LBL-1072

SLAC-226

SLAC-226
4/21/80

ELECTRON TRAJECTORY PROGRAM

MASTER

William B. Herrmannsfeldt

SLAC-Report-226

November 1979

Prepared for the Department of Energy
under Contract Number DE-AC03-76SF00515.

SLAC-226
UC-28
(A)

ELECTRON TRAJECTORY PROGRAM*

William B. Herrmannsfeldt

Stanford Linear Accelerator Center

Stanford University, Stanford, California 94305

November 1979

Prepared for the Department of Energy

under Contract Number DE-AC03-76SF00515.

Printed in the United States of America. Available from National Technical Information Service, U.S. Department of Commerce, 5285 Port Royal Road, Springfield, Virginia 22161. Price: Printed copy $6.50; Microfiche $3.00.

* This supersedes SLAC-51 and SLAC-166.

ABSTRACT

The SLAC Electron Trajectory Program is described and instructions and examples for users are given. The program is specifically written to compute trajectories of charged particles in electrostatic and magnetostatic focusing systems including the effects of space charge and self-magnetic fields. Starting options include Child's Law conditions on cathodes of various shapes. Either rectangular or cylindrically symmetric geometry may be used. Magnetic fields may be specified using arbitrary configurations of coils, or the output of a magnet program such as Poisson or by an externally calculated array of the axial fields.

The program is available in IBM FORTRAN but can be easily converted for use on other brands of hardware. The program is intended to be used with a plotter whose interface the user must provide.

DISCLAIMER

This book was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

TABLE OF CONTENTS

LIST OF FIGURES

I. INTRODUCTION

This report is intended as a user's reference manual for the SLAC Electron Trajectory Program. It contains all the currently relevant material from the earlier publications about this program which were SLAC-51 and SLAC-166. In addition, I have included specific instructions for using a number of the special features which have been added to the program. These features have usually been incorporated as a direct result of the needs of some particular user and I wish to take this opportunity to express thanks to everyone who has at some time or other suggested improvements to the program. I think we have all benefited by this open process and it is for the purpose of making all these features better available that this report is being prepared. The most recent version of the program has benefited greatly from some careful program house cleaning, including a complete revision of the plotting sections, making the problem of interfacing with other plotter systems much easier. It is a pleasure to acknowledge the contributions of Glen Herrmannsfeldt in making these improvements.

II. APPLICATION

The SLAC Electron Optics Program is specifically written to calculate electron trajectories in electrostatic and magnetostatic fields. Poisson's equation is solved by finite difference equations using boundary conditions defined by specifying the type and position of the boundary. Electric fields are determined by differentiating the potential distribution. The electron trajectory equations are fully relativistic and account for all possible electric and magnetic field components. Space charge forces are realized through appropriate deposition of charge on one cycle followed by another solution of Poisson's equation which is in turn followed by another cycle of trajectory calculations.

The program may be used in either rectangular or cylindrical coordinates. A special option allows space charge forces in a cylindrical beam to be calculated in a rectangularly symmetric array of electric and magnetic fields. Magnetic fields are read in either as axial strengths or as arrays of coils with specified coordinates and currents. The preferred technique of defining the magnetic field is to calculate the axial field from an arbitrary configuration of solenoids. Alternatively, the program accepts the output data from a magnet design program, which can include the effects of saturable iron. In cylindrical coordinates, the magnetic fields are axially symmetric. Off-axis field components are calculated by a sixth-order expansion of the radial coordinate. In rectangular coordinates the external field is assumed to be normal to the plane of the problem, which is assumed to be the median plane. Off-

median plane components are calculated by expansion of the perpendicular coordinate.

Electron trajectories may be started by three methods:

1. Child's law for spherical geometry based on Pierce geometry.
2. Child's law for generalized cathodes including effects of holes, shadow grids and other irregularities.
3. Direct input of the starting conditions, including the output from previously run problems.

The program is designed to yield a combination of printed and plotted output. Printed output includes all input data, maps of the potential fields, starting conditions for each cycle, and final conditions for each cycle. Plotted output is made for the trajectory calculations and for equipotential lines. Plotted output may be obtained for selected cycles always including the last cycle.

III. IMPLEMENTATION

The program is written in IBM-style FORTRAN IV. Reasonable application requires about 400 K bytes of total storage. Running times vary greatly with the problem and the computer. However a "typical" problem run on an IBM 370-168 takes about 2 minutes.

The program is designed for use with a computer controlled plotter. Data needed for plotting are placed on an external storage device (disk) from which they are called by a plotter interface program. Such a program calling standard CALCOMP routines is available and can be used as a model for users with other plotter systems.

IV. GENERAL DESCRIPTION

Starting with the input boundary description, the program first solves Laplace's equation (i.e., Poisson's equation without space charge). The result of this calculation, together with all the boundary information is then printed.

Next, the first iteration of electron trajectories is started. These are initiated by one of four schemes: (1) "GENERAL" cathode in which electrons are started assuming Child's law holds near a surface designated as the cathode; (2) "SPHERE" for a spherical cathode (cylindrical in rectangular coordinates) in which the electrons are assumed to be emitted at right angles to the surface defined by a radius of curvature and a radial limit. Child's law for space charge limited current is again used. (3) "CARDS" in which the specific starting conditions for each ray are specified. (4) "GENCARD" which combines the versatility of "CARDS" with the assumptions of Child's law from "GENERAL."

On the first iteration cycle, space charge forces are calculated from the assumption of paraxial flow. As the rays are traced through the program, space charge is computed and stored in a separate array. After all the electron trajectories have been calculated, the program begins the second cycle by solving Poisson's equation with the space charge from the first iteration. For problems meeting the paraxial assumptions, especially if relativistic electron beams are involved, this one cycle may be sufficient to solve the entire problem.

Subsequent iteration cycles (as many as are requested) follow the above pattern. The Child's law calculations for the starting conditions are remade for every cycle. Perveance converges through the iterative process by averaging the perveance used for the previous cycle with the perveance calculated directly from the solution of Poisson's equation.

An additional starting option is "LAPLACE" intended for any application of Laplace's equation not involving electron ray tracing. In this case the number of cycles is used simply to improve the accuracy of the solution to Laplace's equation. The "LAPLACE" option includes a provision for inputting arbitrary data in the "space charge" array.

The program always operates in two dimensions; either R and Z in cylindrical coordinates or Y and X in rectangular coordinates. The rectangular coordinate output retains the R and Z labels however. Electron orbits are calculated through azimuthal changes (labeled "PHI") referenced to the Z axis. In rectangular coordinates, PHI is actually the third Cartesian coordinate.

Magnetic fields, except for the self-magnetic field of a beam, are input directly in one of three ways: (1) by specifying the field along the Z-axis, (2) by specifying a set of coils (giving position, radius and current), or (3) by using the vector potential output from a magnet program. In cylindrical coordinates, the field is interpreted as an axial magnetic field with radial terms as required by Maxwell's equations. In rectangular coordinates the field is interpreted as going in the PHI direction, i.e., at right angles to the plane of the problem. The rectangular coordinate field is assumed to extend to infinity in Y (R) and

the PHI = 0 plane (the plane of the problem) is assumed to be the median plane. The B_x (B_z) terms are calculated for PHI ≠ 0 from Maxwell's equations.

Self-magnetic fields are calculated for both coordinate systems from the current in the rays on the present cycle. It is generally assumed that the rays are sequentially numbered from the axis outwards. The self-magnetic field calculation assumes all the current from the previous rays lies on the axis in an infinitely long conductor. If the ray being calculated crosses the last preceding ray, then the current from that ray is dropped. However, if the ray continues to cross other rays, then the current from those rays is only dropped if the ray goes below the minimum radius of a previous ray. If several rays cross the axis, the results are apt to be somewhat incorrect, depending of course, on how significant the self-magnetic field is. Note that if the self-magnetic field is very significant, then almost by definition, one is dealing with a very intense relativistic beam. This problem is generally better suited to the paraxial ray approach, as solved in the first cycle, or to a program such as EBQ (by Art Paul of LBL) which handles the cancellation of space charge by self-magnetic field directly, rather than by the off-setting effects of two large terms.

In rectangular coordinates, the self-magnetic field assumes symmetry about the y = 0 (R = 0) plane. If this is not correct, or if for other reasons it is desired to turn off the self-magnetic field, then an external field of strength zero can be specified. In any case, in rectangular coordinates, the self-magnetic field functions only if there is no external field.

A single variable controls plotting. If this variable, MI, is set to zero to reject all plotting, then on the first and last cycles every tenth point that would have been plotted is printed so that it may be hand plotted. Normally at least the last cycle is plotted. The first cycle may also be plotted or one may even plot every cycle. All plots may include equipotential plots, either separate or overlaid with the trajectory plots. If there is an external magnetic field, then this field is also plotted, overlaid on the trajectory plots. Finally, there are a pair of simple plots; current density vs. radius and alpha vs. radius. (Alpha = $\tan^{-1}$ dR/dZ).

V. POISSON EQUATION SOLVER

A. General Description

The program contains a subroutine which reads in data cards describing the boundary conditions and calculates the coefficients of the finite difference equations for each mesh point within the problem. Other subroutines are made to proceed to generate the solution to Poisson's equation which match those boundary conditions. The solution is found in terms of a set of points which form a mesh of identical squares. It is recognized that a provision for a rectangular mesh (i.e., different horizontal and vertical spacing) would improve the utility of the program and it is planned to incorporate this feature as soon as possible. The potential is calculated for each intersection of the mesh. Figure 1 shows a small section of the mesh.

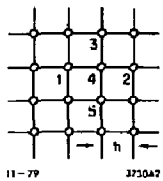


Fig. 1. Section of mesh for solution of Poisson's equation.

In rectangular coordinates, the finite difference form of Poisson's equation is

$$V_1 + V_2 + V_3 + V_5 - 4V_4 = (R.H.) \qquad (1)$$

where the V's refer to the numbered points in Fig. 1 and R.H. is the value of the right-hand side of Poisson's equation at point 4 when written in the form

$$\nabla^2 V = (R.H.) \qquad (2)$$

All equations use the mesh space, h, as the basic unit, so h does not appear explicitly.

For problems with cylindrical symmetry, the finite difference equation becomes

$$RV_1 + RV_2 + (R + 1/2)V_3 + (R - 1/2)V_5 - 4RV_4 = R \times (R.H.) \qquad (3)$$

where R is the distance in mesh units from the axis of symmetry to the point at 4.

A number of references[1-6] give the derivation of these equations and the special equations at boundaries. Three types of boundaries are of interest. A Dirichlet boundary is that boundary on which the potential is known. In an electrostatic problem, this would be an electrode fixed at a given potential. An ordinary Neumann boundary is one which lies coincident with the mesh and on which the normal derivative of the potential is known. In practice, the only value of the normal derivative that is ever known is zero. Thus, for example, the axis of symmetry of a cylindrically symmetric device has the normal derivative equal to zero and is a Neumann boundary.

However, the axis of a cylindrical symmetry problem is a special case for which the difference equation is

$$V_1 + V_2 + 4V_3 - 6V_4 = (R.H.) \qquad (4)$$

The difference equation for ordinary Neumann boundaries parallel to either axis can be derived from Eqs. (1), (3) or (4) by setting the potentials which straddle the boundary equal to each other. Thus a vertical Neumann boundary in cylindrical coordinates has the form

$$2R\,V_{(1,2)} + (R + 1/2)V_3 + (R - 1/2)V_5 - 4RV_4 = R \times (R.H.) \text{ (off-axis)} \qquad (5)$$

where the subscript 1 or 2 applies to the point inside the problem.

The third type of boundary is the general Neumann boundary, i.e., one which does not lie along a mesh line. It is always assumed that the normal derivative is zero. The program has a provision for overriding the internally computed difference coefficients and it is feasible t-

hand calculate difference coefficients for a general Neumann boundary. However, in practical applications to electron optics problems, it is almost never necessary to go to such extremes.

A special case of general Neumann boundary which can be handled easily is the 45° Neumann boundary. All that is required is to specify each successive point using the ordinary Neumann condition for both coordinates; i.e., both DELTAR and DELTAZ = 0. A tilted boundary that is sufficiently far from the area of most interest can frequently be adequately approximated by a combination of normal and 45° Neumann boundaries.

B. Problem Input

In this section the rules for problem input will be described using an actual example and following through the process card by card. The new user is urged to read this section carefully while the old user or reader trying to gain an overall familiarity with the program may well skip this section. In this section especially, no attempt will be made to be concise.

Condensed instructions for problem input are printed at the head of the source listing and are intended to be up-to-date. A copy of the current version of these instructions in printed in Appendix II. The reader should follow the instructions which are relevant to this discussion while studying the example.

Except for the TITLE, boundary input, and ray starting cards, all input to the program is by means of the NAMELIST option by which certain variables are defined at the place in which the program expects them.

The definitions are by means of short defining statements, e.g., RLIM = 50. A given set of these statements may be placed on one card, but the number of data cards used is unimportant. Each set of inputs is preceded by a designator, e.g., &INPUT1, which must begin in column 2. Never use column 1 of any NAMELIST card. The NAMELIST block is closed by an &END entry.

Preparation for running a problem consists of making a suitable scale drawing on graph paper. Figure 2 shows the region between cathode and grid for the SLAC injection gun. Figure 3 is the line-by-line listing of the input data.

1. Title and Potential Cards

(Title) The first card of the data set is the title card. The contents of this card will appear at various points in the printed output and as the title for the plots.

The second card is &INPUT1, starting in column 2.

The following remarks about array limits apply specifically to the current version of the program. It is suggested that most problems should use about 5000 mesh points although there are occasions when much smaller, or somewhat larger, numbers of mesh points are useful.

The third card is the potential card. It contains the basic information for setting up the program.

(RLIM) RLIM is the maximum size of the problem area in the radial direction. RLIM can be made larger than necessary if it is desired to affect the way plots are scaled.

RLIM is a positive integer; the present limit is 100.

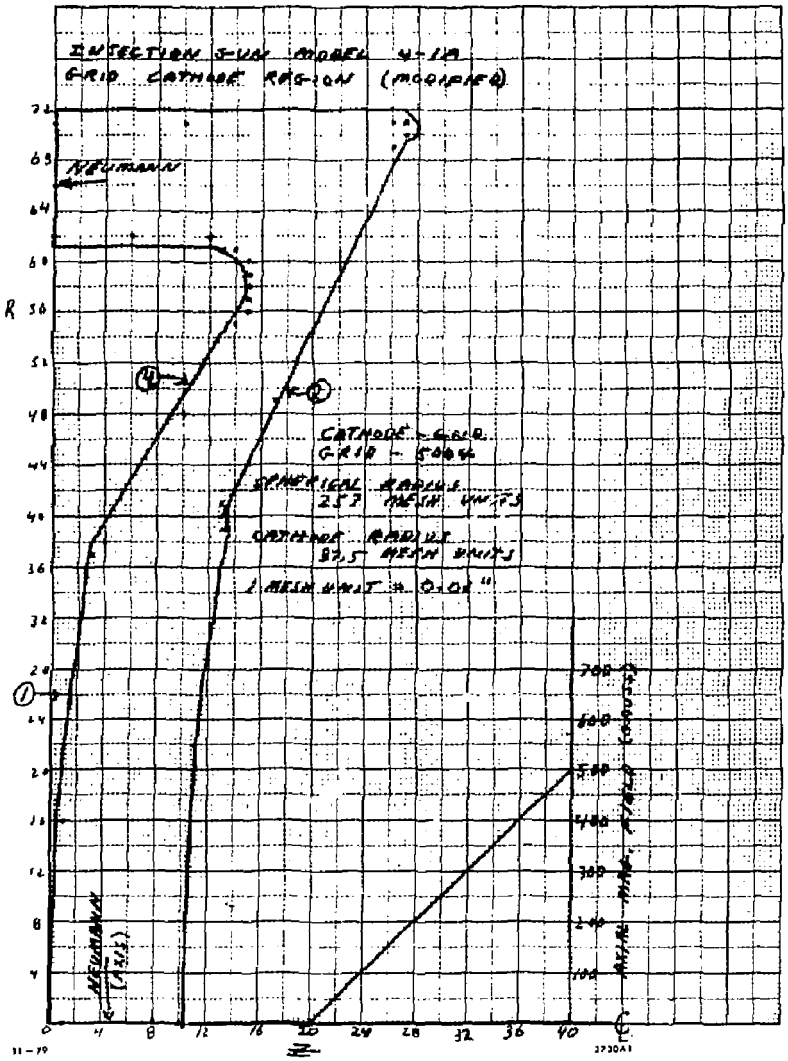


Fig. 2. Example of preparation for a problem.

```
  INJECTION GUN MODEL  4-IA GRID-CHARGE ENCODE REGION (NPH) MCG. 11-20-67  MI=0,JPC=0
&INPUT1
 RLIM=72,ZLIM=40,PCTM=4,PCT=0.0,5000.0,0.0,0.0,MI=0,MAGSEG=1,
&END
&INPUT2
  Z1=20,Z2=40,Z3=20,PC=0.0,25.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,
&END
    1     0     1          0.0       -0.99
    1    16     1          2.0       -0.4
    1    37     3          0.00      -0.1
    4    38     4          2.0       -1.0
    4    48    10          2.0       -0.8
    4    55    14          0.00      -0.6
    4    56    15          2.0       -1.0
    4    57    15          2.0       -0.4
    4    58    15          2.0       -0.3
    4    59    15          2.0       -0.4
    4    60    15          2.0       -1.0
    4    61    14         -0.99       2.0
    4    61    13         -0.7       -0.8
    4    62    12         -0.7        2.0
    4    62     8         -0.7        2.0
    4    62     0         -0.7        0.0
    0    66     0          2.0        0.0
    2    71     0          0.99       0.0
    2    71    10          0.99       2.0
    2    71    26          0.00       2.0
    2    71    27          0.99       0.99
    2    70    27         -0.2        0.00
    2    69    26          2.0        0.8
    2    46    17         -0.3        0.2
    2    41    13          2.0        0.8
    2    40    13          2.0        0.4
    2    39    13          2.0        0.3
    2    22    11          2.0        0.2
    2     0    10          0.0        0.3
    0     0     8          0.0        2.0
    0     0     7          0.0        2.0
 888
&INPUT5
 IZ1=1, IZS=2, IZS=10,  RAD=257, RMAX=37.5, LIMITN=0.01, SPC=0.0,
&END
/*
//
```

Fig. 3. FORTRAN data prepared for the problem shown in Fig. 2.

(ZLIM) ZLIM is the maximum size of the problem in the axial direction. A larger than necessary value of ZLIM may also affect the way the plots are scaled. If an attempt is made to create a boundary which exceeds the limits RLIM by ZLIM, or goes negative, error messages are printed and the program will not attempt the solution of Poisson's equation.

ZLIM is a positive integer; the present limit is 300.

Note that although the problem area is (RLIM + 1) × (ZLIM + 1) mesh points the actual requirement is for (RLIM + 1) × (ZLIM + 2). (An extra column is required as a buffer.) The present limit for the total area is 9001 mesh points.

(POTN) POTN is the number of potentials which are to be read in. There may be reasons to assign different numbers to parts of surfaces which are at the same potential. Normally the cathode will be potential number 1 and the anode will be number 2. Usually the grid, if any, will be number 3. A focus electrode, even if at cathode potential, should be assigned a different number to enable the general cathode starting method to be applied. The present limit for POTN is 101.

POTN is a positive integer for cylindrical symmetry.

POTN is a negative integer for rectangular symmetry.

RECTANGULAR COORDINATES. The code to the program to switch to rectangular coordinates is the sign of POTN. If POTN is negative, the program assumes rectangular symmetry and a message: ***RECTANGULAR COORDINATES, PHI IS TRANSVERSE appears immediately after the list of potentials.

POT(I) The next numbers are the elements of the array of potentials. They are read in in order from 1 to POTN. Potentials are carried in double precision which means that up to 15 significant decimal figures can be used. Examples of valid ways of punching 250 volts are as follows: 250., 250, 2.5E2, 2500E-1, 250.000. For NAMELIST, the list need consist only of POT = (string of potentials separated by commas).

POT(I) is an element of an array of floating point numbers.

Negative potentials are indicated by a minus sign, e.g., -250. Negative potentials are permitted but it is preferable to avoid using them. Since a constant can always be added to all potentials, it is possible to make the most negative potential zero. The reason for avoiding negative numbers is that space charge is negative and some diagnostics of the output are simplified if there are no negative potentials. On the other hand, certain problems have a symmetry that can be quickly examined if a symmetry plane or surface is made to be zero by having equal + and - potentials. Then negative potentials are certainly desirable.

Note that it is acceptable to include potentials corresponding to potential numbers which are not used by the problem. One reason for doing this is to get a desired set of equipotential lines on the plotter output.

The program is intended to be run using engineering units. Thus potentials are in volts and magnetic fields are in gauss. If a problem does not use magnetic fields or relativistic energies, there is no reason not to scale the potentials. The perveance and running time will

not be affected. However, there is also nothing gained by scaling. Of course, when a problem has been run at one set of potentials, all the scaling rules of electron optics may be applied to avoid the cost of running the problem again.

(MI) MI is a code number which determines the selection of plots.

If MI = 0 there are no plots generated. However, every tenth point of the trajectories is printed for the first and last cycles.

The following table, reprinted from the condensed instructions, shows the available options for MI

Cycle for which electron trajectories are plotted:	Initial & Final	All	Final
Plots with equipotential lines superimposed on trajectories:	1	2	3
Separate plots of equipotential lines:	4	5	6
No equipotential lines:	7	8	9

MI is a positive integer or zero. If MI is negative it is interpreted as a deliberate boundary error for help in debugging boundaries.

TYME = X TYME = 5 MAX. PROBLEM RUN TIME (MIN.)

TYME is used to make an internal check of how much time is being used to guard against running out of computer time, as specified on a JOB card, just before printing and plotting the results. TYME uses special machine language subroutines to measure actual use of CPU time which is the parameter used to determine JOB time and charges in a multitask environment. This avoids gross variations in time due to the presence of other jobs on the system. The subroutine must be supplied by non-

Stanford users to suit their hardware or, alternatively, dummy subroutines may be used to defeat this feature. The program only tests for TYME once each cycle and determines that there is adequate time left to do the extra plotting, etc., that is involved in the last cycle, based on the previous cycle time. When time appears limited, the program cuts out intermediate cycles, with a note that:

THERE IS NOT ENOUGH TIME TO DO THE SPECIFIED NUMBER OF CYCLES

TYME does not need to correspond exactly to the job card. The user may wish to modify the value according to his experience, or disable TYME entirely by setting it much larger than his JOB card time.

LSTPOT = 1, 2 or 3 causes the program to print a table of the potentials of all the mesh points. This is the most useful diagnostic available *for the Poisson solution and, when studied together with the equi*potential plot, can show quite subtle errors. The default value: LSTPOT = 0, suppresses this output and thus saves quite a lot of printing if the same or a very similar boundary is run many times. The choices for LSTPOT cause the printing of the first (LAPLACE) solution (LSTPOT = 1), or the last solution (LSTPOT = 2), or the solutions from both the first and last cycles (LSTPOT = 3).

The parameter MAGSEG controls two of the four possible ways of reading in magnetic fields. The example case will be explained in the next paragraph.

2. Magnetic Field Data

Electron optics calculations include the effects of any external magnetic fields that may be present. The input methods for magnetic fields have been greatly revised and will be treated later in a special

section. If there are external magnetic fields then the input could occur at this point. The parameter MAGSEG signals that segments of magnetic field data will follow; one segment for MAGSEG = 1, etc. The namelist &INPUT2 is called MAGSEG times to read in segments, which may be anything from constants to sixth order polynomial functions of Z.

Please note that this discussion is only included here to explain the &INPUT2 namelist data card in Fig. 3. It is grossly incomplete an an explanation of the magnetic field situation which will be found in an expanded form in Section VI-D.

The example problem contains a meaningless magnetic field inserted only as an example. The magnetic field plotted on the right-hand side of Fig. 2 shows an axial field starting at Z = 20 going from 0 to 500 gauss in 20 mesh units. A sixth order expression is used by the program to fit the fields on any segment of the axis. The data on the card are Z1 and Z2, the limits of the range of the segment being described; Z3, the origin for the segment being described, and seven coefficients for the equation:

$$BZA(Z) = \Sigma BC(n)\ (Z - Z3)^{n-1}$$
$$n = 1 \text{ to } 7 \qquad (6)$$

Z1, Z2 and Z3 are integers.

BC(n) is an element of a seven member real array.

The parameters Z1, Z2 andZ3 are read in by simple statements (Z2 = 100, etc.) and are defaulted to 0, ZLIM and 0, respectively. The coefficients, BC, are read in as an array by BC = (string of coefficients separated by commas).

A second option (MACSET = -1) allows the axial array to be read in directly. See Section VI-D for a description of this feature.

3. Boundary Input

The main thing for a user of the program to learn is the technique and conventions used to input boundary data. Since the primary application for the program is for electrostatic optics, the terminology used will be appropriate to that class of problem. Each line on the table in Fig. 3 represents one data card for the problem in Fig. 2. The input uses FORTRAN fixed field input; three integers followed by two floating point numbers. The fixed field format requires one card for each point.

The chief feature of the input routines is the ability to fill in for segments of the problem that the programmer skips. This saves a great deal of labor since a typical problem which uses perhaps 300 boundary points may be specified with about 50 cards. This technique will be called "fitting" in the description for the ability of the program to fit a curve to three specified data points.

Two types of boundaries are used: Dirichlet boundaries are those on which the potential is known. Neumann boundaries are those on which the normal derivative of the potential is known.

Dirichlet boundaries are used to represent metal surfaces. Neumann boundaries represent gaps between surfaces and must be chosen so that the normal component of the field is zero since that is the only value that is ever known in practice. Thus the cathode is a Dirichlet boundary and the axis is a Neumann boundary in a typical example. Neumann boundaries can meet at a corner.

For electrostatic problems it has been found satisfactory to restrict Neumann boundaries to lie along mesh lines. Dirichlet boundaries may have any shape desired although the mesh spacing limits the resolution of the smallest details which can be effectively used. Slanted Neumann boundaries are possible however, and the input technique will be described later in this section.

A boundary point is defined as any mesh point less than one mesh unit from the boundary of the problem, but always within the boundary. The points on a Neumann boundary are always boundary points. The points on a Dirichlet boundary are never boundary points. This difference, which is inherent in the formulation and not just a program convention, gives rise to a code to determine which type boundary is being specified. Thus, if the distance from a point to a boundary in either the R or Z direction is zero, then that boundary is defined as a Neumann boundary.

1. Potential number, integer, corresponds to the surface numbers denoting elements of the array POT (n) described earlier.
2. R, integer, the value of the radial coordinate of the mesh at the boundary point.
3. Z, integer, the value of the axial coordinate of the mesh at the boundary point.
4. DELTAR, floating point, the distance from the mesh point to the boundary in the radial direction. DELTAR is negative if the boundary intersects the radial line at a point in the minus direction from the mesh point. If the intersection is greater than one mesh unit from the boundary point then the intersection is not significant. Any number greater than 1.0 could be used but typically the distance is specified as 2.0 if it is greater than 1.0.

5. DELTAZ, floating point, the distance from the mesh point to the boundary in the Z or axial direction. The same rules as for DELTAR, above, apply.

In the case of a point on a Neumann boundary, the potential number is not significant. If the point is simultaneously within one mesh unit of a Dirichlet boundary, then the potential number is the number for that surface. Otherwise it is customary to punch a zero for the potential number. It is important to realize that a zero for the potential number is not the code number for a Neumann boundary. Repeating, the code for a Neumann boundary is a zero for DELTAR if the boundary is parallel to the axis. If the boundary is a radial plane, then the code is DELTAZ = 0.

A mesh point cannot simultaneously be a boundary point for two Dirichlet surfaces at different potentials. This is not usually a problem for the programmer. However, there can be situations when it is necessary to make some adjustment in the problem to avoid a situation in which, either DELTAR or DELTAZ should have two values, or in which DELTAR and DELTAZ refer to two different surfaces in which neither is a Neumann boundary.

Note that this also means that a single point cannot be a complete row or a complete column. A column must have a top point and a bottom point, each of which has a DELTAR between -1.0 and +1.0. Since one point cannot have both of these, one point cannot be a column. The same

thing applies to rows. However, the program applies tests for the columns only.

Boundary points must be read in in sequential order. Adjacent points must be within one mesh unit in both R and Z. If a boundary point is not within one mesh unit of the previous point, then a special procedure starts with the purpose of determining and filling in the missing point or points. This procedure, referred to as "fitting," fits a second degree equation to the three boundary points defined by the two cards referred to above and the immediately next card. The equation is either of the forms

$$R = AZ^2 + BZ + C \qquad \text{SLOPE} \leq 1.0 \qquad (7)$$

or

$$Z = A'R^2 + B'R + C' \qquad \text{SLOPE} > 1.0 \qquad (8)$$

depending on whether SLOPE = ABS[(22 + 1) A + B] is less than or greater than unity.

Use of fitting demands some care and understanding on the part of the user. It should not be used on curves with more than one curvature or on curves that go through too large an angle, i.e., never more than 45^{o}. It is more useful on long straight or slightly curving segments.

Three points always define a segment and if the third point is missing or goes around a corner to another segment, the result will be chaotic.

The programmer must realize that each boundary point may actually define two points on the surface at the intersections in the R and Z directions. If both points do not lie on the same segment, the results

are unpredictable. This is a common difficulty at inside corners of Dirichlet boundaries. The solution is to provide a data card for one extra point in each direction from the corner.

In the special, but quite common, case in which one of the surfaces at a corner is a Neumann boundary, the program takes account of the corner ambiguity and no extra cards are required.

The boundary output listing shown on Fig. 4 will now be examined in detail as an example. Notice that there are seven columns; POINT, CARD, POTENTIAL, R. Z, DELTAR, DELTAZ. The POINT column is just the point number. The CARD column contains a sequential number if such a card exists; otherwise it contains a zero. The remaining columns contain the identical data as are found on the cards, or the data resulting from fitting. It is useful to compare Figs. 2, 3 and 4 as the following discussion progresses.

Card number one: Potential number one, (cathode), R = 0, Z = 1, (this is the usual starting place), DELTAR = 0.0, (code for Neumann boundary along the axis), DELTAZ = -0.99,(-1.0 could have been used but 1.0 for the DELTA terms can result in some confusion for the fitting routine). The point R = 0, Z = 0 could also have been used but it is risky to use -0.01, for example, for DELTAZ because the curve could try to cross the Z = 0 line before R = 1, thus resulting in a point with two values of DELTAR, 0.0 and some positive fraction. This would also have the result of adding another column to the problem *without* increasing the resolution or the actual area, thus resulting in a fractional slow down. Thus 0.99 or 0.999 is frequently used for DELTAR or DELTAZ.

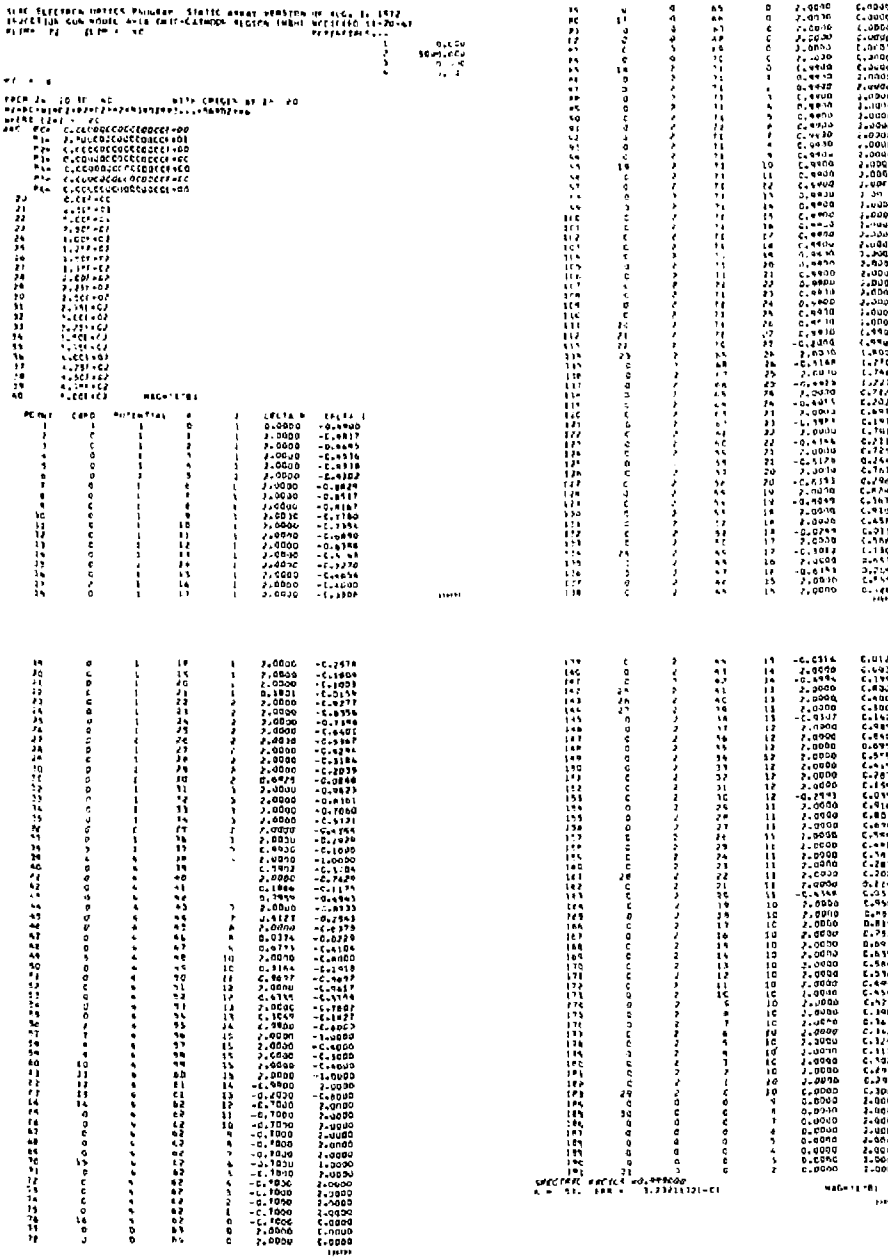


Fig. 4. Output listing of boundary data for the problem of Fig. 2.

Card number two: POT = 1, R = 16, Z = 1, DELTAR = 2.0, DELTAZ = 0.4. Since R = 16 is more than one unit from R = 0 on card one, the automatic fitting routine will be called. It will read the next card which must also be on the cathode surface. The DELTAR = 2.0 indicates that the boundary does not cross within one mesh unit in the R direction.

Card number three: POT = 1, R = 37, Z = 3, DELTAR = 0.99, DELTAZ = -0.1. Both DELTAR and DELTAZ refer to the same curve segment, so there is no ambiguity for the fitting. This is the third card for the fitting set for the cathode. The coordinates of the points through which the curve will fit are: (r = 0, z = 0.01), (r = 16.0, z = 0.6) and (r = 37.99, z = 3.0). It will use Eq. (3) rather than Eq. (2) because the absolute value of the slope is greater than one.

Card number four: POT = 4, R = 38, Z = 4, DELTAR = 2.0, DELTAZ = -1.0. POT = 4 is used to permit the focus electrode, which this surface is, to be distinguished from the cathode. The -1.0 for DELTAZ is inadvisable but works on the first point of the set of three. No fitting since R and Z are 1 mesh unit from those on card 3.

Card number five: POT = 4, R = 48, Z = 10, DELTAR = 20, DELTAZ = -0.8. This card causes the automatic fitting procedure to be called.

Card number six: POT = 4, R = 55, Z = 15, DELTAR = 0.99, DELTAZ = -0.6. This is the third card of the set and fits the straight section of the focus electrode.

The next several cards define the boundary around the point on the focus electrode. The logic should be obvious by inspection. Fitting is used for the top of the focus electrode.

Card number sixteen: POT = 4, R = 62, Z = 0, DELTAR = -0.7, DELTAZ = 0.0. This card is interesting because it defines the end of the segment to be fit along the top of the focus electrode and the beginning of the Neumann segment along Z = 0. Because of the Neumann condition (DELTAZ = 0.4) the program recognizes the corner condition and fits to the point (r = 61.3, z = 0.0).

Card number seventeen: POT = 0, R = 66, Z = 0, DELTAR = 2.0, DELTAZ = 0.0. This is a case where one might forget to skip a point and make R = 63 ... don't. Also note especially the DELTAR = 2.0 ... there is no surface in the R direction for more than one mesh unit, even though the point lies right on the Neumann boundary.

Card number eighteen: POT = 2, R = 71, Z = 0, DELTAR = 0.99, DELTAZ = 0.0. Potential 2 is for the anode, which is the role played by the gun grid in this example. The 0.0 for DELTAZ signifies the vertical Neumann boundary. Note that this card is used to begin the next fitting segment.

Card number twenty: POT = 2, R = 71, Z = 27, DELTAR = 0.99, DELTAZ = 2.0. This is an "extra" card inserted to avoid the corner ambiguity which would occur if the fitting program had to use the next card which points to two different line segments of the same surface.

Cards number twenty-one and twenth-two: POT = 2, R = 71 and R = 70, Z = 27, DELTAR = 0.99 and 0.2, and DELTAZ = 0.99. These two cards form a short column to avoid a column of length one at the corner. Clearly they do not agree with the design surface, but the location is such that the discrepancy cannot affect the solution.

The last three boundary cards define the Neumann segment on the axis. Note that the last card, POT = 0, R = 0, Z = 2, DELTAR = 0.0, DELTAZ = 2.0, specifies the point immediately adjacent to the first point, thus completely defining the boundary. The boundary must be completed in this way without ever repeating a boundary point.

The next card, with 888 in the POT field, or any other potential number greater than POTN, terminates the boundary input. The next step in the program is to calculate the difference equations and to perform some checks on the boundary data.

4. Special Boundary Conditions

A curved or slanted Neumann boundary, except for 45°, requires the general Neumann conditions. The special case of a 45° Neumann boundary is correctly described in both DELTAR = 0 and DELTAZ = 0. General Neumann and other boundary conditions such as dielectric surfaces, may be put in as calculated values by overwriting the difference equations calculated by the program. The normal ending to the boundary data is by a potential number greater than POTN. If 999 is used, the program will commence reading cards containing R and Z; the coordinates of an existing boundary point, and D1, D2, D3 and D5; the four coefficients of the difference equation for the point (R,Z).

R and Z are integers locating an existing boundary point. D1, D2, D3 and D5 are the real positive coefficients of the difference equation at (R,Z).

Any number of such cards may be used in any sequence. An R value greater than RLIM terminates this input.

Dielectric materials may be simulated by special boundary values at the dielectric surface. The rules for this are summarized in the condensed instructions and will be explained in Section VI.1.

5. Boundary Diagnostics

If the input data are acceptable, the next message printed on the output is: SPECTRAL RADIUS=0.995. The spectral radius is a constant used by the program for the convergence of the solution of Poisson's equation.

BOUNDARY ERROR IN COLUMN XX

If this message appears somewhere in the middle of the listing of boundary data, it is a signal that the boundary data have exceeded the limits of the problem, $0 \leq R \leq$ RLIM and $0 \leq Z \leq$ ZLIM, or that the boundary data have exceeded the maximum number allowed which is 901. Thus, this message appears if the boundary calculation goes into a loop. Loops usually result from an error in boundary fitting as might be caused by omitting one of the three points of a line segment. Normally the program will attempt to pick up the boundary computation and complete the listing. However, the problem will not attempt to run and there may be other errors caused by the program in trying to interpret the rest of the boundary.

BOUNDARY ERROR IN COLUMN XX

If this message appears at the end of the boundary listing it indicates that the program checks have found an error. The program checks are based on the requirement that each column must have a top and a bottom. Since there can be more than one segment to a column, the requirement translates to mean that there must be an even number of ends for

each value of Z. An end is defined by a DELTAR value between +1 and -1. Thus the programmer need only determine why there are not an even number of such points for the indicated column.

Note that there are similar checks which could be made but aren't. Each row must have two ends also, but no such check is included. Also obviously a bottom end must have DELTAR between 0.0 and -1.0, i.e., not greater than 0.0. This and similar boundary mistakes are left to the programmer's care to prevent or correct.

BOUNDARY ERROR OR MI NEGATIVE

If this message appears at the end of the boundary listing the programmer must check for messages of the previous two types. If there are none, and he has set MI negative, then the boundary data have passed the program checks. It is worthwhile for the programmer to look at all the output carefully to catch other boundary errors. The programmer should also always endeavor to get at least one plot including equipotential lines of any new geometry. Unsuspected errors frequently become glaringly obvious on examination of a plot. The optional printout of the table of potentials caused by LSTPOT > 1 should always be used for a new or revised boundary configuration.

C. Poisson's Equation

After reading the boundary input, and before reading the starting conditions, the program makes the first solution of Poisson's equation (actually Laplace's equation at this point since there is no space charge, hence right-hand side (R.H.) equals zero). The description of the input data for the example will be interrupted here for a brief description of the mechanics of the solution of Poisson's equation.

The program solves the complete set of equations for one column at a time. Mathematically, a matrix for a column consists of a tridiagonal matrix which must be solved (inverted) to find values for the potentials of each of the points in one column. To do this, the adjacent columns are assumed to contain "known" values, and the end points are also "knowns." That is, either the value is known or, in the case of a Neumann boundary, the adjacent point is assumed to be the same as the point being solved since the derivative is zero. The relaxation method is known as the "semi-iterative Chebyshev" method and is described by Varga.[4]

Each column consists of two or more points, with upper and lower end points being boundary points for which $-1.0 \leq$ DELTAR ≤ 1.0. Thus each column has at the top and bottom a condition, either Neumann or Dirichlet, that permits the program to write a set of n equations in n unknowns for that column. A column of the problem area defined simply by the value of Z, may have more than one segment which must each meet the above definition of a "column." Each such column must have its proper ends. In the example problem, there are two columns for each value of Z up to and including Z = 14.

When a column is solved, the adjacent columns are considered fixed. Alternate columns are solved so that on two passes first the odd numbered columns and then the even numbered columns are solved. After 50 iterations, or less if the error criterion is satisfied, the calculation is stopped and a message is printed:

N = 51, ERR = X.XXE - XX

This is the signal that after 50 iterations (the counter is already set to 51) the maximum error is expected to be ERR in volts. The actual test is on the largest single change in the iteration, but the value printed takes into consideration the dimensions of the problem. The convergence criterion can be adjusted by using the parameter ERROR (see VI.A.(4)). It is automatically tightened by a factor of ten for the final cycle. Certain problems using large areas of Neumann boundaries, are subject to slow convergence so that the results may be incorrect. This can be remedied either by iterating for more cycles or by giving the program a better starting distribution. These techniques will be described in a subsequent section. Generally the iteration process is quite satisfactory and after 50 iterations the field is sufficiently determined to start ray tracing leading to the inclusion of space charge.

After finishing the first cycle of Poisson's equation, a potential map, or POTLIST, is printed giving the potential (normalized to 100% of the maximum potential) for every point in the RLIM by ZLIM space. Since this includes background points (points behind the surfaces) one can usually trace the outline of the problem. The background points have the initial values and should not be confused with the internal points. The POTLIST is an exceptionally effective diagnostic device and should always be studied for peculiarities. An error in boundary data may, for example, leave a strange zero in the middle of the high potential part of a device, thereby greatly distorting the fields. When used together with the equipotential plots, it is possible to pinpoint errors in a few minutes. The POTLIST is suppressed by the default value of LSTPOT = 0.

VI. STARTING CONDITIONS

After the first calculation of Poisson's equation, the program reads the starting conditions. The format is NAMELIST consisting of defining equations in which the variable is named followed by an "equal" sign and the value. Only those variables that need to be altered from the default conditions need to be specified. The sample problem demonstrates how little data needs to be specified in many cases. Using the sample problem, the following remarks will illustrate the technique. In the rest of this section, a brief description will be given for each of the options currently included in the programs. Since other options can always be added, the user must refer to the comments in the program for the up-to-date implementation.

The sample problem is coded as a spherical diode or Pierce gun. The card with &INPUT5 signals that the namelist entries follow. The entry START = 'SPHERE' directs that the spherical diode conditions will be used. The entries RAD = 257 and RMAX = 37.5 give the spherical radius and cathode radius respectively. UNITIN = 0.01 specifies that the scale of the problem is 0.01 inches/mesh unit. All problem scaling is in MKSA units so that UNITIN is immediately converted to UNIT in meters. After reading these items the program prints a table of all the starting parameters.

The starting conditions are described in the following sections according to function as follows:

A. Universal; apply to more than one case,

B. Equipotential lines; controls equipotential plotting,

C. Plotting; plot controls,

D. Magnetic fields; input and calculation parameters for magnetic fields,

E. General cathode; parameters controlling the general cathode option,

F. Spherical cathode; parameters specifically applicable to START = 'SPHERE'.

G. Card starting; parameters controlling the use of specified starting conditions.

H. Laplace starting; parameters controlling the use of the program for applications other than ray tracing.

A. Universal Parameters

For each starting parameters, there is a default value which will be the value used if it is not changed by the input. In the following discussions, the entries will be given as described by the program comments with the format:

INSTRUCTION DEFAULT,MAX COMMENT

This will be followed by a discussion of the use of the parameter.

When a second number, separated by a comma, appears for the default value, it refers to the maximum allowed value, usually determined by array limits.

(1) PERVO = X.XX PERVO = 0 ZERO USES LAPLACE/2

PERVO is the initial value of the perveance of the beam for either the START = 'SPHERE' or START = 'GENERAL' methods. Perveance is defined as the constant K in the expression

$$I = K V^{3/2} \times 10^{-6} \qquad (9)$$

where K is expressed in micropervs so that, for example, a microperveance 1.0 device operating at 10^4 volts would have a current of 1.0 ampere. The entry X.XX indicates that a decimal number is the expected value. When a single X is used, it implies that an integer is expected. The X's do not indicate the input format; the number of significant figures is not restricted except by the computer hardware, and by the logic of the program.

PERVO normally controls only the perveance of the first cycle. However, it may be "held" for any desired number of cycles by using HOLD = X. The process by which the program determines perveance is to average the perveance calculated for a given cycle with the perveance actually used in the preceding cycle. The new averaged value is then used to determine the current per ray. The averaging process has proven very effective in quickly arriving at a stable value. It has been so successful that it is frequently better to start with the averaging method than with a value "known" to be "correct" from experiment or from prior calculations. The default value PERVO = 0 is a code instruction which takes the value of perveance calculated for the LAPLACE solution and simply divides it by two to arrive at the perveance for the first

cycle. The new user of the program is advised to use the default value until specific experiences lead him to try something else.

(2) HOLD = X HOLD = 1 PERVO 'HOLDS' FOR HOLD ITERATIONS

HOLD = 2 or more causes the input value of PERVO to remain unchanged by the averaging process for HOLD iterations. There are some problems, particularly with very non-uniform cathode loading, where using HOLD helps establish the necessary space charge environment for the process to stabilize. A more frequent application is to simulate emission limited conditions by running the entire problem with a fixed reduced perveance. Then, of course, HOLD must be at least as large as NS.

(3) PE = X.X PE = 2.0 INITIAL ENERGY AT CATHODE (EV)

PE is the incremental energy that is added to every trajectory to account for the combined effect of work function potential and thermal energy. Like PERVO and HOLD, PE is only used for starting with one of the Child's Law routines for calculating the initial conditions. It is normally not necessary to have any initial PE, but some small changes may be observed by varying it. In a few low emission devices, it has been found essential to have some initial energy to avoid instabilities near the cathode.

(4) ERROR = X.X ERROR = 1.0 MULTIPLIES ERROR TEST

ERROR = 2.0 doubles the built in error test by which the program determines that an adequate solution of Poisson's equation has been reached. If the problem is slow to converge, particularly if there are large areas of Neumann boundary, it may be necessary to reduce the

allowed error, e.g., ERROR = 0.1, to get the program to converge at all. Slow convergence is indicated if each cycle only iterates three times, prints N = 3, ERR = nnn, and calculates the trajectories. On the last cycle, the error test is reduced by a factor of 10 from whatever level was set by the user. Some hints about convergence problems will be found in a later section.

(5) UNIT = X.XXX UNIT = 0.001 METERS/MESH UNIT

(6) UNITIN = X.XXX (SEE UNIT) INCHES/MESH UNIT

The default scale value for the program is 0.001 meters/mesh unit. If a value is given for UNITIN (inches/mesh unit) this value will be immediately converted to meters. Except for problems using magnetic fields, the optics of an electron gun does not depend on the scale factor. All the standard rules of scaling in electron optics can be used once a problem has been solved.

(7) MAXRAY = XX MAXRAY = 27, 51 MAXIMUM NUMBER O' RAYS

IF MAXRAY IS NEGATIVE, THE NUMBER OF RAYS=ABS(MAXRAYS)

MAXRAY determines the maximum number of electron trajectories that can be calculated. The arrays for trajectories have a limit of 51. The number of rays used by START = 'GENERAL' or START = 'SPHERE' is determined by a program algorithm unless the value read in is negative. Within the limit MAXRAY, the program tries to make an integral number of rays per mesh unit at the cathode.

(8) STEP = 0.XX STEP = 0.8 MESH UNITS/STEP

STEP is the iteration step length for ray tracing. It must be less than 1.0 for the program to properly account for space charge, calculate

magnetic fields, etc., when crossing a mesh line. The equations of motion are time dependent, thus the program uses STEP to calculate step time from the velocity at the start of the step. Since the electron can accelerate during a step, it may actually go slightly farther than STEP. The default value is about the largest that should be used. If magnetic fields are present, STEP should usually be reduced at least a factor of two. On the last cycle, STEP is automatically reduced by a factor of two. Shortening the step means more time will be required for a problem. As a rule of thumb, the program spends roughly half of the time with Poisson's equation and half with the ray tracing. Thus reducing STEP by a factor of two could increase cost by about 25% the first time but may nearly double it thereafter. The Runge-Kutta method is used to solve the differential equations of motion. Because of the necessity to take small steps anyway, and because of the time needed, the program does not use any of the "predictor-corrector" techniques of verifying step length. Experience has shown that errors due to STEP being too large, especially if magnetic fields are included, become glaringly obvious when the plots are examined. The most frequent effect is for a trajectory to get too close to the axis, violate conservation of angular momentum in one step, and fly out of the problem area with beta > 1.0, where beta = v/c. An error message to this effect is printed when a ray ends with beta > 1.0. At the very least, this is a signal to reduce STEP in subsequent runs.

(9) NS = X NS = 7 NUMBER OF ITERATIONS

NS defines the number of program cycles to be made. In the program, NL is used as the running variable to record the number of cycles left to

be run. Initially NL = NS. The default value is usually acceptable unless the program is having trouble converging on the perveance. For the special case of no space charge, it is advisable to still use NS = 2 to gain the insight afforded by the reduction of ERROR and STEP on the final cycle. For START = 'LAPLACE', NS is the number of times that Laplace's equation will be cycled.

(10) SPC = 0.XX SPC = 0.5 ESTIMATED SPACE CHARGE

SPC SIMULATES PARAXIAL APPROXIMATION ON FIRST CYCLE. SPC IS THE FRACTION OF THE RADIAL FORCE USED. SPC = 1 FOR FULL EFFECT, SPC = 0 FOR NO EFFECT.

SPC determines the fraction of the ordinary radial electrostatic force that will be applied to the rays on the first cycle. In a device in which space charge forces play a strong part in the focusing, the electrostatic fields usually have a strong radial restoring effect. If not opposed by space charge on the first cycle, these forces may cause the rays to strongly over focus leading to a poor initial distribution of the space charge. The full contribution, SPC = 1.0, adds a term to the radial equation of motion simulating all the current, of all the rays calculated, to lie in a conductor on the axis. Thus it is assumed that the rays are calculated in sequence starting with the ray nearest to the axis. In the case of an electron gun calculation starting at the cathode, a better choice is SPC = 0.5 which attenuates the force by 0.5. Near the cathode, this corresponds to a current starting from the cathode and extending infinitely in only one direction. Further from the cathode, SPC = 0.5 is a less logical choice, but the beam is less sensitive to

radial forces as it gains in energy. Empirically, it has been found that SPC = 0.5 is a good choice for gun problems involving starting from the cathode. For other types of problems, the user should be aware of the fact that SPC exists and can be changed. In rectangular coordinates, SPC simulates an infinite sheet of current on the axis. If the problem does not involve reflection about the R = 0 plane, then there is a transverse force (which does not depend on distance from the x-axis) which should be turned off by SPC = 0.0. Since SPC only affects the first cycle, the program will usually forgive any misuse of it. SPC can be useful in arriving at a satisfactory solution of one usually difficult problem, that of a long thin beam with magnetic fields providing the focusing. This can be a difficult problem to get to stabilize because of the poor aspect ratio which frequently finds a large fraction of the beam within one or two mesh units of the axis. However, it is usually well represented by the paraxial approximation so that a single cycle run, NS = 1, with SPC = 1, will frequently result in a good solution. In this case one must be sure that STEP is small enough and that an adequate solution of Laplace's equation was attained, since ERROR had no effect on the first cycle.

(11) PHILIM = X.X PHILIM = 0.0 AZIMUTHAL LIMIT

PHILIM .NE. 0 ENDS TRAJECTORY AT PHI .GT. PHILIM

For special applications, it is possible to establish an orbit that would continue until the program is stopped. An example is an electron orbiting in a uniform magnetic field. PHILIM has the units of PHI; radians in cylindrical coordinates and mesh units in rectangular coordinates.

(12) SAVE = 1 SAVE = 0 SAVE = 1 SAVES BOUNDARIES

TO USE SAVE = 1, OMIT BOUNDARY CARDS FROM NEXT PROBLEM

SAVE = 1 is a signal to the program to expect a second problem run immediately after the first problem, and that the second problem will use the same boundary conditions. It is always possible to run tandem problems although, at most computer facilities, there is no particular incentive to do so. Programs are usually run from load modules, or from a library of compiled subroutines to be linked with very little expense, and separate problems can be run independently without the risk that a failure in the first problem will affect or knock out the second one. However, in the case where successive problems use the same boundary conditions, considerable savings in effort and computer time can result by saving the boundaries, which also saves the arrays of potentials and space charge.

The SAVE = 1 parameter is put in the starting conditions of the first problem, not the second one unless there is still to be a third problem. The data deck for the second problem starts immediately after the last data card of the first deck with no EOF or /* control cards. The second deck is complete in every respect including title, potential, magnetic fields, etc., except that the boundary cards and the accompanying large potential number card are omitted. The potentials can be changed between runs; if the largest potential is changed, the program will scale all potentials in the potential map proportionately. Otherwise the program will start out just as if a cold start was being made, except that the old solution, including the last space charge array, is used as a "preload."

One example of the use of SAVE is to be able to trace rays with small changes of either voltage or magnetic fields. Another use is in the case in which the Laplace solution is difficult to achieve because of extended lengths of Neumann boundaries. In this case, it may help to run the first part with START = 'LAPLACE' (see section VI-H) and SAVE = 1 and then do the ray tracing in the following problem. This saves the time and expense of ray tracing in an incorrect potential distribution. This procedure is not normally required since the usual procedure allows the program to improve the solution on successive iterations as the space charge is entered.

The special case of a pair of electrodes separated by a long length of Neumann boundary parallel to the z-axis causes special problems with convergence that might respond to the approach using START = 'LAPLACE'. An alternative approach, which is easier, is to introduce a few boundary points along the top or bottom Neumann boundaries, with potential numbers. If the corresponding voltages, which must be entered in the potential list, represent approximate values for the potentials in the final solution at that point, then the starting load to the program will be much better than the normal starting load. Usually the starting load is of very little significance, but in this special case it can be crucial. The special boundary points are exactly like the usual Neumann points, except that the potential number is given and refers to an appropriate element of the POT array. After the preload, the Neumann points relax as usual and the potentials change accordingly.

(13) SAVE = 2 SAVE = 0 USES FINAL DATA FROM PREVIOUS RUN TO START THIS RUN. USE ONLY WHEN START = 'CARDS'.

Save = 2 allows consecutive runs to use the final conditions of a preceding problem as the initial conditions of the succeeding problem. Necessary scaling and positioning adjustments are made as described under START = 'CARDS', below. The SAVE = 2 goes to INPUT5 of the second run.

Note that the dual use of SAVE = 1 and SAVE = 2 in one problem is not permitted, but that SAVE = 1 on the first problem followed by SAVE = 2 in the second is both permitted and quite common. It simulates the repeated use of a drift tube, periodic focusing section, etc.

(14) MASS = X.X MASS = 0.0 MASS > 0 FOR IONS
MASS IS THE MASS TO CHARGE RATIO, 1.0 FOR PROTONS
USE MASS > 0 FOR RAYS WITHOUT INERTIA; CAN BE USED FOR MAGNETIC FLUX LINES OR ELECTRIC FIELD LINES.

MASS is used to signal the program that particles other than electrons are to be followed. The units are in 1836 electron masses, so that a proton would be 1.0 and a doubly ionized tritium ion would be 3/2 = 1.5, for example. The Child's Law routines for starting still function. Note that the intrinsic charge built into the program is negative. Ion problems are normally run as if charge is negative, although negative currents (positive charges) are permitted for START = 'CARDS'.

(15) AV = X AV = 0 SPACE CHARGE AVERAGED LAST AV CYCLES

(16) AVR = X.X AVR = 1.0 WEIGHT OF PREVIOUS CYCLE FOR AV

AV and AVR are companion parameters to help improve stability by averaging the contribution of space charge over successive cycles. It

should not be confused with the different process of emission averaging to determine perveance. In fact, to keep the emission averaging and space charge averaging from affecting each other, it is suggested that AV be small enough so that the emission averaging is essentially complete before space charge averaging starts. Note that AV is for the last AV cycles, e.g., if NS = 7 and AV = 3, then only cycles 5, 6 and 7 are averaged. However, this may have a very small effect since the trajectory calculations of cycle 5 are not affected and the space charge determined by the cycle 7 is never used (since there is no cycle 8). Thus the effect of averaging is only observed for AV-1 cycles. AVR determines the weight of the previous cycle such that with AVR = 1.0, the space charge from the previous cycle is weighted equally with the present cycle. AVR can have any value, $0 < AVR < \infty$.

Experience with averaging has shown the effect to be less dramatic than one might anticipate. A poorly designed gun, with strong spherical aberrations and resulting crossovers, is likely to be unstable and converge poorly even with averaging. Also, application of averaging to relativistic high intensity beams does not do much to solve the inherent difficulty caused by the fact that the self-magnetic field forces nearly cancel the space charge forces. With the two-cycle format of the program (i.e., space charge from the previous cycle and self-fields from the present cycle) the program has difficulty converging on long beam transport problems. The solution to this situation is frequently to use the first cycle only with the paraxial approximation and SPC = 1.0 as described in VI.A.10 above.

(17) BEND = X.X BEND = 0.0 MAGNETIC BENDING FIELD IN GAUSS IN THE DIRECTION NORMAL TO THE R-Z PLANE FOR THE AXIALLY SYMMETRIC PROBLEMS. FIELD MUST BE UNIFORM. THE EFFECTS OF SELF-MAGNETIC FIELD ARE LOST AND SPACE CHARGE IS STILL AXIALLY SYMMETRIC SO THAT IF BEAM IS DEFLECTED, CHARGE DISTRIBUTION IS PROBABLY INCORRECT. AN AXIAL FIELD MUST BE INCLUDED IN THE INPUT, EVEN IF IT IS ZERO, E.G., BC=0 IN INPUT2.

This feature is most useful for problems with little or no space charge. Various types of photo tubes have tight tolerance for transverse magnetic field effects. Residual transverse fields, earth's field, etc., can be calculated. Note that a cylindrical beam in a rectangular coordinate geometry, including transverse field and space charge, can be simulated as described below in Section VI.G.4.

(18) MAGMLT = X.X MAGMLT = 1.0 MULTIPLIES BZA ARRAY

MAGMLT multiplies the entire BZA ( ) array after it has been read in or calculated internally. It also multiplies the entire vector potential array if that option is used. It can be thought of as a knob on all the magnetic field generating power supplies.

(19) IPRP = K1, K2,...K6 IPRP = 0 UP TO SIX RAY NUMBERS FOR POINT-BY-POINT PRINTOUT:

K, RHO, ZETA, RDOT, ZDOT, TDOT, PHI, BR, BZ, STEP, RPHI

In special situations, especially when program behavior is not as expected, it is useful to be able to print out every iterative step. This feature operates on the last program cycle. Thus if for example a bug is stopping the program in the first cycle, it is necessary to set

NS = 1 and set IPRP = (the number of the trajectory at question). Note that it is possible to generate a great deal of paper this way. In some cases, one might rather have other items printed than those in the above list. It is a simple change to substitute ER, EZ, etc., for BR, BZ, for example.

(20) ZEND = X.X ZEND = 1000.0 EXACT END OF TRAJECTORY

CAUTION: IF ZEND IS NOT THE RIGHT-HAND BOUNDARY, THE SPACE CHARGE DISTRIBUTION MAY BE INCORRECT.

Normally a trajectory is calculated until the program can no longer determine the electric fields. Thus the trajectories usually go up to one-half mesh unit beyond the boundaries. In special situations, such as high-resolution photo tubes, this makes exact interpretation of the results difficult. Setting ZEND to a specific value causes the program to back up to this value when a trajectory passes through this value of zeta.

(21) VION = X.X VION = -128 LOWEST POTENTIAL PERMITTED

USE VION TO SIMULATE SPACE CHARGE NEUTRALIZATION.

Space charge depression can be reduced in a real device by positive ions in an electron device or by electron clouds in an ion beam. Since the program normally runs with negative charges, the above cases both result in negative space charge depression. If it is desired to limit the depression, VION can be set to the lowest depressed potential that is desired. The default value is intended to be low enough so that it will never disturb a practical problem.

B. Equipotential Plots

Under the heading:

INPUT FOR EQUIPOTENTIAL PLOTS,

the instructions list the parameters which may be used to control the output of the equipotential lines.

If the plot control parameter MI, on the potential card, has been set to MI ≤ 6, then the subroutines which draw equipotential lines will be called at the appropriate times. If the entire problem is at one potential, it is usually better not to call for equipotential plots.

The method used in the program to find the equipotential lines consists of first finding a starting point for the potential to be followed, and then following a line of constant potential from that point. This does not guarantee that every point of that potential will necessarily be found and plotted. If POT (2) ≠ 0 the program always draws the equipotential line for V = b · POT (2) where b = 0.05, 0.15, 0.25, 0.35, ... 0.95. Also if POT (3) ≠ 0, the program draws lines for V = b · POT (3) where b = 0.2, 0.4, 0.6, 0.8, 1.0 Normally the lines are started at the points on the axis which are at that potential. The expectation is that POT (2) will be used for the anode and POT (3) will be used for the grid, if any. If, for example, one is designing a gridded gun to be operated at $V_G = 0.01\ V_A$, then, by first designing the gun as a diode, and plotting POT (3) at 0.01 POT (2), one gets the ideal contour for the grid to be electrically invisible.

(1) EQUIPR = X.X EQUIPR = 0.0 R-INTERSECT. FOR EQUIP. LINES

EQUIPR is the radius of the line along which the program hunts for the potentials which are to be plotted. It sometimes happens, particularly

in rectangular coordinates, that the equipotential lines do not intersect the z-axis, (R = 0 line). EQUIPR lets the programmer indicate along which horizontal line the program should look for the starting points.

(2) LM = XXX LM = 303 LENGTH OF EQUIPOTENTIALS

LM is the array limit for the points to be plotted for any one equipotential. If a line simply stops in midstream, it may be desired to increase LM. Arrays BX and BY must be as large as LM.

(3) EQLN = 0 to 20 EQLN = 1 *NO. OF CORRECTIONS

EQLN controls the iterative corrections made as each point is found along the equipotential line. These corrections prevent the lines from deviating from sharply curving equipotential lines. The default value, EQLN = 1, is usually adequate.

(4) EQST = X EQST = 2 *STEPS PER MESH UNIT

EQST gives the density of points for the equipotential plots. The maximum length of a line is given by the ratio LM/EQST. If EQST is too small (steps too long), fine detail may be smoothed over.

*ALSO APPLIED TO GENERAL CATHODE

This footnote warns that the starting surface for the GENERAL CATHODE routine is generated just like an equipotential (but is not plotted), and thus the parameters EQLN and EQST may determine the accuracy of the starting surface. It is primarily for this application that EQLN and EQST are made variable parameters.

(5) IZ1 = X, IZ2 = X, IZS = X IZ1 = 0, IZ2 = -1, IZS = 10 EXTRA EQUIPOTENTIALS AT THE INDICATED VALUES OF Z.

IZ1 and IZ2 are the end points of a line segment, at EQUIPR, along which some extra equipotential lines will be started. The lines will be equally spaced by IZS, instead of by voltage, so that their density will not mean field gradient. The default value, IZ2 = -1, turns this device off.

C. Plotting Controls

(1) SCALE = 'YES' SCALE = ' ' 'YES'=DIFFERENT X,Y SCALES

SCALE = 'YES' allows the axis routines to adjust both the X and Y scales to take maximum advantage of the size of the paper. The default value constrains the axis to have the same scale factor in both directions, thus preserving the actual proportions. Using SCALE = 'YES' allows the plots to show more detail between trajectories in problems with low height/length ratios.

(2) SX = XX SX = 22 MAX. HORIZ. PLOT LENGTH

(3) SY = XX SY = 9 MAX VERTICAL PLOT HEIGHT

SX and SY control the area for each picture. The dimensions are given in inches. SX can be adjusted to suit the length of a given problem.

Plot data generated by the program are stored on an external file (disk) in a format very similar to that normally used as input to the software supplied with CALCOMP plotters. A separate job, or second job step, can then be run to generate the plots. A simple program is

printed in the appendix to convert these data to make CALCOMP plots. Other plotter software such as that used at Stanford can be programmed by making the appropriate calls to the local subroutines. With the changes that resulted in the above system, a programmer at another installation does not need to search for plotting commands within the electron trajectory program. Conversion to local software is usually quite simplified.

D. Magnetic Fields

Magnetic fields play a vital role in steering and focusing many kinds of electron beam devices. The capabilities and limitations of the magnetic field implementation in the program will be described in this section. The following areas will be discussed:

1. Magnetic Field Input; (a) axial, (b) ideal coils, (c) vector potential data;
2. Off-axis field expansions;
3. Magnetic fields in Rectangular Coordinates.

1. Magnetic Field Input

In the present implementation of the program, there are three methods of inputting magnetic field data:

(a) By reading in the field on the axis using either a polynomial expansion or by reading the full array,

(b) By specifying ideal coils (radius, position and strength).

(c) By reading in vector potential data from the output of a two-dimensional magnet design program such as TRIM or POISSON.

(a) The data cards for an axial magnetic field are put in before the boundary data. The format was briefly described in Section V.B. The input data for the polynomial method consist of MAGSEG segments of data including: 'Z1' to 'Z2' with origin at 'Z3' (three integers) and seven coefficients, BZ, B1, B2,...,B6;

B = BZ + B1 * DZ + B2 * DZ ** 2 +...+B6 * DZ ** 6, where DZ = Z - Z3.

For the sixth order expansion, the field must start six units behind the cathode or starting point, and go six units past ZLIM. In rectangular coordinates, the normal magnetic field is in the transverse (phi) direction.

The NAMELIST input for RLIM, etc., (&INPUT1) includes the parameter MAGSEG (default MAGSEG = 0) which determines how many segments are to be read, each with &INPUT2 and &END cards. Each segment consists of the data for Z1, Z2 and Z3 followed by the array BC in NAMELIST format.

Z1 and Z2 are the end points of a line segment on the axis (Z1 ≤ Z2) in the range $-6 \le Z1, Z2 \le ZLIM + 6$. It is necessary to permit fields to be described beyond the ends of the problem in order that the off-axis fields can be calculated at the ends of the problem. Z3 is the local origin for the polynomial expansion in powers of DZ = Z - Z3. Having a local origin simplifies the input of, for example, a straight line that does not go through (0,0). As many of the coefficients BZ, B1, etc., can be used as are necessary, simply by setting the remaining ones to zero.

In cylindrical coordinates, this field must be in the axial direction. In rectangular coordinates, the field on the axis may be either in the direction normal to the plane of the plot, i.e., in the PHI

direction, where PHI is the orthogonal linear coordinate to R and Z, or in the R (vertical) direction.

With the above format, data can be entered with any degree of polynomial up to 6. The data may be divided into segments ranging from a point at a time to the whole length of the problem. Typically, magnetic measurements of an axially symmetric permanent magnet will be taken on the axis. The data are then frequently smoothed by a polynomial least squares fitting program and the resulting coefficients read into the program. Alternatively, a field may be designated by the user as in the example problem, segmented into short lengths of quadratic or linear dependence, and read in to the program. Either method will usually give a good representation of the field on the axis. However, difficulties arise when the program needs to calculate the off-axis fields. These will be described in Section 2, below.

A separate provision allows one to read in the BZA array directly. Note that this array starts with BZA(1) at Z = -6 and goes to BZA(ZLIM + 11) at Z = ZLIM + 6. The program switches to this mode by having MAGSEG < 0, i.e., if MAGSEG = -1, then a different NAMELIST, &INPUT3, is called to read the array BZA ( ). If measured and/or plotted data are used, note especially the inherent risks in expanding such data for the off-axis field components. This format lends itself readily to computer calculated output, properly edited, and with up to 15 effective decimal digits.

(b) The data for ideal coils are read in as part of the INPUT5 starting conditions. The starting conditions pertaining to magnetic fields are as follows:

MAGNETIC FIELDS

METHOD ONE: READ IN AXIAL FIELD.

RMAG = X.X	RMAG = RLIM/2	OFF-AXIS MAG FIELD LISTING
ZMAG = X.X	ZMAG = ZLIM + 6	B CONSTANT BEYOND ZMAG
MAGORD = X	MAGORD = 2	HIGHEST ORDER FIELD TERM ≤ 6

IF MAGORD < 0, RECT. COORD. MAG FIELD ARRAY BZA IS IN THE R DIRECTION

NMAG = X	NMAG = 0	NO. OF FIELD COILS (SEE BELOW)

METHOD TWO: READ IN POSITION AND STRENGTH OF NMAG IDEAL COILS

NELL = 1	NELL = 0	1 FOR ELLIPTIC INTEGRALS
CR(I) = X.X	CR(I) = RLIM	RADIUS OF COIL (MESH UNITS)
CZ(I) = X.X	CZ(I) = 0.0	AXIAL POSITION OF COIL
M(I) = X.X	CM(I) = 0.0	CURRENT IN AMPERE TURNS

B(AXIS) = 0.2 * CM * PI * CR ** 2/SQRT (((Z-CZ) ** 2 + CR ** 2)) ** 3 GAUSS

WHERE I IS COIL NUMBER, E.G., XZ(2) = 20.0.

'METHOD ONE' REFERS TO THE POLYNOMIAL INPUT JUST DESCRIBED.

(1) RMAG = X.X	RMAG = RLIM/2	OFF-AXIS MAG FIELD LISTINGS

RMAG is used only by an output routine that prints the axial and radial components of the magnetic field at the radius RMAG. The default value is chosen to be typical of the maximum radius of the beam, but it should be adjusted to suit the problem. For a pencil beam, RMAG should be equal to the expected average beam radius (in mesh units). This printout is a useful diagnostic device to check on unrealistic off-axis components that can result if the inputs have discontinuities in one of the higher derivatives.

(2) ZMAG = X.X ZMAG = ZLIM + 6 B CONSTANT BEYOND ZMAG

ZMAG permits some simplification of data by setting the axial field from ZMAG to ZLIM + 6 equal to the calculated value at ZMAG. The principal use for ZMAG is where a converging magnetic field in the gun region merges into the uniform field of a solenoid. The field expressions or coils must describe a field which converges to parallelism at the solenoid entrance, and ZMAG is then the Z coordinate (in mesh units) of this point.

The default value of ZMAG (ZLIM + 6) ensures that it then has no effect in the working region up to ZLIM.

ZMAG is a positive integer.

(3) MAGORD = X MAGORD = 2 HIGHEST ORDER FIELD TERM ≤ 6

MAGORD is the highest order term, in powers of R, that will be used to calculate off-axis fields. It is not related to the power of the polynomial input. Usually MAGORD has one of the values, 2, 4 or 6. If MAGORD is higher than warranted by the quality of the data, particularly if data from magnetic measurements are used, then the off-axis fields may be just plain nonsense. If MAGORD < 0 (rectangular coordinates only), the array BZA ( ), on the z-axis, is taken to be in the R directions. Off axis expansion, in powers of R, are used to generate BZ (off axis). This case is suitable for quadrupole symmetry in rectangular coordinates as viewed end-on to the beam.

(4) NMAG = X NMAG = 0 NO. OF FIELD COILS

"Method Two" refers to the method of ideal coils. NMAG is the number of ideal circular current loops, centered on the axis and lying in

planes perpendicular to the axis. NMAG may have any positive integer value, but practical field shapes can usually be represented by no more than 11 coils, which is the array size. Each coil is described by three parameters:

CR(I) = radius of coil (mesh units);
CZ(I) = axial position of coil;
CM(I) = ampere-turns;
where I = 1 to NMAG

The index is not related to the strength or position of the coils. Some methods of obtaining CR and CM values that will fit a desired field are discussed in Ref. 7.

The subsidiary parameters RMAG and ZMAG which have been discussed above, apply equally to method two (coils) as to method one.

All CR( ) values must be positive (not zero, or a zero divide will occur); CR is not restricted to be within RLIM, but may have any positive value. It need not be an integer. The CR values should be larger than the beam radius to avoid strong local non-uniformities.

CZ( ) values may be positive, negative or zero, integer or decimal, and are not restricted by ZLIM. The program calculates the field only within the working space RLIM × ZLIM, but the coils may be inside or outside this space.

CM( ) values are unrestricted.

All the coil data are entered in the &INPUT5 NAMELIST block.

Examples of magnet field entry using coils (these data represent a field converging into a solenoid which starts at Z = 100):

```
(last boundary card)
888
&INPUT$
(usual START cards)
NMAG = 3,
ZMAG = 100,
RMAG = 5,
CR(1) = 150,
CZ(1) = 6.8,
CM(1) = -900,
CR(2) = 50.0,
CZ(2) = 50.0,
CM(2) = -2000,
CR(3) = 32.0,
CZ(3) = 100.0,
CM(3) = 31000,
&END
(Card start data, if any)
/*
```

2. Off-Axis Field Expansions

The two input methods described above both result in an array of fields from Z = -6 to Z = ZLIM + 6. The array is for the axial field and is in double precision. With this number of significant figures, it is possible to get meaningful results for finite differences up to the sixth difference, which is necessary for the sixth order derivative

used to find the off-axis fields. Each difference requires one larger value of n in Z ± n, the range used to find the field at Z, at any radius. The range Z ± 6 requires that the fields be specified beyond the limits of the problem from Z = -6 to Z = ZLIM + 6.

To sixth order, the field expansions are[8]

$$B_z = B_z(Z) - R^2(d^2B/dZ^2 - d^4B/dz^4 \cdot R^2/16 + d^6B/dz^6 \cdot R^4/576)/4 \quad (10)$$

$$B_r = -R(dB/dZ - d^3B/dZ^3 \cdot R^2/8 + d^5B/dz^5 \cdot R^5/192)/2 \quad (11)$$

By specifying MAGORD = 2 or MAGORD = 4, the derivatives higher than MAGORD are set to zero. This results in a less accurate expansion, if the original data are worthy of the high order differences. If they are not, then the result of the lower order expansion is apt to be far more acceptable. Generally, measured data, no matter how smoothed, are only worthy of second order expansion. Synthesized data from an ideal curve, if there is only one segment, can generally be expanded to fourth order. Coil data can be expanded to sixth order. Note, however, that it is virtually impossible to use the full sixth order expansion with either measured data or arbitrary polynominls, especially if more than one segment is to be fit together without running the risk of having a very unphysical result. The off-axis fields generated by poor models, or ones with insufficient accuracy, are apt to show very wild fluctuations with extremely large peak values.

3. Rectangular Coordinate Expansions

In rectangular coordinates, the usual expansion is normal to the plane of the paper. The central plane, with coordinate PHI = 0, can be

thought of as the median plane of a magnet whose pole face is normal to the z-axis, i.e., dB/dR = 0.

The off-median-plane expansion is

$$B_{PHI} = B_{PHI}(Z) - PHI^2 \cdot d^2B/dZ^2 \qquad (12)$$

$$B_z = PHI \cdot dB/dZ \qquad (13)$$

The alternative expansion has the median plane lying normal to the R-Z plane, at R = 0. The off-axis expansion is then in the R direction.

The second order expansion has been adequate for the applications that have been made. One example is the "alpha" magnet deflection system used to bend the low energy SLAC beam from the gun to the line of the accelerator. A proper choice of angle makes the vertical focusing of the pole face edge compensate for the vertical phase space of the beam. Runs at different entrance angles, using the measured field profile of the magnet, were used to determine the optimum angle. Space charge of a cylindrical beam, in rectangular coordinates can be included in such runs by the features described for CARD starting in section VI.C.

4. Elliptic Integrals

For coil input (Method Two"), if elliptic integral routines are available at compilation, a table of off-axis fields with elliptic integral calculations is printed. If NELL = 1 in &INPUT5, the elliptic integrals are used for the ray tracing.

(c) Inputting Vector Potential Data

In &INPUT1, the option INTPA = .TRUE., calls for &INPUTA to be called next. The condensed instructions are:


```
&INPUTA          (TO INPUT VECTOR POTENTIAL DATA)
     RRO=X.X   RRO=0.0     POSITION OF FIRST ELEMENT OF A( ), IN MU
     ZZO=X.X   ZZO=0.0     RELATIVE TO ORIGIN OF GUN PROB.
     DELR=X.X  DELR=1.0    INCREMENT IN R (CM) FROM POSSON/EDIT
     DELZ=Z.Z  DELZ=1.0    INCREMENT IN Z (CM) FROM POISSON/EDIT
     RLMAG=XX  RLMAG=30    NUMBER OF ROWS OF A( ) DATA
     ZLMAG=XX  ZLMAG=200   NUMBER OF COLUMNS OF A( ) DATA
     A( )      VECTOR POTENTIAL DATA ARRAY OF A, EXCEPT A*R AT R=0
               UNITS OF A IN GAUSS-CM. A( ) IS A LINEAR ARRAY WITH
               COLUMNS RLMAG LONG.  MAX SIZE OF A( ) IS 8000.
```

Use of this option requires the output from a magnet design program, such as POISSON, which solves for the magnetic field including iron segments, which may even be partially saturated. The output of such programs is usually in the form of an array of the azimuthal component of the vector potential A( ). This array is currently set to a maximum of 8000 elements, but may be reduced to one element to save space for users not interested in this option. The array elements correspond to points in a rectangular mesh which does not need to coincide with the mesh used for the electrostatic problem. To save running time for the magnet program and to reduce storage requirements for the data, it is preferable to identify a rectangular area that is expected to include the space that the electron trajectories will require. The array starts at RRO, ZZO, proceeds in steps of DELR in columns RLMAG long, and contains ZLMAG columns separated by increments DELZ. During operation, the program finds the differences from the four points nearest the particle to find the components BR and BZ.

E. General Cathode and GENCARD

START GENERAL

START = 'GENERAL'	START = 'GENERAL'	GENERAL CATHODE
RC = X.XXX	RC = 0.0	LOWER END OF STARTING SURFACE
ZC = X.XX	ZC = Z+CATHODEZ	CATHODEZ IS Z VALUE OF BOUNDARY FROM FIRST DATA CARD.
CL = X.XX	CL = RLIM	MAXIMUM LENGTH OF STARTING SURFACE
DENS = XX.X	DENS = 10.0	MAXIMUM EMISSION (A/CM**2)
BETA2 = 1.0	BETA2= 0.0	IF > 0.0 USES LANGMUIR-BLODGETT
RAD = X.X	---	USE RAD FOR WIRE RADIUS IN RECTANGULAR COORDINATES, BETA2 > 0.0
SURFACE = X	SURFAC = 1	STARTING SURFACE ITERATION

USE POT(5) FOR NON-EMITTING SURFACE, E.G. HOLLOW CATHODE OR SHADOW GRID. DO NOT USE POT(3) OR POT(5) FOR FOCUS ELECTRODE ... USE POT(4) TO STOP ELECTRONS ON IMPACT.

START GENCARD

START = 'GENCARD'	START = 'GENENERAL'	GENERAL WITH CARD START

HAVE UP TO MAXRAY CARDS WHICH SPECIFY:
1) RAY NO.
2) INITIAL RADIUS R
3) INITIAL AXIAL VALUE Z
4) DISTANCE FROM CATHODE DX (CATHODE MUST BE POT(1)).
5) EFFECTIVE SPACING BETWEEN RAYS, DR.
6) PARAMETER WHICH MODIFIES CHILD LANGMUIR EQUATION. ALPH2

NORMAL DX IS 1.0 TO 2.0 MESH UNITS.
NORMAL DR IS 1.0 BUT MAY BE VARIED ALONG THE SURFACE.
NORMAL ALPH2 IS 1.0 FOR A PLAIN DIODE.
FOR CYLINDRICAL COORDINATES:
ALPH2=(ALPHA*(RADIUS OF CURVATURE)/(STARTING STEP))**2
FOR RECTANGULAR COORDINATES:
ALPH2=(BETA**2)*(RADIUS OF CURVATURE)/(STARTING STEP)
WHERE ALPHA AND BETA ARE AS DEFINED IN THE LITERATURE,E.G., SPANGENBERG FOR BETA AND BREWER IN SEPTIER, VOL 11, FOR ALPHA.
FORMAT IS THE SAME AS FOR CARD STARTING; RAY NO..R.Z.DX.DR.ALPH2 (15.5X.5F(10.5)).

This section describes the use of the GENERAL cathode method which applies to anything that cannot be described using the assumptions of a spherical cathode. It includes the GENCARD option.

In calculating starting conditions using Child's Law, the basic assumption is that of space charge limited emission. Mathematically, this means that the electric field on the surface of the cathode is zero. Thus, in order to calculate the emission current, the calculation must start some finite distance from the cathode. This leads to the use of Langmuir diodes, or pill boxes, which become annular in shape in cylindrical coordinates. The typical thickness is 2.0 mesh units, with the range 1.0 to 3.0 generally acceptable.

The basic Child-Langmuir equation for emission in a plane diode is[9]

$$J = \frac{2.335 \times 10^{-6} V^{3/2}}{x^2} \text{ in amperes per unit area} \qquad (14)$$

The 3/2 power dependence of the thermionic emission current density leads directly to the concept of perveance here defined as the constant K in the expression

$$I = K V^{3/2} \times 10^{-6} \qquad (15)$$

Since K depends only on geometric factors, the perveance becomes an identifying characteristic of the device. Because of common usage, perveance for the program is expressed with the implied factor of 10^{-6}, i.e., microperveance having units microamperes per volt$^{3/2}$.

The central problem for the GENERAL cathode starting routine is to define the starting surface and to calculate the distance x for the thickness of the pill box. The starting surface is initiated at the point (RC,ZC) with default values RC = 0 and ZC = 2.0 + CATHODEZ. The default point represents a point on the axis 2 mesh units in front of the Z value of the first boundary point. If the cathode does not start on the axis, a different value for RC must be used. If the first boundary point does not describe the beginning of the cathode, then a different value of ZC must be used.

The term CATHODEZ refers explicitly to the value Z + ΔZ of the first boundary point. It is frequently convenient to make the R = 0 intercept of the cathode be the first boundary point, but there is no rule about this. The starting step (or diode thickness) of 2.0 mesh units can also be adjusted by using a different value of ZC. The parameter ST, used for spherical starting, does not apply to GENERAL starting.

The starting surface is calculated by starting an equipotential line at (RC, ZC) and following it, in one direction only, until one of three things happens:

1. The line leaves the boundary of the problem.
2. The line becomes longer than the parameter CL. (default; CL = RLIM)
3. The boundary points intercepted by a line drawn at right angles to the starting surface, extended to the left as viewed along the line starting at (RC, ZC), cease to be represented by POT(1) or POT(5). Emission will occur from surfaces represented by POT(1). No emission will occur from POT(5) surfaces; hollow cathodes or shadow grids may use POT(5). Any other potential number will cause the line to stop, with the exception that POT(3), usually used for grids, will not stop the line because it may be so close to the starting surface that confusion would result. Thus the notes suggest using POT(4) to end the starting surface.

Tests 1 and 2, above, are included as "safety valves." Test 3 is intended to determine the length of the starting surface. As the starting surface has to follow a more tortuous curve, due to holes, wires and corners, the equipotential parameters EQLN and EQST may be adjusted as described in Section VI.A.

DENS = X.X DENS = 10.0 MAX EMISSION (A/CM**2)

DENS limits the current density to a maximum value controlled by the user. It can be used to limit the emission as in temperature limited emission. The normal use is to avoid extreme values of current from local high-field points until space charge depression becomes effective on subsequent iterations. Note that temperature limited emission can also be simulated by using PERVO and HOLD as described in Section VI.A.

BETA2 = 1.0 BETA2 = 0.0 IF > 0.0, USES LANGMUIR-BLODGETT

RAD = X.X USE RAD FOR WIRE RADIUS IN RECT. COORD. BETA2 > 0.0

BETA2 and RAD refer to the parameters β^2 and r_c in the Langmuir-Blodgett[10] theory of emission between coaxial cylinders. The material is covered in Ref. 8. The Langmuir equations are included in the program for the particular case of emission from an array of wires in

rectangular coordinates. BETA2 is calculated internally once it has been activated by the user specifying a value greater than 0.0. The program uses the distance from the wire, the radius RAD of the wire, and the Langmuir equations to calculate currents in each ray. More than one wire can be used provided that the starting surface can get from one wire to the next by "seeing" POT(5) surfaces between wires. The wires that emit are of course POT(1). The current per mesh unit in length (in rectangular coordinates) is

$$I/\ell = 14.66 \times 10^{-6} V^{3/2}/(r \cdot \beta^2) \quad \text{amperes/mesh unit} \qquad (16)$$

where r is the starting radius in mesh units and

$$\beta^2 = U(1 - 0.4\ U + 0.344\ U^2) \quad \text{where } U = \ln(r/RAD). \qquad (17)$$

The more usual configuration of emission from a flat or concave surface in cylindrical coordinates is treated by the program if BETA2 = 0.0. Then the program treats the annular pill boxes formed by dividing the starting surface into a number of equal segments. The number of rays is calculated by the program to be the largest number (≤ MAXRAY) that can be distributed evenly along the starting line, i.e., 1 or 2 per mesh unit, not 1.5!

The program determines the potential at the point on the starting surface from which the rays are to start and calculates the starting velocity and the current using either the equation for cylindrical emission, if in rectangular coordinates, or the equation for emission from concentric spheres[11] in cylindrical coordinates:

$$I = \frac{2.335 \times 10^{-6} V^{3/2}}{r_c^2 (-\alpha)^2} \quad \rho\ \delta\ \rho \quad \text{amperes per radian} \qquad (18)$$

where

$$(-\alpha)^2 = (\gamma - 0.3\gamma^2 + 0.75\gamma^3 - \ldots)^2 \qquad (19)$$

and

$$\gamma = \ln[(r_c - x)/r_c] \qquad (20)$$

where, as in (14), x is the thickness of the pill box, and in which r_c is the radius of the cathode and ρ and $\delta\rho$ are the radius and thickness of the annular ring on the starting surface. This equation calculates the current in a one radian segment of the annular ring. The program prints this current in the one radian segment in the table of initial conditions. Under final conditions, the current is printed divided by the initial radius, ρ. This column gives a measure of current density to determine uniformity of cathode loading. The cathode radius r_c is estimated for general cathodes by comparing the length of the cathode to the length of the starting surface. This may be incorrect if the cathode does not have a constant radius of curvature but the result is so close to the simple $1/x^2$ dependence that the discrepancy does not seem generally significant.

For cases involving cylindrical coordinates, for spherical and general cathodes, the starting step is much smaller than the radius of curvature. Thus, it is possible to simplify (19) by expanding it to second order in (x/r_c):

$$r_c^2 (-\alpha)^2 = x^2(1 + 1.6\ x/r_c + 2.06\ x^2/r_c^2) \qquad (21)$$

in which x has been redefined as positive for the usual case of a concave spherical emitting surface. With this change, (14) and (18) are essentially the same except for the correction factor, the term in parentheses in (21), called ALPH2 in the program. It is this term that is called for explicitly in the input for GENCARD.

SURFAC = X SURFAC = 1 STARTING SURFACE CYCLES

SURFAC controls the number of program cycles for which the starting surface will be regenerated. Frequently, the most satisfactory looking starting surface is generated on the first cycle, without space charge depression. The starting surface, it should be recalled, is only a locus of starting points from which particles start out in the direction of the electric field. The potential difference between the starting point and the cathode determines the initial particle velocity and the current for that ray. As space charge depression is included, the shape of the starting surface may, or may not change, although generally the potential on it will change. In any case, it is well to limit the number of cycles during which the surface is recomputed so that the final cycles converge to a stable solution. SURFAC controls the number of such cycles and, while it may often be more than one, it should generally be 2 or 3 less than NS, the total number of cycles.

General Cathode Diagnostics

If the START = 'GENERAL' option is selected, the program will print a special table of the appropriate constants: RC, ZC, CATHODE LENGTH, MAXRAYS, etc. After successful calculation of a starting surface, the message

STARTING SURFACE: LENGTH = X.X ENDS AT RHO = X.X, ZETA = X.X

will appear. Next the headings for the initial conditions will be printed followed by the initial condition data.

If the starting surface fails by not being able to trace an equipotential for at least two mesh units, or because it is asked for points outside of the problem, then the message:

GENERAL CATHODE STARTING SURFACE FAILED : LENGTH = X.X

ENDS AT RHO = X.X ZETA = X.X

is printed. If SURFAC > 1 and this failure occurs on the second program cycle, then the program will cycle once more with a smaller perveance (currently 80%) and try again to fit the starting surface. Otherwise, the program will terminate, but in either case the complete potential map will be printed to aid in diagnosis of the difficulty.

GENCARD is a starting option introduced to permit better response to highly nonuniform cathodes. A specific example would be the sharp outer corner of a right cylinder emitting from the end face. This corner is usually handled poorly by START = 'GENERAL' because of implicit assumptions that the radius of curvature of the surface is much greater than the starting step. GENCARD was specifically intended for use with high current field emission devices, but applies also to thermionic emitters.

GENCARD combines some of the functions of GENERAL with the basic philosophy of CARDS in which the user specified all the starting conditions. In GENCARD, the user specifies the initial coordinates R_o, Z_o;

the effective distance to the cathode DX; the spacing between rays DR; and the "fudge factor" ALPH2. Thus the user has defined all the parameters needed to start the space charge limited problem except initial energy and direction. These are calculated by the second part of SUBROUTINE CHILDA which is the subroutine called by GENERAL. The first part of CHILDA calculates the starting surface, and is not needed by GENCARD.

The parameter ALPH2 is the term in parentheses on the right side of (21). In rectangular coordinates, ALPH2 corresponds to the BETA^2 of the literature with (STARTING STEP/CYLINDRICAL RADIUS)$^{1st\ power}$ factored out. The effect of this is to make the normal, i.e., plain diode, value of ALPH2 = 1. Anything else is a perturbation at the user's control.

F. Spherical Cathode

START SPHERE

START = 'SPHERE'	START = 'GENERAL' SPHERICAL CATHODE
RAD = X.XX	RAD = 2*ZLIM SPHERICAL RADIUS
RMAX = X.XX	RMAX = RLIM CATHODE RADIUS
ORAD = X.XX	ORAD = CATHODEZ CENTER OF CATHODE
ST = X.XX	ST = 2.0 STARTING STEP
	'SPHERE' ALSO WORKS FOR CYLINDRICAL CATHODE IN RECTANGULAR COORDINATES

IF START = 'SPHERE' is elected, the program will first print the special table of parameters for the spherical cathode: SPHERICAL RADIUS, CATHODE RADIUS, CATHODE CENTER, etc. The first two values, RAD and RMAX, determine the essential geometry of the spherical cathode as shown in Fig. 5. Obviously the default values, 2 × ZLIM and RLIM

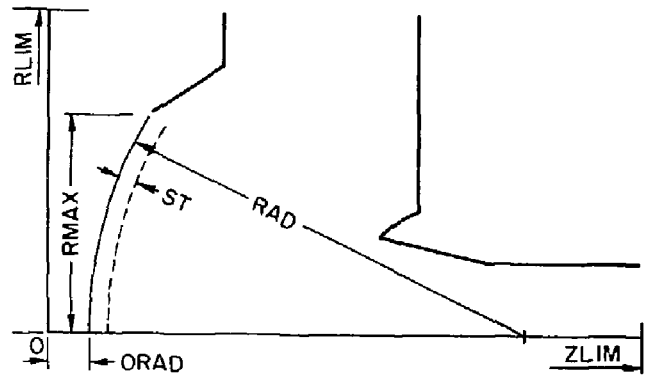


Fig. 5. Basic geometry for spherical cathode configurations defining the input parameters.

respectively, have almost no chance of being correct, so the user must specify them. The default value for ORAD, the cathode center, is at CATHODEZ, the first boundary point as defined for the general cathode in Section VI.E. The starting step ST, is the value used for the thickness of the Langmuir pill boxes. As in the START = 'GENERAL' case, in cylindrical coordinates these pill boxes are annular rings and the current is that current in a one radian segment of that ring. The current is calculated as in Eqs. 18-20 using the geometry of Fig. 5. Figure 6 is the plotted output of the sample problem of Fig. 2 using START = 'SPHERE'.

In rectangular coordinates, START = 'SPHERE' operates with the same input and the same geometry to calculate the current per mesh unit in the direction normal to the plane of the paper. Again, as in START = 'GENERAL' Eqs. 16-17 are used according to Ref. 8.

Immediately after printing the headings the spherical cathode routines print a message:

ITERATION NO. X, I = X.X MICROAMPS, PERVEANCE = X.X MICROPERV.

The current and perveance printed are those calculated according to the fields and geometry by the appropriate equations as indicated above. In other words, these are the unnormalized values. After printing this message, the program averages the perveance according to the method described under PERVO in Section VI.Z. The initial currents that are printed out with the initial conditions reflect this averaging process. Between the initial and final conditions, the same message as above is printed, except with the normalized values for current and perveance. As in START = 'GENERAL' the currents printed with the final conditions

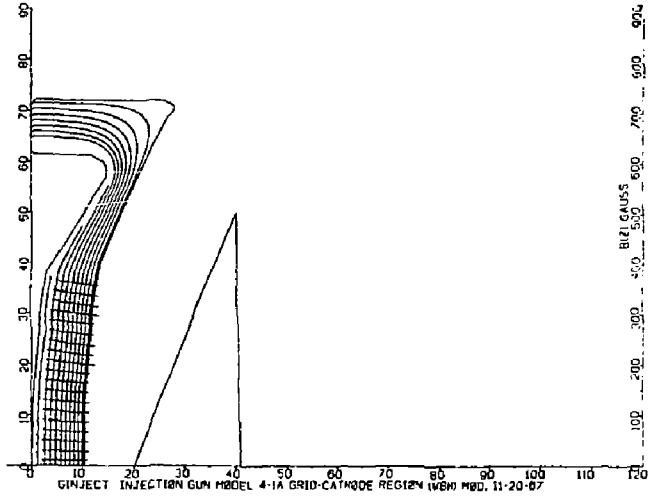


Fig. 6. Plotted output of sample problem shown in Fig. 2.

are divided by the initial radius (if in cylindrical coordinates) and thus give a measure of uniformity of cathode loading.

The special case of magnetic fields reaching the cathode, i.e., "immersed flow" is treated by both SPHERE and GENERAL according to Busch's theorem.[12] The program must use magnetic fields on the cathode and on the starting surface to integrate the azimuthal motion through the gap between the cathode and the starting surface. If there is any inconsistency in the off-axis magnetic fields within ± 6 mesh units of the entire range of the starting area, then peculiar bunching of the rays will occur. That is why the proper use of MAGORD and the careful input of fields near the cathode were stressed in Section VI.D. Fortunately, any problem of this sort becomes immediately obvious on examination of either the starting conditions or the plots.

C. Card Starting

The program starting instructions are as follows:

```
START = 'CARDS'              START = 'GENERAL'       CARD STARTING
   ZO = X.XX                   ZO = 0.0          OLD ORIGIN IN NEW FRAME
   SKAL = X.XX                 SKAL = 1.0        OLD MESH/NEW MESH
   HAVE UP TO MAXRAY DATA CARDS WITH (1 INTEGER, 6 FLOAT PT.) NO., R,
Z, EMERGY (EV), ANGLE (RADIANS), CURRENT (MICROAMPERES IN ONE RADIAN
SEGMENT), TRANSVERSE ANGLE, TRANSVERSE POSITION (PHI).  FORMAT I5, 5X,
7F10.5.  OLD USERS GETTING THE NEW VERSION OF THE PROGRAM SHOULD NOTE
THE CHANGE TO TRANSVERSE ANGLE AND TOTAL KINETIC ENERGY.
   STOP READING WITH RAY NO. GREATER THAN MAXRAY.
IF RECTANGULAR COORDINATES:
   PHI IS TRANSVERSE POSITION IN MESH UNITS.
   CURRENT IS MICROAMPERES IN ONE MESH UNIT DEEP SEGMENT.
****SPECIAL TESTS IN RAINST; CROSSING OR 3-D CPACE CHRAGE**
IRAT=1                       IRAT=0                  3-D SPACE CHARGE
IRAT=2                       IRAT=0                  CROSSING DETECTION
   USE OF NEGATIVE RAY NUMBERS:
A) IF IRAT=1              (3-D SPACE CHARGE)
   1) MAKE RAY NUMBERS NEGATIVE FOR BEAM EDGE CARDS.
USE BEAM EDGE CARDS (I0=0) TO STIMULATE SPACE CHARGE SPREADING
OF A CYLINDRICAL BEAM OF CURRENT I AND RADIUS R IN RECT. COORD.
```



```
   PAIRS OF BEAM EDGE CARDS PRECEDE SETS OF RAY CARDS DEFINING
PART OF BEAM FOR WHICH 3-D SPACE CHARGE SPREADING IS TO BE SIMULATED.
SEVERAL PARTS, DIFFERENTIATED BY SELECTED ATTRIBUTES; E.G., ENERGY
ALPHA OR RADIUS. CAN BE USED SIMULTANEOUSLY WITH ANY NUMBER OF RAYS
IN EACH PART.  END OF PART IS DEFINED BY NEXT RAY WITH NEGATIVE RAY
NUMBER, WHICH BEGINS THE NEXT PART.
   TO SIMULATE CYLINDRICAL BEAM SPACE CHARGE IN RECT. COORD. MAKE
   CURRENT PER MESH UNIT, I' = I/(PI * R) INSTEAD OF I' = 2 * I/(PI * R)
   WHICH WOULD HAVE THE SAME CURRENT DENSITY.  IN OTHER WORDS, MAKE
   I'(K) = I(K)/(2 * R(K)) INSTEAD OF I(K)/R(K).  NOTE THAT THIS REQUIRES
   TWICE AS MANY RAYS AS FOR CYLINDRICAL BEAM WITH SYMMETRY.
   BEAM EDGE CARDS (RAY < 0) APPLY TO OFF-AXIS PENCIL IN CYL. COORD.
```

The START = 'CARDS' mode uses data cards for the initial conditions rather than computing the initial conditions from a thermionic model. There are several typical applications for this feature that will be described in some detail. These are:

1. The simplest case of user specified data.
2. Use of cards generated by a preceding run to restart in a new segment of the same problem.
3. Study thermal and other perturbing influences on a beam.
4. Rectangular coordinate application with a cylindrical beam, including cylindrical space charge and off-axis bends.

1. Format for User Specified Data

If START = 'CARDS' has been selected, the program will respond by printing a table of appropriate parameters: STEP, NS, Z(0), SKAL, UNIT. Following the end of the NAMELIST input &END card, the program will expect to read up to MAXRAY cards with the starting data. A card with ray number greater than MAXRAY will terminate this input. If MAXRAY cards are present, the termination card should be used anyway. However, no effort should be made to make MAXRAY agree with the number of cards used, so long as it is big enough. The computer can, after all, count better than most humans.

Data to be entered on the ray cards consist of a ray number and the initial values for R, Z, ENERGY, ANGLE, CURRENT, TRANSVERSE ANGLE and TRANSVERSE POSITION. The format is I5, X5, 7F (10,5).

(a) Ray Number: the ray number is only included for user convenience, and for the termination purpose described above. Rays are numbered by the program, sequentially as the cards are read in. Negative ray numbers have special implications that will be described below.

(b) R: the initial radial position in mesh units.

(c) Z: the initial axial position in mesh units.

(d) ENERGY (EV): The initial kinetic energy of the particle in electron volts. It should be obvious, but sometimes requires stating, that ENERGY has nothing whatever to do with the potential values on the boundaries, or on the potential at which the ray tracing starts. For ray tracing, only fields are important, not absolute potentials.

(e) ANGLE: the initial angle that the ray makes with respect to the z-axis, in radians.

(f) CURRENT: the current in microamperes for a one radian segment of that ray. In rectangular coordinates, it is for a one mesh unit deep segment.

(g) TRANSVERSE ANGLE: the angle normal to the R-Z plane.

(h) PHI: the initial transverse position. In rectangular symmetry, PHI is a linear coordinate, measured in mesh units. In cylindrical symmetry, PHI is the azimuthal position in radians.

2. Use of Program Generated Cards

At the end of a run, the program generates a set of cards with the final conditions of each ray according to the above format. These cards may be punched, or saved as a data set in card format on a direct access device. If it is planned to use the cards in a subsequent run, it is only necessary to be sure they are saved somehow. In a pinch, the same data are printed in the final conditions of the output and can be hand punched.

Typically, these cards are intended to be used in a subsequent segment of a problem. Thus the results of the sample problem, Fig. 2, are intended to be used in the complete gun with card starting just past the grid. Between runs, it is normal to expect that a different scale and origin will be used, otherwise there is not much reason for the second run. The companion parameters ZO and SKAL are used to modify the data, as read in on the cards, as follows:

ZO = X.XX	ZO = 0.0	OLD ORIGIN IN NEW FRAME
SKAL = X.XX	SKAL = 1.0	OLD MESH/NEW MESH

In words, if the first problem is plotted on the same graph with the second problem, then the origin of the first problem will be found displaced left or right by ZO mesh units in the new coordinate system. Usually ZO is negative. SKAL is interpreted as the ratio of sizes of mesh units (in meters). Thus a problem in which many mesh units were used to calculate cathode conditions will have a relatively smaller mesh than the follow on problem and SKAL < 1.0 in this example.

3. Thermal Effects

```
SUBROUTINE THERM IS CALLED IF THE PARAMETER TC > 0.
TC=XXXX.X                TC = 0              KELVIN TEMP. OF CATHODE
TWO MODELS ARE INCLUDED IN THIS VERSION
KRAY=3                   KRAY=1              THREE RAY SPLIT
KRAY=5                   KRAY=1              FIVE RAY SPLIT
THREE RAY SPLIT PUTS CURRENTS IN  1-2-1  RATIO WITH 2 PARTS IN
UNDEFLECTED RAY AND 1 PART EACH IN RAYS WITH V(PERP)=SQRT(2KT/M)
IN R-Z PLANE, UP AND DOWN RELATIVE TO UNDEFLECTED RAY.

FIVE RAY SPLIT PUTS CURRENTS IN  1-9-0-9-1  RATIO WITH
V(PERP)=2*SQRT(2KT/M) FOR 1 PART RAYS AND V(PERP)=1*SQRT(2KT/M)
FOR 9 PART RAYS.  NO CURRENT IN CENTER RAY.

USERS SHOULD FEEL FREE TO MODIFY SUBROUTINE THERM.
THERM CAN BE CALLED FOR START='SPHERE', 'GENERAL', 'CARDS'.
OR 'GENCARD'.
IT CANNOT BE USED FOR START='CARDS' WITH SAVE=2.
```

4. Rectangular Coordinates with Cylindrical Beams

The basic assumption in rectangular coordinates is that the beam consists of a sheet extending infinitely in the directions in-and-out of the problem. The space charge forces on such a beam are much greater than in cylindrical symmetry because the field does not fall off by 1/R. However, if the current is properly reduced, the transverse space charge forces can be made the same as they would be for a cylindrical beam. Further reductions in the current can compensate for further expansion of the beam.

Consider first a uniform density cylindrical beam of total current I and radius R. The current density is $J = I/\pi R^2$. If one wished to have a rectangular symmetry beam of thickness 2R at the same current density, the total current per unit length would be

$$I' = 2RJ = 2I/\pi R \qquad \text{(equal densities)} \qquad (22)$$

One could divide I' by some integer n and make n rays, suitably spaced, each with a current of I'/n. If one wishes to use starting data from a previous run, then each ray has a current per unit length I(K)/R(K). Unless the rectangular beam has reflection symmetry on the z-axis, there would have to be twice as many trajectories created as in cylindrical symmetry to represent both halves of the beam.

Consider now a particle of charge e on the edge of a cylindrical beam of radius R and current I. The radial space charge force on the particle is

$$md^2R/dt^2 = eI/(2\pi R\epsilon_0 \dot{z}) \qquad (23)$$

The force on the similar particle next to a current sheet in rectangular symmetry is

$$md^2y/dt^2 = eI'/(2\dot{z}\epsilon_0) \qquad (24)$$

To make $d^2R/dt^2 = d^2y/dt^2$ we have only to require

$$I' = I/\pi R \qquad \text{(equal forces)} \qquad (25)$$

This is just one half of the result for equal densities in Eq. (21). Thus, if the results from the previous run were treated as described above, except divided by two, then the initial space charge forces on the rays would be the same as in cylindrical coordinates.

A special feature allows the user to designate groups of rays, as few as one per group, to be bounded by "beam edge" cards which do not carry current. As the beam edge cards spread apart, the current on all

rays within the group is reduced proportionately. The groups may cross or overlap, but should not cross their own beam edge rays. The initial conditions of the beam edge rays can be chosen so that they do not cross the rays of the group. Beam edge cards are designated by being inserted, with negative ray numbers, in pairs just before the members of their group. Successive groups would thus be separated by the pair of beam edge cards for the next group.

Beam edge cards may also be used in cylindrical coordinates. In this case, the effect would be of an off-axis pencil beam, i.e., not an annular ring. Assuming that the thickness of the pencil is small compared to the radial displacement, the same factor of one-half should be applied to the initial currents as was derived for rectangular coordinates.

B) IF IRAT=2 (R-Z AND PHI CROSSOVERS)
1) R-Z: MAKE RAY NUMBERS NEGATIVE FOR SEQUENTIAL RAYS FOR WHICH FINAL CROSSOVER SHOULD BE DETECTED. CROSSINGS WILL BE LISTED AND PLOTTED. NEGATIVE RAY NUMBERS SHOULD BE IN PAIRS. TO FIND CROSSOVERS WITH Z AXIS, RUN A RAY WITH R=0,ALPHA=0 PRECEDING THE RAY TO TEST AXIS CROSSING.
2) PHI: LEAVE RAY NUMBERS POSITIVE FOR TRANSVERSE RAYS TO DETECT LAST CROSSING OF PHI=PI * INTEGER.

A special application of beam edge cards is to specifically detect crossovers. For this application, the beam edge control code is set to IRAT=2 in &INPUT5. The program instruction comments appear above. This feature is used to find the locus of foci to determine the position of the scintillator surface in image intensifier tubes. No space charge is involved. Pairs of trajectories, started sequentially from the same point with different initial conditions (energy and direction) are focused to a crossing, which must be located exactly. The program finds such crossovers and prints a table of their coordinates.

H. Laplace's Equation Applications

START = 'LAPLACE' START = 'GENERAL' NO RAY TRACING

NS = X NS = 7 NUMBER OF LAPLACE CYCLES

ADD DATA CARDS WITH (R,Z SPACE CHARGE) FOR NON-ZERO POINTS. END POINT INPUT BY R > RLIM.

Laplace's equation has many applications besides solving electrostatic potential problems. Some examples are temperature distributions and magnetic fields.

As a reminder, by Laplace's equation one usually means $\nabla^2\phi = 0$ while Poisson's equation is $\nabla^2\phi = \rho$. The program always solves Poisson's equation but with $\rho = 0$ on the first iteration. However, if one selects START = 'LAPLACE', one can then add data cards with the coordinates (R,A), and the right hand or space charge term for any non-zero point. These data are appended after the end of the starting namelist and are terminated by R > RLIM.

The program will then cycle for NS cycles on just these data, with no ray tracing. It prints the potential map or POTLIST before and after the last cycle to show how things may be changing. Following the last cycle, the program prints a list of the fields, i.e., the derivatives of the potentials, on all the boundaries. Fields at specified interior points can be obtained by making a dummy boundary go through such points. Dummy boundary points have DELTAR = DELTAZ = 2.0 and can be fitted according to the same rules as Neumann boundaries, i.e., along mesh lines. The fields are normalized to 100% of the field on the first boundary point. Choose it carefully, i.e., not where the field is near zero.

To do ray tracing with the arbitrary space charge solution found by LAPLACE, it is simply necessary to set SAVE=1 in &INPUT5 of the first, LAPLACE, problem followed by a second problem, without boundary data, but with ray tracing starting instructions. See the discussion under SAVE=1 in Section VI.A.12.

I. Dielectric Boundaries

The input provision for special boundary points, described in Section V can be used for the particular case of a dielectric boundary. The difference equations are only affected on the boundary of the dielectric. The normal method of using this feature is to specify dummy boundary points, i.e., points with DELTAR = DELTA2 = 2.0, which can be put in point-by-point or with the fitting (three-point) method as if the points were Neumann boundaries. That is, they must lie on mesh lines.

The difference equations were derived by Seeger[13] for the special cases of horizontal and vertical dielectric boundaries. These relatively simple cases are sufficient for most applications because the actual position and angle of even a curved dielectric are relatively less important to the fields in the vicinity than the fact that the boundary is located nearby. Thus a good approximation results from a stepwise simulation of the dielectric and a small displacement to the nearest mesh point does very little to the fields a few mesh units away.

The coefficients of the difference equation are given by Eq. (3) in Section IV, and can be expressed as:

$$\begin{aligned} &\text{LEFT} = \text{RIGHT} = R && \text{(Vacuum)} \\ &\text{UP} = R + 1/2 \\ &\text{DOWN} = R - 1/2 && (26) \end{aligned}$$

For a horizontal dielectric, where ε_1 is the dielectric constant for the lower region and ε_2 is the constant for the upper region, the coefficients become:

$$\begin{aligned} &\text{LEFT} = \text{RIGHT} = [\varepsilon_1(R - 1/2) + \varepsilon_2(R + 1/2)]/2 && \text{(horizontal)} \\ &\text{UP} = \varepsilon_2(R + 1/2) \\ &\text{DOWN} = \varepsilon_1(R - 1/2) && (27) \end{aligned}$$

For a vertical dielectric boundary, the coefficients become

$$\begin{aligned} &\text{LEFT} = \varepsilon_1 R \qquad \text{RIGHT} = \varepsilon_2 R && \text{(vertical)} \\ &\text{UP} = (\varepsilon_1 + \varepsilon_2)(R + 1/2)/2 \\ &\text{DOWN} = (\varepsilon_1 + \varepsilon_2)(R - 1/2)/2 && (28) \end{aligned}$$

where ε_1 is the dielectric constant for the left side region and ε_2 is the constant for the right side region. For rectangular coordinates, set all the R's and $(R \pm 1/2)$'s to unity.

The terms LEFT, RIGHT, UP and DOWN refer to the points, 1, 2, 3 and 5 respectively in Fig. 1. The notes summarizing Eqs. (27) and (28) in the program instructions are reprinted below;

SPECIAL BOUNDARY POINTS (INCLUDING GENERAL NEUMANN BOUNDARIES) USE 999 IN COLUMNS 3-5 TO END BOUNDARY INPUT. BOUNDARY MUST INCLUDE ALL POINTS TO BE USED AND ALL POT NUMBERS. THEN INCLUDE ANY NUMBER OF CARDS WITH R, Z AND FOUR DIFFERENCE NUMBERS FOR LEFT, RIGHT, UP AND DOWN,

SEQUENTIALLY. NUMBERS SHOULD ADD TO 4 * R OR 4 IF RECTANGULAR COORDINATES. END WITH R > RLIM. FOR GENERAL NEUMANN, SEE APPENDIX II. TERMS ARE 4 * TAN@/1+TAN@) AND 4/TAN@) WHERE TAN@ < 1.

HORIZONTAL DIELECTRIC BOUNDARY:

LEFT = RIGHT = (EL * (R - 0.5) + E2 * (R+0.5))/2

UP = E2 * (R + 0.5)

DOWN = E1 * (R - 0.5)

where E1 OR E2 = 1.0 FOR VACUUM AND E2 IS UPPER 'MATERIAL'.

VERTICAL DIELECTRIC BOUNDARY:

LEFT = E1 * R RIGHT = E2 * R

UP = (E1 + E2) * (R + 0.5)/2

DOWN = (E1 + E2) * (R - 0.5)/2

WHERE E2 IS RIGHT HAND 'MATERIAL'.

VII. TRAJECTORY CALCULATIONS

The program uses a fourth-order Runge-Kutta method of solving the relativistic differential equations given below. Suitable substitutions are used to reduce the three second-order equations to six first-order differential equations.

The independent variable is time but the time interval is calculated from the allowed iteration step and the velocity. It is necessary to use fairly short steps because of the auxiliary calculations that must be made at each mesh unit. Thus it is generally not helpful

to use any self-checking "corrector" solving routine. If some unusual application requires shorter iteration steps, the results usually show this by their internal inconsistency.

The relativistic differential equations are derived in Appendix I and are

$$\ddot{Z} = \alpha(1 - \beta^2)^{1/2}\left[-E_z(1 - \dot{Z}^2) + \ddot{Z}RE_r + \ddot{Z}AE_\varphi - cR\dot{B}_\varphi + cA\dot{B}_r\right] , \quad (29)$$

$$\ddot{R} = \alpha(1 - \beta^2)^{1/2}\left[-E_r(1 - \dot{R}^2) + \ddot{Z}RE_z + \ddot{R}AE_\varphi + cZ\dot{B}_\varphi - cA\dot{B}_z\right] + \frac{\dot{A}^2}{R} \quad (30)$$

and

$$\ddot{A} = \alpha(1 - \beta^2)^{1/2}\left[-E_\varphi(1 - \dot{A}^2) + \ddot{Z}AE_z + \ddot{R}AE_r - cZ\dot{B}_r - cR\dot{B}_r\right] - \frac{\ddot{R}\dot{A}}{R} . \quad (31)$$

where

$$\beta^2 = \dot{Z}^2 + \dot{R}^2 + \dot{A}^2 \quad \text{and} \quad \beta = v/c \quad (32)$$

The constant $\alpha = e\lambda/m_0c^2$ where e is the magnitude of the electron charge (the "-" sign is in the equations), m_0c^2 is the rest energy of the electron and λ is the constant of proportionality between the real coordinates and the dimensionless coordinates. Thus

$$z = \lambda Z, \qquad r = \lambda R, \qquad a = \lambda A \qquad \text{and} \quad ct = \lambda T \quad (33)$$

By an arbitrary choice, $\lambda = 5.11 \times 10^5$ mesh units so that $\alpha = 1.0$ mesh unit per volt. Inspection of the differential equations shows that they are dimensionally correct if the electric fields are specified in volts per mesh unit.

Dimensionally E = vB, so that in mksa units E is in volts per meter, v is in meters per second and B is in webers per meter.[2] Then cB has units of volts per meter. To convert to program fields of volts per mesh unit, fields are multiplied by the value UNIT in meters per mesh unit. Magnetic field input to the program is in gauss, which is the common engineering unit, and is internally converted to webers/meter^2.

The azimuthal magnetic field B_φ comes from the current in the electron beam and is called the self-magnetic field of the beam. The magnetic field created by an axial current is

$$B_\varphi = \frac{\mu_0}{2\pi} \frac{I}{r} \text{ webers/meter}^2 \qquad (34)$$

The field is assumed to be due to an infinite conductor which is a pretty good approximation in the area in which the field is significant. After multiplying B_φ by the scale factor and expressing r in meters which requires multiplying r by the scale factor also, the scale factor cancels as might be expected. Thus the scale factor only enters for external magnetic fields. The current I in Eq. (34) is the summation of the current in the trajectories at lower radii than the trajectory being calculated, but including the one being calculated.

Two field components are neglected. The azimuthal electric field is neglected because of the axial symmetry assumed. The axial magnetic field can have a contribution from the beam due to azimuthal velocity of the beam. The magnitude has been shown to be less than one gauss in most practical cases and so is neglected.

The space charge is calculated to supply the right side of Poisson's equation which is

$$\nabla^2 V = \frac{\rho}{\epsilon_0} = \frac{J}{vc_0} \qquad (35)$$

The element of area for J is (r × 1.0) mesh units^2 where r is the particle radius. The velocity is only the Z-component since the space charge is being spread between adjacent points on the same column. The one mesh unit space between adjacent points accounts for the 1.0 in the area expression above.

In the finite difference form, Eq. (3) replaces Eq. (35), and the right hand side becomes

$$RO = \frac{36\pi \times 10^9 \; IO(K) \times 10^{-6}}{ABS(ZDOT) \times 3 \times 10^8} = (3.77^{**} - 4) \; IO(K)/ABS(ZDOT) \qquad (36)$$

where RO is to be spread between two points in inverse ratio to the distance the ray is between them, IO(K) is the current in the one radian segment of the ray (in microamperes) and ZDOT is the velocity in units of c. If the angle of inclination, dR/dZ, exceeds 45°, the calculation is made for RDOT. The absolute value of ZDOT is used to allow a negative ZDOT. The explicit value of R in Eq. (3) is canceled by the R which would convert the current to current density, thus avoiding special problems as R → 0.

In practice, however, there are still some space charge problems near the axis. In rectangular coordinates, if the axis is a plane of symmetry, then any trajectory between R = 0 and R = 1 has a mirror image

between R = 0 and R = -1. To account for all the space charge on the axis, the calculated charge is doubled. In cylindrical coordinates, it has been found necessary to multiply the axial space charge by an empirical factor of 5.5. While no satisfactory explanation of this has ever appeared, the behavior of ideal laminar beams in test problems is markedly improved and highly convergent beams appear to behave as expected.

VIII. TRAJECTORY ANALYSIS

The program does some analysis of the quality of the beam resulting in a quantity which is similar to the phase volume, or emittance, of the beam. For those not familiar with the concept of phase volume, the material presented by Steffan[13] is a good introduction. The direct application of the concept of phase volume to electron guns was derived by Miller.[14]

The simplest formulation of phase volume is to consider the area of an ellipse plotted in dr/dz vs. r. Assume that the beam (e.g., the first standard deviation) fills this ellipse. Subsequent drifting and focusing can be shown to affect only the aspect ratio of the ellipse, and the rotation of the major axis, but not the area. The ellipse can become unrecognizable through nonlinear elements.

At the end of each computer run, two extra plots are generated. One is a plot of current density as a function of final radius, i.e., the beam profile. The second plot is a point plot of the location in dr/dz vs. r of the final conditions of each ray. Figure 7 is the plot in dr/dz for the sample problem in Fig. 2. Using this second plot, the

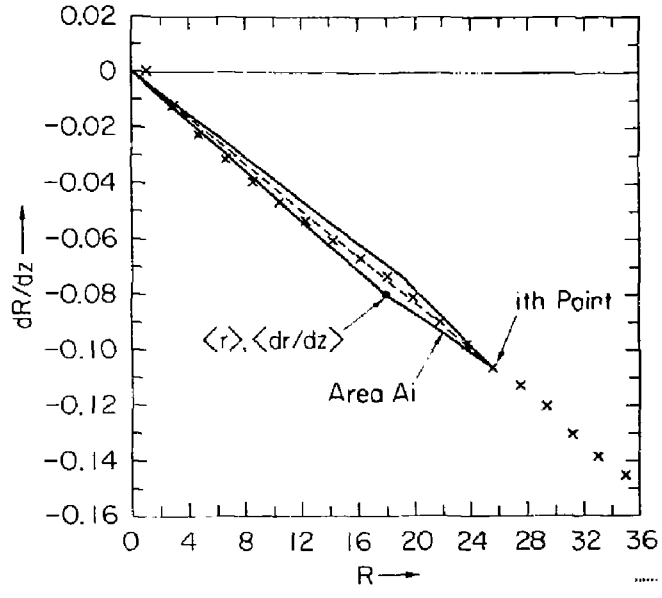


Fig. 7. Phase space calculation for problem shown in Fig. 2.

effective phase area is calculated at the end of each run according to the method described by Miller.[14] First the center of the distribution is calculated, with suitable weighting for the current of each ray. This results in a location <r>, <dr/dz> in the half-plane. Then the area A_i for the ith trajectory is calculated as the weighted cross product between the ith point, r_i, (dr_i/dz), and the center of the distribution. The resulting expression which is used in the program is

$$A_x \equiv \frac{3\pi^2}{2}\left[\frac{\sum_i I_i\left\{\frac{dr_i}{dz}\left(\sum_j I_j r_j\right) - r_i\left(\sum_j I_j \frac{dr_j}{dz}\right)\right\}^2}{\left(\sum_i I_i\right)^3}\right]^{1/2} \tag{37}$$

This definition for the emittance area of a number of discrete points has the following desirable characteristics:

1. It vanishes when the points lie on a straight line through the origin.
2. It approaches the area of the ellipse for a very large number of equally weighted points uniformly distributed in the interior of an ellipse.
3. It is invariant under linear transformations which conserve phase area such as that representing an aberrationless lens.

When multiplied by the particle momentum, Eq. (37) retains the same invariance through subsequent acceleration. That is, transverse momentum times radius is conserved.

APPENDIX I

DERIVATION OF EQUATIONS OF MOTION*

The equations of motion are derived* from the Lorentz force equation

$$\frac{d(m\vec{v})}{dt} = -e(\vec{E} + \vec{v} \times \vec{B}), \tag{1}$$

where e is the magnitude of the charge of an electron. The electron velocity vector v, expressed in cylindrical coordinates is

$$\vec{v} = u_z \dot{z} + u_r \dot{r} + u_\varphi \dot{a} . \tag{2}$$

Here u_z, u_r and u_φ are unit vectors and $\dot{a} = r\dot{\varphi}$ is the azimuthal or peripheral velocity. The left side of Eq. (1) can be found from

$$\frac{d(m\vec{v})}{dt} = \frac{d}{dt}\left(m_o\left(1 - \frac{v^2}{c^2}\right)^{-\frac{1}{2}}\vec{v}\right) \tag{3}$$

where m_o is the electron rest mass. Differentiating Eq. (3) yields

$$\frac{d(m\vec{v})}{dt} = m_o\left(1 - \frac{v^2}{c^2}\right)^{-3/2}\left[\frac{v}{c^2}\frac{dv}{dt}\vec{v} + \left(1 - \frac{v^2}{c^2}\right)\frac{d\vec{v}}{dt}\right] \tag{4}$$

where

$$\frac{d\vec{v}}{dt} = u_z \ddot{z} + u_r (\ddot{r} - r\dot{\varphi}^2) + u_\varphi (2\dot{r}\dot{\varphi} + r\ddot{\varphi}) \tag{5}$$

which becomes

$$\frac{d\vec{v}}{dt} = u_z\ddot{z} + u_r(\ddot{r} - \dot{a}^2/r) + u_p(\ddot{ra}/r + \ddot{a}) . \tag{6}$$

From $v = (\dot{z}^2 + \dot{r}^2 + \dot{a}^2)^{\frac{1}{2}}$ where v is the scalar velocity, (7)

we have

$$\frac{dv}{dt} = \frac{1}{v}(\dot{z}\ddot{z} + \dot{r}\ddot{r} + \dot{a}\ddot{a}) . \tag{8}$$

---

*This derivation was suggested by Dr. Gene Lang in a private communication.

Substituting Eqs. (6), (7) and (8) in Eq. (4) yields

$$\frac{d(m\vec{v})}{dt} = m_o\left(1-\frac{v^2}{c^2}\right)^{-3/2}\left[\frac{1}{c^2}(\dot{z}\ddot{z}+\dot{r}\ddot{r}+\dot{a}\ddot{a})(u_z\dot{z}+u_r\dot{r}+u_\varphi\dot{a})\right.$$
$$\left.+\left(1-\frac{v^2}{c^2}\right)\left\{u_z\ddot{z}+u_r(\ddot{r}-\dot{a}^2/r)+u_\varphi(\dot{r}\dot{a}/r+\ddot{a})\right\}\right]. \qquad (9)$$

Equation (9) can be expanded and grouped by vector components yielding

$$\frac{d(m\vec{v})}{dt} = m_o\left(1-\frac{v^2}{c^2}\right)^{-3/2}\left[u_z\left\{\frac{1}{c^2}\dot{z}(\dot{r}\ddot{r}+\dot{a}\ddot{a})+\ddot{z}\left(1-\frac{v^2}{c^2}+\frac{\dot{z}^2}{c^2}\right)\right\}\right.$$
$$+u_r\left\{\frac{1}{c^2}\dot{r}(\dot{z}\ddot{z}+\dot{a}\ddot{a})-\frac{\dot{a}^2}{r}\left(1-\frac{v^2}{c^2}\right)+\ddot{r}\left(1-\frac{v^2}{c^2}+\frac{\dot{r}^2}{c^2}\right)\right\} \qquad (10)$$
$$\left.+u_\varphi\left\{\frac{1}{c^2}\dot{a}(\dot{z}\ddot{z}+\dot{r}\ddot{r})+\frac{\dot{r}\dot{a}}{r}\left(1-\frac{v^2}{c^2}\right)+\ddot{a}\left(1-\frac{v^2}{c^2}+\frac{\dot{a}^2}{c^2}\right)\right\}\right].$$

A similar vector component expansion can be made for the right side of Eq. (1) yielding

$$\frac{d(m\vec{v})}{dt} = -e\left[u_z(E_z+\dot{r}B_\varphi-\dot{a}B_r)+u_r(E_r+\dot{a}B_z-\dot{z}B_\varphi)+u_\varphi(E_\varphi+\dot{z}B_r-\dot{r}B_z)\right]. \qquad (11)$$

Equating vector components we have finally

$$m_o\left(1-\frac{v^2}{c^2}\right)^{-3/2}\left\{\left(1-\frac{v^2}{c^2}+\frac{\dot{z}^2}{c^2}\right)\ddot{z}+\frac{1}{c^2}\dot{z}\dot{r}\ddot{r}+\frac{1}{c^2}\dot{z}\dot{a}\ddot{a}\right\} = -e(E_z+\dot{r}B_\varphi-\dot{a}B_r), \qquad (12)$$

$$m_o\left(1-\frac{v^2}{c^2}\right)^{-3/2}\left\{\frac{1}{c^2}\dot{r}\ddot{z}\ddot{z}+\left(1-\frac{v^2}{c^2}+\frac{\dot{r}^2}{c^2}\right)\ddot{r}+\frac{1}{c^2}\dot{r}\dot{a}\ddot{a}-\frac{\dot{a}^2}{r}\left(1-\frac{v^2}{c^2}\right)\right\}$$
$$= -e(E_r-\dot{z}B_\varphi+\dot{a}B_z) \qquad (13)$$

and

$$m_o\left(1-\frac{v^2}{c^2}\right)^{-3/2}\left\{\frac{1}{c^2}\dot{a}\dot{z}\ddot{z}+\frac{1}{c^2}\dot{a}\dot{r}\ddot{r}+\left(1-\frac{v^2}{c^2}+\frac{\dot{a}^2}{c^2}\right)\ddot{a}+\frac{\dot{r}\dot{a}}{r}\left(1-\frac{v^2}{c^2}\right)\right\}$$
$$= -e(E_\varphi+\dot{z}B_r-\dot{r}B_z). \qquad (14)$$

For computer programming it is convenient to express the variables in a normalized form. Accordingly, we let

$$z = \lambda Z,\ r = \lambda R,\ a = \lambda A \text{ and } ct = \lambda T. \qquad (15)$$

We differentiate with respect to $T = \frac{ct}{\lambda}$ to get

$$\dot{z} = c\dot{Z},\ \ddot{z} = \frac{c^2\ddot{Z}}{\lambda},$$
$$\dot{r} = c\dot{R},\ \ddot{r} = \frac{c^2\ddot{R}}{\lambda}, \qquad (16)$$
and
$$\dot{a} = c\dot{A},\ \ddot{a} = \frac{c^2\ddot{A}}{\lambda}.$$

From the definitions in Eqs. (16) it follows that

$$\beta^2 = \frac{v^2}{c^2} = \dot{Z}^2+\dot{R}^2+\dot{A}^2. \qquad (17)$$

Making the normalizing substitutions in Eqs. (12), (13), and (17) yields

$$\frac{m_o c^2}{\lambda(1-\beta^2)^{3/2}}\left[(1-\beta^2+\dot{Z}^2)\ddot{Z}+\dot{Z}\dot{R}\ddot{R}+\dot{Z}\dot{A}\ddot{A}\right] = -e\left[E_z+c\dot{R}B_\varphi-c\dot{A}B_r\right], \qquad (18)$$

$$\frac{m_o c^2}{\lambda(1-\beta^2)^{3/2}}\left[\dot{R}\dot{Z}\ddot{Z}+(1-\beta^2+\dot{R}^2)\ddot{R}+\dot{R}\dot{A}\ddot{A}-\frac{\dot{A}^2}{R}(1-\beta^2)\right] = -e\left[E_r-c\dot{Z}B_\varphi+c\dot{A}B_z\right] \qquad (19)$$

and

$$\frac{m_o c^2}{\lambda(1-\beta^2)^{3/2}}\left[\dot{\beta}\ddot{Z} + \dot{A}\ddot{R} + (1-\beta^2+\dot{A}^2)\ddot{K} + \frac{\dot{R}\dot{A}}{R}(1-\beta^2)\right] = -e\left[E_\varphi + c\dot{Z}B_r - c\dot{R}B_z\right]. \quad (20)$$

Our goal is to get separated equations solved for the second order derivative of each of the orthogonal variables. To solve the equations, we arrange them in the form

$$\begin{aligned} A_1\ddot{Z} + B_1\ddot{R} + C_1\ddot{A} &= D_1 \\ A_2\ddot{Z} + B_2\ddot{R} + C_2\ddot{A} &= D_2 \\ A_3\ddot{Z} + B_3\ddot{R} + C_3\ddot{A} &= D_3 \end{aligned} \quad (21)$$

and apply the standard determinant method of solving simultaneous equations. Rearranging Eqs. (18), (19) and (20) in the form of Eq. (21) yields

$$(1-\beta^2+\dot{Z}^2)\ddot{Z} + \dot{Z}\dot{R}\ddot{R} + \dot{Z}\dot{A}\ddot{A} = \frac{-e\lambda}{m_o c^2}(1-\beta^2)^{3/2}(E_z + c\dot{R}B_\varphi - c\dot{A}B_r), \quad (22)$$

$$\dot{R}\dot{Z}\ddot{Z} + (1-\beta^2+\dot{R}^2)\ddot{R} + \dot{R}\dot{A}\ddot{A} = (1-\beta^2)\frac{\dot{A}^2}{R} - \frac{e\lambda}{m_o c^2} \times (E_r - c\dot{Z}B_\varphi + c\dot{A}B_z)(1-\beta^2)^{3/2}, \quad (23)$$

and

$$\dot{A}\dot{Z}\ddot{Z} + \dot{A}\dot{R}\ddot{R} + (1-\beta^2+\dot{A}^2)\ddot{K} = -(1-\beta^2)\frac{\dot{R}\dot{A}}{R} - \frac{c\lambda}{m_o c^2} \times (E_\varphi + c\dot{Z}B_r - c\dot{R}B_z)(1-\beta^2)^{3/2}. \quad (24)$$

The determinant of the coefficients is

$$\begin{aligned} \Delta = (1-\beta^2+\dot{Z}^2)&\left[(1-\beta^2+\dot{R}^2)(1-\beta^2+\dot{A}^2) - \dot{A}^2\dot{R}^2\right] + \dot{Z}\dot{R}\left[\dot{R}\dot{Z}\dot{A}^2 - \dot{Z}\dot{R}(1-\beta^2+\dot{A}^2)\right] \\ &+ \dot{Z}\dot{A}\left[\dot{Z}\dot{R}^2\dot{A} - \dot{A}\dot{Z}(1-\beta^2+\dot{R}^2)\right] - (1-\beta^2+\dot{Z}^2)(1-\beta^2)(1-\beta^2+\dot{A}^2+\dot{R}^2) \\ &- \dot{Z}^2\dot{R}^2(1-\beta^2) - \dot{Z}^2\dot{A}^2(1-\beta^2) - (1-\beta^2)^2(1-\beta^2+\dot{Z}^2+\dot{R}^2+\dot{A}^2) \end{aligned}$$

which is simply

$$\Delta = (1-\beta^2)^2 . \quad (25)$$

It is convenient to let $\alpha = e\lambda/E_o c^2$. The axial acceleration $\ddot{Z}$, is given by

$$\Delta\ddot{Z} = D_1(B_2C_3 - C_2B_3) + D_2(C_1B_3 - B_1C_3) + D_3(B_1C_2 - C_1B_2)$$

which becomes

$$\begin{aligned} (1-\beta^2)^2\ddot{Z} = &\left[-\alpha(1-\beta^2)^{3/2}(E_z + c\dot{R}B_\varphi - c\dot{A}B_r)\right]\left[(1-\beta^2+\dot{R}^2) \times (1-\beta^2+\dot{A}^2) - \dot{R}^2\dot{A}^2\right] \\ &+ \left[(1-\beta^2)\frac{\dot{A}^2}{R} - \alpha(1-\beta^2)^{3/2}(E_r - c\dot{Z}B_\varphi + c\dot{A}B_z)\right] \times \left[\dot{Z}\dot{R}\dot{A}^2 - \dot{Z}\dot{R}(1-\beta^2+\dot{A}^2)\right] \\ &+ \left[-(1-\beta^2)\frac{\dot{R}\dot{A}}{R} - \alpha(1-\beta^2)^{3/2}(E_\varphi + c\dot{Z}B_r - c\dot{R}B_z)\right] \times \left[\dot{Z}\dot{R}^2\dot{A} - (1-\beta^2+\dot{R}^2)\dot{Z}\dot{A}\right]. \end{aligned}$$

Simplified, the above equation yields

$$\begin{aligned} \ddot{Z} = \alpha(1-\beta^2)^{\frac{1}{2}}&\left[-(E_z + c\dot{R}B_\varphi - c\dot{A}B_r)(1-\beta^2+\dot{R}^2+\dot{A}^2) + (E_r - c\dot{Z}B_\varphi + c\dot{A}B_z)\dot{Z}\dot{R}\right. \\ &\left. + (E_\varphi + c\dot{Z}B_r - c\dot{R}B_z)\dot{Z}\dot{A}\right] . \end{aligned}$$

Noting that $(1-\beta^2+\dot{R}^2+\dot{A}^2) = 1 - \dot{Z}^2$, we have finally

$$\ddot{Z} = \alpha(1-\beta^2)^{\frac{1}{2}}\left[-E_z(1-\dot{Z}^2) + \dot{Z}\dot{R}E_r + \dot{Z}\dot{A}E_\varphi - c\dot{R}B_\varphi + c\dot{A}B_r\right] . \quad (26)$$

The radial acceleration $\ddot{R}$, is given by

$$\Delta\ddot{R} = D_1(A_3C_2 - A_2C_3) + D_2(A_1C_3 - A_3C_1) + D_3(A_2C_1 - A_1C_2)$$

which becomes

$$(1-\beta^2)^2\,\ddot{R} = \left[-\alpha(1-\beta^2)^{3/2}(E_z + c\dot{R}B_\varphi - c\dot{A}B_r)\right]\times\left[(\dot{R}\dot{Z}\dot{A}^2 - \dot{R}\ddot{Z}(1-\beta^2+\dot{A}^2)\right]$$

$$+\left[(1-\beta^2)\frac{\dot{A}^2}{R} - \alpha(1-\beta^2)^{3/2}(E_r - c\dot{Z}B_\varphi + c\dot{A}B_z)\right]\times\left[(1-\beta^2+\dot{Z}^2)(1-\beta^2+\dot{A}^2)\right.$$

$$\left.-\dot{Z}^2\dot{A}^2\right]+\left[-(1-\beta^2)\frac{\dot{R}\dot{A}}{R} - \alpha(1-\beta^2)^{3/2}(E_\varphi + c\dot{Z}B_r - c\dot{R}B_z)\right]\times\left[\ddot{Z}\dot{R}\dot{A}\right.$$

$$\left.-\dot{R}\dot{A}(1-\beta^2+\dot{Z}^2)\right].$$

Simplified, the above equation yields

$$\ddot{R} = \alpha(1-\beta^2)^{\frac{1}{2}}\left[E_z\ddot{Z}\dot{R} + c\ddot{Z}\dot{R}^2B_\varphi - c\ddot{Z}\dot{R}\dot{A}B_r - (E_r - c\dot{Z}B_\varphi + c\dot{A}B_z)(1-\beta^2+\dot{Z}^2+\dot{A}^2)\right.$$

$$\left.+E_\varphi\dot{R}\dot{A} + c\dot{Z}B_r\dot{R}\dot{A} - c\dot{R}^2\dot{A}B_z\right] + \frac{\dot{A}^2}{R}(1-\beta^2+\dot{Z}^2+\dot{A}^2) + \frac{\dot{R}^2\dot{A}^2}{R}.$$

Noting that $(1-\beta^2+\dot{Z}^2+\dot{A}^2) = (1-\dot{R}^2)$, we have finally

$$\ddot{R} = \alpha(1-\beta^2)^{\frac{1}{2}}\left[-E_r(1-\dot{R}^2) + E_z\ddot{Z}\dot{R} + E_\varphi\dot{R}\dot{A} + c\ddot{Z}B_\varphi - c\dot{A}B_z\right] + \frac{\dot{A}^2}{R}. \qquad (27)$$

The azimuthal acceleration $\ddot{A}$, is given by

$$\Delta\ddot{A} = D_1(A_2B_3 - A_3B_2) + D_2(A_3B_1 - A_1B_3) + D_3(A_1B_2 - A_2B_1)$$

which becomes

$$(1-\beta^2)^2\,\ddot{A} = \left[-\alpha(1-\beta^2)^{3/2}(E_z + c\dot{R}B_\varphi - c\dot{A}B_r)\right]\left[\dot{A}\dot{R}^2\ddot{Z} - \ddot{Z}\dot{A}\times(1-\beta^2+\dot{R}^2)\right]$$

$$+\left[(1-\beta^2)\frac{\dot{A}^2}{R} - \alpha(1-\beta^2)^{3/2}(E_r - c\dot{Z}B_\varphi + c\dot{A}B_z)\right]\times\left[\dot{A}\dot{R}^2\ddot{R} - \dot{A}\ddot{R}(1-\beta^2+\dot{Z}^2)\right]$$

$$+\left[-(1-\beta^2)\frac{\dot{R}\dot{A}}{R} - \alpha(1-\beta^2)^{3/2}(E_\varphi + c\ddot{Z}B_r - c\dot{R}B_z)\right]\times\left[(1-\beta^2+\dot{Z}^2)(1-\beta^2+\dot{R}^2)\right.$$

$$\left.-\dot{Z}^2\dot{R}^2\right].$$

Simplified, the above equation yields

$$\ddot{A} = \alpha(1-\beta^2)^{\frac{1}{2}}\left[E_z\ddot{Z}\dot{A} + c\dot{R}B_\varphi\ddot{Z}\dot{A} - c\dot{A}^2B_r\ddot{Z} + \dot{A}\dot{R}E_r - c\ddot{Z}\dot{A}\dot{R}B_\varphi + c\dot{A}^2B_z\dot{R}\right.$$

$$\left.-(E_\varphi + c\dot{Z}B_r - c\dot{R}B_z)(1-\beta^2+\dot{Z}^2+\dot{R}^2) - \frac{\dot{A}^2\dot{R}}{R} - \frac{\dot{R}\dot{A}}{R}(1-\beta^2+\dot{Z}^2+\dot{R}^2)\right].$$

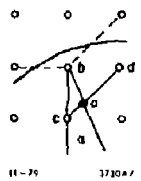
Noting that $(1-\beta^2+\dot{Z}^2+\dot{R}^2) = (1-\dot{A}^2)$ we have finally

$$\ddot{A} = \alpha(1-\beta^2)^{\frac{1}{2}}\left[-E_\varphi(1-\dot{A}^2) + E_z\ddot{Z}\dot{A} + E_r\dot{A}\dot{R} - c\ddot{Z}B_r + c\dot{R}B_z\right] - \frac{\dot{R}\dot{A}}{R}. \qquad (28)$$

APPENDIX II

GENERAL NEUMANN BOUNDARIES

If a boundary with normal derivative equal to zero is as shown, then a problem boundary is drawn as shown by the dashed line. A point at "a" is chosen such that $V_a = V_b$. Point "a" is seen to lie on the normal to the boundary through the point "b" at the intersection between points "c" and "d". The slope of the boundary is given by tan α.



Starting from

$$V_a = V_b \tag{1}$$

we have

$$\frac{V_a - V_c}{\overline{ac}} = \frac{V_d - V_a}{\overline{ad}} \tag{2}$$

where, for example, $\overline{ac}$ is the distance from point "a" to point "c". The mesh interval is taken to be unity. Cross-multiplying, we have

$$\overline{ad}\, V_a - \overline{ad}\, V_c = \overline{ac}\, V_d - \overline{ac}\, V_a$$

or

$$(\overline{ad} + \overline{ac})\, V_a = \overline{ac}\, V_d + \overline{ad}\, V_c \,. \tag{3}$$

But, $\overline{ad} + \overline{ac} = \sqrt{2}$ and $V_a = V_b$, hence

$$\sqrt{2}\ V_b = \overline{ac}\, V_d + \overline{ad}\, V_c \,. \tag{4}$$

From the law of sines,

$$\frac{\overline{ac}}{\sin\alpha} = \frac{1}{\sin(\pi - \frac{\pi}{4} - \alpha)} = \frac{1}{\cos(\frac{\pi}{4} - \alpha)} = \frac{1}{\cos\frac{\pi}{4}\cos\alpha + \sin\frac{\pi}{4}\sin\alpha}$$

which becomes

$$\overline{ac} = \frac{\sqrt{2}\sin\alpha}{\sin\alpha + \cos\alpha} = \frac{\sqrt{2}\tan\alpha}{1 + \tan\alpha} \,. \tag{5}$$

Then the other segment is

$$\overline{ad} = \sqrt{2} - \overline{ac} = \sqrt{2}\left(1 - \frac{\tan\alpha}{1+\tan\alpha}\right) = \frac{\sqrt{2}}{1+\tan\alpha} \,. \tag{6}$$

The complete difference equation from Eq. (4) is

$$\sqrt{2}\ V_b = \frac{\sqrt{2}\tan\alpha}{1+\tan\alpha} V_d + \frac{\sqrt{2}}{1+\tan\alpha} V_c$$

which in the notation used in the main text is

$$\frac{\tan\alpha}{1+\tan\alpha} V_1 + \frac{1}{1+\tan\alpha} V_4 - V_0 = 0 \,. \tag{7}$$

APPENDIX III

INSTRUCTION COMMENTS FROM THE PROGRAM

```
C   SLAC ELECTRON OPTICS PROGRAM: VECTOR POT./PLOTFILE VERSION OF NOV  1979
C
C     W. B. HERRMANNSFELDT
C     STANFORD LINEAR ACCELERATOR CENTER
C     STANFORD UNIVERSITY
C     STANFORD, CALIFORNIA     94305
C
C        SUBROUTINE                    LINE NO.
C
C SUBROUTINE ANALYZ(MI)                                    650.
C SUBROUTINE TIMTST(IT,NL)                                 696.
C SUBROUTINE CHILDA (*)                                    734.
C SUBROUTINE CHILDB                                       1019.
C SUBROUTINE BSET(K,BOOL, * )                             1179.
C SUBROUTINE CHILDB(BR2)                                  1220.
C SUBROUTINE PRFILE                                       1251.
C SUBROUTINE PDTLST                                       1314.
C SUBROUTINE POISSN (N,*)                                 1348.
C SUBROUTINE BOUND (PUTN,MAD,*)                           1447.
C SUBROUTINE COEF(*)                                      1661
C SUBROUTINE TRAJCT (*)                                   1847.
C SUBROUTINE PLOTS                                        2251.
C SUBROUTINE EQUIP (FZ,NO)                                2287.
C SUBROUTINE LAPLAC (*)                                   2340.
C SUBROUTINE FRAME                                        2404.
C SUBROUTINE OSPROC(IEQO,EOB,*)                           2463.
C SUBROUTINE LISTL (SS)                                   2493.
C SUBROUTINE COORD(N,RHO,ZETA)                            2527.
C SUBROUTINE MAGFD                                        2556.
C SUBROUTINE LISTMG                                       2614.
C SUBROUTINE PRTIAL(RHO,ZETA,PU,*)                        2655.
C SUBROUTINE TOUCH(I,L, * )                               2891.
C SUBROUTINE AZP(Z,Y,E,B,C)                               2982.
C FUNCTION ROMAX(B)                                       3004.
C SUBROUTINE RAYNST(IRAT)                                 3023.
C SUBROUTINE PERVNC(MI)                                   3169.
C SUBROUTINE THERM                                        3451.
C SUBROUTINE LOOPS (RHO,ZETA,HR,HZ)                       3529.
C SUBROUTINE SCALE2 (XX,AXLEN,NFTS,XD,XL)                 3560.
C SUBROUTINE HEADA                                        3601.2
C SUBROUTINE CALBRZ(RHO,ZETA,BR,BZ,*)                     3604.3
C SUBROUTINE LEFTI
C
C     ********************************** INSTRUCTIONS **********************
C
C   SAMPLE PROBLEM:
C INJECTION GUN MODEL 4-1A GRID-CATHODE REGION (WBH) MOD.11-20-67  MI=0
C&INPUT1
C RLIM=72.,ZLIM=40.,PUTN=4.,PUT=0.0,5000.0,0.0,0.0,0.0,MI=0,MAGSEG=1,TYME=2.
C&END
C&INPUT2
C Z1=20.,Z2=40.,Z3=20.,BC=0.0,25.0.
C&END
C 1     0     1          0.0       -0.99
C 1    16     1          2.0       -0.4
C 1    37     3          0.99       -0.1
C 4    38     4          2.0        -1.0
C 4    48    10          2.0        -0.8
C 4    55    14          0.99       -0.6
```

```
C  4    56    15          2.0         -1.0
C  4    57    15          2.0         -0.4
C  4    58    15          2.0         -0.3
C  4    59    15          2.0         -0.4
C  4    60    15          2.0         -1.0
C  4    61    14         -0.99         2.0
C  4    61    13         -0.2         -0.8
C  4    62    12         -0.7          2.0
C  4    62     6         -0.7          2.0
C  4    62     0         -0.7          0.0
C  0    66     0          2.0          0.0
C  2    71     0          0.99         0.0
C  2    71    10          0.99         2.0
C  2    71    26          0.99         2.0
C  2    71    27          0.99         0.99
C  2    70    27         -0.2          0.99
C  2    69    26          2.0          0.8
C  2    49    17         -0.3          0.2
C  2    41    13          2.0          0.8
C  2    40    13          2.0          0.4
C  2    39    13          2.0          0.3
C  2    22    11          2.0          0.2
C  2     0    10          0.0          0.3
C  0     0     8          0.0          2.0
C  0     0     2          0.0          2.0
C 888
C&INPUTS
C IZI=1. I22=2, I2S=10,  RAD=257, RMAX=37.5, UNITIN=0.01, SPC=0.0,
C&END
C
C       CARD NO. 1 CONTAINS TITLE ON ONE CARD
C&INPUT1 CARD NO. 2:  &INPUT1, (STARTS IN COL. 2)
C       CARD NO. 3 CONTAINS RLIM, ZLIM, POTN, POT(1), POT(2),....
C       POT(POTN),MI,MAGSEG, ALL IN NAMELIST FORMAT.
C     INSTRUCTION          DEFAULT,MAX          COMMENT
C      RLIM=XX             RLIM=50, 100      HEIGHT OF PROBLEM
C      ZLIM=XX             ZLIM=50, 300      WIDTH OF PROBLEM
C        (SIZE LIMIT (RLIM+1)(ZLIM+2) < 9001)
C       IAX=XX                IAX=0          DEPRESSED AXIS
C      POTN=XX              POTN=101, 101     NUMBER OF POTENTIALS
C      POT(I)=X.X TO POT(POTN) DEFAULT TO ZERO.POTENTIALS IN VOLTS
C        (USE NEGATIVE POTN TO SIGNAL RECTANGULAR COORDINATES)
C      MI=X                  MI=1             PLOT INSTRUCTION, SEE TABLE
C        (IF MI IS NEGATIVE, PROGRAM WILL ONLY PROCESS BOUNDARY DATA)
C      MAGSEG=X            MAGSEG=0           NUMBER OF SEGMENTS OF MAGNETIC
C                                             FIELD DATA TO BE READ NEXT.
C      INTPA=.TRUE.     INTPA=.FALSE.         CALLS INPUTA TO READ VECTOR POTENTIALS.
C       LSTPOT=X           LSTPOT=0      DON'T PRINT POT MAP
C       LSTPOT=1  PRINT FIRST,  =2, PRINT FINAL, =3 PRINT FIRST AND LAST DEC-78
C       TYME = X.X          TYME = 2.0       MAX PROBLEM RUN TIME  MIN.
C          EXPECTED POTENTIALS
C          POT(1) = CATHODE
C          POT(2) = ANODE
C          POT(3) = GRID (CONTROLS EXTRA EQUIPOTENTIALS)
C          POT(4) = FOR A SURFACE WHICH WILL STOP RAYS-NOT A GRID.
C          POT(5) = FOR A SHADOW GRID-NOT FOR FOCUS ELECTRODE
C          OTHER POT( ) VALUES AS DESIRED
C       TABLE FOR VALUES OF MI:(USE MI = 0 FOR NO PLOTS)
C       CYCLE TO BE PLOTTED     INIT & FINAL    ALL     FINAL ONLY
C       WITH EQUIPOTENTIAL LINES      1           2          3
```

```
C       SEPARATE EQUIPOTENTIAL PLOT 4          5       6
C       NO EQUIPOTENTIAL PLOTS      7          8       9
C&END   (INSERT HERE-STARTS IN COL. 2 )
C
C     ------------------------------------------------------------
C        MAGNETIC FIELD METHODS
C         1)   INPUT2 ... POLYNOMIAL SEGMENTS ... MAGSEG=N  IN &INPUT1
C         2)   INPUT3 ... AXIAL FIELD          ... MAGSEG=-1 IN &INPUT1
C         3)   INPUTA ... VECTOR POTENTIAL ARRAY... INIPA=.TRUE. IN &INPUT1
C         4)   INPUT5 ... COIL DATA...FINDS AXIAL FIELDS
C         5)   INPUT6 ... COIL DATA...ELLIPTIC INTEGRALS
C        USE (1) OR (2) FOR RECTANGULAR SYMMETRY
C ------------------------------------------------------------------------
C
C        MAGNETIC FIELD DATA (READ IN MAGSEG SEGMENTS) IN NAMELIST FORMAT
C        WARNING: THIS APPROACH IS VIRTUALLY IMPOSSIBLE TO USE IN A PHYSICALLY
C                 REALISTIC WAY AND IS NOT RECOMMENDED.
C&INPUT2    ( FOR EACH SEGMENT )
C     USE NAMELIST FORMAT FOR THREE INTEGERS, AND AN ARRAY BC
C        OF SEVEN COEFFICIENTS OF VALUE BZ, B1, B2, .... B6
C        B =BZ+B1*DZ+B2*DZ**2+...+B6**6 WHERE DZ=Z-Z3
C       Z TAKES THE VALUES  'Z1' TO 'Z2' WITH ORIGIN AT 'Z3'
C        FOR SIX ORDER EXPANSION, FIELD MUST START 6 UNITS BEHIND
C        CATHODE, OR STARTING POINT, AND GO SIX UNITS PAST ZLIM.
C        IN RECTANGULAR COORDINATES MAGNETIC FIELD IS IN THE
C        TRANSVERSE (PHI) DIRECTION UNLESS MAGORD < 1. (SEE MAGORD, BELOW)
C        IF MAGNETIC FIELD INPUT IS USED IN A RECTANGULAR COORDINATE PROBLEM,
C        THERE IS NO TERM FOR SELF MAGNETIC FIELD, EVEN IF INPUT FIELD IS ZERO.
C        WITHOUT INPUT FIELD SELF-FIELD IS IN PHI DIRECTION. SELF-FIELD IS
C        CALCULATED FROM CURRENT IN RAYS BETWEEN Z-AXIS AND KTH RAY INCLUDING
C        HALF OF IU(K).  INPUT FOR IDEAL COILS IS IN SECTION 5.
C  USE &END   AFTER EACH SEGMENT
C
C&INPUT3
C   POINT BY POINT INPUT OF MAGNETIC FIELDS:
C   IF MAGSEG < 0.  E.G., MAGSEG=-1. THEN USE &INPUT3 TO READ ARRAY
C    BZA=(AXIAL FIELD STARTING AT Z=-6 TO Z=ZLIN+6)
C&END
C
C     ------------------------------------------------------------
C&INPUTA          (TO INPUT VECTOR POTENTIAL DATA)
C          RRO=X.X      RRO=0.0       POSITION OF FIRST ELEMENT OF A().IN MU
C          ZZO=X.X      ZZO=0.0       RELATIVE TO ORIGIN OF GUN PROG.
C          DELR=X.X     DELR=1.0   INCREMENT IN R (CM) FROM POSSON/EDIT
C          DELZ=Z.Z     DELZ=1.0   INCREMENT IN Z (CM) FROM POISSON/EDIT
C          RLMAG=XX     RLMAG=30      NUMBER OF ROWS OF A() DATA
C          ZLMAG=XX     ZLMAG=200     NUMBER OF COLUMNS OF A() DATA
C          A()             VECTOR POTENTIAL DATA ARRAY OF A. EXCEPT A*R AT R=0.
C                          UNITS OF A IN GAUSS-CM. A() IS A LINEAR ARRAY WITH
C                       COLUMNS RLMAG LONG.   MAX SIZE OF A() IS 8000.
C
C     ------------------------------------------------------------
C BOUNDARY INPUT
C     ------------------------------------------------------------
C        BOUNDARY INPUT (3 INTEGERS, 2 FLOATING POINT NUMBERS)
C          PUT. NO., R, Z, DELTA R, DELTA Z
C            FORMAT 3I5,5X,2F10.5
C        TO TERMINATE INPUT, USE POT. NO. >POTN. E.G. 200 IN COL. 3.
C        IF 999 IS USED, SPECIAL BOUNDARIES WILL BE READ. SEE BELOW.
C
```

```
C
C     STARTING CONDITIONS  VALID INSTRUCTIONS AND DEFAULT CONDITIONS
C
C&INPUTS  (INSERT HERE)          C GOES IN COLUME 2.
C&END    (INSERT AFTER START (NSTRUCTIONS)
C          INSTRUCTION           DEFAULT,MAX      COMMENT
C
C    UNIVERSAL PARAMETERS
C
C        PERVO = X.XX            PERVO = 0        ZERO USES LAPLACE/2
C        HOLD = X                HOLD = 1         PERVO 'HOLDS' FOR HOLD
C                                                 ITERATIONS
C        PE = X.X                PE = 2.0         INITIAL ENERGY AT CATHODE
C                                                 IN EV
C        ERROR = X.X             ERROR = 1.0      MULTIPLIES ERROR TEST
C        UNIT = X.XXX            UNIT = 0.001     METERS / MESH UNIT
C        UNITIN = X.XXX          (SEE UNIT)       INCHES/MESH UNIT
C        MAXRAY  = XX            MAXRAY=27,51     MAXIMUM NUMBER OF RAYS
C          IF MAXRAY IS NEGATIVE, THE NUMBER OF RAYS=ABS(MAXRAY)
C        STEP = 0.XX             STEP = 0.8       MESH UNITS / STEP
C        NS = X                  NS = 7           NUMBER OF ITERATIONS
C        SPC = 0.XX              SPC = 0.5        ESTIMATED SPACE CHARGE
C             SPC SIMULATES PARAXIAL APPROXIMATION ON FIRST CYCLE.
C             SPC IS THE FRACTION OF THE RADIAL FORCE USED.
C             SPC=1.0 FOR FULL EFFECT.  SPC=0 FOR NO EFFECT
C        PHILIM=X.X              PHILIM=0.0       AZIMUTHAL LIMIT
C          PHILIM .NE. 0 ENDS TRAJECTORY AT PHI .GT. PHILIM
C        SAVE = 1                SAVE=0 SAVE=1 SAVES BOUNDARIES.
C         TO USE SAVE=1, UNIT BOUNDARY CARDS FROM NEXT PROBLEM.
C        SAVE=2                  SAVE=0           SAVE=2 USES FINAL DATA
C             FROM PREVIOUS RUN TO START THIS RUN.
C             USE ONLY WHEN START='CANDS'.
C        MASS = X.X              MASS = 0         MASS > 0 FOR IONS
C          MASS IS THE MASS TO CHARGE RATIO.  1.0 FOR PROTONS
C          USE MASS<0 FOR RAYS WITHOUT INERTIA: CAN BE USED
C           FOR MAGNETIC FLUX LINES OR ELECTRIC FIELD LINES.
C        AV = X                  AV = 0           SPACE CHARGE AVERAGED
C                                                 LAST AV ITERATION
C        AVR = X.X               AVR = 1.0        WEIGHT OF SPACE CHARGE
C                                 IN PRECEDING PROGRAM CYCLE FOR AV.
C        BEND = X.X              BEND=0.0   MAGNETIC BENDING FIELD
C             IN GAUSS IN THE DIRECTION NORMAL TO THE R-Z PLANE
C             FOR AXIALLY SYMMETRIC PROBLEMS.  FIELD MUST BE
C             UNIFORM.  THE EFFECTS OF SELF-MAGNETIC FIELD ARE LOST
C             AND SPACE CHARGE IS STILL AXIALLY SYMMETRIC SO THAT
C             IF BEAM IS DEFLECTED, CHARGE DISTRIBUTION IS PROBABLY
C             INCORRECT. AN AXIAL FIELD MUST BE INCLUDED IN THE
C             INPUT, EVEN IF IT IS ZERO , E.G., BC=0 IN INPUT2.
C        MAGMLT=X.X              MAGMLT=1.0       MULTIPLIES BZA ARRAY
C        IPSP=K1,K2,...K6        IPSP(I)=0        UP TO SIX RAY NUMBERS FOR POINT
C           BY-POINT PRINTOUT:K,RHO,ZETA,RDOT,ZDOT,TDOT,PHI,BR,BZ,STEP,BPHI
C
C        ZEND=X.X                ZEND=1000.0  EXACT END OF TRAJECTORY
C        CAUTION: IF ZEND IS NOT THE RIGHT-HAND BOUNDARY, THE SPACE
C                 CHARGE DISTRIBUTION MAY BE INCORRECT.
C        VION=X.X                VION=-1E8        LOWEST POTENTIAL PERMITTED
C        USE VION TO SIMULATE SPACE CHARGE NEUTRALIZATION
C
C
C    INPUT FOR EQUIPOTENTIAL PLOTS
```

```
C   ------------------------------------------------------------------
C      EQUIPR = X.X              EQUIPR = 0.0    R-INTERSECTION FOR EQUI-
C                                                POTENTIAL LINES
C      LM = XXX                  LM = 300        LENGTH OF EQUIPOTENTIALS
C      EQLN = 0 TO 20            EQLN = 1        NO. OF CORRECTIONS
C                                                ------------------------
C      EQST = X                  EQST = 2        STEPS PER MESH UNIT
C                                                ------------------------
C          APPLIES ALSO TO GENERAL CATHODE
C      IZ1=X,IZ2=X,IZ3=X         IZ1=0,IZ2=-1    EXTRA EQUIPOTENTIALS AT
C                                IZ3=10          THE INDICATED VALUES OF Z.
C   ------------------------------------------------------------------
C   PLOTTING CONTROLS
C   ------------------------------------------------------------------
C      SCALE = 'YES'             SCALE = '   '   'YES'= DIFFERENT X,Y SCALE
C      SX = XX                   SX = 22         MAXIMUM HORIZONTAL PLOT
C                                                HEIGHT
C      SY = XX                   SY = 9          MAXIMUM VERTICAL PLOT
C                                                HEIGHT
C   ------------------------------------------------------------------
C   MAGNETIC FIELDS; METHOD ONE; READ IN AXIAL FIELD IN SECTION 3(ABOVE)
C   ------------------------------------------------------------------
C      RMAG = X.X                RMAG = RLIM/2   OFF-AXIS MAGNETIC FIELD
C                                                LISTING AT R=RMAG
C      ZMAG = X.X                ZMAG = ZLIM+6   B CONSTANT BEYOND ZMAG
C      MAGORD = 2.4              MAGORD = 6      HIGHEST ORDER FIELD TERM
C                                                IF MAGORD < 1, FOR RECTANGULAR
C                                                COORDINATES. BZA IS IN THE
C                                                R-DIRECTION AND THE OFF AXIS
C                                                EXPANSION IS A FUNCTION OF R.
C      NMAG = X                  NMAG = 0.11     NO. OF FIELD COILS FOR
C                                                METHOD TWO (BELOW)
C   ------------------------------------------------------------------
C   METHOD TWO; READ IN POSITION AND STRENGTH OF NMAG IDEAL COILS.
C   IF NELL=0, PROGRAM CALCULATES AXIAL FIELDS AND PROCEEDS AS IN METHOD ONE.
C   IF NELL=1, THIS METHOD CALCULATES FIELDS USING THE COMPLETE
C   ELLIPTIC INTEGRAL FUNCTIONS.  FIELDS ARE THEN VALID IN ALL SPACE.
C **** ELLIPTIC INTEGRAL METHOD IS VERY SLOW ****
C IF ELLIPTIC INTEGRAL FUNCTIONS ARE INCLUDED WHEN PROGRAM IS
C COMPILED (USER MUST COMMENT OUT THE DUMMY FUNCTIONS AT THE
C END OF THE PROGRAM) THEN, FOR COIL METHOD ONLY, THE PROGRAM
C WILL LIST THE OFF-AXIS FIELDS BY BOTH OFF-AXIS EXPANSIONS
C AND BY USING ELLIPTIC INTEGRALS, EVEN IF NELL=0.  THIS
C PROVIDES AN INTERESTING CHECK ON THE VALIDITY OF THE OFF-
C AXIS EXPANSIONS IN THE USER'S SPECIAL SITUATION.
C      NELL=1                    NELL=0          1 FOR ELLIPTIC INTEGRALS
C
C      CR(I) = X.X               CR(I) = RLIM    RADIUS OF COIL (MESH UNIT)
C      CZ(I) = X.X               CZ(I) = 0.0     AXIAL POSITION OF COIL
C      CM(I) = X.X               CM(I) = 0.0     CURRENT IN AMPERE-TURNS
C   ------------------------------------------------------------------
C START  GENERAL
C   ------------------------------------------------------------------
C      START = 'GENERAL'         START = 'GENERAL'  GENERAL CATHODE
C      RC = X.XX                 RC = 0.0        LOWER END OF STARTING SUR-
C                                                FACE
C      ZC = X.XX                 ZC = Z+CATHODEZ CATHODEZ IS Z VALUE OF
C                                                BOUNDARY FROM FIRST
C                                                DATA CARD.
C      CL = X.XX                 CL = RLIM       MAXIMUM LENGTH OF STARTING
```

```
C                                                    SURFACE
C     DENS = XX.X              DENS = 10.0           MAXIMUM EMISSION (A/CM**2)
C     BETA2 = 1.0              BETA2= 0.0            IF > 0.0 USES LANGMUIR-
C                                                    BLODGETT
C     RAD = X.X                ---                   USE RAD FOR WIRE RADIUS IN
C                                                    RECTANGULAR COORDINATES.
C                                                    BETA2 > 0.0
C     SURFAC  = X              SURFAC  = 1           STARTING SURFACE ITERATION
C                              ------------------------------------------------
C                              USE PUT(5) FOR NON-EMITTING SURFACE. E.G.
C                              HOLLOW CATHODE OR SHADOW GRID. DO NOT USE
C                              PUT(3) OR PUT(5) FOR FOCUS ELECTRODE ...
C                              USE PUT(4) TO STOP ELECTRONS ON IMPACT.
C  ------------------------------------------------------------------------
C  START GENCARD
C  ------------------------------------------------------------------------
C     START = 'GENCARD'        START = 'GENERAL'  GENERAL WITH CARD START
C
C     HAVE UP TO MAXRAY CARDS WHICH SPECIFY:
C       1) RAY NO.
C       2) INITIAL RADIUS R
C       3) INITIAL AXIAL VALUE Z
C       4) DISTANCE FROM CATHODE DX (CATHODE MUST BE PUT(1)).
C       5) EFFECTIVE SPACING BETWEEN RAYS DR.
C       6) PARAMETER WHICH MODIFIES CHILD LANGMUIR EQUATION. ALPH2.
C     NORMAL DX IS 1.0 TO 2.0 MESH UNITS.
C     NORMAL DR IS 1.0 BUT MAY BE VARIED ALONG THE SURFACE.
C     NORMAL ALPH2 IS 1.0 FOR A PLAIN DIODE.
C       FOR CYLINDRICAL COORDINATES:
C     ALPH2=(ALPHA*(RADIUS OF CURVATURE)/(STARTING STEP))**2
C       FOR RECTANGULAR COORDINATES:
C     ALPH2=(BETA**2)*(RADIUS OF CURVATURE)/(STARTING STEP)
C     WHERE ALPHA AND BETA ARE AS DEFINED IN THE LITERATURE.E.G.,
C     SPANGENBERG FOR BETA AND BREWER IN SEPTIER, VOL II, FOR ALPHA
C     FORMAT IS THE SAME AS FOR CARD STARTING; RAY NO.,R,Z,DX,DR,ALPH2
C       ((I5,5X,5F(10.5)).
C  ------------------------------------------------------------------------
C  START  SPHERE
C  ------------------------------------------------------------------------
C     START = 'SPHERE'         START = 'GENERAL'     SPHERICAL CATHODE
C     RAD = X.XX               RAD = 2*ZLIM     SPHERICAL RADIUS
C     RMAX = X.XX              RMAX = RLIM      CATHODE RADIUS
C     ORAD = X.XX              ORAD = CATHODEZ CENTER OF CATHODE
C     ST = X.XX                ST = 2.0         STARTING STEP
C                              ------------------------------------------------
C                              'SPHERE' ALSO WORKS FOR CYLINDRICAL
C                              CATHODE IN RECTANGULAR COORDINATES
C  ------------------------------------------------------------------------
C  START CARDS
C  ------------------------------------------------------------------------
C     START = 'CARDS'          START = 'GENERAL' CARD STARTING
C     ZO = X.XX                ZO = 0.0         OLD ORIGIN IN NEW FRAME
C     SKAL = X.XX              SKAL = 1.0       OLD MESH/NEW MESH
C
C     HAVE UP TO MAXRAY DATA CARDS (1 INTEGER, 6 FLOATING POINT)
C     RAY NO., R, Z, ENERGY(EV), ANGLE(RADIANS), CURRENT(MICROAMPERES
C     IN ONE RADIAN SEGMENT), TRANSVERSE ANGLE.  TRANSVERSE POSITION(PHI)
C     (NOTE CHANGE: TRANSVERSE ANGLE, NOT TRANSVERSE ENERGY; ENERGY IS NOW
C     TOTAL KINETIC ENERGY.)
C       FORMAT I5,5X,7F10.5
```

```
C      STOP READING WITH RAY NO. GREATER THAN MAXRAYS
C      INITIAL TRANSVERSE VELOCITY HAS THE SIGN OF THE TRANSVERSE ANGLE
C
C      IF RECTANGULAR COORDINATES:
C         1) PHI IS TRANSVERSE POSITION IN MESH UNITS.
C         2) CURRENT IS MICROAMPERES IN ONE MESH UNIT DEEP SEGMENT.
C      ***SPECIAL TESTS IN RAYTST: CROSSING OR 3-D SPACE CHARGE**
C      IRAT=1                IRAT=0          3-D SPACE CHARGE
C      IRAT=2                IRAT=0          CROSSING DETECTION
C         USE OF NEGATIVE RAY NUMBERS:
C      A) IF IRAT=1       (3-D SPACE CHARGE)
C         1) MAKE RAY NUMBERS NEGATIVE FOR BEAM EDGE CARDS.
C      USE BEAM EDGE CARDS (IU=0) TO SIMULATE SPACE CHARGE SPREADING
C      OF A CYLINDRICAL BEAM OF CURRENT I AND RADIUS R IN RECT. COORD.
C         PAIRS OF BEAM EDGE CARDS PRECEDE SETS OF RAY CARDS DEFINING
C      PART OF BEAM FOR WHICH 3-D SPACE CHARGE SPREADING IS TO BE SIMULATED
C      SEVERAL PARTS, DIFFERENTIATED BY SELECTED ATTRIBUTES; E.G., ENERGY
C      ALPHA OR RADIUS, CAN BE USED SIMULTANEOUSLY WITH ANY NUMBER OF RAYS
C      IN EACH PART. END OF PART IS DEFINED BY NEXT RAY WITH NEGATIVE RAY
C      NUMBER, WHICH BEGINS THE NEXT PART.
C         2) TO SIMULATE CYLINDRICAL BEAM SPACE CHARGE IN RECTANGULAR
C      COORDINATES MAKE CURRENT PER MESH UNIT, I' = I/(PI*R) INSTEAD
C      OF I' = 2*I/(PI*R) WHICH WOULD HAVE THE SAME CURRENT DENSITY.
C      IN OTHER WORDS, MAKE I'(K) = I(K) / (2*R(K)) INSTEAD OF I(K)/
C      R(K).  NOTE THAT THIS REQUIRES TWICE AS MANY RAYS AS FOR
C      CYLINDRICAL BEAM WITH SYMMETRY. BEAM EDGE CARDS (RAY NO. < 0)
C      ALSO APPLY TO OFF-AXIS PENCIL IN CYLINDRICAL COORDINATES.
C
C     B) IF IRAT=2     (R-Z AND PHI CROSSOVERS)
C         1) R-Z: MAKE RAY NUMBERS NEGATIVE FOR SEQUENTIAL RAYS FOR
C      WHICH FINAL CROSSOVER SHOULD BE DETECTED. CROSSINGS WILL BE
C      LISTED AND PLOTTED.  NEGATIVE RAY NUMBERS SHOULD BE IN PAIRS.
C      TO FIND CROSSOVERS WITH Z AXIS, RUN A RAY WITH R=0,ALPHA=0
C      PRECEDING THE RAY TO TEST AXIS CROSSING.
C         2) PHI: LEAVE RAY NUMBERS POSITIVE FOR TRANSVERSE RAYS TO
C      DETECT LAST CROSSING OF PHI=PI*INTEGER.
C
C       IF SAVE=2, RUN STARTS WITH FINAL RAY DATA FROM PREVIOUS RUN.
C       DO NOT PUT SAVE=2 ON THE FIRST RUN OF A SET.
C-----------------------------------------------------------------------
C     THERMAL EFFECTS
C-----------------------------------------------------------------------
C       SUBROUTINE THERM IS CALLED IF THE PARAMETER TC>0.
C       TC=XXXX.X              TC=0          KELVIN TEMP.  OF CATHODE
C       TWO MODELS ARE INCLUDED IN THIS VERSION
C       KRAY=3                 KRAY=1          THREE RAY SPLIT
C       KRAY=5                 KRAY=1          FIVE RAY SPLIT
C       THREE RAY SPLIT PUTS CURRENTS IN  1-2-1 RATIO WITH 2 PARTS IN
C      UNDEFLECTED RAY AND 1 PART EACH IN RAYS WITH V(PERP)=SQRT(2KT/M)
C      IN R-Z PLANE, UP AND DOWN RELATIVE TO UNDEFLECTED RAY.
C
C       FIVE RAY SPLIT PUTS CURRENTS IN  1-9-0-9-1 RATIO WITH
C      V(PERP)=2*SQRT(2KT/M) FOR 1 PART RAYS AND V(PERP)=1*SQRT(2KT/M)
C      FOR 9 PART RAYS.  NO CURRENT IN CENTER RAY.
C       USERS SHOULD FEEL FREE TO MODIFY SUBROUTINE THERM.
C       THERM CAN BE CALLED FOR START='SPHERE', 'GENERAL', 'CARDS',
C      OR 'GENCARD'.
C       IT CANNOT BE USED FOR START='CARDS' WITH SAVE=2.
C-----------------------------------------------------------------------
C     START LAPLACE
```

```
C     ------------------------------------------------------------------
C        START = 'LAPLACE'      START = 'GENERAL'     NO RAY TRACING
C        NS = X                 NS = 7          NUMBER OF LAPLACE CYCLES
C           ADD DATA CARDS WITH (R,Z, SPACE CHARGE) FOR NON-ZERO POINTS.
C           FORMAT (2I5,E20.7)
C                   END CARD INPUT WITH R > RLIM.
C
C     ------------------------------------------------------------------
C
C     SPECIAL BOUNDARY POINTS (INCLUDING GENERAL NEUMANN BOUNDARIES)
C     ------------------------------------------------------------------
C        USE 999 IN COLS. 3-5 TO END BOUNDARY INPUT. BOUNDARY
C        MUST INCLUDE ALL POINTS TO BE USED AND ALL POT NUMBERS. THEN
C        INCLUDE ANY NUMBER OF CARDS WITH R,Z AND FOUR DIFFERENCE
C        NUMBERS FOR LEFT, RIGHT, UP, AND DOWN, SEQUENTIALLY. NUMBERS
C        SHOULD ADD TO 4*R OR 4 IF RECTANGULAR COORDINATES. END WITH
C        R > RLIM. FOR GENERAL NEUMANN, SEE APPENDIX II OF SLAC 166.
C        TERMS ARE 4*(TAN A)/(1+ TAN A) AND 4/(1 + TAN A) WHERE TAN A <1
C     ------------------------------------------------------------------
C     HORIZONTAL DIELECTRIC BOUNDARY
C     ------------------------------------------------------------------
C        LEFT=RIGHT=(E1*(R-.5)+E2*(R+.5))/2
C        UP = E2*(R+.5)       DOWN = E1*(R-.5)
C        WHERE E1 OR E2 = 1.0 FOR VACUUM AND E2 IS UPPER 'MATERIAL'.
C     ------------------------------------------------------------------
C     VERTICAL DIELECTRIC BOUNDARY
C     ------------------------------------------------------------------
C        LEFT = E1*R          RIGHT = E2*R
C        UP = (E1+E2)*(R+.5)/2   DOWN = (E1+E2)*(R-.5)/2
C        WHERE E2 IS RIGHT HAND 'MATERIAL'.
C     ------------------------------------------------------------------
C     SUMMARY OF FILE 1 FORMAT FOR PLOT DATA OUTPUT
C     ------------------------------------------------------------------
C  WRITE(1)I,L,A,B,C,D,(X(J),J=1,L),(Y(J),J=1,L)
C
C  WHERE:
C  I=0 THROUGH 8
C  FOR I=0,7,8  PLOT A LINE
C        L=NUMBER OF DATA POINTS TO BE PLOTTED
C        X, Y ARE ARRAYS OF LENGTH >= L, WITH X,Y DATA
C   FOR I=1, PLOT X AXIS. FOR I=2, PLOT Y AXIS
C        L=NUMBER OF COMPUTER WORDS IN TITLE
C           FOR IBM/360  L=(N+3)/4 IF N=NUMBER OF CHARS
C        A=SCALE (DATA UNITS/INCH)
C        B=AXIS LENGTH (INCHES)
C        C=X COORD OF Y AXIS, OR Y COORD OF X (OTHER COORD IS 0.)
C        D=DATA VALUE TO APPEAR ON LOWER END OF AXIS
C  FOR I=3,  END OF PICTURE, GET A CLEAN AREA ON PAPER, ETC.
C        L=1;    A,B,C,D,X,Y=0.0
C  FOR I=4, CLOSE PLOT, THIS IS THE LAST RECORD OF THE FILE
C        L=1;    A,B,C,D,X,Y=0.
C  FOR I=5, PLOT POINTS (OR X'S, OR SOME SYMBOL)
C        L,A,B,C,D,X,Y   SAME AS FOR I=0 (LINES)
C  FOR I=6, SET SCALE FACTOR
C           A=X AXIS LENGTH
C           B=Y AXIS LENGTH
C           C=$X (FROM &INPUT5)
C           D=$Y
C        PLOT AREA MUST BE AT LEAST -0.5<X<A+0.5  -0.5<Y<B+0.5
C        C AND D CAN BE USED IF NEEDED.
```

```
C       THE TITLE ON THE AXIS SHOULD BE UNDER THE X AXIS,
C       AND TO THE LEFT OF THE Y AXIS  (THE PROGRAM CAN PLOT
C       MORE THAN ONE Y AXIS ON A PLOT, SO BE CAREFUL.)
C    I LESS THAN 0, OR GREATER THAN 8 SHOULDN'T HAPPEN, BUT CHECK IT.
C
C
C
C     ---------------------------------------------------------------
C    ARRAY SIZES
C     ---------------------------------------------------------------
C
C   MAX SIZE OF POTENTIAL ARRAY,101,  ADJUST POTN=101,POT(101),LLL=1,101
C   MAX BOUNDARY SIZE; 901, ADJUST BOND1,BOND2,BOND3 OBOND1,OBOND2,
C      ABCX(901+4),ABCY(901+4),ORDER(901+101),XT(901+101,6)
C   MAX RLIM 100, ADJUST ORDER(901+101),XT(901+101,6),A(101,5),X(101)
C   MAX NUMBER OF RAYS; 51, ADJUST AL(51+2),IO(51),II(51),RR(51+2),RNIN(
C      TPHI(51),VV(51),XD(9,51),ZZ(51),I1(51),LL(51),IRMIN=51
C   MAX SIZE OF PROBLEM; 9001, (RLIM+1)*(ZLIM+2) <= 9001
C     ADJUST TYPE(9001),U(9001),HH(9001).
C   MAX ZLIM; 300,  ADJUST BX(301+2),BY(301+2),RZX(2*301+2),RZY(2*301+2)
C    RZY INIT, LOOP =1,2*301+2,   BZA(301+14),   IBZA=301+14 , RARR(301)
C      LM=301          LENGTH OF EQUIPOTENTIAL
C   MAX NUMBER OF COLUMNS; 401,ADJUST CINC(3,401),
C      (SHOULD BE LARGER THAN ZLIM)
C   RARR(3,151)  ONLY FOR RECT. SPRD. IN CYL. COORD.
C      ********************************* MAIN ******************************
C
```

APPENDIX IV

SAMPLE PLOTTER INTERFACE PROGRAM

```
      REAL X(1000),Y(1000)
      REAL*8 T(9)
     #/'LINE','X-AXIS','Y-AXIS','NEW PIC.','CLOSE','POINTS','OPEN',
     # 'LINE','LINE'/
      CALL STRTP2(17)
1     READ(1,END=99)I,L,DX,DY,SX,SY,(X(J),J=1,L),(Y(J),J=1,L)
      WRITE(6,101) T(I+1),I,L,DX,DY,SX,SY
101   FORMAT(1X,A8,2I2,4F10.4)
      I=I+1
      GOTO (10,11,12,13,14,15,1,16,10),I
C  CHECK FOR ERRORS
      I=I-1
      WRITE(6,100)  I
100   FORMAT(' OOPS.',I2,' FOUND IN FILE')
      GO TO 1
C  DRAW A LINE
10    X(L+1)=SX
      X(L+2)=DX
      Y(L+1)=SY
      Y(L+2)=DY
      CALL LINE2(X,Y,L,1,0,0)
      GO TO 1
C  DRAW AN X-AXIS
11    CALL AXIS2(SX,0.,X,-L*4,DX,0.,SY,DY)
      SAVEDX=DX
      GO TO 1
C  DRAW Y-AXIS
12    CALL AXIS2(SX,0.0,X,L*4,DX,90.0,SY,DY)
      GO TO 1
C  END OF PLOT
13    CALL PLOT2(SAVEDX+7.,0.,-3)
      GO TO 1
C  CLOSE FILE (TAPE)
14    CALL ENDP2
      GO TO 1
C  PLOT X'S
15    X(L+1)=SX
      X(L+2)=DX
      Y(L+1)=SY
      Y(L+2)=DY
      CALL LINE2(X,Y,L,1,-1,4)
      GO TO 1
99    WRITE(6,102)
102   FORMAT(' END OF FILE FOUND.')
      STOP
      END
```

APPENDIX V

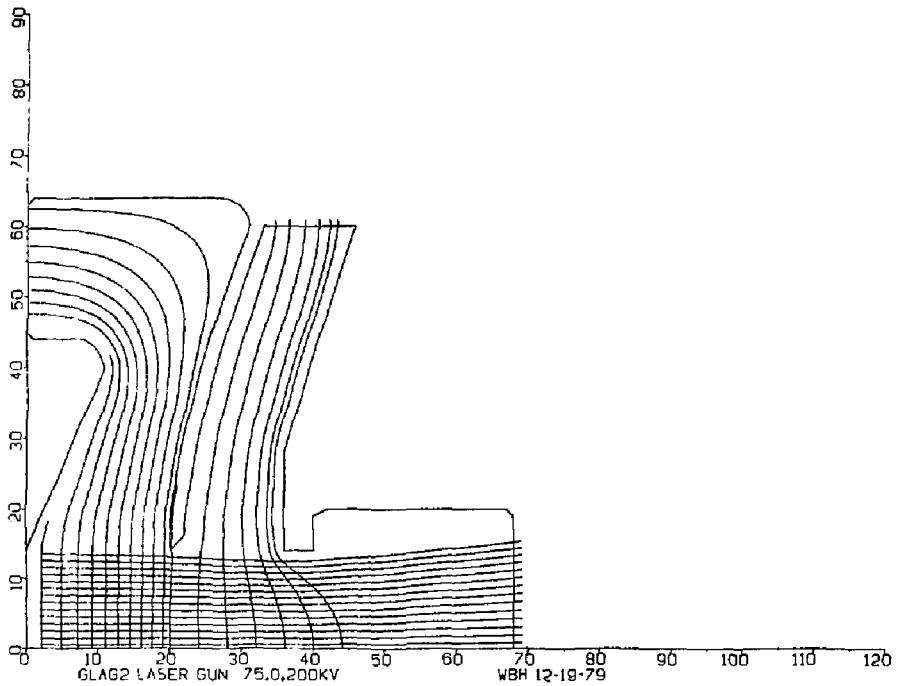


Fig. 8. Sample output for a very high perveance gun.

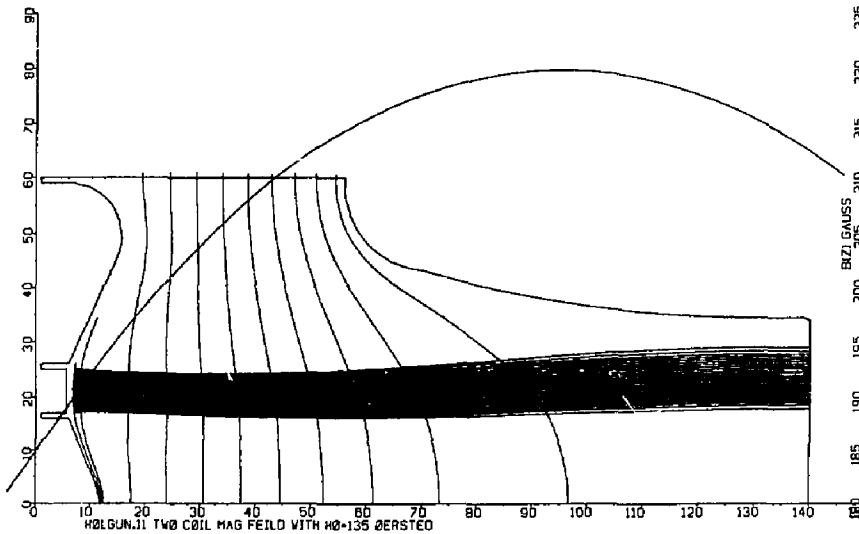


Fig. 9. Sample output for a hollow beam gun.

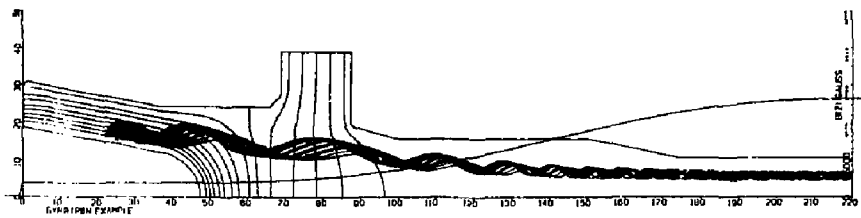


Fig. 10. Sample output for a gyrotron gun.

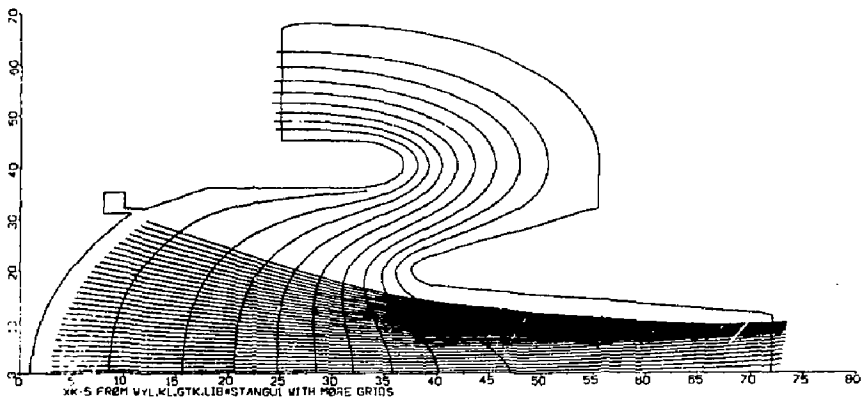


Fig. 11. Sample output for a klystron gun.

REFERENCES

1. Forsythe and Wasow, Finite Difference Methods for Partial Differential Equations, Wiley and Sons.
2. D. N. de G. Allen, M.A., Relaxation Methods, McGraw Hill.
3. F. S. Shaw, Relaxation Methods, Dover Publications.
4. Richard S. Varga, Matrix Iterative Analysis, Prentice Hall.
5. Vladimir Hamza: NASA TN 1323, TN 1665, TN 1711.
6. A. Septier, Focusing of Charged Particles, Academic Press (1967). See Chapter 1.2 by C. Weber in Volume I for the most relevant material to this application.
7. J. R. M. Vaughan, "Representation of Axisymmetric Magnetic Fields in Computer Program," IEEE Transactions on Electron Devices, ED-19, No. 2, Feb. 1972, pp. 144-151.
8. K. R. Spangenburg, Vacuum Tubes, McGraw Hill, New York (1948).
9. K. R. Spangenburg, Fundamentals of Electron Devices, McGraw Hill, New York (1957).
10. I. Langmuir and K. B. Blodgett, Phys. Rev. 22, 347 (1923).
11. G. R. Brewer, High Intensity Electron Guns, Focusing of Charged Particles, Vol. II, A. Septier, Ed., Academic Press, New York (1967).
12. H. Busch, Ann. Physik 4, 80, 974 (1926), treated by G. R. Brewer. Focusing of High-Density Electron Beams, Focusing of Charged Particles, Vol. II., A. Septier, Ed., Academic Press, New York (1967).
13. J. A. Seeger, Proceedings of the IEEE, 56, 1393, August 1968.

14. K. G. Steffan, High Energy Beam Optics, Interscience Monographs, John Wiley and Sons, New York (1965).
15. R. H. Miller, J. Berk and T. O. McKinney, The Electron Gun for the Stanford Two-Mil. Accelerator, IEEE Transactions on Nuclear Science, June 1967, Vol. NS-17, No. 3.