angle of refraction at a material interface is independent of wavelength, which is an approximation. The absorption coefficients and the percent reflected may vary with wavelength.

As the ray is traced through the system, the amount absorbed for different wavelengths varies. Therefore some wavelengths will be destroyed by Russian Roulette while other wavelengths survive. The ray is traced until all wavelengths have lost their intensity either by absorption or by Russian Roulette.

The energy deposited in a material by a given ray is the sum of the contributions of the various wavelengths.

1.6 GEOMETRY SPECIFICATIONS

A flexible method for describing the geometry of the system has been developed which allows simple input and fast computation. At the same time, it allows for great generality in geometrical configuration.

A problem common to many Monte Carlo codes is their extreme slowness when applied to complicated geometries. This slowness is caused by the need to check for possible intersections with all possible boundaries. The running time thus becomes roughly proportional to the number of boundaries in the system. It becomes quickly evident that some technique is needed which eliminates the necessity for checking all boundaries for each intersection.

One possible solution to this problem is to divide space into subregions and to check only for possible intersections with those boundaries that exist in the subregion and with the boundaries of the subregion itself. Some modern Monte Carlo codes use this method, but they are often inconvenient to use because of the cumbersome geometrical definitions required for each subregion.

Systems, Science and Software (S³) has developed a method which uses the subdivision method recursively. The geometry is specified by a greatly simplified geometrical input, as the subregions used are the natural geometrical shapes of the system.

1.7 SEGMENTS

Each part of space in the problem to be studied is divided into one or more segments. A single piece of material may belong to several segments or a segment may contain several materials in the form of subsegments. Two kinds of segments are defined: Mother segments and Daughter segments.

The Mother segments act as subregions and the Daughter segments lie within Mother segments. When a photon lies in a particular Mother segment, only the intersections with boundaries of that Mother segment and its Daughter segments need to be checked. The boundaries of the Mother segment shield the photon therefore from the details in the space outside the Mother segment.

It frequently is also desirable to shield the photon from details in certain areas within the Mother segment. This may be done by enclosing these details in a Daughter segment and declaring this Daughter segment to be a Mother segment at a lower level. We now have a recursive definition of Mother and Daughter segments, where any Mother segment may be a Daughter segment of a higher level Mother and any Daughter segment may be a lower level Mother segment.

The geometry of the complete system fits into one Mother segment. This Mother segment usually will have several Daughter segments that are Mother segments in turn.

This recursive subdivision may be continued to any desired depth. Any Daughter segment, that is not a Mother segment, must consist of a homogeneous material.

A Mother segment is a segment that is partially or completely filled by one or more Daughter segments. That part of space within a Mother segment that is not occupied by Daughter segments has the material properties of the Mother segment. The volume of the Mother segment is the volume of the space within the Mother segment that is not occupied by Daughter segments.

Whenever possible, Mother segments should have simple shapes, e.g., rectangular parallelepiped, cylinder, sphere. A simple shape allows an efficient check for boundary crossings and reduces computer time. However, in principle a

Mother segment may have any shape, even though that may be expensive in computer time.

Although the possible number of Daughter segments within a Mother segment is unlimited, in practice this number should be quite low (e.g., below 5) to reduce the number of boundary crossing checks. After all, the Mother segment was introduced to reduce the number of potential boundary crossings.

A Daughter segment must lie completely within its Mother segment. Each Daughter segment is therefore associated with only one Mother segment. Each Daughter segment may in turn be a lower level Mother segment.

For each Mother segment, all of the external boundaries must be specified. Boundaries of Daughters need not be specified for the Mother. For each Daughter segment, all of the confining boundaries must be specified.

In specifying the boundaries of a given segment, it is necessary to know on which side of the boundary the segment lies. For a plane, one must specify if the segment is on the origin or non-origin side of the plane. This is accomplished by specifying a positive boundary number for the origin side and a negative boundary number to indicate that the segment is on the non-origin side. For curved surfaces, a positive boundary number means that the segment is inside the surface and a negative boundary number indicates that the segment lies outside the boundary.

1.8 SURFACE DEFINITIONS

The geometric surfaces of the system are defined conveniently for vector calculations. In theory, any mathematically definable geometric shape can be used with the ZAP program by adding the appropriate subroutines to the

computer code. The definitions for the geometry types of most general interest are summarized below:

1.8.1 Plane

A plane is defined by the directional cosines of the normal to the plane and the coordinate of a point which lies on the plane (Fig. 2.1).

1.8.2 Cylinder

A cylindrical surface is defined by the directional cosines of the cylindrical axis, the coordinates of a point on the axis, and the radius (Fig. 2.2).

1.8.3 Conic Cylinder

A conic cylinder whose major axis is perpendicular to the Z axis (Fig. 2.4) is defined by the directional cosines of the major axis of the conic, the X-Y coordinates of the focus, and the X-Y coordinates of two points on the conic surface. The two points must not be symmetric relative to the axis, although one of them may lie on the axis.

Whether the conic represents a parabola, hyperbola, or ellipse is defined by the value of the eccentricity, which can be calculated from the data provided.

parabola: $e_k = 1$

hyperbola: $e_k > 1$

ellipse: $e_k < 1$

1.8.4 Sphere

A sphere is defined by the coordinates of the center of the sphere and the radius (Fig. 2.3).

1.8.5 Half-cone

A half-cone whose axis of symmetry is parallel to the Z axis (Fig. 2.5) is defined by the coordinates of the vertex, and coordinates of a point on the surface of the half cone.

1.8.6 Paraboloid of Revolution

A paraboloid of revolution whose axis of symmetry is parallel to the Z axis (Fig. 2.7) may be defined by specifying the coordinates of the vertex and the Z coordinate of the focus.

1.8.7 Prolate Ellipsoid of Revolution

A prolate ellipsoid of revolution whose axis of symmetry is parallel to the Z axis (Fig. 2.6) is defined by the coordinates of one of the vertices, the Z coordinate of the focus nearest the chosen vertex, and the coordinates of a point on the surface.

1.8.8 Hyperboloid of Revolution

One sheet of a hyperboloid of revolution whose axis of symmetry is parallel to the Z axis (Fig. 2.8) is defined by the vertex of the sheet of interest, the Z coordinate of the focus of the sheet of interest and the coordinates of a point on the surface. This point may lie on either sheet.

1.8.9 Helical Tube

A helical tube must have its major axis parallel to the Z axis. A point on the axis is chosen such that the

perpendicular from the point to the centerline of the helical tube lies along the X axis and points from the point to the centerline in the positive X direction. The perpendicular distance from the axis to the centerline is specified and the radius of the tube must be specified. The only other parameter required is proportional to the distance between successive coils.

1.8.10 Boundary Intersections

In order to determine which boundary is next intersected by a ray, it is necessary to calculate along the direction of the ray the distance to all pertinent boundaries and choose the boundary which gives the smallest positive distance. Because of the segmentation, it is not necessary to calculate the distance to all of the boundaries in the system, but merely to a subset of these boundaries defined by the current Mother and its daughters. This is accomplished as follows:

1. When a photon is inside a Daughter segment, the following boundaries should be checked:
 - a. The boundaries of the current Daughter segment.
 - b. The boundaries of the Mother segment.
2. When a photon is inside a Mother segment, but not inside any of its Daughter segments, the following boundaries should be checked:
 - a. The boundaries of all Daughter segments in this Mother segment.
 - b. The boundaries of the Mother segment.

1.9 NEW SEGMENT DETERMINATION

After a photon has crossed a boundary, the segment containing the photon must be determined. This requires checking each boundary of the segment to determine if the photon lies on the desired side of the boundary. If the photon lies on the wrong side of any boundary of a segment, it is not in that segment. This search is conducted as follows:

1. Is the photon still in the Mother Segment?
If the answer is yes,
2. Check each Daughter segment until one is found that contains the photon.

If the photon does not lie in any of the Daughter segments, it lies by definition in the Mother segment of space. If the photon lies within a Daughter segment, a check must be made to see if this Daughter segment is a Mother segment at a lower level. If such is the case, the procedure must be repeated starting at (2) for this new Mother segment.

If the photon lies outside the current Mother segment, the procedure should be repeated starting with (1) for the next higher level Mother segment.

The search thus continues until the final segment containing the photon has been found. This might involve climbing several Mother levels and then descending several Mother levels. The algorithm is written recursively, and is not concerned with the absolute Mother level. When a photon is found outside the highest level Mother, which defines the boundaries of the complete geometry, an error stop will occur.

1.10 NORMAL TO THE SURFACE

In order to calculate the angle of incidence and the angle of refraction for a ray intersecting a surface, it is necessary to determine the directional cosines of the normal to that surface at the point of intersection. The equations used to calculate the directional cosines of the normal depend on the geometry type of the surface.

1.11 ANGLE OF REFLECTION

When a ray intersects a surface, part or all of the ray reflects off the surface at some angle which must be determined. In the case of specular reflection, this angle is the same as the angle of incidence, but in a different direction. With a diffuse or Lambertian surface, the ray reflects with an angle chosen at random from a known distribution.

1.12 ANGLE OF REFRACTION

When part or all of a ray is transmitted through a real surface, the transmitted ray may be refracted. The directional cosines of the refracted ray must then be calculated.

1.13 PERCENT OF REFLECTION

Each boundary is assigned a reflection type which defines the method used to determine the percent of the ray that is reflected. Each reflection type is designed to handle a particular type of boundary that might be necessary in setting up a geometrical system. All reflections are assumed to be specular with the exception of diffuse reflection. The following types of reflection are currently in the ZAP program.

1.13.1 Fresnel's Reflection for Dielectrics

The surface of laser rods and lamps or any other type of glass component will generally be represented by Fresnel's reflection.

1.13.2 No Reflection

A non-reflective or dummy boundary is used primarily for segmentation purposes. The material should be the same on each side of the boundary. Hence there is no refraction either.

1.13.3 Total Reflection

Total reflection boundaries are used primarily for boundaries of symmetry, but can also be used for other purposes where one wishes the impending ray to be totally reflected.

1.13.4 Constant Reflection

Rays are reflected with constant percent of the intensity going with the reflected ray regardless of material properties or the angle of incidence.

1.13.5 Table Reflection

Rays are reflected by a percent which is dependent on the angle of incidence. A table is provided which gives the percent as a function of angle, and interpolation in the table gives the percent to be used.

1.13.6 Function Reflection

Any number of functions might be used to give the percent of reflection. One of particular interest is

$$\text{percent reflection} = 1 - A \cos^2 \theta$$

where

θ = angle of incidence

A = a coefficient between 0 and 1.

1.13.7 Diffusion Reflection

In the case of a diffuse surface (see angle of reflection), the percent to be reflected is considered to be a constant which may be chosen for the particular surface under consideration.

1.13.8 Wavelength Dependent Reflection

The percent of the ray to be reflected may be dependent only on the wavelength, or may be dependent on the wavelength and the angle of incidence.

1.14 SOURCE DEFINITION

1.14.1 Surface Source

One type of source that is available is a surface source for a cylinder whose axis is parallel to the Z axis. A random point is chosen on the surface of the cylinder, and a ray is started from that point. The direction of the ray is determined by a subroutine that is provided by the user. This subroutine provides an angle θ with the normal to the surface at the point, and an angle ϕ with the cylinder axis in the plane tangent to the point. The distribution of the rays is therefore under user control, and can be

chosen to fit the model desired. From these two angles, the directional cosines of the ray can be calculated. An option provides for the rays to be directed outwards from the cylinder surface or inwards from the cylinder surface.

1.14.2 Volume Source

If desired, a volume source may be used. For a volume source, a starting point is chosen at random within the source segment. A direction is then chosen in such a way as to give isotropic emission. This is accomplished by choosing another random point within a unit sphere having the starting point as the center. These two points uniquely define the ray and the directional cosines are easily calculated.

More than one source may be included within a given system. For each source the number of rays to be created is input and an initial energy distribution for the desired wavelengths must be provided. This distribution may be read in from cards or may be input by providing a subroutine to produce the desired results.

1.15 ENERGY DEPOSITION

When a ray passes through an absorbing material, energy is deposited in the material for each of the wavelengths being considered. For each wavelength the ray intensity is decayed exponentially, and the amount lost by the ray is deposited in the material through which the ray has passed. The amount lost by different wavelengths is dependent on the distance traveled and the absorption coefficient of the material for each wavelength. It is therefore necessary to provide an absorption distribution as a function of wavelength for each material.

1.16 LOGIC FLOW

The flow of logic using the ZAP program is as follows:

1. Generate a ray or continue with a leg of the previous ray.
2. Determine the number of the segment containing the ray.
3. Find the distance to the intersection point or points for each of the appropriate boundaries and choose the boundary with the smallest positive distance.
4. Deposit energy in the segment through which the ray has just passed.
5. Determine the segment number on the opposite side of the intersected boundary.
6. Find the normal to the surface at the point of intersection.
7. Calculate the angle of reflection, the angle of refraction and the percent of the ray to be reflected. Calculate the directional cosines of the reflected and transmitted rays.
8. Play Russian Roulette with each wavelength of the reflected and transmitted rays. If the wavelength survives, double its intensity. If it does not survive, zero its intensity. The same random number is used for each wavelength.

9. Determine if the transmitted ray still has live wavelengths. If it does, start again with step 3. If it does not, go back to step 1.

The above procedure is repeated until the desired number of rays have been traced to extinction. After all rays have been completed, the final segment and material results are printed, and the problem ends.

1.17 OTHER USES

The ZAP program is a technique for solving optics problems of many types. Although it was developed for the analysis of laser efficiency it is also a general optics program that may be applied to a wide range of problems.

An example of another use is in the area of designing reflectors for spotlights. The problem would be set up with a reflector and spherical light source within the reflector. At some desired distance in front of the reflector a black nonreflecting plane would be placed, and the material behind the plane would be divided up into segments. The final absorbed energy distribution in the target segments would give a very effective measure of the focusing ability of the reflector. Many optics problems involving light sources and eventual light absorption may be solved with the ZAP program.

2. THE PROGRAM

2.1 PROGRAMMING PHILOSOPHY

The ZAP program was developed on the UNIVAC 1108 and was written in basic FORTRAN IV and is as machine independent as is possible. A version exists for the CDC-3800.

Throughout the development of the program, every effort was made to make the program as general as possible. In those cases that require special geometric shapes not available in the present code, new geometry routines may be incorporated in the program by adding the name of the geometry to a table and writing four subroutines to handle the tasks required for each geometric shape. Likewise, new types of reflection or ray creation may be added in the same manner.

As far as is possible, each subroutine or function in the code handles a given task without regard to what has happened before the routine was called. In that way these routines may be called from various sections of the program to perform the same task for different purposes. Obviously this is not always possible, but a definite effort was made in that direction whenever possible. One obvious drawback to this is the resulting large number of subroutines, some of which are so short as to be almost ridiculous. However, it was felt that the advantages far outweigh the disadvantages. Changing the way a particular task is performed usually means replacing a small subroutine rather than making changes to a large subroutine.

The remainder of this section presents the mathematical equations used for source generation, reflection and refraction. The equations used for each of the allowable geometries are given next.

2.2 SOURCES

After a ray has been absorbed or eliminated by Russian Roulette, a new ray must be created. The intensity distribution of the new ray is acquired from a table (ESOR(N,F)) that was set up at the beginning of the run, where N = the source number and F is the wave length index.

The position and direction of the new ray depends on the type of source being used.

Let

$$\left. \begin{array}{l} x_i \\ y_i \\ z_i \end{array} \right\} = \text{the coordinates of the starting point of the ray.}$$

$$\left. \begin{array}{l} \alpha_i \\ \beta_i \\ \gamma_i \end{array} \right\} = \text{the directional cosines of the ray.}$$

2.2.1 Surface Source

For a surface source the new ray is created at the surface of a cylinder whose axis is parallel to the Z axis. The ray may be directed either inwards or outwards from the surface.

Let

- θ_1 = an angle in the plane perpendicular to the cylinder axis
 θ = the angle of the ray with the normal to the surface
 ϕ = an angle with the cylinder axis in the plane perpendicular to the normal
 Z_{MIN} = minimum z value for starting point of ray
 Δz = distance along the z axis for starting point of ray
 x_k, y_k = coordinate of a point on the cylinder axis.
 $RAND$ = a random number between 0 and 1.

Choose a random angle θ_1 $0 \leq \theta_1 \leq 2\pi$

$$x_i = x_k + (d_k \pm \epsilon) * \cos \theta_1$$

$$y_i = y_k + (d_k \pm \epsilon) * \sin \theta_1$$

$$z_i = Z_{MIN} + \Delta z * RAND.$$

In order to calculate the directional cosines of the ray it is first necessary to find the directional cosines of the normal to the surface at the point (x_i, y_i, z_i) .

$$\left. \begin{matrix} \alpha_n \\ \beta_n \\ \gamma_n \end{matrix} \right\} = \text{directional cosines of the normal to the surface at the starting point}$$

The angles θ and ϕ are determined by a special subroutine. This subroutine must be provided by the user, and will be written to give the desired distribution of rays.

$$0 \leq \theta \leq \frac{\pi}{2}$$

$$0 \leq \phi \leq 2\pi$$

then

$$\alpha_i = \alpha_n \cos \theta + \beta_n \sin \theta \sin \phi$$

$$\beta_i = \beta_n \cos \theta - \alpha_n \sin \theta \sin \phi$$

$$\gamma_i = \cos \theta \sin \phi$$

If it is desired for the ray to point inwards from the surface, the signs of its directional cosines (α_i , β_i , γ_i) are reversed.

2.2.2 Volume Source

For a volume source the new ray is created at a random point within the source segment and the direction of the ray is chosen to give isotropic emission.

The random starting point is determined by creating a random point (x_i , y_i , z_i) within a rectangular parallelepiped that encloses the source segment. Then a check is made to determine if the point lies within the segment. If not, new random points are chosen until one is found that does lie within the segment.

To find the directional cosines, random components of a vector are found.

$$\Delta x = \text{a random number } (-1 \leq \Delta x \leq 1)$$

$$\Delta y = \text{a random number } (-1 \leq \Delta y \leq 1)$$

$$\Delta z = \text{a random number } (-1 \leq \Delta z \leq 1)$$

then

$$R = \sqrt{\Delta x^2 + \Delta y^2 + \Delta z^2}$$

If $R \leq 1$, the random components are used. If not, other sets of random components are determined until an appropriate set is found.

Finally

$$\alpha_i = \Delta x/R$$

$$\beta_i = \Delta y/R$$

$$\gamma_i = \Delta z/R$$

2.2.3 Ray Input

The third way to create new rays is to input the starting points and directional cosines of a systematically selected set of rays. Then when a new ray is needed, it is available in a table. This feature is primarily useful in debugging.

2.3 REFLECTION

2.3.1 Specular Reflection

When a ray intersects a specular surface, part or all of the ray is reflected at the same angle with the normal and in the same plane that contains the normal and the impinging ray. The directional cosines of the reflected ray are calculated as follows:

$$b = -2(\alpha_i \alpha_n + \beta_i \beta_n + \gamma_i \gamma_n)$$

$$\alpha_r = \alpha_i + b\alpha_n$$

$$\beta_r = \beta_i + b\beta_n$$

$$\gamma_r = \gamma_i + b\gamma_n$$

where

$$\left. \begin{array}{l} \alpha_i \\ \beta_i \\ \gamma_i \end{array} \right\} = \text{the directional cosines of the ray}$$

$$\left. \begin{array}{l} \alpha_n \\ \beta_n \\ \gamma_n \end{array} \right\} = \text{the directional cosines of the normal to the surface.}$$

2.3.2 Diffuse Reflection

Diffuse (Lambertian) reflection is calculated by giving the reflected ray a pseudo-random direction with a Lambertian distribution. Two angles (θ and ϕ) represent the angle between the reflected ray and the normal to the surface, and the angle the projection of the ray onto a plane perpendicular to the normal makes with some line in that plane, respectively.

By definition, a Lambertian distribution is such that the probability distribution $P(\theta)$ varies as $\cos \theta$, while $P(\phi)$ is constant. The problem then is to randomly choose θ and ϕ in a manner that gives the desired probability distributions. One method of doing this is to choose a random ϕ ($0 \leq \phi \leq 2\pi$) and a random $\cos^2 \theta$ ($0.0 \leq \cos^2 \theta \leq 1.0$). That is not what has been done in the ZAP code however.

If the reflected ray is a unit vector, the probability distribution of the projection of that ray onto the plane perpendicular to the normal should be constant. This fact is the key to the technique used here.

With uniform random numbers R_i ($0 \leq R_i \leq 1$) we get the diffusely reflected directional cosines of the ray relative to the normal ($\vec{N}$) and two orthogonal axes ($\vec{P}, \vec{Q}$) in the perpendicular plane as follows:

Let

α_R = projection of ray onto $\vec{P}$

β_R = projection of ray onto $\vec{Q}$

γ_R = projection of ray onto $\vec{N}$

Then choose

$$\alpha_R = 2R_i - 1$$

$$\beta_R = 2R_{i+1} - 1$$

If $\alpha_R^2 + \beta_R^2 > 1.0$, choose two new random numbers and calculate new values for α_R and β_R .

Then

$$\gamma = \sqrt{1.0 - (\alpha_R^2 + \beta_R^2)}$$

Normal $\vec{N}$ has directional cosines $\alpha_n, \beta_n, \gamma_n$.

If $\alpha_n = 1.0$ choose

$$\alpha_p = 0$$

$$\beta_p = 1$$

$$\gamma_p = 0$$

Otherwise the directional cosines of $\vec{P}$ are

$$\alpha_p = \frac{1 - \alpha_n^2}{\sqrt{1 - \alpha_n^2}}$$

$$\beta_p = \frac{\alpha_n \beta_n}{\sqrt{1 - \alpha_n^2}}$$

$$\gamma_p = \frac{\alpha_n \gamma_n}{\sqrt{1 - \alpha_n^2}}$$

The directional cosines of $\vec{Q}$ are

$$\alpha_Q = \beta_n \gamma_P - \beta_P \gamma_n$$

$$\beta_Q = \alpha_P \gamma_n - \alpha_n \gamma_P$$

$$\gamma_Q = \alpha_n \beta_P - \alpha_P \beta_n$$

Finally, the directional cosines of the ray are:

$$\alpha_r = \alpha_R \alpha_P + \beta_R \alpha_Q + \gamma_R \alpha_n$$

$$\beta_r = \alpha_R \beta_P + \beta_R \beta_Q + \gamma_R \beta_n$$

$$\gamma_r = \alpha_R \gamma_P + \beta_R \gamma_Q + \gamma_R \gamma_n$$

2.3.3 Percent of Reflection for Dielectrics.

When a ray intersects the surface of a laser rod, lamp, or any other glass component, the percent of the ray to be reflected must be calculated. The ZAP code uses Fresnel's equation for reflection to calculate this percentage.

Let

$\cos a$ = cosine of the angle of incidence

$\cos b$ = cosine of the angle of refraction

REF_{m1} = refractive index of the material the ray is leaving.

REF_{m2} = refractive index of the material the ray is entering.

then

$$m = \frac{REF_{m1}}{REF_{m2}}$$

$$\cos a = |\alpha_i \alpha_n + \beta_i \beta_n + \gamma_i \gamma_n|$$

$$\cos b = |\alpha_{rf} \alpha_n + \beta_{rf} \beta_n + \gamma_{rf} \gamma_n|$$

$$r_s = \left(\frac{\cos b - m \cos a}{\cos b + m \cos a} \right)^2$$

$$r_p = r_s \left(\frac{\cos a \cos b - m(1 - \cos^2 a)}{\cos a \cos b + m(1 - \cos^2 a)} \right)$$

$$\text{Percent reflected} = \frac{r_s + r_p}{2}$$

2.4 REFRACTION

When part or all of a ray is transmitted through a surface separating different materials, the transmitted ray is refracted because of the difference in material densities. The directional cosines of the refracted ray are calculated from the following equations.

Let

$$\left. \begin{array}{l} \alpha_{rf} \\ \beta_{rf} \\ \gamma_{rf} \end{array} \right\} = \text{the directional cosines of the refracted ray}$$

REF_{m1} = the refractive index of the material
the ray is leaving

REF_{m2} = the refractive index of the material
the ray is entering

$\left. \begin{matrix} \alpha_n \\ \beta_n \\ \gamma_n \end{matrix} \right\}$ = the directional cosines of the normal
to the surface at the point of inter-
section .

$\left. \begin{matrix} \alpha_i \\ \beta_i \\ \gamma_i \end{matrix} \right\}$ = the directional cosines of the impend-
ing ray

then

$$a = \frac{REF_{m1}}{REF_{m2}}$$

$$c = \alpha_i \alpha_n + \beta_i \beta_n + \gamma_i \gamma_n$$

$$b = ac \left(-1 + \sqrt{1 + (1-a^2)/a^2 c^2} \right)$$

$$\alpha_{rf} = a\alpha_i + b\alpha_n$$

$$\beta_{rf} = a\beta_i + b\beta_n$$

$$\gamma_{rf} = a\gamma_i + b\gamma_n$$

2.5 DEBUGGING

2.5.1 Printer Plots

A printer plot of the geometry cut by a plane may be obtained for as many different planes as desired. In each case a rectangular section of the plane is defined as input. The geometry within this rectangle is the geometry that will be plotted.

The technique used to generate the points that represent the geometric boundaries is as follows: fake rays are created and traced across the plane. Each time an intersection with a boundary occurs, the coordinates of the intersection are noted, and a blank is plotted. The material number is plotted between intersections. Rays are not reflected, and are traced until they leave the area of interest.

The only major restriction in defining the rectangular plotting area is that the rectangle must be completely enclosed by the overall mother region (region number 1). This is no real restriction since the overall mother can be made very large with no loss of computer time or accuracy.

The input required to define the rectangular plotting area consists of three points p_0 , p_1 , and p_2' . Point p_0 will become the lower left hand corner of the plot and point p_1 will be the lower right hand corner. Point p_2' will be on the top line of the plot.

Fake rays are created at even intervals along the lines that form the top and lefthand sides of the plotting area, and are traced perpendicular to the line on which they are created. The number of rays to be created is determined in such a way that one ray corresponds to each print line and one ray corresponds to each print column.

NX = number of vertical rays

NY = number of horizontal rays

Once an intersection point (x,y,z) has been found, the coordinate system must be rotated and translated to correspond to the plotting area. The equations needed are:

$$l_x = \sqrt{(x_1-x_0)^2 + (y_1-y_0)^2 + (z_1-z_0)^2}$$

l_x = length of line p_0p_1

$$\left. \begin{aligned} a_{xx} &= \frac{x_1-x_0}{l_x} \\ a_{xy} &= \frac{y_1-y_0}{l_x} \\ a_{xz} &= \frac{z_1-z_0}{l_x} \end{aligned} \right\} \text{directional cosines of } p_0p_1$$

Point p_2 must be found

$$k = a_{xx}(x'_2-x_0) + a_{xy}(y'_2-y_0) + a_{xz}(z'_2-z_0)$$

$$x_2 = x'_2 - k a_{xx}$$

$$y_2 = y'_2 - k a_{xy}$$

$$z_2 = z'_2 - k a_{xz}$$

then

$$l_y = \sqrt{(x_2-x_0)^2 + (y_2-y_0)^2 + (z_2-z_0)^2}$$

l_y = length of line p_0p_2

$$\left. \begin{aligned} a_{yx} &= \frac{x_2 - x_0}{l_y} \\ a_{yy} &= \frac{y_2 - y_0}{l_y} \\ a_{yz} &= \frac{z_2 - z_0}{l_y} \end{aligned} \right\} \text{directional cosines of } p_0 p_2$$

Rays are created at equal intervals along the top and lefthand side of the plotting area.

$NX = 130$ = number of print positions to be used

$\Delta s_x = l_x / (N - 1)$ = interval along horizontal direction.

$\Delta s_y = \frac{10}{6} \Delta s_x$ = interval along the vertical direction

$$Ny = l_y / \Delta s_y + 1.5$$

Horizontal ray number n has the starting coordinates:

$$x_s = x_2 - (n - .5) \Delta s_y a_{yx}$$

$$y_s = y_2 - (n - .5) \Delta s_y a_{yy}$$

$$z_s = z_2 - (n - .5) \Delta s_y a_{yz}$$

and directional cosines

$$\alpha = a_{xx}$$

$$\beta = a_{xy}$$

$$\gamma = a_{xz}$$

Vertical ray number n has the starting coordinates:

$$x_s = x_2 + (n-1)\Delta s_x a_{xx}$$

$$y_s = y_2 + (n-1)\Delta s_x a_{xy}$$

$$z_s = z_2 + (n-1)\Delta s_x a_{xz}$$

and directional cosines

$$\alpha = -a_{yx}$$

$$\beta = -a_{yy}$$

$$\gamma = -a_{yz}$$

For each intersection point x, y, z the plotting coordinates are calculated as

$$\bar{X} = a_{xx}(x-x_0) + a_{xy}(y-y_0) + a_{xz}(z-z_0)$$

$$\bar{Y} = a_{yx}(x-x_0) + a_{yy}(y-y_0) + a_{yz}(z-z_0)$$

2.5.2 Random Errors

The random errors in the results of the ZAP program are due to statistical variations in the number of rays passing through a particular segment and due to the variation in the amount of energy deposited by each ray passing through this segment. The latter variation is due to variations in the path length through the segment and variations in the ray energy upon entry into the segment. It turns out that the variation in the number of rays usually contributes to the largest fraction of the random error.

The random error is displayed in the output as a percentage error of one standard deviation. When the number of rays contributing to a segment is less than four, ***** is printed to indicate the lack of data. Large errors are by their very nature unreliable as their calculation is based on unreliable results. An error larger than 20% indicates unreliable results, where an error below 5% indicates usable results.

The percentage error is calculated according to the equation presented below:

$$ER_i = \sqrt{\frac{\sum E^2 - \frac{1}{N} (\sum E)^2}{\sum E}}$$

where

ER_i	Percentage error in segment i for one standard deviation
N	Total number of rays in the problem
E	Energy deposited in segment i by a ray
$\sum E$	Total energy deposited in segment i

2.6 GEOMETRY

2.6.1 Equations for a Plane

A plane is defined by the directional cosines of the normal to the plane and the coordinates of a point that lies on the plane. An additional piece of information needed in many of the equations is the distance from the origin to the plane. The directional cosines of the normal must be such

that the normal vector points from the origin to the plane. The calculation of the distance from the origin to the plane provides a check on the direction, since the distance will be negative if the vector points in the wrong direction.

Define:

$$\left. \begin{matrix} \alpha_k \\ \beta_k \\ \gamma_k \end{matrix} \right\} = \text{the directional cosines of the normal to the plane}$$

$$\left. \begin{matrix} x_k \\ y_k \\ z_k \end{matrix} \right\} = \text{the coordinates of a point on the plane}$$

2.6.1.1 Distance from the Origin to the Plane

$$d_k = \alpha_k x_k + \beta_k y_k + \gamma_k z_k$$

2.6.1.2 Intersection of a ray with a plane:

Let

$$L = \text{distance along the ray to the point of intersection}$$

$$\left. \begin{matrix} \alpha_i \\ \beta_i \\ \gamma_i \end{matrix} \right\} = \text{directional cosines of the ray}$$

$$\left. \begin{matrix} x_i \\ y_i \\ z_i \end{matrix} \right\} = \text{coordinates of the starting point of the ray}$$

then

$$L = \frac{d_k - (\alpha_k x_i + \beta_k y_i + \gamma_k z_i)}{\alpha_i \alpha_k + \beta_i \beta_k + \gamma_i \gamma_k}$$

The point of intersection (x,y,z) can then be calculated:

$$x = x_i + L\alpha_i$$

$$y = y_i + L\beta_i$$

$$z = z_i + L\gamma_i$$

2.6.1.3 Normal to the Point of Intersection

In the case of a plane, the normal to the point of intersection does not require an actual calculation, since the normal to the plane is the same everywhere and is part of the definition of the plane. Hence:

$$\alpha_n = \alpha_k$$

$$\beta_n = \beta_k$$

$$\gamma_n = \gamma_k$$

2.6.1.4 Position of Point Relative to Plane

In determining if a point is inside a given segment, it is necessary to determine on which side of a plane the point lies. This requires finding the perpendicular distance from the plane to the point.

$$D = \alpha_k x_i + \beta_k y_i + \gamma_k z_i - d_k$$

where

$$\left. \begin{array}{l} x_i \\ y_i \\ z_i \end{array} \right\} = \text{the coordinates of the point}$$

2.6.2 Equations for a Cylindrical Surface

A cylindrical surface is defined by the directional cosines of the cylinder axis, the coordinates of a point on the axis, and the radius of the cylinder.

Define

$$\left. \begin{array}{l} \alpha_k \\ \beta_k \\ \gamma_k \end{array} \right\} = \text{directional cosines of cylinder axis.}$$

$$\left. \begin{array}{l} x_k \\ y_k \\ z_k \end{array} \right\} = \text{coordinates of a point on the axis.}$$

d_k = the radius of the cylinder.

2.6.2.1 Intersection of a Ray with a Cylinder

Let

L = distance along the ray to the point of intersection.

$\left. \begin{matrix} \alpha_i \\ \beta_i \\ \gamma_i \end{matrix} \right\}$ = directional cosines of the ray.

$\left. \begin{matrix} x_i \\ y_i \\ z_i \end{matrix} \right\}$ = coordinates of the starting point of the ray

Then

$$\Delta x = x_i - x_k$$

$$\Delta y = y_i - y_k$$

$$\Delta z = z_i - z_k$$

$$xx = \alpha_i \alpha_k + \beta_i \beta_k + \gamma_i \gamma_k$$

$$yy = \alpha_i \Delta x + \beta_i \Delta y + \gamma_i \Delta z$$

$$zz = \alpha_k \Delta x + \beta_k \Delta y + \gamma_k \Delta z$$

$$(AA)^2 = \Delta x^2 + \Delta y^2 + \Delta z^2$$

$$L = \frac{(zz)(xx) - yy \pm \sqrt{(yy - zz^2 xx)^2 - (1 - xx)^2 (AA^2 - zz^2 - d_k^2)}}{(1 - xx^2)}$$

2.6.2.2 In or Out of a Cylinder

To determine if a point lies inside a cylinder, it is necessary to calculate the distance from the point to the cylinder axis and compare this distance with the cylinder radius.

Let

D = distance of point from cylinder axis

$$\left. \begin{array}{l} x_i \\ y_i \\ z_i \end{array} \right\} = \text{coordinates of a point}$$

Then

$$\Delta x = x_i - x_k$$

$$\Delta y = y_i - y_k$$

$$\Delta z = z_i - z_k$$

$$D^2 = \Delta x^2 + \Delta y^2 + \Delta z^2 - (a_k \Delta x + b_k \Delta y + c_k \Delta z)^2$$

if $D^2 < d_k^2$ point lies inside cylinder.

2.6.2.3 Normal to a Cylinder

Let

$$\left. \begin{array}{c} x \\ y \\ z \end{array} \right\} = \text{a point on the cylinder surface}$$

$$\left. \begin{array}{c} \alpha_n \\ \beta_n \\ \gamma_n \end{array} \right\} = \text{the directional cosines of the normal to the cylinder at the point } (x, y, z)$$

Calculate

$$\Delta x = x - x_k$$

$$\Delta y = y - y_k$$

$$\Delta z = z - z_k$$

$$C = \alpha_k \Delta x + \beta_k \Delta y + \gamma_k \Delta z$$

$$\alpha_n = \frac{\Delta x - C \alpha_k}{d_k}$$

$$\beta_n = \frac{\Delta y - C \beta_k}{d_k}$$

$$\gamma_n = \frac{\Delta z - C \gamma_k}{d_k}$$

2.6.3 Equations for a Sphere

A sphere is defined by the coordinates of the center and the radius.

Define:

$$\left. \begin{array}{l} x_k \\ y_k \\ z_k \end{array} \right\} = \text{the coordinates of the center of the sphere}$$

$$d_k = \text{the radius of the sphere}$$

2.6.3.1 Intersection of a Ray with a Sphere

Let

$$L = \text{distance along the ray to the point of intersection}$$

$$\left. \begin{array}{l} \alpha_i \\ \beta_i \\ \gamma_i \end{array} \right\} = \text{directional cosines of the ray}$$

$$\left. \begin{array}{l} x_i \\ y_i \\ z_i \end{array} \right\} = \text{coordinates of the starting point of the ray}$$

then

$$\Delta x = x_i - x_k$$

$$\Delta y = y_i - y_k$$

$$\Delta z = z_i - z_k$$

$$B = \Delta x \alpha_i + \Delta y \beta_i + \Delta z \gamma_i$$

$$C = \Delta x^2 + \Delta y^2 + \Delta z^2 - d_k^2$$

$$L = -B \pm \sqrt{B^2 - C}$$

2.6.3.2 In or Out of a Sphere

To determine if a point is inside a sphere, one must calculate the distance from the point to the center of the sphere and compare this with the radius.

Let

D = distance of point from center of sphere

$$\left. \begin{array}{l} x_i \\ y_i \\ z_i \end{array} \right\} = \text{coordinates of the point}$$

then

$$D^2 = (x_i - x_k)^2 + (y_i - y_k)^2 + (z_i - z_k)^2$$

If $D^2 < d_k^2$, the point is inside the sphere.

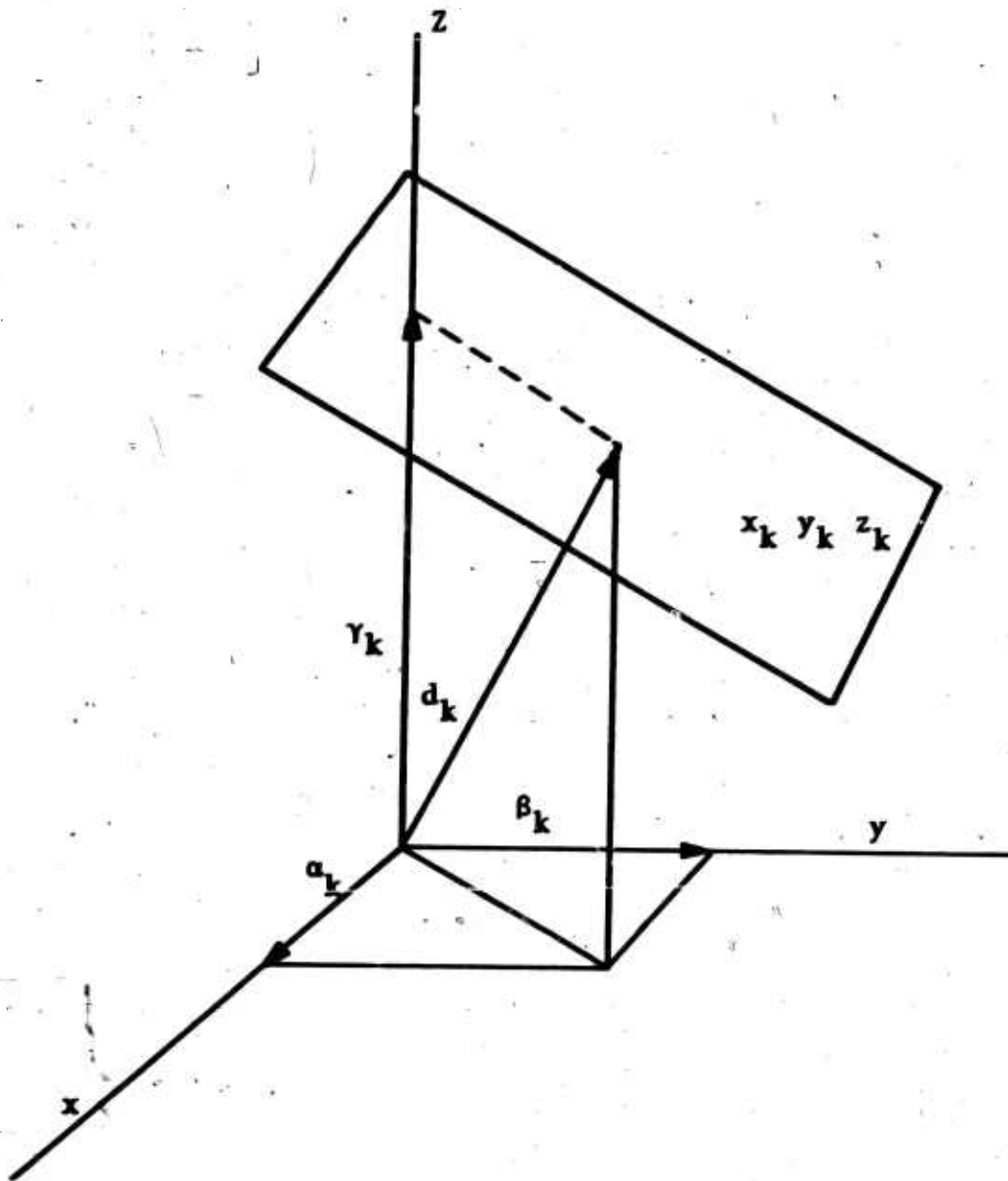


Figure 2.1 A plane whose normal is not parallel to any of the major axis.

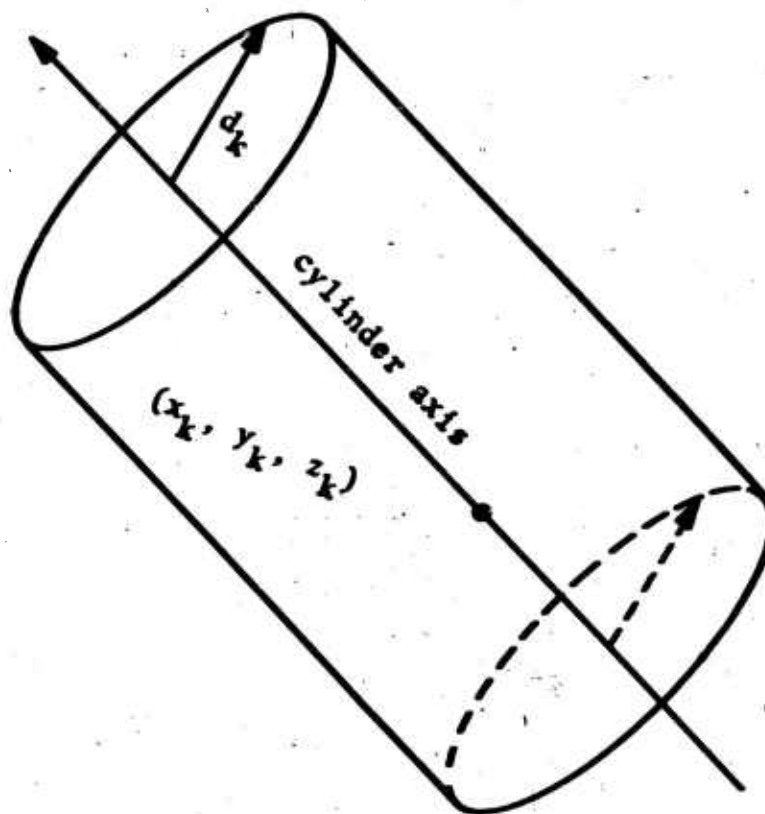


Figure 2.2 Cylinder whose axis is not parallel to any of the major axis.

2.6.3.3 Normal to a Sphere

Let

$$\left. \begin{array}{c} x \\ y \\ z \end{array} \right\} = \text{a point on the surface of the sphere}$$

$$\left. \begin{array}{c} \alpha_n \\ \beta_n \\ \gamma_n \end{array} \right\} = \text{the directional cosines of the normal to the surface at the point } (x, y, z)$$

Calculate

$$\alpha_n = (x - x_k) / d_k$$

$$\beta_n = (y - y_k) / d_k$$

$$\gamma_n = (z - z_k) / d_k$$

2.6.4 Equations for a Conic Cylinder

The conic cylinder is limited to one whose conic axis is perpendicular to the Z axis and whose cylindrical axis is parallel to the Z axis. In this case, only the x-y plane must be considered in defining the surface. The conic is defined by the coordinates of the focus, the directional cosines of the conic axis, and the coordinates of two points which lie on the surface. In addition, the eccentricity and the distance from the focus to the directrix are calculated for use in the equations. See Figure 2.4.

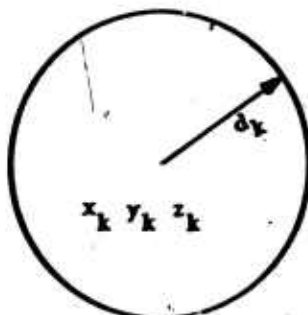


Figure 2.3 Sphere

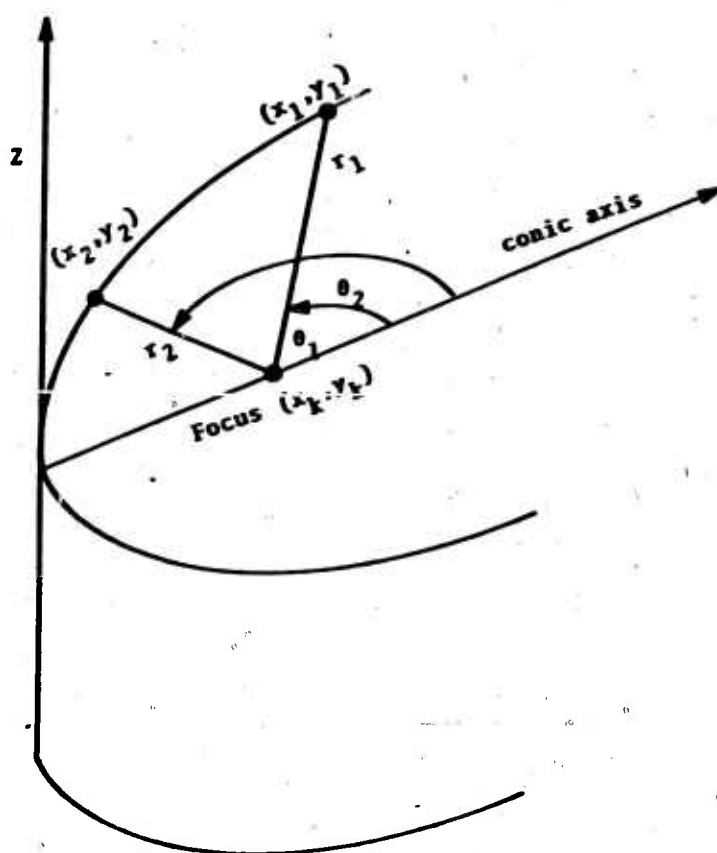


Figure 2.4 Conic cylinder with axis parallel to Z-axis

Define

$$\left. \begin{array}{l} \alpha_k \\ \beta_k \end{array} \right\} = \text{directional cosines of the conic axis}$$

$$\left. \begin{array}{l} x_k \\ y_k \end{array} \right\} = \text{coordinates of the focus}$$

$$\left. \begin{array}{l} x_1 \\ y_1 \\ x_2 \\ y_2 \end{array} \right\} = \text{coordinates of two points on the surface}$$

$$e_k = \text{eccentricity}$$

$$d_k = \text{distance from focus to directrix}$$

2.6.4.1 Eccentricity and Distance from Focus to Directrix

$$r_1 = \sqrt{(x_1 - x_k)^2 + (y_1 - y_k)^2}$$

$$r_2 = \sqrt{(x_2 - x_k)^2 + (y_2 - y_k)^2}$$

$$\alpha_1 = \frac{x_1 - x_k}{r_1}$$

$$\beta_1 = \frac{y_1 - y_k}{r_1}$$

$$\alpha_2 = \frac{x_2 - x_k}{r_2}$$

$$\beta_2 = \frac{y_2 - y_k}{r_2}$$

$$\cos \theta_1 = \alpha_k \alpha_1 + \beta_k \beta_1$$

$$\cos \theta_2 = \alpha_k \alpha_2 + \beta_k \beta_2$$

$$e_k = \frac{r_1 - r_2}{r_1 \cos \theta_1 + r_2 \cos \theta_2}$$

$$d_k = \frac{r_1 r_2 (\cos \theta_1 - \cos \theta_2)}{r_1 - r_2}$$

2.6.4.2 Intersection of a Ray with a Conic Cylinder

Let

$$\left. \begin{array}{l} x_i \\ y_i \\ z_i \end{array} \right\} = \text{coordinates of starting point of ray}$$

$$\left. \begin{array}{l} \alpha_i \\ \beta_i \\ \gamma_i \end{array} \right\} = \text{directional cosines of the ray}$$

$$L = \text{distance along the ray to the point of intersection.}$$

then

$$\Delta x = x_i - x_k$$

$$\Delta y = y_i - y_k$$

$$\cos \theta = \sqrt{1 - \gamma_i^2}$$

$$\alpha' = \alpha_i / \cos \theta$$

$$\beta' = \beta_i / \cos \theta$$

$$EE = \alpha' \Delta x + \beta' \Delta y$$

$$FF = \alpha_k \Delta x + \beta_k \Delta y$$

$$GG = \alpha_k \alpha' + \beta_k \beta'$$

$$BB^2 = \Delta x^2 + \Delta y^2$$

$$P = 1 - e_k^2 GG^2$$

$$H = EE - e_k^2 GG (FF + d_k^2)$$

$$Q = BB^2 - e_k^2 (FF + d_k^2)$$

$$L = \frac{-H \pm \sqrt{H^2 - PQ}}{P \cos \theta}$$

2.6.4.3 In or Out of a Conic Cylinder

To determine if a point (x_i, y_i) lies inside of a conic cylinder it is necessary to calculate the distance from the focus to the surface in the direction of the point, and the distance from the focus to the point and then compare the two distances.

Let

$$\left. \begin{array}{l} x_i \\ y_i \end{array} \right\} = \text{the coordinates of the point}$$

$$D = \text{the distance from focus to point}$$

$$R = \text{distance from focus to surface}$$

then

$$\Delta x = x_i - x_k$$

$$\Delta y = y_i - y_k$$

$$D = \sqrt{\Delta x^2 + \Delta y^2}$$

$$\alpha' = \Delta x / D$$

$$\beta' = \Delta y / D$$

$$R = \frac{d_k e_k}{1 - e_k (\alpha_k \alpha' + \beta_k \beta')}$$

if $D < R$, the point is inside the conic.

2.6.4.4 Normal to a Conic Cylinder

Let

$$\left. \begin{array}{l} \alpha_n \\ \beta_n \\ \gamma_n \end{array} \right\} = \text{directional cosines of the normal to the surface at the point } (x, y, z)$$

then

$$\Delta x = x - x_k$$

$$\Delta y = y - y_k$$

$$DD = \sqrt{\Delta x^2 + \Delta y^2}$$

$$\alpha' = \Delta x / DD$$

$$\beta' = \Delta y / DD$$

$$\cos \theta = \alpha_k \alpha' + \beta_k \beta'$$

$$\sin \theta = \alpha_k \beta' - \alpha' \beta_k$$

$$\alpha_n = \frac{\alpha_k \cos \theta - \beta_k \sin \theta - e_k \alpha_k}{\sqrt{1 + e_k^2 - 2e_k \cos \theta}}$$

$$\beta_n = \frac{\alpha_k \sin \theta + \beta_k \cos \theta - e_k \beta_k}{\sqrt{1 + e_k^2 - 2e_k \cos \theta}}$$

$$\gamma_n = 0$$

2.6.5 Equations for a Cone

A right circular half cone whose axis is parallel to the Z axis is defined by the coordinates of the vertex and the coordinates of a point on the surface of the cone. If the point lies above the vertex, the half cone is a cup. If it lies below the vertex the half cone is a cap (see Figure 2.5).

Define:

$$\left. \begin{array}{l} x_k \\ y_k \\ z_k \end{array} \right\} = \text{the coordinates of the vertex}$$

$$\left. \begin{array}{l} x_1 \\ y_1 \\ z_1 \end{array} \right\} = \text{the coordinates of a point on the surface}$$

$$d_k = \frac{(z_1 - z_k)^2}{(x_1 - x_k)^2 + (y_1 - y_k)^2 + (z_1 - z_k)^2}$$

$$e_k = z_1 - z_k$$

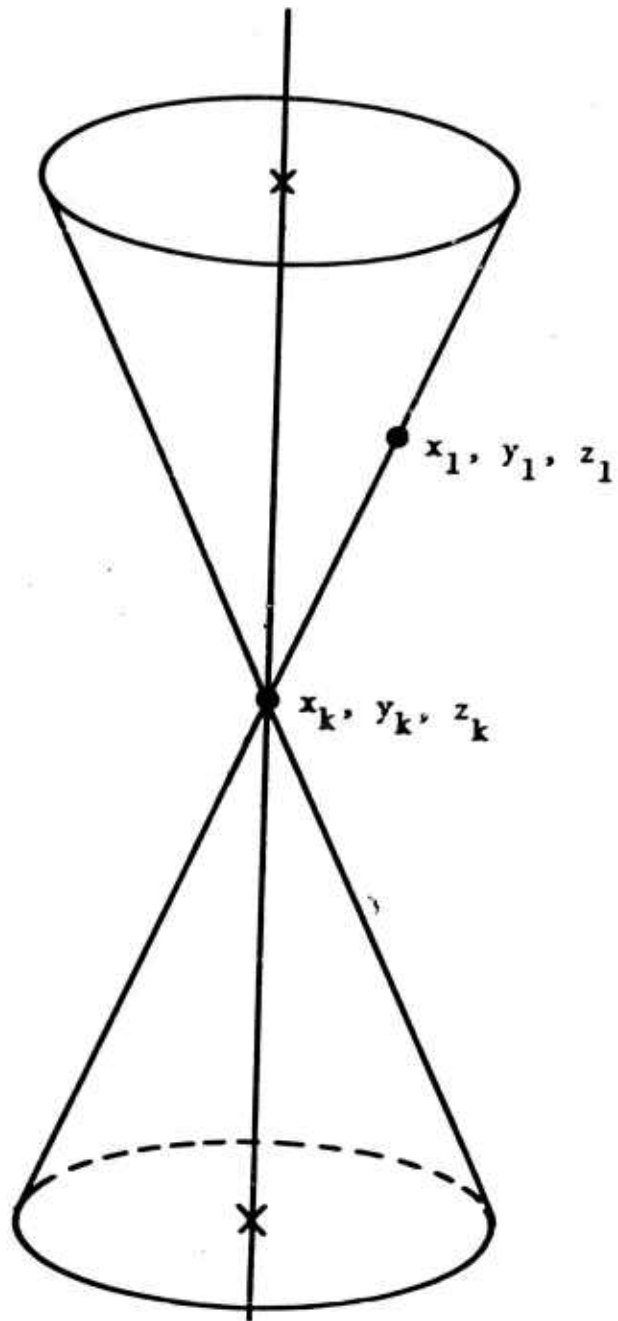


Figure 2.5 Cone with axis parallel to z-axis

2.6.5.1 Intersection of a Ray with a Half Cone

Let

$$\left. \begin{matrix} x_i \\ y_i \\ z_i \end{matrix} \right\} = \text{coordinates of the starting point of a ray}$$

$$\left. \begin{matrix} \alpha_i \\ \beta_i \\ \gamma_i \end{matrix} \right\} = \text{directional cosines of a ray}$$

$$L = \text{distance along the ray to the point of intersection.}$$

Then

$$\Delta x = x_i - x_k$$

$$\Delta y = y_i - y_k$$

$$\Delta z = z_i - z_k$$

$$A = \gamma_i^2 - d_k$$

$$B = (1 - d_k) \gamma_k \Delta z - d_k (\alpha_i \Delta x + \beta_i \Delta y)$$

$$C = (1 - d_k) \Delta z^2 - d_k (\Delta x^2 + \Delta y^2)$$

$$L_{\pm} = \frac{-B \pm \sqrt{B^2 - AC}}{A}$$

We must now choose the smallest positive length that gives an intersection point in the proper half of the cone. Let

$$e_+ = (z_i + \gamma_i L_+) (e_k)$$

$$e_- = (z_i + \gamma_i L_-) (e_k)$$

if e_+ is positive, L_+ gives an intersection in the proper half of the cone.

if e_- is positive, L_- gives an intersection in the proper half of the cone.

2.6.5.2 In or Out of a Half Cone

To determine if a point (x_i, y_i, z_i) lies inside of a half cone, the following procedure is used.

$$\Delta x = x_i - x_k$$

$$\Delta y = y_i - y_k$$

$$\Delta z = z_i - z_k$$

$$EP = \Delta z * e_k$$

$$FP = \Delta z^2 - d_k (\Delta x^2 + \Delta y^2 + \Delta z^2)$$

if $(FP)(EP) > 0$, the point lies inside the half cone.

2.6.5.3 Normal to a Cone

Let

$$\left. \begin{matrix} \alpha_n \\ \beta_n \\ \gamma_n \end{matrix} \right\} = \text{the directional cosines of the normal to the surface at the point } (x, y, z)$$

then

$$\Delta x = x - x_k$$

$$\Delta y = y - y_k$$

$$\Delta z = z - z_k$$

$$R = \sqrt{d_k^2 (\Delta x^2 + \Delta y^2) + (1 - d_k)^2 \Delta z^2}$$

$$\alpha_n = d_k \Delta x / R$$

$$\beta_n = d_k \Delta y / R$$

$$\gamma_n = -(1 - d_k) \Delta z / R$$

2.6.6 Equations for an Ellipsoid of Revolution

The ellipsoid of revolution must have its axis of revolution parallel to the Z axis. It is defined by the coordinates of one of the vertices (x_k, y_k, z_k) , the coordinates of a point on the surface (x_1, y_1, z_1) and the Z coordinate of the focus (z_f) nearest to the vertex chosen. See Figure 2.6.

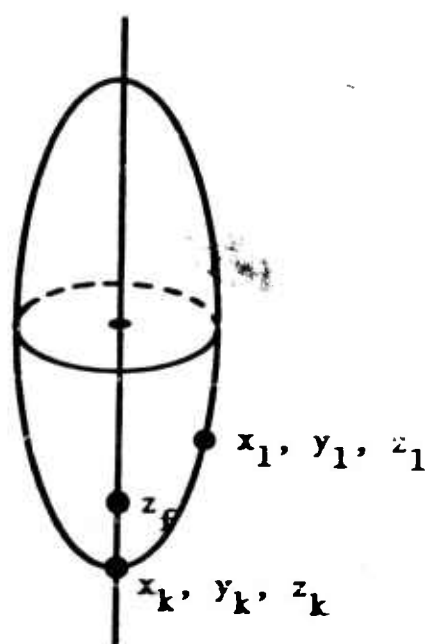


Figure 2.6 Ellipsoid of Revolution with axis parallel to Z-axis

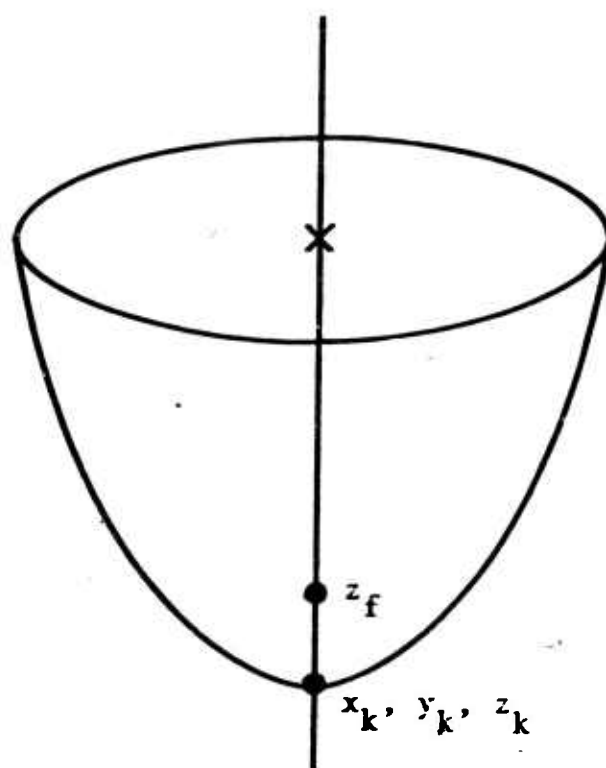


Figure 2.7 Paraboloid of revolution with axis parallel to Z-axis

Let

$$f = z_f - z_k$$

A restriction on the choice of the point (x_1, y_1, z_1) is:

$$2(z_1 - z_k) f - (z_1 - z_k)^2 \leq (x_1 - x_k)^2 + (y_1 - y_k)^2 \leq 4f(z_1 - z_k)$$

The parameters a^2 and b are calculated as follows:

$$w^2 = (x_1 - x_k)^2 + (y_1 - y_k)^2$$

$$z^* = z_1 - z_k$$

$$SQ = \text{sign}(f) * \sqrt{(z^* - f)^2 + w^2}$$

$$b = (z^* + f + SQ) / \left(4 - \frac{w^2}{z^* f}\right)$$

$$a^2 = b^2 - (b - f)^2$$

2.6.6.1 Intersection of a Ray with the Surface

$$\Delta x = x_i - x_k$$

$$\Delta y = y_i - y_k$$

$$\Delta z = z_i - z_k$$

$$C = 1 - \frac{\Delta x^2 + \Delta y^2}{a^2} - \frac{(\Delta z - b)^2}{b^2}$$

$$B = - \left[\frac{x_i \Delta x + y_i \Delta y}{a^2} + \frac{y_i (\Delta z - b)}{b^2} \right]$$

$$A = - \left[\frac{\alpha_i^2 + z_i^2}{a^2} + \frac{y_i^2}{b^2} \right]$$

if $B^2 - AC < 0$, there is no intersection.

Calculate

$$L_{\pm} = \frac{-B \pm \sqrt{B^2 - AC}}{A}$$

The intersection of interest is the minimum positive value of $L_{\pm}$.

2.6.6.2 In or Out of an Ellipsoid of Revolution

To determine if point (x_i, y_i, z_i) is inside of the surface calculate:

$$F(\vec{P}) = 1 - \frac{(x_i - x_k)^2}{a^2} - \frac{(y_i - y_k)^2}{b^2} - \frac{(z_i - z_k - b)^2}{b^2}$$

if $F(P) > 0$, the point is inside of the ellipsoid.

2.6.6.3 Normal to the Surface at a Point (x, y, z)

$$R = \sqrt{\frac{(x - x_k)^2}{a^4} + \frac{(y - y_k)^2}{b^4} + \frac{(z - z_k - b)^2}{b^4}}$$

The directional cosines of the normal are:

$$\alpha_n = \frac{x - x_k}{a^2 R}$$

$$\beta_n = \frac{y - y_k}{a^2 R}$$

$$\gamma_n = \frac{z - z_k - b}{b^2 R}$$

2.6.7 Equations for a Paraboloid of Revolution

A paraboloid of revolution is limited to the case where the axis of revolution is parallel to the Z axis. It is defined by providing the coordinates of the vertex, and the Z coordinate of the focus. If the focus is above the vertex the figure is a cup. If the focus is below the vertex the figure is a cap. See Figure 2.7.

Define:

$$\left. \begin{matrix} x_k \\ y_k \\ z_k \end{matrix} \right\} = \text{the coordinates of the vertex}$$

$$z_f = \text{the } z \text{ coordinate of the focus}$$

$$f = z_f - z_k$$

2.6.7.1 Intersection of a Ray with a Paraboloid of Revolution

Let:

$$\left. \begin{matrix} \alpha_i \\ \beta_i \\ \gamma_i \end{matrix} \right\} = \text{directional cosines of the ray}$$

$$\left. \begin{array}{l} x_i \\ y_i \\ z_i \end{array} \right\} = \text{starting point of the ray}$$

Then

$$C = 4f(z_i - z_k) - (x_i - x_k)^2 - (y_i - y_k)^2$$

$$B = 2fy_i - \alpha_i(x_i - x_k) - \beta_i(y_i - y_k)$$

$$A = -(\alpha_i^2 + \beta_i^2)$$

if $B^2 < AC$ there is no intersection

$$L_{\pm} = \frac{-B \pm \sqrt{B^2 - AC}}{A}$$

Choose the minimum positive value of $L_{\pm}$

2.6.7.2 In or Out of a Paraboloid of Revolution

$$F(\vec{P}) = 4f(z_i - z_k) - (x_i - x_k)^2 - (y_i - y_k)^2$$

if $F(\vec{P}) > 0$, the point (x_i, y_i, z_i) lies inside.

2.6.7.3 Normal to a Point on the Surface

Let

$$\left. \begin{array}{l} x \\ y \\ z \end{array} \right\} = \text{a point on the surface}$$

$$\left. \begin{array}{l} \alpha_n \\ \beta_n \\ \gamma_n \end{array} \right\} = \text{the directional cosines of the normal to the surface at the point } (x, y, z)$$

Calculate

$$R = \sqrt{(x-x_k)^2 + (y-y_k)^2 + 4f^2}$$

$$\alpha_n = \frac{x-x_k}{R}$$

$$\beta_n = \frac{y-y_k}{R}$$

$$\gamma_n = \frac{-2f}{R}$$

2.6.8 Equations for a Hyperboloid of Revolution

A hyperboloid of revolution whose axis is parallel to the Z axis is defined by the coordinates of the vertex of the sheet of interest (x_k, y_k, z_k) , the Z coordinate of the focus of the sheet of interest (z_f) and the coordinates of a point on the surface of either sheet (x_1, y_1, z_1) . See Figure 2.8.

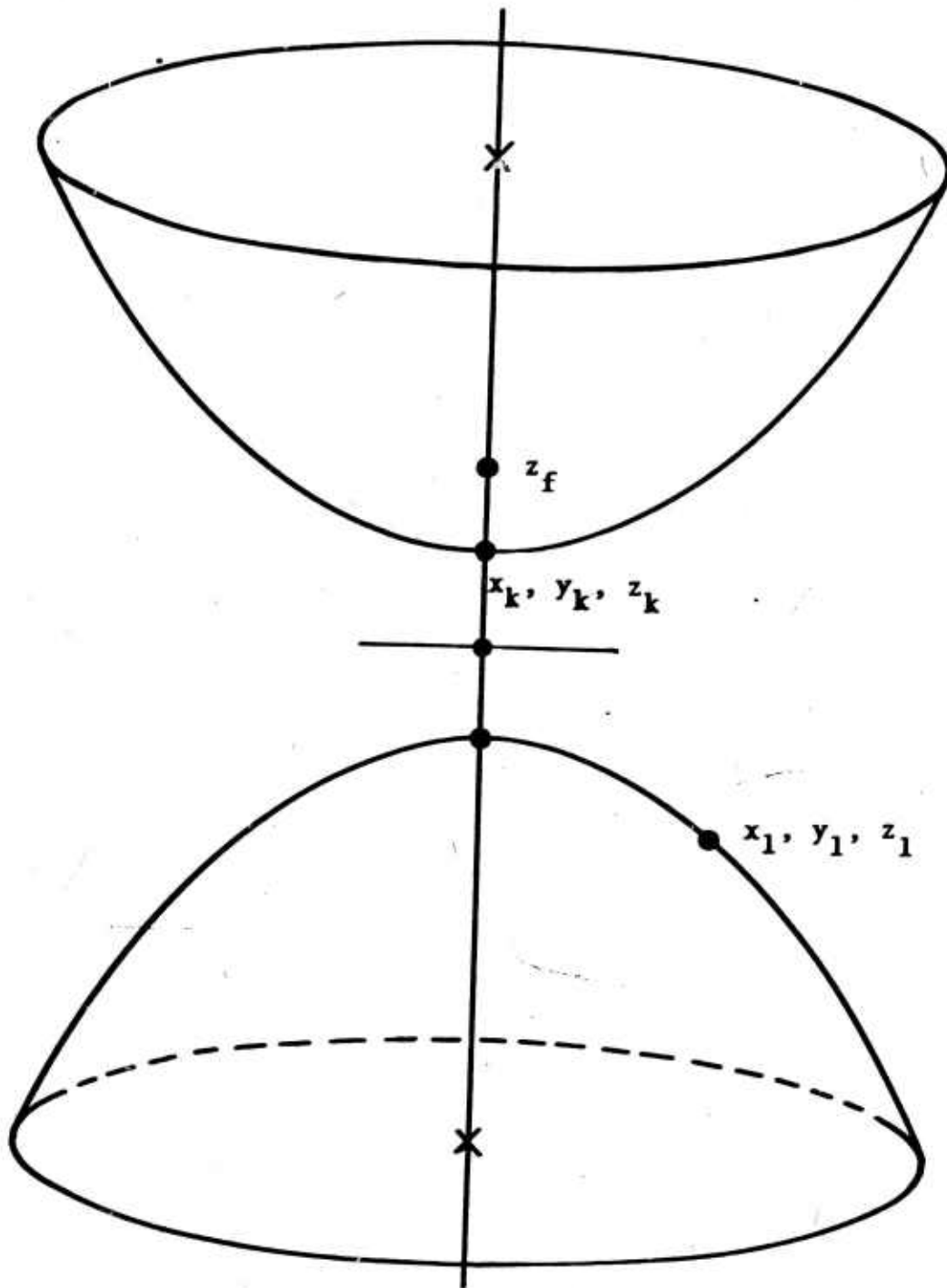


Figure 2.8 Hyperboloid of Revolution

Let

$$f = z_f - z_k$$

A restriction on the choice of the point (x_1, y_1, z_1) is:

$$4f(z_1 - z_k) < (x_1 - x_k)^2 + (y_1 - y_k)^2$$

The parameters a^2 and b are calculated as follows:

$$W^2 = (x_1 - x_k)^2 + (y_1 - y_k)^2$$

$$z^* = z_1 - z_k$$

$$SQ = \text{Sign}(z^*) * \sqrt{(z^* - f)^2 + W^2}$$

$$b = (z^* + f + SQ) / \left(\frac{W^2}{z^* f} - 4 \right)$$

$$a^2 = (b + f)^2 - b^2$$

2.6.8.1 Intersection of a Ray with the Surface

$$\Delta x = x_i - x_k$$

$$\Delta y = y_i - y_k$$

$$\Delta z = z_i - z_k$$

$$C = -1 - \frac{\Delta x^2 + \Delta y^2}{a^2} + \frac{(\Delta z + b)^2}{b^2}$$

$$B = \frac{\gamma(\Delta z + b)}{b^2} - \frac{\alpha \Delta x + \beta \Delta y}{a^2}$$

$$A = \frac{\gamma^2}{b^2} - \frac{\alpha^2 + \beta^2}{a^2}$$

if $B^2 - AC < 0$ there is no intersection.

Calculate

$$L_{\pm} = \frac{-B \pm \sqrt{B^2 - AC}}{A}$$

The intersection of interest is the minimum positive value of $L_{\pm}$ that will give an intersection on the desired sheet.

2.6.8.2 In or Out of a Hyperboloid of Revolution

To determine if point (x_i, y_i, z_i) is inside of the surface, calculate:

$$F(\vec{P}) = -1 - \frac{\Delta x^2 + \Delta y^2}{a^2} + \frac{(\Delta z + b)^2}{b^2}$$

If $F(\vec{P}) > 0$ and $\Delta z + b > 0$, the point is inside.

2.6.8.3 Normal to a Point on the Surface

Let

$$\left. \begin{array}{l} x \\ y \\ z \end{array} \right\} = \text{a point on the surface}$$

$$\left. \begin{array}{l} \alpha_n \\ \beta_n \\ \gamma_n \end{array} \right\} = \text{the directional cosines of the normal to the surface at the point } (x,y,z)$$

Calculate

$$R = \sqrt{\frac{\Delta x^2 + \Delta y^2}{a^4} + \frac{(\Delta z + b)^2}{b^4}}$$

$$\alpha_n = \frac{\Delta x}{a^2 R}$$

$$\beta_n = \frac{\Delta y}{a^2 R}$$

$$\gamma_n = -\frac{\Delta z + b}{b^2 R}$$

2.6.9 Equations for a Helix

A helix whose main axis is parallel to the Z axis is defined by

$$\left. \begin{array}{l} x_h \\ y_h \end{array} \right\} = \text{coordinates of a point on the axis of the helix}$$

$$z_h = \text{Point on the helical wire in the positive } x \text{ direction from the axis of the helix}$$

$$\kappa = \text{Parameter defining the distance between successive coils (D)} \\ D = |2\pi\kappa| \text{ if } \kappa < 0, \text{ coil is left-handed.}$$

ρ = radius from axis to helical wire

a = radius of the helical tube.

The equations used are quite voluminous. They are derived and presented in Appendix A.1.

3. INPUT - OUTPUT

3.1 SETUP CONSIDERATIONS

The use of any large generalized computer program requires a certain amount of knowledge of how the program works and requires a learning process the same as with a piece of complicated machinery. The more general a program is, the more understanding and experience is required to effectively make use of it. The ZAP code is no exception in this respect.

In particular, a thorough understanding of the mother-daughter relationship and its consequences is very important, since this is the basis of the ZAP technique. To try to set up complicated geometries without this understanding will invariably result in boundaries existing in areas where they are not supposed to exist. To cut off the extension of a plane for example, requires that the area where the plane actually exists be enclosed in a Mother segment. Then the plane will not extend outside that Mother segment.

Unfortunately, with Monte Carlo techniques gross errors can occur in the setup of a problem without causing any indication that these errors exist. Boundaries may exist in areas where they are not intended without ever being obvious in the results. Therefore, extreme care and thought should be put into the setup of a problem. To help spot a bad setup, test data points and geometry plots have been included in the program.

The bad effects of boundaries extending into areas where they do not belong has perhaps been over emphasized in the above paragraphs. Regardless of the type of boundary under consideration, it is treated as a dummy boundary and has no ill effects if the same material exists on each side of the boundary. Conversely, it must be kept in mind that

if the same material or two different materials with the same index of refraction are separated by a boundary, that boundary will be treated as a dummy regardless of how it is defined.

It is highly recommended that anyone planning extensive use of the ZAP program should become as familiar as is possible with the inner workings of the code.

3.1.1 Miscellaneous Rules

1. Planes must not pass through the origin (0,0,0).
2. Each Mother segment must also exist as a daughter segment (except for the overall Mother segment).
3. The overall Mother segment must be segment #1.
4. Each Mother segment must have at least one daughter.
5. A Mother segment must completely contain each of its daughters. Conversely a daughter must be completely contained by its Mother.
6. Different materials that are to be separated by a real boundary must not have the same refractive index.
7. Planes defined for geometry plots must not coincide with a real plane in the geometric system.
8. A segment must be "totally" confined by its confining boundaries. For example, an open ended cylinder does not qualify as a segment.

9. Any rectangular area to be plotted must lie completely within the overall Mother segment.

3.1.2 User Routines

Provisions have been made to allow the user to provide subroutines to handle certain functions within the program. A description of these routines is given below as they apply to the functions desired.

3.1.2.1 Wavelength Dependent Absorption Coefficients

In a multifrequency problem the user must provide a subroutine or subroutines that calculate and store the absorption coefficients as a function of wavelength and material number. The name of the subroutine for each material is given on the material data card and must be of the form:

ABSX

where X is a number from 1 to 9.

The argument list for the subroutine is:

SUBROUTINE ABSX(XLAM,FMAX,TSTG,M)

where

XLAM = name of the array that contains frequencies.

FMAX = the number of frequencies (integer).

TSTG = storage area for absorption coefficients from F=1 to F=FMAX.

M = material number.

This subroutine must determine and store the absorption coefficients for material M into the array TSTG(F), F=1,FMAX.

One subroutine may be provided for each material or the same subroutine may be used for different materials.

3.1.2.2 Wavelength Dependent Intensities

In a multifrequency run, the ray intensity may vary with wavelength. Therefore a subroutine to provide intensity as a function of wavelength must be provided. The name of the subroutine to be used for each lamp type is given on the lamp data card and will be of the form:

INTENX

where X is an integer from 1 to 9.

The argument list is:

SUBROUTINE INTENX(XLAM,FMAX,TSTG,LAMNO)

where:

XLAM = name of the frequency array

FMAX = the number of frequencies (integer)

TSTG = an array to store intensity as a function of wavelength for lamp number LAMNO

LAMNO = lamp number.

This subroutine must determine and store ray intensities for lamp number LAMNO into the array:

TSTG(F), F=1, FMAX

One subroutine may be provided for each lamp number, or the same routine may be used for different lamp numbers.

3.1.2.3 Surface Source Distribution

When a surface source is used, a subroutine must be provided that will give the angles used to calculate the directional cosines for a given ray. The name of this routine is given on the lamp data card and must be of the form:

. ANGLEX

where X is a number from 1 to 9.

The argument list for the subroutine is:

SUBROUTINE ANGLEX(TH, PHI)

where

TH = an angle (θ) with the normal to the surface.

PHI = an angle (ϕ) in the plane perpendicular to the normal.

This routine must provide values for θ and ϕ where:

$$0 \leq \theta \leq \pi/2$$

$$0 \leq \phi \leq 2\pi$$

The routine is called each time a new ray is created and should provide values of θ and ϕ that correspond to some desired probability distribution.

An example would be to provide a Lambertian distribution. In this case the subroutine would generate a random ϕ and a random $\cos^2 \theta$. From $\cos^2 \theta$, θ would then be found.

3.1.2.4 Wavelength Dependent Fraction of Reflection

When the fraction of a ray's intensity to be reflected is dependent on the wavelength, a subroutine must be provided to give the fraction for each wavelength. The name of the subroutine must be of the form:

REFLXX

where XX is the reflection number of the boundary.

There are two separate cases that may be considered.

1. Fraction dependent on wavelength only
2. Fraction dependent on wavelength and angle of incidence.

In case 1 the argument list is:

SUBROUTINE REFLXX(XLAM,FMAX,TSTG,DUM)

where

XLAM = name of wavelength array
 FMAX = number of wavelengths (integer)
 TSTG = array for storing fractions of reflection
 DUM = dummy argument.

In this case the routine is called at the beginning of the problem and the information is stored for later use. The subroutine must determine and store the fractions of reflection into the array:

TSTG(F),F=1,FMAX.

In case 2 the argument list is:

SUBROUTINE REFLXX(XLAM,FMAX,TSTG,COSGAM)

where

XLAM = name of wavelength array
 FMAX = number of wavelengths (integer)
 TSTG = array for storing fractions
 COSGAM = the cos of the angle of incidence.

In this case the subroutine is called when an intersection occurs and must calculate the fraction to be reflected for each wavelength for the particular angle of incidence indicated. These fractions are then stored into the array

TSTG(F), F=1,FMAX

3.2 INPUT

Input to the ZAP code is in the form of data cards. Special keypunch forms have been designed for each of the various types of input cards, with the exception of the general input which uses a standard FORTRAN data sheet.

The entries use a combination of integer, real and alphanumeric formats. In the case of real and integer input, either a blank or zero may be entered if the number is zero.

All directional cosines entered are checked by the code and the sum of the squares of the three directional cosines must be within the range of 1.0 ± 10^{-3} . If they are not, the program will terminate execution with a debug print.

Care must be taken that no plane passes through the origin. If this happens, the program will terminate execution.

Wherever possible default conditions have been provided so that if the number is blank or zero, the default number is used.

3.2.1 General Input (three cards)

Card #1 - 72 character comment or title card

Card #2

Entry Number	Description	Format	Columns
1)	FRAC = Base fraction. When a ray intensity drops below this fraction of its initial intensity, Russian Roulette is played. Default number = .5.	E12.4	1-12
2)	DZ = Distance along Z axis between planes of continuity. If no planes of continuity exist, this number is ignored.	E12.4	13-24
3)	WAVMIN = minimum wavelength. (May be omitted for single wavelength.)	E12.4	25-36
4)	WAVMAX = maximum wavelength. If there is only one wavelength, this number is ignored.	E12.4	37-48

Card #3

Entry Number	Description	Format	Columns
1)	FMAX = Number of wavelengths	I6	1-6
2)	IOPRNT = Option for printing each ray created. 0 = no 1 = yes	I6	7-12
3)	IDEBUG = Option for debug prints. 0 = no 1 = yes	I6	13-18
4)	IR = starting random integer. Default = 1.	I16	19-34

No blank card following general input cards.

3.2.2 Reflection Data

For each special reflection a reflection data card must be provided. The reflection number must be greater than ten, since reflection numbers 1-10 are reserved for built-in reflection types.

Entry Number	Description	Format	Columns
1)	Reflection number (greater than ten).	Integer	2-4
2)	Reflection type (left justified). CONST. = Constant reflection (fraction reflected will be input). FUNCT. = Function of the form: fraction reflected = $1.0 - A(\cos \theta)^2$ where A is input and $\cos \theta$ is the angle made by the impinging ray with the normal to the surface. TABLE. = An input table of ten fractions which indicate the amount reflected for different angles with the normal. Angles are from 0° to 90° in 10° intervals. DIFFUS = Diffuse reflection. Fraction to be reflected is input. WAVCAL = Reflection dependent on wavelength. The fraction of each wavelength to be reflected is calculated as a function of the wavelength and the angle of incidence. A subroutine REFLXX must be provided where XX is the reflection number. Example, reflection number 12 means that the subroutine is REFL12. (See section on User Subroutines for details.) WAVINP = Reflection dependent on wavelength only. A subroutine named REFLXX must be provided where XX is the reflection number. This subroutine is called immediately at the time this card is read, and fills an array with wavelength dependent fractions of reflection to be used later. (See section on User Subroutines.)	Alpha	6-11

3)	For CONST load fraction to be reflected. For FUNCT load value for A. For DIFFUS load fraction to be reflected. For TABLE load fraction for 0°. For WAVCAL or WAVINP this entry is ignored.	Real	13-17
4-12	For TABLE load fractions to be reflected for angles 10°-90°. Blanks for other reflection types.	Real	19-71

A blank card must follow the last reflection data card. If no reflection data was loaded, the blank is still required.

3.2.3 Material Data

For each material a material data card must be provided.

Entry Number	Description	Format	Columns
1)	Material number	Integer	2-4
2)	Name of routine to generate absorption coefficients as a function of wavelength. (See section on User Routines). If this is a single wavelength run, this entry is ignored.	Alpha	6-11
3)	Absorption coefficient (length^{-1}) If an input routine was specified above, this number is ignored.	Real	13-18
4)	Refractive index (must be non-zero)	Real	20-25
5)	Lasing wavelength (for multiple wavelength runs only). This is an optional input value.	Real	27-32
6)	39 character description to be printed	Alpha	34-72

A blank card must follow the last material data card.

3.2.4 Boundary Data

For each boundary a boundary data card must be provided. The form of the data on the card depends on the geometry type.

Entry Number	Description	Format	Columns
1)	Boundary number	Integer	2-4
2)	Reflection number 1 = total transmission, no refraction 2 = total reflection. May or may not represent boundary of symmetry. 3 = Fresnel's reflection for dielectrics. 4 = Top boundary of continuity. 5 = Bottom boundary of continuity. 6-10 = Reserved for further built-in reflection types. 11-50 = Special reflection types defined on reflection data sheet.	Integer	6-8
3)	Geometry type (left justified in columns provided on data sheet). CYLIND = cylinder PLANE^ = plane CONIC^ = conic SPHERE = sphere CONE^^ = half-cone with axis parallel to Z axis PARAB^ = paraboloid of revolution with axis parallel to Z axis. HYPERB = hyperboloid of revolution with axis parallel to the Z axis. ELLIPS^ = ellipsoid of revolution whose axis is parallel to the Z axis. HELIX = helical tube with axis parallel to the Z axis	Alpha	10-15

Boundary Data (continued)

Entry Number	Description	Format	Columns
4-11)	Parameters for the particular geometry type being described:	Real	
	PLANE:		
	Directional cosines for the normal to the plane. Normal must point from origin toward plane.		17-22
	4) α		24-29
	5) β		31-36
	6) γ		38-43
	7) X		45-50
	8) Y		52-57
	9) Z		
	10-11) -- Not used		
	CYLIND:		
	Directional cosines for cylindrical axis.		17-22
	4) α		24-29
	5) β		31-36
	6) γ		38-43
	7) X		45-50
	8) Y		52-57
	9) Z		59-64
	10) R = Radius of cylinder		
	11) -- Not used		
	CONIC:		
	Directional cosines for axis.		17-22
	4) α		24-29
	5) β		31-36
	6) X_0		38-43
	7) Y_0		45-50
	8) X_1		52-57
	9) Y_1		59-64
	10) X_2		66-71
	11) Y_2		
	SPHERE:		
	Coordinates of center of sphere		17-22
	4) X_c		24-29
	5) Y_c		31-36
	6) Z_c		33-43
	7) R = Radius of sphere		
	9-11) Not used		

Boundary Data (continued)

Entry Number	Description	Format	Columns
	CONE: 4) X_v 5) Y_v 6) Z_v } Coordinates of the vertex 7) X 8) Y 9) Z } Coordinates of a point on the surface of the half-cone to be considered. 10-11) -- Not used.		17-22 24-29 31-36 38-43 45-50 52-57
	PARAB: 4) X_v 5) Y_v 6) Z_v } Coordinates of the vertex 7) $Z_f = Z$ coordinate of the focus		17-22 24-29 31-36
	HYPERB: 4) X_v 5) Y_v 6) Z_v } Coordinates of vertex for sheet to be considered. 7) X 8) Y 9) Z } Coordinates of a point on the surface. This point may be on either sheet. 10) $Z_f = Z$ coordinate of focus for sheet to be considered. 11) -- Not used.		17-22 24-29 31-36 38-43 45-50 52-57 59-64
	ELLIPS: 4) X_v 5) Y_v 6) Z_v } Coordinates of vertex 7) X 8) Y 9) Z } Coordinates of a point on the surface of the ellipsoid.		17-22 24-29 31-36 38-43 45-50 52-57

Boundary Data (continued)

Entry Number	Description	Format	Columns
	ELLIPS (continued)		
10)	Z_f = Z-coordinate of the focus nearest the chosen vertex.		59-64
11)	-- Not used.		
	Restriction:		
	$A \leq B \leq C$		
	where		
	$A = 2(Z-Z_v)(Z_f-Z_v) - (Z-Z_v)^2$		
	$B = (X-X_v)^2 + (Y-Y_v)^2$		
	$C = 4(Z_f-Z_v)(Z-Z_v)$		
	HELIX		
4)	X_h Coordinates of a point on the		17-22
5)	Y_h helical axis. The nearest		24-29
6)	Z_h point on the helical wire is assumed to be in the positive X direction from this point.		31-36
7)	κ = Parameter associated with distance between consecutive coils. $D = 2\pi\kappa$. If $\kappa < 0$, coil is lefthanded.		38-43
8)	ρ = radius from axis to helical wire.		45-50
9)	a = radius of tube.		52-57
10=11)	Not used.		

3.2.5 Segment Data

For each segment, a segment data card must be provided.

Entry Number	Description	Format	Columns
1)	Segment number	Integer	2-4
2)	Material number	Integer	6-8
3-13)	Boundary numbers indicating the confining boundaries of the segment. A positive boundary number indicates the origin side of a plane or inside of other geometries. A negative boundary indicates the non-origin side of a plane, or the outside of other geometries.	Integer	10-63
14)	Segment volume (if known).	Real	65-71

A blank card must follow the last segment data card.

3.2.6 Structure Data

For each mother-daughter relationship, a structure data card must be provided.

Entry Number	Description	Format	Columns
1)	Mother segment	Integer	2-4
2-18)	Daughter segments	Integer	6-68

A blank card must follow the last structure data card.

3.2.7 Lamp Data

For each lamp type a lamp data card must be loaded. This data describes primarily the method to be used for creating rays. Many different sources may refer to the same lamp type.

Entry Number	Description	Format	Columns
1)	Lamp number	Integer	2-4
2)	Lamp type (left justified) VOLUME = volume source OUTSUR = cylindrical surface source with rays directed outward from the surface. INNSUR = cylindrical surface source with rays directed inward from the surface. RAYINP = directional cosines and starting coordinates are input for each ray.	Alpha	6-11
3)	Name of routine to be used for generating intensity as a function of wavelength. This routine will have a name INTENX where X is a number from 1 to 9 (see user routines) and must be provided by the user. If this is a single wavelength run, this entry is ignored.	Alpha	13-18
4)	Name of routine to be used for calculating the angle with the normal for rays created if this is a surface source. If not a surface source this entry is ignored. See user routines for description of this routine.	Alpha	20-25
5)	Intensity of rays if this is a single wavelength run. If multiple wavelength, this entry is ignored.	Real	27-37

A blank card must follow the last Lamp Data card.

3.2.8 Source Data

A source data card must be provided for each source in the problem.

Entry Number	Description	Format	Columns
1)	Source number	Integer	2-4
2)	Segment number or boundary number of source. If the lamp type designated is for a volume source, load a segment number. If the lamp type designated is for a surface source, load a boundary number.	Integer	6-8
3)	Lamp number - refers to a lamp data card.	Integer	10-12
4)	Number of rays for this source.	Integer	14-17
5-10)	Parameters which depend on the particular lamp type designated. VOLUME: 5) XMIN 6) XMAX Minimum and maximum 7) YMIN Values of X,Y and Z 8) YMAX which describe a 9) ZMIN rectangle within which 10) ZMAX the source segment is contained. INNSUR or OUTSUR: 5-8) Blank 9) ZMIN Minimum and maximum values 10) ZMAX of Z for creating rays. RAYINP: Entries 5-10) are ignored.	Real	

For RAYINP, this card is immediately followed by Ray data cards for the number of rays specified above.

A blank card must follow the last Source Data card.

3.2.9 Ray Data

If rays are to be input individually, one data card must be input for each ray to be created. Use source data sheets.

Entry Number	Description	Format	Columns
1-4)	Blank		
5)	α	Real	19-25
6)	β		27-33
7)	γ		35-41
8)	X	Real	43-49
9)	Y		51-57
10)	Z		59-65

3.2.10 Test Data

Test data cards are used to test the geometrical setup for errors. This consists of referencing a segment number and indicating that a specified point lies either inside or outside of the segment. The point is then tested to see if it does lie on the proper side of the segment. As many sets of test data as desired may be loaded.

Entry Number	Description	Format	Columns
1)	JTEST = Segment number to be tested. if JTEST>0, point should lie inside segment. If JTEST<0, point should lie outside of segment.	Integer	1-6
2)	IOP = Option for testing to see if point lies within the confining boundaries of the segment, or to see if the point lies within this segment rather than some other segment. IOP = 0, Test segment boundaries only IOP = 1, Determine which segment contains the point and determine if it is the desired segment.	Integer	7-12
3)	X } Y } Z } = Coordinates of the test point	Real	13-24
4)		Real	25-36
5)		Real	37-48

3.2.11 Plot Data

Each plot data card indicates a rectangular area within a plane that slices through the system. This plane may be at any desired orientation. The geometric boundaries sliced by the plane within the rectangular area are then plotted. The only restriction is that the rectangular area must be completely contained within the overall mother region (Region #1). This is no real restriction since the overall mother region may be made as large as desired without affecting the speed or accuracy of the program.

Entry Number	Description	Format	Columns
1)	X_0 } Y_0 } Z_0 } Coordinates of the point to be used for the lower left-hand corner of the plotting area.	Real	1-8
2)		Real	9-16
3)		Real	17-24
4)	X_1 } Y_1 } Z_1 } Coordinates of the point to be used for the lower right-hand corner of the plotting area.	Real	25-32
5)		Real	33-40
6)		Real	41-48
7)	X_2' } Y_2' } Z_2' } Coordinates of a point that lies somewhere along the top line of the plotting area.	Real	49-56
8)		Real	57-64
9)		Real	65-72

A blank card must follow the last of the plot data cards.

3.3 OUTPUT

Output from the ZAP code consists of printer plots of various cuts through the geometry and printed summaries of the energy deposition for each segment and for each material.

Under the heading SEGMENT RESULTS the following quantities are printed for each segment:

1. Mother segment number = mother of the segment being considered.
2. Segment number.
3. Material number in segment.
4. Total Energy = Sum of energy deposited in segment.
5. Scaled Energy = Sum of energy scaled by lasing wavelength.
6. Energy per unit volume.
7. ~~Percent total energy~~ = The percent of the total energy in the system that is in this segment.
8. Percent error = probable statistical percent error. (See Section 2.5.2)
9. Number of legs = actual number of ray legs traced through segment.
10. Number of rays = number of different rays traced through segment, even though each ray may have had several legs passing through the segment.

Under the heading MATERIAL results the following quantities are printed:

1. Total absorbed energy = sum of segment energies...
2. Total source energy - sum of initial ray intensities.
3. Total number of rays.
4. Final random integer = Random integer that could be used to continue the run without changing the random number sequence.

For each material the following quantities are printed:

1. Material number
2. Material volume = sum of volumes of segments containing material.
3. Total energy = sum of energies of segments containing material.
4. Scaled energy = sum of scaled segment energies for material.
5. Energy per unit volume.
6. Percent of total energy = The percent of the total energy in the system that is in this material.
7. Percent error = Probable statistical percent error. (See Section 2.5.2)
8. Number of rays = number of different rays that passed through the material.

3.4 SAMPLE PROBLEM #1

The first sample problem consists of a cylindrical lamp and a cylindrical laser rod encased in an elliptic laser cavity (Fig. 3.1). The area of interest is chosen to be in the body of the system so that end conditions are ignored. In this case each end is considered to be a boundary of symmetry (a simple boundary of total reflection).

A single wavelength is considered and parameters such as absorption coefficients and refractive indices are not intended to be physically real but were chosen arbitrarily for demonstration purposes.

The laser system is encased in a 20 cm sphere that is considered to be the overall Mother segment (segment #1). The area inside the cavity was chosen to be segment #2 and is a mother containing two daughters. These daughters are respectively, segment #3 which is the cylindrical lamp and segment #4 which is the cylindrical laser rod. The laser rod is a mother and has daughters #5 and #6 which are simply the front and back half of the laser rod. Note that the area of segment #4 is completely filled by segments #5 and #6 so that segment #4 has no actual volume of its own.

The material outside of the cavity is chosen to be material #1, the air inside the cavity is material #2, the lamp is composed of material #3 and the laser rod consists of material #4. These material numbers appear in their proper places in the printer plots (3.2 through 3.6).

Boundary #1 is the sphere encasing the system. Boundary #2 is the elliptic cylinder that represents the cavity wall and is chosen to be a boundary of constant reflection with 90% reflection and 10% transmission. Boundaries #3 and #4 are cylinders defining the lamp and laser rod respectively

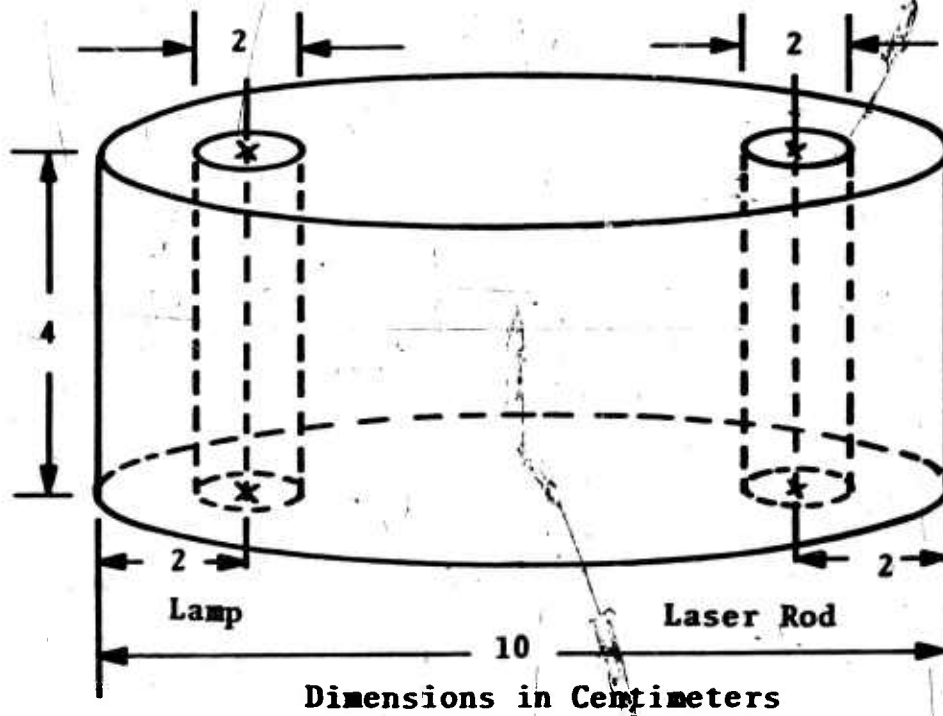


Figure 3.1 A simple laser cavity used for sample problem #1.

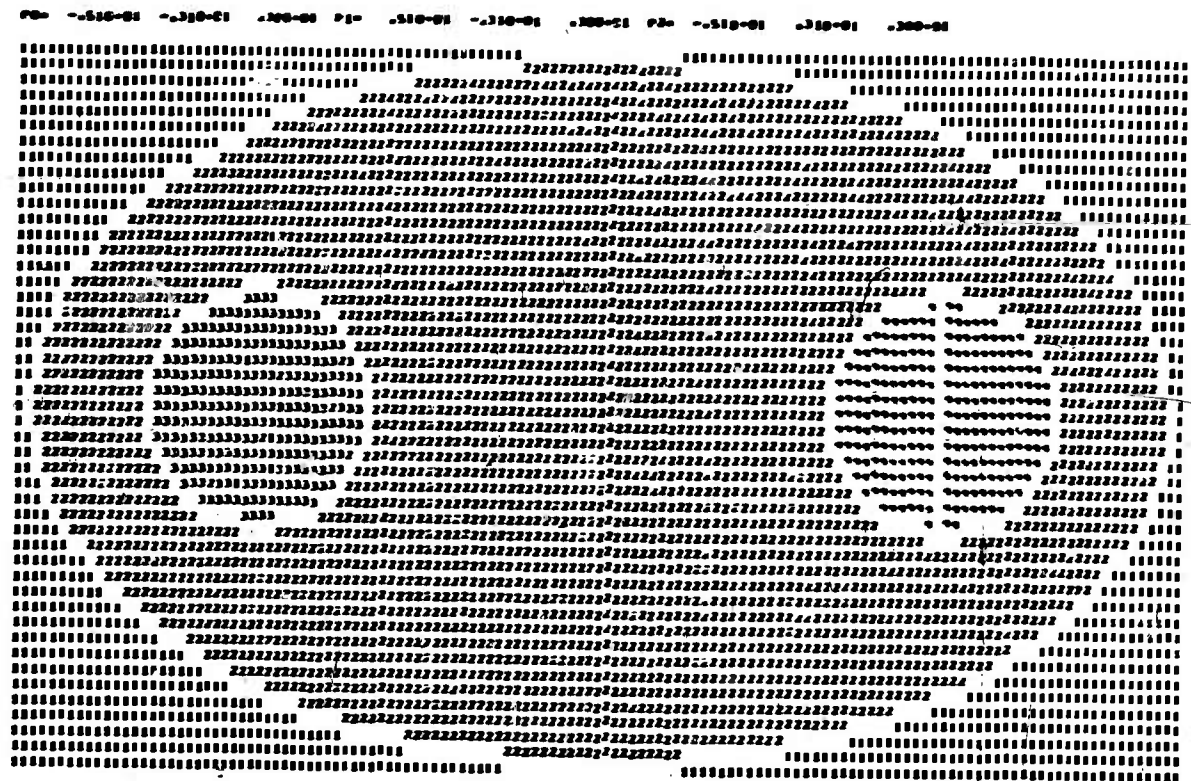


Figure 3.2 Top view of laser system used in sample problem #1.

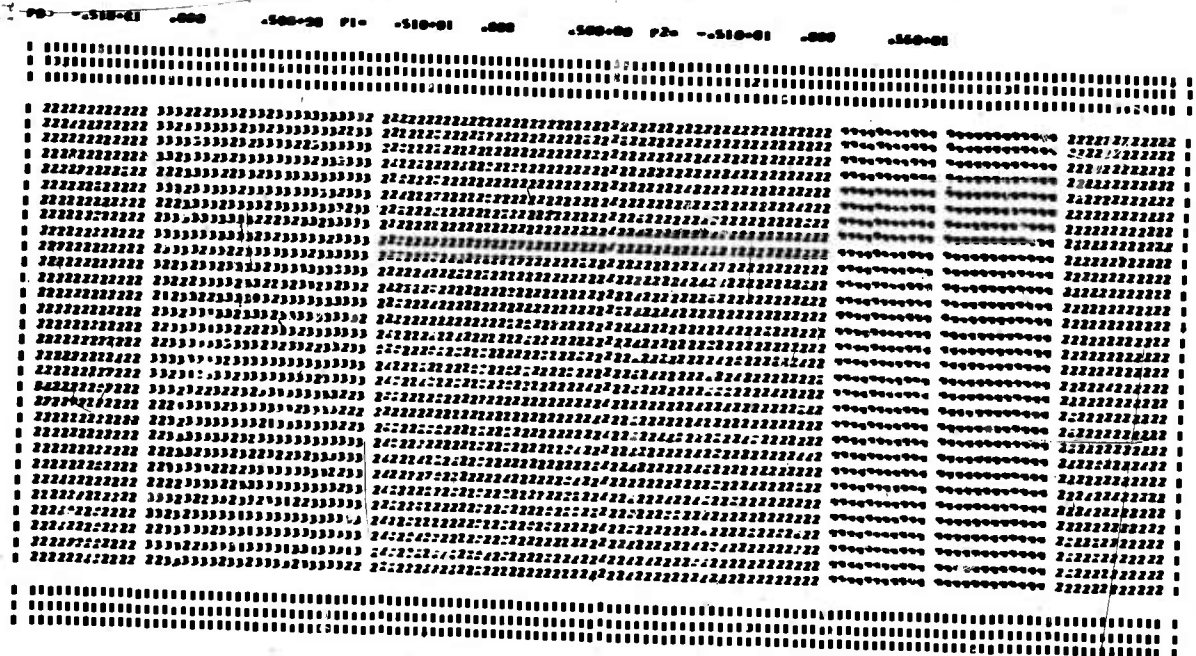


Figure 3.3 Side view of laser system used in sample problem #1.

93

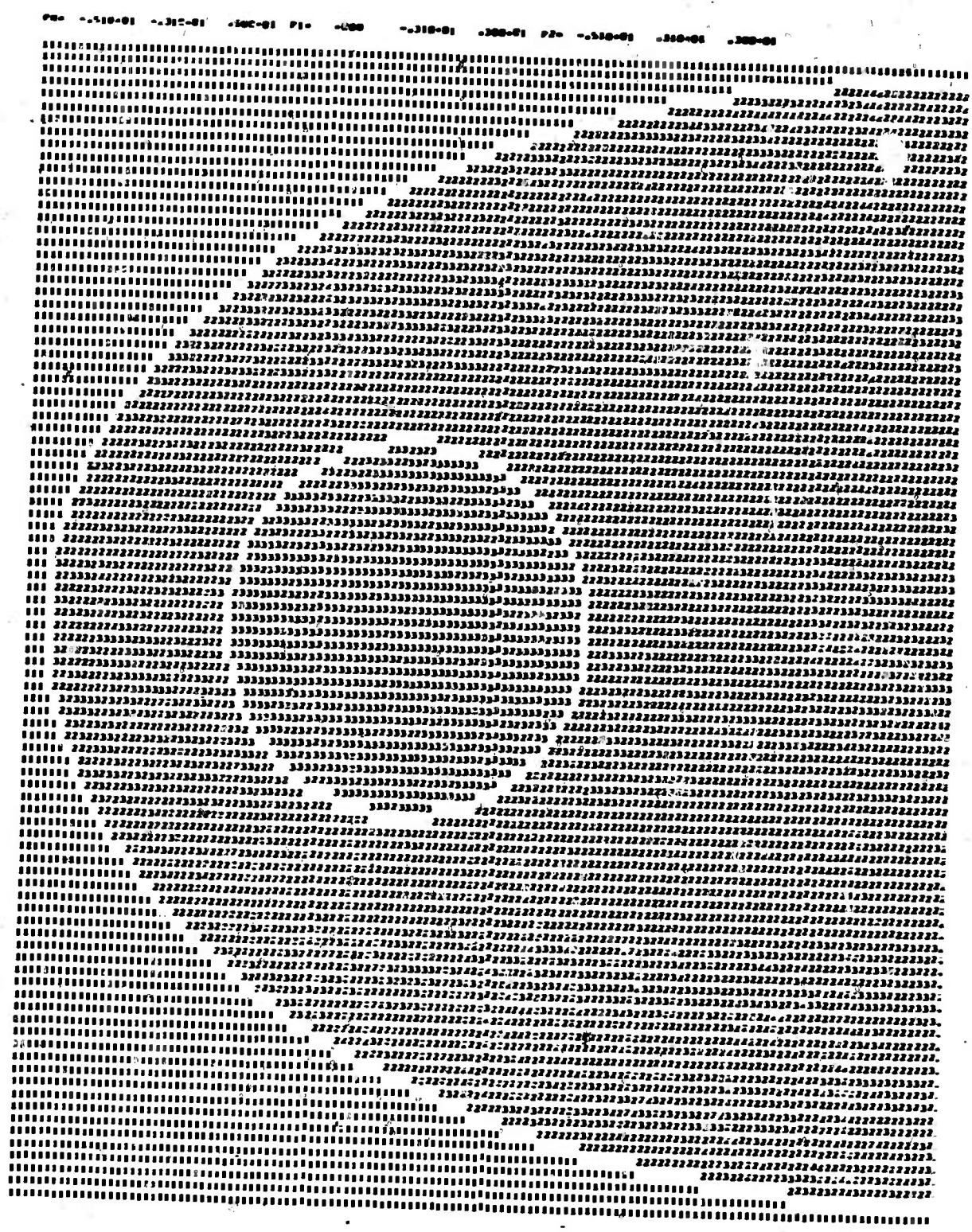


Fig. 3.5 Top view of left half of laser system used in sample problem #1.

[illegible]

Figure 3.6 Top view of lower half of laser system used in sample problem #1.

and allow Fresnel's reflection. The end planes are planes of symmetry (total reflection) and are boundaries #5 and #6. The plane that divides the laser rod into two segments is boundary #7 and is a dummy boundary.

The lamp (segment #3) is a volume source that emits rays with an intensity of 1.0. Only 100 rays were emitted for this sample run.

A listing of the input cards used and a listing of the results are given below.

INPUT DATA

SAMPLE PROBLEM NUMBER-1

.5 1 0 0 1.0 0.0

11 CONST .9

1 1000.0 2.0
2 .1 1.0METAL APPLIED SYSTEM
FLUX3 -1 1.0
4 -1 1.7LAMP
LAMP 225

1 2 SPHERE 0.0

2 11 CONTC 1.0

3 3 CYLIND 0.0

4 3 CYLIND 0.0

5 2 PLANE 0.0

6 2 PLANE 0.0

7 1 PLANE 1.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

0.0

1 1 1

2 2 2

3 3 3

4 4 4

5 4 4

6 4 4

1 2

2 3

4 5

1 1

1 3 1

1 100 -4.0

1 100 -4.0

1 100 -4.0

1 100 -4.0

1 100 -4.0

1 100 -4.0

1 100 -4.0

1 100 -4.0

1 100 -4.0

1 100 -4.0

1 100 -4.0

1 100 -4.0

1 100 -4.0

1 100 -4.0

1 100 -4.0

1 100 -4.0

1 100 -4.0

1 100 -4.0

1 100 -4.0

1 100 -4.0

1 100 -4.0

1 100 -4.0

1 100 -4.0

1 100 -4.0

1 100 -4.0

1 100 -4.0

1 100 -4.0

1 100 -4.0

1 100 -4.0

1 100 -4.0

1 100 -4.0

1 100 -4.0

1 100 -4.0

1 100 -4.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

1.0

RESULTS

GENERAL INPUT FOR ZAP

SAMPLE PROBLEM NUMBER=1
 ROUTE FRACTIONS= .000000
 DISTANCE BETWEEN CONTINUATIVE BOUNDARIES= .0000
 MINIMUM WAVELENGTH= .000001
 MAXIMUM WAVELENGTH= .0000
 NUMBER OF WAVELENGTHS= 1
 RAY PRINT OPTIONS= 0
 DEBUG PRINT OPTIONS= 0
 STARTING RANDOM INTEGER= 1

REFLECTION NUMBER	TYPE	REFLECTION DATA									
11	CONST	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000

MATERIAL NUMBER	INPUT ROUTINE	ABSORPTION COEFFICIENT	REFRACTIVE INDEX	MATERIAL DATA WAVELENGTH	DESCRIPTION
1		1.000000000	2.000000	.000000	METAL AROUND SYSTEM
2		.100000	1.000000	.000000	FLUID
3		.100000	1.500000	.000000	LAMP
4		.100000	1.700000	5.000000	LASER ROD

RESULTS

BOUNDARY DATA

BOUND NUMB	REFL NUMB	BOUND TYPE	ALPHA	BETA	GAMMA	X	Y	Z	X2	Y2	X3	Y3	O18	REC
1	2	SPHERE	1.000	.000	.000	.000	.000	.000	.000	.000	.000	.000	20.000	
2	11	CONIC	.000	.000	.000	.000	.000	.000	.000	.000	.000	.000	2.250	.400
3	3	CYLIND	.000	.000	1.000	.000	.000	.000	.000	.000	.000	.000	1.000	
4	3	CYLIND	.000	.000	1.000	.000	.000	.000	.000	.000	.000	.000	1.000	
5	2	PLANE	.000	.000	1.000	.000	.000	.000	.000	.000	.000	.000	1.000	
6	2	PLANE	.000	.000	1.000	.000	.000	.000	.000	.000	.000	.000	8.000	
7	1	PLANE	1.000	.000	.000	.000	.000	.000	.000	.000	.000	.000	3.000	

SEGMENT DATA

SEGMENT NUMBER	MATERIAL NUMBER	CONFINING BOUNDARIES	VOLUME
1	1	1	.000000
2	2	2	.000000
3	3	3	.000000
4	4	4	.000000
5	5	5	.000000
6	6	6	.000000

STRUCTURE DATA

DAUGHTER SEGMENTS

MOTHER SEGMENTS

1	2	4	6

RESULTS

LAMP DATA

LAMP NUMBER	LAMP TYPE	LAMP VOLUME	INTENSITY ROUTINE	ANGLE ROUTINE	INTENSITY
1					.1000+01

SOURCE DATA

SOURCE NUMBER	SEGMENT NUMBER	BOUNDARY NUMBER	LAMP NUMBER	NUMBER OF RAYS	GEOMETRIC SOURCE DATA
1	3	0	1	100	-4.00000 -2.00000

TEST DATA

SEGMENT	OPTION	X	Y	Z
1	1	-.1000+01	.0000	.0000+01
2	1	.0000	.0000	.3000+01
3	1	-.3000+01	.0000	.3000+01
4	1	.3000+01	.0000	.3000+01
5	0	.0000	.0000	.3000+01

RESULTS

SEGMENT RESULTS

MOTHER SEGMENT NUMBER	SEGMENT NUMBER	MATERIAL NUMBER	TOTAL ENERGY	SCALED ENERGY	ENERGY PER UNIT VOLUME	PERCENT TOTAL ENERGY	PERCENT ERROR	NUMBER OF LEAD	NUMBER OF RAYS
1	2	2	.3048+02	.0000	.0000	31.890	20.893	191	39
2	3	3	.6043+02	.0000	.0000	42.408	9.138	874	100
4	4	4	.0000	.0000	.0000	.000	.00000	0	0
	5	5	.1679+01	.3389+00	.0000	1.731	30.870	22	10
	6	6	.1487+01	.2918+00	.0000	1.002	35.828	20	12

MATERIAL RESULTS

TOTAL ABSORBED ENERGY = 9.7029+01
 TOTAL SOURCE ENERGY = 1.0000+02
 TOTAL NUMBER OF RAYS = 100
 FINAL RANDOM INTEGER = 8328333409

MATERIAL NUMBER	MATERIAL VOLUME	TOTAL ENERGY	SCALED ENERGY	ENERGY PER UNIT VOLUME	PERCENT OF TOTAL ENERGY	PERCENT ERROR	NUMBER OF RAYS
1	.0000	.2610+01	.0000	.0000	2.689	49.233	4
2	.0000	.3048+02	.0000	.0000	31.890	20.788	39
3	.0000	.6043+02	.0000	.0000	62.423	9.092	100
4	.0000	.3137+01	.4274+00	.0000	3.233	38.026	14

071N

3.5 SAMPLE PROBLEM #2

3.5.1 Introduction

In order to demonstrate the actual use of ZAP, the simple laser system shown in Figure 3.7 will be analyzed. In this laser, a spherical metal reflector contains a xenon flashlamp and a neodymium-doped glass laser rod; the lamp and rod are held at their ends by cylindrical holders. When the lamp is pulsed, power is produced throughout the volume of xenon; the amount produced depends on wavelength. The power travels through the plasma to the wall of the lamp (some is absorbed on the way), refracts through the wall, and enters the reflector volume. There the power bounces off the walls and other surfaces, travels through the glass laser rod and reflector surfaces. We are interested in the fractions of the initial power which end up in the rod, lamp, and walls, and also in the power distribution in the laser rod.

3.5.2 Initial Analysis of the Geometry

We can use the symmetry of the laser assembly to improve the statistics of our results, by putting perfectly reflecting planes in the two planes of symmetry (see Figure 3.7). This in effect gives four times as many rays per unit of run time (we could get the same result without the planes and their time-losing reflection by adding together the depositions in symmetrically equivalent volumes, but the computer time lost to the reflections is cheap compared to the human time gained in not adding numbers). The introduction of the symmetry planes reduces the problem to a quadrant of a sphere, as shown in Figure 3.8. The decomposition of the quadrant into successive mother and daughter segments is also shown in Figure 3.8. (The rod is divided into eight segments of essentially equal volume so that the energy distribution in

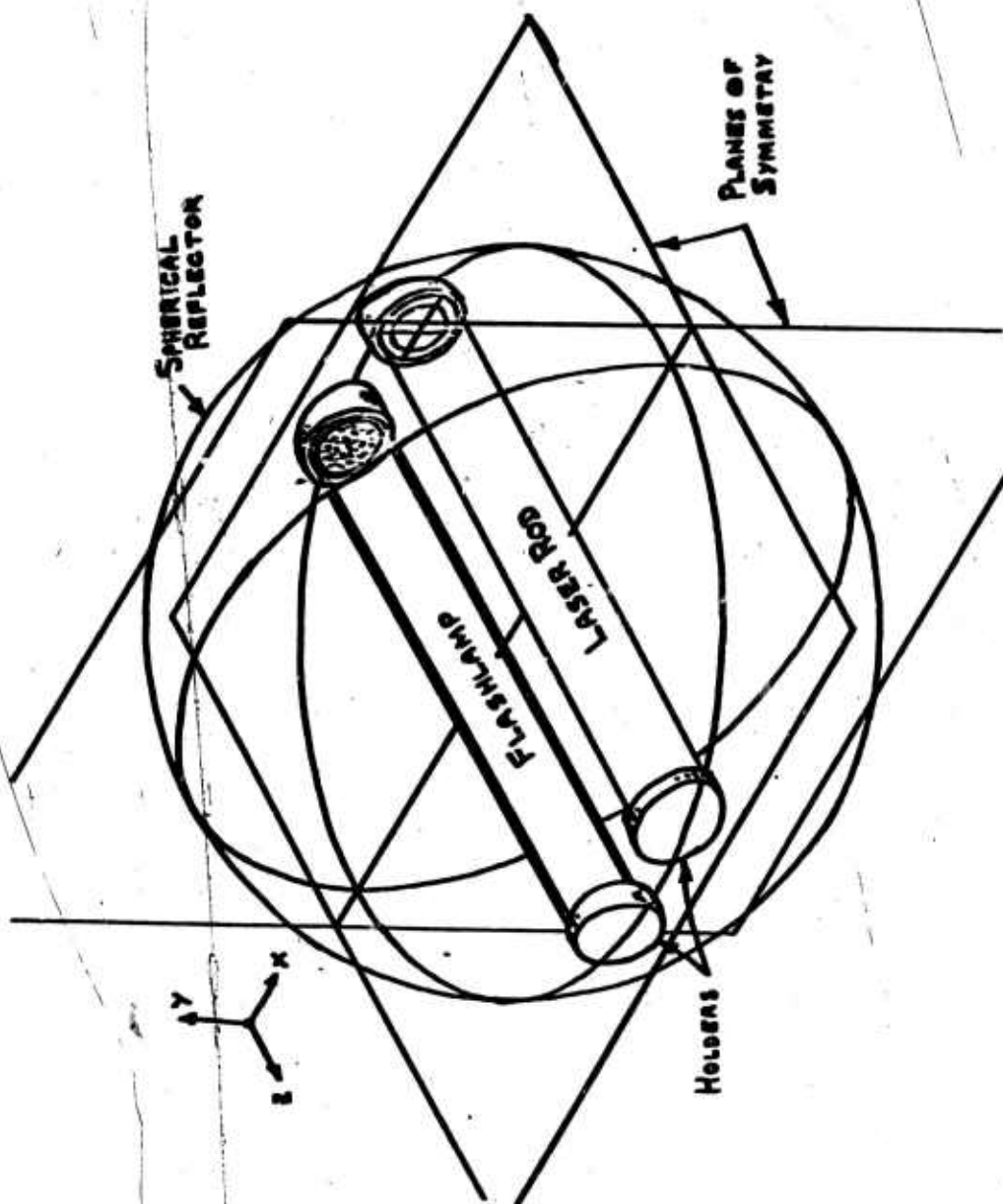


Figure 3.7 General view of laser analyzed. A 1 cm diameter xenon flashlamp pumps a 1 cm diameter Nd^{3+} -doped glass rod. Both are 10 cm long, and are enclosed in an 11 cm diameter spherical reflector. Cylindrical end pieces hold the rod and lamp.

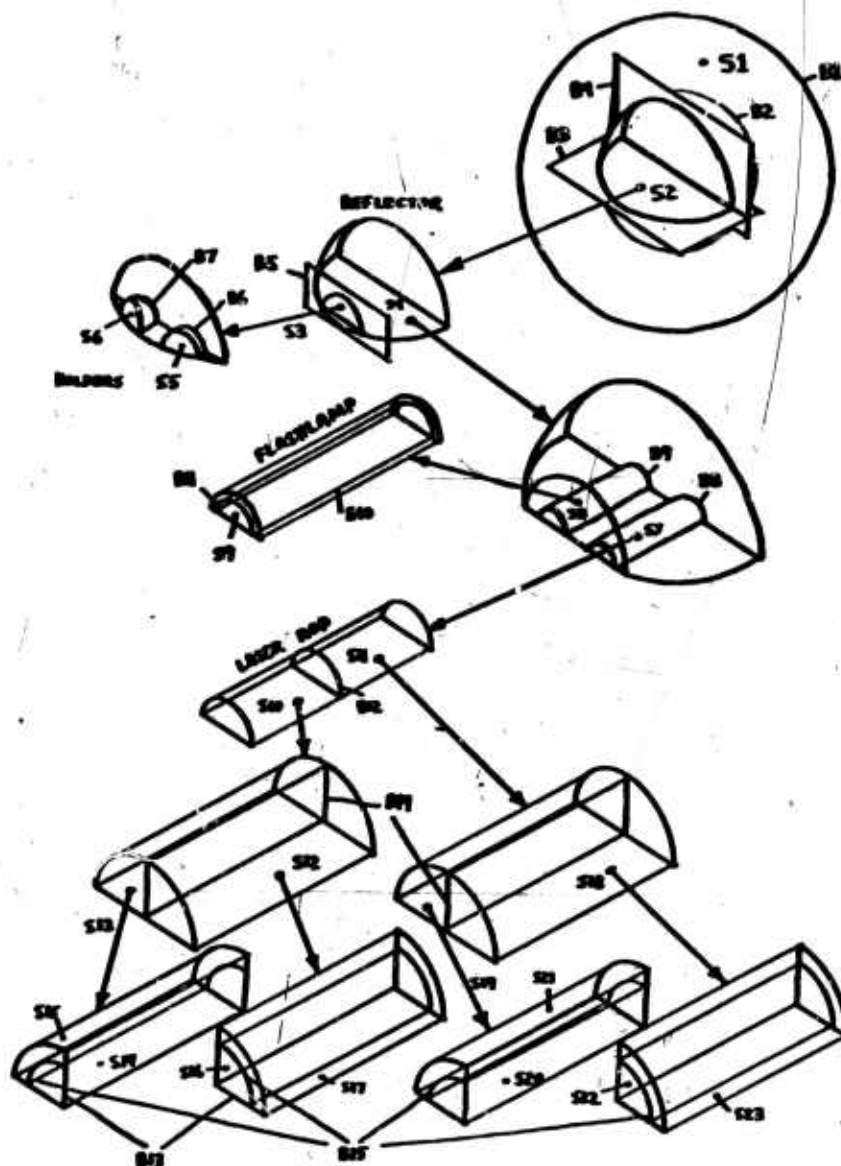


Figure 3.8 The decomposition of the laser assembly for ZAP input. Boundaries (indicated by numbers starting with B) are chosen so that all needed segments (indicated by numbers starting with S) can be outlined. The mother-daughter relationships are shown by arrows pointing from mother to daughter(s), or by inclusion in the mother. The laser rod is segmented so that the energy deposition pattern can be found.

it may be found.) The decomposition proceeds in a straightforward way from the all-enclosing segment 1 through smaller and smaller segments, always keeping the number of daughters small at each level. In fact, the number of daughters per level may be unnecessarily small - a comparison run with more daughters per mother in the rod gave essentially identical run times. Once the mother-daughter decomposition of the geometry is decided, the preparation of the input is easy (if tedious).

3.5.3 Input Preparation

While reading the following explanation of the input information the reader should refer to the listing of the computer input and output reproduced at the end of this section.

3.5.3.1 Reflection Data

Aside from the built-in reflection types 1, 2 and 3 (dummy boundary, total reflection, and dielectric boundary respectively) we will need three defined reflection types:

1. reflection varying smoothly from .85 for rays normal to the surface to 1.0 at grazing incidence to represent the reflector and other internal metal surfaces;
2. diffuse reflection returning half the incident light, to roughly represent the complicated end and electrode structure in the lamp;
3. reflection varying smoothly from .95 for rays normal to the surface to 1.0 at grazing incidence to model the dielectric mirrors at the ends of the laser rod.

These three types of reflection are entered as reflection types 11, 12 and 13.

3.5.3.2 Material Data

The metal, air and quartz materials are entered in a straightforward manner (the refractive indices for the metals and air are made slightly different so that ZAP will know that the boundaries between them are not just dummy boundaries, as it would assume if their indices were the same - see subroutine BOUNC). The xenon lamp plasma and Nd^{3+} - doped glass have absorption coefficients which vary with wavelength, and the user-written subroutines ABS2 and ABS3 must be supplied. These routines (see listing at end of section) are self-explanatory.

The absorption coefficient of the metals is made very large so that any light entering them will be immediately absorbed. Note that a small absorption has been added to the glass at all wavelengths; this is to insure some absorption (even if small) for all wavelengths, to avoid rays that dribble on forever.

3.5.3.3 Boundary Data

The boundaries (in this case only spheres, cylinders and planes) are just those necessary to specify the segments previously decided on. Because ZAP currently handles surfaces through the origin, the center of the geometry is displaced by .001 on each axis. When two reflection properties are required for different parts of a boundary, two coinciding boundaries are specified with different reflections, and the choice between them is made when the boundaries of segments are specified (note boundaries 5, 11 and 13).

3.5.3.4 Segment Data

Once the boundaries are specified, the segments are defined by giving all boundaries of each (and the volume, if the energy density is desired for that segment). The process is quite simple.

3.5.3.5 Structure Data

The mother-daughter relationships are entered for each segment. The information is most easily prepared by reference to a decomposition diagram such as Figure 3.8.

3.5.3.6 Lamp Data

For this problem there is only one type of source. This is the volume-emitting xenon plasma in the lamp. Its properties are specified in the user-supplied subroutine INTEN1, which is reproduced at the end of this section.

3.5.3.7 Source Data

There is only one source, namely the xenon lamp. It is specified geometrically by one boundary and a rectangular parallelepiped. For this run 500 rays are called for from the lamp, in order to get acceptably low errors in the segment deposition energies in the rod.

3.5.3.8 Test Data

Test points are specified for all segments in this example, although this is not necessary in general. Option 1 is used except when the segment volume is zero (because daughter segments take up all its volume), in which case Option 0 is used. Segments 1 and 2 also have points specified outside them (negative segment number). All points are correct, so no error messages appear.

3.5.3.9 Plot Data

Five plots are called for. There is a cross section near the center of the sphere (but just above the symmetry boundary) perpendicular to the rod and lamp; detail of the rod and lamp is poor because of the limited printer resolution. Another plot parallel to and through the rod and lamp suffers from the same lack of detail, although the general configuration is clear from both. The third plot is a close-up of the lamp-rod-holders-sphere area of the second plot; it clearly shows the quartz lamp wall, rod segmentation, and holder details. The fourth and fifth plots are close-up views of cuts across the rod and lamp respectively. These plots are very useful in checking that the geometry has been specified just as desired, and that no silly errors have crept in.

3.5.4 Discussion of Output

The output due to test points and plot requests has been discussed above. Here the power deposition results will be discussed.

3.5.4.1 Segment Results

The most interesting information in the segment results is the pattern of power deposition in the laser rod. The half of the rod near the sphere wall (segments 14, 15, 16 and 17) and the half near the center (segments 20, 21, 22 and 23) get essentially the same amount of scaled power - 47.13% and 52.86% respectively (we get these figures by adding the numbers in the percent scaled power column). The lensing effect of the laser rod has concentrated more power in the core of the rod (segments 14, 16, 20 and 22) than in the skin (segments 15, 17, 21 and 23); the core got 60.46% and the skin 39.53%. This effect is apparent in both the center and end

sections of the rod. Because the 500 rays have been enough to give approximately 10% errors in the deposition powers in the rod segments (see percent error column), these conclusions are reasonably reliable.

3.5.4.2 Material Results

The material results tell us the distribution of power among the various elements of the laser assembly. From the percent total power column, we see that the lamp absorbs by far the greatest fraction of the initial power. However, the lamp is a volume emitter and much of the power created in its volume may be absorbed before it ever gets out. To find out how much is internally reabsorbed in this way, another run was made with the absorption coefficient of the "air" in the cavity equal to 1000 cm^{-1} . The lamp was found to absorb 76.59% (error 2.29%) of the radiation before it got out for the first time, leaving 23.21% (error 7.64%) of the initial power as the amount actually emitted from the lamp (the remaining .2% was absorbed in the lamp ends). Thus we find that the cavity and holders absorbed 9.58% of the initial power (41.28% of the emitted power), the rod absorbed 6.70% (29.30%), and the lamp reabsorbed 7.02% (30.25%). The power returned to the lamp will act much like the lamp electrical input power (because the thermalization time in the dense lamp plasma is very short). If a fraction R of the lamp input is radiated as light with the remaining $1-R$ lost as heat, and if R is not changed greatly by the self-loading due to returned power (probably a good assumption for the low self-loading here, but more questionable with higher loadings), we can make a rough self-consistent calculation of what happens to the returned power and what the deposition fractions are as

follows (assume all spectral regions act like the .4 - .92 μ region we calculated):

The input electrical power is P

The power returned to the lamp from the cavity is Q

The total input is thus $P+Q$

The lamp light out is $R(P+Q)$

Then $Q = R(P+Q)F$ or $Q = (RFP)/(1-RF)$ where F is the fraction of lamp output reabsorbed (F is .3025 for the geometry we analyzed).

Then if F^1 is the fraction of P re-absorbed, we have $F^1 = Q/P = (FR)/(1-RF)$

And if L is the power absorbed by the laser rod

And if G is the fraction of lamp output which the rod gets (G is .2390 for our laser)

Then $L = GR(P+Q) = (GRP)/(1-RF)$ and so $G^1 = L/P = (GR)/(1-RF)$. Similarly, if C is the fraction of lamp output the cavity gets then $C^1 = (CR)/(1-RF)$.

Thus we see that self-consistency requires that all deposition fractions be multiplied by the factor $R/(1-RF)$. Taking the reasonable value of .5 for R , we find this factor to be about .59 for geometry we analyzed.

From the material power and scaled power for the laser glass (material 6) we find that 57% of the power the laser glass absorbed was useful laser power (the remainder was converted to heat in the rod when the absorbed photon energy was reduced to upper-laser level energy by radiationless transitions after absorption). The self-consistent fraction of electrical input the rod absorbed is .18, so .10 was useful

for laser operation and .08 went to rod heat (we have assumed a quantum efficiency of 1.0, which may be too high).

Our results, modified by the above self-consistency argument, thus imply that:

The electrical input was	1.00
The lamp heat loss was	.58
The cavity wall absorption was	.24
The laser rod heat loss was	.08
The upper-laser-level result was	.10

3.5.4.3 Summary Results

The total absorbed power and total source power are slightly different because of statistical fluctuation in the way rays are killed off. The execution time (on a CDC 3800, which has floating point add-subtract, multiply, and divide times of about 2, 5 and 10 μ s) was 1855.469 seconds. A run with one ray shows that the setup time is about 30 seconds, and thus the run time was roughly

$$T \approx 30 + 3.7 N \text{ seconds}$$

where N is the number of rays (each with 200 wavelengths) traced. The program and storage occupied about 60,000 words (48 bits each) of storage.

3.5.5 Conclusions

We have found that about 10% of the input electrical power is converted into a pumping rate into the upper laser level for the laser geometry and lamp conditions we specified. The actual laser output available will be decreased by spontaneous emission and (for pulsed lasers) by the fact that not all the upper-level energy can be extracted. However, the upper-level pump rate is probably the best measure of cavity efficiency and thus the best number to use when comparing cavities.

SAMPLE PROBLEM #2 - USER SUPPLIED SUBROUTINES

```

SUBROUTINE ABS2 ( XLAM, MAXF, TSTG, N )
C***** THIS SUBROUTINE READS IN THE CURRENT DENSITY AND DIAMETER
C***** OF THE FLASHLAMP, AND THE SCALE FACTOR FOR INPUT DIMENSIONS.
C***** THE SCALE FACTOR SPECIFIES THE NUMBER OF CENTIMETERS IN ONE
C***** UNIT OF THE QUANTITY USED FOR INPUT SIZE SPECIFICATIONS.
C***** THUS IF THE INPUT IS IN INCHES, SCALE = 2.54, AND IF THE
C***** INPUT IS IN CENTIMETERS, SCALE = 1.0. THE CURRENT DENSITY IS
C***** IN AMPS PER SQUARE INPUT UNIT, AND THE DIAMETER IN INPUT
C***** UNITS. THEREFORE, THE SCALE OF THE ENTIRE PROBLEM MAY BE
C***** CHANGED MERELY BY CHANGING SCALE. THIS SUBROUTINE SHOULD BE
C***** ENTERED BEFORE ABS3, BECAUSE ABS3 USES THE INFORMATION READ
C***** BY THIS ROUTINE AND STORED IN COMMON.
      DIMENSION XLAM(1), TSTG(1)
      COMMON /CUR/ CUR, DIAM, SCALE
      READ 20, CUR, DIAM, SCALE
      PRINT 30, CUR, DIAM, SCALE
C***** CONVERT TO MKS UNITS
      CUR = 10000.*CUR/SCALE/SCALE
      DIAM = .01*DIAM*SCALE
C***** FILL UP ABSORPTION COEFF ARRAY
      DO 10 IA=1,MAXF
        TSTG(IA) = ALPHA(XLAM(IA),CUR)*.01*SCALE
      10 CONTINUE
      RETURN
20 FORMAT(3E20.9)
30 FORMAT(20I,23HLAMP CURRENT DENSITY = ,E16.9,
1 30I AMPS PER SQUARE INPUT LENGTH UNIT/
2 20I,19HDIAMETER = ,E16.9,19H INPUT LENGTH UNITS/
3 20I,15HSCALE FACTOR = ,E16.9,
4 30I CENTIMETERS PER INPUT LENGTH UNIT)
      END

```

SAMPLE PROBLEM #2 - USER SUPPLIED SUBROUTINES

```

SUBROUTINE ABS3 ( XLAM, MAXF, TSTG, N )
C***** THIS SUBROUTINE DETERMINES, FOR EACH OF MAXF WAVELENGTHS
C***** STORED IN XLAM, THE ABSORPTION COEFFICIENT (IN RECIPROCAL
C***** INPUT LENGTH UNITS) OF 0-1 ED-2 ND3+ LASER GLASS. THE
C***** INPUT INFORMATION TO THE SUBROUTINE IS A DECK OF 261 CARDS
C***** CONTAINING WAVELENGTHS (IN MICRONS) AND ABSORPTION COEFFI-
C***** CIENTS (IN PER CENTINTEIENS) AT 20 ANGSTROM INTERVALS FROM
C***** 4000 TO 9200 ANGSTROMS. EACH CARD HAS A SEQUENCE NUMBER
C***** IN I3 FORMAT, 5 BLANKS, AND THE WAVELENGTH AND ABSORPTION
C***** COEFFICIENT IN ZE10.6 FORMAT.
C***** FOR EACH WAVELENGTH IN XLAM, THE SUBROUTINE DETERMINES IF
C***** THAT WAVELENGTH IS WITHIN THE 4000 - 9200 ANGSTROM REGION.
C***** IF IT IS NOT, THE ABSORPTION COEFFICIENT AT THAT WAVELENGTH
C***** IS SET TO A SMALL VALUE. IF IT IS, THE ABSORPTION COEFFI-
C***** CIENT IS FOUND BY LINEAR INTERPOLATION BETWEEN THE TWO
C***** NEAREST DATA CARD VALUES.
DIMENSION XLAM(1), TSTG(1), WAVAR(261), ALPHAR(262)
COMMON /CUR/ CUR, DIAM, SCALE

C***** READ IN DATA CARD DECK
DO 30 IA=1,261
  READ 80,10,WAV,ALPH
C***** CHECK FOR CARD OUT OF SEQUENCE
  IF(IA-10) 10,20,10
10  PRINT 90,10
20  WAVAR(IA) = WAV
  ALPHAR(IA) = (ALPH+.01)*SCALE
30 CONTINUE
C***** PUT IN EXTRA ELEMENT FOR INTERPOLATION AT LAST POINT
  ALPHAR(262) = .01*SCALE

C***** FILL TSTG WITH ABSORPTION COEFFICIENTS CORRESPONDING TO XLAMS
DO 70 IC=1,MAXF
  YLAM = XLAM(IC)
C***** TEST IF BELOW MIN WAVELENGTH
  IF(YLAM-.4) 40,60,50
40  TSTG(IC) = .01*SCALE
  GO TO 70
C***** TEST IF ABOVE MAX WAVELENGTH
50  IF(YLAM-.92) 60,10,40
C***** IF BETWEEN LIMITS, FIND INDEX OF DATA CARD JUST BELOW
60  N = (YLAM-.4)*500. + 1.
C***** FIND AN. CO. BY LINEAR INTERPOLATION BETWEEN BRACKETING CARDS
  TSTG(IC)=ALPHAR(N)+(YLAM-WAVAR(N))*500.*(ALPHAR(N+1)-ALPHAR(N))
70 CONTINUE
  PRINT 100
  RETURN
80 FORMAT(13,5X,ZE10.6)
90 FORMAT('00GLASS ABSORPTION CARD #13 OUT OF ORDER - CONTINUING')
100 FORMAT(20X,20GLASS DATA READ IN FROM CAMOS)
END

```

SAMPLE PROBLEM #2 - USER SUPPLIED SUBROUTINES

```

SUBROUTINE INTEN1 ( XLAM, MAXF, TSTG, LAMNO )
C***** THIS SUBROUTINE STORES IN TSTG FOR EACH WAVELENGTH IN XLAM
C***** THE LAMP VOLUME EMISSION IN WATTS PER MICRON PER CUBIC INPUT
C***** LENGTH UNIT. IT IS BASED ON THE FACT THAT THE VOLUME
C***** EMISSION IS 4 TIMES THE BLACK BODY SPECTRAL EMISSIVITY TIMES
C***** THE ABSORPTION-EMISSION COEFFICIENT FOR A PLASMA IN LOCAL
C***** THERMAL EQUILIBRIUM. THE TEMPERATURE USED FOR THE BLACK BODY
C***** EMISSIVITY IS ASSUMED UNIFORM ACROSS THE LAMP.
      DIMENSION XLAM(1), TSTG(1)
      COMMON /CUR/ CUR, DIAM, SCALE
C***** THE TEMPERATURE OF THE LAMP IS FOUND FROM AN EMPIRICAL FORMULA
      TLO = 6450.*DIAM**.03*CUR**.01
      THI = 63.*DIAM**.27*CUR**.36
      T = (TLO**6+THI**6)**.1666667
C***** THE SPECIFIC OUTPUT IS THE BLACK BODY EMITTANCE TIMES THE
C***** ABSORPTION COEFFICIENT TIMES FOUR.
      DO 10 IA=1,MAXF
        X = XLAM(IA)
        DENOM = EXP(14.4*THI/(1+387.9/X/T+.69)) - 1.
        BLACK = 3.7405E8/X/X/X/X/X/DENOM
        TSTG(IA) = 4.*ALPHA(X,CUR)*BLACK*SCALE*SCALE*SCALE*1.E-6
      10 CONTINUE
      RETURN
      END

```

SAMPLE PROBLEM #2 - USER SUPPLIED SUBROUTINES

```

      FUNCTION ALPHA ( WAVE, CUR )
      C***** THIS FUNCTION RETURNS THE APPROXIMATE ABSORPTION-EMISSION
      C***** COEFFICIENT AT WAVELENGTH WAVE (MICRONS) OF A XENON PLASMA
      C***** WITH CURRENT DENSITY CUR (AMPS PER SQUARE METER) FLOWING THRU
      C***** IT. THE APPROXIMATION CONSISTS OF A SET OF LORENTZIAN LINES
      C***** OF FIXED POSITION AND WIDTH PLUS A CONTINUUM, ALL OF WHICH
      C***** ARE PROPORTIONAL TO THE SQUARE OF THE CURRENT DENSITY.
      C***** THE RETURNED VALUE IS IN RECIPROCAL METERS.
      C***** MODEL BY JOHN W. TRENHOLME AS OF 27 MARCH 1970
      C*****
      C***** THE FOLLOWING DATA BLOCK CONTAINS THE XENON LINE SPECTRUM
      C***** USED TO FIND THE ABSORPTION-EMISSION COEFFICIENT. EACH GROUP
      C***** OF THREE NUMBERS REPRESENTS THE CENTER WAVELENGTH (MICRONS),
      C***** PEAK ABSORPTION-EMISSION (PER METER), AND FULL WIDTH AT HALF
      C***** MAXIMUM (MICRONS) FOR ONE LORENTZIAN LINE AT 1000 AMPS PER
      C***** SQUARE CENTIMETER (1.E7 AMPS PER SQUARE METER).
      DIMENSION RE (A)
      DATA (AXE=0), 1XE= .4560,1...015, .4920,1...015, .6077,2...015,
      1 .7643,4...015, .8232,375...000, .8240,300...003,
      2 .8367,225...004, .8409,225...003, .8819,375...009,
      3 .9952,300...004, .9845,375...003, .9162,375...004,
      4 .9374,150...007, .9513,150...007, .9800,375...004,
      5 .9923,375...004, 1.0107,120...007, 1.0065,30...007,
      6 1.1100,45...007, 1.1742,75...007, 1.2623,45...007,
      7 1.3656,30...007, 1.4180,30...015, 1.4732,75...007,
      8 1.5418,30...007, 1.6052,15...007, 1.6720,10...007)
      C***** FIRST THE APPROXIMATE CONTINUUM IS EVALUATED AT 1000 A/CM2
      AL = 15.*EXP(-8.*(WAVE-.7)*(WAVE-.7)) * 1.
      C***** THEN THE LINES ARE ADDED
      DO 10 J=1,NXE,3
      AL = AL + RE(J)/14.*(WAVE-RE(J))/(RE(J)+2)*((WAVE-RE(J))/RE(J)+2)
      1 ) * 1.)
      10 CONTINUE
      C***** THE RESULT IS SCALED BY THE SQUARE OF THE CURRENT DENSITY
      RATIO = CUR/1.E7
      ALPHA = AL*RATIO*RATIO
      RETURN
      END

```

SAMPLE PROBLEM #2 - TEST RUN INPUT

MULTIFREQUENCY TEST ON SPHERE WITH MOD AND LAMP

.5	0.0	.4	.92
200	0	0	1
11 FUNCT	.15		
12 DIFFUS	.5		
13 FUNCT	.05		
1	1000.	.99	0.
2	1000.	.98	0.
3	0.	1.	0.
4	.001	1.46	0.
5 AMS2	0.	1.01	0.
3000.			
6 AM53	0.	1.56	1.06
1	4.000000-001		0-1 EU-2 M73 LASER GLASS IN MOD
2	4.020000-001		0.000000-000
3	4.040000-001		2.150000-001
4	4.060000-001		2.273000-001
5	4.080000-001		1.920000-001
6	4.100000-001		1.609000-001
7	4.120000-001		1.536000-001
8	4.140000-001		1.766000-001
9	4.160000-001		1.555000-001
10	4.180000-001		1.536000-001
11	4.200000-001		1.612000-001
12	4.220000-001		1.766000-001
13	4.240000-001		1.609000-001
14	4.260000-001		1.612000-001
15	4.280000-001		1.920000-001
16	4.300000-001		3.456000-001
17	4.320000-001		4.992000-001
18	4.340000-001		3.072000-001
19	4.360000-001		1.920000-001
20	4.380000-001		1.456200-001
21	4.400000-001		1.305000-001
22	4.420000-001		1.152000-001
23	4.440000-001		1.152000-001
24	4.460000-001		9.216000-002
25	4.480000-001		1.302000-001
26	4.500000-001		1.456200-001
27	4.520000-001		1.609000-001
28	4.540000-001		2.073000-001
29	4.560000-001		2.841000-001
30	4.580000-001		2.995200-001
31	4.600000-001		3.072000-001
32	4.620000-001		3.225000-001
33	4.640000-001		3.072000-001
34	4.660000-001		3.456000-001
35	4.680000-001		3.840000-001
36	4.700000-001		4.167200-001
37	4.720000-001		4.830000-001
38	4.740000-001		4.454000-001
39	4.760000-001		4.454000-001
40	4.780000-001		4.377600-001
41	4.800000-001		3.840000-001
42	4.820000-001		3.072000-001
43	4.840000-001		1.996000-001
44	4.860000-001		1.536000-001
45	4.880000-001		1.302000-001
46	4.900000-001		1.228000-001
47	4.920000-001		1.305600-001

SAMPLE PROBLEM #2 - TEST RUN INPUT

48	4.940000-001	1.459200-001
49	4.960000-001	2.073000-001
50	4.980000-001	2.686800-001
51	5.000000-001	2.686800-001
52	5.020000-001	3.072000-001
53	5.040000-001	3.040000-001
54	5.060000-001	4.992000-001
55	5.080000-001	5.990000-001
56	5.100000-001	7.907200-001
57	5.120000-001	1.036000-000
58	5.140000-001	1.213400-000
59	5.160000-001	1.036000-000
60	5.180000-001	9.660000-001
61	5.200000-001	9.660000-001
62	5.220000-001	1.013700-000
63	5.240000-001	1.336320-000
64	5.260000-001	1.950000-000
65	5.280000-001	1.950720-000
66	5.300000-001	1.600000-000
67	5.320000-001	1.420000-000
68	5.340000-001	1.036000-000
69	5.360000-001	6.912600-001
70	5.380000-001	4.224000-001
71	5.400000-001	2.611200-001
72	5.420000-001	1.920000-001
73	5.440000-001	9.216000-002
74	5.460000-001	4.600000-002
75	5.480000-001	4.600000-002
76	5.500000-001	5.376000-002
77	5.520000-001	4.000000-002
78	5.540000-001	6.144000-002
79	5.560000-001	7.000000-002
80	5.580000-001	9.900000-002
81	5.600000-001	1.152000-001
82	5.620000-001	1.300000-001
83	5.640000-001	2.600000-001
84	5.660000-001	6.144000-001
85	5.680000-001	1.170000-000
86	5.700000-001	1.000000-000
87	5.720000-001	3.187200-000
88	5.740000-001	4.100000-000
89	5.760000-001	4.492000-000
90	5.780000-001	4.646000-000
91	5.800000-001	4.646000-000
92	5.820000-001	5.414400-000
93	5.840000-001	6.500000-000
94	5.860000-001	7.100000-000
95	5.880000-001	6.720000-000
96	5.900000-001	5.292000-000
97	5.920000-001	4.224000-000
98	5.940000-001	3.302000-000
99	5.960000-001	2.611200-000
100	5.980000-001	1.920000-000
101	6.000000-001	1.420000-000
102	6.020000-001	1.075200-000
103	6.040000-001	7.600000-001
104	6.060000-001	5.376000-001
105	6.080000-001	3.040000-001
106	6.100000-001	2.000000-001
107	6.120000-001	2.227200-001
108	6.140000-001	1.459200-001
109	6.160000-001	1.228000-001
110	6.180000-001	1.228000-001
111	6.200000-001	1.342000-001
112	6.220000-001	1.536000-001
113	6.240000-001	1.536000-001
114	6.260000-001	1.536000-001
115	6.280000-001	1.536000-001
116	6.300000-001	1.302000-001
117	6.320000-001	1.305000-001
118	6.340000-001	1.228000-001
119	6.360000-001	1.075200-001

SAMPLE PROBLEM #2 - TEST RUN INPUT

120	6.340000-001	9.216000-002
121	6.400000-001	8.440000-002
122	6.420000-001	7.600000-002
123	6.440000-001	7.000000-002
124	6.460000-001	7.000000-002
125	6.480000-001	6.912000-002
126	6.500000-001	6.912000-002
127	6.520000-001	6.912000-002
128	6.540000-001	6.912000-002
129	6.560000-001	6.912000-002
130	6.580000-001	7.000000-002
131	6.600000-001	7.000000-002
132	6.620000-001	7.000000-002
133	6.640000-001	7.000000-002
134	6.660000-001	7.000000-002
135	6.680000-001	9.216000-002
136	6.700000-001	1.220000-001
137	6.720000-001	1.050000-001
138	6.740000-001	1.536000-001
139	6.760000-001	2.073000-001
140	6.780000-001	2.304000-001
141	6.800000-001	2.011200-001
142	6.820000-001	2.995200-001
143	6.840000-001	3.456000-001
144	6.860000-001	3.225000-001
145	6.880000-001	2.001600-001
146	6.900000-001	2.304000-001
147	6.920000-001	1.536000-001
148	6.940000-001	1.152000-001
149	6.960000-001	6.912000-002
150	6.980000-001	6.912000-002
151	7.000000-001	3.040000-002
152	7.020000-001	4.000000-002
153	7.040000-001	4.000000-002
154	7.060000-001	4.000000-002
155	7.080000-001	4.000000-002
156	7.100000-001	4.000000-002
157	7.120000-001	4.000000-002
158	7.140000-001	5.376000-002
159	7.160000-001	6.144000-002
160	7.180000-001	6.144000-002
161	7.200000-001	7.000000-002
162	7.220000-001	9.216000-002
163	7.240000-001	1.305000-001
164	7.260000-001	1.920000-001
165	7.280000-001	2.001600-001
166	7.300000-001	3.040000-001
167	7.320000-001	6.912000-001
168	7.340000-001	1.302400-000
169	7.360000-001	2.304000-000
170	7.380000-001	3.000000-000
171	7.400000-001	3.032000-000
172	7.420000-001	3.763200-000
173	7.440000-001	3.000000-000
174	7.460000-001	3.648000-000
175	7.480000-001	4.070400-000

SAMPLE PROBLEM #2 - TEST RUN INPUT

176	7.540000-001	4.147200+000
177	7.520000-001	3.456000+000
178	7.540000-001	2.995200+000
179	7.500000-001	2.534400+000
180	7.540000-001	1.803200+000
181	7.600000-001	1.344000+000
182	7.620000-001	1.075200+000
183	7.640000-001	7.600000-001
184	7.640000-001	6.144000-001
185	7.640000-001	4.664000-001
186	7.700000-001	3.004000-001
187	7.720000-001	3.072000-001
188	7.740000-001	3.072000-001
189	7.700000-001	3.072000-001
190	7.740000-001	3.225600-001
191	7.800000-001	3.046400-001
192	7.820000-001	4.600000-001
193	7.840000-001	6.144000-001
194	7.860000-001	7.600000-001
195	7.840000-001	1.036000+000
196	7.900000-001	1.305600+000
197	7.920000-001	1.704000+000
198	7.940000-001	2.304000+000
199	7.960000-001	2.995200+000
200	7.940000-001	3.456000+000
201	8.000000-001	3.001600+000
202	8.020000-001	3.916000+000
203	8.040000-001	4.302400+000
204	8.060000-001	6.254200+000
205	8.040000-001	6.102400+000
206	8.100000-001	4.761600+000
207	8.120000-001	3.417600+000
208	8.140000-001	2.606000+000
209	8.140000-001	1.996800+000
210	8.140000-001	1.536000+000
211	8.200000-001	1.222400+000
212	8.220000-001	1.075200+000
213	8.240000-001	8.832000-001
214	8.240000-001	7.292400-001
215	8.260000-001	5.376000-001
216	8.300000-001	4.224000-001
217	8.320000-001	3.456000-001
218	8.340000-001	2.457600-001
219	8.360000-001	1.766400-001
220	8.340000-001	1.302400-001
221	8.400000-001	1.302400-001
222	8.420000-001	1.302400-001
223	8.440000-001	1.302400-001
224	8.460000-001	1.302400-001
225	8.440000-001	1.302400-001
226	8.500000-001	1.459200-001
227	8.520000-001	1.536000-001
228	8.540000-001	1.609600-001
229	8.500000-001	2.150400-001
230	8.540000-001	3.225600-001
231	8.600000-001	4.224000-001
232	8.620000-001	5.996400-001
233	8.640000-001	7.445600-001
234	8.600000-001	5.216000-001
235	8.640000-001	9.600000-001
236	8.700000-001	9.600000-001
237	8.720000-001	9.676000-001
238	8.740000-001	1.036000+000
239	8.760000-001	1.305600+000

SAMPLE PROBLEM #2 - TEST RUN INPUT

240	8.740000-001	1.000000-000
241	8.900000-001	1.500000-000
242	8.400000-001	1.300000-000
243	8.800000-001	1.100000-000
244	8.500000-001	8.000000-000
245	8.400000-001	7.200000-001
246	8.900000-001	6.700000-001
247	8.920000-001	5.700000-001
248	8.900000-001	4.800000-001
249	8.900000-001	3.800000-001
250	8.900000-001	3.400000-001
251	9.000000-001	2.900000-001
252	9.000000-001	2.600000-001
253	9.000000-001	2.150000-001
254	9.000000-001	1.500000-001
255	9.000000-001	1.150000-001
256	9.100000-001	7.000000-002
257	9.100000-001	6.100000-002
258	9.100000-001	4.600000-002
259	9.100000-001	3.070000-002
260	9.100000-001	1.500000-002
261	9.200000-001	0.000000-000

1	2	SPHERE	.001	.001	.001	100.			
2	11	SPHERE	.001	.001	.001	5.5			
3	2	PLANE		1.		.001	.001	.001	
4	2	PLANE			1.	.001	.001	.001	
5	1	PLANE		1.	.001	.001	.001	.001	
6	11	CYLIND		1.	1.	.001	.001	5.	
7	11	CYLIND		1.	-1.	.001	.001	5.	.55
8	3	CYLIND		1.	1.	.001	.001	5.	.05
9	3	CYLIND		1.	-1.	.001	.001	.001	.5
10	3	CYLIND		1.	-1.	.001	.001	.001	.6
11	12	PLANE		1.	.001	.001	.001	.001	.5
12	1	PLANE		1.	.001	.001	.001	5.	
13	13	PLANE		1.	.001	.001	.001	2.5	
14	1	PLANE	1.		.001	.001	.001	5.	
15	1	CYLIND		1.	1.	.001	.001	.001	.354

[illegible]

SAMPLE PROBLEM #2 - TEST RUN INPUT

```

2  3  4
3  5  6
4  7  8
7 10 11
8  9
10 12 13
11 10 19
12 16 17
13 14 15
18 22 23
19 20 21

```

1 VOLUME INTEN1

1	9	1	500	-1.5	-.5	.001	.5	.001	5.
1	1	10.			10.		10.		
-1	1	60.			60.		60.		
2	0	2.			3.		3.		
-2	0	2.			-3.		3.		
3	1	.3			1.		5.3		
4	1	2.			1.		3.		
5	1	1.1			.3		5.2		
6	1	-.9			.2		5.3		
7	0	1.2			.1		2.4		
8	1	-1.4			.4		4.		
9	1	-.6			.2		.1		
10	0	.9			.4		3.		
11	0	1.3			.3		2.		
12	0	1.1			.4		4.		
13	0	.9			.1		3.		
14	1	.8			.2		4.		
15	1	.7			.3		3.		
16	1	1.2			.2		4.		
17	1	1.3			.3		3.		
18	0	1.2			.2		2.		
19	0	.8			.4		1.		
20	1	.7			.1		2.		
21	1	.6			.2		1.		
22	1	1.3			.1		2.		
23	1	1.4			.2		1.		
-6.	-.5	.01	6.	-.5	.01	-6.	6.	.01	
-6.	.01	-.5	6.	.01	-.5	-6.	.01	6.	
-1.7	.01	4.	1.6	.01	4.	-1.7	.01	6.	
.45	-.05	.01	1.55	-.05	.01	.45	.55	.01	
-1.65	-.1	.01	-.35	-.1	.01	-1.65	.65	.01	

SAMPLE PROBLEM #2 - TEST RUN OUTPUT

GENERAL INPUT FOR ZAP

MULTIFREQUENCY TEST ON SPHERE WITH RCD AND LAMP

ROULETTE FRACTION= 5.0000-001
 DISTANCE BETWEEN CONTINUATIVE BOUNDARIES= 0.0020-000
 MINIMUM WAVELENGTH= 4.0000-001
 MAXIMUM WAVELENGTH= 0.2000-001
 NUMBER OF WAVELENGTHS= 200
 RAY PRINT OPTION= 0
 PERUG PRINT OPTION= 0
 STARTING RANDOM INTEGER= 1

SAMPLE PROBLEM #2 - TEST RUN OUTPUT

REFLECTION NUMBER	TYPE	REFLECTION DATA									
		1	2	3	4	5	6	7	8	9	10
11	PUNCT	0.1500	-0.0000	-0.0000	-0.0000	-0.0000	-0.0000	-0.0000	-0.0000	-0.0000	-0.0000
12	DIFFUS	0.5000	-0.0000	-0.0000	-0.0000	-0.0000	-0.0000	-0.0000	-0.0000	-0.0000	-0.0000
13	PUNCT	0.0500	-0.0000	-0.0000	-0.0000	-0.0000	-0.0000	-0.0000	-0.0000	-0.0000	-0.0000

MATERIAL NUMBER	INPUT ROUTINE	MATERIAL DATA									
		ABSORPTION COEFFICIENT	REFRACTIVE INDEX	LASING WAVELENGTH	DESCRIPTION						
1		1000.000000	0.990000	0.000000	METAL SPHERICAL REFLECTOR						
2		1000.000000	0.980000	0.000000	METAL CYLINDRICAL HOLOGERS						
3		0.000000	1.000000	0.000000	AIR INSIDE SYSTEM						
4		0.001000	1.460000	0.000000	QUARTZ LAMP ENVELOPE						
5	AR52	0.000000	1.010000	0.000000	XENON PLASMA IN LAMP						
		LAMP CURRENT DENSITY = 3.0-0000000-000 AMPS PER SQUARE INPUT LENGTH UNIT									
		ASSUMED DIAMETER = 1.000000000-000 INPUT LENGTH UNIT									
		SCALE FACTOR = 1.000000000-000 CENTIMETERS PER INPUT LENGTH UNIT									
6	AR53	0.000000	1.560000	1.060000	0-1 ED-2 ND10 LASER GLASS IN ROO						
		GLASS DATA READ IN FROM CARDS									

SAMPLE PROBLEM #2 - TEST RUN OUTPUT

3SR-405

BOUNDARY DATA													
BOUND NUMB	REFL NUMB	BOUND TYPE	ALPHA	BETA	GAMMA	X	Y	Z		X2	Y2	X3	Y3
1	2	SPHERE				0.001	0.001	0.001					
2	11	SPHERE				0.001	0.001	0.001					100.000
3	2	PLANE	0.000	1.000	0.000	0.001	0.001	0.001					9.500
4	2	PLANE	0.000	1.000	0.000	0.001	0.001	0.001					0.001
5	1	PLANE	0.000	0.000	1.000	0.001	0.001	0.000					0.001
6	11	CYLIND	0.000	0.000	1.000	0.000	0.001	0.000					9.000
7	11	CYLIND	0.000	0.000	1.000	-1.000	0.001	0.000					0.550
8	3	CYLIND	0.000	0.000	1.000	-1.000	0.001	0.001					0.650
9	3	CYLIND	0.000	0.000	1.000	-1.000	0.001	0.001					0.500
10	12	PLANE	0.000	0.000	1.000	0.001	0.001	0.001					0.400
11	1	PLANE	0.000	0.000	1.000	0.001	0.001	0.000					0.500
12	1	PLANE	0.000	0.000	1.000	0.001	0.001	2.500					9.000
13	1	PLANE	0.000	0.000	1.000	0.001	0.001	0.000					2.500
14	1	PLANE	0.000	0.000	1.000	0.001	0.001	0.001					9.000
15	1	CYLIND	0.000	0.000	1.000	0.001	0.001	0.001					1.000
						1.000	0.001	0.001					0.354

SEGMENT DATA													
SEGMENT NUMBER	MATERIAL NUMBER	CONFINING BOUNDARIES										VOLUME	
1	1	1	2	2	2	2	2	2	2	2	2	0.000000	0.000000
2	2	2	2	2	2	2	2	2	2	2	2	0.000000	0.000000
3	3	3	3	3	3	3	3	3	3	3	3	0.000000	0.000000
4	4	4	4	4	4	4	4	4	4	4	4	0.000000	0.000000
5	5	5	5	5	5	5	5	5	5	5	5	0.000000	0.000000
6	6	6	6	6	6	6	6	6	6	6	6	0.000000	0.000000
7	7	7	7	7	7	7	7	7	7	7	7	0.000000	0.000000
8	8	8	8	8	8	8	8	8	8	8	8	0.000000	0.000000
9	9	9	9	9	9	9	9	9	9	9	9	0.000000	0.000000
10	10	10	10	10	10	10	10	10	10	10	10	0.000000	0.000000
11	11	11	11	11	11	11	11	11	11	11	11	0.000000	0.000000
12	12	12	12	12	12	12	12	12	12	12	12	0.000000	0.000000
13	13	13	13	13	13	13	13	13	13	13	13	0.000000	0.000000
14	14	14	14	14	14	14	14	14	14	14	14	0.000000	0.000000
15	15	15	15	15	15	15	15	15	15	15	15	0.000000	0.000000
16	16	16	16	16	16	16	16	16	16	16	16	0.000000	0.000000
17	17	17	17	17	17	17	17	17	17	17	17	0.000000	0.000000
18	18	18	18	18	18	18	18	18	18	18	18	0.000000	0.000000
19	19	19	19	19	19	19	19	19	19	19	19	0.000000	0.000000
20	20	20	20	20	20	20	20	20	20	20	20	0.000000	0.000000
21	21	21	21	21	21	21	21	21	21	21	21	0.000000	0.000000
22	22	22	22	22	22	22	22	22	22	22	22	0.000000	0.000000
23	23	23	23	23	23	23	23	23	23	23	23	0.000000	0.000000

SAMPLE PROBLEM #2 - TEST RUN OUTPUT

STRUCTURE DATA

DAUGHTER SEGMENTS

MOTHER SEGMENT

1	2
2	3
3	4
4	5
5	6
6	7
7	8
8	9
9	10
10	11
11	12
12	13
13	14
14	15
15	16
16	17
17	18
18	19
19	20
20	21

LAMP DATA

LAMP NUMBER	LAMP TYPE	INTENSITY ROUTINE	ANGLE ROUTINE	INTENSITY
1	VOLUME	INTEN1		-0.0000+000

3SR-405

SAMPLE PROBLEM #2 - TEST RUN OUTPUT

SOURCE NUMBER	SPHERICAL NUMBER	BOUNDARY NUMBER	LAMP NUMBER	NUMBER OF RAYS	SOURCE DATA			
					GEOMETRIC SOURCE DATA			
1	0	0	1	500	-1.9000	-0.5000	0.00100	0.00100
								9.00000

TEST DATA

SEQUENT OPTION	X	Y	Z
1	1.0000+001	1.0000+001	1.0000+001
-1	4.0000+001	6.0000+001	6.0000+001
-2	2.0000+000	3.0000+000	3.0000+000
-3	2.0000+000	-3.0000+000	3.0000+000
4	3.0000+001	1.0000+000	3.0000+000
5	2.0000+000	1.0000+000	3.0000+000
6	1.0000+000	2.0000+001	3.0000+000
7	-9.0000+001	2.0000+001	3.0000+000
8	-1.2000+000	1.0000+001	2.4000+000
9	-1.4000+000	4.0000+001	4.0000+000
0	-6.0000+001	2.0000+001	1.0000+001
1	9.0000+001	4.0000+001	3.0000+000
2	1.0000+000	3.0000+001	2.0000+000
3	1.0000+001	4.0000+001	4.0000+000
4	9.0000+001	1.0000+001	3.0000+000
5	7.0000+001	2.0000+001	4.0000+000
6	1.2000+000	2.0000+001	4.0000+000
7	1.0000+000	2.0000+001	3.0000+000
8	1.2000+000	2.0000+001	2.0000+000
9	7.0000+001	4.0000+001	1.0000+000
0	7.0000+001	1.0000+001	2.0000+000
1	6.0000+001	2.0000+001	1.0000+000
2	1.0000+000	1.0000+001	2.0000+000
3	1.4000+000	2.0000+001	1.0000+000

PLOT OF PLANE DEFINED BY POINTS P0 P1 P2

```

P0 (LOWER LEFT CORNER OF PLOT) HAS          P1 (LOWER RIGHT CORNER OF PLOT) HAS
X=6.000+000 Y=1.000+002 Z=5.000+001          X= 6.000+000 Y= 1.000+002 Z=5.000+001
X=6.000+000 Y=1.000+002 Z=5.000+001          P2 (SOMEPHERE ALONG TOP LINE) HAS
X=6.000+000 Y=1.000+002 Z= 6.000+000

```

[illegible]

129

P0 (LOWER LEFT CORNER OF PLOT) HAS
X= 4.500-001 Y=5.000-002 Z= 1.000-002

P1 (LOWER RIGHT CORNER OF PLOT) HAS
X= 1.500-000 Y=5.000-002 Z= 1.000-002

P2 (SOMewhere ALONG TOP LINE) HAS
X= 4.500-001 Y= 5.000-001 Z= 1.000-002

PLOT OF PLANE DEFINED BY POINTS P0 P1 P2

131

SAMPLE PROBLEM #2 - TEST RUN OUTPUT

SEGMENT RESULTS												
MOTHER SEGMENT NUMBER	DAUGHTER SEGMENT NUMBER	MATERIAL TYPE	TOTAL SPGHT PUMPH	SCALED SEGMENT IPUMPH	TOTAL PUMPH DENSITY	SCALPD PUMPH DENSITY	SEGMENT VOLUME USED	PERCENT TOTAL PUMPH	PERCENT ERROR IN TOTAL PCT	PERCENT SCALED PUMPH	TOTAL RAY LEGS IN SEG.	TOTAL IN SEG.
1	1	1	4.53+009	0.00+000	0.00+000	0.00+000	0.00+000	8.44	8.36	0.00	632	262
2	2	3	0.00+000	0.00+000	0.00+000	0.00+000	0.00+000	0.00	0.00	0.00	0	0
3	3	3	0.00+000	0.00+000	0.00+000	0.00+000	0.00+000	0.00	0.00	0.00	1755	216
4	4	3	0.00+000	0.00+000	0.00+000	0.00+000	0.00+000	0.00	0.00	0.00	11928	429
5	5	2	1.46+008	0.00+000	0.00+000	0.00+000	0.00+000	0.27	40.14	0.00	17	16
6	6	2	4.71+008	0.00+000	0.00+000	0.00+000	0.00+000	0.87	26.91	0.00	52	44
7	7	6	0.00+000	0.00+000	0.00+000	0.00+000	0.00+000	0.00	0.00	0.00	0	0
8	8	4	3.27+006	0.00+000	0.00+000	0.00+000	0.00+000	0.01	4.76	0.00	3057	441
9	9	6	0.00+000	0.00+000	0.00+000	0.00+000	0.00+000	0.00	0.00	0.00	0	0
10	10	6	0.00+000	0.00+000	0.00+000	0.00+000	0.00+000	0.00	0.00	0.00	0	0
11	11	6	0.00+000	0.00+000	0.00+000	0.00+000	0.00+000	0.00	0.00	0.00	0	0
12	12	6	0.00+000	0.00+000	0.00+000	0.00+000	0.00+000	0.00	0.00	0.00	0	0
13	13	6	0.00+000	0.00+000	0.00+000	0.00+000	0.00+000	0.00	0.00	0.00	0	0
14	14	6	0.00+000	0.00+000	0.00+000	0.00+000	0.00+000	0.00	0.00	0.00	0	0
15	15	6	0.00+000	0.00+000	0.00+000	0.00+000	0.00+000	0.00	0.00	0.00	0	0
16	16	6	0.00+000	0.00+000	0.00+000	0.00+000	0.00+000	0.00	0.00	0.00	0	0
17	17	6	5.13+008	2.90+008	2.09+009	1.18+009	2.46+001	0.95	9.83	13.89	709	204
18	18	6	3.08+008	1.78+008	1.26+009	7.25+008	2.45+001	0.57	9.84	8.51	622	206
19	19	6	5.35+008	3.03+008	2.17+009	1.23+009	2.46+001	0.99	10.28	14.49	748	200
20	20	6	3.69+008	2.14+008	1.51+009	8.74+008	2.45+001	0.69	9.84	10.24	648	204
21	21	6	5.45+008	3.07+008	2.22+009	1.28+009	2.46+001	1.01	10.41	14.71	796	204
22	22	6	3.64+008	2.10+008	1.49+009	8.57+008	2.45+001	0.68	10.24	10.05	702	208
23	23	6	6.00+008	3.63+008	2.60+009	1.48+009	2.46+001	1.19	9.63	17.37	814	209
24	24	6	3.90+008	2.24+008	1.59+009	9.15+008	2.45+001	0.72	9.23	10.73	729	209

SAMPLE PROBLEM #2 - TEST RUN OUTPUT

MATERIAL RESULTS									
MATERIAL TYPE NUMBER	TOTAL MATERIAL POWER	SCALED MATERIAL POWER	TOTAL POWER DENSITY	SCALED POWER DENSITY	MATERIAL VOLUME USED	PERCENT TOTAL POWER	PERCENT ERROR IN TOTAL PCT POWER	PERCENT SCALED POWER	TOTAL RAYS IN SEGMENTS OF MAT.
1	4.85+009	0.00+000	0.00+000	0.00+000	0.00+000	0.44	0.29	0.00	262
2	4.14+008	0.00+000	0.00+000	0.00+000	0.00+000	1.14	22.76	0.00	35
3	0.00+000	0.00+000	0.00+000	0.00+000	0.00+000	0.00	0.00	0.00	429
4	3.27+006	0.00+000	0.00+000	0.00+000	0.00+000	0.01	4.78	0.00	441
5	4.50+010	0.00+000	0.00+000	0.00+000	0.00+000	83.61	0.94	0.00	500
6	3.66+009	2.09+009	1.87+009	1.06+009	1.96+009	4.80	5.08	100.00	367
									18

SUMMARY RESULTS

TOTAL ABSORBED POWER = 9.3873+010

TOTAL SCALPD POWER = 2.0902+009

TOTAL SOURCE POWER = 9.3612+010

TOTAL NUMBER OF RAYS = 1093949679

FINAL RANDOM INTEGER = 1093949679

EXECUTION TIME = 1093.469 SECONDS

APPENDIX A

A.1 HELIX

A.1.1 Introduction

This Appendix describes the procedure that is followed for finding the intersections of a ray with a helical pipe. Two related procedures for determining whether a given point is inside or outside of a given helical pipe and for determining the normal to a helical pipe at a given point on its surface are also discussed.

A.1.1.1 Notation and Conventions

The region R of interest in which the helical pipe lies has a fixed coordinate system. An equation like $\vec{V} = [x, y, z]$ means that $\vec{V}$ is a vector* with components x, y , and z . Vectors will be used to represent points, line segments, displacements of points and directions within the region R . The length of $\vec{V}$ is denoted by $|\vec{V}|$. $|[x, y, z]| = (x^2 + y^2 + z^2)^{1/2}$. Curves will be represented in parametric form. For example, if $\vec{P} = [x, y, z]$ and $\vec{D} = [\alpha, \beta, \gamma]$ is a direction (in which case $\alpha^2 + \beta^2 + \gamma^2 = 1$), then $\vec{L}(s) = \vec{P} + s\vec{D} = [x, y, z] + s[\alpha, \beta, \gamma] = [x + \alpha s, y + \beta s, z + \gamma s]$ is the line through $\vec{P}$ in the direction of $\vec{D}$. More precisely, it is the point on the line that is a distance s from the point $\vec{P}$. Similarly $\vec{H}(\theta) = [x + \rho \cos \theta, y + \rho \sin \theta, z + \kappa \theta]$ is a helix with axis parallel to the z -axis and passing through the point $[x, y, z]$. (More properties of the helix will be given in the following sections, particularly, the next.)

*The notation used is essentially that of Louis Brand, Vector Analysis, Wiley & Sons (1957) except that he uses bold face type instead of the arrow to denote vectors.

Surfaces will be represented in two ways: by parametric formulas and by locus equations. For example, the plane passing through the three points $\vec{P}_0$, $\vec{P}_1$, and $\vec{P}_2$ is given by the parametric formula

$$\vec{r}(u,v) = \vec{P}_0 + u(\vec{P}_1 - \vec{P}_0) + v(\vec{P}_2 - \vec{P}_0) \quad (\text{A.1})$$

or the equation*

$$0 = (\vec{P} - \vec{P}_0) \cdot (\vec{P}_1 - \vec{P}_0) \times (\vec{P}_2 - \vec{P}_0) \quad (\text{A.2})$$

which is satisfied by just those points that lie on the plane and no others.

A.1.2 The Helix

A parametric formula for a helix was given in A.1.1.1. With the addition of a subscript h on some of the terms, it is the one that will be used consistently in all that follows, viz.

$$\vec{H}(\theta) = [x_h + c \cos \theta, y_h + c \sin \theta, z_h + \kappa \theta] \quad (\text{A.3})$$

The axis of the helix is parallel to the z -axis and passes through the point $\vec{P}_h = [x_h, y_h, z_h]$. The radius of the circular cylinder on which the helix is wrapped is c . The pitch of the helix is $2\pi\kappa$, the distance traveled in the z direction per revolution. The helix is right-handed if $\kappa > 0$. The parameter θ represents the angle of rotation required to move from the point $\vec{H}(0) = [x_h + c, y_h, z_h]$ to the point $\vec{H}(\theta)$.

* See Brand, §10 and 11 for definitions of scalar (dot) and vector (cross) products.

A.1.2.1 Tangent

The tangent* $\vec{t}(\theta)$ to the helix at the point $\vec{H}(\theta)$ is found by differentiation:

$$\begin{aligned}\vec{t}(\theta) &= \frac{d}{d\theta} \vec{H}(\theta) \\ &= \left[\frac{d}{d\theta}(x_h + \rho \cos \theta), \frac{d}{d\theta}(y_h + \rho \sin \theta), \frac{d}{d\theta}(z_h + \kappa \theta) \right] \quad (\text{A.4}) \\ &= [-\rho \sin \theta, \rho \cos \theta, \kappa] .\end{aligned}$$

The length of $\vec{t}(\theta)$ is the distance the point $\vec{H}(\theta)$ moves along the helix when θ increases by 1. The unit tangent $\hat{\vec{t}}(\theta) = \vec{t}(\theta)/(\rho^2 + \kappa^2)^{1/2}$ is the unit vector in the direction of the tangent. It is the direction of the line tangent to the helix at $\vec{H}(\theta)$, so the parametric form for the tangent line is $\vec{H}(\theta) + s\hat{\vec{t}}(\theta)$ where s , not θ , is the parameter.

A.1.2.2 Principal Normal

The principal normal** to a curve at any point is the direction from that point to the center of curvature. It is perpendicular to the tangent. It may be found by differentiating the tangent once again. Since $d\vec{t}(\theta)/d\theta = [-\rho \cos \theta, -\rho \sin \theta, 0]$, which has length ρ , the principal normal is $\hat{\vec{N}}(\theta) = [-\cos \theta, -\sin \theta, 0]$. It is always perpendicular to the Z-axis as well as to $\vec{t}(\theta)$ and it points from $\vec{H}(\theta)$ toward the axis of the helix.

* See Brand, §27.

** Brand, §29.

The radius of curvature of the helix is given by

$$\rho_c = \frac{\left| \frac{d\vec{H}(\theta)}{d\theta} \right|^2}{\left| \frac{d^2\vec{H}(\theta)}{d\theta^2} \right|} = \rho + \kappa^2/\rho, \quad (\text{A.5})$$

which can be worked out by straightforward algebra and differentiation.

The binormal of a curve is the direction perpendicular to the tangent and the principal normal which may be determined by taking the vector product of those directions. In our case, the binormal is given by

$$\begin{aligned} \vec{M}(\theta) &= \vec{T}(\theta) \times \vec{N}(\theta) \\ &= [\kappa \sin \theta, -\kappa \cos \theta, \rho] / (\rho^2 + \kappa^2)^{1/2}. \end{aligned} \quad (\text{A.6})$$

A.1.3 The Helical Pipe

The surface that we must deal with is, loosely speaking, a circular cylinder whose axis is a helix which is to say that a plane perpendicular to the helix at some point $\vec{P}_h$ cuts the surface in the immediate vicinity of $\vec{P}_h$ in a circle with center $\vec{P}_h$ and radius a . A parametric formula for the surface can be given as

$$\begin{aligned} \vec{Q}(\theta, \phi) &= \vec{H}(\theta) + a \cos \phi \vec{N}(\theta) + a \sin \phi \vec{M}(\theta) \\ &= \left[x_h + \rho \cos \theta - a \cos \phi \cos \theta + \kappa a \sin \phi \sin \theta / \right. \\ &\quad \left. (\rho^2 + \kappa^2)^{1/2}, y_h + \rho \sin \theta - a \cos \phi \sin \theta - \kappa a \sin \phi \cos \theta / \right. \\ &\quad \left. (\rho^2 + \kappa^2)^{1/2}, z_h + \kappa \theta + \rho a \sin \phi / (\rho^2 + \kappa^2)^{1/2} \right]. \end{aligned}$$

Notice that when $c=0$ the formula reduces to one for a circular cylinder of radius a and if $\kappa=0$ it reduces to one for a torus.

A.1.4 The Ray and the Helical Pipe

The origin of the ray will always be denoted by $\vec{P}_0 = [x_0, y_0, z_0]$ and its direction by $\vec{D} = [a, \hat{a}, \gamma]$ where, of course, $a^2 + \hat{a}^2 + \gamma^2 = 1$. We will always use $\vec{P}(s) = \vec{P}_0 + s\vec{D} = [x_0 + sa, y_0 + s\hat{a}, z_0 + s\gamma]$ as the parametric formula for the ray where s denotes the distance from $\vec{P}_0$ to $\vec{P}(s)$. Since $\vec{P}_0$ is the origin of the ray, $s > 0$ for points of the ray.

Let b denote the distance between the axis of the helix and line of the ray and let $\vec{P}(s_b)$ (it is possible that $s_b < 0$) be the point which is closest to the axis. Since $\vec{D}$ is perpendicular to the shortest line joining $\vec{P}(s_b)$ to the axis, we must have $a(x_h - x_0 - s_b a) + \hat{a}(y_h - y_0 - s_b \hat{a}) = 0$. Therefore

$$s_b = [a(x_h - x_0) + \hat{a}(y_h - y_0)] / (a^2 + \hat{a}^2) \quad (\text{A.7})$$

and

$$b^2 = (x_h - x_0 - s_b a)^2 + (y_h - y_0 - s_b \hat{a})^2 \quad (\text{A.8})$$

As we soon will see, it is important to know whether the ray passes the axis on the right or left. The ray passes the axis on the right if and only if the three vectors $\vec{P}_0 - \vec{P}_h$, $\vec{D}$ and $\vec{k}$ form a basis for a right-handed coordinate system which, in turn, is true if and only if the triple product $\sigma = (\vec{P}_0 - \vec{P}_h) \cdot (\vec{D} \times \vec{k})$ is positive. But

$$\vec{D} \times \vec{k} = [a, \hat{a}, \gamma] \times [0, 0, 1] = [\hat{a}, -a, 0] \quad , \quad (\text{A.9})$$

and hence

$$\begin{aligned}\sigma &= [x_0 - x_h, y_0 - y_h, z_0 - z_h] \cdot [\xi, -\alpha, 0] \\ &= \xi(x_0 - x_h) - \alpha(y_0 - y_h)\end{aligned}\quad (\text{A.10})$$

The magnitude, as well as the sign, of σ is significant. Notice that $\vec{D} \times \vec{k}$, being perpendicular to both $\vec{D}$ and $\vec{k}$, is parallel to the shortest segment joining $\vec{P}_h + s\vec{k}$ (the axis) to $\vec{P}_0 + s\vec{D}$ (the ray). Hence, b must be the length of the projection of $\vec{P}_0 - \vec{P}_h$ on $\vec{D} \times \vec{k}$,

$$b = \frac{|(\vec{P}_0 - \vec{P}_h) \cdot \vec{D} \times \vec{k}|}{|\vec{D} \times \vec{k}|} = \frac{|c|}{(\alpha^2 + \xi^2)^{\frac{1}{2}}}$$

A.1.5 The Turns of the Helix

When the ray and the axis of the helix are skew, there is a unique plane Π_r containing the ray which is parallel to the axis. Consider, then, the half-plane extending away from the ray in a direction perpendicular to Π_r and having the axis as boundary. This half-plane cuts the helix into a series of individual loops, that we will call turns, whose endpoints are as far as possible from the plane Π_r . The turns are arcs of the helix $\vec{H}(\varphi)$ corresponding to intervals $\varphi_n \leq \varphi \leq \varphi_{n+1}$ where $\varphi_n = \varphi_0 + 2n\pi$, and

$$\sin \varphi_0 = (y_h - y_0 - \xi s_b)/b \quad (\text{A.12})$$

$$\cos \varphi_0 = (x_h - x_0 - \alpha s_b)/b \quad (\text{A.13})$$

The vector $[\cos \psi_0, \sin \psi_0, 0]$ is parallel to the shortest segment joining the axis to the ray, which is perpendicular to $\vec{D}$. Therefore

$$\alpha \cos \psi_0 + \beta \sin \psi_0 = 0 \quad (\text{A.14})$$

A.1.6 The Sequence of the Turns

The basis of the plan that is used for finding the first intersection is a procedure for calculating the intersections of a line with one turn of the helical pipe. For the sake of efficiency, it is desirable to know the order in which the turns are encountered by a point as it travels along the ray. The present section is devoted to the description of a method for plotting rays that is well-designed to illustrate how they proceed from turn to turn.

To see how the plot may be generated, consider a half-plane that is bounded on its edge by the axis of the helix and which cuts the ray at some point and let a curve be traced out on the half-plane by the cut-point as the half-plane rotates about its edge and the point travels along the ray.

Let $r(s)$ be the distance from the axis to $\vec{P}(s)$. Then,

$$\begin{aligned} r(s) &= \left((x_0 + \alpha s - x_h)^2 + (y_0 + \beta s - y_h)^2 \right)^{\frac{1}{2}} \\ &= \left(b^2 + \mu^2 (s - s_b)^2 \right)^{\frac{1}{2}} \end{aligned} \quad (\text{A.15})$$

Let $w(s)$ be the distance from $\vec{P}(s)$ to the half-plane that passes through $\vec{P}(s_b)$.

Then

$$w(s) = \mu(s-s_b) \quad (\text{A.16})$$

where

$$\mu = (\alpha^2 + \beta^2)^{\frac{1}{2}} \text{sgn}(\sigma) \quad (\text{A.17})$$

i.e., μ is the length of the projection of $\vec{D}$ on the plane $z=0$ and it is positive if the ray passes the axis on the right as it does in Fig. A.2. It follows from Eqs. (A.10) and (A.11) that

$$\mu = \sigma/b \quad (\text{A.18})$$

Finally, let

$$\phi(s) = \arctan w(s)/b \quad (\text{A.19})$$

and

$$z^*(s) = z_0 + \gamma s - \kappa \phi(s). \quad (\text{A.20})$$

The significance of $\phi(s)$ is shown in Fig. A.2. It is the angle that the plane through $P(s)$ makes with the plane through $P(s_b)$. The sign of μ in Eq. (A.17) was selected so that ϕ and θ are both measured in the right-handed or counterclockwise sense. The purpose of $\phi(s)$ is to provide the term $\kappa \phi(s)$ in Eq. (A.20) which compensates for the rise (or fall) of the helix as the point on the ray passes the axis. Thus $z^*(s)$ represents the height of $\vec{P}(s)$ above a level that rises with the helix. The desired plot, then, is generated by a point having $r(s)$ as ordinate and $z^*(s)$ as abscissa. On this plot, the turns of the helix appear as fixed points with coordinates (r_n, z_n^*) where $r_n = r$ and $z_n^* = z_h + \kappa(-\phi_0 + (2n+1)\pi)$.

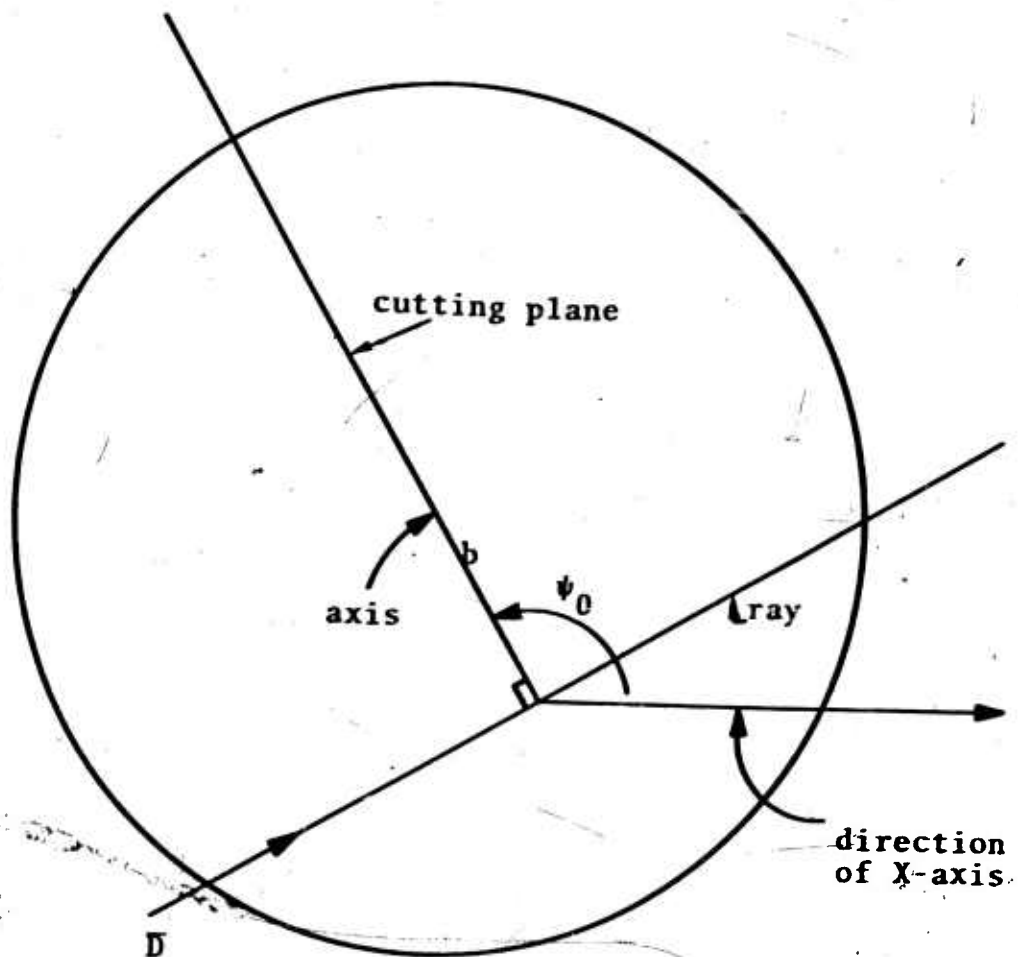


Figure A.1 A view of the helix and ray looking down the axis of the helix.

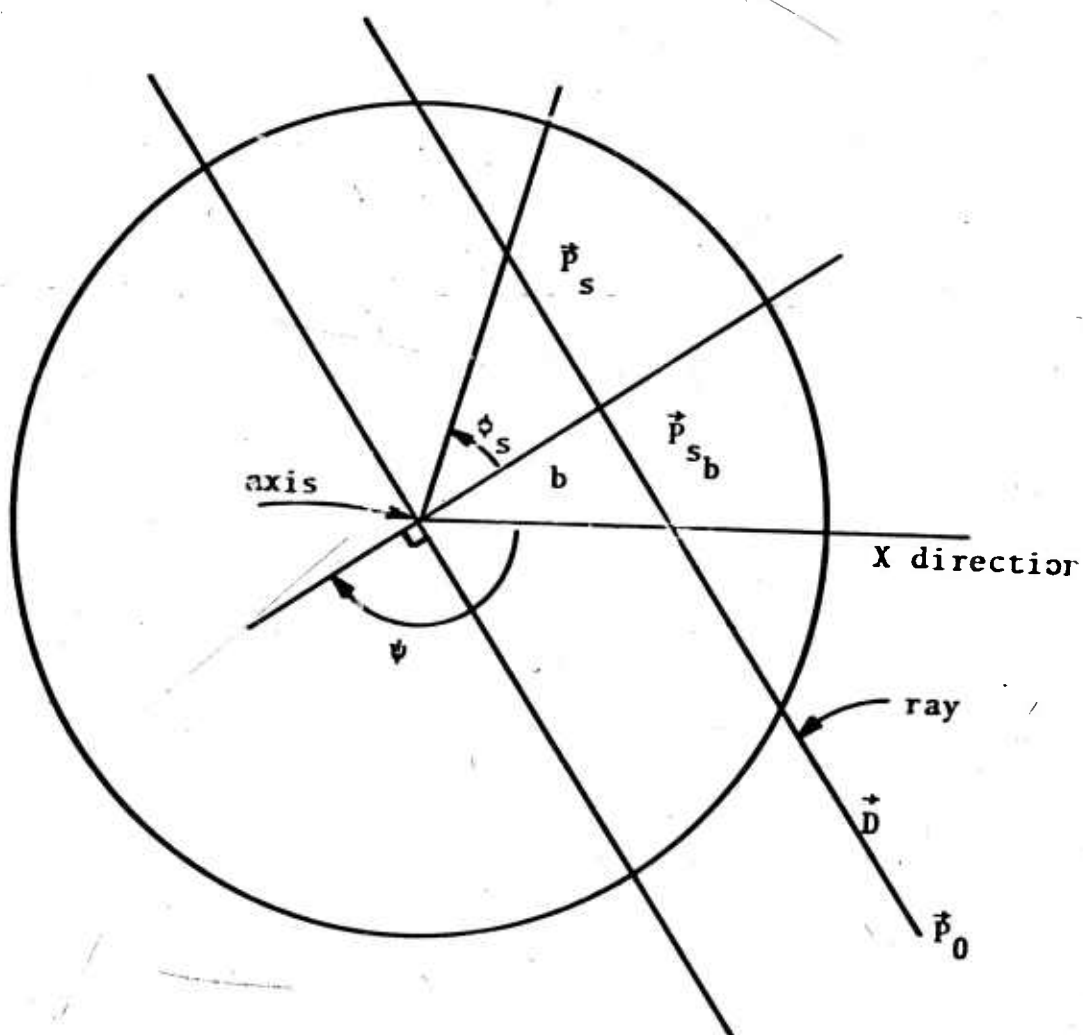


Figure A.2 Another top view of the helix and ray.

The equation of the trace of the ray is obtained by eliminating s from Eqs. (A.15) and (A.20). When $\kappa=0$ the result is

$$\left(\frac{z^* - z_b}{\gamma b}\right)^2 - \left(\frac{r}{\mu b}\right)^2 = 1 \quad (\text{A.21})$$

where $z_b = z_0 + s_b$, so that the trace is a hyperbola whose center has coordinates $z^* = z_b$ and $r = 0$. The effect of $\kappa \neq 0$ depends on the relative values of $\kappa\mu/\gamma$ and b . If $\kappa\mu/\gamma < 0$ then the ray goes counter to the helix and the hyperbola is stretched by the amount $\pi\kappa$, the distance the helix rises during a half-turn. In this case, the ray meets the turns of the helical pipe in their natural sequence.

If $\kappa\mu/\gamma > 0$ then the ray and the helix both rise or both fall and if the magnitude of κ is large enough so that $\kappa\mu/\gamma > b$ then helix rises or falls faster than the ray while the ray is close to the axis; the hyperbola is compressed by $\pi\kappa$ and a loop forms.

To verify this fact, observe that

$$\begin{aligned} dz^*(s)/ds &= \gamma - \frac{\kappa\mu b}{\mu^2(s-s_b)^2 + b^2} \\ &= \gamma \left\{ 1 - \frac{\frac{\kappa\mu}{\gamma b}}{\left(\frac{\mu(s-s_b)}{b}\right)^2 + 1} \right\} \end{aligned} \quad (\text{A.22})$$

which changes sign if $\kappa\mu/\gamma > b$ at $s = s_b + \lambda$ where

$$\lambda = \sqrt{\frac{b}{\mu^2} \left(\frac{\kappa\mu}{\gamma} - b \right)} \quad (\text{A.23})$$

When $0 < \kappa\mu/\gamma < b$, the hyperbola is still compressed by $-\kappa$ but not enough to form a loop.

Figure A.3 shows examples of the three types of traces that rays may have. The traces of the turns of the helical pipe are more or less elliptical. It is seen that when $\kappa\mu/\gamma < b$, the ray intersects the turns consecutively, each one at most twice. When $b < \kappa\mu/\gamma$, all turns except the two nearest $z^* = z_b$ are still passed in natural sequence and each is intersected at most twice. On the other hand, the two exceptional turns (i.e., the two with z^* nearest z_b), may both be intersected four times by a ray whose trace resembles the one pictured in Figure A.3.

A.1.7 Numbering the Turns

The strategy that is used to find the first intersection of a ray with a helical pipe is to determine the sequence in which the ray approaches successive turns and then to examine the turns in that sequence to find the first that is actually intersected. In implementing this strategy, the turns are assigned ordinal numbers as follows.

Let

$$v = 2\pi \operatorname{sgn}(\kappa\gamma) , \quad \psi_n = \psi_0 + nv$$

where

$$\cos \psi_0 = (y_h - y_0 - \kappa s_b)/b$$

and

$$\sin \psi_0 = (y_h - s_0 - \kappa s_b)/b$$

Then, turn n is the turn for which $\psi_n < \psi < \psi_{n+1}$.

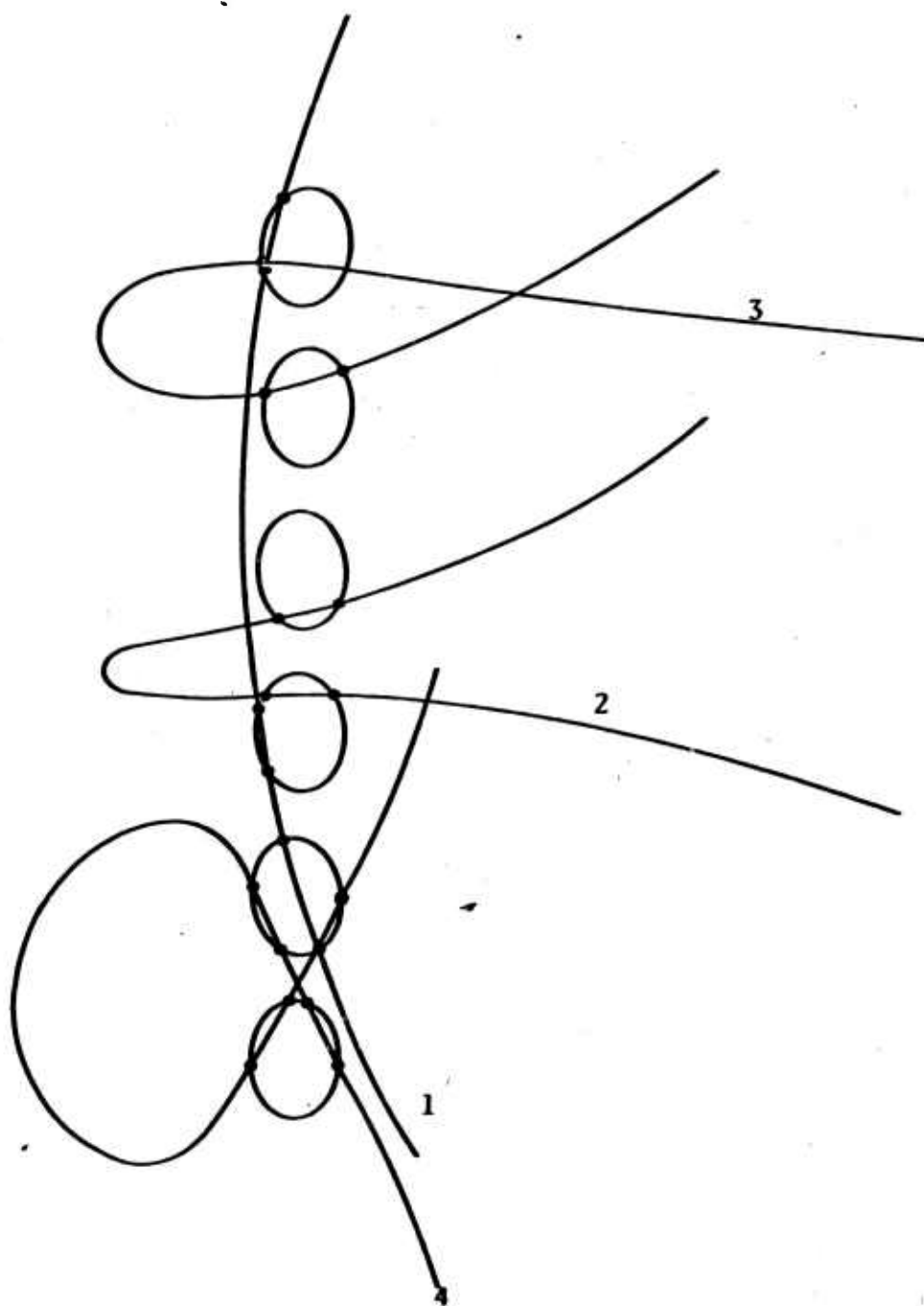


Figure A.3 Traces of typical rays: 1) $\kappa\mu/\gamma < 0$, 2) $0 < \kappa\mu/\gamma < b$, 3) $b < \kappa\mu/\gamma$, 4) ray with 8 intersections

This definition of the number of a turn differs from that of A1.5 in that here we require that the sign of v agree with that of $\kappa\gamma$. This has the pleasant effect of insuring that n will increase as s increases (except for that segment of the ray corresponding to a loop in the trace). In order to give precision to the above statement, the function $n(s)$ is defined to be the number of the loop that is closest to the point $\vec{P}(s) = \vec{P}_0 + s\vec{D}$. The value of n which is assigned to points midway between turns is unimportant since such points are necessarily exterior to the helical pipe.

A.1.8 Seven Special Points on a Ray

There are certain points on a ray that are special enough to receive specific names. They are listed below together with a formula for the value s at which the point is attained. When the value of s for some special point on a particular ray is negative or complex, it means that that point does not occur. The points are

P_0 , the origin of the ray. $s_0 = 0$

P_1 , the point where the ray enters the cylinder that encloses the pipe. $s_1 = s_b - \sqrt{(\rho+a)^2 - b^2}$. If $b > \rho+a$, the ray does not penetrate the cylinder.

P_2 , the point where the ray enters the cylinder on which the pipe is wrapped.
 $s_2 = s_b - \sqrt{(\rho-a)^2 + b^2}$. If $b > \rho-a$, the ray does not penetrate the inner cylinder.

P_3 , the point where the ray emerges from the inner cylinder. $s_3 = s_b + \sqrt{(\rho-a)^2 + b^2}$.

P_4 , the point where the ray emerges from the outer cylinder. $s_4 = s_b + \sqrt{(\rho+a)^2 + b^2}$.

P_5 , the point where the ray reaches the bottom of the loop in its trace. $s_5 = s_b + \lambda$. There is no loop if λ is complex.

P_6 , the point where the ray reaches the top of the loop in its trace. $s_6 = s_b - \lambda$.

The number of the turn nearest P_k is $n_k = n(s_k)$. Note that

$$s_1 \leq s_2 \leq s_3 \leq s_4, \quad s_5 \geq s_6$$

$$n_1 \leq n_2 \leq n_3 \leq n_4, \quad n_5 \leq n_6$$

whenever the indicated quantities are real.

The subroutine INTHEL which we consider next is not exactly simple but it is straightforward. The complications that arise have to do with the singularities that arise when one or more of the quantities γ , μ and b become too small. It will be observed that when $n_6 > n_5$ these two turns are treated simultaneously because they can generate interlocking intersections.

A.1.9 The Distance from a Point to a Line

The distance from the point $\vec{P} = [x, y, z]$ to the line $\vec{L}(s) = \vec{P}_0 + s\vec{D} = [x_0 + \alpha s, y_0 + \beta s, z_0 + \gamma s]$ is attained where $\vec{D} \cdot (\vec{L}(s) - \vec{P}) = 0$, i.e., the segment that joining $\vec{P}$ to the closest point on $\vec{L}$ is perpendicular to the line. If $s = s_m$ at that point, then

$$s_m = \frac{\vec{D} \cdot (\vec{P}_0 - \vec{P})}{\vec{D} \cdot \vec{D}}$$

$$= \alpha(x_0 - x) + \beta(y_0 - y) + \gamma(z_0 - z)$$

and the distance from the point to the line is

$$d = \left[(x_0 + \alpha s_m - x)^2 + (y_0 + \beta s_m - y)^2 + (z_0 + \gamma s_m - z)^2 \right]^{\frac{1}{2}} \quad (\text{A.25})$$

A.1.10 The Distance from a Point to a Turn

It is often necessary to determine the distance from a given point to a given turn of the helix. The distance from $\vec{P} = [x, y, z]$ to $\vec{H}(\theta)$ is easily verified to be $(2G)^{\frac{1}{2}}$ where

$$G = \frac{1}{2}(x - x_h - \rho \cos \theta)^2 + \frac{1}{2}(y - y_h - \rho \sin \theta)^2 + \frac{1}{2}(z - z_h - \kappa \theta)^2 \quad (\text{A.26})$$

A change of θ will reduce the distance if it will reduce G . The distance will be a minimum when $dG/d\theta = 0$ and $d^2G/d\theta^2 > 0$. The first two derivatives of G turn out to be

$$\begin{aligned} \frac{dG}{d\theta} &= \rho \sin \theta (x - x_h - \rho \cos \theta) - \rho \cos \theta (y - y_h - \rho \sin \theta) - \kappa (z - z_h - \kappa \theta) \\ &= \rho \sin \theta (x - s_h) - \rho \cos \theta (y - y_h) - \kappa (z - z_h - \kappa \theta) \end{aligned} \quad (\text{A.27})$$

$$\frac{d^2G}{d\theta^2} = \rho \cos \theta (x - x_h) + \rho \sin \theta (y - y_h) + \kappa^2$$

The problem reduces to that of finding the nearest root of $dG/d\theta$. A given estimate of the θ of the closest point is improved using Newton's method which is to set

$$\theta_{\text{new}} = \theta - \frac{\frac{dG}{d\theta}}{\frac{d^2G}{d\theta^2}} \quad (\text{A.28})$$

It is well-known that Newton's method can fail and the present application is no exception. The trouble arises when $d^2G/d\theta^2 < 0$ which usually causes a maximum rather than a minimum.

to be approached. The problem is overcome by setting $\theta_{\text{new}} = \theta - \text{sgn}(dG/d\theta)$ whenever $d^2G/d\theta^2 < 0$, i.e., by moving one radian toward the minimum.

The question of when to stop this iterative approach to the minimum cannot be answered out of context and therefore we leave it to be discussed when it occurs in a specific application.

A.1.11 The Inside of the Pipe

It is sometimes necessary to know if a given point $\vec{P} = [x, y, z]$ is inside or outside the pipe. The point $\vec{P}$ is inside or outside the pipe. The point $\vec{P}$ is inside if its distance from the helix is less than a . This assertion follows from the definition of the surface of the pipe which, according to the parametric formula given in A.1.3, consists of those points lying at a distance a from the helix. Thus the problem reduces to finding the distance from $\vec{P}$ to the helix.

The first step in the procedure for finding the distance is to find the values of θ for those points where the helix intersects the half-plane that is bounded by the axis and contains the point $\vec{P}$. These θ -values are

$$\theta_n = \text{ATAN2}(y - y_h, x - x_h) + 2\pi n \quad . \quad (\text{A.29})$$

The value of n which minimizes $|\vec{P} - H(\theta_n)|$ is the one that minimizes $|z - z_h - \kappa\theta|$ which is, in turn

$$n = \left[\frac{z - z_h - \kappa\theta_0}{2\pi\kappa} + \frac{1}{2} \right] \quad (\text{A.30})$$

where $[x]$ denotes the greatest integer that is not greater than x . Caution: in FORTRAN, $\text{INT}(x) = [x] + 1$ when $x < 0$.

Using $\vec{H}(\theta_n)$ as a starting point, a procedure like that of A1.10 is used to search for the closest point with the refinement of a quadratic term. Thus

$$\frac{d^3 G}{d\theta^3} = -\rho \sin \theta (x - x_h) + \rho \cos \theta (y - y_h) \quad (\text{A.31})$$

and we write

$$0 = \frac{dG}{d\theta} + (\theta_{\text{new}} - \theta) \frac{d^2 G}{d\theta^2} + \frac{(\theta_{\text{new}} - \theta)^2}{2} \frac{d^3 G}{d\theta^3} \quad (\text{A.32})$$

Writing G' , G'' , G''' for the successive derivatives of G and δ for $\theta_{\text{new}} - \theta$, we get

$$\begin{aligned} \delta &= \frac{-G'' + ((G'')^2 - 2G'G''')^{\frac{1}{2}}}{2G'''} \\ &= \frac{-G'}{((G'')^2 - 2G'G''')^{\frac{1}{2}}} \end{aligned} \quad (\text{A.33})$$

The second form, which is obtained from the first by rationalizing the numerator, is preferred, especially when G''' is small.

The iteration may well be stopped when $|S| < .01$ because the error will be less than $a(1 - \cos \delta) < .5 \times 10^{-4} a$.

The method fails for points on the axis but these points are outside the pipe because $a < \rho$.

A.1.12 The Distance from the Ray to a Turn

A ray approaches every turn at least once and never more than twice. If it approaches turn n at two points, θ_a and θ_b , then θ_a and θ_b will be separated by $\psi_n + \frac{1}{2}\nu$, the mid-point of the turn

The procedure for finding where and how often a particular turn approaches a ray is a combination of techniques described above. First, the point $\theta_1 = \psi_n + \frac{1}{4}\nu$ is selected. The equation (A.24) is used to find the point on the ray closest to $\vec{H}(\theta)$. Let its parameter be called $s(\theta_1)$. Next the method discussed in A1.10 is applied to find a θ_2 such that $\vec{H}(\theta_2)$ is closer to $\vec{P}(s(\theta_1))$ than $\vec{H}(\theta_1)$ and then $s(\theta_2)$ is calculated. The process is repeated until a convergence criterion is satisfied. When the procedure was first designed, the plan was to apply just one step of the algorithm described in A1.10 per iteration but certain test cases showed signs of instability. The dynamics of the instability are not understood but it was found that repeating the algorithm until $|\theta_{\text{new}} - \theta| < .01$ was sufficient to damp the oscillations.

The whole procedure is repeated with $\theta_1 = \psi_n + \frac{3}{4}\nu$. When the ray approaches the turn on both sides of $\psi_n + \frac{1}{2}\nu$, the points of closest approach are found with an accuracy of better than .01 in θ . If the ray approaches the turn on only one side, the procedure on both sides will usually converge to that point but occasionally, when θ_1 is set to the "wrong" side, it will drift around the back of the helix to an adjacent turn.

A.1.13 Finding the Intersections

If a ray is found to be within a distance a of a turn at a point where it is closest, it can be inferred that it intersects that turn. Two procedures are used for localizing intersections, one is used if $|\vec{D} \cdot \vec{T}(\theta)| < 0.3$, else the other is used.

In case $|\vec{D} \cdot \vec{T}(\theta)| < 0.3$, a first approximation to the location of the intersections is made by calculating the intersection of the ray with a circular cylinder of radius a and axis through $\vec{H}(\theta)$ in the direction $\vec{T}(\theta)$. If c is the distance from $\vec{H}(\theta)$ to the ray then the intersection of the ray with the cylinder will be at $\varepsilon(\theta) + \Delta s$ where

$$\Delta s = \left(\frac{a^2 - c^2}{1 - (\vec{D} \cdot \vec{T})^2} \right)^{1/2} \quad (\text{A.34})$$

The first approximations to the intersections of the ray with the turn of pipe are further refined. Let $\vec{P}_a = [x_a, y_a, z_a]$ denote one of the first approximants. The procedure begins with the application of A1.10 to determine the θ corresponding to the point on the helix that is closest to the point P_a . Then the closer intersection of the ray to the tangent cylinder at the new θ is determined. The equations used are discussed below. First, the equation of the cylinder is

$$|\vec{P} - \vec{H}(\theta)|^2 = a^2 + (\vec{P} - \vec{H}(\theta)) \cdot \vec{T}(\theta)^2 \quad (\text{A.35})$$

Next, we substitute $\vec{P}_a + u\vec{D}$ for $\vec{P}$ in Eq. (A.35) and solve for u , which yields

$$u^2(1 - (\vec{D} \cdot \vec{T})^2) + 2u\vec{D} \cdot (\vec{P}_a - \vec{H}) + |\vec{P}_a - \vec{H}|^2 - a^2 = 0 \quad (\text{A.36})$$

or

$$u = \frac{a^2 - c^2}{b \left(1 + \left(1 + d(a^2 - c^2)/b^2 \right)^{1/2} \right)} \quad (\text{A.37})$$

where

$$c^2 = |\vec{P}_0 - \vec{H}|^2, \quad d = 1 - (\vec{D} \cdot \vec{T})^2, \quad b = \vec{D} \cdot (\vec{P}_a - \vec{H})$$

and $\vec{T} \cdot (\vec{P}_0 - \vec{H}) = 0$ (because the segment joining $\vec{P}_0$ to the helix at $\vec{H}$ is perpendicular to the tangent $\vec{T}$).

Then $s+u$ is substituted for s and θ is further refined. If the change in θ is less than .01, the current value of s is accepted, else another step is taken.

When $|\vec{D} \cdot \vec{T}| > .7$, the approximation in Eq. (A.35) will often yield such poor accuracy that the refinement technique fails. For that reason, an entirely different method is used when $|\vec{D} \cdot \vec{T}|$ is large. The criterion for the alternate method is $|\vec{D} \cdot \vec{T}| > .3$.

In the beginning of this section, $s(\theta)$ was defined to be the value of the parameter of the point on the ray closest to $\vec{H}(\theta)$. We now define a new point on the ray to be associated with θ ; and that is the point where the ray intersects the plane $\Pi(\theta)$ that is normal to $\vec{T}(\theta)$ and is cut by the helix at $\vec{H}(\theta)$. The equation for $\Pi(\theta)$

$$\vec{P} \cdot \vec{T}(\theta) = \vec{H}(\theta) \cdot \vec{T}(\theta) \quad (\text{A.38})$$

substituting $\vec{P}_0 + s\vec{D}$ for $\vec{P}$ in Eq. (A.38) yields an equation in s , the parameter of the desired point:

$$\begin{aligned} & (x_0 + as)(-\rho \sin \theta) + (y_0 + bs)\rho \cos \theta + (z_0 + \gamma s)\kappa \\ &= (x_h + \rho \cos \theta)(-\rho \sin \theta) + (y_h + \rho \sin \theta)\rho \cos \theta + (z_h + \kappa \theta)\kappa \end{aligned}$$

Denoting the solution by $s_p(\theta)$, we get

$$s_p(\theta) = \frac{(x_h - x_0)(-\rho \sin \theta) + (y_h - y_0)\rho \sin \theta + (z_h - z_0 + \kappa \theta)\kappa}{-\rho \sin \theta + b\rho \cos \theta + \gamma \kappa} \quad (\text{A.39})$$

Note that the denominator of the RHS of Eq. (A.39) is $(\rho^2 + \kappa^2)^{1/2} \vec{D} \cdot \vec{T}$, making $s_p(\theta)$ large when $\vec{D} \cdot \vec{T}$ is small. The important thing about the point $\vec{P}(s_p(\theta))$ is that $\vec{H}(\theta)$ is locally the closest point on the helix. Therefore, $\vec{P}(s_p(\theta))$ is a point of intersection if the distance $|\vec{P}(s_p(\theta)) - \vec{H}(\theta)| = a$.

Let $B(\theta)$ be defined as

$$\begin{aligned} B(\theta) &= \frac{1}{2} (x_0 + as_p(\theta) - x_h - \rho \cos \theta)^2 \\ &\quad + \frac{1}{2} (y_0 + bs_p(\theta) - y_h - \rho \sin \theta)^2 \\ &\quad + \frac{1}{2} (z_0 + \gamma s_p(\theta) - z_h - \kappa \theta)^2, \end{aligned} \quad (\text{A.40})$$

or, in vector notation, $B(\theta) = \frac{1}{2} |\vec{P}(s_p(\theta)) - \vec{H}(\theta)|^2$ or, in words, half the square of the distance from $\vec{P}(s_p(\theta))$ to $\vec{H}(\theta)$. Thus the problem of finding intersections reduces to that of solving the equation

$$B(\theta) = \frac{a^2}{2} \quad (\text{A.41})$$

When $|\vec{D} \cdot \vec{r}(\theta)| \ll a$, Eq. (A.41) is hard to solve because $B(\theta)$ becomes very large in the vicinity of the root and consequently the cylinder approximation is employed in such cases.

A.1.14 Finding the Roots of $B(\theta) = a^2/2$

The process of finding the roots of Eq. (A.41) involves $dB/d\theta$ which we denote by B' , as well as B itself. The formula B' follows from Eq. (A.);

$$\begin{aligned} B'(\theta) = & (x_0 + \alpha s_p(\theta) - x_h - \rho \cos \theta) (\alpha s'_p(\theta) + \rho \sin \theta) \\ & + (y_0 + \beta s_p(\theta) - y_h - \rho \sin \theta) (\beta s'_p(\theta) - \rho \cos \theta) \\ & + (z_0 + \gamma s_p(\theta) - z_h - \kappa \theta) (\gamma s'_p(\theta) - \kappa) \end{aligned} \quad (A.42)$$

where

$$s'_p(\theta) = (DX' - ND')/D^2 \quad (A.43)$$

$$\begin{aligned} N = & (x_h - x_0)(-\rho \sin \theta) + (y_h - y_0) \rho \cos \theta \\ & + (z_h - z_0 + \kappa \theta) \kappa \end{aligned} \quad (A.44)$$

$$N' = (x_h - x_0)(-\rho \cos \theta) + (y_h - y_0)(-\rho \sin \theta) + \kappa^2 \quad (A.45)$$

$$D = -\alpha \rho \sin \theta + \beta \rho \cos \theta + \gamma \kappa \quad (A.46)$$

$$D' = -\alpha \rho \cos \theta - \beta \rho \sin \theta \quad (A.47)$$

The procedure is to examine $B(\theta)$ and $B'(\theta)$ at steps of about $\pi/4$ looking for intervals that might contain one or more roots. When a change in the sign of $B(\theta) - a^2/2$ is observed between the endpoints of an interval, a rather ordinary root

finder is evoked to find the location of the location of the intersection to the desired accuracy. But a sign change in $B(\theta) - a^2/2$ will occur only when a single root lies within the interval. Two indicators are used to detect the occurrence of double roots. The first is a sign change in B' . If B' changes sign as an interval is traversed, then an extremum in B must occur within the interval. The extremum is ignored if it is in the wrong direction, e.g., if it is a maximum and B is greater than $a^2/2$ at the end-points. But if the extremum is in the right direction, a linear interpolation of B' is applied to provide an estimate of θ at the extreme point. If B is on the other side of $a^2/2$ at the estimated extreme point, then two roots are bracketed and the root-finder is called forth for each. Otherwise, the reduction of $|B - a^2/2|$ is computed and if it is reduced by more than $1/2$, the interpolation is repeated, if not, the interval is abandoned.

The second indicator of a double root is used to detect intervals in which a double sign change of B' has occurred. The quantity that is tested is

$$c = \frac{B(\theta_1) - B(\theta_2)}{(\theta_1 - \theta_2)(B'(\theta_1) + B'(\theta_2))} \quad (\text{A.48})$$

where θ_1 and θ_2 are the values of θ at the end points of the interval. Thus, c is the ratio of the difference quotient of B to twice the average of the derivatives at the endpoints. For a quadratic, $c = 1/2$. If $c < 1/4$, the interval is considered to be too large and it is subdivided. The subintervals are then processed in the same way as full-sized intervals.

In the above, it has been assumed that no interval can contain more than two roots of $B - a^2/2$. To be sure that such is the case, each turn is divided in the middle into two half-turns which are treated separately. Each half-turn contains

at most two roots and at most one minimum of B' . The latter fact is also used to reduce the number of branches at a few points in the program.

The techniques described above would occasionally fail in intervals containing a zero in the denominator of the RHS of Eq. (A.39) for $s_p(\theta)$. The point to be avoided is found by solving the equation $D(\theta) = 0$ where D is defined by equation (A.46). The solution is simplified by substituting $\theta_0 + \zeta$ for θ where $\theta_0 = \psi_n + \frac{1}{2}\nu$. Then

$$\begin{aligned} D &= -\alpha \rho \sin(\theta_0 + \zeta) + \beta \rho \cos(\theta_0 + \zeta) + \gamma \kappa \\ &= -\alpha \rho \sin \theta_0 \cos \zeta - \alpha \rho \cos \theta_0 \sin \zeta \\ &\quad + \beta \rho \cos \theta_0 \cos \zeta - \beta \rho \sin \theta_0 \sin \zeta + \gamma \kappa \end{aligned}$$

Now $\frac{1}{2}\nu = \frac{\pi}{2}$ so $\cos \theta_0 = -\cos \psi_n = -\cos \psi_0$ and $\sin \theta_0 = -\sin \psi_n = -\sin \psi_0$. It then follows from Eq. (A.14) that $\alpha \cos \theta_0 + \beta \sin \theta_0 = 0$. Therefore, when $D = 0$

$$\cos \zeta = \frac{\gamma \kappa}{\rho(\alpha \sin \theta_0 - \beta \cos \theta_0)} = -\frac{\gamma \kappa}{\rho \mu} \quad (\text{A.49})$$

If $|\gamma \kappa / \rho \mu| > 1$, then $\vec{f}(\theta)$ is never perpendicular to $\vec{D}$, but otherwise, $\vec{D} \cdot \vec{f}(\theta) = 0$ for $\theta = \theta_0 + \zeta$. At those two points, $s_p(\theta)$ and $B(\theta)$ become infinite and the plane $\Pi(\theta)$ defined by Eq. (A.38) becomes parallel to the ray. To determine which side of $\Pi(\theta_0 + \zeta)$ contains the ray, we calculate $d_1 \equiv \vec{f}(\theta_0 + \zeta) \cdot (\vec{P}_0 - \vec{H}(\theta_0 + \zeta))$ which is the distance from the plane to the ray except that d_1 is positive or negative according as the ray is or is not on the side of $\Pi(\theta_0 + \zeta)$ to which $\vec{f}$ points.

The next question is how close to $\theta_0 + \zeta$ can the intersection (if any) come. The following approximation will be adequate for our purposes. Recall that the radius of curvature of the helix is $\rho + \kappa^2/\rho$. The intersection might be as far as $\rho + a$ from the axis or $\rho + a + \kappa^2/\rho$ from the center of curvature which would minimize the distance from $\theta_0 + \zeta$ to the intersection. In that case, if the plane moves a distance d_1 to go from $\theta_0 + \zeta$ to the intersection, θ must change by more than

$$\theta_{\min} = \frac{\rho + \kappa^2/\rho}{\rho + a + \kappa^2/\rho} \frac{d_1}{(\rho^2 + \kappa^2)^{1/2}} = \frac{d_1 (\rho^2 + \kappa^2)^{1/2}}{\rho^2 + \kappa^2 + \rho a} \quad (\text{A.50})$$

where $(\rho^2 + \kappa^2)^{1/2} = |\vec{t}(\theta)| = |\vec{d}H(\theta)/d\theta|$, the distance a point on the helix moves per radian increase in θ . Thus, of $\theta_{\min} > 0$ we look for intersections between $\theta_0 + \zeta + \theta_{\min}$ and $\theta_0 + \pi$ and if $\theta_{\min} < 0$ we look between θ_0 and $\theta_0 + \zeta + \theta_{\min}$. In either case, $B(\theta)$ is well-behaved over the range.

Similar considerations apply to $\theta_0 - \zeta$. It should be emphasized that if $\theta_{\min}$ is very small, then $\vec{D} \cdot \vec{t}$ is small and the problem of searching for roots of $B(\theta)$ are avoided by using the cylinder approximation.

A.1.15 Assembling the Pieces

Having considered all the components of the procedure for finding the first intersection of a ray with a helical pipe, we proceed to describe how they are put together. The flow of the calculation goes as follows.

A.1.15.1 Step 1

Fundamental constants ($b, \mu, \lambda, \dots$) are computed. Six special values s_i , $1 \leq i \leq 6$ of the parameter s are evaluated together with $n_i = n(s_i)$ and $n_0 = n(0)$. The number n of the first turn to be investigated is selected using the following decision rules.

1. if $s_1 > 0$ set $n = n_1$
2. if $s_1 < 0 < s_2$ set $n = n_0$
3. if $s_2 < 0 < s_3$ set $n = n_3$
4. if $s_3 < 0 < s_4$ set $n = n_0$
5. if $s_4 < 0$ then there is no intersection
6. if $n = n_6$ as set by one of the above, change n to n_5 .

A.1.15.2 Step 2

Having selected a turn for investigation, the method of A1.12 is applied to determine whether the line of the ray comes close enough to either half of the turn to intersect the pipe. If so, go to step A.1.15.4 otherwise go to step A.1.15.3.

A.1.15.3 Step 3

It has been established that there is no intersection with turn n ($s < 0$ does not count).

1. if $n = n_5 < n_6$, set $n = n_6$ and go to A.1.15.2 otherwise, set $n = n + 1$ and continue.

2. if $n > n_4$ there is no intersection.
3. If $n_2 < n < n_3$, set $n = n_3$.
4. Go to step A.1.15.2

A.1.15.4 Step 4

An intersection has been identified. The value of $\vec{D} \cdot \vec{T}$ is determined at the closest known pair of points between the ray and the half-turn and on that basis, the B-method or the cylinder approximation is used to determine the location s^* of the intersection to within an acceptable tolerance. If $0 < s^* < s^{**}$ where s^{**} is the best previously determined intersection (initially; $s^{**} = 10^{21}$) then set $s^{**} = s^*$. Both half-turns are considered. If $n = n_5 < n_6$ then set $n = n_6$ and repeat this step. Otherwise, one or more intersections with $s > 0$ have been found and the procedure for ordering the turns for consideration assures us that the minimum of these positive s 's must correspond to the first intersection.

APPENDIX B

B.1 AUTOMATIC SEGMENTATION

A separate computer program has been developed to automatically segment a generalized circular cylinder. The code is called LAZY and produces data input cards for the ZAP code.

It is assumed that the cylinder is homogeneous and is confined between two end planes. The outer surface of the cylinder and the two parallel end planes are considered to be Fresnel's reflection. All internal cylinders and planes are treated as dummy boundaries.

Either a full or half cylinder may be generated. If a half cylinder is used, the plane through the center of the cylinder is treated as a plane of symmetry.

An option exists for treating a cylindrical shell. In this case the segments are those confined between an outer cylinder and an inner cylinder.

The segments are formed by generating a specified number of equally spaced planes between and parallel to the two end planes, and a specified number of concentric cylinders within the outer and inner cylinders. In addition, planes are generated at equal angles through and parallel to the cylinder axis. In this way each segment is confined by four planes and one or two cylinders.

In addition to generating boundary and segment cards, the program will generate structure data cards.

The cards generated by the program will be only a portion of the cards needed to set up a ZAP problem and are intended primarily as a means to segment the laser rod.

B.2 INPUT FOR LAZY

Input to the LAZY code is in the form of data cards. Standard Fortran Data keypunch forms are used and formats are 12I6 and 6E12.4.

Output from the code consists of punched cards and a listing of the card images.

Care must be taken that no plane passes through the origin.

Card #1 (12I6)

- 1) IOPSYM = Option for generating only half of a cylinder with a plane of symmetry.
 - 1 = Plane of symmetry with segments on non-origin side of plane.
 - 0 = Full cylinder, no plane of symmetry.
 - +1 = Plane of symmetry with segments on origin side of plane.
- 2) MOMOP = Option for generating mother segments and mother-daughter structure data cards.
 - 0 = No
 - 1 = Yes
- 3) KBAR = First boundary number to use
- 4) MBAR = Number of slices to generate (MBAR+1 parallel planes are generated).
- 5) LBAR = Number of rings to generate (LBAR or LBAR+1 concentric cylinders are generated).
- 6) NBAR = Number of wedges to generate (for a full cylinder, NBAR must be an even number. The number of planes generated is NBAR for a half cylinder and NBAR/2 for a full cylinder).

- 7) JEAR = First segment number to use.
 8) MAT = Material number for segments generated.

9-12) Not used

Card #2 (6E12.4)

- 1) α_p
 2) β_p
 3) γ_p Directional cosines of the normal to end planes. If the distance to either end plane is negative, the negative of these cosines is used.
- 4) α_s
 5) β_s
 6) γ_s Directional cosines of the normal to the plane of symmetry.

Card #3 (6E12.4)

- 1) X_{p1}
 2) Y_{p1}
 3) Z_{p1} Coordinates of a point on one of the end planes.
- 4) X_{p2}
 5) Y_{p2}
 6) Z_{p2} Coordinates of a point on the other end plane.

Card #4 (6E12.4)

- 1) α_c
 2) β_c
 3) γ_c Directional cosines of the cylindrical axis.
- 4) X_c
 5) Y_c
 6) Z_c Coordinates of a point on the cylindrical axis.

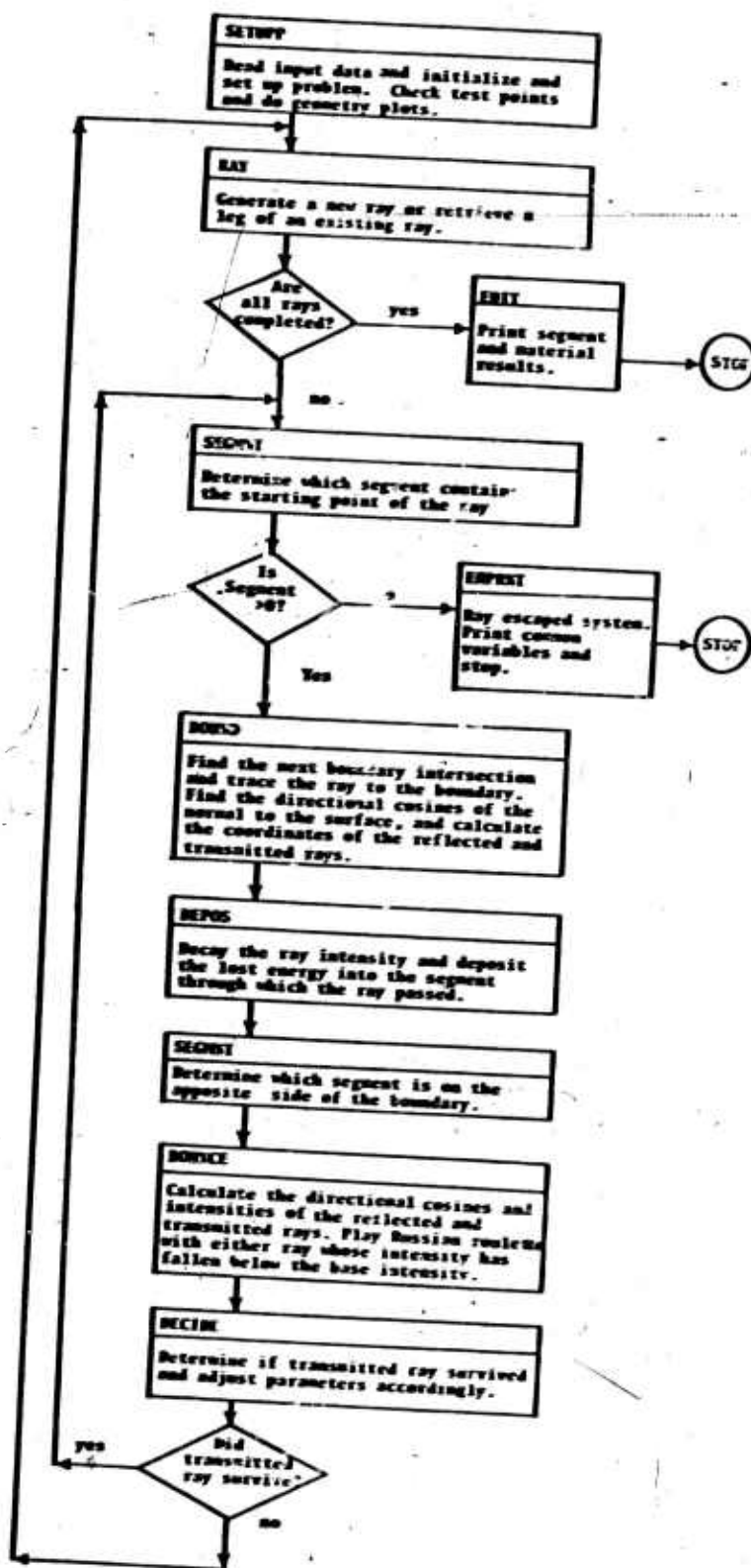
Card #5 (6E12.4)

- 1) RC = Radius of outer cylinder.
- 2) RI = Radius of inner cylinder (usually 0.0).
- 3-6) Not used

APPENDIX C

ZAP FLOW DIAGRAM AND PROGRAM LISTING

ZAP FLOW DIAGRAM




```

PROGRAM ZAP
C.....
C..... ZAP IS THE MAIN PROGRAM AND IS PRIMARILY A CONTROL REGION.
C.....
C..... THE MEANING AND MAXIMUM SIZE OF THE SUBSCRIPTS IS AS FOLLOWS
C.....
C..... F WAVELENGTH      200      NOTE THAT F IS INTEGER
C..... I STORED RAY      40
C..... IN INPUT RAY      500
C..... J SEGMENT         200
C..... K BOUNDARY        200
C..... KP BOUNDARY PTR   900
C..... L REFLECT TYPE    20
C..... LN LAMP TYPE      20
C..... M MATERIAL        20
C..... N SOURCE          50
C.....
C..... THE MEANING OF THE VARIABLES IN COMMON IS AS FOLLOWS
C.....
C..... ARCO(M,F)=ABSORPTION COEFFICIENT OF MATERIAL M AND WAVELENGTH
C..... AL= ALPHA OF RAY BEING FOLLOWED.
C..... ALK(K)= X DIRECTIONAL COSINE ASSOCIATED WITH BOUNDARY K.
C..... ALN=ALPHA OF NORMAL.
C..... ALPHA(I)= X DIRECTIONAL COSINE OF STORED RAY I.
C..... ALR= ALPHA OF REFLECTED RAY.
C..... ALRAY(IN)=ARRAY OF ALPHA FOR INPUT RAYS.
C..... ALRF= ALPHA OF REFRACTED RAY.
C..... BE= BETA OF RAY BEING FOLLOWED.
C..... BEK(K)= Y DIRECTIONAL COSINE ASSOCIATED WITH BOUNDARY K.
C..... BEN=BETA OF NORMAL.
C..... BER= BETA OF REFLECTED RAY.
C..... BERAY(IN)=ARRAY OF BETA FOR INPUT RAYS.
C..... BERF= BETA OF REFRACTED RAY.
C..... BETA(I)= Y DIRECTIONAL COSINE OF STORED RAY I.
C..... COORDS(IND,N)= SIX COORDINATE LIMITS FOR SOURCE N.
C..... DEV(I)= DEVIATION OR ERROR FOR SEGMENT J.
C..... DIS= DISTANCE FROM POINT TO INTERSECTION.
C..... DK(K)= DISTANCE ASSOCIATED WITH BOUNDARY K.
C..... DMIN= MINIMUM INTERSECTION DISTANCE FOUND.
C..... DZ= DISTANCE BETWEEN TOP AND BOTTOM BOUNDARY.
C..... E(I,F)= ENERGY OF STORED RAY I AT WAVELENGTH F.
C..... EPASE(F)= BASE ENERGY FOR RAY AS A FUNCTION OF WAVELENGTH.
C..... EK(K)= ECCENTRICITY IF BOUNDARY K IS A CONIC.
C..... ELAM(I)= SCALED ENERGY DEPOSITED IN SEGMENT J.
C..... ELAMAT(M)= SCALED ENERGY FOR MATERIAL M.
C..... EMAT(M)= ENERGY DEPOSITED IN MATERIAL M.
C..... ENERGY(J)= SEGMENT ENERGY.
C..... EPS= SMALL DISTANCE FOR MOVING RAY POSITION OFF OF BOUNDARY.
C..... ESOR(LN,F)= RAY ENERGY FOR LAMP TYPE LN AND WAVELENGTH F.
C..... FMAX= NUMBER OF WAVELENGTHS(INTEGER).
C..... GA= GAMMA OF RAY BEING FOLLOWED.
C..... GAK(K)= Z DIRECTIONAL COSINE ASSOCIATED WITH BOUNDARY K.
C..... GAMMA(I)= Z DIRECTIONAL COSINE OF STORED RAY I.
C..... GAN=GAMMA OF NORMAL.
C..... GAR= GAMMA OF REFLECTED RAY.
C..... GARAY(IN)=ARRAY OF GAMMA FOR INPUT RAYS.
C..... GARF= GAMMA OF REFRACTED RAY.
C..... IFLAG(I)= FLAG INDICATING IF STORED RAY I IS LIVE OR DEAD.
C..... IPAX= NUMBER OF RAYS STORED.

```

C***** INOW= PRESENT RAY INDEX.
 C***** IOPRNT= OPTION FOR PRINTING EACH RAY CREATED.
 C***** IPOINT(N)=A POINTER POINTING TO FIRST STORED RAY FOR SOURCE N.
 C***** IR= LAST RANDOM INTEGER USED.
 C***** IRAT= RAY NUMBER BEING TRACED.
 C***** ISMAX(N)= NUMBER OF RAYS TO CREATE FOR SOURCE N.
 C***** ISOR(ILN)= INDEX OF SOURCE TYPE FOR LAMP TYPE LN.
 C***** JF(J)= FIRST DAUGHTER OF MOTHER SEGMENT J.
 C***** JFL(J)= FLAG INDICATING IF RAY DEPOSITED ENERGY IN SEGMENT J.
 C***** JL(J)= LAST DAUGHTER OF MOTHER SEGMENT J.
 C***** JM(J)= MOTHER OF DAUGHTER SEGMENT J.
 C***** JMAX= MAXIMUM SEGMENT NUMBER.
 C***** JN(J)= NEXT SISTER IN HIERARCHY.
 C***** JNOW= SEGMENT RAY IS IN NOW.
 C***** JCLO= SEGMENT RAY WAS IN.
 C***** JP(J)= PREVIOUS SISTER IN HIERARCHY.
 C***** JRAY(J)= NUMBER OF RAYS WHICH HAVE PASSED THROUGH SEGMENT J.
 C***** JSOR(N)= SEGMENT NUMBER OF SOURCE NUMBER N.
 C***** KBND(KP)= BOUNDARY NUMBERS FOR SEGMENTS.
 C***** KFLAG(K)= FLAG INDICATING IF BOUNDARY K HAS ALREADY BEEN CHECKED.
 C***** KINT= BOUNDARY BEING CONSIDERED FOR INTERSECTION.
 C***** KMAX= MAXIMUM BOUNDARY NUMBER.
 C***** KNO= BOUNDARY WHERE RAY INTERSECTED.
 C***** KP(J)= INDEX POINTING TO BOUNDARY NUMBERS FOR SEGMENT J.
 C***** KTY(K)= GEOMETRY TYPE FOR BOUNDARY K.
 C***** LA-T(N)= LAMP TYPE NUMBER FOR SOURCE NUMBER N.
 C***** LA-LE(ILN)= INDEX OF ANGLE ROUTINE FOR LAMP TYPE LN.
 C***** LREFL(K)= REFLECTION NUMBER FOR BOUNDARY K.
 C***** LTYPE(IL)= REFLECTION TYPE FOR REFLECTION NUMBER L.
 C***** MAT(J)= MATERIAL NUMBER FOR SEGMENT J.
 C***** MFL(M)= FLAG INDICATING IF RAY DEPOSITED ENERGY IN MATERIAL M.
 C***** MPAX= MAXIMUM MATERIAL NUMBER.
 C***** MPAT(M)= NUMBER OF RAYS THAT PASSED THROUGH MATERIAL M.
 C***** MNO= SOURCE NOW BEING USED.
 C***** MSOR= MAXIMUM SOURCE NUMBER.
 C***** MURAY(J)= NUMBER OF RAYS THAT PASSED THROUGH SEGMENT J.
 C***** PER= PERCENT OF RAY TRANSMITTED.
 C***** PFD(L,F)= REFLECTION DATA FOR REFLECTION L AND WAVELENGTH F.
 C***** REF(M)= INDEX OF REFRACTION FOR MATERIAL M.
 C***** SCRMAT(M)= TERM NEEDED FOR ERROR ANALYSIS.
 C***** SUMRA= SUMMATION OF RAY ENERGY.
 C***** SUMRAY(J)= TOTAL ENERGY DEPOSITED IN SEGMENT J BY ONE RAY.
 C***** SUMSOR(J)= SUM OF RAY ENERGY SQUARED IN SEGMENT J.
 C***** TITLE(I2)= VARIABLE ALPHANUMERIC TITLE WHICH DESCRIBES RUN.
 C***** VMAT(M)= VOLUME OF MATERIAL M.
 C***** VCL(J)= VOLUME OF SEGMENT J.
 C***** XI(I)= X COORDINATE OF STORED RAY I.
 C***** XIP= POINT USED FOR FINDING POSITION AND INTERSECTION.
 C***** XK(K)= X COORDINATE OF POINT ASSOCIATED WITH BOUNDARY K.
 C***** XLAM(F)= ARRAY OF WAVELENGTHS.
 C***** XNO= X OF INTERSECTION POINT.
 C***** XCLO= X OF STARTING POINT OF RAY.
 C***** XRAY(IN)= X COORDINATES OF INPUT RAYS.
 C***** XREF= X OF REFLECTED RAY.
 C***** XTRAN= X OF TRANSMITTED RAY.
 C***** YI(I)= Y COORDINATE OF STORED RAY I.
 C***** YIP= POINT USED FOR FINDING POSITION AND INTERSECTION.
 C***** YK(K)= Y COORDINATE OF POINT ASSOCIATED WITH BOUNDARY K.
 C***** YNO= Y OF INTERSECTION POINT.
 C***** YCLO= Y OF STARTING POINT OF RAY.
 C***** YRAY(IN)= Y COORDINATES OF INPUT RAYS.
 C***** YREF= Y OF REFLECTED RAY.

```

***** YTRAN= Y OF TRANSMITTED RAY.
***** Z(I)= Z COORDINATE OF STORED RAY I.
***** ZI= POINT USED FOR FINDING POSITION AND INTERSECTION.
***** Z(K)= Z COORDINATE OF POINT ASSOCIATED WITH BOUNDARY K.
***** ZLAN(N)= LASER WAVELENGTH OF MATERIAL N.
***** ZLOD= Z OF INTERSECTION POINT.
***** ZOLD= Z OF STARTING POINT OF RAY.
***** ZRAY(I)= Z COORDINATES OF INPUT RAYS.
***** ZREF= Z OF REFLECTED RAY.
***** ZTRAN= Z OF TRANSMITTED RAY.
COMMON /69/ ABCO(20,200)
1 COORDS(6,50) E(40,200) EBASZ(200) EBAY(500)
2 GARAY(543) RED(20,200) TITLE(12) ESON(20,200)
3 VOL(200) XLAN(200) XRAY(500) YMAT(20)
4 ZRAY(500)
COMMON /ARRAYS/ ALX(200) ALP-A(40) BEK(200)
1 OPTA(40) DEV(200) BK(200) EK(200)
2 ELA(200) ELAYAT(20) ERAT(20) ENERGY(200)
3 GAK(200) GAMMA(40) REF(20) SOMAT(20)
4 SUMRAY(200) SUMSOR(200) X(40) XK(200)
5 Y(40) YK(200) Z(40) ZX(200)
6 ZLAN(20)
COMMON /IARRAY/ IFLAG(40) IPOINT(50)
1 ISHAX(50) ISOR(20) JF(200) JFL(200)
2 JL(200) JN(200) JN(200) JP(200)
3 JRAY(200) JSOR(50) KBAD(900) KFLAG(200)
4 KP(200) KTYP(200) LAMP(50) LANGLE(20)
5 LREFL(200) LTYPE(20) NAT(200) NFL(20)
6 NATRAY(20)
COMMON /PARAMS/ AL ALN ALR ALRF DE
1 DEN DERF DIS DMIN DZ EPS
2 GA GAN GAR GARF PER
3 SUMRA XI XACH XOLD XREF XTRAN YI
4 YNOLD YREF YTRAN ZI
COMMON /IPARAM/ FMAX IMAX INON
1 IDPRAY ICPRAY IR IRAY JMAX JNON
2 INAT KMAX KAON KMAX KNON KNSOR JOLD
INTEGER F,FMAX
***** THE L2-SE NUMBERED COMMON BLOCK IS FORCED TO LOAD OVER THE
***** LOADER IN BANK 0. THIS KEEPING THE PROGRAMS IN BANK 1. WHICH
***** IS FASTER.
BANK. 101. /69/
***** BECAUSE THE NUMBERED COMMON BLOCK, USING OTHER STORAGE, IS
***** NOT SET TO ZERO ON LOADING, IT MUST BE SEPARATELY ZEROED.
DIMENSION BUNT(23932)
EQUIVALENCE (BUNT(1),ABCO(1))
TIME = TIMELEFT(1)
B9 106 IAN=23932
BUNT(1A) = 0.
106 CONTINUE
***** READ INPUT DATA AND SET UP PROBLEM.
CALL SETUP
***** GET THE NEXT RAY TO BE PROCESSED.
100 CALL RAY(IFLG)
***** IF THERE ARE NO MORE RAYS GO TO 102.
IF(IFLG)102,103,103
***** DETERMINE WHICH SEGMENT CONTAINS STARTING POINT OF RAY.
103 CALL SEGMENT
***** IF SEGMENT NUMBER IS ZERO THE RAY IS OUTSIDE THE SYSTEM.
IF(JNO=)104,104,101
***** FIND POINT WHERE RAY INTERSECTS THE NEAREST BOUNDARY.

```

```

101 CALL DEND
C..... DEPOSIT ENERGY IN SEGMENT RAY HAS JUST CROSSED.
      CALL DEPOS
C..... DETERMINE WHICH SEGMENT CONTAINS TRANSMITTED RAY.
      CALL SFGINT
C..... GET DIRECTIONAL COSINES AND ENERGY OF REFLECTED AND
C..... TRANSMITTED RAYS. PLAY RUSSIAN ROULET IF EITHER RAYS
C..... ENERGY LEVEL IS BELOW CHOSEN VALUE.
105 CALL BOUNCE
C..... DETERMINE IF NEW RAY IS NEEDED.
      CALL DECIDE(IFLG)
      IF(IFLG)101-101-100
102 CONTINUE
C..... PRINT FINAL (OR SEMI-FINAL) ANSWERS.
      CALL EDIT
      TIME = TIME + TIMELEFT(1)
      PRINT 10, TIME
10 FORMAT(1X, 'TIME EXECUTION TIME = ', F8.3, 'SECONDS')
      STOP
C..... ERROR CONDITION. RAY HAS ESCAPED SYSTEM.
104 CALL ERPRINT(6H ZAP)
      STOP
      END

```

```

SURROUTINE ANGLES(TH,PH)
C*****
C***** SURROUTINE ANGLES CALLS THE DESIRED ANGLE ROUTINE.
C*****
COMMON /69/ ABCO(20,200) ALRAY(500) BEAT(500)
1 COCOS(16,50) E(40,200) EBASE(200) ESOR(20,200)
2 GARAY(500) RED(20,200) TITLE(12) VMAT(20)
3 VOL(200) XLAM(200) XRAY(500) YRAY(500)
4 ZRAY(500)
COMMON /ARRAYS/ ALK(200) ALPHA(40) BEK(200)
1 PETA(40) DEV(200) DK(200) EK(200)
2 FLAM(200) ELAMAT(20) EMAT(20) ENERGY(200)
3 GAK(200) GAMMA(40) REF(20) SQRMAT(20)
4 SUMRAY(200) SUNSOR(200) X(40) XK(200)
5 Y(40) YK(200) Z(40) ZK(200)
6 ZLAM(20)
COMMON /JARRAY/ IFLAG(40) IPOINT(50)
1 ISWAX(50) ISOR(20) JF(200) JFL(200)
2 J(200) JM(200) JN(200) JP(200)
3 JAY(200) JSOR(50) KND(900) KFLAG(200)
4 KP(200) KYP(200) LAMP(50) LANGLE(20)
5 LREFL(200) LTYPE(20) MAT(200) MFL(20)
6 MMAT(20) MMURAY(200)
COMMON /PARAMS/ AL ALN ALR ALNF ME
1 OEN OER OENF DIS DMIN OZ EPS
2 GA GAN GAR GAF PER
3 SUMP A XI XND XOLD XREF XTRAN V1
4 YND YOLD YREF YTRAN Z1
5 ZND ZOLD ZREF ZTRAN ZMAX INOW
COMMON /JPARAM/ FMAX IMAX JMAX JND JOLD
1 IOPRAY IOPRNT IR IRAY JND JOLD
2 KJAT KMAX KND KMAX KND KND KND
INTEGER F,FMAX
C***** GET LAMP NUMBER FOR SOURCE KNOW.
LT=LAMP(KND)
C***** GET NUMBER OF ROUTINE TO CALL.
LA=LANGLE(LT)
C***** GO TO DESIRED CALL.
GO TO (100,200,300,400,500,600,700,800,900),LA
100 CALL ANGLE1(TH,PH)
GO TO 110
200 CALL ANGLE2(TH,PH)
GO TO 110
300 CALL ANGLE3(TH,PH)
GO TO 110
400 CALL ANGLE4(TH,PH)
GO TO 110
500 CALL ANGLE5(TH,PH)
GO TO 110
600 CALL ANGLE6(TH,PH)
GO TO 110
700 CALL ANGLE7(TH,PH)
GO TO 110
800 CALL ANGLE8(TH,PH)
GO TO 110
900 CALL ANGLE9(TH,PH)
110 RETURN
END

```

```

SUBROUTINE ABSORP(IAB,M)
C*****
C***** SUBROUTINE ABSORP CALLS THE DESIRED ROUTINE FOR GENERATING
C***** ABSORPTION COEFFICIENTS AS A FUNCTION OF WAVELENGTH.
C*****
COMMON /69/      ABCO(20,200)  ,ALRAY(500)  ,DERAY(500)
1  ,COORCS(6,55)  ,E(40,200)    ,EBASE(200)  ,ESOR(20,200)
2  ,GARAY(500)    ,RED(20,200)  ,TITLE(12)  ,VMAT(20)
3  ,VOL(200)      ,XLAM(200)    ,XRAY(500)    ,YRAY(500)
4  ,ZRAY(500)
COMMON /ARRAYS/  ALK(200)      ,ALPHA(40)    ,DEK(200)
1  ,RETA(40)      ,DEV(200)     ,DK(200)      ,EK(200)
2  ,ELAM(200)     ,ELAMAT(20)   ,EMAT(20)     ,ENERGY(200)
3  ,GAK(200)      ,GAMMA(40)    ,REF(20)      ,SORMAT(20)
4  ,SUMRAY(200)   ,SUMSOR(200)  ,X(40)       ,XK(200)
5  ,V(40)         ,YK(200)     ,Z(40)        ,ZK(200)
6  ,ZLAM(20)
COMMON /IARRAY/  IFLAG(40)     ,IPOINT(50)
1  ,ISNAX(50)     ,ISOR(20)     ,JF(200)      ,JFL(200)
2  ,JL(200)       ,JM(200)     ,JN(200)      ,JP(200)
3  ,JRAY(200)     ,JSOR(50)    ,KQAD(900)    ,KFLAG(200)
4  ,KP(200)       ,KTP(200)    ,LAMP(50)     ,LANGLE(20)
5  ,LREFL(200)   ,LTYPE(20)   ,MAT(200)     ,NFL(20)
6  ,MMAT(20)
COMMON /PARAMS/  AL            ,ALN          ,ALR          ,ALRF          ,RE          ,
1  ,GA            ,GAR          ,GARF         ,DIN          ,DZ          ,EPS          ,
2  ,SUMRA         ,XI           ,XAN          ,XOLD          ,PER          ,
3  ,YMON          ,YOLD         ,TREF         ,TTMAN         ,ZREF         ,XREF         ,XTRAM        ,VI          ,
4  ,ZMON          ,ZOLD         ,ZREF         ,ZTRAM         ,ZREF         ,XREF         ,XTRAM        ,VI          ,
COMMON /IPARAM/  FMAX          ,INAX          ,INOW          ,
1  ,IOPRAY        ,IOPRINT      ,IR           ,IRAY          ,JMAX          ,JMON          ,JOLD          ,
2  ,KINT          ,KMAX         ,KNOW         ,NMAX          ,NMAX          ,NMAX          ,NSOR          ,JOLD          ,
INTEGER F,FMAX
DIMENSION TSTG(200)
DIMENSION IABTAB(9)
DATA(IABTAB=6HAB51,6HAB52,6HAB53,
1  ,6HAB54,6HAB55,6HAB56,6HAB57,6HAB58,6HAB59)
C***** FIND NUMBER OF ROUTINE TO CALL,
DO 101 N=1,9
IF(IAB-IABTAB(N))101,102,101
101 CONTINUE
C***** ROUTINE NAME NOT IN TABLE.
PRINT I0,IAB
10 FORMAT(IH1,21HABSORPTION SUBROUTINE,1X,A6,13H NOT IN TABLE)
STOP
C***** GO TO DESIRED CALL
102 GO TO(100,200,300,400,500,600,700,800,900),N
100 CALL ABS1(XLAM,FMAX,TSTG,N)
GO TO 110
200 CALL ABS2(XLAM,FMAX,TSTG,N)
GO TO 110
300 CALL ABS3(XLAM,FMAX,TSTG,N)
GO TO 110
400 CALL ABS4(XLAM,FMAX,TSTG,N)
GO TO 110
500 CALL ABS5(XLAM,FMAX,TSTG,N)
GO TO 110
600 CALL ABS6(XLAM,FMAX,TSTG,N)
GO TO 110
700 CALL ABS7(XLAM,FMAX,TSTG,N)
GO TO 110
800 CALL ABS8(XLAM,FMAX,TSTG,N)
GO TO 110
900 CALL ABS9(XLAM,FMAX,TSTG,N)
110 CONTINUE
C***** STORE ABSORPTION COEFFICIENTS FOR MATERIAL NO. M,
DO 103 F=1,FMAX
ABCO(M,F)=TSTG(F)
103 CONTINUE
RETURN
END

```

```

SUBROUTINE BCALC
C*****
C***** BCALC CALCULATES FUNCTION B(TH), DERIVATIVE BP(TH), AND
C***** DISTANCE STH(TH).
C***** PART OF HELIX INTERSECTION CALCULATION.
C*****
COMMON /69/ ABCO(20,200) ,ALRAY(500) ,DERAY(500)
1 ,COORDS(6,50) ,E(40,200) ,EBASE(200) ,ESOH(20,200)
2 ,GARAY(500) ,RED(20,200) ,TITLE(12) ,VMAT(20)
3 ,VOL(200) ,XLAM(200) ,XRAY(500) ,YRAY(500)
4 ,ZRAY(500)
COMMON /ARRAYS/ ALX(200) ,ALPHA(40) ,BEK(200)
1 ,META(40) ,DEV(200) ,DK(200) ,EK(200)
2 ,ELAM(200) ,ELAMAT(20) ,EMAT(20) ,ENERGY(200)
3 ,GAK(200) ,GAMMA(40) ,REF(20) ,SOBMAT(20)
4 ,SUMRAY(200) ,SUMSOR(200) ,X(40) ,XK(200)
5 ,Y(40) ,YK(200) ,Z(40) ,ZK(200)
6 ,ZLAM(20)
COMMON /IARRAY/ IFLAG(40) ,IPOINT(50)
1 ,ISMAX(50) ,JF(200) ,JFL(200)
2 ,JL(200) ,JM(200) ,JN(200) ,JP(200)
3 ,JRAY(200) ,JSOR(50) ,KEND(100) ,KFLAG(200)
4 ,KIP(200) ,KTP(200) ,LAMP(50) ,LANGLE(20)
5 ,LREFL(200) ,LTYPE(20) ,MAT(200) ,NFL(20)
6 ,MMAT(20) ,MMRAY(200)
COMMON /PARAMS/ AL ,ALN ,ALR ,ALNF ,BE ,
1 ,GA ,GAN ,GAR ,GARF ,PER ,BZ ,EPS
2 ,SUNRA ,XI ,XAN ,XOLD ,XREF ,XTRAN ,YI
3 ,YNOV ,YOLD ,YREF ,YTRAN ,ZI
4 ,ZNOV ,ZOLD ,ZREF ,ZTRAN
COMMON /IPARAM/ FMAX ,IMAX ,INOV
1 ,IOPRAY ,IOPRNT ,IR ,IRAY ,JMAX ,JNOV
2 ,KIAT ,KMAX ,KNOV ,KMAX ,KNOV ,KSOR ,JOLD
INTEGER F,FMAX
COMMON /HELCON/ A,ABAR,B,BI,BIP,BP,CAP,DELTH,DELX,DELY,DELZ,DHIT,
1 ,PSIZ,REV,RHO,STH,TH,THI,SORAS,ORS,CSTH,SNTH,NCSTH,RSAT,OSTH,
2 ,COSTAN,THZ,DTHZ
C***** CALCULATE PARAMETERS.
STH=SIGN(TH)
CTH=COS(TH)
RCOS=RHO*CTH
RSIN=RHO*STH
DEN=-AL*RSIN+BE*RCOS+GA*CAP
ENM=-DELX*RSIN+DELY*RCOS+(DELZ*CAP*TH)*CAP
C***** CALCULATE DISTANCE STH.
STH=ENM/DEN
C***** CALCULATE DERIVATIVE OF STH.
SP=((DELX*RCOS-DELY*RSIN+CAP**2)*DEN-ENM*(-AL*RCOS-BE*RSIN))
SP=SP/(DEN**2)
ALP=DELX*RCOS-AL*STH
BEB=DELY*RSIN-BE*STH
GAR=DELZ*CAP*TH-GA*STH
C***** CALCULATE FUNCTION B.
B=.5*ALB**2+.5*BEB**2+.5*GAR**2
C***** CALCULATE DERIVATIVE OF B.
BP=ALB*(-RSIN-AL*SP)+BEB*(RCOS-BE*SP)+GAR*(CAP-GA*SP)
RETURN
END

```

```

SUBROUTINE BMDAT
C*****
C***** SUB-ROUTINE BMDAT READS BOUNDARY DATA AND SETS UP BOUNDARIES.
C*****
COMMON /G9/ AECO(2L,200) ALRAY(500) BERAY(500)
1 COORDS(6,50) E(40,200) EBASE(200) CSOR(20,200)
2 GARAY(500) RED(20,200) TITLE(12) VMAT(20)
3 VOL(200) XLAM(200) XRAY(500) VRAY(500)
4 ZRAY(500)
COMMON /ARRAYS/ ALK(200) ALPHA(40) BEK(200)
1 PETA(40) DEV(200) DK(200) EK(200)
2 ELAM(200) ELAMAT(20) EPAT(20) ENRGY(200)
3 GAK(200) GAMMA(40) REF(20) SCGMAT(20)
4 SUMRAY(200) SUMSOR(200) X(40) XK(200)
5 Y(40) YK(200) Z(40) ZK(200)
6 ZLAM(20)
COMMON /IARRAY/ IFLAG(40) IPOINT(50)
1 ISMAX(50) ISOR(20) JF(200) JFL(200)
2 JL(200) JN(200) JN(200) JP(200)
3 JMAT(200) JSOR(50) KEND(900) KFLAG(200)
4 KEP(200) KTYPE(200) LAMP(50) LANGLE(20)
5 LREFL(200) LTYPE(20) MAT(200) MFL(20)
6 MAT(20)
COMMON /PARAMS/ AL ALN ALR ALRF BE
1 BEN BER BERF DIS DMIN DL EPS
2 GA GAN GAR GARF PER U
3 SUMMA XI XND XOLD XREF XTRAN YI
4 YND YOLD YREF YTRAN ZI
5 ZND ZOLD ZREF ZTRAN
COMMON /PARA/ FMAX INAX INON
1 IOPRAY IOPRINT IR IRAY JMAX JON
2 KIAT KMAX KND KMAX KNON KSOR JOLD
INTEGER F,FMAX
COMMON /MDCOM/MDPAR(12)
DI=ENSTION KTABLE(10)
DATA(KTABLE=6,MPANE,6,MCYLIND,6,MCONE,6,MSPHERE,6,MCONE,6,MPARAB,
6,MYF,6,6,6,ELLIPS,6,HELIX,6,6)
PRINT 12
12 FORMAT(1H,53X,13#BOUNDARY DATA//6H BOUND,2X,4#REFL,3X,
5#BOUND,3X,5#ALPHA,4X,4#ETA,3X,5#GAMMA,7X,1#X,7X,1#Y,
7X,1#Z,6X,2#X2,6X,2#Y2,6X,2#X3,6X,2#Y3,5X,3#DIS,5X,3#ECC/
6H MDP,2X,4#MND,4X,4#TYPE/)
MDP=5
KMAX=0
C***** LOOP THROUGH ALL BOUNDARIES.
104 READ 10,K,LREFI,KTYPI,(BMDPAR(N),N=1,8)
10 FORMAT(2I4,1X,A6,PF7,5)
C***** IF K GREATER THAN KMAX, KMAX=K.
IF(K)111,111,105
105 IF(K-KMAX)106,106,107
107 KMAX=K
C***** DETERMINE GEOMETRY TYPE
106 DO 108 KT=1,10
KTYPI=KT
IF(KTYPI-KTABLE(KT))100,109,100
108 CONTINUE
C***** GEOMETRY TYPE NOT IN KTABLE.
CALL ERPRINT(6,BMDAT)
C***** STORE REFLECTION NUMBER.
109 LREFL(K)=LREFI
KTYPI=KTYPI(K)

```



```

C***** BRANCH ON GEOMETRY TYPE.
102 GO TO(100,200,300,400,500,600,700,800,900,1000)MT/PI
C***** GENERATE A PLANE.
100 CALL SETPLA(K)
GO TO 110
C***** GENERATE A CYLINDER.
200 CALL SETCYL(K)
GO TO 110
C***** GENERATE A CONIC.
300 CALL SETCON(K)
GO TO 110
C***** GENERATE A SPHERE
400 CALL SETSPH(K)
GO TO 110
C***** GENERATE A HALF-CONE
500 CALL SETCN(K)
GO TO 110
C***** GENERATE A PARABOLOID
600 CALL SETPAR(K)
GO TO 110
C***** GENERATE A HYPERBULOID
700 CALL SETHYP(K)
GO TO 110
C***** GENERATE AN ELLIPSOID
800 CALL SETELL(K)
GO TO 110
C***** GENERATE A HELIX
900 CALL SETHEL(K)
GO TO 110
1000 CONTINUE
GO TO 110
110 MLP=MLP+1
C***** IF NUMBER OF LINES PRINTED EXCEEDS 52, EJECT AND PRINT TITLE.
IF(MLP-52)104,104,112
112 MLP=5
PRINT 12
C***** GO READ NEXT BOUNDARY CARD.
GO TO 104
111 RETURN
END

```

SUBROUTINE BOUNCE

 ***** BOUNCE DETERMINES THE DIRECTIONAL COSINES AND ENERGY OF
 ***** THE TRANSMITTED AND REFLECTED RAYS AND DETERMINES IF THEY
 ***** SURVIVE.

```

COMMON /69/ ABCO(20,200) ALRAY(500) BERAY(500)
1 COGNES(6,50) E(40,200) EBASE(200) ESOR(20,200)
2 GARAY(500) RED(20,200) TITLE(12) VNAT(20)
3 VVL(200) XLAM(200) XRAT(500) YRAY(500)
4 ZRAY(500)

COMMON /ARRAYS/ ALK(200) ALP=4(46) REK(200)
1 RETA(40) DEV(200) DK(200) EK(200)
2 CLAM(200) ELAMAT(20) ERAT(20) ENERGY(200)
3 GAK(200) GAMMA(40) REF(20) SCRMAT(20)
4 SUPRAY(200) SUMSOR(200) X(40) XK(200)
5 Y(40) YK(200) Z(40) ZK(200)
6 ZLAM(20)

COMMON /IARRAY/ IFLAG(40) IPOINT(50)
1 ISMAX(50) ISOR(20) JF(200) JFL(200)
2 JL(200) JP(200) JA(200) JP(200)
3 JRAY(200) JSOR(50) KEND(900) KFLAG(200)
4 KAP(200) KTYPE(200) LAMP(50) LANGLE(20)
5 LREFL(200) LTYPE(20) MAT(200) PFL(20)
6 MURAY(200)

COMMON /PARAMS/ AL ALN ALR ALRF DE
1 DER DERF OIS ODIR OZ EPS
2 GA GAN GAR GARF PER
3 SUMRA XI INON XCLO XREF XTRAN VI
4 YLOW YOLD YPEF YTHAN ZI
5 ZLOW ZOLD ZPEF ZTRAN

COMMON /IPARAM/ FMAX IPAX INON
1 IOPRAY IOPRNT IR IRAY JMAX JON
2 JINT JNEX KNON MPAX MNON NSOR JOLD

INTEGER F,F AX
I=INON
K=KEND

***** SFT COORDINATES OF TRANSMITTED RAY.
X(1)=XTRAN
Z(1)=ZTRAN
Y(1)=YTRAN

***** IF RAY ENERGY LESS THAN .001*(BASE ENERGY), SKIP CALCULATIONS
IEFLAG=0
DO 114 F=1,FMAX
IF(E(1,F)-(1.E-3)*EBASE(F))115,115,116
115 E(1,F)=0.0
GO TO 114
116 IEFLAG=1
114 CONTINUE
IF(IEFLAG)120,120,102

***** IF DUMMY BOUNDARY, GO TO 103.
102 IF(ILREFL(K)-1)109,103,109
***** IF TOP CONTINUATIVE BOUNDARY, GO TO 103.
100 IF(ILREFL(K)-4)110,103,110
***** IF BOTTOM CONTINUATIVE BOUNDARY, GO TO 103.
110 IF(ILREFL(K)-5)104,103,104
***** PLAY RUSSIAN ROULET WITH TRANSMITTED RAY.
103 CALL ROULET(1)
GO TO 120
104 M1=MAT(JOLD)
M2=MAT(JMON)
  
```

```

C***** IF REFRACTIVE INDICES ARE EQUAL, TREAT AS DUMMY BOUNDARY.
      IF (ABS(REF(MI)-REF(ME))-1.E-103.103.130
C***** CALCULATE DIRECTIONAL COSINES OF REFLECTED RAY.
      130 CALL REFLECT
C***** STORE REFLECTED RAY ON INDEX IMAX.
      IMAX=IMAX+1
      ALPHA(IMAX)=ALR
      BETA(IMAX)=BER
      GAMMA(IMAX)=GAR
      XI(IMAX)=XREF
      Y(IMAX)=YREF
      Z(IMAX)=ZREF
C***** IF BOUNDARY OF TOTAL REFLECTION, GO TO 105.
      IF (LREFL(K)-2)106.105.106
C***** REFLECT TOTAL RAY.
      105 DO 111 F=1,FMAX
      E(IMAX,F)=E(I,F)
      E(I,F)=0.0
      111 CONTINUE
C***** PLAY RUSSIAN ROULET WITH REFLECTED RAY.
      CALL ROULET(IMAX)
      GO TO 120
C***** CALCULATE DIRECTIONAL COSINES OF TRANSMITTED RAY AND PERCENT
C***** OF RAY TO BE TRANSMITTED.
      106 CALL REF2AC
C***** PLAY RUSSIAN ROULET WITH TRANSMITTED RAY.
      CALL ROULET(I)
C***** PLAY RUSSIAN ROULET WITH REFLECTED RAY.
      100 CALL ROULET(IMAX)
      120 CALL DEBUG(NGUNCE)
C***** SET FLAG INDICATING RAY EXISTS
      IFLAG(IMAX)=1
      IFLAG(I)=1
      RETURN
      END

```

05/24/71

SUBROUTINE BOUND

```

***** BOUND DETERMINES POINT WHERE RAY INTERSECTS NEAREST BOUNDARY. BOUND
*****
COMMON /G9/ ABCD(20,200) ALRAY(500) DERAY(500)
1 COORDS(6,50) E(40,200) ERASE(200) ESOR(20,200)
2 ERAY(500) RED(20,200) TITLE(12) VNAT(20)
3 VOL(200) XLAN(200) XRAY(500) YRAY(500)
4 ZRAY(500)
COMMON /ARAYS/ ALX(200) ALPHA(40) BEX(200)
1 BETA(40) BEV(200) DK(200) EK(200)
2 ELAN(200) ELANAT(20) ENAT(20) ENERGY(200)
3 GAX(200) GAMMA(40) REF(20) SCRNAT(20)
4 SUPRAY(200) SUNSCR(200) X(40) XK(200)
5 Y(40) YK(200) Z(40) ZK(200)
6 ZLAN(20)
COMMON /IARAY/ IFLAG(40) IPOINT(50)
1 ISMAX(50) ISOR(20) JF(200) JFL(200)
2 JL(200) JM(200) JN(200) JP(200)
3 JRAY(200) JSOR(50) KEND(900) KFLAG(200)
4 KIP(200) KTOP(200) LAMP(50) LANGLE(20)
5 LFL(200) LTYPE(20) MAY(200) MFL(20)
6 MPAT(20) MURRAY(200)
COMMON /PARAMS/ AL ALN ALR ALRF BE
1 GEN GER GEF DIS DMIN DZ EPS
2 GA GAN GAR GAPP PEN
3 SUNRA XI XNOV XOLD XREF XTRAN VI
4 YNOV YOLD YREF YT-1 ZI
5 ZNOV ZOLD ZREF ZT-1 INK INOW JNOV JOLD
COMMON /IPARIN/ FMAX IR IRL JMAX JNOV JSOR
1 IOPRAY IOPRINT IIR IRL JMAX JNOV JSOR
2 IINT KMAX KNOV KMAX KNOV KMAX KNOV
INTEGER F,FMAX
DMIN=1.E-70
***** CHECK BOUNDARIES OF SEGMENT CONTAINING STARTING POINT OF RAY. BOUND
101 J=JNOV BOUND
102 CALL JBOUND(J) BOUND
***** IF DISTANCE LESS THAN MINIMUM, SET MINIMUM EQUAL TO DISTANCE. BOUND
IF(DIS-DMIN)105,130,104 BOUND
***** IF DISTANCE EQUAL MINIMUM AND J=JNOV, CHOOSE NEW BOUNDARY. BOUND
130 IF(J-JNOV)100,105,104 BOUND
105 KNOV=INT BOUND
DMIN=DIS BOUND
***** IF JNOV IS A MOTHER, GO TO 106. BOUND
104 IF(JF(JNOV))100,110,106 BOUND
***** SET J EQUAL TO FIRST DAUGHTER. BOUND
106 J=JF(JNOV) BOUND
GO TO 107 BOUND
***** ARE THERE MORE SEGMENT TO CHECK. IF NOT GO TO 110. BOUND
112 IF(JN(J))109,110,109 BOUND
***** CHECK NEXT SEGMENT IN HIERARCHY. BOUND
109 J=JN(J) BOUND
107 CALL JBOUND(J) BOUND
***** IF DIS LESS THAN DMIN, SAVE NEW BOUNDARY. BOUND
IF(DIS-DMIN)111,131,112 BOUND
***** IF DIS EQUAL DMIN AND J=JNOV, SAVE NEW BOUNDARY. BOUND

```

05-2471

[illegible]

```

SUBROUTINE CCSTAN
C.....
C..... CCSTAN CALCULATES COSTAN.
C..... PART OF HELIX INTERSECTION CALCULATION.
C.....
COMMON /69/      ABCO(20,200)  •ALRAY(500)  •BERAY(500)
1 •COORDS(6,50)  •E(40,200)    •EBASE(200)  •ESOR(20,200)
2 •GARAY(500)    •RED(20,200)  •TITLE(12)  •VNAT(20)
3 •VOL(200)      •XLAN(200)    •XRAY(500)  •YRAY(500)
4 •ZRAY(500)
COMMON /ARRAYS/  ALK(200)      •ALPHA(40)  •BEK(200)
1 •META(40)      •DEV(200)    •EK(200)    •EK(200)
2 •ELAN(200)     •ELANAT(20)  •ENAT(20)  •ENERGY(200)
3 •GAK(200)      •GAMMA(40)  •REF(20)   •SOMAT(20)
4 •SUMRAY(200)   •SUMSUR(200) •X(40)     •XK(200)
5 •Y(40)         •YK(200)    •Z(40)     •ZK(200)
6 •ZLAN(20)
COMMON /IARRAY/  IFLAG(40)    •IPOINT(50)
•ISMAX(50)      •ISOR(20)    •JF(200)   •JFL(200)
•JL(200)        •JM(200)    •JN(200)   •JP(200)
•JRAY(200)      •JSOR(50)   •KND(1000) •KFLAG(200)
•KIP(200)       •KTYP(200)  •LAMP(50)  •LANGLE(20)
•LREFL(200)     •LTYPE(20)  •MAT(200)  •NFL(20)
•MMAT(20)       •MURAY(200)
COMMON /PARAMS/  AL          •ALN      •ALR      •ALRF     •BE      •
•GA          •BER          •DIS      •DMIN    •DZ      •EPS      •
•SUMRA      •XI          •ZKON      •ZOLF     •PER      •
•YOLD      •YOLD      •YREF     •YIMAN   •ZI      •XTRAM    •YI      •
•ZKON      •ZOLD      •ZREF     •ZTRAM
COMMON /IPARAM/  FMAX        •INAX     •INON
•IOPRAY      •IOPINT      •IR      •IRAY    •JMAX    •JNON    •JOLD
•KINT        •KPAX      •KNON    •PMAX    •PMON    •NSOK
INTEGER F,FMAX
COMMON /HELCON/A,ABAR,B,B1,BIP,BP,CAP,DELTH,DELX,DELY,DELZ,DMIT,
•PSIZ,REV,RNO,STN,TH,TN1,SOMLS,SRKS,CSTN,SNTH,RCSTN,RSNTH,BSTN,
•COSTAN,THZ,OTHZ
CSTN=CCS(TH)
SNTH=SIN(TH)
RCSTN=END+CSTN
RSNTH=END+SNTH
COSTAN=(-RSNTH*AL+RCSTN*BE+CAP*GA)/SORKS
RETURN
END

```

```

SUBROUTINE CHECK(REG,AA,BB,GG)
C.....
C..... SUBROUTINE CHECK ASSURES THAT THE SUM OF THE SQUARES OF A
C..... SET OF DIRECTIONAL COSINES ADDS UP TO ONE.
C.....
ONE=AA**2+BB**2+GG**2
C..... IF THE SUM OF THE SQUARES IS LESS THAN 1.E-3, GO TO 102.
IF(ABS(ONE-1.0)-2.E-2)102,102,101
101 PRINT 10,REG,AA,BB,GG,ONE
10 FORMAT(1H, 'A6, 4E12.4)
C..... ERROR IN DIRECTIONAL COSINES.
CALL ERRPRN(4,ONE)
STOP
C..... ADJUST ALPHA TO MAKE COSINES MORE ACCURATE.
C..... ADJUST COSINES
102 SO=SQRT(ONE)
AA=AA/SO
BB=BB/SO
GG=GG/SO
104 RETURN
END

```

```

SUBROUTINE CNOR(K)
C.....
C..... SUBROUTINE CNOR FINDS THE DIRECTIONAL COSINES
C..... OF THE NORMAL TO A POINT ON A CONE.
C.....
COMMON /69/ ABCO(20,200) ALRAY(500) BERAY(500)
1 COORDS(6,50) E(40,200) EBASE(200) ESOR(20,200)
2 GARA(500) RED(20,200) TITLE(12) VMAT(20)
3 VOL(200) XLAM(200) XRAY(500) YRAY(500)
4 ZRAY(500)
COMMON /ARRAYS/ ALK(200) ALPHA(40) BEK(200)
1 RETA(40) DEV(200) DM(200) EK(200)
2 ELAM(200) ELAMAT(20) ENAT(20) ENEMAT(200)
3 GAK(200) GAMMA(40) REF(20) SORMAT(20)
4 SUMRAY(200) SUMSOR(200) X(40) XK(200)
5 Y(40) YK(200) Z(40) ZK(200)
6 ZLAM(20)
COMMON /IARRAY/ IFLAG(40) IPOINT(50)
1 ISMAX(50) ISOR(20) JF(200) JFL(200)
2 JL(200) JM(200) JM(200) JP(200)
3 JRAY(200) JSOR(50) KEND(900) KFLAG(200)
4 KP(200) KTOP(200) LAMP(50) LANGLE(20)
5 LREFL(200) LTYPE(20) MAT(200) NFL(20)
6 MNAT(20) SUMRAY(200)
COMMON /PARAMS/ AL ALN ALR ALRF RE
1 BEN BER BERF DIS DMIN DZ EPS
2 GA GAN GAR GARF PER XOLD
3 SUMRA XI XOW XOLF XREF XTRAN VI
4 YNOW YOLD YREF YTHAN ZI
5 ZNOW ZOLD ZREF ZTHAN
COMMON /IPARAM/ INOV INOV INOV JOLD
1 IOPRAY IOPRNT IR IRAY JMAX JON NSOR
2 KINT KMAX KNON MMAX MNON
INTEGER F,FMAX
DELX=XAC-XK(K)
DELY=YAC-YK(K)
DELZ=ZAC-ZK(K)
SO=SQRT(OR(K)**2*(DELX**2+DELY**2)+(1-OR(K))**2*DELZ**2)
C..... CALCULATE DIRECTIONAL COSINES OF NORMAL.
ALA=OR(K)*DELX/SO
BEA=OR(K)*DELY/SO
GAA=(OR(K)-1.0)*DELZ/SO
C..... CHECK DIRECTIONAL COSINES
CALL CHECK(6,CNOR,ALN,BEN,GAN)
CALL DEBUG(6,CNOR)
RETURN
END

```



```

SOURCE TYPE SOURCE REF(ILL)
COMMON /66/ ARCO(20,200) .ARAY(500) .DEAR(500)
1 .CCCACS(6,50) .E(40,200) .EEISE(200) .ESOR(20,200)
2 .GARAY(500) .RED(20,200) .TITLE(12) .VRAT(20)
3 .VOL(200) .XLAN(200) .XRAY(500) .YRAT(500)
4 .ZRAY(500)
COMMON /ARRAYS/ ALK(200) .ALF(40) .BEK(200)
1 .BETA(40) .DEV(200) .DK(200) .EK(200)
2 .ELAN(200) .ELANAT(20) .ENAT(20) .ENERGY(200)
3 .FAK(200) .GAMMA(40) .REF(20) .SOMU(20)
4 .SUNRAY(200) .SUNSOR(200) .X(40) .XK(200)
5 .Y(40) .YK(200) .Z(40) .ZK(200)
6 .ZLAN(20)
COMMON /IARRAY/ IFLAG(40) .IPOI/IT(50)
1 .ISAX(50) .ISOR(20) .JF(200) .JFL(200)
2 .JL(200) .JM(200) .JN(200) .JP(200)
3 .JRAY(200) .JSOR(50) .KNO(900) .KFLAG(200)
4 .KP(200) .KTYP(200) .LAMP(50) .LANGLE(20)
5 .LREFL(200) .LTYPE(20) .MAT(200) .NFL(20)
6 .MAY(20) .MURRAY(200)
COMMON /PARAMS/ AL .ALN .ALR .ALRF .BE .
1 .BER .BERF .OIS .OMIN .OL .EPS
2 .GA .GAN .GAR .GARF .PER .
3 .SARA .XI .XAN .XOLD .XREF .XTRAN .VI
4 .YNC .YOLD .YREF .YTRAN .ZI .
5 .ZNC .ZOLD .ZREF .ZTRAN
COMMON /PARAM/ FMAX .IMAX .IMON .
1 .JOPAY .JOPAYT .IR .IRAY .JMAX .JMON .JOLD
2 .INT .IPAY .KNO .KMAX .KMON .KSOR
INTEGER F,FMAX
***** SET PERCENT OF TRANSMISSION TO AN INPUT CONSTANT.
FEP=1.0-RED(ILL,1)
RETURN
END

```

```

SUBROUTINE CONNOR(K)
C*****
C***** SUBROUTINE CONNOR CALCULATES THE DIRECTIONAL COSINES OF THE
C***** NORMAL TO A POINT ON THE SURFACE OF A CONIC.
C*****
COMMON /C9/ ABCO(20,200) ,ALRAY(500) ,BERAY(500)
1 ,COORDS(2,50) ,E(40,200) ,E9ASE(200) ,ESOR(20,200)
2 ,GARAY(500) ,RED(20,200) ,TITLE(12) ,VRAT(20)
3 ,VOL(200) ,XLAM(200) ,XINAY(500) ,YRAY(500)
4 ,ZRAY(500)
COMMON /ARRAYS/ ALK(200) ,ALPHA(40) ,BEK(200)
1 ,META(40) ,DEV(200) ,DK(200) ,EK(200)
2 ,ELAM(200) ,ELAMAT(20) ,ENAT(20) ,ENERGY(200)
3 ,GAK(200) ,GAMMA(40) ,REF(20) ,SORPAT(20)
4 ,SINRAY(200) ,SUNSOR(200) ,X(40) ,XK(200)
5 ,Y(40) ,YK(200) ,Z(40) ,ZK(200)
6 ,ZLAM(20)
COMMON /IARRAY/ IFLAG(40) ,IPOINT(50)
1 ,ISMAX(50) ,ISOR(20) ,JF(200) ,JFL(200)
2 ,JL(200) ,JM(200) ,JN(200) ,JP(200)
3 ,JRAY(200) ,JSOR(50) ,KBAO(900) ,KFLAG(200)
4 ,KP(20) ,KTYP(200) ,LAMP(50) ,LANGLE(20)
5 ,LPFL(200) ,LTYPE(20) ,MAT(200) ,PFL(20)
6 ,PMRAY(200)
COMMON /PARAMS/ AL ,ALN ,ALR ,ALRF ,RE
1 ,GA ,GAN ,GAR ,GANF ,PER ,DZ ,FPS
2 ,SINRA ,XI ,XINOW ,XCLO ,XREF ,XTRAA ,YI
3 ,YNO ,YOLD ,YREF ,YTRAN ,ZI
4 ,ZNO ,ZOLD ,ZREF ,ZTRAN
COMMON /IPARAM/ FMAX ,IRAX ,IRAY ,JMAX ,JRAY ,JNO ,JOLD
1 ,IOPRAY ,IOPRAYT ,IR ,IRAY ,JMAX ,JNO ,JOLD
2 ,KINT ,KMAX ,KNOW ,PMAX ,XNO ,XSOR
INTEGER F,FMAX
C***** CALCULATE LENGTH AND COSINES OF LINE FROM FOCUS TO POINT.
DX=XNO-XK(K)
DY=YNO-YK(K)
R=SQRT(DX**2+DY**2)
ALL=DX/R
DEL=DY/R
CALL CHECK(6*COMMON,ALL,DEL,0.0)
C***** CALCULATE COS AND SIN OF ANGLE FROM AXIS TO LINE FROM
C***** FOCUS TO POINT ON SURFACE.
COSTH=ALK(K)*ALL+BEK(K)*DEL
SINTH=ALK(K)*DEL-BEK(K)*ALL
DEN=SQRT(1.0+EK(K)**2-2.0*EK(K)*COSTH)
C***** CALCULATE COSINES OF NORMAL TO POINT ON SURFACE.
ALN=(ALK(K)*COSTH-BEK(K)*SINTH-EK(K)*ALK(K))/DEN
BEA=(ALK(K)*SINTH-BEK(K)*COSTH-EK(K)*BEK(K))/DEN
GAA=0
C***** CHECK DIRECTIONAL COSINES.
CALL CHECK(6*COMMON,ALN,BEA,GAA)
CALL DEBUG(6*COMMON)
RETURN
END

```

```

SUBROUTINE CYL
C.....
C..... CYL HANDLES CASES WHERE COSTAN IS SMALL.
C..... PART OF HELIX INTERSECTION CALCULATION.
C.....
COMMON /69/      ABCO(20,200)  •ALRAY(500)  •DERAY(500)
1 •COORDS(6,50)  •E(40,200)    •EBASE(200)  •ESOR(20,200)
2 •GARAY(500)    •RED(20,200)  •TITLE(12)  •VMAT(20)
3 •VOL(200)      •XLAN(200)    •XRAY(500)  •YRAY(500)
4 •ZRAY(500)
COMMON /ARRAYS/  ALK(200)      •ALPHA(40)    •REK(200)
1 •PETA(40)      •DEV(200)    •DK(400)    •EK(200)
2 •ELAN(200)     •ELANAT(20)  •EMAT(20)   •ENERGY(200)
3 •GAK(200)      •GAPHA(40)   •REF(20)    •SOMMAT(20)
4 •SIMRAY(200)   •SUMSO(200)  •X(40)      •XK(200)
5 •Y(40)         •YK(200)    •Z(40)
6 •ZLAN(20)
COMMON /IARRAY/  IFLAG(40)    •IPOINT(50)
•ISMAX(50)      •ISOR(20)    •JF(200)    •JFL(200)
•JL(200)        •JM(200)    •JA(200)    •JP(200)
•JRAY(200)      •JSOR(50)   •KEND(900)  •KFLAG(200)
•KP(200)        •KTP(200)   •LAMP(50)   •LANGLE(20)
•LRFFL(200)     •LTYPE(20)  •MAT(200)   •NFL(20)
•MMAT(20)      •MURRAY(200)
COMMON /PARAMS/  AL          •ALN        •ALR        •ALRF       •RE
•BEA          •BER        •BERF       •DIS        •DMIN       •DZ         •FPS
•GA          •GAN        •GAP        •GAW        •GFW        •PEN
•SUMMA       •XI         •XAOH       •XOLD       •XREF       •XTRAN      •VI
•YNOH       •YOLD       •YREF       •YTHAN     •ZI
•ZNOH       •ZOLD       •ZREF       •ZTRAN
COMMON /IPARAM/  FMAX        •INAX       •INON
•IOPRAY      •ICORNT      •IR         •IRAY       •JMAX       •JON
•KINT        •KMAX       •KAGH       •KMAX       •KROH       •KSOR
INTEGER F,FMAX
COMMON /HELCON/ A,ABAR,0.01,BIP,BP,CAP,DELTH,DELX,DELY,DELZ,DHIT,
•PSIZ,REV,RND,STH,TH,THI,SO,K5,RK5,CSTH,SATH,MCSTH,RSATH,BSTH,
•COSTAN,THZ,DTHZ
12 CONTINUE
C..... CALCULATE STH AND BSTH
CALL SOFTH
C..... CALCULATE TH AND DELTH
CALL THOFS
C..... IF ITERATION HAS NOT CONVERGED, GO TO 12
IF (ABS(DELTH).GT..01) GO TO 12
DELS=SQRT((A**2-BSTH)/(1.-COSTAN**2))
STHC=STH
IF (STHC-2.*DELS.LT.0.) RETURN
IF (STHC-2.*DELS.GT.DHIT) RETURN
STH=STHC-DELS
C..... CALCULATE INTERSECTION DISTANCE.
CALL RCYL
IF (STH.LT.0) GO TO 14
IF (STH.GT.DHIT) RETURN
DHIT=STH
RETURN
14 CONTINUE
STH=STHC-DELS
C..... FIND OTHER INTERSECTION
CALL RCYL
IF (0..LT.STH .AND. STH.LT.DHIT) DHIT=STH
RETURN
END

```

```

SUBROUTINE CYLNOR(K)
C*****
C***** CYLNOR CALCULATES DIRECTIONAL COSINES OF THE NORMAL TO A
C***** POINT ON THE SURFACE OF A CYLINDER.
COMMON /69/ ABCO(20,200) ALRAY(500) DERAY(500)
1 COORDS(6,50) E(40,200) EBASE(200) ESOR(20,200)
2 GARAY(500) RED(20,200) TITLE(12) VBAT(20)
3 VOL(200) XLAR(200) XRAY(500) YRAY(500)
4 ZRAY(500)
COMMON /ARRAYS/ ALK(200) ALPHA(40) BEK(200)
1 RETA(40) DEV(200) DK(200) EK(200)
2 ELAM(200) ELAMAT(20) EMAT(20) ENERGY(200)
3 GAK(200) GAMMA(40) REF(20) SOMMAT(20)
4 SUMRAY(200) SUMSOR(200) X(40) XK(200)
5 Y(40) YK(200) Z(40) ZK(200)
6 ZLAM(20)
COMMON /IARRAY/ IFLAG(40) IPOINT(50)
1 ISMAX(50) JF(200) JFL(200)
2 JL(200) JH(200) JH(200) JP(200)
3 JRAY(200) JSOR(50) KNO(900) KFLAG(200)
4 KRF(200) KTYPI(200) LAMP(50) LANGLE(20)
5 LREFL(200) LTYPE(20) MAT(200) MFL(20)
6 MNMAT(20) MNRAY(200)
COMMON /PARAMS/ AL ALR ALWF BE
1 BEN DER DERF DIS DMIN DZ EPS
2 GA GAN GAR GARF PER
3 SINGA XI XAON XOLD XREF XTRAK VI
4 YND YOLD YREF YTHAN ZI
5 ZND ZOLD ZREF ZTRAK
COMMON /IPARAM/ FMAX IMAX INOW
1 IOPRAY IOPRNT IN IRAY JMAX
2 INT KMAX KNOW NMAX NNOV NSOR JOLD
INTEGER F,FMAX
C***** CALCULATE COSINES OF VECTOR FROM POINT ON CYLINDRICAL AXIS
C***** TO POINT ON THE SURFACE OF THE CYLINDER.
DELX=XAON-XK(K)
DELY=YAON-YK(K)
DELZ=ZAON-ZK(K)
CC=DELX*ALK(K)+DELY*BEK(K)+DELZ*GAK(K)
C***** CALCULATE COSINES OF NORMAL TO POINT ON SURFACE.
ALN=(DELX-CC*ALK(K))/DK(K)
BEA=(DELY-CC*BEK(K))/DK(K)
GAA=(DELZ-CC*GAK(K))/DK(K)
C***** CHECK DIRECTIONAL COSINES.
CALL CHECK(6,CYLNOR,ALN,BEA,GAA)
CALL DEBUG(6,CYLNOR)
RETURN
END

```

```

      SUBROUTINE LOCALCIN
      C*****
      C***** LOCALC DETERMINES WHICH PART OF THE COIL TO LOOK IN AND
      C***** WHICH TECHNIQUE TO USE TO CALCULATE THE INTERSECTION.
      C***** PART OF HELIX INTERSECTION CALCULATION.
      C*****
      COMMON /69/      ISCO(20,200)      ALRAY(500)      REAY(500)
      1  CCOAT(50,50)  E(40,200)      EBASE(200)      ESDR(20,200)
      2  RARAY(500)    RED(20,200)      TITLE(12)      VMAT(20)
      3  VTL(200)      PLAM(200)      ARAY(500)      VTHAT(500)
      4  VTA(500)
      COMMON /ARRAYS/  ALK(200)      ALPHA(40)      BEK(200)
      1  AETA(40)      DEK(200)      DK(200)      EK(200)
      2  ELAM(200)      ELAMAT(20)      EMAT(20)      ENERGY(200)
      3  GAK(200)      GAMMA(40)      REF(20)      SUBMAT(20)
      4  SUMRAY(200)    SUMSDR(200)      K(40)      XK(200)
      5  V(40)      VK(200)      Z(40)      ZK(200)
      6  ZLAM(20)
      COMMON /ARRAY/    IFLAG(40)      IPCINT(50)      .
      1  ISWAT(50)      ISDR(20)      JF(200)      JFL(200)      .
      2  JL(200)      JM(200)      JN(200)      JP(200)      .
      3  JRAY(200)      JSDR(50)      KEND(500)      KFLAG(200)      .
      4  KIP(200)      KTOP(200)      LAMP(50)      LANGLE(20)      .
      5  LHEFL(200)      LTT-E(20)      MAT(200)      MFL(20)      .
      6  MMAT(20)      MMUMAT(200)
      COMMON /PARAM/    AL      ALN      ALR      ALRF      RE      .
      1  DEN      REW      BEWF      DIS      DMIN      DZ      EPS      .
      2  GBI      GBN      GAR      GARF      PER      .
      3  SLOWA      RI      RND      ROLD      RREF      ATRAW      VI      .
      4  VNC      VOLD      VREF      VTHAN      ZI      .
      5  Z      ZOLD      ZREF      ZTHAN
      COMMON /PARAM/    FMAX      IMAX      INOW      .
      1  IORAY      IORNT      IR      IGAY      JMAX      JNOW      JCOLD      .
      2  KINT      KNT      KND      KMAX      KNDW      KNSCH
      INTERSE=F*F*AT
      COMMON/HELCO/ALABE,BUBLEIP,EP,CAP,DELTH,DELX,DELY,DELZ,DHIT,
      1  PS1Z,REY,RD,STH,TH,T1,SS,SWAS,CSTH,SATH,CSTH,SATH,STH
      C*****
      C***** CALCULATE PARAMETERS NEEDED.
      F=0
      RESER=0.2*CAP**2
      SORR=SGRT(R*5)
      PS1=PS1Z
      THZ=PS1Z*.5*REV
      DELZ=DELZ
      DELZ=DELZ*AN*REV*CAP
      F=0
      IDCF=0
      DT=Z*1.57
      12  CONTINUE
      TH1=THZ*DT*Z
      TH=TH1
      DELT=10.
      C***** CALCULATE COSTAN(COS OF ANGLE BETWEEN RAY AND TANGENT)
      CALL COSTAN
      DT=Z*DT*Z
      14  CONTINUE
      C***** CALCULATE S(TH).
      CALL SFTM
      IF(ABS(DELT)-.01) GO TO 145

```

```

R=(TH1-TH)/DTM2
IF(FTM.LT.2.*R) GO TO 21
IF(ABS(DELT).LT..001) GO TO 15
145 CONTINUE
***** CALCULATE TH(S).
CALL THOFS
GO TO 14
15 CONTINUE
IF(IDCF.EQ.0) GO TO 16
CALL ZCALC
GO TO 9
16 CONTINUE
IF(R.LT.-1. .OR.DTM2.EQ.0.) GO TO 9
GO TO 12
21 CONTINUE
***** SET SWITCH.
IS=6*IDCF+1
IF(ABS(COSTAN).GT..3) IS=IS+3
IF(R.GT.1.) IS=IS+2
IF(R.LT.-1.) IS=IS+1
IF(DTM2.LT.0.) IS=IS+12
***** BRANCH ON SWITCH.
GO TO ( 23,27,27,29,27,27,25,23,22,24,
      22,22,24,23,22,26,22,12,15,
      1 2 3 4 5 6 7 8 9 0
C 15W R COSTAN IDCF DTM2 STAT I D E C I S I O N T A B L E
C 1 0 SMALL 0 0 23 CYL GO TO 9
C 2 -1 SMALL 0 0 27 GO TO 9
C 3 +1 SMALL 0 0 27 GO TO 9
C 4 0 LARGE 0 0 25 CHANGE SIGN,ZCALC,SET
C 5 -1 LARGE 0 0 27 GO TO 9
C 6 +1 LARGE 0 0 27 GO TO 9
C 7 0 SMALL 1 0 25 CYL,ZCALC GO TO 9
C 8 -1 SMALL 1 0 23 CYL GO TO 9
C 9 +1 SMALL 1 0 22 ZCALC GO TO 9
C 10 0 LARGE 1 0 28 ZCALC (BOTH SIDES ARE
C 11 -1 LARGE 1 0 22 ZCALC GO TO 9
C 12 +1 LARGE 1 0 22 ZCALC GO TO 9
C 13 0 SMALL 0 - 24 CYL GO TO 12
C 14 -1 SMALL 0 - 23 CYL GO TO 9
C 15 +1 SMALL 0 - 12 GO TO 01
C 16 0 LARGE 0 -2L 26 SET IDCF
C 17 -1 LARGE 0 -2L 22 ZCALC (WITH TH,LT,TH2
C 18 +1 LARGE 0 -2L 12
C 19 0 1 - IMPOSSIBLE
22 CONTINUE
CALL ZCALC
GO TO 9
23 CONTINUE
CALL CYL
GO TO 9
24 CONTINUE
CALL CYL
GO TO 12
25 CONTINUE
CALL CYL
CALL ZCALC
GO TO 9
26 CONTINUE
IDCF=1
GO TO 12
27 CONTINUE
GO TO 9
28 CONTINUE
CALL ZCALC
29 CONTINUE
DTM2=-DTM2
CALL ZCALC
9 CONTINUE
DELT=DELTZ
RETURN
END

```

```

SUBROUTINE DEBUG(RES)
C*****
C***** SUBROUTINE DEBUG GIVES A DEBUG PRINT AND IS CALLED FROM
C***** NUMEROLS ROUTINES.
C*****
COMMON /66/      ABCD(20,200)  ,ALRAY(500)  ,BERAY(500)
1 ,CDORDS(50,50)  ,E(40,200)   ,E-USE(200)  ,ESCR(20,200)
2 ,GARY(500)     ,HED(20,200)  ,TITLE(12)   ,VPAT(20)
3 ,VOLI(200)     ,ALAM(200)    ,ANAY(500)    ,YNAT(500)
4 ,ZPAT(500)
COMMON /APRAYS/  ALN(200)      ,ALP-1(40)    ,BER(200)
1 ,RET(40)       ,DEN(200)     ,OK(200)      ,EN(200)
2 ,ELAM(200)     ,ELAMAT(20)   ,EMAT(20)     ,ENERGY(200)
3 ,GAF(200)      ,GAMMA(40)    ,REF(20)      ,SUMMAT(20)
4 ,SUM-AT(200)   ,SUMSCR(200)  ,A(40)       ,KK(200)
5 ,V(40)         ,VK(200)      ,Z(40)        ,ZK(200)
6 ,ZLAM(20)
COMMON /APRAY-AY/ IFLAG(40)    ,IPPOINT(50)  ,
,ISWAA(50)      ,JSCR(20)      ,JF(200)      ,JFL(200)  ,
,JL(200)        ,JM(200)      ,JA(200)      ,JP(200)  ,
,JMAY(200)      ,JSCR(50)     ,KEND(500)    ,KFLAG(200) ,
,KP(200)        ,KTP(200)     ,LAMP(50)     ,LANGLE(20) ,
,LREFL(200)     ,LTYPE(20)    ,MAT(200)     ,NFL(20)  ,
,OMAT(200)      ,OMMAY(200)
COMMON /PARMS/   AL          ,ALN          ,ALR          ,ALRF          ,RE          ,
,REN          ,REN          ,BERF          ,DIS          ,DMIN          ,DZ          ,EPS          ,
,SI          ,SI          ,GAR          ,GARF          ,PER          ,          ,          ,
,SUMMA          ,          ,AND          ,ACLD          ,XREF          ,ATRA          ,VI          ,
,YN          ,YN          ,YREF          ,YTHAN          ,ZI          ,          ,
,ZL          ,ZL          ,ZREF          ,ZTRAN          ,
COMMON /PAR-AY/  FMAX          ,IMAX          ,INOW          ,
,IPRAY          ,IPR          ,JMAX          ,JACH          ,JOLD          ,
,KNOW          ,KNOW          ,OMAX          ,ONOW          ,NSCH
INTEGER F,F-1,F-2
COMMON /-55/  FRAC          ,IDEGUS
IF (IDEGUS) 100,100,101
100 RETURN
101 CONTINUE
PRINT 10,RES
PRINT 10,DEBUG PRINT IN ,AB
PRINT 11,JOLD,JND,IND,IMAX,KNOW
11 FORMAT(1H ,6H JOLD=,112,6H JND=,112,6H IND=,112,
,6H IMAX=,112,6H KNOW=,112)
PRINT 12,AL,RE,SI,VI,ZI
12 FORMAT(1H ,6H AL=, E12.4,6H RE=, E12.4,6H SI=, E12.4,
,6H VI=, E12.4,6H ZI=, E12.4)
PRINT 13,ALN,BER,GAR,XREF,Y-REF,ZREF
13 FORMAT(1H ,6H ALN=, E12.4,6H BER=, E12.4,6H GAR=, E12.4,
,6H XREF=, E12.4,6H YREF=, E12.4,6H ZREF=, E12.4)
PRINT 14,ALRF,BERF,GARF,ATRA,YTHAN,ZTRAN
14 FORMAT(1H ,6H ALRF=, E12.4,6H BERF=, E12.4,6H GARF=, E12.4,
,6H ATRA=, E12.4,6H YTHAN=, E12.4,6H ZTRAN=, E12.4)
PRINT 15,ALN,REN,GAN,IND,YNOW,ZNOW
15 FORMAT(1H ,6H ALN=, E12.4,6H REN=, E12.4,6H GAN=, E12.4,
,6H IND=, E12.4,6H YNOW=, E12.4,6H ZNOW=, E12.4)
PRINT 16,DIS,DMIN,PER
16 FORMAT(1H ,6H DIS=, E12.4,6H DMIN=, E12.4,6H PER=, E12.4)
RETURN
END

```



```

SUBROUTINE DEPOS
C*****
C***** SUBROUTINE DEPOS DEPOSITES ENERGY IN THE SEGMENT WHICH
C***** HAS BEEN PASSED BY A RAY.
C*****
COMMON /69/ ARCC(28,266) VALRAY(500) BERAY(500)
1 .COSARCS(16,58) .E(16,266) .EBASE(200) .ESOR(20,266)
2 .SERAY(500) .RED(28,266) .TITLE(12) .VMAT(20)
3 .TOL(20) .XLAY(200) .XRAY(500) .YRAY(500)
4 .ZRAY(500)
COMMON /70/ ALK(200) ALPHA(40) BEK(200)
1 .BETA(40) .DE(200) .DN(200) .ER(200)
2 .ELAY(200) .ELAYAT(20) .EMAT(20) .ENERGY(200)
3 .GAMMA(40) .GAMMAT(40) .REF(20) .SURMAT(20)
4 .SUMRAY(200) .SUMSOR(200) .X(40) .XK(200)
5 .Y(40) .YK(200) .Z(40) .ZK(200)
6 .ZLAY(20)
COMMON /71/ IFLAG(40) IPCINT(50)
1 .ISOR(50) .JSC(20) .JF(200) .JFL(200)
2 .JL(200) .JM(200) .JN(200) .JP(200)
3 .JRAY(200) .JSC(50) .KEND(500) .KFLAG(200)
4 .KP(200) .KTYE(200) .LAMP(50) .LANGE(20)
5 .LREFL(200) .LTYPE(20) .MAT(200) .MFL(20)
6 .NMAT(200) .NMURAY(200)
COMMON /72/ AL ALR ALRF RE
1 .BER .BERF .DIS .C-IN .DZ .EPS
2 .GA .GAN .GAR .GARE .PER .
3 .SUGA .XI .XND .ACLO .REF .ATHAN .YI
4 .YND .YOLD .YREF .YTHAN .ZI
5 .ZND .ZOLD .ZREF .ZTRAN
COMMON /73/ FMAX INCH INCH
1 .JICRAY .JICPINT .IR .IRAY .JMAX .JND .JOLD
2 .KINT .KMAX .KND .KMAX .KND .KSOR
INTER FMAX
C***** SET SEGMENT INDEX.
JL=10
C***** SET MATERIAL INDEX.
NMAT(1)
C***** SET RAY INDEX.
I=IND
DO 102 F=1,FMAX
IF(ARCC(M,F))103,102,100
100 LOLD=E(I,F)
C***** CALCULATE NEW RAY ENERGY.
E(I,F)=ECLO*EXP(-ARCC(M,F)*DMIN)
C***** DEPOSIT ENERGY LOST BY RAY IN SEGMENT.
SUMRAY(J)=SUMRAY(J)+EOLD-E(I,F)
C***** GET SCALE FACTOR
CALL SCALE(M,XLAY,F,SCA)
C***** SCALE ENERGY DEPOSITED.
ELM(J)=ELM(J)+(ECLO-E(I,F))*SCA
102 CONTINUE
101 JFL(J)=I
MFL(M)=I
C***** LOCATE NUMBER OF RAYS PASSING THROUGH SEGMENT.
JRAY(J)=JRAY(J)+1
CALL DEBUG(M,DEPOS)
RETURN
END

```

```

SUBROUTINE DIFREFILL
C*****
C***** DIFREF CALCULATES DIFFUSE REFLECTION *****
C*****
COMMON /69/ ABCO(22,22) ALRAY(500) BERAY(500)
1 COORDS(6,50) E(40,200) EBASE(200) ESCO(20,200)
2 GARAY(500) RED(20,200) TITLE(12) VMAT(20)
3 VOL(200) XLAM(200) XRAY(500) YRAY(500)
4 ZRAY(500)
COMMON /ARRAYS/ ALK(200) ALP-A(40) BER(200)
1 META(40) DEV(200) DC(200) EK(200)
2 FLAM(200) ELAMAT(20) EMAT(20) ENERGY(200)
3 EAK(200) GAMMA(40) REF(20) SLMAT(20)
4 SUMRAY(200) SUMSOR(200) X(40) XA(200)
5 Y(40) YK(200) Z(40) ZK(200)
6 ZLAM(20)
COMMON /ARRAY/ IFLAG(40) IPPOINT(50)
1 ISMAX(50) ISOR(20) JF(200) JFL(200)
2 JL(200) JM(200) JN(200) JP(200)
3 JRAY(200) JSOR(50) KEND(500) KFLAG(200)
4 KP(200) KTOP(200) LAMP(50) LANGLE(20)
5 LREFL(200) LTYPE(20) MAT(200) PFL(20)
6 PMAT(20)
COMMON /PARAMS/ AL ALN ALR ALRF RE
1 BEN BER BERF DIS DMIA DZ EPS
2 GA GAN GAR GAMF PER
3 SUMRA XI XAN XOLD XREF XTRAN VI
4 YNCH YOLD YREF YTHAN ZI
5 ZNCH ZOLD ZREF ZTHAN
COMMON /IPARAM/ FMAX IMAX INGO
1 IOPRAY IOPENT IN IRAY JMAX JNCH
2 KINT KMAX KACH PMAX PNC PMSOR JOLD
INTEGER F,FMAX
C***** CALCULATE RANDOM DELX, DELY, AND DELZ FOR POINT.
101 DELX=2.0*RAND(0)-1.0
DELY=2.0*RAND(0)-1.0
IF(1.0-DELX**2-DELY**2)101,107,107
107 DELZ=SQRT(1.0-DELX**2-DELY**2)
IF(1.0-ALN**2)106,106,102
106 ALF=0.0
BER=1.0
GAP=0.0
GO TO 105
102 SQ=SQRT(1.0-ALN**2)
ALP=(1.0-ALN**2)/SQ
BER=-ALN*BER/SQ
GAP=-ALN*GAN/SQ
105 AL2=REF*GAP-REP*GAN
BER=ALP*GAN-ALN*GAP
GAC=ALN*REP-ALP*BER
C***** CALCULATE DIRECTIONAL COSINES FOR REFLECTED RAY
ALR=DELX*ALP+DELY*ALQ+DELZ*ALN
BER=DELX*BER+DELY*BER+DELZ*BER
GAR=DELX*GAR+DELY*GAR+DELZ*GAR
C***** GET COS ANGLE OF REFLECTED RAY WITH NORMAL
COSTH=ALR*ALN+GAR*GAN+BER*BER
C***** GET COS ANGLE OF INCIDENT RAY WITH NORMAL
COSANG=AL*ALN+BE*BER+GA*GAN
C***** IS REFLECTED RAY ON REFLECTION SIDE OF SURFACE
IF(COSTH<COSANG)104,104,103
C***** CHANGE DIRECTION OF REFLECTED RAY
103 ALP=-ALP
BER=-BER
GAR=-GAR
C***** CALCULATE PERCENT OF TRANSMISSION
104 PER=1.0-REFILL(1)
C***** CHECK DIRECTIONAL COSINES.
CALL CHECK(6,DIFREF,ALR,REHIGAN)
RETURN
END

```

```

SUBROUTINE DSUBR(START)
C.....
C..... DSUB STARTS THE LOG THE PROPER ANGLES TO ISOLATE AND
C..... FIND ALL ROOTS THAT MAY EXIST FOR COIL A.
C..... PART OF HELIX INTERSECTION CALCULATION.
C.....
COMMON /A59/ AFD(20,200) ALRAY(500) BEWAY(500)
1 .COEFFS(5,50) E(40,200) EBASE(200) ESCH(20,200)
2 .GARY(500) RED(20,200) TITLE(12) VMAT(20)
3 .VFL(200) XLIM(200) XRAY(500) YMAT(500)
4 .ZRAY(500)
COMMON /ARRAYS/ ALK(200) ALPHA(40) BEK(200)
1 .RETA(40) DEY(200) DK(200) EK(200)
2 .FLAM(200) ELAMAT(20) EMAT(20) ENEMGY(200)
3 .GAK(200) GAMMA(40) REF(20) SOMMAT(20)
4 .S-PRAY(200) SLWSCR(200) X(40) XK(200)
5 .Y(40) YK(200) Z(40) ZK(200)
6 .ZLAM(20)
COMMON /IARRAY/ IFLAG(40) IPOINT(50)
1 .ISCR(50) ISCR(20) JF(200) JFL(200)
2 .JL(200) JM(200) JA(200) JP(200)
3 .JRAY(200) JSCH(50) KRND(900) KFLAG(200)
4 .KPL(200) KTYPE(200) LAMP(50) LANGLE(20)
5 .LFFL(200) LTYPE(20) MAT(200) NFL(20)
6 .NMAT(20) NMURAY(200)
COMMON /PARAMS/ AL ALN ALR ALRF RE
1 .BEN BEN BER OIS CMIN DZ EPS
2 .GA GAN GAR GANF PER
3 .SUNRA XI AND ACLO XREF XTRAA VI
4 .YNCH YOLD YEFF YTHAN ZI
5 .ZOLD ZLFC ZTHAN
COMMON /IPARAM/ FMAX IMAX IMON
1 .IDPRAY ICPRNT IR IRAY JMAX JMON
2 .KINT KMAX KMON MMAX MNCU MSON
3 .JOLD
INTEGER F,FMAX
COMMON /HELCON/A,ABAR,B,B1,BIP,BP,CAP,DELTH,DELX,DELY,DELZ,DMIT,
1 .PS17,REFY,MNO,STH,T,TMI,SDRKS,RKS,CSTH,SNTH,RCSTH,RSNTH,BSTH,
2 .COSTAN,T-Z,DT-Z
101 M=0
C..... STORE PREVIOUS ANGLE.
102 T=TMI
B1=B
BIP=BP
C..... ADVANCE STEP COUNTER.
M=M+1
C..... CALCULATE NEW ANGLE.
T=T+DELTH
C..... CALCULATE B AND BP.
CALL BCALC
C..... IF SIGN REVERSAL CALL REVS.
IF (RDP(1)103,104,104)
103 CALL REVS
C..... IF MOVING AWAY FROM ROOT AND B GREATER THAN ABAR GO TO NEXT S
IF (DELTH*BP)106,106,105
105 IF (R-ABAR)106,106,111
C..... IF MOVING AWAY FROM ROOT, CALL INFL.
104 IF (DELTH*BP)108,108,107
107 CALL INFL(1FL)
C..... IF A CHANGE OF SIGN WAS FOUND, GO TO NEXT STEP(106).
IF (1FL)106,106,115
C..... IF B LESS THAN ABAR, GO TO NEXT STEP.
115 IF (R-ABAR)106,106,111
C..... IF ROOT IS BRACKETED, FIND ROOT.
108 IF (R-ABAR)*(B1-ABAR)109,109,106
109 CALL FINDR
C..... IF MORE STEPS, GO TO 102.
106 IF (M-STAR)102,111,111
111 CONTINUE
110 RETURN
END

```

FTAS,5A

05/29/71

```

SUBROUTINE EDIT
C.....
C..... SUBROUTINE EDIT PRINTS FINAL RESULTS.....
C.....
C.....
      COMMON /C9/      A=CP(20,200)      ,ALP(200)      ,BEP(200)
      1 ,CEPDS(6,50)    ,E(40,200)      ,EP(200)      ,ESCP(20,200)
      2 ,GAP(200)      ,MED(20,200)      ,TITLE(12)      ,YMAT(20)
      3 ,VEL(200)      ,ZLAN(200)      ,ZRAY(200)      ,YRAY(200)
      4 ,ZRAY(200)
      COMMON /JAHAY/    ALK(200)      ,ALPHA(40)      ,BEK(200)
      1 ,BETA(40)      ,DEV(200)      ,IK(200)      ,EK(200)
      2 ,ELAP(200)      ,ELAPAT(20)      ,EPAT(20)      ,ENERGY(200)
      3 ,GAM(200)      ,GAMMA(40)      ,HE(20)      ,SCMAT(20)
      4 ,SLM(200)      ,SLMSC(200)      ,X(40)      ,Xo(200)
      5 ,Y(40)      ,YK(200)      ,Z(40)      ,Zo(200)
      6 ,ZLAN(20)
      COMMON /JAHAY/    IFLAG(40)      ,IPRINT(50)
      1 ,ISW(20)      ,IF(200)      ,JFL(200)
      2 ,JN(200)      ,JN(200)      ,JN(200)
      3 ,JAT(200)      ,JSE(50)      ,KPAD(500)      ,KPLAS(200)
      4 ,K(200)      ,KTY(200)      ,LAMP(50)      ,LANGLE(20)
      5 ,LREFL(200)      ,LTYPE(20)      ,PAT(200)      ,PFL(20)
      6 ,PAT(20)      ,RUMRAY(200)
      COMMON /PARAMS/   AL      ,ALA      ,ALW      ,ALPF      ,BE      ,
      1 ,BEH      ,BERF      ,EIS      ,DMIN      ,DZ      ,EPS      ,
      2 ,GA      ,GAN      ,GAR      ,GARF      ,PEF      ,
      3 ,SLMGA      ,XI      ,XNSH      ,XFLD      ,XREF      ,XTRAN      ,YI      ,
      4 ,YAG      ,YELD      ,YEF      ,YTRAA      ,ZI      ,
      5 ,ZAG      ,ZELD      ,ZEF      ,ZTRAA
      COMMON /PI-AM/    FMAX      ,IMAX      ,INBW      ,
      1 ,IEPRAY      ,IMPACT      ,JP      ,JPAY      ,JMAX      ,JNBW      ,
      2 ,JAT      ,KMAX      ,KNEW      ,PPAY      ,PMAX      ,JNBW      ,JNSH      ,
      3 ,JNBW      ,JNSH
      INTEGER F,IMAX
C..... CALCULATE TOTAL ENERGY IN SYSTEM,
      DIV=ELAT(IJAY)/(IJAY+1)
      ALX=0
C..... PRINT PAGE HEADING,
      PRINT 20
      20 FORTAT(1=1,53A,15=SEGMENT RESULTS/)
C..... CALCULATE TOTAL ENERGY DEPOSITED,
      SLME=0.0
      SLMLAN=0.0
      DO 100 J=1,JMAX
      SLMLAN=SLMLAN+ELAN(J)
      100 SLME=SLME+ENERGY(J)
C..... CALCULATE PERCENT ERROR FOR EACH SEGMENT,
      DO 120 J=1,JMAX
      IF(RUMRAY(J)-J)121,121,122
      121 DEV(J)=100.0
      GO TO 120
      122 IF(ENERGY(J)-1,E=0)121,121,126
      126 DEV(J)=SQRT(DIV*(SLMSC(J)-(ENERGY(J)**2)/RAY))/ENERGY(J)
      120 CONTINUE
C..... PRINT PAGE HEADING,
      PRINT 10
      10 FORTAT(1=0,6=2TH,2X,6=CALC,2X,6=MATERIAL,1X,5=TOTAL,4X,

```


85129172

C52

```

SUBROUTINE ELLNDR(K)
C*****
C***** ELLNDR CALCULATES DIRECTIONAL COSINES OF NORMAL
C***** TO A POINT ON AN ELLIPSOID OF REVOLUTION.
C*****
COMMON /69/ ABCO(20,200) ,ALRAY(500) ,BERAY(500)
1 ,COORDS(6,50) ,E(40,200) ,EBASE(200) ,ESOR(20,200)
2 ,GARAY(500) ,RED(20,200) ,TITLE(12) ,YMAT(20)
3 ,VOL(200) ,XLAM(200) ,XRAY(500) ,YRAY(500)
4 ,ZRAY(500)
COMMON /ZARRAYS/ ALK(200) ,ALPHA(40) ,BEK(200)
1 ,BETA(40) ,DEV(200) ,DK(200) ,EK(200)
2 ,ELAM(200) ,ELAMAT(20) ,EMAT(20) ,ENERGY(200)
3 ,GAK(200) ,GAMMA(40) ,REF(20) ,SGRMAT(20)
4 ,SUMRAY(200) ,SUMSCR(200) ,X(40) ,YK(200)
5 ,Y(40) ,YK(200) ,Z(40) ,ZK(200)
6 ,ZLAM(20)
COMMON /IARRAYS/ IFLAG(40) ,IPOINT(50)
1 ,ISMAX(50) ,ISCR(20) ,JF(200) ,JFL(200)
2 ,JL(200) ,JM(200) ,JM(200) ,JP(200)
3 ,JRAY(200) ,JSCR(50) ,KEND(500) ,KFLAG(200)
4 ,KP(200) ,KTP(200) ,LAMP(50) ,LANGE(20)
5 ,LREFL(200) ,LTYPE(20) ,MAT(200) ,NFL(20)
6 ,NMAT(20) ,NUMRAY(200)
COMMON /PARAMS/ AL ,ALN ,ALR ,ALRF ,RE
1 ,BEN ,BER ,BERF ,DIS ,DMIN ,DZ ,EPS
2 ,G1 ,GAN ,GAR ,GARF ,PER ,
3 ,SOURA ,XI ,XNG ,XCLD ,XREF ,XTRAN ,VI
4 ,YAC ,YCLD ,YREF ,YTRAN ,ZI ,
5 ,ZAC ,ZCLD ,ZREF ,ZTRAN
COMMON /IPARAMS/ FMAX ,IMAX ,INOW ,
1 ,ICORAY ,ICORNT ,IR ,IRAT ,JMAX ,JNOW ,JOLD
2 ,KINT ,KMAX ,KNOW ,MMAX ,MMIN ,NSOR
INTEGER F,FMAX
C***** CALCULATE X,Y,Z COMPONENTS OF NORMAL VECTOR.
DELXA=(XAC-X(K))/DK(K)
DELYA=(YAC-Y(K))/DK(K)
DELZA=(ZAC-Z(K)-EK(K))/EK(K)**2
C***** CALCULATE LENGTH OF NORMAL VECTOR
SQ=SQRT(DELXA**2+DELYA**2+DELZA**2)
C***** CALCULATE DIRECTIONAL COSINES OF NORMAL VECTOR.
ALN=DELXA/SQ
BEN=DELYA/SQ
GAN=DELZA/SQ
C***** CHECK DIRECTIONAL COSINES.
CALL CHECK(6NELLNDR,ALN,BEN,GAN)
CALL DEBUG(6NELLNDR)
RETURN
END

```

```

SUBROUTINE ELLNOR(K)
C*****
C***** ELLNOR CALCULATES DIRECTIONAL COSINES OF NORMAL
C***** TO A POINT ON AN ELLIPSOID OF REVOLUTION.
C*****
COMMON /A9/ ABCO(20,200) *ALP(500) *BERAY(500)
1 *CORNPS(6,50) *E(40,200) *EBASE(200) *ESOP(20,200)
2 *GPRAY(500) *RED(20,200) *TITLE(12) *VMAT(20)
3 *VOL(200) *XLAM(200) *XRAY(500) *YRAY(500)
4 *ZRAY(500)
COMMON /ARRAYS/ ALK(200) *ALPHA(40) *BEK(200)
1 *BETA(40) *DEV(200) *DK(200) *EK(200)
2 *FLAM(200) *ELAMAT(20) *EMAT(20) *ENERGY(200)
3 *GAK(200) *GAMMA(40) *REF(20) *SGRMAT(20)
4 *SINRAY(200) *SLWSCR(200) *X(40) *XK(200)
5 *Y(40) *YK(200) *Z(40) *ZK(200)
6 *ZLAM(20)
COMMON /IARRAY/ IFLAG(40) *IPOINT(50)
*ISMAX(50) *ISOR(20) *JF(200) *JFL(200)
*JL(200) *JM(200) *JN(200) *JP(200)
*JPAY(200) *JSOR(50) *KBD(900) *KFLAG(200)
*KP(200) *KTP(200) *LAMP(50) *LANGLE(20)
*LGFFL(200) *LTYPE(20) *MAT(200) *MFL(20)
*LMAT(20) *MMRAY(200)
COMMON /PARAMS/ AL *ALN *ALR *ALRF *RE
*BER *BERF *DIS *DMIN *DZ *EPS
*GAN *GAR *GARF *PER
*SUQA *XI *XND *XCLO *XPEF *XTRAN *YI
*YAC *YCLD *YREF *YTHAN *ZI
*ZAC *ZCLD *ZREF *ZTHAN
COMMON /IPARAM/ FMAX *IMAX *INOW
*JOPRAY *JOPRNT *IR *IRAY *JMAX *JNOW *JOLD
*KNAT *KNAX *KNOW *PMAX *PNOW *NSOR
INTEGER F,FMAX
C***** CALCULATE X,Y,Z COMPONENTS OF NORMAL VECTOR.
DELXA=(XND-X(K))/DK(K)
DELYA=(YAC-Y(K))/DK(K)
DELZA=(ZAC-Z(K)-EK(K))/EK(K)**2
C***** CALCULATE LENGTH OF NORMAL VECTOR.
SQ=SQRT(DELXA**2+DELYA**2+DELZA**2)
C***** CALCULATE DIRECTIONAL COSINES OF NORMAL VECTOR.
ALN=DELXA/SQ
BEN=DELYA/SQ
GAN=DELZA/SQ
C***** CHECK DIRECTIONAL COSINES.
CALL CHECK(6*ELLNOR,ALN,BEN,GAN)
CALL DEBUG(6*ELLNOR)
RETURN
END

```



```

SUBROUTINE ERRANT(RES)
C.....
C..... SUBROUTINE ERRANT PRINTS SUBROUTINE NAME WHERE ERROR OCCURRED
C.....
COMMON /A9/      A9C(20,200)  *ALRAY(500)  *BERAY(500)
1 *COPR(5,6,50)  *E(4,200)    *EBASE(200)  *ESCH(20,200)
2 *HARAY(500)    *REDI(20,200) *TITLE(12)   *VMAT(20)
3 *VOL(200)      *XLAM(200)   *XRAY(500)   *YPAY(500)
4 *YRAY(500)
COMMON /ARRAYS/  ALK(200)      *ALP-A(40)   *BEK(200)
1 *RETA(40)      *DEV(200)    *DK(200)   *EK(200)
2 *FLAM(200)     *ELAMAT(20)  *EMAT(20)  *ENERGY(200)
3 *GAK(200)      *GEMPA(40)   *PEF(20)   *SQMAT(20)
4 *SMRAY(200)    *SLUSCH(200) *X(40)     *XK(200)
5 *V(40)         *VK(200)     *Z(40)     *ZK(200)
6 *ZLAM(20)
COMMON /IARRAY/  IFLAG(40)    *IPOINT(50)  *
*ISWAT(50)      *ISOR(20)     *JF(200)   *JFL(200)  *
*JL(200)        *JM(200)     *JA(200)   *JP(200)   *
*JRAY(200)      *JSCR(50)    *KB(40,900) *KFLAG(200) *
*KP(200)        *KTP(200)    *LAMP(50)   *LANGLE(20) *
*LRFL(200)      *LTYPE(20)   *MAT(200)   *MFL(20)   *
*MMAY(20)       *NOMG(200)
COMMON /PPRAYS/  AL          *ALN          *ALR          *ALRF         *RE
*BER          *BERF         *DIS          *DMIN         *DZ          *EPS
*GAN          *GAR          *GANF         *PER          *
*SCMDA        *XI           *KCD          *KREF          *XTRAN        *YI
*YCD          *YCD          *YREF         *YTHAN         *ZI          *
*ZCD          *ZCD          *ZREF         *ZTRAN         *
COMMON /PPRINT/  FMAX        *IMAX          *INOW          *
*IDPRAY        *IDPRINT     *IR          *IRAY          *JMAX          *JNOW          *JOLD
*IRAY          *KMAX        *KNOW          *KRAY          *KNOW          *NSOW
INTEGER F,FMAX
C..... PRINT SUBROUTINE NAME WHERE ERROR OCCURRED.
PRINT 10,RES
10 FORMAT(1-1,13-ERROR IN RES ,A6)
C..... PRINT VALUES OF COMMON VARIABLES.
PRINT 11,JOLD,JNOW,INOW,IMAX,KNOW
11 FORMAT(1M ,6M JOLD=,I12,6M JNOW=,I12,6M INOW=,I12,
6M IMAX=,I12,6M KNOW=,I12)
PRINT 12,AL,BE,GA,XI,YI,ZI
12 FORMAT(1M ,6M AL=, E12.4,6M BE=, E12.4,6M GA=, E12.4,
6M XI=, E12.4,6M YI=, E12.4,6M ZI=, E12.4)
PRINT 13,ALR,BER,GAR,XREF,YREF,ZREF
13 FORMAT(1M ,6M ALR=, E12.4,6M BER=, E12.4,6M GAR=, E12.4,
6M XREF=, E12.4,6M YREF=, E12.4,6M ZREF=, E12.4)
PRINT 14,ALRF,BERF,GARF,XTRAN,YTRAN,ZTRAN
14 FORMAT(1M ,6M ALRF=, E12.4,6M BERF=, E12.4,6M GARF=, E12.4,
6M XTRAN=, E12.4,6M YTRAN=, E12.4,6M ZTRAN=, E12.4)
PRINT 15,ALNO,BEN,GAN,ANC,YNC,ZNC
15 FORMAT(1M ,6M ALNO=, E12.4,6M BEN=, E12.4,6M GAN=, E12.4,
6M ANC=, E12.4,6M YNC=, E12.4,6M ZNC=, E12.4)
PRINT 16,DIS,DMIN,PER
16 FORMAT(1M ,6M DIS=, E12.4,6M DMIN=, E12.4,6M PER=, E12.4)
C..... PRINT SEGMENT AND MATERIAL RESULTS.
CALL ECIT
STOP
END

```

```

SUBROUTINE FINCR
C.....
C..... FINCR FINDS THE ROOT AFTER IT HAS BEEN ISOLATED.
C..... PART OF RELIN INTERSECTION CALCULATION.
C.....
COMMON /69/      ABCO(20,200)  ,ALRAY(500)  ,BERAY(500)
1 ,COORDS(6,50)  ,E(40,200)    ,ERASE(200)  ,ESOR(20,200)
2 ,EARAY(500)    ,RED(20,200)  ,TITLE(12)   ,VMAT(20)
3 ,VFL(200)      ,XLAN(200)    ,XRAY(500)    ,YRAY(500)
4 ,ZRAY(500)
COMMON /ARRAYS/  ALK(200)      ,ALPHA(40)    ,BEK(200)
1 ,ETA(40)       ,DEV(200)     ,DK(200)     ,EK(200)
2 ,FLAM(200)    ,ELAMAT(20)   ,EMAT(20)    ,ENERGY(200)
3 ,EAT(200)     ,GAMMA(40)    ,REF(20)     ,SGRMAT(20)
4 ,SMRAY(200)   ,SUSOR(200)   ,X(40)      ,XK(200)
5 ,Y(40)        ,YK(200)     ,Z(40)      ,ZK(200)
6 ,ZLAN(20)
COMMON /TARRAY/  IFLAG(40)    ,IPPOINT(50)
1 ,ISMAX(50)    ,ISOR(20)     ,JF(200)     ,JFL(200)
2 ,JL(200)      ,JM(200)     ,JM(200)     ,JP(200)
3 ,JRY(200)     ,JSOR(50)    ,KBD(900)    ,KFLAG(200)
4 ,KP(200)      ,KTYP(200)   ,LAMP(50)    ,LANSLE(20)
5 ,LREFL(200)   ,LTYPE(20)  ,MAT(200)    ,PFL(20)
6 ,PMT(20)      ,PMRAY(200)
COMMON /PARAMS/  AL          ,ALN          ,ALR          ,ALRF          ,BE          ,
1 ,BEN          ,BER          ,DIS          ,DMIN          ,DZ          ,EPS          ,
2 ,GA          ,GAN          ,GAR          ,GANF          ,PER          ,
3 ,SUMRA          ,XI          ,XNOW          ,XOLD          ,XREF          ,XTRAN          ,YI          ,
4 ,YNC          ,YOLD          ,YHEF          ,YTRAN          ,ZI          ,
5 ,ZNC          ,ZOLD          ,ZREF          ,ZTRAN          ,
COMMON /IPAR1/   FMAX          ,IMAX          ,INOW          ,JNO          ,JOLD          ,
1 ,ICRAY          ,ICRNT          ,IR          ,IRAY          ,JMAX          ,JNO          ,JOLD          ,
2 ,KINT          ,KMAX          ,KNO          ,KMAX          ,KNO          ,KNO          ,
INTEGER F,FMAX
COMMON /ELCOM/ A,ABAR,B,BI,BIP,BP,CAP,DELTH,DELX,CELY,CELZ,DMIT,
1 ,PSIZ,REV,RND,STH,TN,TNI,SOAKS,RKS,CSTH,SATH,RCSTH,RSATH,BSTH,
2 ,CCSTAN,T4Z,DTHZ
ICFLAG=0
IC=0
C..... SAVE STARTING VALUES.
TMB=TM
B6=B
BAP=BP
TM7=TM1
B7=B1
B7P=BIP
C..... SET BOUNDING VALUES.
100 TMB=TM
B6=B
BAP=BP
C..... CALCULATE NEW ANGLE.
TM=TM8-(B6-ABAR)/BEP
C..... IF NEW ANGLE NOT BETWEEN TM7 AND TMB, GO TO 102.
IF((TM-TMB)*(TM-TM7))>0.101,101,102
102 TM=(TM7+TMB)/2.0
C..... CALCULATE NEW B AND BP.
101 CALL BCALC
B=(B-ABAR)/(B6-ABAR)
IC=IC+1
C..... IF MORE THAN 100 ITERATIONS, PRINT AND RETURN.
IF(IC-100)>100,100,111

```

```

111 P = 111.TM7.TM.TM.B.BP.BB.BBP
112 FOR I=1 TO 100: DO CONY IN FLAGR.(1P7E12.4)
    CALL REPORT(6-FINDR)
    GO TO 106
109 CONTINUE
    IF(112-111).GT.1E-12: GO TO 112
C..... IF CONVERGED, TRY ONE MORE ITERATION AND RETURN.
112 IF(ABS(112-111).GT.1E-12): GO TO 112
113 IF(112-111).GT.1E-12: GO TO 112
C..... CHOOSE BOUNCING ANGLE AND TRY AGAIN.
103 IF(112-111).GT.1E-12: GO TO 103
106 TM7=TM
    GO TO 100
C..... IF INTERSECTION DISTANCE IS NEGATIVE, DONT SAVE.
104 IF(STM).LT.0: GO TO 104
C..... IF INTERSECTION DISTANCE BETTER THAN PREVIOUS VALLE, SAVE IT.
107 DMIT=MIN(DMIT,STM)
108 TM7=6
    BP=BP
    RETURN
END

```

```

SUBROUTINE FLAREF(ILL)
COMMON /69/ ABCD(20,20) *ALPH(500) *BENAY(500)
1 *COSR(50) *E(40,200) *ESASE(200) *ESOR(20,200)
2 *ERAY(500) *RED(20,200) *TITLE(12) *PMAT(20)
3 *VNL(200) *XLAM(200) *XRAY(500) *YRAY(500)
4 *ZRAY(500)
COMMON /ARRAYS/ ALK(200) *ALPHA(40) *BEK(200)
1 *RETS(40) *DEV(200) *DK(200) *EN(200)
2 *ELAM(200) *ELAMAT(20) *EMAT(20) *ENERGY(200)
3 *GAK(200) *GAMMA(40) *REF(20) *SCMAT(20)
4 *SUMRAY(200) *SUMSSR(200) *X(40) *XK(200)
5 *Y(40) *YK(200) *Z(40) *ZK(200)
6 *ZL(20)
COMMON /JARRAY/ IFLAG(40) *JPCINT(50)
*ISWAX(50) *ISOR(20) *JF(200) *JFL(200)
*JL(200) *JP(200) *JN(200) *JP(200)
*JRAY(200) *JSCR(50) *KSN(900) *KFLAG(200)
*KP(200) *KTYP(200) *LAYR(50) *LANSLE(20)
*LRFL(200) *LTYPE(20) *MAT(200) *PFL(20)
*PMAT(20) *PMURAY(200)
COMMON /PARAMS/ AL *ALN *ALR *ALRF *RE
*GA *GER *GEFF *DIS *DMJA *CZ *EPS
*SUMRA *XI *XNG *GAF *PER *
*YNG *YCLD *YREF *YTAN *ZI *XTRAN *YI
*ZC *ZCLD *ZREF *ZTAN
COMMON /IPAR20/ FMAX *INCH *INCH
*ICPRAY *ICPRINT *IR *IRAY *JMAX *JACH *JOLD
*KL *KMAX *KACH *KMAX *KACH *KACH *KACH
INTEGER F,FMAX
C***** FIND COSINE OF ANGLE OF INCIDENCE.
COSTH=AL*AL+RE*BEN+GA*GAN
COSTH=COSTH**2
C***** CALCULATE PERCENT OF TRANSMISSION.
PER=RED(ILL)*(COSTH)
RETURN
END

```



```

***** CALCULATE COORDS OF UPPER LEFT CORNER OF PLOT AREA.
      XT=XT0-CX*AXX
      YT=YT0-CY*AXY
      ZT=ZT0-CZ*AAZ
      DELX=XT0-XZ=0
      DELY=YT0-YZ=0
      DELZ=ZT0-ZZ=0
      TL=SQRT(DELX**2+DELY**2+DELZ**2)
***** CALCULATE DIRECTIONAL COSINES OF VERTICAL RAYS.
      AX=DELX/TL
      AY=DELY/TL
      AZ=DELZ/TL
***** CALCULATE RAY INTERVALS
      DSX=XL/(NX-1)
      DSY=1.66667*DSX
      NY=(YL/DSY+1.5)
      AL=AX
      BE=AY
      GA=AZ
      JNC=1
      DO 102 N=1,NY
***** SET STARTING POINT FOR VERTICAL RAY NO. N.
      XCCL(N)=YT0+AXX*(N-1)*DSX
      YCCL(N)=YT0+AXY*(N-1)*DSX
      ZCCL(N)=ZT0+AAZ*(N-1)*DSX
      X1=XCCL(N)
      Y1=YCCL(N)
      Z1=ZCCL(N)
***** FIND SEGMENT CONTAINING STARTING POINT.
      CALL SEGNT
      IF(JNC)110,116,101
***** FIND FIRST BOUNDARY INTERSECTION
      101 CALL BOUND
***** STORE SEGMENT NUMBER AND DISTANCE TO INTERSECTION.
      JCCL(N)=JNC
      DCCL(N)=CMIN
      DO 103 K=1,KMAX
      RFLAG(K)=0
      103 CONTINUE
      102 CONTINUE
      NY=NY-I
***** DO ALL HORIZONTAL RAYS.
      DO 104 I=1,NY
      N1=0
***** SET STARTING POINT OF RAY.
      X1=XT0-(I-.5)*DSY*AXX
      Y1=YT0-(I-.5)*DSY*AXY
      Z1=ZT0-(I-.5)*DSY*AAZ
      AL=AXX
      BE=AXY
      GA=AAZ
      DL=0.0
      JNC=1
***** FIND SEGMENT CONTAINING POINT.
      113 CALL SEGNT
      IF(JNC)110,116,105
***** FIND BOUNDARY INTERSECTION.
      105 CALL BOUND
      DO 106 K=1,KMAX
      RFLAG(K)=0
      106 CONTINUE
***** SET PRINT CHARACTER FOR MATERIAL NUMBER.

```

```

      N=MOD(MAT(JNC)-1,34)+1
      MPRT=1/CHAR(M)
      DL=DL+DMIN
C***** IF DISTANCE TRAVELED GREATER THAN LENGTH OF RECTANGLE.
C***** GO TO 106.
      IF(DL-XL)107,107,108
C***** CALCULATE PLOT COORDINATES.
      107 XRAY=AX+(X1-XZERO)*ARY*(Y1-YZERO)/AAZ*(Z1-ZZERO)
      YRAY=AY+(Y1-YZERO)*AYY*(Y1-YZERO)/AAZ*(Z1-ZZERO)
C***** SET COLUMN NUMBER OF BOUNDARY INTERSECTION.
      NZ=(XRAY/CS)+1.5)
C***** SET PRINT CHARACTER TO BLANK.
      LINE(NZ)=1H
      NOPEN=N+1
      NT=NT+NZ-1
      IF(NT-NTNOM)109,109,112
C***** STORE MATERIAL NUMBER IN ALL PRINT POSITIONS BETWEEN
C***** CONSECUTIVE INTERSECTIONS.
      109 DO 111 N=OPEN,NTNOM
      LINE(N)=MPRT
      111 CONTINUE
      112 N1=NZ
C***** GO LOOK FOR NEXT INTERSECTION.
      GO TO 113
C***** STORE MATERIAL NUMBER FOR REMAINDER OF PRINT POSITIONS.
      108 NOPEN=N+1
      IF(NOPEN-N1)114,114,116
      114 DO 115 N=NOPEN,N1
      LINE(N)=MPRT
      115 CONTINUE
      116 AL=AYX
      BE=AYY
      GA=AYZ
C***** LOOP THROUGH ALL VERTICAL RAYS.
      DO 117 K=1,NX
C***** MOVE RAY STARTING POINT BY ONE LINE WIDTH.
      XCOL(N)=XCOL(N)+AL*DSY
      YCOL(N)=YCOL(N)+BE*DSY
      ZCOL(N)=ZCOL(N)+GA*DSY
      DCOL(N)=DCOL(N)+DSY
C***** IF INTERSECTION WAS NOT CROSSED GO TO 117.
      IF(DCOL(N))118,117,1.7
      118 X1=XCOL(N)
      Y1=YCOL(N)
      Z1=ZCOL(N)
      JNC=JCOL(N)
C***** FIND NEW SEGMENT NUMBER.
      CALL SEGMENT
      IF(JNC)119,119,119
C***** FIND NEXT BOUNDARY INTERSECTION.
      119 CALL BOUND
      DO 120 K=1,KMAX
      KFLAG(K)=0
      120 CONTINUE
C***** SET BLANK FOR PRINT POSITION.
      LINE(N)=1H
      DCOL(N)=DMIN
      117 CONTINUE
C***** PRINT LINE
      PRINT 10,LINE
      10 FORMAT(1H,13C1)
C***** GO DO NEXT HORIZONTAL RAY.
      104 CONTINUE
C***** GO LOOK AT NEXT DATA CARD.
      GO TO 123
      121 JNC=0
      RETURN
      110 CALL ERPRINT(6,MGEOPLOT)
      STOP
      END

```

```

SUBROUTINE MELNOR(
C.....
C..... MELNOR CALCULATES DIRECTIONAL COSINES FOR THE NORMAL TO
C..... A POINT ON THE SURFACE OF A HELIX.
C.....
COMMON /69/ ABCD(20,20) ,ALRAY(500) ,BERAY(500)
1 ,COORDS(6,50) ,E(40,200) ,EPASE(200) ,ESOR(20,200)
2 ,GARAY(500) ,RED(20,200) ,TITLE(12) ,VMAT(20)
3 ,VOL(200) ,XLAM(200) ,XRAY(500) ,YRAY(500)
4 ,ZRAY(500)
COMMON /ARRAYS/ ALK(200) ,ALPHA(40) ,BEK(200)
1 ,BETA(40) ,DEV(200) ,DK(200) ,EK(200)
2 ,ELAM(200) ,ELAMAT(20) ,EMAT(20) ,ENERGY(200)
3 ,GAK(200) ,GAMMA(40) ,REF(20) ,SLIPAT(20)
4 ,SUMRAY(200) ,SLMESH(200) ,XI(40) ,XK(200)
5 ,Y(40) ,YK(200) ,ZI(40) ,ZK(200)
6 ,ZLAM(20)
COMMON /IARRAY/ IFLAG(40) ,IPPOINT(50)
,ISMAX(50) ,ISOR(20) ,JF(200) ,JFL(200)
,JL(200) ,JM(200) ,JN(200) ,JP(200)
,JRAY(200) ,JSOR(50) ,KEND(900) ,KFLAG(200)
,KP(200) ,KTYPE(200) ,LAMP(50) ,LANGLE(20)
,LREFL(200) ,LTYPE(20) ,MAT(200) ,MFL(20)
,MNAT(20) ,MURRAY(200)
COMMON /PARAMS/ AL ,ALN ,ALR ,ALNF ,RE
,BEN ,BER ,BERF ,DIS ,DMIN ,DZ ,EPS
,GA ,GAN ,GAR ,GARF ,PER
,SUMRA ,XI ,XNO ,XCLO ,XREF ,XTRAN ,YI
,YNO ,YOLD ,YREF ,YTRAN ,ZI
,ZNO ,ZCLO ,ZREF ,ZTRAN
COMMON /IPARAM/ FMAX ,INAX ,INOW
,IOPRAY ,IOPRNT ,IR ,IRAY ,JMAX ,JNO ,JOLD
,KINT ,KMAX ,KNO ,KMAX ,KNO ,NSJ
INTEGER F,FMAX
C..... FIND POINT ON HELICAL WIRE CLOSEST TO POINT ON SURFACE.
CALL PCINT(K,ALPT,BEPT,GAPT,XNO,YNO,ZNO)
R=SQRT(ALPT**2+BEPT**2+GAPT**2)
C..... CALCULATE DIRECTIONAL COSINES OF NORMAL.
AL=ALPT/R
BE=BEPT/R
GA=GAPT/R
XNO=XNO-(R-DK(K))*ALN
YNO=YNO-(R-DK(K))*BEN
ZNO=ZNO-(R-DK(K))*GAN
CALL CHECK(6,MELNOR,ALN,BE,GA)
CALL DEBUG(6,MELNOR)
RETURN
END

```


C43

```

SUBROUTINE IRANK
C.....
C..... I-RANK GETS THE NEXT NEW RAY.
C.....
COMMON /69/ ABCD(20,200) *ALRAY(500) *BERAY(500)
1 *COORDS(16,50) *E(40,200) *EBASE(200) *ESCH(20,200)
2 *GAPAY(500) *REG(20,200) *TITLE(12) *VPAT(20)
3 *VOL(200) *XLAN(200) *XRAY(500) *YPAT(500)
4 *ZRAY(500)
COMMON /ARRAYS/ ALK(200) *ALPHA(40) *BEK(200)
1 *BETA(40) *DEV(200) *DK(200) *EK(200)
2 *ELAM(200) *ELAMAT(20) *EMAT(20) *ENERGY(200)
3 *GAK(200) *GAMMA(40) *REF(20) *SCRMAT(20)
4 *SIMRAY(200) *SUMSOR(200) *X(40) *XK(200)
5 *Y(40) *YK(200) *Z(40) *ZK(200)
6 *ZLAN(20)
COMMON /IARRAY/ IFLAG(40) *IPCINT(50)
*ISMAX(50) *ISOR(20) *JF(200) *JFL(200)
*JL(200) *JM(200) *JN(200) *JP(200)
*JRAY(200) *JSCR(50) *KEND(500) *KFLAG(200)
*KPD(200) *KTP(200) *LAMP(50) *LANGLE(20)
*LPFFL(200) *LTYPE(20) *MAL(200) *PFL(20)
*PMAT(20) *NUMRAY(200)
COMMON /PARAMS/ AL *ALN *ALR *ALRF *RE
*BER *BERF *D15 *DMIN *DZ *EPS
*GA *GAN *GAP *GANF *PER *
*SUMRA *XI *XAD *XCLD *XREF *XTRAN *VI
*YAD *YOLD *YREF *YTRAN *ZI
*ZAD *ZOLD *ZREF *ZTRAN
COMMON /IPARAM/ FMAX *IPIX *ILOW
*LOPRAY *LOPRNT *IP *IRAY *JMAX *JACH *JOLD
*KNAT *KNAX *KNOW *MPAX *MNOH *NSOR
INTEGER F,FMAX
COMMON /BASE/ FRAC *IDEBUG
DATA (COUNT=0)
C..... COUNT NUMBER OF RAYS FROM SOURCE NUMBER NNOH.
ICOUNT=ICOUNT+1
C..... IF ALL RAYS FOR SOURCE HAVE BEEN CREATED, GO TO NEXT SOURCE.
IF(ICOUNT-ISMAX(NNOH))101,101,100
C..... GO TO NEXT SOURCE.
100 NNOH=NNOH+1
ICOUNT=1
IF(NNOH-NSOR)101,101,110
101 IF(ISMAX(NNOH))100,100,105
C..... INCREASE RAY INDEX.
105 INAX=INAX+1
C..... GET LAMP NUMBER.
LT=LAMP(NNOH)
C..... GET SOURCE TYPE.
ISOR=ISOR(LT)
C..... BRANCH ON SOURCE TYPE.
GO TO(103,200,200,400)*ISOR
C..... GET RAY FROM VOLUME SOURCE.
103 CALL VSORCE
GO TO 104
C..... GET RAY FROM SURFACE SOURCE.
200 CALL SSORCE
GO TO 104
C..... GET RAY FROM INPUT TABLE.
400 CALL RAYSOR(ICOUNT)
GO TO 104

```

```

C***** STORE RAY ENERGY AND UPDATE TOTAL ENERGY AND RAY COUNT.
104 DO 106 F=1,FMAX
    E(IMAX,F)=ESCH(LT,F)
    ERASE(F)=ESCH(LT,F)*FRAC
    IF(ERASE(F))107,107,108
107 ERASE(F)=1.0
106 SUMRA=SUMRA+E(IMAX,F)
106 CONTINUE
    IFLAG(IMAX)=1
    IRAY=IRAY+1
C***** IF IOPRNT GREATER THAN 0, PRINT RAY PARAMETERS.
    IF(IOPRNT)110,110,101
101 PRINT 10,IRAY,Z(PWA(IMAX)),BETA(IMAX),GAMMA(IMAX),
    *X(IMAX),Y(IMAX),Z(IMAX)
10 FORMAT(1M,15-CREATED RAY NO.,16/4M AL=, E12.4,4M BE=, E12.4,
    *4M SA=, E12.4,3M X=, E12.4,3M Y=, E12.4,3M Z=, E12.4)
110 RETURN
    END

```

```

      FUNCTION INCN(K)
      C*****
      C***** FUNCTION INCN DETERMINES IF POINT IS INSIDE A CONE.
      C*****
      COMMON /69/      ABCD(20,200)      *ALRAY(500)      *BERAY(500)
      1 *C2ORCS(6,50)  *E(40,200)      *EBASE(200)      *ESOR(20,200)
      2 *CARAY(500)      *RED(20,200)      *TITLE(12)      *VMAT(20)
      3 *VOL(200)      *XLAM(200)      *XRAY(500)      *YHAY(500)
      4 *ZRAY(500)
      COMMON /ARRAYS/  ALX(200)      *ALPHA(40)      *BEK(200)
      1 *BETA(40)      *DEV(200)      *OK(200)      *EK(200)
      2 *ELAM(200)      *ELAMAT(20)      *EMAT(20)      *ENERGY(200)
      3 *GAK(200)      *GAMMA(40)      *REF(20)      *SCRMAT(20)
      4 *SUMRAY(200)      *SUMSOR(200)      *X(40)      *XK(200)
      5 *Y(40)      *YK(200)      *Z(40)      *ZK(200)
      6 *ZLAM(20)
      COMMON /IARRAY/      IFLAG(40)      *IPOINT(50)      *
      *ISMAX(50)      *ISOR(20)      *JF(200)      *JFL(200)      *
      *JL(200)      *JM(200)      *JN(200)      *JP(200)      *
      *JRAY(200)      *JSOR(50)      *KBND(900)      *KFLAG(200)      *
      *KP(200)      *KTP(200)      *LAMP(50)      *LANGLE(20)      *
      *LREFL(200)      *LTYPE(20)      *MAT(200)      *MFL(20)      *
      *MAT(20)      *NUMRAY(200)
      COMMON /PARAMS/      AL      *ALN      *ALR      *ALRF      *BE      *
      *BEX      *BER      *BERF      *DIS      *DMIN      *DZ      *EPS      *
      *GA      *GAN      *GAR      *GANF      *PER      *
      *SUNRA      *XI      *INCH      *XCLO      *XREF      *XTRAN      *YI      *
      *YND      *YOLD      *YREF      *YTRAN      *ZI      *
      *ZND      *ZOLD      *ZREF      *ZTRAN
      COMMON /IPARAM/      FMAX      *IMAX      *INOM      *
      *IOPRAY      *IOPENT      *IR      *IRAY      *JMAX      *JNOM      *JOLC      *
      *KIAT      *KNEX      *KNOW      *MMAX      *MNOW      *MSOR
      INTEGER F,FMAX
      DELZ=ZI-ZK(K)
      C***** IF POINT ON WRONG SIDE OF VERTEX GO TO 101.
      IF(DELZ*EK(K))101,101,103
      103 DELX=XI-XK(K)
      DELY=YI-YK(K)
      C***** PUT POINT INTO EQUATION OF CONE(FP).
      FP=(1.0-DK(K))*DELZ**2-DK(K)*(DELX**2+DELY**2)
      C***** IF FP LESS THAN 0, POINT IS OUTSIDE CONE.
      IF(FP)101,102,102
      C***** POINT IS INSIDE.
      102 INCN=1
      RETURN
      C***** POINT IS OUTSIDE.
      101 INCN=0
      RETURN
      END

```

```

      FUNCTION INCON(K)
      C*****
      C***** FUNCTION INCON DETERMINES IF A POINT(XI,YI,ZI) IS INSIDE
      C***** CONICAL BOUNDARY NUMBER K.
      C***** INCON=1 FOR YES.
      C***** INCON=-1 FOR NO.
      C*****
      COMMON /69/      ABCO(20,200)      ALRAY(500)      BERAY(500)
      1 *CODRES(6,50)  *E(40,200)      *EBASE(200)      *ESCR(20,200)
      2 *GARAY(500)    *REC(20,200)      *TITLE(12)      *VMAT(20)
      3 *VPL(200)      *XLA(1200)      *XRAY(500)      *YRAY(500)
      4 *ZRAY(500)
      COMMON /AFRAYS/   ALK(200)          ALPHA(40)        BEK(200)
      1 *RETA(40)      *DEV(200)          DK(200)          EK(200)
      2 *ELA(200)      *ELAT(20)          EMAT(20)          ENERGY(200)
      3 *GAK(200)      *GAMMA(40)        REF(20)          SGMAT(20)
      4 *SUMRAY(200)    *SUMSCH(200)      XI(40)           XK(200)
      5 *Y(40)         *YK(200)          ZI(40)           ZK(200)
      6 *ZLAN(20)
      COMMON /IARRAY/   IFLAG(40)        IPOINT(50)
      *IK=1X(50)      *ISCR(20)          *JF(200)          *JFL(200)
      *JL(200)        *JM(200)          *JA(200)          *JP(200)
      *JRAY(200)      *JSCR(50)         *KBND(900)        *KFLAG(200)
      *KZ(200)        *KTYP(200)        *LAMP(50)         *LANGLE(20)
      *LREFL(200)     *LTYPE(20)        *MAT(200)         *NFL(20)
      *NMAT(20)       *NMURAY(200)
      COMMON /PARAMS/   AL              ALN              ALR              ALRF              RE
      *BER              BERF              DIS              DMIN              DZ              EPS
      *GA              GAN              GAR              GARF              PER
      *SUNDA            XI              XND              XCOLD              XREF              XTRAN              YI
      *YND              YOLD              YREF              YTHAN              ZI
      *ZND              ZOLD              ZREF              ZTRAN
      COMMON /IPARAM/   FMAX              IMAX              INOW
      *IDPRAY           *ICPRNT          IR              IRAY              JMAX              JNOW              JOLD
      *KJAY             *KMAX            KND              KMAX              MNCH              MSON
      INTEGER F,FMAX
      C***** CALCULATE DISTANCE FROM FOCUS TO POINT.
      DX=XI-XK(K)
      DY=YI-YK(K)
      RR=SQRT(DI**2+DY**2)
      IF(RR-1.E-6)102,102,103
      C***** FIND COSINES OF VECTOR FROM FOCUS TO POINT.
      100 ALL=DX/RR
      BEL=DY/RR
      CALL CHECK(54 INCON,ALL,BEL,6.0)
      DEN=1.0-EK(K)*ALL*ALK(K)+BEL*BEK(K)
      IF(ABS(DEN)-1.E-6)102,102,101
      C***** FIND DISTANCE FROM FOCUS TO SURFACE.
      101 RAT=(DK(K)*EK(K))/DEN
      IF(RR-RAT)102,102,103
      C***** POINT IS INSIDE CONIC.
      102 INCON=1
      GO TO 104
      C***** POINT IS OUTSIDE CONIC.
      103 INCON=-1
      104 CONTINUE
      RETURN
      END

```

```

      FUNCTION INCYL(K)
      C*****
      C***** FUNCTION INCYL DETERMINES IF A POINT(XI,YI,ZI) IS INSIDE
      C***** CYLINDRICAL BOUNDARY NUMBER K.
      C***** INCYL=1 FOR YES.
      C***** INCYL=-1 FOR NO.
      C*****
      COMMON /69/      ABC(20,200)      ALRAY(500)      ELWAY(500)
      1  COORDS(6,50)  E(40,200)      EBASE(200)      ESCR(20,200)
      2  GARAY(500)    RED(20,200)      TITLE(12)      VMAT(20)
      3  VOL(200)      XLAM(200)      XRAY(500)      YRAY(500)
      4  ZRAY(500)
      COMMON /ARRAYS/  ALK(200)      ALPHA(40)      BEK(200)
      1  META(40)      DEV(200)      DK(200)      EK(200)
      2  FLAM(200)      ELAMAT(20)    EMAT(20)      ENERGY(200)
      3  GAK(200)      GAMMA(40)      REF(20)      SCRMAT(20)
      4  SMRAY(200)    SUNSOR(200)    X(40)      XK(200)
      5  Y(40)      YK(200)      Z(40)      ZK(200)
      6  ZLAM(20)
      COMMON /IARRAY/  IFLAG(40)      IPOINT(50)
      1  ISMAX(50)      JSOR(20)      JF(200)      JFL(200)
      2  JL(200)      JM(200)      JN(200)      JP(200)
      3  JRAY(200)      JSOR(50)      KEND(900)      KFLAG(200)
      4  KP(200)      KTOP(200)      LAMP(50)      LAAGLE(20)
      5  LRFFL(200)    LTYPE(20)      MAT(200)      MFL(20)
      6  MMAT(20)      MMURAY(200)
      COMMON /PARAMS/  AL      ALN      ALR      ALRF      RE
      1  BEA      BEB      BERF      DIS      DMIN      DZ      EPS
      2  GA      GAN      GAR      GAMF      PER
      3  SUMRA      XI      XND      ACLO      XREF      XTRAN      YI
      4  YND      YOLD      YREF      YTRAN      ZI
      5  ZND      ZOLD      ZREF      ZTRAN
      COMMON /IPARAM/  FMAX      IMAX      INOW
      1  IOGAY      IOPRNT      IR      IRAY      JMAX      JNOW
      2  KINT      KMAX      KND      MMAX      MNOW      NSOR      JOLD
      INTEGER F,FMAX
      C***** FIND DISTANCE FROM      CYLINDRICAL AXIS TO POINT.
      DX=XI-XK(K)
      DY=YI-YK(K)
      ZD=ZI-ZK(K)
      DD=DX**2+DY**2+ZD**2-(ALK(K)*DX+BK(K)*DY+GAK(K)*ZD)**2
      C***** IS DISTANCE LESS THAN RADIUS**2
      IF(DD-CK(K)**2)101,10,102
      C***** POINT IS INSIDE CYLINDER.
      101 INCYL =1
      RETURN
      C***** POINT IS OUTSIDE CYLINDER.
      102 INCYL=-1
      RETURN
      END

```

```

      FUNCTION INELL(K)
      C.....
      C..... FUNCTION INELL FINDS IF POINT IS INSIDE ELLIPSOID.
      C.....
      COMMON /69/      ABCO(20,200)      ALRAY(500)      BERAY(500)
      1  COMORS(6,50)    E(40,200)      EBASE(200)      ESOR(20,200)
      2  GARAY(500)      REC(20,200)      TITLE(12)      VMAT(20)
      3  VOL(200)        XLAM(200)      XRAY(500)      YRAY(500)
      4  ZRAY(500)
      COMMON /ARRAYS/   ALK(200)        ALPHA(40)      BEK(200)
      1  PFTA(40)        DEV(200)        DK(200)        EK(200)
      2  ELAM(200)      ELAMAT(20)      EMAT(20)      ENERGY(200)
      3  GAK(200)        GAMMA(40)      REF(20)        SCMAT(20)
      4  SUMRAY(200)     SUMSOR(200)     X(40)          XK(200)
      5  Y(40)          YK(200)        Z(40)          ZK(200)
      6  ZLAM(20)
      COMMON /IARRAY/   IFLAG(40)      IPOINT(50)
      1  ISX(50)          ISOR(20)      JF(200)        JFL(200)
      2  JL(200)          JM(200)        JN(200)        JP(200)
      3  JRAY(200)        JSOR(50)      KAND(900)      KFLAG(200)
      4  KIP(200)         KTOP(200)     LAMP(50)       LANGLE(20)
      5  LREFL(200)       LTYPE(20)     MAT(200)       MFL(20)
      6  MMAT(20)         MMURAY(200)
      COMMON /PARMS/     AL            ALN            ALR            ALRF            ME
      1  BEN            GER            GERF           DIS            CMIN           DZ            EPS
      2  GA            GEN            GAR            GAKF           PER            XCLD           XREF           XTRAN           VI
      3  YNC            YOLD           YREF           YTRAN          ZI
      4  ZNOW           ZOLD           ZREF           ZTRAN          INOW           JNOW           JOLD
      5  IOPRAY         IOPONT         IR            IRAY           JMAX           JMAX           JMAX           JMAX           JMAX
      6  KINT           KMAX           KMAX           KMAX           KMAX           KMAX           KMAX           KMAX           KMAX
      INTEGER F,FMAX
      C..... CALCULATE FUNCTION FP.
      FP=.0-((XI-X(K))**2+(YI-Y(K))**2)/DK(K)
      1  -((ZI-Z(K)-EK(K)/EK(K))**2
      C..... IF FP LESS THAN 0. POINT IS OUTSIDE
      IF(FP)102,101,101
      C..... POINT INSIDE
      101 INELL=1
      RETURN
      C..... POINT OUTSIDE
      102 INELL=-1
      RETURN
      END

```

```

SUBROUTINE INFL(IFL)
C.....
C..... INFL LOOKS FOR AN INFLECTION POINT.
C..... PART OF RELIN INTERSECTION CALCULATION.
C.....
COMMON /69/ ABCO(20,200) ,ALRAY(500) ,BERAY(500)
1 ,COORDS(6,50) ,E(40,200) ,EBASE(200) ,ESOR(20,200)
2 ,GARAY(500) ,RED(20,200) ,TITLE(12) ,VMAT(20)
3 ,VOL(200) ,XLAM(200) ,XRAY(500) ,YRAY(500)
4 ,ZRAY(500)
COMMON /ARRAYS/ ALK(200) ,ALPHA(40) ,BEK(200)
1 ,BETA(40) ,DEV(200) ,DK(200) ,EK(200)
2 ,ELAM(200) ,ELAMAT(20) ,EMAT(20) ,ENERGY(200)
3 ,GAK(200) ,GAPPA(40) ,REF(20) ,SDRMAT(20)
4 ,SUMRAY(200) ,SUMSQN(200) ,X(40) ,XK(200)
5 ,Y(40) ,YK(200) ,Z(40) ,ZK(200)
6 ,ZLAM(20)
COMMON /JARRAY/ IFLAG(40) ,IPPOINT(50)
1 ,ISMAX(50) ,JSOR(20) ,JF(200) ,JFL(200)
2 ,JL(200) ,JM(200) ,JN(200) ,JP(200)
3 ,JRAY(200) ,JSOR(50) ,KBND(900) ,KFLAG(200)
4 ,KP(200) ,KTYP(200) ,LAMP(50) ,LANGLE(20)
5 ,LREFL(200) ,LTYPE(20) ,MAT(200) ,NFL(20)
6 ,MMAT(20) ,MURAY(200)
COMMON /PARAMS/ AL ,ALN ,ALR ,ALRF ,RE
1 ,BEN ,BER ,BERF ,DIS ,DMIN ,DZ ,EPS
2 ,GA ,GAN ,GAR ,GANF ,PER
3 ,SUMHA ,XI ,XND ,XCLO ,XREF ,XTRAN ,YI
4 ,YND ,YOLD ,YHEF ,YTRAN ,ZI
5 ,ZND ,ZCLO ,ZREF ,ZTRAN
COMMON /IFPARAM/ FMAX ,IMAX ,INGD
1 ,IOPRAY ,IOPRNT ,IR ,IRAY ,JMAX ,JNGD ,JOLC
2 ,KINT ,KMAX ,KND ,MMAX ,MND ,MSOR
INTEGER F,FMAX
COMMON /HELCON/ A,ABAR,B,B1,BIP,EP,CAP,DELT,DELX,DELY,DELZ,D-IT,
1 ,PS17,REV,RND,STN,TH,TH1,SGRKS,SKS,CSTH,SNTH,RCSTH,RSNTH,BS-4,
2 ,COSTAN,THZ,OTHZ
IFL=0
C..... IF FUNCTION LOOKS LIKE A PARABOLA, GO TO 102.
1F((B-R1)/((HP-BIP)*(TH-TH1))-25)101,102,102
C..... IF ROOT IS BRACKETED, FIND ROOT.
102 1F((B-ABAR)*(B1-ABAR))103,103,110
103 CALL FINDR
GO TO 110
C..... SAVE STARTING VALUES.
101 TH2=TH
B2=B
B2P=BP
TH3=TH1
B3=B1
B3P=B1P
C..... CALCULATE NEW ANGLE
104 TH=(TH+TH1)/2.0
C..... CALCULATE NEW B AND BP.
CALL BCALC
C..... IF SIGN REVERSAL, CALL REVS TWICE.
1F(PI*BP)105,105,106
105 CALL REVS
IFL=1
TH1=TH
B1=B

```



```

      B1P=RP
      TM=TM2
      B=RP
      BP=RP
      CALL REVS
      GO TO 110
C..... IF FUNCTION LOOKS LIKE A PARABOLA, GO TO 107.
      106 IF((R-R1)/((BP-R1P)*(TM-TM1))-.25)10+.107.107
C..... IF ROOT IS BRACKETED, FIND ROOT
      107 IF((R-R1P)*(B1-ABAR))108.108.109
      108 CALL FINDER
C..... SET LIMITS FOR OTHER PART OF FUNCTION.
      109 T=1+TM
      B1=R
      B1P=RP
      TM=T 2
      B=R
      BP=RP
C..... IF FUNCTION LOOKS LIKE PARABOLA, GO TO 111.
      IF((R-R1)/((BP-R1P)*(TM-TM1))-.25)10+.111.111
C..... IF ROOT IS BRACKETED, FIND ROOT.
      111 IF((R-R1P)*(B1-ABAR))112.112.110
      112 CALL FINDER
      110 RETURN
      END

```

```

      FUNCTION INHEL(K)
      C*****
      C***** IF K DETERMINES IF POINT XI,YI,ZI IS INSIDE MELIX.
      C*****
      COMMON /I/      AECO(20,200)  *ALRAY(500)  *GERAY(500)
      1 *COCRS(1,50)  *E(40,200)    *EBASE(200)  *ESQ(20,200)
      2 *GARAY(500)   *RED(20,200)  *TITLE(12)   *VPAT(20)
      3 *VOL(200)     *XLAM(200)    *XRAY(500)   *YRAY(500)
      4 *ZRAY(500)
      COMMON /ARRAYS/  ALK(200)      *ALPHA(40)    *BEK(200)
      1 *PETA(40)      *DEV(200)     *DK(200)     *EK(200)
      2 *ELAM(200)     *ELAMAT(20)   *EMAT(20)    *ENERGY(200)
      3 *GAK(200)      *GAMMA(40)   *REF(20)     *SCRMAT(20)
      4 *SUMRAY(200)   *SUMSOR(200) *X(40)       *YK(200)
      5 *Y(40)         *YK(200)     *Z(40)       *ZK(200)
      6 *ZLAM(20)
      COMMON /IARRAY/  IFLAS(40)    *IPOINT(50)  *
      *ISMAX(50)      *ISCR(20)     *JF(200)    *JFL(200)  *
      *JL(200)        *JM(200)     *JA(200)    *JP(200)  *
      *JRAY(200)      *JSCR(50)    *KSD(500)   *KFLAG(200) *
      *KP(200)        *KTP(200)    *LAMP(50)   *LANGLE(20) *
      *LREFL(200)     *LTYPE(20)   *PAT(200)   *PFL(20)  *
      *MMAT(20)       *MMRAY(200)
      COMMON /PARAMS/  AL            *ALN        *ALR        *ALRF       *RE        *
      *BEN            *BER          *DIS        *DWIN      *DZ         *EPS        *
      *GA             *GAR          *GARF       *PER        *          *
      *SUMRA         *XI            *XAG        *XCLO      *XREF      *XTRAN     *YI        *
      *YMCV          *YELD         *YREF       *YTRAN     *ZI         *
      *ZMCV          *ZCLO         *ZREF       *ZTRAN     *
      COMMON /IPSEAL/  FMAX          *IMAX        *INOW        *
      *ICPRAY        *ICPANT       *IR          *IRAY       *JMAX       *JOLC      *
      *KINT          *KMAX         *KNOW       *JMAX       *JMAX       *JMAX
      INTEGER F,FMAX
      DELX=XI-XK(K)
      DELY=YI-YK(K)
      C***** CALCULATE DISTANCE FROM AXIS TO POINT.
      R=DELX**2+DELY**2
      RIA=(REK(K)-DK(K))**2
      ROUT=(PEK(K)-DA(K))**2
      C***** IF POINT NOT BETWEEN INNER AND OUTER RADIUS, GO TO 102.
      IF(R-RIA)102,103,103
      103 IF(R-ROUT)104,104,102
      104 CONTINUE
      C***** LOCATE NEAREST POINT ON MELICAL WIRE.
      CALL POINT(X,ALPT,BEPT,GAPT,XI,YI,ZI)
      C***** CALCULATE DISTANCE SQUARED BETWEEN TWO POINTS.
      DS=ALPT**2+BEPT**2+GAPT**2
      C***** IF DISTANCE SQUARED LESS THAN TUBE RADIUS SQUARED, GO TO 101.
      IF(DS-DA(K)**2)101,101,102
      C***** POINT IS INSIDE
      101 IN=EL=1
      RETURN
      C***** POINT IS OUTSIDE
      102 IN=EL=-1
      RETURN
      END

```

```

      FUNCTION INMYP(K)
      C*****
      C***** FUNCTION INMYP FINDS IF POINT IS INSIDE HYPERBOLIC.
      C*****
      COMMON /69/ ABCD(20,200) ,ALRAY(500) ,BEAT(500)
      1 ,COORDS(6,50) ,E(40,200) ,ESASE(200) ,ESOR(20,200)
      2 ,GARAY(500) ,RED(20,200) ,TITLE(12) ,VPAT(20)
      3 ,VCL(200) ,XLAM(200) ,XRAY(500) ,YRAY(500)
      4 ,ZRAY(500)
      COMMON /ARRAYS/ ALK(200) ,ALPHA(40) ,BEK(200)
      1 ,BETA(40) ,DEV(200) ,DK(200) ,EK(200)
      2 ,ELAM(200) ,ELAMAT(20) ,EMAT(20) ,ENERGY(200)
      3 ,GAK(200) ,GAMMA(40) ,REF(20) ,SCRMAT(20)
      4 ,SUMRAY(200) ,SUMSGR(200) ,X(40) ,XK(200)
      5 ,Y(40) ,YK(200) ,Z(40) ,ZK(200)
      6 ,ZLAP(20)
      COMMON /IARRAY/ IFLAG(40) ,IPOINT(50)
      ,ISMAX(50) ,JF(200) ,JFL(200)
      ,JL(200) ,JM(200) ,JN(200) ,JP(200)
      ,JRAY(200) ,JSOR(50) ,KBND(900) ,KFLAG(200)
      ,KP(200) ,KTYP(200) ,LAMP(50) ,LANGLE(20)
      ,LOFFL(200) ,LTYPE(20) ,MAT(200) ,MFL(20)
      ,NMAT(20) ,NUMRAY(200)
      COMMON /PLCAMS/ AL ,ALN ,ALR ,ALRF ,RE
      ,BER ,BERF ,DIS ,DMIN ,DZ ,EPS
      ,GA ,GAN ,GARF ,GARF ,PER
      ,SUORA ,XI ,XND ,XCLD ,XREF ,XTRAN ,YI
      ,YNCH ,YCLD ,YREF ,YTRAN ,ZI
      ,ZND ,ZCLD ,ZREF ,ZTRAN
      COMMON /IPARAM/ FMAX ,IMAX ,INOW
      ,IOPRAY ,ICORNT ,IN ,IRAY ,JMAX ,JNOW
      ,KIAT ,KMAX ,KAOW ,MMAX ,MNOW ,NSOR ,JOLD
      INTEGER F,FMAX
      C***** CALCULATE FUNCTION FP.
      DELZ=ZI-ZK(K)
      IF (DELZ*GAK(K))102,102,103
      103 DELX=XI-XK(K)
      DELY=YI-YK(K)
      FP=1.0-(DELX**2+DELY**2)/DK(K)+((DELZ*EK(K))**2)/(EK(K)**2)
      C***** IF FP LESS THAN 0, POINT IS OUTSIDE.
      IF (FP)102,101,101
      C***** POINT INSIDE
      101 INMYP=1
      RETURN
      C***** POINT OUTSIDE
      102 INMYP=0
      RETURN
      END

```

```

      FUNCTION IAPAR(K)
      C*****
      C***** FUNCTION IAPAR FINDS IF PCINT IS INSIDE PARASOLCID.
      C*****
      COMMON /69/      AHCO(20,200)      ALRAY(500)      BLRAY(500)
1  CCOORDS(6,50)      E(40,200)      EBASE(200)      ESCR(20,200)
2  FARAY(500)      RED(20,200)      TITLE(12)      VMAT(20)
3  VOL(200)      XLAM(200)      XRAY(500)      YHAY(500)
4  ZRAY(5,3)
      COMMON /70/      ALK(200)      ALPHA(40)      BEK(200)
1  RETA(40)      DEV(200)      DK(200)      EK(200)
2  ELAM(200)      ELAMAT(20)      EMAT(20)      ENRGY(200)
3  GAK(200)      GAMMA(40)      REF(20)      SGRMAT(20)
4  SUMRAY(200)      SUMSCR(200)      X(40)      XK(200)
5  Y(40)      YK(200)      Z(40)      ZK(200)
6  ZLAM(20)
      COMMON /71/      IFLAG(40)      IPOINT(50)
      ISMAX(50)      ISOR(20)      JF(200)      JFL(200)
      JL(200)      JM(200)      JN(200)      JP(200)
      JRAY(200)      JSOR(50)      KBND(900)      KFLAG(200)
      KP(200)      KTYP(200)      LAMP(50)      LANGLE(20)
      LREFL(200)      LTYPE(20)      MAT(200)      MFL(20)
      NMAT(20)      NUMRAY(200)
      COMMON /72/      AL      ALN      ALR      ALRF      BE
      BEN      BER      BERF      DIS      DMIN      DZ      EPS
      GA      GAN      GAR      GANF      PER
      SUMRA      XI      XANW      XCLO      XREF      XTRAN      YI
      YNO      YOLD      YREF      YTRAN      ZI
      ZNOY      ZOLD      ZREF      ZTRAN
      COMMON /73/      FMAX      IMAX      INOW
      IOFRAY      ICPRNT      IR      IRAY      JMAX      JACW      JOLD
      KINT      KMAX      KNO      KMAX      KNO      NSOR
      INTEGER F,FMAX
      C***** CALCULATE FUNCTION FP.
      FP=4.0*DK(K)*(ZI-ZK(K))-(XI-XK(K))*2-(YI-YK(K))*2
      C***** IF FP LESS THAN 0. POINT IS OUTSIDE
      IF (FP)101,102,102
      C***** POINT IS INSIDE.
      102 IAPAR=1
      RETURN
      C***** POINT IS OUTSIDE.
      101 IAPAR=-1
      RETURN
      END

```

```

      FUNCTION INPLAN(K)
      C*****
      C***** FUNCTION INPLAN DETERMINES IF A POINT(XI,YI,ZI) IS ON
      C***** THE ORIGIN SIDE OF PLANE BOUNDARY NUMBER K.
      C***** INPLAN=1 FOR YES.
      C***** INPLAN=-1 FOR NO.
      C*****
      COMMON /69/ ABCD(20,200) *ALRAY(500) *BERAY(500)
      1 *COORDS(16,50) *E(40,200) *EJASE(200) *ESOR(20,200)
      2 *GARAY(500) *RED(20,200) *TITLE(12) *VMAT(20)
      3 *VOL(200) *XLAN(200) *XRAY(500) *YRAY(500)
      4 *ZRAY(500)
      COMMON /ARRAYS/ ALK(200) *ALPHA(40) *BEK(200)
      1 *BETA(40) *DEV(200) *DK(200) *EK(200)
      2 *ELAM(200) *ELAMAT(20) *EMAT(20) *ENERGY(200)
      3 *GAK(200) *GAMMA(40) *REF(20) *SQRMAT(20)
      4 *SUMRAY(200) *SUMSOR(200) *X(40) *XK(200)
      5 *Y(40) *YK(200) *Z(40) *ZK(200)
      6 *ZLAN(20)
      COMMON /IARRAY/ IFLAG(40) *IPOINI(50)
      *ISMAX(50) *ISOR(20) *JF(200) *JFL(200)
      *JL(200) *JM(200) *JN(200) *JP(200)
      *JFZ(200) *JSOR(50) *KBND(900) *KFLAG(200)
      *KP(200) *KTYPE(200) *LAMP(50) *LANGLE(20)
      *LFL(200) *LTYPE(20) *MAT(200) *NFL(20)
      *MURAY(200)
      COMMON /PARAMS/ AL *ALN *ALR *ALRF *RE
      *BEN *BER *BERF *DIS *DMA *DZ *EPS
      *GA *GAN *GAR *GANF *PER
      *SUMRA *XI *XNOW *XOLD *XREF *XTRAN *YI
      *YNOW *YOLD *YREF *YTRAN *ZI
      *ZNOW *ZOLD *ZREF *ZTRAN
      COMMON /IPARAM/ FMAX *INAX *INOW
      *IDPRAY *IDPRAT *IR *IRAY *JMAX *JNOW
      *KINT *KMAX *KADN *MMAX *MNOW *NSOR *JOLD
      INTEGER F,FMAX
      C***** FIND DISTANCE FROM POINT TO PLANE.
      DD=ALK(K)*XI-BEK(K)*YI-GAK(K)*ZI-DK(K)
      IF(DD)101,102,100
      C***** POINT NOT ON ORIGIN SIDE.
      100 INPLAN=-1
      RETURN
      C***** POINT ON ORIGIN SIDE.
      101 INPLAN=1
      RETURN
      C***** POINT ON PLANE.
      102 CALL ERPRNT(69,INPLAN)
      RETURN
      END

```

```

      FUNCTION INSEG(J)
      .....
      ..... FUNCTION INSEG DETERMINES IF POINT(XI,VI,ZI) IS INSIDE
      ..... SEGMENT NUMBER J.
      ..... I=SEG=1 FOR YES.
      ..... I=SEG=0 FOR NO.
      .....
      COMMON /66/      AEC0(20,200)      ALRAY(500)      BLRAY(500)
      1 .COORDS(6,50)  .E(40,200)      .EBASE(200)      .ESCR(20,200)
      2 .GARAY(500)     .RED(20,200)     .TITLE(12)      .VMAT(20)
      3 .VPL(200)       .XLAM(200)       .XRAY(500)      .YRAY(500)
      4 .ZRAY(500)
      COMMON /ARRAYS/   ALK(200)         ALPHA(40)      BEK(200)
      1 .META(40)       .DEV(200)        .DK(200)        .EK(200)
      2 .FLAM(200)      .ELAMAT(20)      .EMAT(20)       .ENERGY(200)
      3 .GAK(200)       .GAMMA(40)       .REF(20)        .SCWMAT(20)
      4 .SUMRAY(200)    .SUMSCR(200)     .X(40)          .XK(200)
      5 .Y(40)          .YK(200)         .Z(40)          .ZK(200)
      6 .ZLAM(20)
      COMMON /IARRAY/   IFLAG(40)       IPCINT(50)
      .ISMAX(50)        .ISCR(20)       .JF(200)        .JFL(200)
      .JL(200)          .JM(200)        .JN(200)        .JP(200)
      .JRAY(200)        .JSOR(50)       .KEND(900)      .KFLAG(220)
      .KP(200)          .KTYP(200)      .LAMP(50)       .LANGLE(26)
      .LREFL(200)       .LTYPE(20)      .MAT(200)       .MFL(20)
      .MWAT(20)        .NMURAY(200)
      COMMON /PARAMS/   AL              ALN            ALR            ALRF           RE
      .BER             .BER            .DIS           .DMIN          .DZ            .EPS
      .GA              .GAN            .GARF          .PER           .
      .SUMRA           .XI             .XNOW          .XCLD          .XREF          .XTRAN        .VI
      .YNOW            .YCLD           .YREF          .YTRAN        .ZI            .
      .ZNOW            .ZCLD           .ZREF          .ZTRAN
      COMMON /IPARAM/   FMAX            IPAX           INOW           .
      .ICORAY          .ICORINT        .IR            .IRAY          .JMAX          .JNOW          .JOLD
      .KITY            .KMAX           .KNOW          .KMAX          .KNOW          .KSOR
      INTEGER F, FMAX
      ..... GET INDEX FOR BOUNDARY ARRAY FOR SEGMENT J.
      N=KP(J)
      ..... GET INDEX K FOR ONE BOUNDARY OF SEGMENT J.
      104 K=IARS(KEND(N))
      ..... GET GEOMETRY TYPE.
      KT=KTYP(K)
      KSIGN=KPAD(N)
      ..... BRANCH ON GEOMETRY TYPE.
      GO TO(100,200,300,400,500,600,700,800,900),KT
      ..... IF POINT ON DESIRED SIDE OF PLANE, GO TO 101
      100 IF((IPLAN(K)*KSIGN)/102,102,101
      ..... IF POINT ON DESIRED SIDE OF CYLINDER, GO TO 101
      200 IF((INCYL(K)*KSIGN)/102,102,101
      ..... IF POINT ON DESIRED SIDE OF CONIC, GO TO 101
      300 IF((INCON(K)*KSIGN)/102,102,101
      ..... IF POINT ON DESIRED SIDE OF SPHERE, GO TO 101.
      400 IF((INSPH(K)*KSIGN)/102,102,101
      ..... IF POINT ON DESIRED SIDE OF CONE , GO TO 101.
      500 IF((INCA(K)*KSIGN)/102,102,101
      ..... IF POINT ON DESIRED SIDE OF PARABOLOID, GO TO 101.
      600 IF((INPAR(K)*KSIGN)/102,102,101
      ..... IF POINT ON DESIRED SIDE OF HYPERBOLOID, GO TO 101.
      700 IF((INHYP(K)*KSIGN)/102,102,101
      ..... IF POINT ON DESIRED SIDE OF ELLIPSOID, GO TO 101.
      800 IF((INELL(K)*KSIGN)/102,102,101
      ..... IF POINT ON DESIRED SIDE OF HELIX, GO TO 101.
      900 IF((INHEL(K)*KSIGN)/102,102,101
      ..... LOOK AT NEXT BOUNDARY.
      101 N=N+1
      IF(N=0)104,103,104
      ..... POINT IS INSIDE SEGMENT.
      103 INSEG=1
      GO TO 110
      ..... POINT IS OUTSIDE SEGMENT.
      102 INSEG=0
      IF CONTINUE
      RETURN
      END

```

```

      FUNCTION INSPH(K)
      C.....
      C..... FUNCTION INSPH FINDS IF POINT IS INSIDE SPHERE.
      C.....
      COMMON /69/      ABCD(20,200)      ALRAY(500)      BEHAY(500)
1  COORDS(6,50)      E(40,200)      EBASE(200)      ESCR(20,200)
2  GARAY(500)      RED(20,200)      TITLE(12)      VPMAT(20)
3  VOL(200)      XLAM(200)      XRAY(500)      YRAY(500)
4  ZRAY(500)
      COMMON /ARRAYS/  ALK(200)      ALPHA(40)      BEK(200)
1  RETA(40)      DEV(200)      DK(200)      EK(200)
2  ELAM(200)      ELAMAT(20)      EMAT(20)      ENERGY(200)
3  GAK(200)      GAMMA(40)      REF(20)      SUMMAT(20)
4  SUMRAY(200)      SUMSCR(200)      X(40)      XK(200)
5  Y(40)      YK(200)      Z(40)      ZK(200)
6  ZLAM(20)
      COMMON /ARRAY/  IFLAG(40)      IPCINT(50)
      ISMAX(50)      ISCR(20)      JF(200)      JFL(200)
      JL(200)      JM(200)      JN(200)      JPL(200)
      JRAY(200)      JSCR(50)      KEND(900)      KFLAG(200)
      KP(200)      KTOP(200)      LAMP(50)      LAAGLE(20)
      LREFL(200)      LTYPE(20)      MAT(200)      MFL(20)
      NMAT(20)      NUMRAY(200)
      COMMON /PARAMS/  AL      ALN      ALR      ALRF      RE
      BEN      RER      BERF      DIS      DMIN      DZ      EPS
      GA      GAN      GAR      GARF      PER
      SUMRA      XI      XND      XCLD      XREF      XTRAM      VI
      YND      YCLD      YREF      YTRAN      ZI
      ZND      ZCLD      ZREF      ZTRAN
      COMMON /IFRAME/  FMAX      IMAX      INOW
      ICPRAY      ICPRAT      IR      IY      JMAX      JNOW      JOLD
      KIAT      KMAX      KND      MAX      MIN      NSOR
      INTEGER F,FMAX
      C..... CALCULATE DISTANCE TO CENTER OF SPHERE.
      SS=(XJ-XK(K))**2+(YJ-YK(K))**2+(ZJ-ZK(K))**2
      C..... IF DISTANCE GREATER THAN RADIUS, POINT IS OUTSIDE.
      IF(SS-CK(K)**2)101,101,102
      C..... POINT IS INSIDE.
      101 INSPH=1
      RETURN
      C..... POINT IS OUTSIDE.
      102 INSPH=-1
      RETURN
      END

```

```

SUBROUTINE INTCH(K,DMIT)
C*****
C***** INTCH FINDS INTERSECTION OF RAY WITH A CONE. SEE WRITEUP
C***** FOR EQUATIONS USED.
C*****
COMMON /69/ ABCO(20,200) ,ALRAY(500) ,BERAY(500)
1 ,CNOPTS(6,50) ,E(40,200) ,EHASE(200) ,ESOR(20,200)
2 ,GARAY(500) ,RED(20,200) ,TITLE(12) ,VMAT(20)
3 ,VOL(200) ,XLAN(200) ,XRAY(500) ,YMAT(500)
4 ,ZRAY(500)
COMMON /ARRAYS/ ALK(200) ,ALPHA(40) ,BEK(200)
1 ,RETA(40) ,DEV(200) ,DK(200) ,EK(200)
2 ,ELAM(200) ,ELAMAT(20) ,EMAT(20) ,ENERGY(200)
3 ,GAK(200) ,GAMMA(40) ,REF(20) ,SCRMAT(20)
4 ,SUMRAY(200) ,SUMSOR(200) ,X(40) ,XK(200)
5 ,Y(40) ,YK(200) ,Z(40) ,ZK(200)
6 ,ZLAN(20)
COMMON /IARRAY/ IFLAE(40) ,IPOINT(50)
1 ,ISX(50) ,JSOR(20) ,JF(200) ,JFL(200)
2 ,JL(200) ,JM(200) ,JH(200) ,JP(200)
3 ,JRAY(200) ,JSOR(50) ,KEND(900) ,KFLAG(200)
4 ,KP(200) ,KTP(200) ,LAMP(50) ,LANGE(20)
5 ,LRFL(200) ,LTYPE(20) ,MAT(200) ,MFL(20)
6 ,MMAT(20) ,MMURAY(200)
COMMON /PARAMS/ AL ,ALN ,ALR ,ALRF ,RE
1 ,BER ,BERF ,DIS ,DMIN ,DZ ,EPS
2 ,GA ,GAN ,GAR ,GANF ,PER
3 ,SUMRA ,XI ,XNOW ,XCLO ,XREF ,XTRAN ,YI
4 ,YNOW ,YOLD ,YREF ,YTRAN ,ZI
5 ,ZNOW ,ZOLD ,ZREF ,ZTRAN
COMMON /PARAM/ FMAX ,IMAX ,INOW
1 ,IOPRAY ,IOPRAT ,IR ,IRAY ,JMAX ,JACF ,JOLD
2 ,KINT ,KPAK ,KNOW ,MMAX ,NNOW ,NSON
INTEGER F,FMAX
C***** CALCULATE PARAMETERS NEEDED FOR DISTANCE EQUATION.
DELX=XI-XK(K)
DELY=YI-YK(K)
DELZ=ZI-ZK(K)
C=DELZ**2-DK(K)*(DELX**2+DELY**2+DELZ**2)
B=(1.0-DK(K))*GA*DELZ-DK(K)*(AL*DELX+BE*DELY)
A=GA**2-DK(K)
SQ=R**2-A**2
IF(SQ)101,102,102
102 ROOT=SQRT(SQ)
C***** CALCULATE BOTH ROOTS AND CHOOSE THE SMALLEST POSITIVE
C***** ROOT THAT IS IN THE DESIRED HALF OF CONE
DMIT1=(-B+ROOT)/A
DMIT2=(-B-ROOT)/A
IF(DMIT1)101,103,103
103 EPL=(ZI+DMIT1*GA-ZK(K))*EK(K)
EMI=(ZI+DMIT2*GA-ZK(K))*EK(K)
IF(DMIT2)105,105,104
104 IF(EMI)105,105,106
105 IF(EPL)101,101,107
106 DMIT=DMIT2
GO TO 108
107 DMIT=DMIT1
GO TO 108
101 DMIT=1.E+21
108 CALL DEBUG(6,DMITCN)
RETURN
END

```



```

SUBROUTINE INTCON(K,DNIT)
C*****
C***** SURROUTINE INTCON CALCULATES THE DISTANCE FROM POINT
C***** XI,YI,ZI TO THE POINT WHERE THE RAY INTERSECTS CONIC
C***** BOUNDARY K. SEE WRITEUP FOR EXACT EQUATIONS USED.
C*****
COMMON /69/ ABCO(20,200) ,ALRAY(500) ,BERAT(500)
1 ,CWORDS(6,50) ,E(40,200) ,EMASE(200) ,ESOM(20,200)
2 ,GARAY(500) ,RED(20,200) ,TITLE(12) ,VMAT(20)
3 ,VOL(200) ,XLAN(200) ,XRAY(500) ,YRAY(500)
4 ,ZRAY(500)
COMMON /ARRAYS/ ALK(200) ,ALPHA(40) ,BEK(200)
1 ,BETA(40) ,DEV(200) ,DK(200) ,EK(200)
2 ,FLAN(200) ,ELAMAT(20) ,EMAT(20) ,ENERGY(200)
3 ,EAK(200) ,GAMMA(40) ,REF(20) ,SONMAT(20)
4 ,SUMRAY(200) ,SUMSOR(200) ,X(40) ,XK(200)
5 ,Y(40) ,YK(200) ,Z(40) ,ZK(200)
6 ,ZLAN(20)
COMMON /ARRAY/ IFLAG(40) ,IPOINT(50) ,
,ISOR(20) ,JF(200) ,JFL(200) ,
,JL(200) ,JM(200) ,JN(200) ,JP(200) ,
,JRAY(200) ,JSOR(50) ,KBD(900) ,KFLAG(200) ,
,KP(200) ,KTP(200) ,LAMP(50) ,LANGLE(20) ,
,LREFL(200) ,LTYPE(20) ,MAT(200) ,MFL(20) ,
,MNAT(20) ,MNRAY(200)
COMMON /PARAMS/ AL ,ALH ,ALR ,ALRF ,HE ,
,BE ,BER ,BERF ,DIS ,DMIN ,DZ ,EPS ,
,GA ,GAN ,GAR ,GARF ,PER ,
,SUMRA ,XI ,XNO ,XOLD ,XREF ,XTRAN ,YI ,
,YNO ,YOLD ,YREF ,YTRAN ,ZI ,
,ZNO ,ZOLD ,ZREF ,ZTRAN
COMMON /IPARAM/ FMAX ,IMAX ,IMON ,
,IOPRAY ,ICORNT ,IR ,IRAY ,JMAX ,JMON ,JOLD ,
,KIAT ,KMAX ,KAO ,MMAX ,MNO ,MSOR
INTEGER F,FMAX
C***** CALCULATE COEFFICIENTS FOR DISTANCE EQUATION.
DX=XI-XK(K)
DY=YI-YK(K)
IF(ABS((5A)**2-1.0)-1.E-6)104,104,101
101 COSTH=SQRT(1.0-6A**2)
ALL=AL/COSTH
BEL=BE/COSTH
CALL CHECK(MINTCON,ALL,BEL,0.0)
EE=DX*ALL+DY*BEL
FF=DX*ALK(K)+DY*BEK(K)
GG=ALK(K)*ALL+BEK(K)*BEL
HS=DX**2+DY**2
PP=1.0-(EK(K)**2)*(GG**2)
IF(ABS(PP)-1.E-20)104,104,102
102 MM=FE-EK(K)**2*GG*(FF+DK(K))
OO=HS-EK(K)**2*(FF+DK(K))**2
MSPO=MM**2-PP*OO
IF(MSPO)104,103,103
C***** FIND BOTH ROOTS OF THE EQUATION AND CHOOSE THE SHORTEST
C***** POSITIVE ROOT.
103 ROOT=SQRT(MSPO)
DNIT1=(-MM+ROOT)/(PP+COSTH)
DNIT2=(-MM-ROOT)/(PP+COSTH)
DNIT=AMIN(DNIT1,DNIT2)
IF(DNIT)105,105,107
105 DNIT=AMAXI(DNIT1,DNIT2)
IF(DNIT)104,104,107
104 DNIT=1.E21
107 CONTINUE
CALL DEBUG(MINTCON)
RETURN
END

```

```

SUBROUTINE INTCYL(K,DHIT)
C*****
C***** SUBROUTINE INTCYL CALCULATES THE DISTANCE FROM POINT
C***** XI,YI,ZI IN THE DIRECTION AL,BE,GA TO THE POINT WHERE THE
C***** RAY INTERSECTS CYLINDRICAL BOUNDARY K. SEE WRITEUP FOR
C***** EXACT EQUATIONS USED.
C*****
COMMON /69/ ABCO(20,200) ,ALRAY(500) ,BERAY(500)
1 ,COORDS(6,50) ,E(40,200) ,EGASE(200) ,ESOH(20,200)
2 ,GARAY(500) ,RED(20,200) ,TITLE(12) ,VMAT(20)
3 ,VOL(200) ,ZLAM(200) ,XRAY(500) ,YRAY(500)
4 ,ZRAY(500)
COMMON /ARRAYS/ ALK(200) ,ALPHA(40) ,BEK(200)
1 ,RFTA(40) ,DEV(200) ,DK(200) ,EK(200)
2 ,ELAM(200) ,ELAMAT(20) ,EMAT(20) ,ENERGY(200)
3 ,GAK(200) ,GAMMA(40) ,REF(20) ,SCMAT(20)
4 ,SUMRAY(200) ,SUMSOR(200) ,X(40) ,XK(200)
5 ,Y(40) ,YK(200) ,Z(40) ,ZK(200)
6 ,ZLAM(20)
COMMON /IARRAY/ IFLAG(40) ,IPOINT(50)
1 ,ISMAX(50) ,JSOR(20) ,JF(200) ,JFL(200)
2 ,JL(200) ,JM(200) ,JH(200) ,JP(200)
3 ,JRAY(200) ,JSOR(50) ,KEND(900) ,KFLAG(200)
4 ,KP(200) ,KTP(200) ,LAMP(50) ,LAAGLE(20)
5 ,LREFL(200) ,LTYPE(20) ,MAT(200) ,PFL(20)
6 ,MAT(20) ,MUMRAY(200)
COMMON /PARAMS/ AL ,ALN ,ALR ,ALRF ,RE
1 ,BER ,BERF ,DIS ,DWIN ,DZ ,EPS
2 ,GA ,GAN ,GAR ,GARF ,PER
3 ,SUNRA ,XI ,XAN ,XCLO ,XREF ,XTRAN ,YI
4 ,YNO ,YOLD ,YREF ,YTRAN ,ZI
5 ,ZAC ,ZOLD ,ZREF ,ZTRAN
COMMON /IPARAM/ FMAX ,IMAX ,INOW
1 ,IOPRAY ,IOPRNT ,IR ,IRAY ,JMAX ,JNOW
2 ,KINT ,KPAX ,KNOV ,MMAX ,MNOV ,NSOR ,JOLD
INTEGER F,FMAX
C***** CALCULATE COEFFICIENTS FOR DISTANCE EQUATION.
XX=AL*ALK(K)+BE*BEK(K)+GA*GAK(K)
OMIS=1.0-XX**2
IF (ABS(OMIS)-1.E-6) 101,101,102
102 DELX=XI-XK(K)
DELY=YI-YK(K)
DELZ=ZI-ZK(K)
YY=AL*DELX+BE*DELY+GA*DELZ
ZZ=ALK(K)*DELX+BEK(K)*DELY+GAK(K)*DELZ
ASQ=DELX**2+DELY**2+DELZ**2
SOTERM=(YY-ZZ*XX)**2-(OMIS)*(ASQ-ZZ**2-DK(K)**2)
IF (SOTERM) 101,103,103
C***** FIND BOTH ROOTS OF THE EQUATION AND CHOOSE THE SHORTEST
C***** POSITIVE ROOT.
103 ROOT=SQRT(SOTERM)
ZKMY=ZZ+XX-YY
DHIT1=(ZKMY+ROOT)/OMIS
DHIT2=(ZKMY-ROOT)/OMIS
DHIT=AMIN1(DHIT1,DHIT2)
IF (DHIT) 104,104,105
104 DHIT=AMAX1(DHIT1,DHIT2)
IF (DHIT) 101,101,105
101 DHIT=1.E+21
105 CONTINUE
CALL DEBUG(6HINTCYL)

RETURN
END

```

```

SUBROUTINE INTDAT(IT,LT)
C*****
C***** SUBROUTINE INTDAT CALLS THE DESIRED ROUTINE TO
C***** CALCULATE WAVELENGTH DEPENDENT XRAY INTENSITIES.
C*****
COMMON /A9/ ABCD(20,200) ,ALRAY(500) ,PLRAY(500)
1 ,CDRPS(6,50) ,E(40,200) ,EBASE(200) ,ESON(20,200)
2 ,GARAY(500) ,RED(20,200) ,TITLE(12) ,VMAT(20)
3 ,VVL(200) ,XLAM(200) ,XRAY(500) ,YRAY(500)
4 ,ZRAY(500)
COMMON /ARRAYS/ ALK(200) ,ALPHA(40) ,BEK(200)
1 ,BETA(40) ,DEV(200) ,DK(200) ,EK(200)
2 ,ELAM(200) ,ELAMAT(20) ,EMAT(20) ,ENERGY(200)
3 ,GAK(200) ,GAMMA(40) ,REF(20) ,SORMAT(20)
4 ,SUMRAY(200) ,SUMSOR(200) ,X(40) ,XK(200)
5 ,Y(40) ,YK(200) ,Z(40) ,ZK(200)
6 ,ZLAM(20)
COMMON /IARRAY/ IFLAG(40) ,IPOINT(50)
1 ,ISAT(50) ,ISOR(20) ,JF(200) ,JFL(200)
2 ,JL(200) ,JM(200) ,JM(200) ,JP(200)
3 ,JRAY(200) ,JSOR(50) ,KBD(900) ,KFLAG(200)
4 ,KP(200) ,KTP(200) ,LAMP(50) ,LANGE(20)
5 ,LREFL(200) ,LTYPE(20) ,MAT(200) ,MFL(20)
6 ,MRAY(20) ,MURAY(200)
COMMON /PARAMS/ AL ,ALN ,ALR ,ALRF ,RE
1 ,BEN ,BER ,BERF ,DIS ,DMIN ,DZ ,EPS
2 ,GA ,GAR ,GARF ,GOLD ,GREF ,XTRAM ,YI
3 ,SUMRA ,XI ,XRC ,XOLD ,XREF ,ZI
4 ,ZMOD ,YOLD ,YREF ,YTRAN ,ZTRAN
COMMON /IPARAM/ FMAX ,IMAX ,INOW ,JNOW ,JOLD
1 ,ICRAY ,ICPRNT ,IF ,IRAY ,JMAX ,JNSOR
2 ,KI ,KPAI ,KNOW ,MMAX ,MNOW
INTEGER F,FMAX
DIMENSION TSTG(200)
C***** GO TO DESIRED CALL.
GO TO(100,200,300,400,500,600,700,800,900),IT
100 CALL INTEN1(XLAM,FMAX,TSTG,LT)
GO TO 110
200 CALL INTEN2(XLAM,FMAX,TSTG,LT)
GO TO 110
300 CALL INTEN3(XLAM,FMAX,TSTG,LT)
GO TO 110
400 CALL INTEN4(XLAM,FMAX,TSTG,LT)
GO TO 110
500 CALL INTEN5(XLAM,FMAX,TSTG,LT)
GO TO 110
600 CALL INTEN6(XLAM,FMAX,TSTG,LT)
GO TO 110
700 CALL INTEN7(XLAM,FMAX,TSTG,LT)
GO TO 110
800 CALL INTEN7(XLAM,FMAX,TSTG,LT)
GO TO 110
900 CALL INTEN9(XLAM,FMAX,TSTG,LT)
DO 100 F=1,FMAX
ESOR(LT,F)=TSTG(F)
100 CONTINUE
RETURN
END

```

SUBROUTINE INTELL(K,DMIT)

C.....
C..... INTELL FINDS THE INTERSECTION DISTANCE OF A RAY WITH
C..... AN ELLIPSOID. SEE *RITUP FOR EQUATIONS USED.
C.....

COMMON /69/ ABCO(20,200) *ALRAY(500) *BENAY(500)
1 *COORDS(6,50) *E(40,200) *ENASE(200) *ESOH(20,200)
2 *GARAY(500) *RED(20,200) *TITLE(12) *VMAT(20)
3 *VOL(200) *XLAN(200) *XRAT(500) *YHAY(500)
4 *ZRAY(500)
COMMON /ARRAYS/ ALK(200) *ALPHA(40) *BEK(200)
1 *RFTA(40) *DEV(200) *DK(200) *EK(200)
2 *ELAM(200) *ELAMAT(20) *EMAT(20) *ENENGY(200)
3 *GAK(200) *GAMMA(40) *REF(20) *SUMMAT(20)
4 *SUMRAY(200) *SUMSOR(200) *X(40) *XK(200)
5 *Y(40) *YK(200) *Z(40) *ZK(200)
6 *ZLAN(20)
COMMON /IARRAY/ IFLAG(40) *IPOINT(50) *
*ISMAX(50) *ISOR(20) *JF(200) *JFL(200) *
*JL(200) *JM(200) *JN(200) *JP(200) *
*JRAY(200) *JSOR(50) *KBND(900) *KFLAG(200) *
*KP(200) *KTYP(200) *LAMP(50) *LAAGLE(20) *
*LREFL(200) *LTYPE(20) *MAT(200) *MFL(20) *
*MMAT(20) *MMRAY(200)
COMMON /PARAMS/ AL *ALN *ALR *ALRF *BE *
*BEH *BEH *BERF *DIS *DMIN *DZ *EPS *
*GA *GAN *GAR *GANF *PEN * *
*SUMRA *XI *XNOW *XOLD *XNEF *XTRAN *YI *
*YNEW *YOLD *YREF *YTHAN *ZI *
*ZNEW *ZOLD *ZREF *ZTRAN *
COMMON /IPARAM/ FMAX *IMAX *INOW *
*IOPRAY *IOPRMT *IH *IRAT *JMAX *JNCW *JOLD *
*KIAT *KPAX *KNOW *MMAX *MNOW *MSOR

INTEGER F,FMAX

C..... CALCULATE COEFFICIENTS NEEDED IN EQUATIONS.

DELX=XI-XK(K)
DELY=YI-YK(K)
DELZB=ZI-ZK(K)-EK(K)
BSC=EK(K)**2
C=1.0-(DELX**2+DELY**2)/DK(K)-DELZB**2/BSC
B=-(AL*DELX+BE*DELY)/DK(K)+GA*DELZB/BSC
A=-(AL**2+BE**2)/DK(K)+GA**2/BSC
SO=9**2-A*C
IF(SO)101,102,102
102 ROOT=SCRT(SO)

C..... FIND BOTH ROOTS AND CHOOSE SMALLEST POSITIVE ROOT.

DMIT1=(-B+ROOT)/A
DMIT2=(-B-ROOT)/A
DMIT=AMIN1(DMIT1,DMIT2)
IF(DMIT)104,104,105
104 DMIT=AMAX1(DMIT1,DMIT2)
IF(DMIT)101,101,105
101 DMIT=I.E*2I
105 CONTINUE
CALL DEBUG(6HINTELL)
RETURN
END

SUBROUTINE INTHEL(K,M1)

 ***** INTHEL LOOKS FOR THE NEAREST INTERSECTION OF A RAY WITH
 ***** MELICAL SURFACE. INTHEL PRIMARILY DETERMINES WHICH (IF ANY)
 ***** COILS ARE POSSIBLE FOR INTERSECTIONS AND CALLS DCALC FOR
 ***** EACH POSSIBLE COIL.

```

COMMON /69/ ARCC(20,200) ,ALRAY(500) ,DELRAY(500)
1 ,COORDS(6,50) ,E(40,200) ,EBASE(200) ,ESOR(20,200)
2 ,GARAY(500) ,RED(20,200) ,TITLE(12) ,VMAT(20)
3 ,VOL(200) ,XLAN(200) ,XRAY(500) ,YRAY(500)
4 ,ZRAY(500)
COMMON /ARRAYS/ ALK(200) ,ALPHA(40) ,BEK(200)
1 ,BETA(40) ,DEV(200) ,DK(200) ,EK(200)
2 ,ELAN(200) ,ELAMAT(20) ,EMAT(20) ,ENHNGY(200)
3 ,GAK(200) ,GAMMA(40) ,REF(20) ,SCNMAT(20)
4 ,SINRAY(200) ,SUMSOR(200) ,X(40) ,XK(200)
5 ,Y(40) ,YK(200) ,Z(40) ,ZK(200)
6 ,ZLAN(200)
COMMON /IARRAY/ IFLAG(40) ,IPCINT(50)
1 ,ISPL(50) ,ISOR(20) ,JF(200) ,JFL(200)
2 ,JL(200) ,JM(200) ,JK(200) ,JP(200)
3 ,JRAY(200) ,JSOR(50) ,KMC(1000) ,KFLAG(200)
4 ,KP(200) ,KTYP(200) ,LAMP(50) ,LANGLE(20)
5 ,LREFL(200) ,LTYPE(20) ,MAT(200) ,MFL(20)
6 ,MMRAY(200)
COMMON /PARAMS/ AL ,ALN ,ALR ,ALRF ,RE
1 ,BEN ,BET ,BEF ,DIS ,DMIN ,DZ ,EPS
2 ,GA ,GAN ,GAR ,GARF ,PER
3 ,SUNRA ,XI ,XAN ,XOLD ,XREF ,XTRAN ,YI
4 ,YAN ,YOLD ,YREF ,YTRAN ,ZI
COMMON /IPARAM/ FMAX ,IPAX ,INOW
1 ,IOPRAY ,ICPRAY ,IP ,IRAY ,JMAX ,JNOW ,JOLD
2 ,KINT ,KMAX ,KNC ,KMAX ,KNOW ,KSOR
INTEGER F,FMAX
COMMON /MELCC/ A,ABAR,S,BI,BIP,BP,CAP,DELTH,DELX,DELY,DELZ,DMIT,
PS17,REV,R-J,SIN,TH,TN1,SCNRS,RKS,CSTN,SA1N,RCSTN,RSATN,BSTN,
COSTAN,THZ,DTN2
DMIT=1.E-21

```

***** CALCULATE PARAMETERS NEEDED.

```

CAP=ALK(K)
BNC=BEK(K)
APK(K)
ABAR=1.0+2.0
CALL PERTG(IAL,RE,SA)
SI=0.0
DELX=XK(K)-XI
DELY=YK(K)-YI
DELZ=ZK(K)-ZI
TMP1=AL*DELY-RE*DELX
TMP2=DELX**2+DELY**2+DELZ**2
FMI=SIGN(1.0,TMP1)*SQRT(IAL**2+BE**2)
SA=(DELX*AL-DELY*RE)/(IAL**2+BE**2)
BSC=(DELX-S*AL)**2+(DELY-S*BE)**2
RCUTS=(F-C-A)**2
RIASQ=(B-C-A)**2

```

***** IF POINT ON RAY NEAREST TO MELICAL AXIS NOT INSIDE OUTER

***** RADIUS OF MELIA, RETURN.

```

IF(RIASQ-RCUTS)10,110,110

```

```

101 RMOSQ=DELX**2+DELY**2

```

```

      UOUT=SQRT(ROUTSQ-B50)/ABS(FMU)
      C***** IF STARTING POINT NOT INSIDE OUTER RADIUS, GO TO 102.
      IF(PHOSQ-RC-TSD)104,104,102
      102 DOT=(AL*DELX+FE*DELY)
      C***** IF RAY MOVING AWAY FROM HELIX, RETURN.
      IF(DOT)110,110,103
      C***** CALCULATE COIL NUMBERS AND OTHER NEEDED PARAMETERS.
      103 SI=SA-LOUT
      104 S4=SA-LOUT
      REV=4.78318531*SIGN(1.0,SA*CAP)
      XR=XI+AL*SA
      YR=YI+FE*SB
      ZR=ZI+EA*SB
      IF(ABS(YR(K)-YR)-1.E-5)140,140,141
      140 IF(ABS(XR(K)-XR)-1.E-5)142,142,141
      142 PSIZ=ATAN2(-DELX,DELY)
      GO TO 143
      141 PSIZ=ATAN2(YR(K)-YR,XR(K)-XR)
      143 N1=INT(S1)
      IF(N1.GT.100) GO TO 110
      N4=INT(S4)
      SOT=RJASQ-B50
      C***** IF POINT ON RAY NEAREST TO AXIS IS INSIDE INNER RADIUS, GO TO
      IF(SOT)106,106,105
      106 N2=10**5
      N3=10**8
      GO TO 107
      105 U1L=SQRT(SQ1)/ABS(FMU)
      S2=SA-L1N
      S3=SA-L1N
      N2=INT(S2)
      N3=INT(S3)
      IF(S2.LT. 0 .AND. S3.GT. 100) GO TO 110
      107 BCL=SQRT(B50)
      C***** TEST TO SEE IF N5 AND N6 ARE NEEDED.
      IF(FMU*CAP/GA-PCL)113,113,100
      100 XLA=SQRT((CAP*BCL/GA/FMU-B50/FIL**2)
      S5=SA-XLA
      S6=SA-XLA
      N5=INT(S5)
      N6=INT(S6)
      IF(N5-A6)109,113,113
      109 IF(XLA**2-B50-ROUYSQ)114,114,111
      111 CONTINUE
      CALL DCALC(N1)
      IF(N1-A5)112,110,112
      112 CALL DCALC(N5)
      GO TO 110
      113 N5=10**8
      N6=10**8
      C***** LOOP THROUGH POSSIBLE COILS UNTIL AN INTERSECTION IS FOUND
      C***** OR UNTIL ALL POSSIBLE COILS HAVE BEEN TRIED.
      114 N=n1
      123 IF(N-A7)117,117,115
      115 IF(N-A3)116,117,117
      116 N=n3
      117 IF(N-A6)119,118,119
      118 N=n5
      119 CONTINUE
      IF(N.GT.200)GO TO 110
      C***** LOOK FOR INTERSECTION WITH COIL N.
      CALL DCALC(N)
      IF(N-N5)121,120,121
      120 N=n6
      GO TO 119
      121 IF(DN17-1.E-20)110,122,122
      122 N=n1
      IF(N-A4)123,123,110
      110 CALL DEBUG(6,N,INTHEL)
      DN1=DN17
      RETURN
      END

```

```

SUBROUTINE INTNYP(K,DHIT)
C.....
C..... INTNYP FINDS THE INTERSECTION DISTANCE OF A RAY WITH
C..... A HYPERFOLOID. SEE DHITUP FOR EQUATIONS USED.
C.....
COMMON /69/ ABCO(20,200) *ALRAY(500) *BERAY(500)
1 *COORDS(6,50) *E(46,200) *ERASE(200) *ESOR(20,200)
2 *GARAY(500) *RED(20,200) *TITLE(12) *VNAT(20)
3 *VOL(200) *XLAN(200) *XRAY(500) *YNAT(500)
4 *ZRAY(500)
COMMON /ARRAYS/ ALK(200) *ALPHA(40) *BEK(200)
1 *RETA(40) *DEV(200) *DK(200) *EK(200)
2 *FLAM(200) *ELAMAT(20) *EMAT(20) *ENERGY(200)
3 *GAK(200) *GAMMA(40) *REF(20) *SQMAT(20)
4 *SUMRAY(200) *SUMSQH(200) *X(40) *XK(200)
5 *Y(40) *YK(200) *Z(40) *ZK(200)
6 *ZLAN(20)
COMMON /ARRAY/ IFLAG(40) *IPOINT(50)
*ISMAX(50) *ISOR(20) *JF(200) *JFL(200)
*JL(200) *JM(200) *JP(200)
*JRAY(200) *JSOR(50) *KEND(900) *KFLAG(200)
*KP(200) *KTP(200) *LAMP(50) *LANGLE(20)
*LPFFL(200) *LTYPE(20) *MAT(200) *MFL(20)
*MMAT(20) *MMRAY(200)
COMMON /PARAMS/ AL *ALN *ALR *ALRF *RE
*BEH *BERF *DIS *DWIN *DZ *EPS
*GA *GAR *GARF *PER
*SUMRA *XI *XANOW *XOLD *XREF *XTRAN *YI
*YND *YOLD *YREF *YTRAN *ZI
*ZND *ZOLD *ZREF *ZTRAN
COMMON /IPARAM/ FMAX *IMAX *INOW
*IOPRAY *IOPRNT *IR *IRAY *JMAX *JNOW
*KIAT *KMAX *KNOW *PMAX *PNOW *NSOR *JOLD
INTEGER F,FMAX
C..... CALCULATE COEFFICIENTS NEEDED FOR EQUATIONS.
DELX=XI-XK(K)
DELY=YI-YK(K)
DELZH=ZI-ZK(K)*EK(K)
A=(GA/FK(K))**2-(AL**2+BE**2)/DK(K)
B=GA*DELZH/(EK(K)**2)-(AL*DELX+BE*DELY)/DK(K)
C=-1.0-(IFELX**2+DELY**2)/DK(K)+(DELZH**2)/(EK(K)**2)
SQH=**2-A*C
IF(SQ)101,102,102
102 RMT=SQRT(SQ)
C..... CALCULATE BOTH ROOTS AND CHOOSE THE SMALLEST POSITIVE
C..... VALUE THAT LIES IN THE DESIRED SHEET.
DHIT1=(-R+RMT)/A
DHIT2=(-R-RMT)/A
EPL=(ZI-DHIT1*GA-ZK(K))*GAK(K)
EM=(ZI-DHIT2*GA-ZK(K))*GAK(K)
IF(EPL)104,104,103
104 IF(EM)101,101,105
105 DHIT=DHIT2
GO TO 106
103 IF(FY)107,107,108
107 DHIT=DHIT1
GO TO 106
108 DHIT=AMIN1(DHIT1,DHIT2)
IF(DHIT)109,109,110
109 DHIT=AMAX1(DHIT1,DHIT2)
106 IF(DHIT)101,101,110
101 DHIT=1.E+21
110 CALL DEBUG(6,INTNYP)
RETURN
END

```

```

SUBROUTINE INTPAR(K,DHIT)
C.....
C..... INTPAR FINDS THE DISTANCE TO THE NEAREST INTERSECTION OF
C..... A RAY WITH A PARABOLA. SEE WRITEUP FOR EQUATIONS USED.
C.....
COMMON /69/
1  COORDS(6,50)  E(40,200)  ALRAY(500)  BERAY(500)
2  GARAY(500)    RED(20,200)  EHASE(200)  ESOR(20,200)
3  VOL(200)      XLAM(200)    XRAY(500)   VPAT(20)
4  TRAY(500)
COMMON /ARRAYS/
1  RETA(40)      ALK(200)      ALPHA(40)    BEK(200)
2  FLAP(200)     ELAMAT(20)    DK(200)     EK(200)
3  FAK(200)      GAMMA(40)     EMAT(20)    ENERGY(200)
4  SUMRAY(200)   SUMSON(200)    REF(20)     SCHMAT(20)
5  Y(40)         YK(200)       X(40)       XK(200)
6  ZLAM(20)      ZK(200)       Z(40)       ZK(200)
COMMON /IARRAY/
1  ISMAX(50)     ISOR(20)      IFLAG(40)   IPOINT(50)
2  JL(200)       JM(200)       JF(200)     JFL(200)
3  JRAY(200)     JSOR(50)      JN(200)     JP(200)
4  JP(200)       KTOP(200)     KEND(900)   KFLAG(200)
5  LREFL(200)    LTYPE(20)    LAMP(50)    LANGLE(20)
6  MAT(20)       NUMRAY(200)    MAT(200)    MFL(20)
COMMON /PARAMS/
1  BEH           BEF           ALN           ALR           ALRF          E
2  GA           GAR           DIS           DMIN          DZ           EPS
3  SUMRA        XI           XND          GANF          PER           X
4  YND          YOLD         YREF         YTHAN         XREF          XTRAN        YI
5  ZND          ZOLD         ZREF         ZTHAN         ZI           X
COMMON /IPARAM/
1  IOPRAY        IOPRNT       FMAX         INAX          INOW          J
2  KINT          KPAX         IR           IRAY          JMAX          JNOW          JOLD
3  KNOW          KNOW         INMAX        INOW          NSOR
INTEGER F,FMAX
C..... CALCULATE COEFFICIENTS NEEDED FOR EQUATIONS.
DELX=XI-XK(K)
DELY=YI-YK(K)
DELZ=ZI-ZK(K)
C=4.0*CK(K)*DELZ-DELX**2-DELY**2
B=2.0*DK(K)*GA-AL*DELX-BE*DELY
A=(AL**2+BE**2)
SQ=R**2-A*C
IF(SQ)101,102,102
102 ROOT=SQRT(SQ)
C..... FIND BOTH ROOTS AND CHOOSE SMALLEST POSITIVE VALUE.
DHIT1=(-B-ROOT)/A
DHIT2=(-B+ROOT)/A
DHIT=AMIN1(DHIT1,DHIT2)
IF(DHIT)104,104,105
104 DHIT=AMAX1(DHIT1,DHIT2)
IF(DHIT)101,101,105
101 DHIT=1.E+21
105 CONTINUE
CALL DEBUG(6,INTPAR)
RETURN
END

```



```

      SUBROUTINE INTPLA(K,DMIT)
C*****
C***** SUBROUTINE INTPLA CALCULATE THE DISTANCE FROM THE POINT
C***** XI,YI,ZI IN THE DIRECTION AL,BE,GA TO THE POINT WHERE THE
C***** RAY INTERSECTS THE PLANE BOUNDARY K.
C*****
      COMMON /69/ ABCO(20,200) ,ALRAY(500) ,BERAY(500)
1 ,COORDS(6,50) ,E(40,200) ,EBASE(200) ,ESOR(20,200)
2 ,GARAY(500) ,RED(20,200) ,TITLE(12) ,VMAT(20)
3 ,VOL(200) ,XLAM(200) ,XRAY(500) ,YRAY(500)
4 ,ZRAY(500)
      COMMON /ARRAYS/ ALK(200) ,ALPHA(40) ,BEK(200)
1 ,BETA(40) ,DEV(200) ,DK(200) ,EK(200)
2 ,ELAM(200) ,ELAMAT(20) ,ENAT(20) ,ENERGY(200)
3 ,GAK(200) ,GAMMA(40) ,REF(20) ,SORMAT(20)
4 ,SUMRAY(200) ,SUMSOR(200) ,X(40) ,XK(200)
5 ,Y(40) ,YK(200) ,Z(40) ,ZK(200)
6 ,ZLAM(20)
      COMMON /IARRAY/ IFLAG(40) ,IPOINT(50)
1 ,ISMAX(50) ,ISOR(20) ,JF(200) ,JFL(200)
2 ,JL(200) ,JM(200) ,JN(200) ,JP(200)
3 ,JRAY(200) ,JSOR(50) ,KBND(900) ,KFLAG(200)
4 ,KP(200) ,KTP(200) ,LAMP(50) ,LANGE(20)
5 ,LREFL(200) ,LTYPE(20) ,MAT(200) ,MFL(20)
6 ,MMAT(20) ,MMRAY(200)
      COMMON /PARAMS/ AL ,ALN ,ALR ,ALRF ,BE ,
1 ,BEN ,BER ,BERF ,DIS ,DMIN ,DZ ,EPS
2 ,GA ,GAN ,GAR ,GANF ,PER ,
3 ,SUMRA ,XI ,XNOW ,XOLD ,XREF ,XTRAN ,YI
4 ,YNOW ,YOLD ,YREF ,YTRAN ,ZI
5 ,ZNOW ,ZOLD ,ZREF ,ZTRAN
      COMMON /IPARAM/ FMAX ,INOW
1 ,IOPRAY ,IOPRNT ,IR ,IRAY ,JMAX ,JNOW
2 ,KINT ,KMAX ,KNOW ,MMAX ,MNOW ,MSOR ,JOLD
      INTEGER F,FMAX
C***** FIND THE COSINE OF THE ANGLE BETWEEN THE RAY AND THE
C***** NORMAL TO THE PLANE
      COSANG=AL*ALK(K)+BE*BEK(K)+GA*GAK(K)
      IF(ABS(COSANG)-1.E-01)103,103,101
C***** CALCULATE THE DISTANCE TO THE POINT OF INTERSECTION.
101 DMIT=(DK(K)-(ALK(K)*XI+BEK(K)*YI+GAK(K)*ZI))/COSANG
      IF(DMIT)103,104,104
103 DMIT=1.E21
104 CONTINUE
      CALL DEBUG(6,INTPLA)
      RETURN
      ENN

```

```

SUBROUTINE INTSPH(K,DNIT)
C*****
C***** INTER- FILES THE NEAREST INTERSECTION OF A RAY WITH A SPHERE.
C*****
COMMON /69/ ABCO(20,200) ,ALRAY(500) ,BCRAY(500)
1 ,COORDS(4,50) ,E(40,200) ,E0ASL(200) ,ESOR(20,200)
2 ,CAPAY(500) ,RED(20,200) ,TITLE(12) ,VMAT(20)
3 ,VOL(200) ,XLAN(200) ,XRAY(500) ,YMAT(500)
4 ,ZRAY(500)
COMMON /ARRAYS/ ALK(200) ,ALPHA(40) ,BEK(200)
1 ,AFTA(40) ,DEV(200) ,DK(200) ,EK(200)
2 ,ELAM(200) ,ELAMAT(20) ,EMAT(20) ,ENERGY(200)
3 ,EAK(200) ,GAPMA(40) ,REF(20) ,SCRMAT(20)
4 ,SIMPAY(200) ,SL*SOR(200) ,X(40) ,XK(200)
5 ,Y(40) ,YK(200) ,Z(40) ,ZK(200)
6 ,ZLAN(20)
COMMON /IARRAY/ IFLAG(40) ,IPGINT(50)
1 ,ISMAX(50) ,ISOR(20) ,JF(200) ,JFL(200)
2 ,JL(200) ,JM(200) ,JN(200) ,JP(200)
3 ,JPAY(200) ,JSOR(50) ,KNO(900) ,KFLAG(200)
4 ,KFL(200) ,KTYPE(200) ,LAMP(50) ,LAAGLF(20)
5 ,LREFL(200) ,LTYPE(20) ,MAT(200) ,MFL(20)
6 ,MAY(20) ,NUMRAY(200)
COMMON /PARAMS/ AL ,ALR ,ALRF ,RE
1 ,BEN ,BGR ,BGRF ,DIS ,DMIN ,DZ ,EPS
2 ,GA ,GAN ,GAR ,GAF ,PER ,
3 ,SUMRA ,XI ,XCH ,XCLD ,XREF ,XTRAN ,YI
4 ,YMIN ,YOLD ,YREF ,YTRAN ,ZI
5 ,ZMIN ,ZOLD ,ZREF ,ZTRAN
COMMON /PARAM/ FMAX ,IMAX ,INCH ,JMAX ,JACH ,JOLD
1 ,JOPRAY ,JOPENT ,IR ,IRAY ,JMAX ,JACH ,JSOR
2 ,KINT ,KMAX ,KACH ,KMAX ,KNO ,KSOR
INTEGER F,FMAX
C***** CALCULATE COEFFICIENTS NEEDED.
DELX=XI-X(K)
DELY=YI-Y(K)
DELZ=ZI-Z(K)
C=DELX**2+DELY**2+DELZ**2-CK(K)**2
B=DELX*AL+DELY*BE+DELZ*GA
SO=R**2-C
IF(C<0) GO TO 102,102
102 ROOT=SQRT(SO)
C***** FIND BOTH ROOTS AND CHOOSE SMALLEST POSITIVE VALUE.
DNIT1=R-ROOT
DNIT2=R+ROOT
DNIT=AMIN1(DNIT1,DNIT2)
IF(DNIT1<0.100+105
104 DNIT=AMAX1(DNIT1,DNIT2)
IF(DNIT1<0.101+105
101 DNIT=1.E+21
105 CONTINUE
CALL DEBUG(DNITSPH)
RETURN
END

```

```

SUBROUTINE JROUND(J)
C.....
C..... SUBROUTINE JROUND FINDS THE DISTANCE FROM POINT XI,YI,ZI
C..... IN THE DIRECTION AL,RE,GA TO EACH OF THE BOUNDARIES OF
C..... SEGMENT J AND CHOOSES THE SHORTEST POSITIVE DISTANCE.
C.....
COMMON /69/ ABCD(20,200) ,ALRAY(500) ,BERAY(500)
1 ,COORDS(6,50) ,E(40,200) ,EBASE(200) ,ESOR(20,200)
2 ,CARAY(500) ,REU(20,200) ,TITLE(12) ,VMAT(20)
3 ,VOL(200) ,XLAN(200) ,XRAY(500) ,YRAY(500)
4 ,ZRAY(500)
COMMON /ARRAYS/ ALK(200) ,ALPHA(40) ,BEK(200)
1 ,BETA(40) ,DEV(200) ,DK(200) ,EK(200)
2 ,FLAM(200) ,ELAMAT(20) ,EMAT(20) ,ENERGY(2,50)
3 ,GAK(200) ,GAMMA(40) ,REF(20) ,SONMAT(20)
4 ,SIMPAY(200) ,SUMSOR(200) ,X(40) ,XK(200)
5 ,Y(40) ,YK(200) ,Z(40) ,ZK(200)
6 ,ZLAN(20)
COMMON /IARRAY/ IFLAG(40) ,IPOINT(50)
,ISMAX(50) ,ISOR(20) ,JF(200) ,JFL(200)
,JL(200) ,JM(200) ,JN(200) ,JP(200)
,JRAY(200) ,JSOR(50) ,KBND(900) ,KFLAG(200)
,KP(200) ,KTP(200) ,LAMP(50) ,LANGE(20)
,LREFL(200) ,LTYPE(20) ,MAT(200) ,P(1,20)
,MAT(20) ,MUNRAY(200)
COMMON /PARAMS/ AL ,ALN ,ALR ,ALRF ,RE
,BEN ,BER ,BERF ,OIS ,DMIN ,LZ ,EPS
,GA ,GAN ,GAR ,GANF ,PER
,SUMRA ,XI ,XNOW ,XOLD ,XREF ,XTRAN ,YI
,YNOV ,YOLD ,YREF ,YTRAN ,ZI
,ZNOV ,ZOLD ,ZREF ,ZTRAN
COMMON /IPARAM/ FMAX ,IMAX ,INOW
,IOPRAY ,IOPRNT ,IR ,IRAY ,JMAX ,JNOW ,JOLD
,KINT ,KMAX ,KNOW ,MMAX ,MNOW ,MSOR
INTEGER F,FMAX
C..... SET INDEX FOR BOUNDARY ARRAY OF SEGMENT J.
NKP(J)
DIS=1.E20
C..... CHOOSE A BOUNDARY NUMBER K.
101 K=JARS(KBND(N))
C..... IF BOUNDARY HAS ALREADY BEEN CHECKED, GO TO 104.
IF(KFLAG(K))106,106,104
106 KFLAG(K)=1
KT=KTP(K)
C..... BRANCH ON BOUNDARY TYPE.
GO TO(100,200,300,400,500,600,700,800,900),KT
C..... FIND DISTANCE TO PLANE BOUNDARY NUMBER K,
100 CALL INTPLA(K,DMIT)
GO TO 102
C..... FIND DISTANCE TO CYLINDRICAL BOUNDARY NUMBER K,
200 CALL INTCYL(K,DMIT)
GO TO 102
C..... FIND DISTANCE TO CONICAL BOUNDARY NUMBER K,
300 CALL INTCON(K,DMIT)
GO TO 102
C..... FIND DISTANCE TO SPHERE.
400 CALL INTSPH(K,DMIT)
GO TO 102
C..... FIND DISTANCE TO CONE.
500 CALL INTCN(K,DMIT)
GO TO 102

```

```

C***** FIND DISTANCE TO PARABOLOID.
600 CALL IATPAR(K,DMIT)
GO TO 102
C***** FIND DISTANCE TO HYPERCLOID.
700 CALL IATHYP(K,DMIT)
GO TO 102
C***** FIND DISTANCE TO ELLIPSOID.
800 CALL INTELL(K,DMIT)
GO TO 102
C***** FIND DISTANCE TO HELIX.
900 CALL IATHEL(K,DMIT)
GO TO 102
C***** IS DISTANCE LESS THAN MINIMUM DISTANCE.
102 IF(DMIT-DIS)103,104,104
103 DIS=DMIT
KINT=K
C***** LOOK AT NEXT BOUNDARY.
104 N=N+1
IF(KAND(N))101,105,101
105 CONTINUE
CALL DEBUG(64,BOUND)
RETURN
END

```

```

SUBROUTINE LAMPAT
C.....
C..... LAMPAT READS LAMP DATA CARDS.
C.....
COMMON /69/ ARCO(20,200) ALRAY(500) BERAY(500)
1 CCHRS(4,50) E(40,200) EBASE(200) ESON(20,200)
2 GARAY(500) RED(20,200) TITLE(12) VMAT(20)
3 VOL(700) XLAM(200) XRAY(500) YMAT(500)
4 ZMAT(500)
COMMON /ARRAYS/ ALK(200) ALPHA(40) BEK(200)
1 RFTA(40) DEV(200) DK(200) EK(200)
2 FLAM(200) ELAMAT(20) EMAT(20) ENENG(200)
3 GAK(200) GAMMA(40) REF(20) SORMAT(20)
4 JPRAY(200) SUMSOR(200) X(40) XK(200)
5 Y(40) YK(200) Z(40) ZK(200)
6 ZLAM(20)
COMMON /IARRAY/ IFLAG(40) IPOINT(50)
1 ISMAX(50) ISOR(20) JF(200) JFL(200)
2 JL(200) JM(200) JN(200) JP(200)
3 JRAY(200) JSOR(50) KBND(900) KFLAG(200)
4 KP(200) KTYP(200) LAMP(50) LANGLE(20)
5 LRFFL(200) LTYPE(20) MAT(200) MFL(20)
6 NMAT(20) NUMRAY(200)
COMMON /PARAMS/ AL ALN ALR ALRF RE
1 BEN BER BEHF DIS DMIN DZ EPS
2 GA GAN GARF GARF PER
3 SUPRA XI XND XOLD XREF XTRAN YI
4 YND YOLD YREF YTHAN ZI
5 ZND ZOLD ZREF ZTRAN
COMMON /IPARAM/ FMAX INAX INON
1 IOPRAY IOPRNT IR IRAY JMAX JNON JOLD
2 KINT KMAX KND MMAX MNON MSOR
INTEGER F,FMAX
DIMENSION LANTAB(10),ISUBT(9),JNGT(9)
DATA(LANTAB,4,VOLUME,6,OUTSUR,6,INNSUR,6,ARRAYIMP,6(6))
DATA(1,ISUBT,4,INTEN1,6,INTEN2,6,INTEN3,6,INTEN4,
6,INTEN5,6,INTEN6,6,INTEN7,6,INTEN8,6,INTEN9)
DATA(JNGT,4,ANGLE1,6,ANGLE2,6,ANGLE3,6,ANGLE4,
6,ANGLE5,6,ANGLE6,6,ANGLE7,6,ANGLE8,6,ANGLE9)
C..... PRINT PAGE HEADING.
PRINT 10
10 FORMAT(1H1,5X,9H LAMP DATA//1H 6X LAMP,8X,
4H LAMP,3X,9H INTENSITY,7X,5H ANGLE,3X,9H INTENSITY/
1H 6H NUMBER,8X,4H TYPE,5X,7H ROUTINE,5X,7H ROUTINE/)
C..... READ A CARD
100 READ 11,LAMPNO,LAMPT,ISUBI,IANG,ELAMP
11 FORMAT(14,3(1X,A6),E12,4)
C..... LAMP NUMBER =ZERO INDICATES END OF DATA.
IF(LAMPNO)110,110,101
C..... PRINT LAMP DATA.
101 PRINT 12,LAMPNO,LAMPT,ISUBI,IANG,ELAMP
12 FORMAT(1H 16,3(6X,A6),E12,4)
C..... SEARCH TABLE FOR SOURCE TYPE.
DO 102 N=1,10
IF(LAMPT-LANTAB(N))102,104,102
102 CONTINUE
C..... SOURCE TYPE NOT IN TABLE.
CALL ERRPRT(6H LAMPAT)
C..... STORE SOURCE TYPE INDEX.
104 ISOR(LAMPNO)=N
C..... IF MULTIPLE WAVELENGTHS GO TO 106

```

```

      IF (FMAT-1) 105,105,106
C***** STORE SOURCE INTENSITY
      105 ESQ(LAMPNO,1)=ELAMP
      GO TO 107
C***** SEARCH TABLE FOR NAME OF INTENSITY ROUTINE.
      106 DO 109 IT=1,9
      IF (ISUM1-ISUMT(IT)) 104,109,100
      109 CONTINUE
C***** INTENSITY ROUTINE NOT IN TABLE.
      CALL EGPNT(6,LAMPDAT)
C***** GET INTENSITY AS A FUNCTION OF WAVELENGTH.
      109 CALL INTDAT(IT,LAMPNO)
C***** IF SURFACE SOURCE, GET NAME OF ANGLE ROUTINE.
      107 IF (ISOR(LAMPNO)-2) 111,112,111
      111 IF (ISOR(LAMPNO)-3) 113,112,113
      112 DO 114 IA=1,9
      IF (IANG-IANGT(IA)) 114,115,114
      114 CONTINUE
      CALL EGPNT(6,LAMPDAT)
      115 LAANGLE(LAMPNO)=IA
      113 GO TO 100
      110 RETURN
      END

```

```

SUBROUTINE MATDAT
C*****
C***** SUBROUTINE MATDAT READS IN MATERIAL DATA FOR EACH MATERIAL.
C*****
COMMON /55/ ABCO(20,200) ,ALRAY(500) ,BERAY(500)
1 ,CDOROS(4,50) ,E(40,200) ,EBASE(200) ,ESOR(20,200)
2 ,GARAY(500) ,RED(20,200) ,TITLE(12) ,VMAT(20)
3 ,VOL(200) ,ZLANI(200) ,XRAY(500) ,YRAY(500)
4 ,ZRAY(500)
COMMON /ARRAYS/ ALK(200) ,ALPHA(40) ,BEK(200)
1 ,BETA(40) ,DEV(200) ,DK(200) ,EK(200)
2 ,ELAM(200) ,ELAMAT(20) ,EMAT(20) ,ENERGY(200)
3 ,EAK(200) ,GAMMA(40) ,REF(20) ,SUMMAT(20)
4 ,SUMRAY(200) ,SUMSOR(200) ,X(40) ,XK(200)
5 ,Y(40) ,YK(200) ,Z(40) ,ZK(200)
6 ,ZLAN(20)
COMMON /ARRAY/ IFLAG(40) ,IPOINT(50) ,
1 ,ISMAX(50) ,ISOR(20) ,JF(200) ,JFL(200) ,
2 ,JL(200) ,JN(200) ,JN(200) ,JP(200) ,
3 ,JRAY(200) ,JSOR(50) ,KBD(900) ,KFLAG(200) ,
4 ,KP(200) ,KTP(200) ,LAMP(50) ,LAANGLE(20) ,
5 ,LREFL(200) ,LTYPE(20) ,MAT(200) ,MFL(20) ,
6 ,MAT(20) ,MURRAY(200)
COMMON /PARAMS/ AL ,ALN ,ALR ,ALRF ,BE ,
1 ,BER ,BERF ,DTS ,DMIN ,DZ ,EPS ,
2 ,GA ,GAN ,GAR ,GASF ,PER ,
3 ,SUMRA ,XI ,XMON ,XCLO ,XREF ,XTRAN ,YI ,
4 ,YMON ,YOLD ,YREF ,YTHAN ,ZI ,
5 ,ZMON ,ZOLD ,ZREF ,ZTHAN
COMMON /IPARAM/ FMAX ,IMAX ,INON ,
1 ,IOPRAY ,IOPRNT ,IR ,IRAY ,JMAX ,JMON ,JOLD ,
2 ,KINT ,KPAI ,KNOW ,NNAX ,NNON ,NSOR
INTEGER F,F-AX
DIMENSION LESC(6)
C***** PRINT PAGE HEADING.
PRINT 0
0 FORMAT(1H1,55X,13H MATERIAL DATA//10H MATERIAL,4X,5H INPUT,
1 ,6X,10H ABSORPTION,6X,10H REFRACTIVE,10X,6H LASING,
2 ,5X,11H DESCRIPTION//1H ,8H NUMBER,2X,7H ROUTINE,5X,
3 ,11H COEFFICIENT,11X,5H INDEX,6X,10H WAVELENGTH/)
MMAX=0
C***** READ ONE MATERIAL DATA CARD.
104 READ 10,M,IAB,ABCOI,REFI,ZLANI,DESC
10 FORMAT(14,1X,A6,3F7.5,6A6)
C***** IF MATERIAL NUMBER IS ZERO, ALL MATERIAL DATA HAS BEEN READ.
IF(M)103,110,103
C***** IF N GREATER THAN MMAX, MMAX=N.
103 IF(N-MMAX)102,102,101
101 MMAX=N
C***** STORE LASING WAVELENGTH AND REFRACTIVE INDEX.
102 REFI=REFI
ZLANI=ZLANI
C***** PRINT MATERIAL DATA.
PRINT 11,M,IAB,ABCOI,REFI,ZLANI,DESC
11 FORMAT(1H 11,1X,A6,3(4X,F12.6),6X,6A6)
C***** STORE ABSORPTION COEFFICIENT.
IF(IAB.NE.0H) GOTO 106
C***** IF NO WAVELENGTH-DEPENDENT ABSORPTION COEFFICIENT, STORE
C***** CONSTANT VALUE SPECIFIED IN ABCO(N,F).
DO 105 F=1,FMAX
105 ABCO(N,F) = ABCOI
GO TO 104
C***** IF 13, CO. DEPENDS ON WAVELENGTH, GET FROM SUBROUTINE.
106 CALL ARSORP(1AB,N)
GO TO 104
110 CONTINUE
RETURN
END

```

```

FUNCTION NINT(S)
C.....
C..... NINT FINDS COIL NUMBER IF WAY MOVES DISTANCE S.
C.....
COMMON /69/ ABCO(20,200) ALRAY(500) DEQAY(500)
1 COORDS(6,50) E(40,200) EBASE(200) ESOR(20,200)
2 GARAY(500) RED(20,200) TITLE(12) VMAT(20)
3 VOL(200) XLAM(200) XRAY(500) YRAY(500)
4 ZRAY(500)
COMMON /ARRAYS/ ALK(200) ALPHA(40) BEK(200)
1 RETA(40) DEV(200) DK(200) EK(200)
2 ELAM(200) ELAMAT(20) EMAT(20) ENERGY(200)
3 GAK(200) GAMMA(40) REF(20) SORMAT(20)
4 SUMRAY(200) SUMSOR(200) X(40) XK(200)
5 Y(40) YK(200) Z(40) ZK(200)
6 ZLAM(20)
COMMON /IARRAY/ IFLAG(40) IPOINT(50)
ISMAX(50) ISOR(20) JF(200) JFL(200)
JL(200) JM(200) JP(200)
JRAY(200) JSOR(50) KNO(900) KFLAG(200)
KP(200) KTYPI(200) LAMP(50) LANGLE(20)
LREFL(200) LTYPE(20) MAT(200) NFL(20)
MMAT(20) MMURAY(200)
COMMON /PARAMS/ AL ALN ALR ALRF RE
BER BERF DIS DWIN DZ EPS
GA GAN GAR GARF PER
SUMRA XI INOW XOLD XREF XTRAN YI
YNOW YOLD YREF YTRAN ZI
ZNOW ZOLD ZREF ZTHAN
COMMON /IPARAM/ FMAX INOW
IOPRAY IOPRNT IR IRAY JMAX JNOW JOLD
KINT KMAX KNOW NMAX NNOV NSOR
INTEGER F,FMAX
COMMON /HELCON/ A,ABAR,B,BI,BIP,BP,CAP,DELTH,DELT,DELY,DELZ,DWIT,
PSI7,REV,RNO,STH,TH,THI,SORKS,RKS,CSTH,SNTH,RCSTH,RSNTH,BSTH,
COSTAN,THZ,DTNZ
BSO=RE+S-DELY
ASO=AL+S-DELT
IF (BSO.EQ.0.0.AND.ASO.EQ.0.0) ASO=1.0
THST=ATAN2(BSO,ASO)
NST=FIX((THST-PSI7)/REV*.1.E+4)-10000
THZ=THST-NST*REV
NINT=FIX((6A*S-DELZ-CAP*THZ)/(CAP*REV)*.1.E+4.5)-10000
RETURN
END

```



```

SUBROUTINE NORMAL
C.....
C..... SUBROUTINE NORMAL CALCULATES THE NORMAL TO BOUNDARY KNOWN.
C.....
COMMON /69/ ABCO(20,200) ,ALRAY(500) ,BERAY(500)
1 ,CDOZNS(6,50) ,E(40,200) ,EBASE(200) ,ESON(20,200)
2 ,GARAY(500) ,REDI(20,200) ,TITLE(12) ,VMAT(20)
3 ,VOL(200) ,XLAM(200) ,XRAY(500) ,YRAY(500)
4 ,ZRAY(500)
COMMON /ARRAYS/ ALK(200) ,ALPHA(40) ,BEK(200)
1 ,BETA(40) ,DEV(200) ,DK(200) ,EK(200)
2 ,ELAM(200) ,ELAMAT(20) ,EMAT(20) ,ENERGY(200)
3 ,GAK(200) ,GAMMA(40) ,REF(20) ,SUMMAT(20)
4 ,SUMRAY(200) ,SUMSOR(200) ,X(40) ,XK(200)
5 ,Y(40) ,YK(200) ,Z(40) ,ZK(200)
6 ,ZLAM(20)
COMMON /IARRAY/ IFLAG(40) ,IPPOINT(50)
,ISMAX(50) ,ISOR(20) ,JF(200) ,JFL(200)
,JL(200) ,JM(200) ,JN(200) ,JP(200)
,JRAY(200) ,JSOR(50) ,KMD(900) ,KFLAG(200)
,KP(200) ,KTP(200) ,LAMP(50) ,LAAGLE(20)
,LREFL(200) ,LTYPE(20) ,MAT(200) ,NFL(20)
,OMAT(20) ,OMRAY(200)
COMMON /PARAMS/ AL ,ALN ,ALR ,ALRF ,RE
,GA ,GER ,GARF ,DIS ,DMIN ,DZ ,EPS
,SUMRA ,XI ,KNOW ,XOLD ,XREF ,XTRAN ,YI
,YNOV ,YOLD ,YREF ,YTRAN ,ZI
,ZNOV ,ZOLD ,ZREF ,ZTRAN
COMMON /IPARAP/ FMAX ,INAX ,INOW
,IOCRAY ,IOCRANT ,IR ,IRAY ,JMAX ,JNOW
,KIAT ,KMAX ,KNOW ,NMAX ,NNOW ,NSOR ,JOLD
INTEGER F,FMAX
K=KCN
C..... GET GEOMETRY TYPE FOR BOUNDARY NUMBER K.
KT=KTP(K)
C..... BRANCH ON GEOMETRY TYPE.
GO TO(100,200,300,400,500,600,700,800,900),KT
C..... FIND NORMAL TO PLANE.
100 CALL PLANOR(K)
GO TO 101
C..... FIND NORMAL TO CYLINDER.
200 CALL CYLOR(K)
GO TO 101
C..... FIND NORMAL TO CONIC.
300 CALL CONOR(K)
GO TO 101
C..... SPHERE
400 CALL SPHOR(K)
GO TO 101
C..... CONE
500 CALL CANOR(K)
GO TO 101
C..... PARABOLOID
600 CALL PARHOR(K)
GO TO 101
C..... HYPERBOLOID
700 CALL HYPHOR(K)
GO TO 101
C..... ELLIPSOID
800 CALL ELLHOR(K)
GO TO 101
C..... HELIX
900 CALL HELHOR(K)
GO TO 101
101 CONTINUE
CALL CDBUG(6,NORMAL)
RETURN
END

```

SUBROUTINE PACKUM

 ***** SUBROUTINE PACKUM PACKS THE RAY ARRAYS TO ELIMINATE THE
 ***** STORAGE USED BY RAYS WHICH HAVE BEEN DESTROYED.

COMMON /A9/ ABCD(20,200) ALRAY(500) BLRAY(500)
 1 COORDS(16,50) E(140,200) EBASE(200) ESCP(20,200)
 2 CARAY(500) RED(20,200) TITLE(12) VPAT(20)
 3 VOL(200) XLAN(200) XRAY(500) YRAY(500)
 4 ZRAY(500)

COMMON /ARRAYS/ ALK(200) ALPHA(40) BEK(200)
 1 RFTZ(40) DEV(200) CK(200) EK(200)
 2 ELAT(200) ELAMAT(20) EMAT(20) ENERGY(200)
 3 GAK(200) GAMMA(40) REF(20) SCMAT(20)
 4 SUMRAY(200) SUMSOR(200) X(40) XK(200)
 5 Y(40) YK(200) Z(40) ZK(200)
 6 ZLAN(20)

COMMON /ARRAY/ IFLAG(40) IPOINT(50)
 ISAX(50) ISOR(20) JF(200) JFL(200)
 JL(200) JN(200) JN(200) JP(200)
 JRAY(200) JSOR(50) KQAO(900) KFLAG(200)
 KP(200) KYP(200) LAMP(50) LAAGLE(20)
 LREFL(200) LTYPE(20) MAT(200) MFL(20)
 MNAT(20) MURRAY(200)

COMMON /PARAMS/ AL ALN ALR ALRF RE
 BEN BEN BEBF DIS DMIN DZ EPS
 GA GAN GAR GARF PER
 SUMRA XI XAO XOLD XREF XTRAN VI
 YNTH YCLD YREF YTRAN ZI
 ZNCH ZCLD ZGEF ZTRAN
 COMMON /IPARAM/ IMAX INCH INCH INCH JOLD
 IOPRAY ICORNT IR IRAY JMAX JMAX JMAX
 KRAY KPAK KPAK KPAK KPAK KPAK KPAK
 INTEGER F,FMAX

***** SET EFFECTIVE INDEX=1.

IEF=1

***** LOOP THROUGH RAY STORAGE.

DO 100 I=1,IMAX

IF(IFLAG(I))100,100,101

***** RE-STORE RAY ON EFFECTIVE INDEX.

101 ALPHA(IEF)=ALPHA(I)

BETA(IEF)=BETA(I)

GAMMA(IEF)=GAMMA(I)

X(IEF)=X(I)

Y(IEF)=Y(I)

Z(IEF)=Z(I)

DO 103 F=1,FMAX

E(IEF,F)=E(I,F)

103 CONTINUE

IFLAG(IEF)=1

***** INCREASE EFFECTIVE INDEX BY 1.

IEF=IEF+1

100 CONTINUE

IMAX=IEF-1

RETURN

END

```

SUBROUTINE PARNOR(K)
C*****
C***** PARNOR CALCULATES THE DIRECTIONAL COSINES OF THE NORMAL
C***** TO A POINT ON THE SURFACE OF A PARABOLOID.
C*****
COMMON /69/ ABCO(20,200) *ALRAY(500) *BERAY(500)
1 *CORRPS(6,50) *E(40,200) *ESASE(200) *ESCH(20,200)
2 *GARAY(500) *RED(20,200) *TITLE(12) *VNAT(20)
3 *VOL(200) *XLAM(200) *XRAY(500) *YNAT(500)
4 *ZRAY(500)
COMMON /ARRAYS/ ALK(200) *ALPHA(40) *BEK(200)
1 *BETA(40) *DEV(200) *DK(200) *EK(200)
2 *ELAM(200) *ELAMAT(20) *EMAT(20) *ENERGY(200)
3 *EAK(200) *GAMMA(40) *REF(20) *SOMMAT(20)
4 *SMRAY(200) *SUMSOR(200) *X(40) *XK(200)
5 *Y(40) *YK(200) *Z(40) *ZK(200)
6 *ZLAM(20)
COMMON /IARRAY/ IFLAG(40) *IPOINT(50)
*ISMAX(50) *ISOR(20) *JF(200) *JFL(200)
*JL(200) *JM(200) *JP(200)
*JRAY(200) *JSOR(50) *KMD(900) *KFLAG(200)
*KP(200) *KTP(200) *LAMP(50) *LANGLE(20)
*REFL(200) *LTYPE(20) *MAT(200) *NFL(20)
*MMAT(20) *MMRAY(200)
COMMON /PARAMS/ AL *ALN *ALR *ALRF *RE
*BEA *BER *BERF *DIS *DMIN *DZ *EPS
*GA *GAN *GAR *GARF *PER
*SOJRA *XI *XND *XCLO *XREF *XTRAN *YI
*YND *YOLD *YNEF *YTRAN *ZI
*ZND *ZOLD *ZREF *ZTRAN
COMMON /IPARAM/ FMAX *INAX *INOW
*IOFFAY *IOFFNT *IR *IRAY *JMAX *JNOW
*KNAT *KMAX *KNO *KMAX *KNO *NSOR
INTEGER F,FMAX
C***** CALCULATE COEFFICIENTS NEEDED.
DELX=XND-XK(K)
DELY=YAND-YK(K)
DELZ=ZAND-ZK(K)
C***** CALCULATE LENGTH OF NORMAL VECTOR.
SQ=SQRT(DELX**2+DELY**2+DELZ**2)
C***** CALCULATE DIRECTIONAL COSINES OF NORMAL.
ALA=DELX/SQ
BEA=DELY/SQ
GA=(-2.0*DK(K))/SQ
C***** CHECK DIRECTIONAL COSINES.
CALL CHECK(5,PARNOR,ALA,BEA,GAN)
CALL DEBUG(6,PARNOR)
RETURN
END

```

```

SURROUTINE PERCNT
C*****
C***** SURROUTINE PERCNT DETERMINES WHAT PERCENT OF THE RAY IS
C***** TRANSMITTED. USES EQUATION FOR DIELECTRICS.
C*****
COMMON /69/ ABCO(20,200) ,ALRAY(500) ,BLRAY(500)
1 ,CNGPCS(6,50) ,E(40,200) ,EBASE(200) ,ESOR(20,200)
2 ,GARAY(500) ,RED(20,200) ,TITLE(12) ,VMAI(20)
3 ,VOL(200) ,XLAN(200) ,XRAY(500) ,YHAT(500)
4 ,ZRAY(500)
COMMON /ARRAYS/ ALK(200) ,ALPHA(40) ,BEK(200)
1 ,DETA(40) ,DEVI(200) ,DK(200) ,EK(200)
2 ,ELAM(200) ,ELAMAT(20) ,EMAT(20) ,ENERGY(200)
3 ,GAK(200) ,GAMMA(40) ,REF(20) ,SGRMAT(20)
4 ,SUMRAY(200) ,SUMSOR(200) ,X(40) ,XK(200)
5 ,Y(40) ,YK(200) ,Z(40) ,ZK(200)
6 ,ZLAN(20)
COMMON /IARRAY/ IFLAG(40) ,IPOINT(50)
,ISMAX(50) ,ISOR(20) ,JF(200) ,JFL(200)
,JL(200) ,JM(200) ,JN(200) ,JP(200)
,JRAY(200) ,JSOR(50) ,KNO(900) ,KFLAG(200)
,KP(200) ,KTYP(200) ,LAMP(50) ,LANGE(20)
,LRFFL(200) ,LTYPE(20) ,MAT(200) ,MFL(20)
,MHAY(20) ,MURRAY(200)
COMMON /PARAMS/ AL ,ALN ,ALR ,ALRF ,BE ,
,BEN ,BER ,BERF ,DIS ,DMIN ,DZ ,EPS ,
,GA ,GAR ,GARF ,GOLD ,GREF ,GTRAN ,YI ,
,SUNGA ,XI ,XNO ,XOLD ,XREF ,XZ ,
,YNO ,YOLO ,YREF ,YTRAN ,ZI ,
,ZNO ,ZOLO ,ZREF ,ZTRAN
COMMON /IPARAM/ FMAX ,IMAX ,INOW ,
,IOPRAY ,IOPRYT ,IR ,IRAY ,JMAX ,JNOW ,JOLD ,
,KINT ,KMAX ,KNOW ,MMAX ,MNOW ,MSOR
INTEGER F,FMAX
C***** SET INDICES FOR MATERIALS ON EACH SIDE OF BOUNDARY.
M1=MAT(JOLD)
M2=MAT(JNOW)
C***** FIND COS OF ANGLE BETWEEN INCOMING RAY AND NORMAL.
COSA=ABS(AL*ALN+BE*BER+GA*GAR)
C***** FIND COS OF ANGLE BETWEEN TRANSMITTED RAY AND NORMAL.
COST=ABS(ALRF*ALN+BERF*BER+GARF*GAR)
C***** FIND RATIO OF REFRACTIVE INDICES.
EM=REF(M1)/REF(M2)
C***** SOLVE EQUATIONS FOR BOTH MAJOR POLARIZATION AXES(RS,RP).
RS=((COSA-EM*COSA)/(COST-EM*COSA))**2
TMP1=COSA*COST
TMP2=EM*(1.0-COSA**2)
RP=RS*((TMP1-TMP2)/(TMP1+TMP2))**2
C***** FIND PERCENTAGE OF TRANSMISSION.
PER=1.0-(RS+RP)/2.0
104 RETURN
END

```

```

SUBROUTINE PERTAB(AA,BB,GG)
C.....
C..... PERTAB PERTURBS DIRECTION OF RAY IF NEEDED.
C.....
  IF (AA**2+BB**2.LT.1.E-7) AA=1.E-3
  IF (GG**2.LT.1.E-7) GG=1.E-3
  P=SQRT(AA**2+BB**2+GG**2)
  AA=AA/P
  BB=BB/P
  GG=GG/P
  RETURN
END

```

```

SUBROUTINE PLANOR(K)
C*****
C***** PLANOR RETRIEVES THE NORMAL TO A PLANE.
C*****
COMMON /C9/      A=C0(20,200)    ALRAY(500)    BERAY(500)
1  COORDS(6,50)   E(40,200)        EBASE(200)    ESOM(20,200)
2  GARAY(500)     REG(20,200)       VTITLE(12)    VMAT(20)
3  VOL(200)       XLAM(200)         XRAY(500)     YRAY(500)
4  ZRAY(500)
COMMON /ARRAYS/  ALX(200)          ALPHA(40)      BEX(200)
1  RETA(40)       DEV(200)          DM(200)       EX(200)
2  FLA(200)       ELAMAT(20)        EPAT(20)      EENERGY(200)
3  GAX(200)       GAMMA(40)         REF(20)       SCMAT(20)
4  SPURAY(200)    SL-SOR(200)       X(40)         XK(200)
5  Y(40)          YK(200)           Z(40)         ZK(200)
6  ZLAW(20)
COMMON /IARRAY/  IFLAG(40)         IPOINT(50)
1  ISPAK(50)      ISOM(20)          JF(200)       JFL(200)
2  JL(200)        JM(200)           JN(200)       JP(200)
3  JRAY(200)      JSOR(50)          KND(100)      KFLAG(200)
4  KEP(200)       KTYPE(200)        LAMP(50)      LANGLE(20)
5  LREFL(200)     LTYPE(20)         MAT(200)      MFL(20)
6  MNAT(20)       MNURAY(200)
COMMON /PARAMS/  AL      ALR      ALRF      DE      :
1  DEN      DER      DERF      DIS      DMIN      DZ      EPS      :
2  GA      GAN      GAR      GARF      PER      :
3  SL-RA      XI      XND      XOLD      XREF      XTRAN      YI      :
4  YL-RA      YOLD      YREF      YTRAN      ZI      :
5  ZND      ZOLD      ZREF      ZTRAN
COMMON /IPARL/   FLX      INAX      INON      :
1  IOPRAY      IOPRNT      IR      IRAY      JMAX      JNAX      JOLD      :
2  KINT      KMAX      KND      KMAX      KND      KSOR
INTEGER F,FMAX
C***** SET DIRECTIONAL COSINES OF NORMAL TO THE PLANE.
ALA=ALX(K)
BE=BEK(K)
GA=GAX(K)
C***** CHECK DIRECTIONAL COSINES.
CALL CHECK(6,PLANOR,ALA,BE,GAN)
CALL DEBUG(6,PLANOR)
RETURN
END

```

```

SUBROUTINE POINT(K,ALPT,BEPT,GAPT,XXX,YYY,ZZZ)
C.....
C..... POINT LOCATES THE POINT ON THE HELICAL WIRE THAT IS CLOSEST
C..... TO A SPECIFIED POINT AND CALCULATES THE X,Y,Z COMPONENTS OF
C..... THE VECTOR BETWEEN THE TWO POINTS.
C.....
COMMON /69/      ABCD(20,200)  ALRAY(500)  BERAY(500)
1  COORDS(6,50)  E(40,200)      EBASE(200)  ESON(20,200)
2  GARAY(500)    RED(20,200)    TITLE(12)  VMAT(20)
3  VCL(200)      XLAM(200)      KRAY(500)    YRAY(500)
4  ZFEY(500)
COMMON /ARRAYS/  ALK(200)      ALPHA(40)    BEK(200)
1  BETA(40)      DEV(200)      DK(200)      EK(200)
2  FLAM(200)     ELAMAT(20)    EMAT(20)    EMENGY(200)
3  GAK(200)      GAMMA(40)    REF(20)     SORMAT(20)
4  SUMRAY(200)   SUMSOR(200)   X(40)      XK(200)
5  Y(40)         YK(200)      Z(40)        ZK(200)
6  ZLAM(20)
COMMON /IARRAY/  IFLAG(40)    IPOINT(50)
1  ISMAX(50)     JSOR(20)      JF(200)     JFL(200)
2  JL(200)       JM(200)      JN(200)     JP(200)
3  JRAY(200)     JSOR(50)     KBAD(100)   KFLAG(200)
4  KP(200)       KTYPE(200)   LAMP(50)    LANGLE(20)
5  LREFL(200)   LTYPE(20)    MAT(200)    WFL(20)
6  MURAY(200)
COMMON /PARAMS/  AL          ALN          ALR          ALRF          RE
1  BEA          BER          BERF         DIS          DMIN          DZ          EPS
2  GA          GAN          GAR          GARF          PER
3  SUMRA        XI          XNO          XCLO          XREF          XTRAN          YI
4  YNO          YOLD          YREF          YTHAN          ZI
5  ZNO          ZOLD          ZREF          ZTHAN
COMMON /IPARAM/  FMAX        INAX          INOV
1  IOPRAY        IOHNT        IR          IRAY          JMAX          JNO          JOLD
2  KINT          KMAX        KNO          KNOX          JNSOR
INTEGER F,FMAX
DATA(TOPI=6.28310531)
C..... CALCULATE PARAMETERS NEEDED.
DELX=XXX-XK(K)
DELY=YYY-YK(K)
DELZ=ZZZ-ZK(K)
PSI=ATAN2(DELY,DELX)
CAP=ALK(K)
RMOX=DELX*REK(K)
RMOY=DELY*REK(K)
C..... CALCULATE COIL NUMBER AND STARTING ANGLE(TH).
TH=IFX((DELZ-CAP*PSI)/(TOPI*CAP)*1.E+0.5)-10000
TH=PSI+TOPI*AN
C..... ITERATE UNTIL ANGLE(TH) HAS CONVERGED.
100 STH=SIGN(TH)
CTH=COS(TH)
FT=STH*RMOX-C*H*MOY-CAP*(DELZ-CAP*TH)
FP=CTH*RMOX+STH*MOY*CAP*2
FPP=STH*RMOX*CTH*MOY
IF(ABS(FT-FPP).LT.1.E-4*FP*2)GO TO 102
TH=(FT-FP*SQRT(1.0-2.0*FT*FPP/FP*2))/FPP
GO TO 103
102 DTH=-FT/FP
103 TH=TH+DTH
IF(ABS(DTH)-1.E-5)101,100,100
C..... CALCULATE X,Y,Z COMPONENTS
101 ALPT=(DELX-BEK(K)*COS(TH))
BEPT=(DELY-BEK(K)*SIN(TH))
GAPT=(DELZ-CAP*TH)
RETURN
END

```

```

FUNCTION RAND(MNHN)
C*****
C***** FUNCTION RAND GENERATES A RANDOM NUMBER BETWEEN 0.0 AND 1.0.
C*****
      COMMON/IPARAM/      FMAX      OIMAX      OINON      ,
      OIOPRAT , OIOPRAT , IR      OIRAY      OJMAX      OJNON      , JOLD ,
      OINT      , KMAY      , KNO      , PMAX      , MNON      , NSOR
C***** STORE 2*35 -- 1 AS CONSTANT
      DATA(J=34359738367)
C***** THE PSEUDO-RANDOM NUMBER IS GENERATED BY THE MIXED CONGRUEN-
C***** TIAL METHOD, USING A PRIME MULTIPLIER AND MODULUS.
C***** FIND  $IR = 5 \cdot 15 \cdot (1/2 - \text{SGT}(1/6)) \cdot 2 \cdot 35 \cdot 1$ 
      I = IR*30517576125 + 7741067067
C***** TURN OFF OVERFLOW INDICATION IF ON
      IF OVERFLOW FAULT 1,1
C***** REDUCE RESULT MODULO 2**35
      I IR = I .AND. J
C***** FLOAT RESULT
      RIP = IR
C***** MAKE 0 .LE. RESULT .LT. 1
      RAN = RIP/34359738368.
      RETURN
      END

```



```

SUBROUTINE RAY(IFLG)
C.....
C..... SUBROUTINE RAY PICKS THE LAST RAY STORED. IF NO RAYS EXIST,
C..... A NEW RAY IS GENERATED.
C.....
COMMON /69/ ABCD(20,200) ,ALRAY(500) ,BERAY(500)
1 ,COORDS(16,50) ,E(40,200) ,EBASE(200) ,ESOR(20,200)
2 ,RARRAY(500) ,RED(20,200) ,TITLE(12) ,VMAT(20)
3 ,VOL(200) ,XLAM(200) ,XRAY(500) ,YHAY(500)
4 ,7RAY(500)
COMMON /ARRAYS/ ALK(200) ,ALPHA(40) ,BEK(200)
1 ,PFTA(40) ,DEV(200) ,DK(200) ,ER(200)
2 ,ELAM(200) ,ELAMAT(20) ,EMAT(20) ,ENERGY(200)
3 ,RAK(200) ,GAMMA(40) ,REF(20) ,SOMMAT(20)
4 ,SINRAY(200) ,SUMSON(200) ,X(40) ,XK(200)
5 ,Y(40) ,YK(200) ,Z(40) ,ZK(200)
6 ,ZLAM(20)
COMMON /JARRAY/ IFLAG(40) ,IPOINT(50)
1 ,ISMAX(50) ,ISOR(20) ,JF(200) ,JFL(200)
2 ,JL(200) ,JM(200) ,JM(200) ,JP(200)
3 ,JRAY(200) ,JSOR(50) ,KEND(900) ,KFLAG(200)
4 ,KP(200) ,KTYP(200) ,LAMP(50) ,LANGLE(20)
5 ,LREFL(200) ,LTYPE(20) ,MAT(200) ,NFL(20)
6 ,MMAT(20) ,MMRAY(200)
COMMON /PARAMS/ AL ,ALN ,ALR ,ALRF ,RE
1 ,BEN ,BER ,BERF ,DIS ,DMIN ,DZ ,EPS
2 ,GA ,GAN ,GAR ,GANE ,PER
3 ,SUMRA ,XI ,XNOW ,XOLD ,XREF ,XTRAM ,YI
4 ,YNC ,YOLD ,YREF ,YTRAM ,ZI
5 ,ZAC ,ZOLD ,ZREF ,ZTRAM
COMMON /IPARAM/ FMAX ,IMAX ,INOW
1 ,IOPRAY ,IOPRNT ,IR ,IRAY ,JMAX ,JNOW ,JOLD
2 ,KINT ,KNAX ,KNOW ,MMAX ,MNOW ,NSCR
INTEGER F,FMAX
DIMENSION SUMMAT(100)
DATA(NPAC=0)
C..... IF ARRAY IS FULL...PACK ARRAY.
IF (IMAX-39) 166,167,167
167 CALL PACKUM
C..... IF ARRAY IS STILL FULL. DESTROY RAY AND PROCEED TO NEXT RAY.
IF (IMAX-39) 166,166,168
168 PRINT 35,IRAY
35 FORMAT('H ,10MRAY NUMBER,16,33H HAS TOO MANY LEGS. RAY DESTROYED)
C..... SET RAY FLAGS TO 0.
DO 169 I=1,IMAX
IFLAG(I)=0
DO 169 F=1,FMAX
E(I,F)=0.0
169 CONTINUE
IMAX=0
GO TO 104
C..... EVERY TEN RAYS. PACK THE RAY ARRAYS.
166 NPAC=NPAC+1
IF (NPAC-10) 102,103,103
103 CALL PACKUM
NPAC=0
102 CONTINUE
C..... IF NO RAYS EXIST IN STORAGE GO TO 104.
IF (IMAX) 104,104,105
C..... LOOK FOR LAST RAY STORED.
105 DO 100 II=1,IMAX

```

```

      I=IMAX-11+1
      IF(IFLAG(1))100,100,101
100 CONTINUE
      IMAX=0
C***** GENERATE A NEW RAY.
104 CALL IRANK
      DO 107 J=1,JPAX
      SUMSQ(J)=SUMSQ(J)+(SUMRAY(J))**2
      ENERGY(J)=ENERGY(J)+SUMRAY(J)
      NMAT(J)
      IF(N)111,111,112
112 SUMMAT(N)=SUMMAT(N)+SUMRAY(J)
111 CONTINUE
      IF(JFL(J))107,107,108
108 SUMRAY(J)=0.0
      JFL(J)=0
      SUMRAY(J)=SUMRAY(J)+1
107 CONTINUE
      DO 109 N=1,NMAX
      IF(NFL(N))109,109,110
110 NMAT(N)=NMAT(N)+1
      SUMMAT(N)=SUMMAT(N)+SUMRAY(J)**2
      NFL(N)=0
109 SUMMAT(N)=0.0
C***** IF NO RAY EXISTS, SET FLAG AND RETURN.
      IF(IMAX)106,106,105
106 IFLS=-1
      RETURN
C***** SET PARAMETERS FOR TRACING A NEW RAY.
101 INO=1
      JNOV=JOLD
      X1=X(1)
      Y1=Y(1)
      Z1=Z(1)
      XOLD=X1
      YOLD=Y1
      ZOLD=Z1
      AL=ALPHA(1)
      BE=BETA(1)
      GA=GAMMA(1)
      CALL CHECK(6H RAY,AL,BE,GA)
      IFLG=1
      CALL DEBUG(6H RAY)
      RETURN
      END

```

```

SUBROUTINE RAYSOR(ICOUNT)
C*****
C***** RAYSOR FETCHES A RAY FROM AN INPUT TABLE.
C*****
COMMON /69/ ABCD(20,200) ,ALRAY(500) ,BERAY(500)
1 ,CHORDS(6,50) ,E(40,200) ,ERASE(200) ,ESOR(20,200)
2 ,GARAY(500) ,RED(20,200) ,TITLE(12) ,YMAT(20)
3 ,VOL(200) ,XLAM(200) ,XRAY(500) ,YRAY(500)
4 ,ZRAY(500)
COMMON /ARRAYS/ ALK(200) ,ALPHA(40) ,BEK(200)
1 ,BETA(40) ,DEV(200) ,DK(200) ,EK(200)
2 ,ELAM(200) ,ELAMAT(20) ,ENAT(20) ,ENERGY(200)
3 ,GAK(200) ,GAMMA(40) ,REF(20) ,SUMMAT(20)
4 ,SUMRAY(200) ,SUNSOR(200) ,X(40) ,ZE(200)
5 ,Y(40) ,YK(200) ,Z(40) ,ZK(200)
6 ,ZLAM(20)
COMMON /IARRAY/ IFLAG(40) ,IPOINT(50)
,ISMAX(50) ,ISOR(20) ,JF(200) ,JFL(200)
,JL(200) ,JN(200) ,JN(200) ,JP(200)
,JRAY(200) ,JSOR(50) ,KEND(900) ,KFLAG(200)
,KP(200) ,KTYP(200) ,LAMP(50) ,LANGE(20)
,LREFL(200) ,LTYPE(20) ,MAT(200) ,NFL(20)
,OMAT(20) ,MURRAY(200)
COMMON /PARAMS/ AL ,ALN ,ALR ,ALRF ,RE
,BEL ,BER ,BERF ,DIS ,DMIN ,DZ ,EPS
,GA ,GAN ,GAR ,GARF ,PER
,SUMRA ,XI ,XNOW ,XCLO ,XREF ,XTRAN ,YI
,YNOW ,YOLD ,YREF ,YTRAN ,ZI
,ZNEW ,ZOLD ,ZREF ,ZTRAN ,ZINOW
COMMON /IPARAM/ FMAX ,INOW ,JMAX ,JNOW ,JOLD
,IOPRAY ,IOPRNT ,IR ,IRAY ,JMAX ,JNOW ,JSOR
,INT ,KNOW ,KNOW ,KNOW ,KNOW ,KNOW ,KNOW
INTEGER F,FMAX
IP=IPOINT(INOW)*ICOUNT-1
X(INOW)=XRAY(IP)
Y(INOW)=YRAY(IP)
Z(INOW)=ZRAY(IP)
ALPHA(INOW)=ALRAY(IP)
BETA(INOW)=BERAY(IP)
GAMMA(INOW)=GARAY(IP)
CALL CHECK(6,MURRAY,ALPHA(INOW),BETA(INOW),GAMMA(INOW))
RETURN
END

```

SUBROUTINE RCYL

***** RCYL CALCULATES THE INTERSECTION DISTANCE FOR SMALL COSTAN.
 ***** PART OF HELIX INTERSECTION CALCULATION.

```

COMMON /69/      ABCD(20,200)  ,ALRAY(500)  ,BERAY(500)
1  ,COORDS(16,50) ,E(40,200)    ,EBA'E(200)  ,ESOR(20,200)
2  ,GARAY(500)    ,RED(20,200)  ,TITL'E(12)  ,VMAT(20)
3  ,VOL(200)      ,XLAP(200)    ,XRAY(500)    ,YMAT(500)
4  ,ZRAY(500)

COMMON /ARRAYS/  ALK(200)      ,ALPHA(40)    ,BEK(200)
1  ,RETA(40)      ,DEV(200)     ,OK(200)      ,EK(200)
2  ,ELAP(200)     ,ELAMAT(20)   ,EMAT(20)    ,ENENGY(200)
3  ,GAK(200)      ,GAMMA(40)    ,REF(20)      ,SGMMAT(20)
4  ,SUMRAY(200)   ,SUMSOR(200)  ,X(40)      ,XK(200)
5  ,Y(40)         ,YK(200)     ,Z(40)      ,ZK(200)
6  ,ZLAM(20)

COMMON /IARRAY/  IFLAG(40)    ,IPCINT(50)
,ISMAX(50)      ,ISOR(20)     ,JF(200)
,JL(200)        ,JM(200)     ,JM(200)
,JRAY(200)      ,JSOR(50)    ,JND(900)
,KP(200)        ,KTP(200)    ,LAMP(50)
,LREFL(200)     ,LTYPE(20)   ,MAT(200)
,MMAT(20)       ,MURAY(200)  ,MFL(20)

COMMON /PARAMS/  AL          ,ALN          ,ALR          ,ALRF          ,RE
,GA              ,BER        ,DIS          ,GMIN          ,DZ          ,EPS
,SUMRA          ,XI          ,GAR          ,GARF          ,PER
,YNOW           ,YCLO        ,XNOW        ,XCLO          ,XREF          ,XTRAN          ,VI
,ZAOW           ,ZCLO        ,YREF        ,YTRAN          ,ZI
COMMON /IPARLH/  FMAX        ,ZTRAN          ,IMAX          ,INOW
,IOPRAY         ,ICPANT       ,IR          ,IGAY          ,JMAX          ,JNOW          ,JOLD
,KINT           ,KNAX        ,KNOW        ,KNAX          ,KNOW          ,KNOR

INTEGER F,FMAX
COMMON /HELCC/  A,ABAR,B,B1,BIP,BP,CAP,DELT,DELX,DELY,DELZ,DWIT,
,PSIZ,REV,RND,STH,TH,THI,SORRS,RKS,CSTN,SATN,RCSTN,RSATN,BSTN,
,COSTAN,THZ,OTMZ
CALL T-OFS
  
```

```

1  CONTINUE
***** CALCULATE PARAMETERS NEEDED
IR=STH*AL-DELX-RCSTN
YR=STH*BE-DELY-RSATN
ZR=STH*GA-DELZ-CAP*TH
FMSQ=IR**2+YR**2+ZR**2
DOM=(AL*IR+BE*YR+GA*ZR)
OE=FMSQ-2.*C*ABAR
IF(ABS(OE).LT..001*ABAR) RETURN
OE=OE/CDW
OG=(1.-COSTAN**2)/DOM
DELSTH=OE/(1.+SORT(1.-OE*OG))
***** CALCULATE DISTANCE TO INTERSECTION
STH=STH-DELSTH
***** CALCULATE ANGLE TH.
CALL THGFS
GO TO 1
END
  
```

```

SUBROUTINE REFDAT
C***** REFCAT READS REFLECTION DATA.
C*****
COMMON /69/ ABCO(20,200) ALRAY(500) BERAY(500)
1 COORDS(6,50) E(40,200) EBASE(250) ESOR(20,200)
2 GARAY(500) RED(20,200) TITLE(12) VMAT(20)
3 VOL(200) XLAM(200) XRAY(500) YRAY(500)
4 ZRAY(500)
COMMON /ARRAYS/ ALK(200) ALPHA(40) BEK(200)
1 RETA(40) DEV(200) DK(200) EK(200)
2 ELAM(200) ELAMAT(20) ENAT(20) ENERGY(200)
3 GAK(200) GAMMA(40) REF(20) SORMAT(20)
4 SUMRAY(200) SUMSOR(200) X(40) XK(200)
5 Y(40) YK(200) Z(40) ZK(200)
COMMON /IARRAY/ IFLAG(40) IPOINT(50)
1 ISMAX(50) JSOR(20) JF(200) JFL(200)
2 JL(200) JM(200) JN(200) JP(200)
3 JRAY(200) JSOR(50) KNO(900) KFLAG(200)
4 KP(200) KTYP(200) LAMP(50) LANGLE(20)
5 LREFL(200) LTYPE(20) MAT(200) NFL(20)
6 LUMRAY(200)
COMMON /PARAMS/ AL ALN ALR ALRF BE
1 GEN GEN BERF DIS DIS IN
2 GA GA GARF PER DZ EPS
3 SUMRA XI XNOV XOLD XREF XTRAN YI
4 YNOV YOLD YREF YTRAN ZI
5 ZNOV ZOLD ZREF ZTRAN
COMMON /IPARAM/ FMAX IPAX INON
1 IOPRAY IOPRINT IR IGAY JMAX JNOV JOLD
2 KINT KNAX KNOV NMAY NMAYV KSOR
INTEGER F,FMAX
DIMENSION LTABLE(10),TEMP(10)
DATA(LTABLE=6,NTABLE,6,CONST,6,DIFFUS,6,FLUCT,6,MAV,6,CAL,
6,MAV,10,4,64)
C***** PRINT PAGE HEADING.
PRINT 10
10 FORMAT(1H,53X,15HREFLECTION DATA//11H REFLECTION,6X,
4HTYPE//11H NUMBER/)
C***** READ ONE REFLECTION DATA CARD.
109 READ 11,L,LNAME,TEMP
11 FORMAT(14,1X,A6,10F6,4)
C***** REFLECTION NUMBER OF ZERO INDICATES END OF REFLECTION DATA.
IF(L)110,110,103
100 LL=L-10
C***** STORE REFLECTION DATA.
DO 103 N=1,10
103 RED(ILL,N)=TEMP(N)
C***** DETERMINE REFLECTION TYPE.
DO 104 LT=1,10
LTYPE(ILL)=LT
IF(LNAME-LTABLE(LT))104,105,104
104 CONTINUE
C***** REFLECTION TYPE NOT IN TABLE.
CALL ERPRINT(6,REFDAT)
STOP
105 IF(LT=5)106,107,106
106 IF(LT=6)111,112,111
111 PRINT 13,L,LNAME,TEMP
13 FORMAT(1H,110,4X,A6,10F10,4)
GO TO 109
107 PRINT 14,L,LNAME
14 FORMAT(1H,110,4X,A6)
GO TO 109
C***** GENERATE REFLECTION FRACTIONS AS A FUNCTION OF WAVELENGTH.
112 CALL REFIN(ILL)
PRINT 15,L,LNAME,(RED(ILL,F),F=1,FMAX)
15 FORMAT(1H,110,4X,A6/(1H,20X,10F10,4))
GO TO 109
110 RETURN
END

```

```

SUBROUTINE REFAP(ILL)
C*****
C***** REFAP CALLS THE DESIRED ROUTINE FOR GENERATING
C***** REFLECTION FRACTIONS AS A FUNCTION OF WAVELENGTH.
C*****
COMMON /69/ ABCO(20,200) ALRAY(500) BEAT(500)
1 COORCS(6,50) E(40,200) EBASE(200) ESCR(20,200)
2 GARAY(500) RED(20,200) TITLE(12) VPAT(20)
3 VOL(200) XLAM(200) XRAY(500) YPAT(500)
4 ZPAT(500)
COMMON /ARRAYS/ ALK(200) ALPHA(40) BEK(200)
1 META(40) DEV(200) DK(200) EK(200)
2 FLAM(200) ELAMAT(20) EMAT(20) ENRGY(200)
3 GAK(200) GAMMA(40) REF(20) SCMAT(20)
4 SUMRAY(200) SUMSCM(200) X(40) XK(200)
5 Y(40) YK(200) Z(40) ZK(200)
COMMON /IARRAY/ IFLAG(40) IPOINT(50)
1 ISMAX(50) ISOR(20) JF(200) JFL(200)
2 JL(200) JN(200) JN(200) JN(200)
3 JRAY(200) JSCA(50) KANG(900) KFLAG(200)
4 KIP(200) KTOP(200) LAMP(50) LAAGLE(20)
5 LREFL(200) LTYPE(20) MAT(200) MFL(20)
6 MMAT(20) NUPRAY(200)
COMMON /PARAMS/ AL ALN ALR ALRF RE
1 BEA BEB BERF DIS DMIN DZ EPS
2 GA GAN GAR GARF PER
3 SUMRA XI XACH XCLD XREF XTRAN YI
4 TNGD TCLO TREF TTRAN ZI
5 ZNGD ZCLO ZREF ZTRAN
COMMON /IPARMS/ FMAX IMAX IACH
1 IOPRAY ICPRNT IN IRAY JMAX JACH
2 KIAT KPAI KAOI PMAX PNOI ASOR JOLD
INTEGER F,FMAX
DIMENSION TSTG(200)
GO TO(100,200,300,400,500,600,700,800,900),ILL
100 CALL REFL11(XLAM,FMAX,TSTG)
GO TO 110
200 CALL REFL12(XLAM,FMAX,TSTG)
GO TO 110
300 CALL REFL13(XLAM,FMAX,TSTG)
GO TO 110
400 CALL REFL14(XLAM,FMAX,TSTG)
GO TO 110
500 CALL REFL15(XLAM,FMAX,TSTG)
GO TO 110
600 CALL REFL16(XLAM,FMAX,TSTG)
GO TO 110
700 CALL REFL17(XLAM,FMAX,TSTG)
GO TO 110
800 CALL REFL18(XLAM,FMAX,TSTG)
GO TO 110
900 CALL REFL19(XLAM,FMAX,TSTG)
GO TO 110
110 DO 101 F=1,FMAX
REF(ILL,F)=TSTG(F)
101 CONTINUE
RETURN
END

```

```

SUBROUTINE REFLECT
C*****
C***** REFLECT CALCULATES THE DIRECTIONAL COSINES OF THE
C***** REFLECTED RAY.
C*****
COMMON /59/ ABCD(20,200) ,ALRAY(500) ,DERAY(500)
1 ,COCOS(4,50) ,E(40,200) ,EBASE(200) ,ESOR(20,200)
2 ,FARAY(500) ,RED(20,200) ,TITLE(12) ,VMAT(20)
3 ,VOL(200) ,ULAN(200) ,XRAY(500) ,YRAY(500)
4 ,ZRAY(500)
COMMON /ARRAYS/ ALK(200) ,ALPHA(40) ,BEK(200)
1 ,BETA(40) ,DEV(200) ,DK(200) ,EK(200)
2 ,ELAN(200) ,ELAMAT(20) ,ENAT(20) ,ENERGY(200)
3 ,GAK(200) ,GAMMA(40) ,REF(20) ,SORMAT(20)
4 ,SUMRAY(200) ,SUMSOR(200) ,X(40) ,XK(200)
5 ,Y(40) ,YK(200) ,Z(40) ,ZK(200)
6 ,ZLAN(20)
COMMON /IARRAY/ IFLAG(40) ,IPOINT(50)
1 ,ISMAX(50) ,ISOR(20) ,JF(200) ,JFL(200)
2 ,JL(200) ,JM(200) ,JP(200)
3 ,JRAY(200) ,JSOR(50) ,KAND(900) ,KFLAG(200)
4 ,KP(200) ,KTP(200) ,LAMP(50) ,LANGLE(20)
5 ,LREFL(200) ,LTYPE(20) ,MAT(200) ,MFL(20)
6 ,MAT(20) ,MPRAY(200)
COMMON /PARAMS/ AL ,ALN ,ALR ,ALRF ,BE ,
1 ,BER ,BERF ,DIS ,DMIN ,DZ ,EPS
2 ,GA ,GAN ,GAR ,GANF ,PER ,
3 ,SUMRA ,XI ,XADN ,XCLO ,XREF ,XTRAN ,YI
4 ,YMD ,YOLD ,YREF ,YTRAN ,ZI
5 ,ZMD ,ZOLD ,ZREF ,ZTRAN ,INON ,
COMMON /IPRIM/ IOPRAY ,IOPENT ,IR ,IRAY ,JMAX ,JNON ,JOLD
1 ,INT ,KMAX ,KNOW ,KMAX ,KNON ,KSOR
INTEGER F,FMAX
C***** CALCULATE COEFFICIENT NEEDED.
BTMP=-2.0*(AL*ALN+BE*BER+GA*GAN)
C***** CALCULATE COSINES FOR REFLECTED RAY.
ALR=AL-BTMP*ALN
BER=BE-BTMP*BER
GAN=GA-BTMP*GAN
C***** CHECK DIRECTIONAL COSINES.
CALL CHECK(REFLECT,ALR,BER,GAN)
CALL DEBUG(REFLECT)
RETURN
END

```

```

***** SUBROUTINE REFRACTION *****
***** REFRACTION CALCULATE THE DIRECTIONAL COSINES OF THE *****
***** REFRACTED (TRANSMITTED) RAY. *****
*****
COMMON /69/ ABCD(20,200) *ALRAY(500) *BERAY(500)
1 *COINS(6,50) *E(40,200) *EBASE(200) *ESCA(20,200)
2 *CARAY(500) *RED(20,200) *TITLE(12) *VMAT(20)
3 *VCL(200) *XLAM(200) *XRAT(500) *YRAT(500)
4 *ZRAY(500)
COMMON /ARRAYS/ ALK(200) *ALPHA(40) *BEK(200)
1 *RETA(40) *DEV(200) *DK(200) *EK(200)
2 *ELAM(200) *ELAMAT(20) *EMAT(20) *ENERGY(200)
3 *EAK(200) *GAMMA(40) *REF(20) *SCHMAT(20)
4 *SUMRAY(200) *SUMSQ(200) *X(40) *XK(200)
5 *Y(40) *YK(200) *Z(40) *ZK(200)
6 *ZLAP(20)
COMMON /TARRAY/ IFLAG(40) *IPOINT(50)
*ISMAX(50) *ISOR(20) *JF(200) *JFL(200)
*JL(200) *JM(200) *JM(200) *JP(200)
*JRAY(200) *JSOR(50) *KBD(900) *KFLAG(200)
*KP(200) *KTP(200) *LAMP(50) *LANGE(20)
*LRFL(200) *LTYPE(20) *MAT(200) *MFL(20)
*ONAT(20) *MURAY(200)
COMMON /PARAMS/ AL *ALN *ALR *ALRF *RE
*GA *GER *GAR *DIS *DMIN *DZ *EPS
*SUVA *XI *XNO *XCLO *PER * *
*YNO *YOLD *YREF *YTRAN *ZI *
*ZNO *ZOLD *ZREF *ZTRAN *
COMMON /IPARAM/ FMAX *IMAX *JMAX *JNO *JOLD
*IDORAY *IDORAYT *IR *IRAY *JNO *JNO *JNO
*KNAT *KNAT *KNOW *KNOW *KNOW *KNOW
INTEGER F,FMAX
REAL LCOIN
I=1400
***** SET INDICES FOR MATERIALS ON EACH SIDE OF BOUNDARY.
N1=AT(JOLD)
N2=AT(JNO)
***** FIND RATIO OF REFRACTIVE INDICES.
ATMP=REF(N1)/REF(N2)
***** CALCULATE COEFFICIENTS NEEDED.
102 LCOIN=AL*ALN*BE*BE*GA*GAN
***** IF COSINE OF ANGLE OF INCIDENCE LESS THAN 1.E-6 GO TO 101.
IF(ABS(LDOTN)-1.E-6)101,101,103
***** SET DIRECTIONAL COSINES FOR TRANSMITTED RAY.
101 ALRF=AL
BEFF=BE
GAEF=GA
PER=1.0
GO TO 107
103 BTEP=1.0*(1.0-ATMP**2)/(ATMP**2*(LDOIN**2))
IF(ATEM)104,105,105
104 PER=0.0
***** IF REFLECTION NUMBER GREATER THAN 10. GO TO 109.
IF(LREFL(KNO)-10)107,107,111
111 ALRF=AL
BEFF=BE
GAEF=GA
GO TO 109
***** FRESNEL REFLECTION.

```



```

105 BTMP=ATMP*LODA*(1.0-SCNT(8TERM))
C***** CALCULATE COSINES OF REFRACTED RAY.
ALRF=ATMP*AL*BTMP*ALM
BERF=ATMP*BE*BTMP*BEN
GARF=ATMP*GA*BTMP*GAN
CALL CHECK(6HREFRAC,ALRF,PERF,GARF)
106 IF(IREFL(KNOW)-10)108,108,109
C***** FIND PERCENT OF RAY TO BE TRANSMITTED.
108 CALL PERCT
GO TO 107
C***** GET REFLECTION TYPE(LT).
109 L=LREFL(KNOW)
LT=LTYPE(L)
LL=L-10
C***** BRANCH ON REFLECTION TYPE.
GO TO(100,200,300,400,500,600),LT
C***** TABLE LOOKUP REFLECTION.
100 CALL TARREF(LL)
GO TO 107
C***** CONSTANT REFLECTION.
200 CALL CONREF(LL)
GO TO 107
C***** DIFFUSE REFLECTION.
300 CALL DIFREF(LL)
GO TO 107
C***** FUNCTION REFLECTION( PERCENT=1.0-A*(1.0+CCS(TM)))
400 CALL FUNREF(LL)
GO TO 107
500 CALL NAVREF(LL)
600 DO 113 F=1,FMAX
E1(MAX,F)=EED(LL,F)*E(I,F)
E(I,F)=E(I,F)-E1(MAX,F)
113 CONTINUE
GO TO 110
107 DO 112 F=1,FMAX
E1(MAX,F)=(1-PER)*E(I,F)
E(I,F)=PER*E(I,F)
112 CONTINUE
C***** STORE COSINES OF TRANSMITTED RAY.
110 ALP=A(I)=ALRF
BET(I)=BERF
GAN=A(I)=GARF
CALL DEBUG(6HREFRAC)
RETURN
END

```

SUBROUTINE KEYS

***** PART OF MELIX INTERSECTION CALCULATION *****

```

COMMON /69/ ABCO(20,200) ALRAY(500) DERAY(500)
1 COORDS(6,50) F(40,200) EBASE(200) ESON(20,200)
2 GARAY(500) RED(20,200) TITLE(12) YMAT(20)
3 VOL(200) XLAM(200) XRAY(500) YRAY(500)
4 ZRAY(500)

COMMON /ARRAYS/ ALK(200) ALPHA(40) BEK(200)
1 META(40) DEV(200) DS(200) EK(200)
2 ELAM(200) ELAMAT(20) EMAT(20) ENERGY(200)
3 GAK(200) GAMMA(40) REF(20) SORMAT(20)
4 SUMRAY(200) SUMSOR(200) X(40) XK(200)
5 Y(40) YK(200) Z(40) ZK(200)
6 ZLAM(20)

COMMON /TARRAY/ IFLAG(40) IPCINT(50)
1 IS-LX(50) ISOR(20) JF(200) JFL(200)
2 JL(200) JM(200) JM(200) JP(200)
3 JRAY(200) JSOR(50) KNO(900) KFLAG(200)
4 KP(200) KTYP(200) LAMP(50) LANGLE(20)
5 LREFL(200) LTYPE(20) MAT(200) MFL(20)
6 MAT(20) MURAY(200)

COMMON /PARAMS/ AL ALN ALR ALRF RE
1 BEN BER BREF DIS DNIN DZ EPS
2 GSA GAN GAN GANF PER
3 SUMZA XI XNO XCLO XREF XTRAN YI
4 YCLO YOLD YREF YTRAN ZI
5 ZCLO ZOLD ZREF ZTRAN
COMMON /IFLAGS/ FMAX IMAX INO
1 IORRAY IORINT IR IRAY JMAX JON
2 INT KMAX KNO JMAX JNO JORD
INTEGER F,FMAX

COMMON /MELCON/ A,BAR,B,BI,BIP,BP,CAP,DELTH,DELX,DELY,DELZ,DWIT,
* PSTZ,REV,R,Q,STN,TH,TH1,SOXAS,RCS,CSTN,SNTH,NCSTN,RSATN,BSTN,
* COSTAN,THZ,DTHZ
IC=0
IF((B-ABAR)*(B1-ABAR))100,100,101
100 CALL FTNR
GO TO 110
101 IF((B-ABAR)*BP*DELTH)110,104,104
104 BS=B
TH5=TH
BS=BP
TH6=TH
TH4=TH1
B4=B1
B4P=BIP
12 CONTINUE
BTN=(TH-TH4)/(BP-B4P)*BP
TH4=TH
B4=B1
B4P=BP
TH=TH-BTN
IF((TH-TH1)*(TH-TH6))13,10,10
10 CONTINUE
TH=.5*(TH1+TH6)
13 CONTINUE
CALL BCALC
B=(B-ABAR)/(B4-ABAR)
IF(B)106,106,10

```

```

19 CONTINUE
  IF(P.LT. 0.5) GO TO 14
  IF(R00R4P) 14.20.20
20 CONTINUE
  IF(ABS(TM1-TM6).ST..01) GO TO 14
132 CONTINUE
  TM=TM5
  B=BS
  BP=BP
  RETURN
14 CONTINUE
  IF(R00R1P.LT.0) GO TO 15
  TM1=TM
  B1=B
  B1P=BP
  GO TO 16
15 CONTINUE
  TM4=TM
16 CONTINUE
  IC=IC+1
  IF(IC.LT.50) GO TO 12
  PRINT 17, TM,B,RP,TM1,B1,B1P,TM4,B4,B4P,TM5,B5,B5P,TM6
17 FORMAT(9H REVS 17.7( 12E10.3))
  GO TO 132
106 CALL FINDER
  TM1=TM
  B1=B
  B1P=BP
  TM=TM5
  B=BS
  BP=BP
  CALL FINDER
110 CONTINUE
  RETURN
  END

```

```

SUBROUTINE ROULET(1)
C.....
C..... ROULET PLAYS RUSSIAN ROULET WITH RAY NUMBER 1.
C.....
COMMON /69/ ABCO(20,200) ALRAY(500) BERAY(500)
1 COORDS(16,50) E(40,200) EBASE(200) ESQR(20,200)
2 GARAY(500) RED(20,200) TITLE(12) VMAT(20)
3 VOL(200) XLAM(200) XRAY(500) YRAY(500)
4 ZRAY(500)
COMMON /ARRAYS/ ALK(200) ALPHA(40) BEK(200)
1 BETA(40) DEV(200) DK(200) EK(200)
2 ELAM(200) ELAMAT(20) EMAT(20) ENRGY(200)
3 ERK(200) GAMMA(40) REF(20) SQRMAT(20)
4 SIMRAY(200) SUMSQ(200) X(40) XK(200)
5 Y(40) YK(200) Z(40) ZK(200)
6 ZLAM(20)
COMMON /IARRAY/ IFLAG(40) IPCINT(50)
1 ISMAX(50) ISQR(20) JF(200) JFL(200)
2 JL(200) JM(200) JN(200) JPN(200)
3 JRAY(200) JSQR(50) KEND(900) KFLAG(200)
4 KP(200) KTYP(200) LAMP(50) LANGLE(20)
5 LREFL(200) LTYPE(20) MAT(200) MFL(20)
6 MAT(20) NMURAY(200)
COMMON /PARAMS/ AL ALN ALR ALRF RE
1 BEA BER BERF DIS DMIN DZ EPS
2 GA GAN GAR GANF PER XTRAN YI
3 SUMRA XI XND XCLD XREF ZI
4 YND YOLD YREF YTRAN ZTRAN
5 ZND ZOLD ZREF ZTRAN
COMMON /IPARAM/ FMAX IMAX INOW JMAX JNOW JOLD
1 IOPRAY IOPRNT IN IRAY JNSQR
2 KINY KMAX KND MMAX MNOW NSQR
INTEGER F,FMAX
COMMON /F57/ FRAC IDEBUG
100 DO 101 F=1,FMAX
101 IF(F(1,F)-(1.E-6)*EBASE(F)) 101,101,102
102 IF(E(1,F)-EBASE(F)) 103,101,101
101 CONTINUE
GO TO 110
103 RAND(0)
IF(RAND,5) 105,104,104
104 DO 106 F=1,FMAX
106 IF(E(1,F)-EBASE(F)) 107,106,106
107 E(1,F)=2.0*E(1,F)
106 CONTINUE
GO TO 100
105 DO 108 F=1,FMAX
108 IF(E(1,F)-EBASE(F)) 109,108,108
109 E(1,F)=0.0
108 CONTINUE
110 RETURN
END

```

```

SUBROUTINE SCALE(M,YLAM,F,SCA)
C.....
C..... SCALE CHOOSES A SCALE FACTOR FOR SCALING ENERGY DEPOSITED
C..... FOR A GIVEN WAVELENGTH.
C.....
COMMON /69/ ABCO(20,200) *ALRAY(500) *BERAY(500)
1 *CORRS(6,50) *E(40,200) *EBASE(200) *ESOR(20,200)
2 *GARAY(500) *RED(20,200) *TITLE(12) *VMAT(20)
3 *VPL(200) *ZLAM(200) *XRAY(500) *YRAY(500)
4 *ZRAY(500)
COMMON /ARRAYS/ ALK(200) *ALPHA(40) *BEK(200)
1 *BETA(40) *DEV(200) *DK(200) *EK(200)
2 *FLAM(200) *ELAMAT(20) *EMAT(20) *ENERGY(200)
3 *GAK(200) *GAMMA(40) *REF(20) *SCMAT(20)
4 *SMRAY(200) *SUNSOR(200) *XI(40) *XK(200)
5 *YI(40) *YK(200) *Z(40) *ZK(200)
6 *ZLAM(20)
COMMON /TARRAY/
*ISMAX(50) *ISOR(20) *IFLAG(40) *IPOINT(50)
*JL(200) *JM(200) *JF(200) *JFL(200)
*JRAY(200) *JSOR(50) *JA(200) *JP(200)
*KP(200) *KYP(200) *KRD(900) *KFLAG(200)
*LAFFL(200) *LTYPE(20) *LAMP(50) *LANGLE(26)
*MAT(20) *MURRAY(200) *MAT(200) *MFL(20)
COMMON /PARAMS/
*BEH *BER *BERF *CIS *ALR *ALRF *DE
*GA *GAN *GAR *GASF *DMIN *DZ *EPS
*SMRA *XI *XND *XOLD *PER *XREF *XTRAN *YI
*YND *YOLD *YREF *YTHAN *ZI
*ZND *ZOLD *ZREF *ZTHAN
COMMON /PARAM/
*IDPRAY *IDPRNT *IR *IMAX *INOW
*KNAT *KNAX *KNOW *IRAY *JMAX *JND
*JOLD *JND *JOLD
INTEGER F,FMAX
DIMENSION YLAM(1)
C..... IF LASING WAVELENGTH IS ZERO, SCALE FACTOR IS ZERO.
IF(ZLAM(M))20,100,101
100 SCA=0.0
GO TO 102
C..... CALCULATE SCALE FACTOR.
101 SCA=YLAM(F)/ZLAM(M)
102 RETURN
END

```

```

SUBROUTINE SEGQAT
C.....
C..... SEGQAT READS DATA AND SETS UP SEGMENTS.
C.....
COMMON /69/      ARCO(20,200)  *ALRAY(500)  *BERAY(500)
1 *CNGPTS(6,50)  *E(40,200)   *EBASE(200)  *ESCR(20,200)
2 *CARAY(500)    *RED(20,200)  *TITLE(12)  *VMAT(20)
3 *VOL(200)      *XLAN(200)    *XRAY(500)   *YHAY(500)
4 *ZRAY(500)
COMMON /ARRAYS/  ALK(200)      *ALPHA(40)    *BEK(200)
1 *ETA(40)       *DEV(200)     *DK(200)    *EK(200)
2 *ELAM(200)     *ELAMAT(20)   *EMAT(20)   *ENRGY(200)
3 *FAK(200)      *GAMMA(40)    *REF(20)    *SGRMAT(20)
4 *SUMRAY(200)   *SUMSCR(200)  *X(40)     *XK(200)
5 *Y(40)         *YK(200)     *Z(40)     *ZK(200)
6 *ZLAN(20)
COMMON /IARRAY/  IFLAG(40)    *IPOINT(50)
*ISMAY(50)      *ISCR(20)     *JF(200)    *JFL(200)
*JL(200)        *JM(200)     *JN(200)    *JP(200)
*JRAY(200)      *JSOR(50)    *KEND(900)  *KFLAG(200)
*KP(200)        *KTP(200)    *LAMP(50)   *LANGLE(20)
*LREFL(200)     *LTYPE(20)   *MAT(200)   *MFL(20)
*MMAT(20)       *MURRAY(200)
COMMON /PARAMS/  AL          *ALN          *ALN          *ALRF          *RE
*BER            *BERF        *DIS          *DIS          *DZ          *FPS
*GA            *GAN          *GAR          *GARF        *PER          *
*SUMQA         *XI           *XNOW        *XOLD        *XREF        *XTRAN        *YI
*YAC           *YOLD        *YREF        *YTRAN        *ZI          *
*ZAC           *ZOLD        *ZREF        *ZTRAN        *
COMMON /IPARAM/  FMAX        *IMAX          *INOW          *
*ICPRAY        *ICPRNT      *IR           *IRAY          *JMAX          *JNOW          *JOLD
*KIAT          *KMAX        *KNOW        *MMAX          *MACH          *MSOR
INTEGER F,FMAX
DIMENSION KINPUT(20)
PRINT 13
NLP=5
13 FC=VAT(1M,53X,12HSEGMENT DATA//8H SEGMENT,2X,8H MATERIAL,
*10X,20H CONFINING BOUNDARIES,53X,6H VOLUME/8H NUMBER,
*4X,4H NUMBER/)
ND=1
JMAT=0
C..... READ DATA FOR SEGMENT J.
104 READ 10,J,MAT1,(KINPUT(N),N=1,11),VOLI
10 FORMAT(214,1115,F0.6)
C..... ZERO SEGMENT NUMBER INDICATES END OF DATA.
IF(J)106,110,103
C..... COUNT NUMBER OF BOUNDARIES FOR SEGMENT J.
103 DO 105 N=1,11
IF(KINPUT(N))105,106,105
105 N=N+1
C..... PRINT SEGMENT DATA.
106 PRINT 12,J,MAT1,(KINPUT(N),N=1,11),VOLI
12 FORMAT(1M,17,110,5X,1110,6X,F12.6)
C..... IF J GREATER THAN JMAX, JMAX=J,
IF(J-JMAX)101,101,100
100 JMAX=J
C..... SET UP SEGMENT J.
101 KP(J)=NP+1
MAT(J)=MAT1
VOL(J)=VOLI
KEND(KP)=0
DO 102 N=1,NB
NP=NP+N
KEND(NP)=KINPUT(N)
102 CONTINUE
NP=NP+NB+1
NLP=NLP+1
IF(NLP-52)107,107,100
100 NLP=5
PRINT 13
107 CONTINUE
GO TO 104
110 CONTINUE
KEND(NP)=0
RETURN
END

```

```

SUBROUTINE SEGMENT
C.....
C..... SEGMENT FINDS WHICH SEGMENT CONTAINS POINT XI,YI,ZI.
C.....
COMMON /69/      AHC0(20,200)  ALRAY(500)  BEHAY(500)
1  C0C0RS(6,50)  E(40,200)  E9&SE(200)  ESCM(20,200)
2  RRAY(500)  REU(20,200)  TITLE(12)  VPAT(20)
3  VOL(200)  XLAM(200)  XHAY(500)  YRAY(500)
4  ZRAY(500)
COMMON /ARRAYS/  ALK(200)  ALP-A(40)  BEK(200)
1  PFTA(40)  DEV(200)  DK(200)  EK(200)
2  FLAM(200)  ELAMAT(20)  EMAT(20)  ENERGY(200)
3  GAK(200)  GAMMA(40)  REF(20)  SCRPAT(20)
4  SUMRAY(200)  SUMSOR(200)  X(40)  XK(200)
5  Y(40)  YK(200)  Z(40)  ZK(200)
6  ZLAM(20)
COMMON /IARRAY/  IFLAG(40)  IPOINT(50)
1  ISMAX(50)  ISOR(20)  JF(200)  JFL(200)
2  JL(200)  JM(200)  JN(200)  JP(200)
3  JRAY(200)  JSOR(50)  KBND(900)  KFLAG(200)
4  KP(200)  KTYP(200)  LAMP(50)  LANGLE(20)
5  LRFFL(200)  LTYPE(20)  MAT(200)  MFL(20)
6  MNAT(20)  NUMRAY(200)
COMMON /PARAMS/  AL  ALN  ALR  ALRF  RE
1  REA  BER  BERF  OIS  DMIN  DZ  EPS
2  GA  GAM  GAR  GARF  PER
3  SUMRA  XI  ANOW  XCLO  XREF  XTRAN  YI
4  YNOU  YOLD  YREF  YTRAN  ZI
5  ZNOU  ZCLO  ZREF  ZTRAN
COMMON /IPARAM/  FMAX  IMAX  INOW
1  IDPRAY  IDPRNT  IR  IRAY  JMAX  JNOW  JOLD
2  KINT  KMAX  KNOU  MMAX  MNOU  NSOR
INTEGER F,FMAX
JOLD=JNOW
C..... IF JNO= IS MOTHER SEGMENT NUMBER I. SET J=JNO.
IF(JNO=1)101,100,101
100 J=JNO
GO TO 102
C..... SET J EQUAL TO MOTHER OF JNO.
101 J=JM(JNO)
C..... IF POINT IS IN MOTHER SEGMENT NUMBER J, GO TO 104.
102 IF(INSEG(J))103,103,104
103 IF(JM(J))105,105,112
112 J=JM(J)
GO TO 102
105 JNO=0
GO TO 111
C..... POINT IS IN MOTHER SEGMENT J.
104 JNO=J
C..... LOOK AT LAST DAUGHTER IN HIERARCHY.
106 J=JL(J)
C..... IF POINT IS IN SEGMENT J, GO TO 110.
104 IF(INSEG(J))109,109,110
C..... LOOK AT NEXT LOWER SEGMENT IN HIERARCHY.
109 IF(JP(J))111,111,113
113 J=JP(J)
GO TO 108
C..... POINT IS IN SEGMENT J.
110 JNO=J
C..... IF J IS A MOTHER, GO TO 106 AND LOOK AT DAUGHTERS.
IF(JF(J))111,111,106
111 CONTINUE
RETURN
END

```

```

SUBROUTINE SETCN(K)
C*****
C***** SET UP 1 CONE.
C*****
COMMON /69/ ABCD(21,200) ALRAY(500) EERRAY(500)
1 .CNCAT(516,58) .E(149,200) .ESASE(200) .ESCH(20,200)
2 .GIRAY(500) .REG(20,200) .TITLE(12) .MAT(100)
3 .VOL(200) .XLAN(200) .XWAY(500) .YWAY(500)
4 .ZRAY(500)
COMMON /ARRAYS/ ALX(200) ALP-X(40) .SEK(200)
1 .REF1A(40) .DEV(200) .OK(200) .EN(200)
2 .ELAN(200) .ELANAT(20) .EMAT(20) .ENERGY(200)
3 .GAK(200) .GAMMA(40) .REF(20) .SCMAT(20)
4 .SHRAY(200) .SUMSQ-(200) .X(40) .XK(200)
5 .Y(40) .YK(200) .Z(40) .ZK(200)
6 .ZLAN(200)
COMMON /IARRAY/ IFLAG(40) .IPGINT(50)
1 .IS-IX(50) .JSC-(20) .JP(200) .JFL(200)
2 .JL(200) .JM(200) .JN(200) .JP(200)
3 .JRAY(200) .JSOR(50) .KSC(100) .KFLAG(200)
4 .KPD(200) .KTP(200) .LAMP(50) .LANGLE(20)
5 .LREFL(200) .LTYPE(20) .MAT(200) .PFL(20)
6 .PMAT(20) .SUMRAY(200)
COMMON /PARAMS/ AL .ALN .ALR .ALRF .RE
1 .BEA .PEA .BERF .CIS .DMIN .DZ .EPS
2 .GA .GAN .GAR .GANF .PER
3 .SUMPA .XI .XNOV .XOLD .XREF .XTRAN .VI
4 .YNO .YOLD .YREF .YTRAN .ZI
5 .ZNO .ZOLD .ZREF .ZTRAN
COMMON /IPARAM/ FMAX .IPAX .INOV
1 .INDRAY .IOCENT .IR .IRAY .JMAX .JNOV
2 .KINT .KPAI .KNOV .KMAX .KNOV .NSOR .JOLD
INTEGER F,FMAX
COMMON /COMMON/ANOPAR(12)
C***** STORE INPUT DATA.
XK(K)=ANOPAR(1)
YK(K)=ANOPAR(2)
ZK(K)=ANOPAR(3)
EK(K)=ANOPAR(6)-ANOPAR(3)
DELZ=EK(K)
DELY=ANOPAR(4)-XK(K)
DELY=ANOPAR(5)-YK(K)
C***** CALCULATE COS**2 OF ANGLE FROM CONE AXIS TO CONE SURFACE.
DK(K)=DELZ**2/(DELY**2+DELZ**2)
C***** PRINT CONE DATA.
PRINT 11,K,LREFL(K),(ANOPAR(N),N=1,6)
11 FORMAT(1X,15,16,4X,6MCONE,24X,6F5.3)
RETURN
END

```



```

SURF TIME SETCON(K)
C*****
C***** SETCON SETS UP A CONICAL BOUNDARY FROM THE INPUT DATA FOR
C***** T-TY BOUNDARY.
C*****
COMMON /69/ ABCD(20,200) ,ALRAY(500) ,BERAY(500)
1 ,CDTOS(6,50) ,E(40,200) ,ERASE(200) ,ESCH(20,200)
2 ,GARY(500) ,RED(20,200) ,TITLE(12) ,VMAT(20)
3 ,VOL(200) ,XLAN(200) ,XRAY(500) ,YRAY(500)
4 ,ZRAY(500)
COMMON /7ARRAY/ ALK(200) ,ALPHA(40) ,BEK(200)
1 ,BETA(40) ,DEV(200) ,DK(200) ,EK(200)
2 ,FLAM(200) ,ELAMAT(20) ,EMAT(20) ,ENERGY(200)
3 ,GAK(200) ,GAPMA(40) ,REF(20) ,SCRMAT(20)
4 ,SUMRAY(200) ,SUMSOR(200) ,X(40) ,XK(200)
5 ,Y(40) ,YK(200) ,Z(40) ,ZK(200)
6 ,ZLAN(20)
COMMON /IARRAY/ IFLAG(40) ,IPOINT(50)
1 ,ISMAX(50) ,ISDR(20) ,JF(200) ,JFL(200)
2 ,JL(200) ,JM(200) ,JN(200) ,JP(200)
3 ,JRAY(200) ,JSOR(50) ,KEND(986) ,KFLAG(200)
4 ,KP(200) ,KTP(200) ,LAMP(50) ,LANGLE(20)
5 ,LEFL(200) ,LTYPE(20) ,MAT(200) ,MFL(20)
6 ,MAT(20)
COMMON /PAR15/ AL ,ALN ,ALR ,ALRF ,BE ,
1 ,BER ,BERF ,DIS ,DMIN ,DZ ,EPS ,
2 ,GA ,GAN ,GAR ,GASF ,PER ,
3 ,SMDA ,XI ,XND ,XCLO ,XNEF ,XTRAN ,YI ,
4 ,YND ,YCLD ,YREF ,YTRAN ,ZI ,
5 ,ZND ,ZCLD ,ZREF ,ZTRAN
COMMON /PAR16/ FMAX ,IPAX ,INOW ,
1 ,IPRAY ,IDPAX ,IN ,IRAY ,JMAX ,JNOW ,JOLD ,
2 ,KINT ,KMAX ,KNOW ,PMAX ,PNOW ,NSOR
INTEGER F,FMAX
COMMON /RNDCON/BNOPAR(12)
C***** STORE BOUNDARY DATA.
ALK(K)=BNOPAR(1)
BEK(K)=BNOPAR(2)
GAK(K)=9.0
CALL C-ECK(4=SETCON,ALK(K),BEK(K),GAK(K))
XK(K)=BNOPAR(3)
YK(K)=BNOPAR(4)
ZK(K)=9.0
C***** CALCULATE ECCENTRICITY AND DISTANCE TO DIRECTRIX.
DX=BNOPAR(5)-XK(K)
DY=BNOPAR(6)-YK(K)
R1=SQRT(DX**2+DY**2)
AL1=DX/R1
BE1=DY/R1
CALL C-ECK(5=SETCON,AL1,BE1,0.0)
TM=ALK(K)*AL1-BEK(K)*BE1
DX2=BNOPAR(7)-XK(K)
DY2=BNOPAR(8)-YK(K)
R2=SQRT(DX2**2+DY2**2)
IF(ABS(R1-R2)-).E-6)GOTO 101,102
102 AL2=DX2/R2
BE2=DY2/R2
CALL C-ECK(6=SETCON,AL2,BE2,0.0)
TM2=ALK(K)*AL2-BEK(K)*BE2
EK(K)=(R1-R2)/(R1+TM1-R2*TM2)
DK(K)=(R1+R2*(TM1-TM2))/(R1-R2)
C***** PRINT ARRAY VALUES FOR BOUNDARY.
PRINT 10,K,LEFL(K),ALK(K),BEK(K),XK(K),YK(K),
1 ,BNOPAR(5),BNOPAR(6),BNOPAR(7),BNOPAR(8),DK(K),EK(K)
10 FORMAT(1H ,15,10,3X,5-CONIC,2F8.3,8X,2F8.3,8X,
1 ,6F5.3)
NEXTN
10) PRINT 12,K
12 FORMAT(26H INPUT FOR BOUNDARY NUMBER,16,16H CONTAINS ERRORS)
STOP
END

```

```

SUBROUTINE SETCYL(K)
C*****
C***** SETCYL SETS UP A CYLINDRICAL BOUNDARY.
C*****
COMMON /69/ ABC0(20,200) ,ALRA(1500) ,BERAY(500)
1 ,COORDS(6,50) ,E(40,200) ,EBASE(200) ,ESOR(20,200)
2 ,GARAY(500) ,RED(20,200) ,TITLE(12) ,VMAT(20)
3 ,VPL(200) ,XLAM(200) ,XRAY(500) ,YRAY(500)
4 ,ZRAY(500)
COMMON /ARRAYS/ ALK(200) ,ALPHA(40) ,BEK(200)
1 ,RFTA(40) ,DEV(200) ,DK(200) ,EK(200)
2 ,ELAM(200) ,ELAMAT(20) ,EMAT(20) ,ENERGT(200)
3 ,GAK(200) ,GAMMA(40) ,REF(20) ,SGRMAT(20)
4 ,SUMRAY(200) ,SUMSOR(200) ,Y(40) ,XK(200)
5 ,Y(40) ,YK(200) ,Z(40) ,ZK(200)
6 ,ZLAM(20)
COMMON /!ARRAY/ IFLAG(40) ,IPOINT(50)
,ISMAX(50) ,ISOR(20) ,JF(200) ,JFL(200)
,JL(200) ,JM(200) ,JN(200) ,JP(200)
,JRAY(200) ,JSOR(50) ,KEND(900) ,KFLAG(200)
,KP(200) ,KTYP(200) ,LAMP(50) ,LANGE(20)
,LREFL(200) ,LTYPE(20) ,MAT(200) ,MFL(20)
,MAT(20) ,MURAY(200)
COMMON /PARAMS/ AL ,ALN ,ALR ,ALRF ,RE
,BEA ,BER ,BERF ,DIS ,DMIN ,DZ ,EPS
,GA ,GAN ,GAR ,GANF ,PER ,
,SUMRA ,XI ,XND ,XCLO ,XREF ,XTRAN ,VI
,YND ,YOLD ,YNEF ,YTHAN ,ZI ,
,ZND ,ZOLD ,ZREF ,ZTHAN
COMMON /IPARAM/ FMAX ,IMAX ,INOW
,IOPRAY ,IOPRNT ,IR ,IRAT ,JMAX ,JNOW
,KINT ,KMAX ,KNOW ,MMAX ,MNOW ,MSOR ,JOLD
INTEGER F,FMAX
COMMON /RNDCCM/RNDOPAR(12)
C***** STORE BOUNDARY DATA FOR CYLINDER
ALK(K)=RNDOPAR(1)
BEK(K)=RNDOPAR(2)
GAK(K)=RNDOPAR(3)
C***** CHECK DIRECTIONAL COSINES.
CALL CHECK(6,SETCYL,ALK(K),BEK(K),GAK(K))
XK(K)=RNDOPAR(4)
YK(K)=RNDOPAR(5)
ZK(K)=RNDOPAR(6)
DK(K)=RNDOPAR(7)
C***** PRINT BOUNDARY DATA.
PRINT 10,K,LREFL(K),ALK(K),BEK(K),GAK(K),XK(K),YK(K),
,ZK(K),DK(K)
10 FORMAT(1H ,15,16,2X,6HCYLIND,6F8.3,32X,2F8.3)
RETURN
END

```

```

SUBROUTINE SETELL(K)
C.....
C..... SETELL SETS UP ARRAYS FOR AN ELLIPSOID.
C.....
COMMON /69/ ABCD(20,200) *ALRAY(500) *BERAY(500)
1 *CDT(75,56) *E(40,200) *EBASE(200) *ESCR(20,200)
2 *CRAY(500) *RED(20,200) *TITLE(12) *VMAT(20)
3 *VOL(200) *XLAN(200) *XRAY(500) *YHAY(500)
4 *ZRAY(500)
COMMON /ARRAYS/ ALK(200) *ALPHA(40) *BEK(200)
1 *RETA(40) *DEV(200) *DK(200) *EK(200)
2 *FLA(200) *ELAMAT(20) *EMAT(20) *ENERGY(200)
3 *GAK(200) *GAMMA(40) *REF(20) *SCRMAT(20)
4 *SUMRAY(200) *SUMSCR(200) *X(40) *XK(200)
5 *Y(40) *YK(200) *Z(40) *ZK(200)
6 *ZLAN(20)
COMMON /IARRAY/ IFLAG(40) *IPOINT(50)
*ISAX(50) *ISCR(20) *JF(200) *JFL(200)
*JL(200) *JM(200) *JN(200) *JP(200)
*JRAY(200) *JSCR(50) *KEND(900) *KFLAG(200)
*KP(200) *KTP(200) *LAMP(50) *LANGLE(20)
*LRFL(200) *LTYPE(20) *MAT(200) *MFL(20)
*MMAT(20) *MMRAY(200)
COMMON /PARAMS/ AL *ALN *ALR *ALRF *RE
*BEA *BER *BERF *DIS *DMIN *DZ *EPS
*GA *GAN *GAR *GANF *PER
*SUMRA *XI *XNG *XOLD *XREF *XTRAN *YI
*YNO *YOLD *YREF *YTHA *ZI
*ZNO *ZOLD *ZREF *ZTRAN
COMMON /IPARAM/ FMAX *IMAX *INON
*ICORAY *ICORNT *IR *IRAY *JMAX *JNON
*KNAT *KNAT *KNOW *MMAX *MNON *NSCR *JOLD
INTEGER F,FMAX
COMMON /RNDOM/RNDOPAR(12)
C..... STORE INPUT DATA
XK(K)=RNDOPAR(1)
YK(K)=RNDOPAR(2)
ZK(K)=RNDOPAR(3)
X1=RNDOPAR(4)
Y1=RNDOPAR(5)
Z1=RNDOPAR(6)
ZF=RNDOPAR(7)
SF=ZF-ZK(K)
AA=2.0*(Z1-ZK(K))*SF-(Z1-ZK(K))*2
BB=(X1-XK(K))*2*(Y1-YK(K))*2
CC=4.0*SF*(Z1-ZK(K))
C..... SEE IF DATA DESCRIBES AN ELLIPSOID.
IF(AA-RR:101,101,103
101 IF(BB-CC:102,102,103
C..... DATA DOES NOT DESCRIBE ELLIPSOID.
103 PRINT 12,K
12 FORMAT(1H,21,DATA FOR BOUNDARY NO.,16,
*27, DOES NOT FIT AN ELLIPSOID.)
STOP
102 CONTINUE
C..... CALCULATE COEFFICIENTS NEEDED.
NSQ=(X1-XK(K))*2*(Y1-YK(K))*2
ZST=Z1-ZK(K)
R=SQRT((ZST-SF)*2*(50)*SIGN(1.0,SF))
C..... CALCULATE B.
EK(K)=(ZST*SF+R)/(4.0-50/(ZST*SF))
C..... CALCULATE A**2.
DK(K)=EK(K)*2-(EK(K)-SF)*2
C..... PRINT INPUT DATA.
PRINT 11,K,LRFL(K),(RNDOPAR(A),A=1,7)
11 FORMAT(1H,15,16,2X,6#ELLIPS,24X,7F8.3)
RETURN
END

```

```

SUBROUTINE SETMEI (K)
C.....
C.....
COMMON /69/      ABCD(20,200)  ,ALRAY(500)  ,BEAY(500)
1 ,COORDS(6,50)  ,E(40,200)    ,EBASE(200)  ,ESCH(20,200)
2 ,GARAY(500)    ,RED(20,200)  ,TITLE(12)  ,VPAT(20)
3 ,YOL(200)      ,XLAW(200)    ,XRAY(500)    ,YRAY(500)
4 ,ZRAY(500)
COMMON /ARRAYS/  ALK(200)      ,ALPHA(40)    ,BEK(200)
1 ,META(40)      ,DEV(200)     ,DK(200)     ,EK(200)
2 ,ELAW(200)     ,ELAMAT(20)   ,EMAT(20)   ,ENRGY(200)
3 ,GK(200)       ,GAMMA(40)    ,REF(20)     ,SUMMAT(20)
4 ,SUMRAY(200)   ,SUMSOR(200)  ,X(40)     ,XK(200)
5 ,Y(40)         ,YK(20)      ,Z(40)       ,ZK(200)
6 ,ZLAW(20)
COMMON /IARRAY/  IFLAG(40)    ,IPOINT(50)
,ISWA(50)      ,ISOR(20)      ,JF(200)     ,JFL(200)
,JL(200)       ,JM(200)      ,JA(200)     ,JP(200)
,JRAY(200)     ,JSOR(50)     ,KBND(900)   ,KFLAG(200)
,KP(200)       ,KTP(200)     ,LAMP(50)    ,LANGLE(20)
,LREFL(200)    ,LTYPE(20)    ,MAT(200)   ,MFL(20)
,MAT(20)       ,MUMRAY(200)
COMMON /PARAMS/  AL          ,ALN        ,ALR        ,ALRF       ,RE
,BEN          ,BER          ,BERF       ,DIS        ,DMIN       ,DZ        ,EPS
,GAL          ,GLN          ,GAR        ,GAWF       ,PER        ,
,SUMRA        ,XI          ,XNOW       ,XCLO        ,XREF       ,XTRAN       ,YI
,YMOW        ,YOLD        ,YREF       ,YTRAN       ,ZI
,ZNOW        ,ZOLD        ,ZREF       ,ZTRAN
COMMON /IPARIM/  FMAX        ,IMAX        ,INOW
,IOPRAY        ,IOPRNT       ,IR        ,IRAY        ,JMAX        ,JNOW
,KINT          ,KMAX        ,KNOW       ,KMAX        ,KNOW       ,NSOR
,JOLD
INTEGER F,FMAX
COMMON/RNDOM/RNDPAR(12)
IK(K)=RNDPAR(1)
YK(K)=RNDPAR(2)
ZK(K)=RNDPAR(3)
ALK(K)=RNDPAR(4)
BEK(K)=RNDPAR(5)
DK(K)=RNDPAR(6)
PRINT 19,K,LREFL(K),(RNDPAR(N),N=1,6)
10 FORMAT(1H ,15,16,2X,6H RELIX,24X,6F8.3)
RETURN
END

```

```

SUBROUTINE SETHYP(K)
C*****
C***** SETHYP SETS UP THE ARRAYS FOR A HYPERBOLOID.
C*****
COMMON /69/ ARCO(20,200) ,ALRAY(500) ,BERAY(500)
1 ,COPRDS(6,50) ,E(40,200) ,ERASE(200) ,ESOR(20,200)
2 ,GARAY(500) ,RED(20,200) ,TITLE(12) ,VPAT(20)
3 ,VOL(700) ,XLAM(200) ,XHAY(500) ,YHAY(500)
4 ,ZRAY(500)
COMMON /ARRAYS/ ALK(200) ,ALPHA(40) ,BEK(200)
1 ,REFY(40) ,DEV(200) ,DK(200) ,EK(200)
2 ,FLAM(200) ,ELAMAT(20) ,EMAT(20) ,ENERGY(200)
3 ,GAK(200) ,GAMMA(40) ,REF(20) ,SONMAT(20)
4 ,SUMRAY(200) ,SUMSCR(200) ,X(40) ,XK(200)
5 ,Y(40) ,YK(200) ,Z(40) ,ZK(200)
6 ,ZLAM(20)
COMMON /IARRAY/ IFLAG(40) ,IPOINT(50)
1 ,ISMAX(50) ,ISCR(20) ,JF(200) ,JFL(200)
2 ,JL(200) ,JM(200) ,JM(200) ,JP(200)
3 ,JRAY(200) ,JSCR(50) ,KBND(900) ,KFLAG(200)
4 ,KP(200) ,KTP(200) ,LAMP(50) ,LAMPLE(20)
5 ,LREFL(200) ,LTYPE(20) ,MAT(200) ,MFL(20)
6 ,NMAT(20) ,NUMRAY(200)
COMMON /PARAMS/ AL ,ALN ,ALR ,ALRF ,BE ,
1 ,GA ,GAN ,BERF ,DIS ,DMIN ,DZ ,EPS ,
2 ,SUMRA ,XI ,ANOW ,XCLO ,XREF ,XTRAN ,YI ,
3 ,YAG ,YOLD ,YREF ,YTRAN ,ZI ,
4 ,ZAG ,ZOLD ,ZREF ,ZTRAN
COMMON /IPARAM/ FMAX ,IMAX ,INOW ,
1 ,ICPRAY ,ICPRNT ,IR ,IRAY ,IMAX ,JNOW ,JOLD ,
2 ,KINT ,KMAX ,KNOW ,MMAX ,MNOW ,MSOR
INTEGER F,FMAX
COMMON /RNDCOM/RNDPAR(12)
C***** STORE INPUT DATA.
XK(K)=RNDPAR(1)
YK(K)=RNDPAR(2)
ZK(K)=RNDPAR(3)
GAK(K)=RNDPAR(7)-ZK(K)
X=RNDPAR(4)
Y=RNDPAR(5)
Z=RNDPAR(6)
A=4.0*GAK(K)*(Z1-ZK(K))
B=(X1-XK(K))**2+(Y1-YK(K))**2
C***** SEE IF DATA DESCRIBES A HYPERBOLOID.
IF(A-R)101,102,102
102 PRINT 10,K
10 FORMAT(1H1,ZNDATA FOR BOUNDARY NO.,16,
1 ,32NDGES NOT DESCRIBE A HYPERBOLOID.)
STOP
C***** CALCULATE COEFFICIENTS NEEDED.
101 DELX=X1-XK(K)
DELY=Y1-YK(K)
WSC=DELX**2+DELY**2
ZST=Z1-ZK(K)
R=SQRT((ZST-GAK(K))**2+WSC)*SIGN(1.0,ZST)
C***** CALCULATE AND STORE B.
EK(K)=(ZST-GAK(K)*R)/(WSC/(ZST-GAK(K))-4.0)
C***** CALCULATE AND STORE A**2.
DK(K)=(EK(K)*GAK(K))**2-EK(K)**2
C***** PRINT INPUT DATA.
PRINT 11,K,LREFL(K),(RNDPAR(N),N=1,7)
11 FORMAT(1H ,15,16,2X,6HMYPERB,24X,7F8,3)
RETURN
END

```

```

SUBROUTINE SETPAR(K)
C.....
C..... SETPAR SETS UP THE ARRAYS FOR A PARACLOID.
C.....
COMMON /69/ ABCO(20,200) ,ALRAY(500) ,EEHAY(500)
1 ,COORDS(6,50) ,E(40,200) ,EBASE(200) ,ESOM(20,200)
2 ,GARAY(500) ,RED(20,200) ,TITLF(12) ,VMAT(20)
3 ,VOL(200) ,XLAM(200) ,XRAY(500) ,YHAY(500)
4 ,ZRAY(500)
COMMON /ARRAYS/ ALK(200) ,ALPHA(40) ,BEK(200)
1 ,BETA(40) ,DEV(200) ,DK(200) ,EK(200)
2 ,FLAM(200) ,ELAMAT(20) ,EMAT(20) ,ENERGY(200)
3 ,GAK(200) ,GAMMA(40) ,REF(20) ,SCHMAT(20)
4 ,SUMRAY(200) ,SUMSOR(200) ,X(40) ,XK(200)
5 ,Y(40) ,YK(200) ,Z(40) ,ZK(200)
6 ,ZLAM(20)
COMMON /IARRAY/ IFLAG(40) ,IPRINT(50) ,
,ISMAX(50) ,JSOR(20) ,JF(200) ,JFL(200) ,
,JL(200) ,JM(200) ,JK(200) ,JP(200) ,
,JRAY(200) ,JSOR(50) ,KBN(900) ,KFLAG(200) ,
,KP(200) ,KIYP(200) ,LAMP(50) ,LANGLE(20) ,
,LREFL(200) ,LTYPE(20) ,MAT(200) ,MFL(20) ,
,MUMAT(20) ,MUMRAY(200)
COMMON /PARAMS/ AL ,ALN ,ALR ,ALRF ,RE ,
,BEN ,BER ,BERF ,DIS ,DMIN ,DZ ,EPS ,
,GA ,GAN ,GAN ,GAKF ,PER ,
,SUMRA ,XI ,XNOW ,XOLD ,XNEF ,XTRAM ,YI ,
,YNOW ,YOLO ,YREF ,YTRAM ,ZI ,
,ZAD ,ZCLO ,ZREF ,ZTHAN
COMMON /IPARAM/ FMAX ,IMAX ,INOW ,
,IOPRAY ,IOPRNT ,IR ,IRAY ,JMAX ,JNOW ,JOLD ,
,KINT ,KMAX ,KNOW ,NMAX ,NOW ,NSOR
INTEGER F,FMAX
COMMON /RNDCOM/BNOPAR(12)
C..... STORE INPUT DATA.
XK(K)=BNOPAR(1)
YK(K)=BNOPAR(2)
ZK(K)=BNOPAR(3)
DK(K)=BNOPAR(4)-ZK(K)
C..... PRINT INPUT DATA.
PRINT 11,K,LREFL(K),(BNOPAR(N),N=1,4),DK(K)
11 FORMAT(1H ,I5,16,3X,5HPARA8,24X,4F8.3,24X,F8.3)
RETURN
END

```

```

SUBROUTINE SETPLA(K)
C.....
C..... SETPLA SETS UP A PLANE BOUNDARY.
C.....
COMMON /69/      ARCO(20,200)  ,ALRAY(500)  ,BERAY(500)
1  ,COCROS(16,50) ,E(40,200)    ,EHASE(200)  ,ESOR(20,200)
2  ,GARAY(500)    ,RED(20,200)  ,TITLE(12)  ,VMAT(20)
3  ,VOL(200)      ,XLAM(200)    ,XRAY(500)    ,YRAY(500)
4  ,ZRAY(500)
COMMON /ARRAYS/  ALK(200)      ,ALPHA(40)    ,BEK(200)
1  ,DETA(40)      ,DEV(200)     ,DK(200)      ,EK(200)
2  ,ELAM(200)     ,ELAMAT(20)   ,EPAT(20)     ,ENERGY(200)
3  ,GAK(200)      ,GAMMA(40)   ,REF(20)       ,SORMAT(20)
4  ,SUMRAY(200)   ,SUMSOR(200)  ,X(40)       ,XK(200)
5  ,Y(40)         ,YK(200)     ,Z(40)       ,ZK(200)
6  ,ZLAM(20)
COMMON /IAHRAY/  IFLAG(40)     ,IPOINT(50)
,ISMAX(50)      ,ISOR(20)      ,JF(200)     ,JFL(200)
,JL(200)        ,JM(200)      ,JN(200)     ,JP(200)
,JRAY(200)      ,JSOR(50)     ,KBD(900)    ,KFLAG(200)
,KP(200)        ,KTP(200)     ,LAMP(50)    ,LANGLE(20)
,LREFL(200)     ,LTYPE(20)    ,MAT(200)    ,NFL(20)
,MAT(20)        ,NUMRAY(200)
COMMON /PARAMS/  AL            ,ALN         ,ALR         ,ALRF        ,AE
,BER            ,BERF         ,OIS         ,DMIN        ,OZ         ,EPS
,GAM            ,GAR          ,GAMF        ,PER          ,
,SUMRA         ,XI           ,INOW        ,ACLD         ,XREF        ,XTRAN        ,YI
,YOLD          ,YOLD         ,YREF        ,YTRAN        ,ZI
,ZOLD          ,ZOLD         ,ZREF        ,ZTRAN
COMMON /IPARAY/  FMAX          ,IMAX        ,INOW        ,JMAX        ,JNOW        ,JOLD
,IOPRAY        ,IOFFAT        ,IR          ,IRAY        ,JMAX        ,JNOW        ,NSOR
,KINT          ,KMAX          ,KNOW        ,MMAX        ,MNOW
INTEGER F,FMAX
COMMON /RNDOPAR/ RNDOPAR(12)
C..... STORE BOUNDARY DATA.
ALK(K)=RNDOPAR(1)
BEK(K)=RNDOPAR(2)
GAK(K)=RNDOPAR(3)
CALL CHECK(6,SETPLA,ALK(K),BEK(K),GAK(K))
XK(K)=RNDOPAR(4)
YK(K)=RNDOPAR(5)
ZK(K)=RNDOPAR(6)
C..... CALCULATE DISTANCE TO BOUNDARY.
DK(K)=ALK(K)*XK(K)+BEK(K)*YK(K)+GAK(K)*ZK(K)
C..... IF DISTANCE LESS THAN OR EQUAL TO ZERO, PRINT AND STOP.
IF(DK(K))101,101,102
101 PRINT 11,K,DK(K)
11 FORMAT(1H0,12#BOUNDARY NO.,16,15# HAS A NEGATIVE,
20# OR ZERO DISTANCE OF,1PE12.4)
STOP
102 CONTINUE
C..... PRINT BOUNDARY DATA FOR PLANE.
PRINT 10,K,LREFL(K),ALK(K),BEK(K),GAK(K),XK(K),YK(K),
,ZK(K),DK(K)
10 FORMAT(1H ,15,16,3X,5#PLANE,6F8.3,3X,2F8.3)
RETURN
END

```

```

SUBROUTINE SETSPH(K)
C*****
C***** SETSPH SETS UP A SPHERICAL BOUNDARY.
C*****
COMMON /69/ ABCO(20,200) ALRAY(500) BERAY(500)
1 COMROS(6,50) E(40,200) EBASE(200) ESON(20,200)
2 CARAY(500) RED(20,200) TITLE(12) VPAT(20)
3 VOL(200) XLAM(200) XRAY(500) YRAY(500)
4 ZRAY(500)
COMMON /ARRAYS/ ALK(200) ALPHA(40) BEK(200)
1 RETA(40) DEV(200) DK(200) EK(200)
2 FLAM(200) ELAMAT(20) EMAT(20) ENERGY(200)
3 RAK(200) GAMMA(40) REF(20) SGMMAT(20)
4 SJMRAY(200) SUMSOR(200) X(40) XK(200)
5 Y(40) YK(200) Z(40) ZK(200)
6 LAM(20)
COMMON /IARRAY/ IFLAG(40) IPOINT(50)
1 ISMAX(50) ISOR(20) JF(200) JFL(200)
2 JL(200) JM(200) JN(200) JP(200)
3 JRAY(200) JSOR(50) KBND(900) KFLAG(200)
4 KP(700) KTYP(200) LAMP(50) LANGLE(20)
5 LREFL(200) LTYPE(20) MAT(200) MFL(20)
6 MMAT(20) MMURAY(200)
COMMON /PARAMS/ AL ALN ALR ALRF RE
1 BEA BER BERF DIS DMIN DZ EPS
2 GA GAN GAR GAKF PER
3 SUMRA XI XMGW XOLD XREF XTRAN YI
4 YMGW YOLD YREF YTRAN ZI
5 ZMGW ZOLD ZREF ZTMAN
COMMON /IPARAM/ FMAX IMAX INOW
1 IOPRAY IOFRNT IR IRAY JMAX JNOM JOLO
2 KINT KMAX KNOW MMAX MNOW NSOR
INTEGER F,FMAX
COMMON /RNDCON/RNDPAR(12)
C***** STORE INPUT DATA.
XK(K)=RNDPAR(1)
YK(K)=RNDPAR(2)
ZK(K)=RNDPAR(3)
DK(K)=RNDPAR(4)
C***** PRINT INPUT DATA.
PRINT 11,K,LREFL(K),XK(K),YK(K),ZK(K),DK(K)
11 FORMAT(1H,15,16,2X,6HSPHERE,24X,3F8.3,32X,F8.3)
RETURN
END

```



```

SUBROUTINE SETUPP
C.....
C..... SUBROUTINE SETUPP READS THE INPUT DATA AND HANDLES ALL
C..... SETUP AND PROBLEM INITIALIZATION.
C.....
COMMON /69/ ABCG(20,200) ,ALRAY(500) ,BERAY(500)
1 ,CDORS(6,50) ,E(40,200) ,EHASE(200) ,ESOR(20,200)
2 ,GARAY(500) ,REDI(20,200) ,TITLF(12) ,VMAT(20)
3 ,VOL(200) ,XLAM(200) ,XRAY(500) ,YRAY(500)
4 ,ZRAY(500)
COMMON /ARRAYS/ ALK(200) ,ALPHA(40) ,BEK(200)
1 ,BETA(40) ,DEV(200) ,DK(200) ,EK(200)
2 ,FLAM(200) ,ELAMAT(20) ,EMAT(20) ,ENERGY(200)
3 ,GAK(200) ,GAMMA(40) ,REF(20) ,SGRMAT(20)
4 ,SUMRAY(200) ,SUMSOR(200) ,X(40) ,XK(200)
5 ,Y(40) ,YK(200) ,Z(40) ,ZK(200)
6 ,ZLAM(20)
COMMON /IARRAY/ IFLAG(43) ,IPCINT(50) ,
,ISMAX(50) ,JSOR(20) ,JF(200) ,JFL(200) ,
,JL(200) ,JM(200) ,JN(200) ,JP(200) ,
,JRAY(200) ,JSOR(50) ,KBAD(500) ,KFLAG(200) ,
,KP(200) ,KTYP(200) ,LAMP(50) ,LANGLE(20) ,
,LREFL(200) ,LTYPE(20) ,MAT(200) ,MFL(20) ,
,OMAT(20) ,OMMR(200)
COMMON /PARAMS/ AL ,ALN ,ALR ,ALRF ,RE ,
,BEN ,BER ,BERF ,DIS ,DMIN ,DZ ,EPS ,
,GA ,GAN ,GAR ,GARF ,PER ,
,SUMRA ,XI ,XND ,XCLD ,XREF ,XTRAN ,YI ,
,YND ,YOLD ,YREF ,YTRAN ,ZI ,
,ZNGN ,ZCLD ,ZREF ,ZTRAN ,
COMMON /PARC/ FMAX ,INAX ,INOW ,
,IOGRAY ,IOPRNT ,IR ,IRAY ,JMAX ,JNOW ,JOLD ,
,KINT ,KMAX ,KNOW ,MMAX ,MNOW ,MSOR ,
INTEGER F,FMAX
COMMON /BASE/ FRAC ,IDEBUG
C..... INITIALIZE PARAMETERS.
IRAY=0
JOLD=1
EPS=1.E-3
SUMRA=0.0
INAX=0
PRINT 12
12 FORMAT(1H,21HGENERAL INPUT FOR ZAP//)
C..... READ INPUT DATA.
READ 10,TITLE
10 FORMAT(12A6)
READ 11,FFAC,DZ,WAYMIN,WAYMAX
11 FORMAT(6E12.4)
READ 15,FMAX,IOPRNT,IDEBUG,IR
15 FORMAT(316,116)
C..... IF MULTIPLE WAVELENGTHS, CALCULATE AND STORE THEM.
IF(FMAX-1)109,110,111
109 FMAX=1
GO TO 110
111 DF=(WAYMAX-WAYMIN)/(FMAX-1)
WAYF=WAYMIN
DO 112 F=1,FMAX
XLAM(F)=WAYF
ALAM(F)=WAYF
WAYF=WAYF+DF
112 CONTINUE
GO TO 113

```

```

110 XLAM(1)=WAVMIN
113 CONTINUE
    IF (FRAC) 105,105,107
105 FRAC=.5
107 CONTINUE
    PRINT 13,TITLE
13 FORMAT(1H0,12A6//)
    PRINT 14,FRAC,DZ,WAVMIN,WAVMAX,FMAX,10FRNT,DEBUG,IR
14 FORMAT(1H ,18HROULETTE FRACTION=E12.4/
    *42H DISTANCE BETWEEN CONTINUITY BOUNDARIES=E12.4/
    *20H MAXIMUM WAVELENGTH=E12.4/
    *20H MAXIMUM WAVELENGTH=E12.4/
    *23H NUMBER OF WAVELENGTHS=.16/
    *18H RAY PRINT OPTION=.16/20H DEBUG PRINT OPTION=.16/
    *25H STARTING RANDOM INTEGER=.116)
C***** READ REFLECTION DATA
100 CALL REFDAT
C***** READ MATERIAL DATA AND SET UP MATERIAL ARRAYS.
    CALL MATDAT
C***** READ BOUNDARY DATA AND SET UP BOUNDARY ARRAYS.
    CALL BDDAT
C***** READ SEGMENT DATA AND SET UP SEGMENT ARRAYS.
    CALL SEGDAT
C***** READ STRUCTURE DATA AND SET UP SEGMENT STRUCTURE.
    CALL STRUCT
C***** READ LAMP DATA AND SET UP LAMP ARRAYS.
    CALL LAMPAT
C***** READ SOURCE DATA.
    CALL SRCMAT
C***** READ TEST POINT DATA AND TEST SETUP.
    CALL TEST
C***** GENERATE PRINTER PLOTS OF GEOMETRY.
    CALL GEOPLT
    CALL DEBUG(6HSETUP)
    RETURN
END

```

```

SUBROUTINE SOFTM
C.....
C..... SOFTM CALCULATES STM AND BSTM.
C..... PART OF HELIX INTERSECTION CALCULATION.
C.....
COMMON /69/ ABCU(20,200) ,ALRAY(500) ,BENAY(500)
1 ,COORDS(6,50) ,E(40,200) ,EBASE(200) ,ESOM(20,200)
2 ,GARAY(500) ,REU(20,200) ,TITLE(12) ,VMAT(20)
3 ,VOL(200) ,XLAM(200) ,XRAY(500) ,YHAY(500)
4 ,ZRAY(500)
COMMON /ARRAYS/ ALX(200) ,ALPHA(40) ,BEK(200)
1 ,ETA(40) ,DEV(200) ,DK(200) ,EK(200)
2 ,FLAM(200) ,ELAMAT(20) ,EMAT(20) ,ENERGY(200)
3 ,GAK(200) ,GAMMA(40) ,REF(20) ,SONMAT(20)
4 ,SUMRAY(200) ,SUMSOR(200) ,X(40) ,XK(200)
5 ,Y(40) ,YK(200) ,Z(40) ,ZK(200)
6 ,ZLAM(20)
COMMON /IARRAY/ IFLAG(40) ,IPOINT(50) ,
1 ,ISMAX(50) ,ISOR(20) ,JF(200) ,JFL(200) ,
2 ,JL(200) ,JM(200) ,JN(200) ,JP(200) ,
3 ,JRAY(200) ,JSOR(50) ,KEND(900) ,KFLAG(200) ,
4 ,KP(200) ,KTP(200) ,LAMP(50) ,LANGLE(20) ,
5 ,LRFFL(200) ,LTYPE(20) ,MAT(200) ,MFL(20) ,
6 ,MAT(20) ,MUMRAY(200)
COMMON /PARAMS/ AL ,ALN ,ALR ,ALRF ,RE ,
1 ,BEN ,BERF ,DIS ,DMIN ,DZ ,EPS ,
2 ,GA ,GAN ,GAR ,GARF ,PER ,
3 ,SUMRA ,XI ,XNOW ,XOLD ,XREF ,XTRAN ,YI ,
4 ,YKOW ,YOLD ,YHEF ,YTHAN ,ZI ,
5 ,ZKOW ,ZOLD ,ZREF ,ZTHAN ,
COMMON /IPARAM/ FMAX ,IMAX ,INOW ,
1 ,IOPHAY ,IOPRNT ,IR ,IRAY ,JMAX ,JNOW ,JO_D ,
2 ,KINT ,KMAX ,KNOV ,MMAX ,MNOV ,NSOH
INTEGER F,FMAX
COMMON /HELCON/ A,ABAN,B,B1,BIP,BP,CAP,DELTH,DELX,DELY,DELZ,DMIT,
1 ,PSIZ,REV,RNO,STH,TH,TH1,SORAS,RKS,CSTH,SATH,RCSTH,RSNTH,BSTH,
2 ,COSTAN,TNZ,DTNZ
STH=AL*(DELX*HCSTH)+BE*(DELT*RSNTH)+GA*(DELZ*CAP*TH)
BSTH=(DELX*RCSTH-AL*STH)**2+(DELY*NSNTH-BE*STH)**2
3 , (DELZ*CAP*TH-GA*STH)**2
RETURN
END

```

```

SUBROUTINE SORDAT
C*****
C***** SORDAT READS SOURCE DATA CARDS AND SETS UP SOURCES.
C*****
COMMON /69/ ABCO(20,200) ,ALRAY(500) ,BEHAY(500)
1 ,CHARTS(6,50) ,E(40,200) ,EBASE(200) ,ESOM(20,200)
2 ,CARAY(500) ,RED(20,200) ,TITLE(12) ,VMAT(20)
3 ,VOL(200) ,XLAN(200) ,XPAY(500) ,YRAT(500)
4 ,ZRAY(500)
COMMON /ARRAYS/ ALK(200) ,ALPHA(40) ,BEK(200)
1 ,META(40) ,DEV(200) ,DK(200) ,EK(200)
2 ,ELAM(200) ,ELAMAT(20) ,EMAT(20) ,ENEMY(200)
3 ,GAK(200) ,GAPPA(40) ,REF(20) ,SOMAT(20)
4 ,SUMHAY(200) ,SUMSOR(200) ,X(40) ,XK(200)
5 ,Y(40) ,YK(200) ,Z(40) ,ZK(200)
6 ,ZLAN(20)
COMMON /IARRAY/ IFLAG(40) ,IPOINT(50)
1 ,ISMAX(40) ,ISOR(20) ,JF(200) ,JFL(200)
2 ,JL(200) ,JM(200) ,JM(200) ,JP(200)
3 ,JRAY(200) ,JSOR(50) ,KEND(900) ,KFLAG(200)
4 ,KP(200) ,KTP(200) ,LAMP(50) ,LANGLE(20)
5 ,LREF(200) ,LTYPE(20) ,MAT(200) ,MFL(20)
6 ,MPAT(20) ,MURRAY(200)
COMMON /PIPARAMS/ AL ,ALN ,ALR ,ALRF ,RE
1 ,BEK ,BEF ,BERF ,DIS ,DNIN ,DZ ,EPS
2 ,GA ,GAN ,GAR ,GARF ,PER
3 ,SUNRA ,X1 ,XINW ,XCLO ,XREF ,XTRAN ,Y1
4 ,YNOV ,YOLD ,YREF ,YTHAN ,Z1
5 ,ZNOV ,ZOLD ,ZREF ,ZTRAN
COMMON /IPARAM/ FMAX ,IMAX ,INOW
1 ,IOPRAY ,IOPRNT ,IR ,IRAY ,JMAX ,JNOW ,JOLD
2 ,KINT ,KMAX ,KNOW ,MMAX ,MNOW ,NSOR
INTEGER F,FMAX
DIMENSION TEMP(6)
DATA(1F=1)
C***** PRINT PAGE HEADING.
PRINT 9
9 FORMAT(1H1,53A,11HSOURCE DATA//1H ,6HSOURCE,3X,7HSEGMENT,
1 ,2X,4HBOUNDARY,6X,4HLM,4X,6HNUMBER,4X,
2 ,21HGEOMETRIC SOURCE DATA//1H ,6HNUMBER,4X,6HNUMBER,
3 ,4X,6HNUMBER,4X,6HNUMBER,3X,7HOF RAYS/)
NSOR=0
C***** READ A SOURCE DATA CARD.
112 READ 12,N,JSOR,LAMP,ISMAX,TEMP
12 FORMAT(31A,15,6F9,3)
C***** N=0 INDICATES END OF SOURCE DATA.
IF(N)111,111,100
C***** RETRIEVE SOURCE TYPE.
100 ISOR=JSOR(LAMP)
C***** BRANCH ON SOURCE TYPE.
GO TO(101,102,102,104),ISOR
C***** VOLUME SOURCE
101 NBN=0
NSEG=JSOR
GO TO 103
C***** SURFACE SOURCE
102 NBN=JSOR
NSEG=0
GO TO 103
C***** RAY INPUT SOURCE
104 NBN=0

```

```

      NSOR=0
C***** PRINT SOURCE DATA
163 PRINT 13,N,NSOR,NSOR,LAMPN,ISMAXI,TEMP
13 FORMAT(1M,16.4,110.6F12.5)
C***** IF N GREATER THAN NSOR, NSOR=N
      IF(N>NSOR)106,106,107
107 NSOR=N
C***** STORE SOURCE DATA.
106 JSOR(N)=JSOR
      ISMAXI=ISMAXI
      LAMPN=LAMPN
C***** IF RAYS ARE TO BE INPUT, GO TO 108
      IF(15000-4)105,108,105
C***** STORE COORDINATES OF PARALLELOPIPED CONTAINING SOURCE.
105 COORDS(2,N)=TEMP(2)-TEMP(1)
      COORDS(1,N)=TEMP(1)
      COORDS(4,N)=TEMP(4)-TEMP(3)
      COORDS(3,N)=TEMP(3)
      COORDS(6,N)=TEMP(6)-TEMP(5)
      COORDS(5,N)=TEMP(5)
      GO TO 109
C***** READ INPUT RAYS.
109 IPPOINT(N)=IP
      DO 110 I=1,ISMAXI
      READ 14,ALRAY(IP),BERAY(IP),GARAY(IP),XRAY(IP),
      YRAY(IP),ZRAY(IP)
14 FORMAT(17X,6F8.3)
C***** CHECK DIRECTIONAL COSINES.
      CALL CHECK(6NSORDAT,ALRAY(IP),BERAY(IP),GARAY(IP))
C***** INCREASE POINTER BY 1.
      IP=IP+1
110 CONTINUE
C***** GO LOOK AT NEXT SOURCE DATA CARD.
109 GO TO 112
111 RETURN
      END

```

```

SUBROUTINE SPHNOR(K)
C*****
C***** SPHNOR FINDS THE DIRECTIONAL COSINES OF THE NORMAL TO
C***** A POINT ON THE SURFACE OF A SPHERE.
C*****
COMMON /69/ ABCD(20,200) ,ALRAY(500) ,BERAY(500)
1 ,COCOS(6,50) ,E(40,200) ,EBASE(200) ,ESCH(20,200)
2 ,GARAY(500) ,RED(20,200) ,TITLE(12) ,VMAT(20)
3 ,VOL(200) ,XLAM(200) ,XRAY(500) ,YRAY(500)
4 ,ZRAY(500)
COMMON /ARRAYS/ ALK(200) ,ALPHA(40) ,BEK(200)
1 ,BETA(40) ,DEV(200) ,DK(200) ,EK(200)
2 ,ELAM(200) ,ELAMAT(20) ,EMAT(20) ,ENERGY(200)
3 ,GAK(200) ,GAMMA(40) ,REF(20) ,SUNMAT(20)
4 ,SUMRAY(200) ,SUMSUN(200) ,X(40) ,XK(200)
5 ,Y(40) ,YK(200) ,Z(40) ,ZK(200)
6 ,ZLAM(20)
COMMON /TARRAY/ IFLAG(40) ,IPOINT(50) ,
,ISMAX(50) ,ISOR(20) ,JF(200) ,JFL(200) ,
,JL(200) ,JM(200) ,JN(200) ,JP(200) ,
,JRAY(200) ,JSOR(50) ,KBAD(900) ,KFLAG(200) ,
,KP(200) ,KTP(200) ,LAMP(50) ,LANGE(20) ,
,LREFL(200) ,LTYPE(20) ,MAT(200) ,MFL(20) ,
,MAT(20) ,MUMRAY(200)
COMMON /PARAMS/ AL ,ALN ,ALR ,ALRF ,RE ,
,BEN ,BER ,BERF ,DTS ,DMIN ,DZ ,EPS ,
,GA ,GAN ,GAR ,GANF ,PER ,
,SUMRA ,XI ,XNOW ,XCLO ,XREF ,XTRAM ,YI ,
,YNOW ,YOLD ,YREF ,YTHAN ,ZI ,
,ZNOW ,ZOLD ,ZREF ,ZTHAN ,
COMMON /IPARAM/ FRAX ,IPAX ,INOW ,
,IDPRAY ,IDPRNT ,IR ,IRAY ,JMAX ,JNOW ,JOLD ,
,KINT ,KMAX ,KNOW ,PMAX ,PNOW ,PSOR
INTEGER F,FMAX
C***** CALCULATE DIRECTIONAL COSINES.
ALN=(XNOW-XK(K))/DK(K)
BEN=(YNOW-YK(K))/DK(K)
GAN=(ZNOW-ZK(K))/DK(K)
C***** CHECK DIRECTIONAL COSINES.
CALL CHECK(6HSPHNOR,ALN,BEN,GAN)
CALL DERUG(6HSPHNOR)
RETURN
END

```

```

SUBROUTINE SSOURCE
C.....
C..... SSOURCE CREATES A RAY FROM A SURFACE SOURCE.
C.....
COMMON /69/ ABCO(20,200) ,ALRAY(500) ,BLRAY(500)
1 ,COORDS(6,50) ,E(40,200) ,EHASE(200) ,ESOM(20,200)
2 ,FARAY(500) ,RED(20,200) ,TITLE(12) ,VHAT(20)
3 ,VXL(200) ,XLAM(200) ,IRAY(500) ,VHAY(500)
4 ,ZRAY(500)
COMMON /ARRAYS/ ALK(200) ,ALPHA(40) ,BEK(200)
1 ,PETA(40) ,DEV(200) ,DK(200) ,EK(200)
2 ,FLAM(200) ,ELAMAT(20) ,EMAT(20) ,ENERGY(200)
3 ,FAM(200) ,GAMMA(40) ,REF(20) ,SUMMAT(20)
4 ,SUMRAY(200) ,SUMSON(200) ,X(40) ,XK(200)
5 ,Y(40) ,YK(200) ,Z(40) ,ZK(200)
6 ,ZLAM(20)
COMMON /IARRAY/ IFLAG(40) ,IPOINT(50)
1 ,ISMAX(40) ,ISOR(20) ,JF(200) ,JFL(200)
2 ,JL(200) ,JM(200) ,JN(200) ,JP(200)
3 ,JRAY(200) ,JSOR(50) ,KNO(900) ,KFLAG(200)
4 ,KP(200) ,KTPY(200) ,LAMP(50) ,LANGLE(20)
5 ,LREFFL(200) ,LTYPE(20) ,MAT(200) ,MFL(20)
6 ,MMAT(20) ,NUMRAY(200)
COMMON /PARAMS/ AL ,ALN ,ALR ,ALRF ,RE
1 ,BEA ,BER ,BERF ,DIS ,DMIN ,DZ ,EPS
2 ,GA ,GAN ,GAR ,GARF ,PER
3 ,SURA ,XI ,XNOW ,XOLD ,XREF ,XTRAN ,YI
4 ,YNO ,YOLD ,YREF ,YTRAN ,ZI
5 ,ZNOW ,ZOLD ,ZREF ,ZTRAN ,INOW
COMMON /IPARMS/ FMAX ,IMAX ,JMAX ,JNOW ,JOLD
1 ,IDIRAY ,IDIRNT ,IN ,IRAY ,JMAX ,JNOW ,JOLD
2 ,KINT ,KMAX ,KNOW ,PMAX ,PMOW ,NSOR
INTEGER F,FMAX
DATA(TWOPI=6.283185)
I=J=K=1
C..... FIND A RANDOM POINT(X(I),Y(I),Z(I)) ON THE SURFACE
TH=TCPI*RAND(0)
K=JSOR(NNOW)
SN=EPS
IF (ISOR(NNO)-3) 104,103,104
103 SN=-SN
104 CONTINUE
X(I)=XK(K)+(DK(K)*SN)*COS(TH)
Y(I)=YK(K)+(DK(K)*SN)*SIN(TH)
Z(I)=COORDS(5,NNOW)*COORDS(6,NNOW)*RAND(0)
KNOW=K
XNOW=X(I)
YNOW=Y(I)
ZNOW=Z(I)
C..... FIND THE NORMAL TO THE POINT.
CALL NCPAL
C..... GET TWO ANGLES FOR DESIRED PROBABILITY DISTRIBUTION.
CALL ANGLES(TH,PHI)
COSTH=COS(TH)
COSPHI=COS(PHI)
SINPHI=SIN(PHI)
SINTH=SQRT(1.0-COSTH**2)
C..... CALCULATE DIRECTIONAL COSINES OF RAY.
ALPHA(I)=ALN*COSTH+ALN*SINPHI*SINTH
BETA(I)=BERN*COSTH-ALN*SINPHI*SINTH
GAMMA(I)=COSPHI*SINTH
C..... IF RAY SHOULD POINT INWARDS, REVERSE DIRECTION OF RAY.
IF(SN) 101,102,102
101 ALPHA(I)=-ALPHA(I)
BETA(I)=-BETA(I)
GAMMA(I)=-GAMMA(I)
102 RETURN
END

```

```

SUBROUTINE STRUCT
C.....
C..... SURROUTINE STRUCT SETS UP THE SEGMENT STRUCTURE(HIERARCHY).
C.....
COMMON /69/ ABCO(20,200) ,ALRAY(500) ,BRAY(500)
1 ,COORDS(6,50) ,E(40,200) ,EBASE(200) ,ESOR(20,200)
2 ,GARAY(500) ,RED(20,200) ,TITLE(12) ,VMAT(20)
3 ,VCL(200) ,XLAM(200) ,XRAY(500) ,YRAY(500)
4 ,ZRAY(500)
COMMON /ARRAYS/ ALK(200) ,ALPHA(40) ,BEK(200)
1 ,BETA(40) ,DEV(200) ,DK(200) ,EK(200)
2 ,ELAM(200) ,ELAMAT(20) ,EMAT(20) ,ENERGY(200)
3 ,GAK(200) ,GAMMA(40) ,REF(20) ,SCRMAT(20)
4 ,SUMRAY(200) ,SUMSOR(200) ,X(40) ,XK(200)
5 ,Y(40) ,YK(200) ,Z(40) ,ZK(200)
6 ,ZLAM(20)
COMMON /IAR-AY/ IFLAG(40) ,IPOINT(50) ,
,ISMAX(50) ,ISOR(20) ,JF(200) ,JFL(200) ,
,JL(200) ,JM(200) ,JN(200) ,JP(200) ,
,JRAY(200) ,JSOR(50) ,KEND(900) ,KFLAG(200) ,
,KP(200) ,KTYP(200) ,LAMP(50) ,LANGLE(20) ,
,LREFL(200) ,LTYPE(20) ,MAT(200) ,MFL(20) ,
,MNAT(20) ,MURAY(200)
COMMON /PARAMS/ AL ,ALN ,ALR ,ALRF ,RE ,
,BEN ,BEN ,BERF ,DIS ,DMIN ,DZ ,EPS ,
,GA ,GAN ,GAR ,GANF ,PER ,
,SUMRA ,X1 ,XNOW ,XOLD ,XREF ,XTRAN ,YI ,
,YNOW ,YOLD ,YREF ,YTRAN ,ZI ,
,ZNOW ,ZOLD ,ZREF ,ZTRAN
COMMON /IPARAN/ FMAX ,IMAX ,INOW ,
,IOPRAY ,IOPRNT ,IR ,IRAY ,JMAX ,JNOW ,JOLD ,
,KINT ,KMAX ,KNOW ,MMAX ,MNOW ,MSOR
INTEGER F,FMAX
DIMENSION JOAU(22)
PRINT 9
9 FORMAT(1H,53X,14HSTRUCTURE DATA//8H MOTHER,10X,
,17HDAUGHTER SEGMENTS,34X,
,8H SEGMENT/)
NERS=0
C..... READ ONE MOTHER AND ITS DAUGHTERS.
120 READ 10,MOM,(JOAU(L),L=1,17)
10 FORMAT(1H14)
C..... ZERO MOTHER SEGMENT INDICATES END OF DATA.
IF(MOM)122,122,121
C..... COUNT NUMBER OF DAUGHTERS.
121 DO 102 LL=1,17
LL=LL
IF(JOAU(LL))101,101,102
102 CONTINUE
101 LL=LL-1
C..... PRINT MOTHER AND DAUGHTERS.
PRINT 11,MOM,(JOAU(L),L=1,17)
11 FORMAT(1H,17,5X,1716)
LL=1
J=J
C..... ERROR IF MOTHER HAS NO DAUGHTERS.
IF(JOAU(1))103,103,104
103 PRINT 12,JOAU(1)
12 FORMAT(1H,14HNO OTHER SEGMENT,14,17H HAS NO DAUGHTERS)
NERS=NERS+1
GO TO 110

```



```

C***** SET VALUES FOR ARRAYS.
104 JF(MO-1)=JDAU(LL)
105 IF(JDAU(LL))109,109,106
106 JM=J
    J=JDAU(LL)
    JP(J)=JM
C***** ERROR IF SEGMENT APPEARED AS DAUGHTER ON TWO CARDS.
    IF(JM(J))108,108,107
107 PRINT 13,J
13 FORMAT(1H ,7HSEGMENT,14,33H APPEARS AS DAUGHTER ON TWO CARDS)
    NERS=NERS+1
    LL=LL+1
    GO TO 105
C***** SET MOTHER OF SEGMENT J.
108 JM(J)=MOM
    LL=LL+1
    GO TO 105
C***** SET LAST DAUGHTER OF SEGMENT.
109 JL(MOM)=J
    LL=LL-1
111 LL=LL-1
    IF(LL) 110,110,112
112 JP=J
    J=JDAU(LL)
C***** SET NEXT DAUGHTER IN MEIRARCHY.
    JM(J)=JP1
    GO TO 111
110 GO TO 120
122 CONTINUE
C***** CHECK TO SEE IF ANY SEGMENT HAS NO MOTHER.
    DO 116 J=2,JN+1
113 IF(JF(J))116,116,114
114 IF(JM(J))115,115,116
115 PRINT 14,
14 FORMAT(15H MOTHER SEGMENT,14,14H HAS NO MOTHER)
    NERS=NERS+1
116 CONTINUE
117 IF(NERS)119,119,118
118 PRINT 15,NERS
15 FORMAT(1H ,13,33H ERRORS IN SEGMENT STRUCTURE DATA)
    STOP
119 RETURN
    END

```

```

SUBROUTINE TABREF(ILL)
COMMON /69/ ABCO(20,200) ALRAY(500) BERAY(500)
1 CONRNS(6,50) E(40,200) EBASE(200) ESOR(20,200)
2 GARAY(500) RED(20,200) TITLE(12) VMAT(20)
3 VOL(200) XLAM(200) XRAY(500) YRAY(500)
4 ZRAY(500)
COMMON /ARRAYS/ ALK(200) ALPHA(40) BEK(200)
1 BETA(40) DEV(200) DK(200) EK(200)
2 ELAM(200) ELAMAT(20) EMAT(20) ENERGY(200)
3 HAK(200) GAMMA(40) REF(20) SURMAT(20)
4 SUMRAY(200) SUMSOR(200) X(40) XA(200)
5 Y(40) YK(200) Z(40) ZK(200)
6 ZLAM(20)
COMMON /ARRAY/ IFLAG(40) IPOINT(50)
1 ISMAX(50) ISOR(20) JF(200) JFL(200)
2 JL(200) JM(200) JN(200) JP(200)
3 JRAY(200) JSOR(50) KBND(900) KFLAF(200)
4 KRP(200) KTYP(200) LAMP(50) LANGLE(20)
5 LREFL(200) LTYPE(20) MAT(200) MFL(20)
6 NMAT(20) NUMRAY(200)
COMMON /PARAMS/ AL ALR ALR DIS ALRF RE
1 REN BER BERF DIS DMIN DZ EPS
2 GA GAN GAN GAMF PER XTRAN YI
3 SUMRA XI XAOM XCLO XNEF
4 YNO YOLD YREF YTRAN ZI
5 ZNO ZCLO ZREF ZTRAN
COMMON /IPARAM/ FMAX IMAX INOW
1 IOPAY ICPRNT IR IRAY JMAX JNOW JOLD
2 KINT KMAX KNC MNAX MNC NSOR NSON
INTEGER F,FMAX
DIMENSION ANGLE(10)
DATA(ANGLE=1.0,.98481,.93969,.86603,.76604,
.64779,.5,.34202,.17365,0.0)
***** CALCULATE COSINE OF ANGLE OF INCIDENCE.
COSTH=AL*AL+BE*BE+GA*GA
COSTH=ARS(COSTH)
***** FIND POSITION IN TABLE.
DO 100 I=1,10
IF(COSTH-ANGLE(I))100,101,101
100 CONTINUE
***** INTERPOLATE FOR PERCENT TRANSMITTED.
101 PER=1.0-(RED(ILL,I)-(RED(ILL,I)-RED(ILL,I-1))
*(COSTH-ANGLE(I-1))/(ANGLE(I)-ANGLE(I-1)))
RETURN
END

```

SUBROUTINE TEST

 TEST READS CARDS CONTAINING TEST POINTS FOR CHECKING THE
 INPUT GEOMETRIES FOR ERRORS

```

COMMON /59/      ABC(20,200)    ALRAY(500)    BERAY(500)
1  COORDS(16,50)  E(40,200)      EBASE(200)    ESOR(20,200)
2  CARAY(500)     RED(20,200)    TITLE(12)    VMAT(20)
3  VOL(200)       XLAM(200)      XRAY(500)     YRAT(500)
4  ZRAY(500)

COMMON /ARRAYS/  ALK(200)        ALPHA(40)      BEK(200)
1  RETA(40)       DEV(200)       DK(200)       EK(200)
2  ELAM(200)      ELAMAT(20)     EMAT(20)      ENERGY(200)
3  GAK(200)       GAMMA(40)     REF(20)       SORMAT(20)
4  SUMRAY(200)    SUMSOR(200)    X(40)        XK(200)
5  Y(40)          YK(200)        Z(40)        ZK(200)
6  ZLAM(20)

COMMON /IARRAY/  IFLAG(40)       IPOINT(50)
1  ISMAX(50)      ISOR(20)       JF(200)       JFL(200)
2  JL(200)        JM(200)       JM(200)       JP(200)
3  JRAY(200)      JSOR(50)      KBND(900)     KFLAG(200)
4  KP(200)        KTYP(200)     LAMP(50)      LANGLE(20)
5  LREFL(200)     LTYPE(20)     MAT(200)      MFL(20)
6  MMAT(20)       MMURAY(200)

COMMON /PARAMS/  AL              ALN          ALR          ALRF          RE          .
1  BEN           DER            DERF         DIS          DMIN         DZ          EPS          .
2  GA            GAN           GAR           GARF         PER          .
3  SUMRA         XI            XNOW       XOLD         XREF         XTRAN       YI          .
4  YNO          YOLD          YREF         YTHAN        ZI           .
5  ZNOW         ZOLD          ZREF         ZTRAN        .
6  COMMON /IPARAM/ FMAX          INAX         INCW          .
1  IOPRAY        IOPRNT       IR           IRAY         JMAX         JNOW         JOLD          .
2  KINT          KMAX         KNEW       KMAX         KNEW       KNSOR
3  INTEGER F,FMAX
4  PRINT 12
5  12 FORMAT(1H,53X,9HTEST DATA//1H,7HSEGMENT,2H OPTION,
6  11X,1HX,11X,1MY,11X,1NZ/)
7  ISTOP=0
8  ***** READ A CARD
9  112 READ 10,JTEST,10P,XI,YI,ZI
10 10 FORMAT(2I6,3E12,4)
11 ***** ZERO JTEST INDICATES END OF DATA.
12 IF(JTEST)100,110,100
13 ***** SET PRESENT SEGMENT NUMBER.
14 100 JNOW=IARS(JTEST)
15 110 PRINT 13,JTEST,10P,XI,YI,ZI
16 13 FORMAT(1H,17,2X,16,3E12,4)
17 ***** FOR IOP=0, TEST SEGMENT BOUNDARIES ONLY
18 ***** FOR IOP=1, TEST TO SEE IF POINT IS INSIDE SEGMENT.
19 IF(10P)101,102,101
20 ***** FIND SEGMENT CONTAINING POINT
21 101 CALL SEGNT
22 GO TO 105
23 ***** SEE IF POINT IS INSIDE OF THE BOUNDARIES OF JNOW.
24 102 IF(INSEG(JNOW))104,103,104
25 ***** POINT NOT INSIDE SEGMENT JNOW
26 103 JNOW=0
27 GO TO 105
28 ***** POINT IS INSIDE SEGMENT JNOW
29 104 JNOW=IARS(JTEST)
30 ***** IF POINT IS IN SEGMENT JTEST, GO TO 106

```

```

105 IF(JNOW-1ABS(JTEST))107.106.107
C***** SEE IF DESIRED RESULT WAS OBTAINED.
106 IF(JTEST)109.108.108
107 IF(JTEST)108.109.109
108 IERR=0
    GO TO 111
109 IERR=1
C***** DID AN ERROR OCCUR.
111 IF(IERR)112.112.113
113 ISTOP=1
    CALL SEGMENT
    PRINT 11,JNOW
    11 FORMAT(10X,24HABOVE TEST POINT IN SEG .14.14M - RUN ABORTED/)
    GO TO 112
C***** IF ERRORS OCCURRED, STOP
110 IF(ISTOP)115.115.114
114 STOP
115 RETURN
    END

```

```

SUBROUTINE THOFS
C.....
C..... THOFS CALCULATES TH AS A FUNCTION OF STM.
C..... PART OF HELIX INTERSECTION CALCULATION.
C.....
COMMON /69/ ABCO(20,200) ,ALRAY(500) ,HEWAY(500)
1 ,CDORPS(16,50) ,E(40,200) ,EBASE(200) ,ESOR(20,200)
2 ,GARAY(500) ,RED(20,200) ,TITLF(12) ,VMAT(20)
3 ,VOL(200) ,XLAN(200) ,XRAY(500) ,YRAY(500)
4 ,ZRAY(500)
COMMON /ARRAYS/ ALK(200) ,ALPHA(40) ,BEK(200)
1 ,RETA(40) ,DEV(200) ,DK(200) ,EK(200)
2 ,FLAN(200) ,ELANAT(20) ,ENAT(20) ,ENERGY(200)
3 ,GAK(200) ,GAMMA(40) ,REF(20) ,SCHMAT(20)
4 ,SUMRAY(200) ,SUMSOR(200) ,X(40) ,XK(200)
5 ,Y(40) ,YK(200) ,Z(40) ,ZK(200)
6 ,ZLAN(20)
COMMON /IARRAY/ IFLAG(40) ,IPOINT(50)
1 ,ISMAX(50) ,ISOR(20) ,JF(200) ,JFL(200)
2 ,JL(200) ,JM(200) ,JP(200)
3 ,JRAY(200) ,JSOR(50) ,KAND(900) ,KFLAG(200)
4 ,KIP(200) ,KTP(200) ,LAMP(50) ,LANGLE(20)
5 ,LREFL(200) ,LTYPE(20) ,MAT(200) ,NFL(20)
6 ,NMAT(20) ,NMURAY(200)
COMMON /PARAMS/ AL ,ALN ,ALR ,ALRF ,RE
1 ,BEN ,BER ,DIS ,DMIN ,DZ ,EPS
2 ,GA ,GAN ,GAR ,GARF ,PER
3 ,SUMRA ,XI ,ANOW ,XOLD ,XREF ,XTRAN ,YI
4 ,YNOW ,YOLD ,YREF ,YTRAN ,ZI
5 ,ZANOW ,ZOLD ,ZREF ,ZTRAN
COMMON /IPARAM/ FMAX ,IMAX ,INOW
1 ,IOPRAY ,IOPRAYT ,IR ,IRAY ,JMAX ,JNOW ,JOLD
2 ,ORINT ,ORMAX ,ORNOW ,ORMAX ,ORMAX ,ORSOR
INTEGER F,FMAX
COMMON /HELCOM/ A,ABAR,B,B1,BIP,BP,CAP,DELTH,DELX,DELY,DELZ,DMIT,
1 ,PSIZ,REV,RND,STM,TH,TH1,SOH,S,RXS,CSTM,SNTH,RCSTM,RSNTH,BSTM,
2 ,COSTAN,THZ,DT,Z
C..... CALCULATE PARAMETERS NEEDED.
TH=STM
1 CONTINUE
DZTH=GA*STM-DELZ-CAP*TH
DXTH=AL*STM-DELX
DYTH=RE*STM-DELY
FT=RSNTH*DXTH-RCSTM*DYTH-CAP*DZTH
FP=RCSTM*DXTH+RSNTH*DYTH-CAP*DZTH
IF (FP.GT.0.) GO TO 2
FIAT=.5*(DXTH**2+DYTH**2+DZTH**2)
FTP=FIAT/FT
GO TO 3
2 CONTINUE
FTP=FT/FP
3 CONTINUE
IF (ABS(FTP).GT.1) GO TO 4
C..... CALCULATE TH
TH=TH-FTP
C..... CALCULATE COSTAN.
CALL CCSTAN
C..... SEE IF MORE ITERATIONS ARE NEEDED.
IF (ABS(FTP).GT..001) GO TO 1
DELTH=TH-THV
RETURN
4 CONTINUE
DELTH=SIGN(1.,FTP)
TH=TH-DELTH
CALL CCSTAN
RETURN
END

```

```

SUBROUTINE VSOURCE
COMMON /69/ ABCO(20,200) ,ALRAY(500) ,BEHAY(500)
1 ,COORDS(6,50) ,E(40,200) ,EBASE(200) ,ESOH(20,200)
2 ,GARAY(500) ,RED(20,200) ,TITLE(12) ,VMAT(20)
3 ,VOL(200) ,XLAM(200) ,XRAY(500) ,YHAY(500)
4 ,ZRAY(500)
COMMON /ARRAYS/ ALK(200) ,ALPHA(40) ,BEK(200)
1 ,RETA(40) ,DEV(200) ,DK(200) ,EK(200)
2 ,ELAP(200) ,ELAPAT(20) ,EMAT(20) ,ENERGY(200)
3 ,GAK(200) ,GAMMA(40) ,REF(20) ,SGRMAT(20)
4 ,SIMRAY(200) ,SUMSOR(200) ,X(40) ,XK(200)
5 ,Y(40) ,YK(200) ,Z(40) ,ZK(200)
6 ,ZLAM(20)
COMMON /ARRAY/ IFLAG(40) ,IPOINT(50) ,
,ISMAX(50) ,ISOR(20) ,JF(200) ,JFL(200) ,
,JL(200) ,JM(200) ,JN(200) ,JP(200) ,
,JRAY(200) ,JSOR(50) ,KEND(900) ,KFLAG(200) ,
,KP(200) ,KTYP(200) ,LAMP(50) ,LANGLE(20) ,
,LREFL(200) ,LTYPE(20) ,MAT(200) ,MFL(20) ,
,MAT(20) ,MURAY(200)
COMMON /PARAMS/ AL ,ALN ,ALR ,ALRF ,RE ,
,BEN ,BER ,BERF ,DIS ,DMIN ,DZ ,EPS ,
,GA ,GAN ,GAR ,GARF ,PER ,
,SUMRA ,XI ,XNOW ,XOLD ,XREF ,XTRAM ,VI ,
,YNOW ,YOLD ,YREF ,YTRAN ,ZI ,
,ZNOW ,ZOLD ,ZREF ,ZTRAN
COMMON /IPARAM/ FMAX ,IPAX ,INOW ,
,IOPRAY ,IOPRNT ,IR ,IRAY ,JMAX ,JNOW ,JOLD ,
,KINT ,KMAX ,KNOW ,MMAX ,MNOW ,MSOR
INTEGER F,FMAX
C***** CALCULATE COORDINATES OF A POINT WITHIN SOURCE LIMITS.
102 XI=COORDS(1,MNOW)*RAND(0)*COORDS(2,MNOW)
YI=COORDS(3,MNOW)*RAND(0)*COORDS(4,MNOW)
ZI=COORDS(5,MNOW)*RAND(0)*COORDS(6,MNOW)
J=JSOR(MNOW)
C***** SEE IF POINT LIES WITHIN SOURCE SEGMENT.
JNOW=I
CALL SEGMENT
IF(J-JNOW)102,103,102
C***** CALCULATE RANDOM COMPONENTS OF A VECTOR.
103 DELX=2.0*RAND(0)-1.0
DELY=2.0*RAND(0)-1.0
DELZ=2.0*RAND(0)-1.0
C***** CALCULATE LENGTH OF VECTOR.
R=SQRT(DELX**2+DELY**2+DELZ**2)
C***** IF VECTOR LENGTH GREATER THAN 1.0, GO CREATE ANOTHER VECTOR.
IF(R-1.0)104,104,103
C***** CALCULATE DIRECTIONAL COSINES OF RAY.
104 ALPHA(IPAX)=DELX/R
BETA(IPAX)=DELY/R
GAMMA(IPAX)=DELZ/R
C***** STORE COORDINATES OF RAY.
X(IPAX)=XI
Y(IPAX)=YI
Z(IPAX)=ZI
RETURN
END

```

```

SUBROUTINE WAVREF(ILL)
C.....
C..... WAVREF CALLS THE DESIRED ROUTINE TO PROVIDE REFLECTION
C..... FRACTIONS AS A FUNCTION OF WAVELENGTH AND ANGLE
C..... OF INCIDENCE.
C.....
COMMON /69/ ABCO(20,200) ALRAY(500) BERAY(500)
1 CCOORDS(6,50) E(40,200) EBASE(200) ESON(20,200)
2 GARAY(500) RED(20,200) TITLE(12) VMAT(20)
3 VOL(200) XLAM(200) XRAY(500) YRAY(500)
4 ZHAY(500)
COMMON /ARRAYS/ ALK(200) ALPHA(40) BEK(200)
1 AETA(40) DEV(200) DK(200) EK(200)
2 ELAM(200) ELAMAT(20) EMAT(20) ENRGY(200)
3 LAK(200) GAMMA(40) REF(20) SONMAT(20)
4 SUMRAY(200) SUMSOR(200) X(40) XK(200)
5 Y(40) YK(200) Z(40) ZK(200)
6 ZLAM(20)
COMMON /IARRAY/ IFLAG(40) IPOINT(50)
1 ISMAX(50) ISOR(20) JF(200) JFL(200)
2 JL(200) JM(200) JN(200) JP(200)
3 JRAY(200) JSOR(50) KBND(900) KFLAG(200)
4 KP(200) KTYPI(200) LAMP(50) LANGLE(20)
5 LREFL(200) LTYPE(20) MAT(200) MFL(20)
6 MNAT(20) MNURAY(200)
COMMON /PARAMS/ AL ALN ALR ALRF RE
1 BEN BER BERF DIS DMIN DZ EPS
2 GA GAN GAR GARF PER
3 SUMRA XL XNOWN XOLD XREF XTRAN YI
4 YNOD YOLD YREF YTRAN ZI
5 ZNOD ZOLD ZREF ZTHAN
COMMON /IPARAM/ FMAX IMAX IMON
1 IOPRAY IOPRNT IR IRAY JMAX JMON
2 KINT KMAX KNOW MMAX MNON NSOR JOLD
INTEGER F,FMAX
DIMENSION TSTG(200)
COSGAN=AL*ALN+RE*BEN+GA*GAN
60 TO(100,200,300,400,500,600,700,800,900),LL
100 CALL REFL11(XLAM,FMAX,TSTG COSGAN)
60 TO 110
200 CALL REFL12(XLAM,FMAX,TSTG COSGAN)
60 TO 110
300 CALL REFL13(XLAM,FMAX,TSTG COSGAN)
60 TO 110
400 CALL REFL14(XLAM,FMAX,TSTG COSGAN)
60 TO 110
500 CALL REFL15(XLAM,FMAX,TSTG COSGAN)
60 TO 110
600 CALL REFL16(XLAM,FMAX,TSTG COSGAN)
60 TO 110
700 CALL REFL17(XLAM,FMAX,TSTG COSGAN)
60 TO 110
800 CALL REFL18(XLAM,FMAX,TSTG COSGAN)
60 TO 110
900 CALL REFL19(XLAM,FMAX,TSTG COSGAN)
C..... STORE WAVELENGTH DEPENDENT FRACTIONS OF REFLECTION
110 DO 101 F=1,FMAX
REX(ILL,F)=TSTG(F)
101 CONTINUE
RETURN
END

```

```

SUBROUTINE ZCALC
C*****
C***** ZCALC HANDLES CASES WHERE COSTAN IS LARGE.
C***** PART OF MELIX INTERSECTION CALCULATION.
C*****
COMMON /69/ ABCD(20,200) ,ALRAY(500) ,BERAY(500)
1 ,COORDS(6,50) ,E(40,200) ,EBASE(200) ,ESUP(20,200)
2 ,FARAY(500) ,RED(20,200) ,TITLE(12) ,VMAT(20)
3 ,VOL(200) ,XLAN(200) ,XRAY(500) ,YMAT(500)
4 ,ZRAY(500)
COMMON /ARRAYS/ ALK(200) ,ALPHA(40) ,BEK(200)
1 ,BETA(40) ,DEV(200) ,DK(200) ,EK(200)
2 ,FLAN(200) ,ELAMAT(20) ,EMAT(20) ,ENERGY(200)
3 ,GAK(200) ,GAMMA(40) ,REF(20) ,SGMAT(20)
4 ,SUMRAY(200) ,SUMSGR(200) ,X(40) ,XK(200)
5 ,Y(40) ,YK(200) ,Z(40) ,ZK(200)
6 ,ZLAN(20)
COMMON /ARRAY/ IFLAG(40) ,IPOINT(50)
1 ,ISMAX(50) ,ISOR(20) ,JF(200) ,JFL(200)
2 ,JL(200) ,JM(200) ,JN(200) ,JP(200)
3 ,JRAY(200) ,JSC(50) ,KBND(500) ,KFLAG(200)
4 ,KP(200) ,KTP(200) ,LAMP(50) ,LANGLE(20)
5 ,LREFL(200) ,LTYPE(20) ,MAT(200) ,MFL(20)
6 ,MMAT(20) ,MURRAY(200)
COMMON /PARAMS/ AL ,ALN ,ALR ,ALRF ,BE ,
1 ,BER ,BERF ,UIS ,DMIN ,DZ ,EPS
2 ,GA ,GAN ,GAR ,GAMF ,PER ,
3 ,SUMRA ,XI ,XNOW ,XCLO ,XREF ,XTRAN ,YI
4 ,YNO ,YOLD ,YREF ,YTRAN ,ZI
5 ,ZNO ,ZOLD ,ZREF ,ZTRAN
COMMON /IPARAM/ FMAX ,IPAX ,INOW ,
1 ,ICORAY ,IOPRNT ,IR ,IRAY ,JMAX ,JNOW ,JOLD
2 ,KAT ,KNAX ,KNOW ,MMAX ,MAOW ,MSQR
INTEGER F,FMAX
COMMON /MELCCW/ A,ABAR,B,BI,BIP,BP,CAP,DELTH,DELX,DELY,DELZ,DMIT,
1 ,PSIZ,REV,RHO,STH,TH,THI,SGRAS,RSKS,CSTH,SATH,MCSTH,RSHTH,BSTH
COSTAN,T,Z,DTMZ
DATA(MST,RE4)
C***** CALCULATE PARAMETERS.
XII=CAP*GA/(RHO*(AL*SIN(THZ)-BE*COS(THZ)))
C***** SEE IF RAY IS EVER PARALLEL TO PLANE PERPENDICULAR TO TANGENT
IF(XII-1.0)101,102,102
101 IF(XII-COS(2.6))102,102,103
C***** RAY IS NEVER PARALLEL.
102 CONTINUE
DELTH=DTMZ*.5
TH=THZ
GO TO 109
103 ZEYA=ACOS(XII)
ZETA=SIGN(ZETA,DTMZ)
C***** CALCULATE ANGLE WHERE PLANE IS PARALLEL.
104 TH=THZ-ZETA
C***** CALCULATE DISTANCE FROM RAY TO PARALLEL PLANE.
DPERP=(DELX*RHO*SIN(TH)-DELY*RHO*COS(TH))
CAP=(DELZ-CAP*TH)/SGRAS
106 DTH=IN=DPERP*SGRAS/(RSKS*.2*ABAR)
THMAX=THZ-ZETA-DTH
C***** CALCULATE ANGLE FOR STARTING SEARCH.
TH=THZ
IF(DPERP*ZETA)108,108,107
107 TH=SIGN(2.6*ZETA)+THZ

```



```

C***** CALCULATE AMOUNT OF ANGLE FOR EACH STEP IN SEARCH.
108 DELTH=(THMAX-TH)/MSTAR
C***** CALCULATE B AND BP.
109 CALL RCALC
    BZ=B
    BZP=BP
C***** LOOK FOR INTERSECTION.
    CALL DSUB(MSTAR)
    IF (BZ.GT.ABAR) RETURN
    TH=THZ
    B=BZ
    BP=BPZ
    DELTH=DELTH
    CALL DSUB(MSTAR)
110 RETURN
END

```

SUBROUTINE ANGLE1(TM,PHI)
I=1
RETURN
END

SUBROUTINE REFL11(XLAM,FMAX,ISTG,COSGAM)

I=1
RETURN
END

SUBROUTINE INTEN1(XLAM,FMAX,ISTG,LT)

I=1
RETURN
END

SUBROUTINE ABS1(XLAM,FMAX,TSIG,M)

I=1
RETURN
END

SUBROUTINE ANGLE2(TH,PHI)
I=1
RETURN
END

SUBROUTINE REFL12(XL44,FMAX,ISTG,LOGSAP)

I=1
RETURN
END

SUBROUTINE INTEN2(ALAN,FMAX,ISTG,LT)
I=1
RETURN
END

SUBROUTINE ABS2(XLAN,FMAX,TST6,M)
I=1
RETURN
END

SUBROUTINE ANGLES(TH,PHI)

I=1
RETURN
END

SUBROUTINE REFL13(XLAN,FMAX,ISTG,COSGAN)

I=1
RETURN
END

```
ZURROUTINE INTENS(KLAN,FMAX,ISTG,LT)  
I=1  
RETURN  
END
```

SUBROUTINE ABS3(XLAN,FNAX,TSIG,M)

I=1
RETURN
END

SUBROUTINE ANGLE4 (TH, PHI)

I=1
RETURN
END

SUBROUTINE REFL14 (XLAN, FMAX, TSTG, COSGM)
I=1
RETURN
END

C137

SUBROUTINE INTENS(XLAN,FMAX,ISTG,LT)

IO;
RETURN
END

C138

SUBROUTINE ABS4 (XLAM, FMAX, TSIG, M)

I=1
RETURN
END

SUBROUTINE ANGLES(TH,PHI)

I=1
RETURN
END

SUBROUTINE REFLIS(XLAM,FMAX,TSTG,COSGM)
I=1
RETURN
END

SUBROUTINE INTENS(XLAP,FMX,TSIG,LY)
I=1
RETURN
END

SUBROUTINE ABSSEILAM,FNAX,TSTG,M)
I=1
RETURN
END

SUBROUTINE ANGLE6(TN,PHI)
I=1
RETURN
END

SUBROUTINE REFL16(XLAN,FMAX,ISTG,COSGAN)

I=1
RETURN
END

SUBROUTINE INTEND(XLAN,FRAI,TSTG,LT)

I=1

RETURN

END

SUBROUTINE ABS61ZLAN,FMAX,ISTG,M)

I=1
RETURN
END

C147

SUBROUTINE ANGLE7(TH,PHI)

1=1
RETURN
EAD

SUBROUTINE REFL17(XLAM,FMAX,ISTG,COS6AM)

I=1
RETURN
END

SUBROUTINE INTENT (PLAN, FNAX, ISTG, LT)

I=1
RETURN
END

SUBROUTINE ABSY(XL,AM,FMAX,TSTO,M)

I=1
RETURN
END

SUBROUTINE ANGLED(TN,PH)

I=1

RETURN

END

SUBROUTINE REFLIN(XLAN,FMAX,TSTG,COSGM)

I=1
RETURN
END

SUBROUTINE INTEND (ILAM, FMAX, ISTG, LT)

I=1

RETURN

END

SUBROUTINE ABS8(XLIM,FMAX,YSTG,N)

I=1
RETURN
END

SUBROUTINE ANGLE9(TH,PHI)
I=1
RETURN
END

SUBROUTINE REFL19(XLAN,FMAX,TSTG,COS6AM)

I=1
RETURN
END

SUBROUTINE INTEN9(XLAN,FMAX,ISTG,LT)
I=1
RETURN
END

SUBROUTINE ABSO(XLAN,FMAX,ISIG,M)

I=1

RETURN

END

APPENDIX D

ZAP DATA SHEETS

SYSTEMS, SCIENCE AND SOFTWARE

REFLECTION DATA SHEET FOR ZAP

1-10		11-20		21-30		31-40		41-50		51-60		61-70		71-80		
REFL. NO.	REFL. TYPE	PARAMETERS DESCRIBING REFLECTION														ID. SEQUENCE
1																
2																
3																
4																
5																
6																
7																
8																
9																
10																
11																
12																
13																
14																
15																
16																
17																
18																
19																
20																
21																
22																
23																
24																
25																
26																
27																
28																
29																
30																
31																
32																
33																
34																
35																
36																
37																
38																
39																
40																
41																
42																
43																
44																
45																
46																
47																
48																
49																
50																
51																
52																
53																
54																
55																

SYSTEMS, SCIENCE AND SOFTWARE

BOUNDARY DATA SHEET FOR ZAP

[illegible]

SYSTEMS, SCIENCE AND SOFTWARE

STRUCTURE DATA SHEET FOR ZAP

[illegible]

PREPARED FOR NRL BY
SYSTEMS, SCIENCE AND SOFTWARE

1-10						11-20						21-30						31-40						41-50						51-60						61-70						71-80					
LAMP NO.	SOURCE TYPE	INTEN ROUTINE	ANGLE ROUTINE	INTENSITY																											I.D. SEQUENCE																
1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0								
1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0								
1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0								
1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0								
1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0								
1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0								
1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0								
1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0								
1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0								
1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0								
1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0								
1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0								
1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0								
1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0								
1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0								
1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6																																

[illegible]



PREPARED FOR NRL BY

SYSTEMS, SCIENCE AND SOFTWARE

PLOT DATA SHEET FOR ZAP

[illegible]