Lawrence Berkeley National Laboratory
Recent Work

Title

JASON: A DIGITAL COMPUTER PROGRAM FOR NUMERICALLY SOLVING THE LINEAR POISSON EQUATION V .(kV o) = p

Permalink

https://escholarship.org/uc/item/2tn9554q

Authors

Sackett, S.
Healey, R.

Publication Date

1969-02-01

eScholarship.org Powered by the California Digital Library
University of California

Submitted to
Nuclear Instruments and Methods

UCRL-18721 (Rev. 1)
Preprint

RECEIVED
LAWRENCE
RADIATION LABORATORY
APR 2 1969
LIBRARY AND
DOCUMENTS SECTION

JASON - A DIGITAL COMPUTER PROGRAM FOR NUMERICALLY SOLVING THE LINEAR POISSON EQUATION

$$\nabla \cdot (\kappa \nabla \phi) = \rho$$

S. Sackett and R. Healey

February 1969

AEC Contract No. W-7405-eng-48

TWO-WEEK LOAN COPY

This is a Library Circulating Copy which may be borrowed for two weeks. For a personal retention copy, call Tech. Info. Division, Ext. 5545

LAWRENCE RADIATION LABORATORY
UNIVERSITY of CALIFORNIA BERKELEY

UCRL-18721 Rev. 1

DISCLAIMER

This document was prepared as an account of work sponsored by the United States Government. While this document is believed to contain correct information, neither the United States Government nor any agency thereof, nor the Regents of the University of California, nor any of their employees, makes any warranty, express or implied, or assumes any legal 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 its 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, or the Regents of the University of California. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof or the Regents of the University of California.

UCRL-18721 (Rev. 1)
Preprint

JASON - A DIGITAL COMPUTER PROGRAM FOR NUMERICALLY SOLVING THE LINEAR POISSON EQUATION

$$\nabla \cdot (\kappa \nabla \phi) = \rho$$

S. Sackett and R. Healey

Lawrence Radiation Laboratory
University of California
Berkeley, California

February 1969

ABSTRACT

This paper describes the development of a high-speed computer program for the Solution of the Linear Poisson Equation,

$$\nabla \cdot (\kappa \nabla \phi) = \rho,$$

where ϕ and ρ are scalar functions of two independent variables, and κ is a second-rank tensor whose components are functions of two independent variables. The mathematical model is given in detail. Examples showing the application of the program to the solution of problems in electrostatics demonstrates its speed and accuracy.

Outstanding characteristics of JASON, in addition to speed, are the following:

(1) program may be used both for cylindrically symmetric systems and for two-dimensional Cartesian systems,

(2) completely general boundary conditions (Neumann, Dirichlet),

(3) generalized quadrilateral mesh,

(4) utilization of algorithms which ensure continuity of ϕ across mesh lines,

(5) use of block iterative methods for solution of the equations,

(6) ease and simplicity of input,

(7) nonhomogeneous, anisotropic media may be considered through use of the tensor κ.

UCRL-18721 (Rev. 1)
Preprint

MATHEMATICAL MODEL

In this section, a mathematical model for numerical solution of the linear Poisson equation

$$\nabla.(\kappa\nabla\phi) = \rho, \tag{1}$$

is constructed; ϕ and ρ are scalar functions of two independent variables and κ is a second-rank tensor whose components are functions of two independent variables. The development is limited to systems possessing cylindrical symmetry. In addition, it is assumed that the tensor, κ, is diagonal in the coordinate system chosen:

$$\phi = \phi(r,z) \quad ; \quad \rho = \rho(r,z) \quad ; \quad \kappa = \begin{bmatrix} \kappa_r(r,z) & 0 \\ 0 & \kappa_z(r,z) \end{bmatrix}. \tag{1a}$$

We will also show how the equations derived here for the cylindrical case can be applied to two-dimensional Cartesian systems. The development given here follows the work of Zienkiewicz [1].

For systems with cylindrical symmetry, equation (1) can be written as

$$\frac{\partial}{\partial r}\left(\kappa_r \frac{\partial\phi}{\partial r}\right) + \frac{\kappa}{r}r \frac{\partial\phi}{\partial r} + \frac{\partial}{\partial z}\left(\kappa_z \frac{\partial\phi}{\partial z}\right) - \rho(r,z) = 0; \tag{2}$$

this is the Euler-Lagrange equation for the variational problem

$$I(\phi) = \int_R \frac{r}{2}\left[\kappa_r \left(\frac{\partial\phi}{\partial r}\right)^2 + \kappa_z \left(\frac{\partial\phi}{\partial z}\right)^2 + 2\rho\phi\right] drdz + \int_C qr\phi \, ds = \text{minimum}, \tag{3}$$

where R is open region of two-dimensional space,

C is one or more differentiable curves bounding R,

ϕ is assumed to be of class C^1 on R and C.

Applying the variational principle, we find that any function, ϕ, which renders (3) stationary in R, satisfied (2) in R subject to the natural

boundary condition

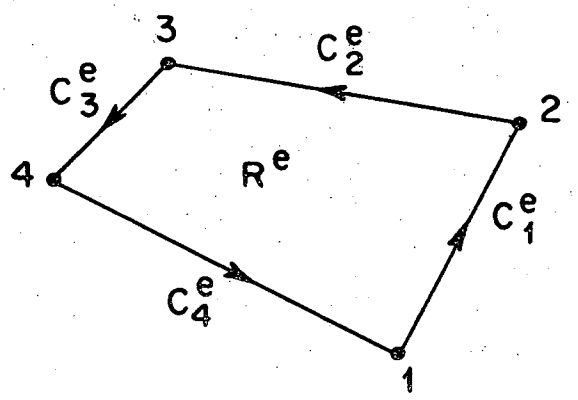
$$\kappa \frac{\partial \phi}{\partial n} + q = 0 \tag{4}$$

on C, where $\partial/\partial n$ denotes differentiation with respect to the outward normal on the contour C [2]. Note that the second integral in (3) is identically zero whenever r is zero (since ϕ is C^1). This ensures that $\partial\phi/\partial n = 0$ whenever r is zero, a necessary condition for cylindrical symmetry.

To construct a set of difference equations approximating (2) in R, we use the "finite element method" [1]. We first divide the region, R, into a finite number of subregions, or elements, R^e, e = 1,...N , such that

$$\begin{aligned} &\bigcup_{e=1}^{N} R^e = R, \\ &R^e \cap R^f = \delta_{ef} R^e \quad ; \quad e,f = 1,\cdots,N, \\ &\delta_{ef} = \begin{cases} 0 & e \neq f \\ 1 & e = f \end{cases}, \end{aligned} \tag{5}$$

and such that the contour C is approximated by an exclusive union of sides of the R^e, e = 1,···,N. For our purposes, we take the elements to be polygons of four nodes (corners):



XBL693-2271

It is clear that a construction of such elements can be easily made to satisfy conditions (5).

Assuming that within each element the variation of ϕ is prescribed by the values of ϕ associated with the nodes of the element, we have

$$\phi^e(r,z) = [N]^e \{\phi\}^e = N_1^e \phi_1 + N_2^e \phi_2 + N_3^e \phi_3 + N_4^e \phi_4, \tag{6}$$

in which the matrix $[N]^e$ involves suitable functions of coordinates.

The condition for ϕ to render $I(\phi)$ stationary is that

$$\frac{\partial I(\phi)}{\partial \phi} = 0.$$

For a given set of values of $\{\phi\} = \bigcup_{e=1}^{N} \{\phi\}^2$, therefore, minimization of (3) may be accomplished approximately by satisfying the set of equations

$$\frac{\partial I(\phi)}{\partial \phi_s} = 0 \quad ; \quad s = 1, \cdots, m, \tag{7}$$

where m is the total number of distinct elements in the set $\{\phi\}$; i.e., the number of distinct nodes in the region $R = \bigcup_{e=1}^{N} R_e$.

Let $I(\phi)^e$ be the contribution of an element, e, to the total integral $I(\phi)$. Then we have

$$I(\phi) = \sum_{e=1}^{N} I(\phi)^e, \tag{8}$$

provided none of the $I(\phi)^e$ is infinite. Since $I(\phi)$, and consequently $I(\phi)^e$, depends only on the first derivative of ϕ, this condition will be satisfied if ϕ is of class C^1 on R and C. This was one of our

assumptions in applying the variational principle. A condition on the matrix $[N]^e$ is, therefore, that it be constructed of functions which are of class C^1 on R^e and C^e, C^e denoting the boundary of the element R^e.

Using equations (3), (6), and (8), we have the following relation:

$$\frac{\partial I(\phi)^e}{\partial \{\phi\}^e} = [S]^e \{\phi\}^e + \{F\}^e, \tag{9}$$

where the elements of the matrix $[S]^e$ are given by

$$S^e_{nm} = \int_{R^e} \kappa_r r \frac{\partial N^e_n}{\partial r} \frac{\partial N^e_m}{\partial r} \, drdz + \int_{R^e} \kappa_z r \frac{\partial N^e_n}{\partial z} \frac{\partial N^e_m}{\partial z} \, drdz \quad ; \; n,m = 1,\cdots,4 \tag{10}$$

and the elements of the vector $\{F\}^e$ are given by

$$F^e_n = \int_{C^e} qrN^e_n \, ds + \int_{R^e} \rho r N^e_n \, drdz \quad ; \; n = 1,\cdots,4. \tag{11}$$

Equations giving the approximate set of potentials can now be obtained from equations (7) and (8) together with (9):

$$\frac{\partial I(\phi)}{\partial \phi_n} = \sum_e \frac{\partial I(\phi)^e}{\partial \phi_n} = 0 \tag{12}$$

$$\Rightarrow [S] \{\phi\} + \{F\} = 0, \tag{13}$$

$$S_{nm} = \sum_e S^e_{nm}, \qquad F_n = \sum_e F^e_n. \tag{14}$$

Note that the sums in (12) and (14) need to be taken only over those elements which share the node n, since

$$\frac{\partial I(\phi)^e}{\partial \phi_n} \equiv 0$$

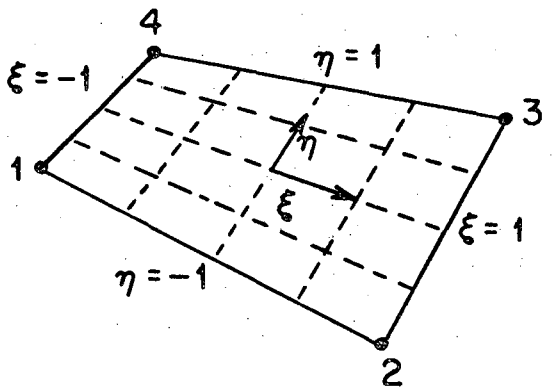
if $I(\phi)^e$ does not contain ϕ_n (and hence node n).

We have yet to construct the matrix $[N]^e$. To do this, we need to find a function of the coordinates, r and z, which suitably describes the variation of ϕ over an element, and ensures convergence of the approximate minimization process given above. As was shown earlier, the function must also be of class C^1 on R^e and C^e to be admissible. This restriction can be shown to be sufficient to ensure convergence, provided the function can take on a constant value in an infinitesimal element.

Considering the above restrictions, an appropriate function is the bilinear form

$$\phi = \alpha_1 + \alpha_2 \xi + \alpha_3 \eta + \alpha_4 \xi\eta, \tag{15}$$

where ξ and η are local skewed coordinates defined as shown:



XBL693-2270

The values of ϕ at the four nodes of an element will then uniquely determine the four coefficients in (15) for that element, and hence the components of $[N]^e$. Solving for the α's in terms of the nodal values of ϕ, we have the following expressions for $[N]^e$:

$$\begin{aligned} N_1^e &= \tfrac{1}{4}(1-\xi)(1-\eta), \\ N_2^e &= \tfrac{1}{4}(1+\xi)(1-\eta), \\ N_3^e &= \tfrac{1}{4}(1+\xi)(1+\eta), \\ N_4^e &= \tfrac{1}{4}(1-\xi)(1+\eta), \end{aligned} \tag{16}$$

where the local coordinated ξ and η are related to the global coordinates r and z by the formulas

$$\begin{aligned} z^e &= N_1^e z_1^e + N_2^e z_2^e + N_3^e z_3^e + N_4^e z_4^e, \\ r^e &= N_1^e r_1^e + N_2^e r_2^e + N_3^e r_3^e + N_4^e r_4^e. \end{aligned} \tag{17}$$

Equations (10) and (11) may now be used to obtain the elements of the matrix $[S]^e$ and the vector $\{F\}^e$, respectively, from (16) and (17):

$$S_{nm}^e = \kappa_r I_1(R^e) + \kappa_z I_2(R^e) \; ; \; n,m = 1,\ldots,4, \tag{18}$$

$$F_n^e = \rho I_3(R^e) + \sum_{j=1}^{4} q_j I_j(C^e) \; ; \; n = 1,\ldots,4, \tag{19}$$

where it has been assumed that the tensor κ and the function $\rho(r,z)$ are constant over a given element, and that the function $q(s)$ is constant over a given boundary section between nodes. The quantities in (18) and (19) denoted by $I_i(D)$ are easily evaluated by the application of Gaussian quadrature to the expressions listed below:

$$I_1(R^e) = \int_{-1}^{+1}\int_{-1}^{+1} \frac{r^e}{|J|^e}\left(\frac{\partial r}{\partial \eta}^e \frac{\partial N}{\partial \xi}^e_n - \frac{\partial r}{\partial \xi}^e \frac{\partial N}{\partial \eta}^e_n\right)\left(\frac{\partial r}{\partial \eta}^e \frac{\partial N}{\partial \xi}^e_m - \frac{\partial r}{\partial \xi}^e \frac{\partial N}{\partial \eta}^e_m\right) d\xi d\eta, \quad (20.1)$$

$$I_2(R^e) = \int_{-1}^{+1}\int_{-1}^{+1} \frac{r^e}{|J|^e}\left(\frac{\partial z}{\partial \xi}^e \frac{\partial N}{\partial \eta}^e_n - \frac{\partial z}{\partial \eta}^e \frac{\partial N}{\partial \xi}^e_n\right)\left(\frac{\partial z}{\partial \xi}^e \frac{\partial N}{\partial \eta}^e_m - \frac{\partial z}{\partial \eta}^e \frac{\partial N}{\partial \xi}^e_m\right) d\xi d\eta, \quad (20.2)$$

$$I_3(R^e) = \int_{-1}^{+1}\int_{-1}^{+1} r^e N_n^e |J|^e d\xi d\eta, \quad (20.3)$$

$$I_1(C^e) = \int_{-1}^{+1} r^e N_n^e \frac{\partial r^e}{\partial \xi} [1 + (z_2^e - z_1^e)^2/(r_2^e - r_1^e)^2]^{\frac{1}{2}}\, d\xi \;;\; \eta=-1, \quad (20.4)$$

$$I_2(C^e) = \int_{-1}^{+1} r^e N_n^e \frac{\partial r^e}{\partial \eta} [1 + (z_3^e - z_2^e)^2/(r_3^e - r_2^e)^e]^{\frac{1}{2}}\, d\eta \;;\; \xi=1, \quad (20.5)$$

$$I_3(C^e) = \int_{+1}^{-1} r^e N_n^e \frac{\partial r^e}{\partial \xi} [1 + (z_4^e - z_3^e)^2/(r_4^e - r_3^e)^2]^{\frac{1}{2}}\, d\xi \;;\; \eta=1, \quad (20.6)$$

$$I_4(C^e) = \int_{+1}^{-1} r^e N_n^e \frac{\partial r^e}{\partial \eta} [1 + (z_1^e - z_4^e)^2/(r_1^e - r_4^e)^2]^{\frac{1}{2}}\, d\eta \;;\; \xi=-1, \quad (20.7)$$

where $|J|^e$ is the Jacobian determinant

$$|J|^e = \left|\frac{\partial z}{\partial \xi}^e \frac{\partial r}{\partial \eta}^e - \frac{\partial z}{\partial \eta}^e \frac{\partial r}{\partial \xi}^e\right|. \quad (21)$$

The total matrix [S] and the total vector {F} are now obtained through the use of (14).

Modification of the Algorithm for 2-D Cartesian Systems

The mathematical model we have just given for cylindrically symmetric systems can, with only minor modification, be applied to two-dimensional Cartesian systems.

In Cartesian coordinates, equations (1) and (3) are, respectively,

$$\frac{\partial}{\partial x}\left(K_x \frac{\partial \phi}{\partial x}\right) + \frac{\partial}{\partial y}\left(K_y \frac{\partial \phi}{\partial y}\right) - \rho(x,y) = 0 \tag{22}$$

and

$$I(\phi) = \int_R \tfrac{1}{2}\left[K_x\left(\frac{\partial \phi}{\partial x}\right)^2 + K_y\left(\frac{\partial \phi}{\partial y}\right)^2 + 2\rho\phi\right]dxdy + \int_C q\,\phi\,ds = \text{minimum}. \tag{23}$$

Carrying the transformation through, we find the following expressions for the coupling matrix [S] and the boundary vector {F}:

$$S^e_{nm} = \int_{R^e} K_x \frac{\partial N^e_m}{\partial x}\frac{\partial N^e_n}{\partial x}\,dxdy + \int_{R^e} K_y \frac{\partial N^e_m}{\partial y}\frac{\partial N^e_n}{\partial y}\,dxdy \;;\; n,m = 1,\ldots,4, \tag{24}$$

$$F^e_n = \int_{C^e} q\,N^e_n\,ds + \int_{R^e} \rho\,N^e_n\,dxdy \;;\; n = 1,\ldots,4. \tag{25}$$

Since we have used the same type of element in both cases, all the statements made regarding regions and boundaries in the cylindrical development are valid here. In addition, the expressions given for the matrix $[N]^e$ in the cylindrical case are identical to those for the Cartesian case.

It is now apparent that all of the difference between the two systems is contained in the quantities

$$I_i\,(R^e)\ ;\ i = 1,2,3$$

and

$$I_i\,(C^e)\ ;\ i = 1,2,3,4.$$

Making the transformation $z \to x$ and $r \to y$ in (17), we find for the Cartesian case

$$I_1(R^e) = \int_{-1}^{+1}\int_{-1}^{+1} \frac{1}{|J|^e}\left(\frac{\partial y^e}{\partial \eta}\frac{\partial N_n^e}{\partial \xi} - \frac{\partial y^e}{\partial \xi}\frac{\partial N_n^e}{\partial \eta}\right)\left(\frac{\partial y^e}{\partial \eta}\frac{\partial N_m^e}{\partial \xi} - \frac{\partial y^e}{\partial \xi}\frac{\partial N_m^e}{\partial \eta}\right)d\xi\,d\eta, \quad (26.1)$$

$$I_2(R^e) = \int_{-1}^{+1}\int_{-1}^{+1} \frac{1}{|J|^e}\left(\frac{\partial x^e}{\partial \xi}\frac{\partial N_n^e}{\partial \eta} - \frac{\partial x^e}{\partial \eta}\frac{\partial N_n^e}{\partial \xi}\right)\left(\frac{\partial x^e}{\partial \xi}\frac{\partial N_m^e}{\partial \eta} - \frac{\partial x^e}{\partial \eta}\frac{\partial N_m^e}{\partial \xi}\right)d\xi\,d\eta, \quad (26.2)$$

$$I_3(R^e) = \int_{-1}^{+1}\int_{-1}^{+1} N_n^e\,|J|^e\,d\xi\,d\eta, \quad (26.3)$$

$$I_1(C^e) = \int_{-1}^{+1} N_n^e \frac{\partial y^e}{\partial \xi}\left[1 + (x_2^e - x_1^e)^2/(y_2^e - y_1^e)^e\right]^{\frac{1}{2}}d\xi\ ;\ \eta = -1, \quad (26.4)$$

$$I_2(C^e) = \int_{-1}^{+1} N_n^e \frac{\partial y^e}{\partial \eta}\left[1 + (x_3^e - x_2^e)^2/(y_3^e - y_2^e)^2\right]^{\frac{1}{2}}d\eta\ ;\ \xi = 1, \quad (26.5)$$

$$I_3(C^e) = \int_{+1}^{-1} N_n^e \frac{\partial y^e}{\partial \xi}\left[1 + (x_4^e - x_3^e)^2/(y_4^e - y_3^e)^2\right]^{\frac{1}{2}}d\xi\ ;\ \eta = 1, \quad (26.6)$$

$$I_4(C^e) = \int_{+1}^{-1} N_n^e \frac{\partial y^e}{\partial \eta}\left[1 + (x_1^e - x_4^e)^2/(y_1^e - y_4^e)^2\right]^{\frac{1}{2}}d\eta\ ;\ \xi = -1, \quad (26.7)$$

where

$$|J|^e = \left| \frac{\partial x}{\partial \xi}^e \frac{\partial y}{\partial \eta}^e - \frac{\partial x}{\partial \eta}^e \frac{\partial y}{\partial \xi}^e \right|. \qquad (27)$$

SOLUTION OF THE DIFFERENCE EQUATIONS

From (10) it is obvious that the matrix [S] is symmetric. Since a given node is coupled only with the nodes of the elements which share it, [S] is block tridiagonal. In addition, if we assume that

$$\kappa_r > 0 \; ; \; \kappa_z > 0,$$
$$r \geq 0 \; ; \; z \geq 0 \tag{28}$$

(so that (1) is elliptic), it can be shown that [S] is positive definite. We may therefore apply the method of "normalized successive block overrelaxation" [3] to the solution of the system (13).

Partitioning [S] by rows into block tridiagonal form yields

$$[S] = \begin{bmatrix} B_1 & C_2 & & & 0 \\ C_2 & B_2 & C_3 & & \\ & \ddots & \ddots & \ddots & \\ & & & & C_L \\ 0 & & & C_L & B_L \end{bmatrix} \tag{29}$$

where the submatrices B_i and C_i are of order n_i, n_i being the number of mesh points on the ith row, and are all tridiagonal. Also, since [S] is symmetric and positive definite, it follows that all of the submatrices B_i ; $i = 1,\ldots,L$ are symmetric and positive definite:

$$B_i = \begin{bmatrix} b_1 & c_2 & & & 0 \\ c_1 & b_2 & c_2 & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & c_{n_i-1} \\ 0 & & & c_{n_i-1} & b_{n_i} \end{bmatrix} \tag{30}$$

Since B_i is a real symmetric and positive definite tridiagonal matrix, it has the unique factorization

$$B_i = D_i T_i' T_i D_i \quad ; \; i = 1,\ldots L, \tag{31}$$

where

$$D_i = \begin{bmatrix} d_1 & & & \\ & d_2 & & 0 \\ & & \ddots & \\ 0 & & & d_{n_i} \end{bmatrix} ; \; d_1 = b_1^{\frac{1}{2}} \quad ; \; d_j = \left[b_j - (c_{j-1}/d_{j-1})^2\right]^{\frac{1}{2}}, \quad j = 2,\cdots,n_i$$

and (32)

$$T_i = \begin{bmatrix} 1 & e_1 & & & \\ & 1 & e_2 & & 0 \\ & & \ddots & \ddots & \\ 0 & & & 1 & e_{n_i-1} \end{bmatrix} \quad ; \; e_j = c_j/(d_j d_{j+1}), \quad j = 1,\cdots,n_i-1 .$$

With the vector of ϕ values and the vector of F Values for the ith row denoted by Φ_i and G_i respectively, and with

$$X_i \equiv D_i \Phi_i \; , \; M_i \equiv -D_i^{-1} G_i \; ; \; i = 1,\ldots,L, \tag{33.1}$$

$$P_i \equiv -D_{i-1}^{-1} C_i D_i^{-1} \quad ; \quad i = 2,\ldots,L, \tag{33.2}$$

normalized successive block overrelaxation is defined by

$$(T_i' \; T_i) \overset{*}{X}_i^{(m+1)} \equiv P_i X_{i-1}^{(m+1)} + P_{i+1} X_{i+1}^{(m)} + M_i, \tag{34.1}$$

$$X_i^{(m+1)} = \omega \left(\overset{*}{X}_i^{(m+1)} - X_i^{(m)}\right) + X_i^{(m)}, \tag{34.2}$$

where m is the iteration number and ω is the overrelaxation factor.

The system of equation's defined by (34.1) can be solved directly by the following algorithm:

$$h_1 = g_1 \quad , \quad h_{j+1} = g_{j+1} - e_j h_j \quad ; \quad j = 1,\ldots,n_i-1, \tag{35.1}$$

$$x_{n_i} = h_{n_i} \quad , \quad x_j = h_j - e_j x_{j+1} \quad ; \quad j = 1,\ldots,n_i-1, \tag{35.2}$$

where we have denoted the j component of the right-hand side of (34.1) by g_j. After the iteration given by (34.2) has converged, Φ_i can be obtained from the solution X_i by application of the relation

$$\Phi_i \equiv D_i^{-1} X_i . \tag{36}$$

Note that the entire process defined by (34.1) and (34.2) takes at most nine multiplications and ten additions per component per iteration, which is the same number as the point-overrelaxation method requires. If the number of iterations is large, the time required to set up the matrices D_i , T_i , M_i, and P_i , and to obtain Φ_i from X_i, will be small compared to the total execution time. Since the rate of convergence of block iteration is theoretically faster than point iteration, its use in JASON will result in more efficient computation.

The optimum overrelaxation factor may be estimated from the vector iterates just as in point overrelaxation [3]. Bounds on the largest eigenvalue of the iteration matrix are obtained by

$$\underline{\lambda}_m \equiv \min_{1 \le i \le n} (x_i^{(m+1)}/x_i^{(m)}) \quad , \quad \overline{\lambda}_m \equiv \max_{1 \le i \le n} (x_i^{(m+1)}/x_i^{(m)}), \quad (37.1)$$

where m is the iteration number and n is the total number of mesh points. We now estimate ω_{opt} by the formula

$$\omega_{opt} = a[2/(1+\sqrt{1-\underline{\lambda}_m})] + b[2/(1+\sqrt{1-\overline{\lambda}_m})] . \quad (37.2)$$

The weighting factors a and b are to be chosen appropriately according to the estimated stability of the problem, and such that a + b = 1. When the difference $\overline{\lambda}_m - \underline{\lambda}_m$ is less than some small value, it is assumed that ω_{opt} has been found, and no new estimates are made.

GENERAL DESCRIPTION OF THE PROGRAM

Mesh Generation

The use of quadrilaterals in the derivation of the JASON algorithm enables us to approximate any arbitrary boundary curve by a union of element sides. In this manner, boundary curves will always lie along mesh lines. As finer mesh spacing will be required in some regions to fit boundaries than in others, a nonuniform mesh is required. Construction of such a mesh by hand can be a formidable task, particularly since, for reasons of stability and accuracy, mesh variations should be smooth. To alleviate such problems, JASON has been provided with a mesh generator.

The method of generation used is that of "Equipotential Zoning" [4]. In this method, the mesh lines are regarded as two intersecting sets of equipotentials Φ and Ψ, which satisfy Laplace's equation in the interior of the region and take on successive integral values along the boundary. Performing a hodograph transformation on the equations $\nabla^2\Phi = 0$ and $\nabla^2\Psi = 0$ produces two new equations which will yield the coordinates of the mesh points (intersections of Φ lines and Ψ lines) directly. These equations are replaced by their representation in finite differences and solved by successive point overrelaxation.

Input to the generator is by regions. Each region is defined by specifying the logical and global coordinates of points on the region boundary (logical coordinates specify which mesh lines the point belongs to; global coordinates specify the position of the point). Except in the case of curves, it is sufficient to specify only the points which

are logical corners of the region. Other boundary points will be computed by linear interpolation. For curved boundaries, sufficient points to define the shape of the curve must be specified. These points are assumed to be all of the mesh points on the curve, and none will be generated. In addition to the boundary-point input for each region, values for K_r , K_z , and ρ, which are assumed to be constant over the region, are required. If they are not specified in the input for the region, they are given the standard values $K_r = K_z = 1.$, $\rho = 0$.

After all the region information has been input, the boundary conditions for the problem are specified. Dirichlet boundary conditions may be imposed at any point in the mesh. Logical and global coordinates for each boundary point, along with a value of ϕ, are input to completely define the boundary. Provision is made for generation of a Dirichlet boundary along a given mesh line. In this case, ϕ is assumed to be constant along the boundary. To avoid costly testing when solving the system of equations (13), all Dirichlet boundary conditions are incorporated directly into the system by appropriate modification of [S] and {F}.

Neumann boundary conditions are restricted to "universe" boundaries and to boundaries between regions having K_r , $K_z \neq 0$ and K_r , $K_z = 0$. The boundary is considered to be composed of the sides of elements in the region having K_r , $K_z \neq 0$. As there is no coupling to mesh points in the region having K_r , $K_z = 0$, its elements need not be considered in the computation. Values of q for each element side composing the boundary, along with the element and side indices, are input to define the boundary. Since the vector {F} incorporates

these conditions directly into the difference equations, no testing will be required when solving the system (13).

Before generation is initiated, all mesh input is scanned for errors. On detection of an error, the program prints out an appropriate comment and sets a flag to terminate execution at the end of the scan. The generated mesh is scanned for errors in the same manner as the data. A plot of the mesh (fig. 1), produced in all cases, aids in the elimination of errors and will reveal any poor "zoning." Since the speed of convergence, as well as the quality of the solution, depends on the smoothness of the mesh, good zoning is of utmost importance.

All of the remaining "problem constants" are input following the mesh generation. A complete description of all the input to JASON is given in Appendix C of reference [7]. The standard values assumed by the problem constants, if unspecified in the input, are also given here. Appendix D of [7] lists the input and output for several test cases.

Main Computational Phase

This section of JASON contains the programming for the algorithm developed in the first half of this paper. To ensure minimum execution time, key subroutines in this section have been optimally coded in machine language (6600 COMPASS). Since it may be difficult to estimate the execution time required for some problems, a restart procedure is provided. The mesh coordinates and the iteration matrices are stored on magnetic tape as soon as they are generated. If it appears that the time limit will be exceeded before the iteration has converged, the current values for the elements of the solution vector are dumped on

the tape. Execution may then be initiated at this point, in a succeeding run, by reading in the tape. A flag, input at the start of the program, will cause the tape to be read and execution to skip directly to the iteration phase.

During the iteration phase, selected parameters are printed every few cycles to monitor the convergence. Two of these, ϵ and δ, we define in the following manner:

$$\epsilon = || x^n - x^{n-1} ||/|| x^n ||, \tag{38.1}$$

$$\delta = || x^n - x^{n-1} ||/|| x^{n-1} - x^{n-2} ||, \tag{38.2}$$

where n denotes the iteration number and x denotes an element of the solution vector X, defined by (33.1). Observe that ϵ is just the Euclidean norm of the relative error. The parameter δ is a measure of the rate of convergence. For convergence, it is normally required that $\epsilon < 10^{-7}$. Once this criterion is satisfied, the solution vector is calculated from (36) and the result is both printed and stored on magnetic tape. An equipotential plot (fig. 3 and 4) may also be produced at this point.

Edit

In most cases, it is not the potential that is of interest, but its derivatives. A set of subroutines for calculating derivatives is therefore written into JASON. The edit routine, similar to that used in another nonuniform mesh code, TRIM [5], fits a harmonic polynomial in the least-squared sense to a specified set of mesh points. The derivatives of the polynomial are then evaluated as approximations to the derivatives of the potential (to produce better averaging of error,

the derivatives are evaluated at the centroids of mesh elements). An edit of all mesh elements is automatically taken following convergence.

POSSIBLE EXTENSIONS OF JASON

General Anisotropy

In deriving the system of equations (13) from (1), it was assumed that the tensor K was diagonal in the global coordinate system. Obviously, this restricts us to consideration of problems in which the principal axes of all materials are parallel to the global coordinate axes. This restriction can be removed by assuming K to be diagonal in some local coordinate system for each element in the mesh. The axes of this local coordinate system are then parallel to the principal axes of the material in that element. Equation (1) is now the governing differential equation in the local coordinate system for each element, and expression (19) may be evaluated for each element using local coordinates. These results are then transformed to the global system before assembly, as specified by (14), into the total [S] matrix and {F} vector. Therefore, if we know the transformation from local to global coordinates for each element, problems with general anisotropy may be considered.

Nonlinear Problems

If we allow the tensor K to be a function of ϕ or its derivatives, equation (1) becomes nonlinear. This, however, in no way affects our derivation of the system of equations (13). Only our method of solution needs to be changed. A method such as "Block Nonlinear Successive Overrelaxation" [6] could be applied to solve the nonlinear system (13). As K will change in value as the iteration progresses, it is obvious from (18) that more storage will be required than for

linear problems.

Acknowledgments

We wish to thank Mr. J. S. Colonias for his interest and support throughout this work. We also wish to thank Dr. P. Concus, to whom we are indebted for checking the derivation of the equations involved.

REFERENCES

[1] O. C. Zienkiewics, The Finite Element Method in Structural and Continuum Mechanics, (McGraw - Hill, Inc., London, 1967)

[2] O. Bolza, Lectures on the Calculus of Variation (Dover Publications, Inc., 1961)

[3] E. H. Cuthill, and R. S. Varga, A Method of Normalized Block Iteration, Journal of the Association for Computing Machinery, Vol. 6 (1959), pp. 236-244.

[4] A. Winslow, Equipotential Zoning of Two-Dimensional Meshes, University of California, Lawrence Radiation Laboratory report UCRL-~~7321~~ 7312 (1963)

[5] J. S. Colonias, TRIM: A Magnetostatic Computer Program for the CDC 6600, University of California, Lawrence Radiation Laboratory report UCRL-18439 (1968)

[6] P. Concus, Numerical Solution of the Minimal Surface Equation by Block Nonlinear Successive Overrelaxation, University of California, Lawrence Radiation Laboratory report UCRL-18238 (1968)

[7] S. Sackett and R. Healey, JASON - A Digital Computer Program for the Numerical Solution of the Linear Poisson Equation $\nabla\cdot(\kappa\nabla\phi) = \rho$, University of California, Lawrence Radiation Laboratory report UCRL-18721 (February 1969)

FIGURE LEGEND

Fig. 1 Example of mesh generation results.

Fig. 2 Comparison of JASON solution with analytic solution for a 7-cm-radius grounded conducting sphere in a uniform field.

Fig. 3 Equipotential plot of solution to a problem with mutiple boundaries.

Fig. 4 Equipotential plot of solution to a problem with $\frac{\partial \phi}{\partial n} = 0$ on the upper boundary.

Note: All examples have cylindrical symmetry about the z axis.

7312
7880
7784

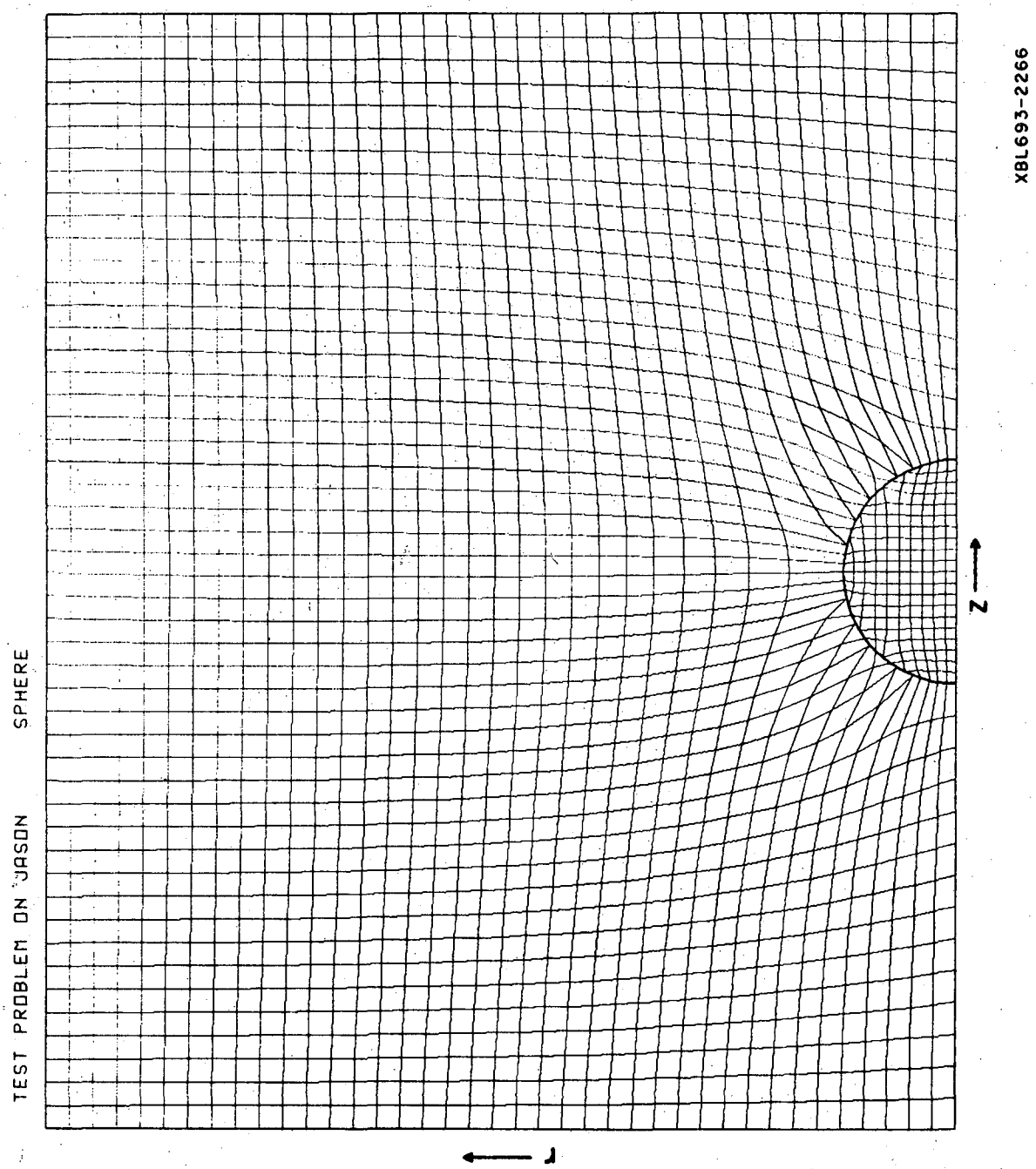


Fig. 1

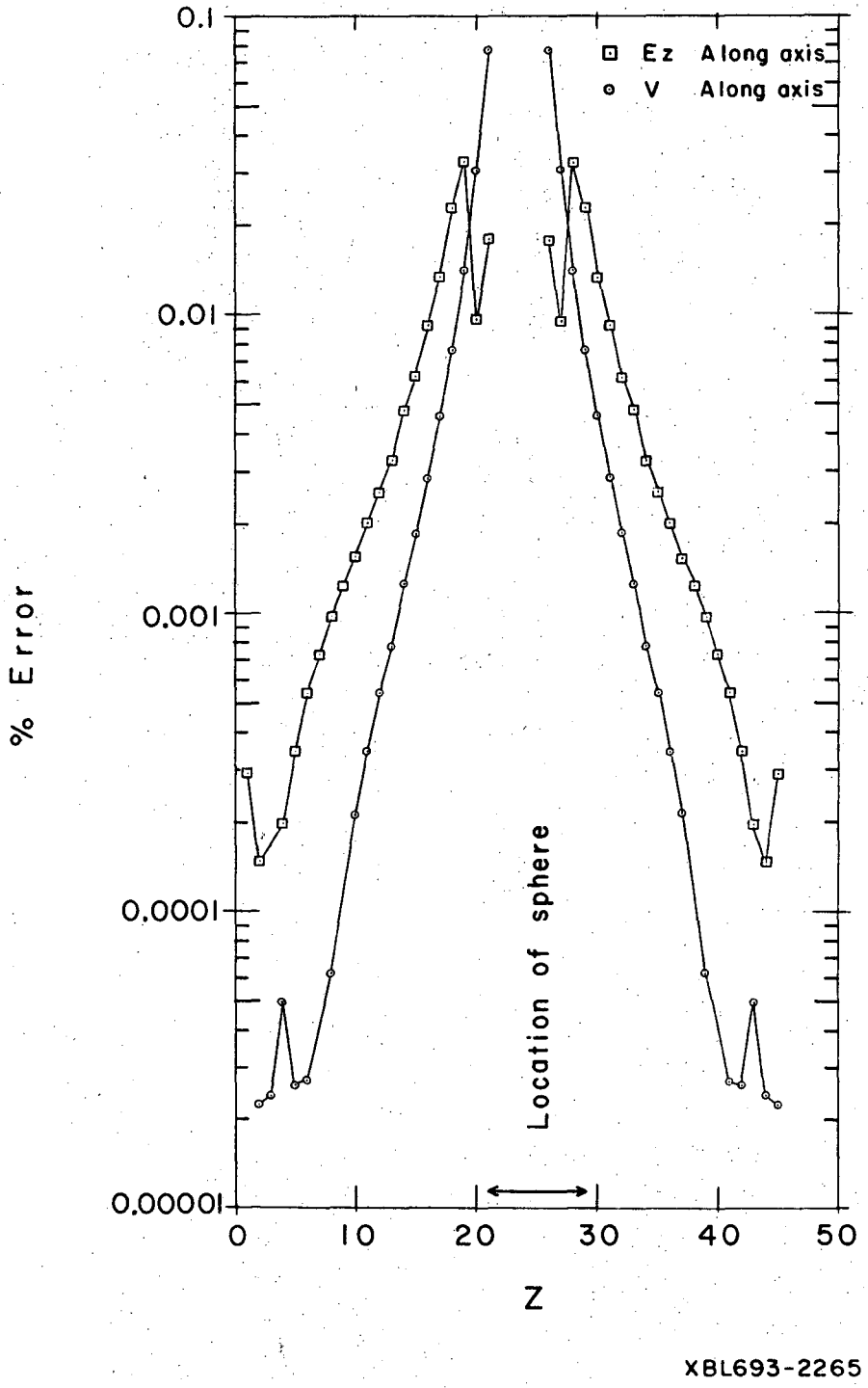

0.1
0.01
0.001
0.0001
0.00001
% Error
Ez Along axis
V Along axis
Location of sphere
0
10
20
30
40
50
Z
XBL693-2265

Fig. 2

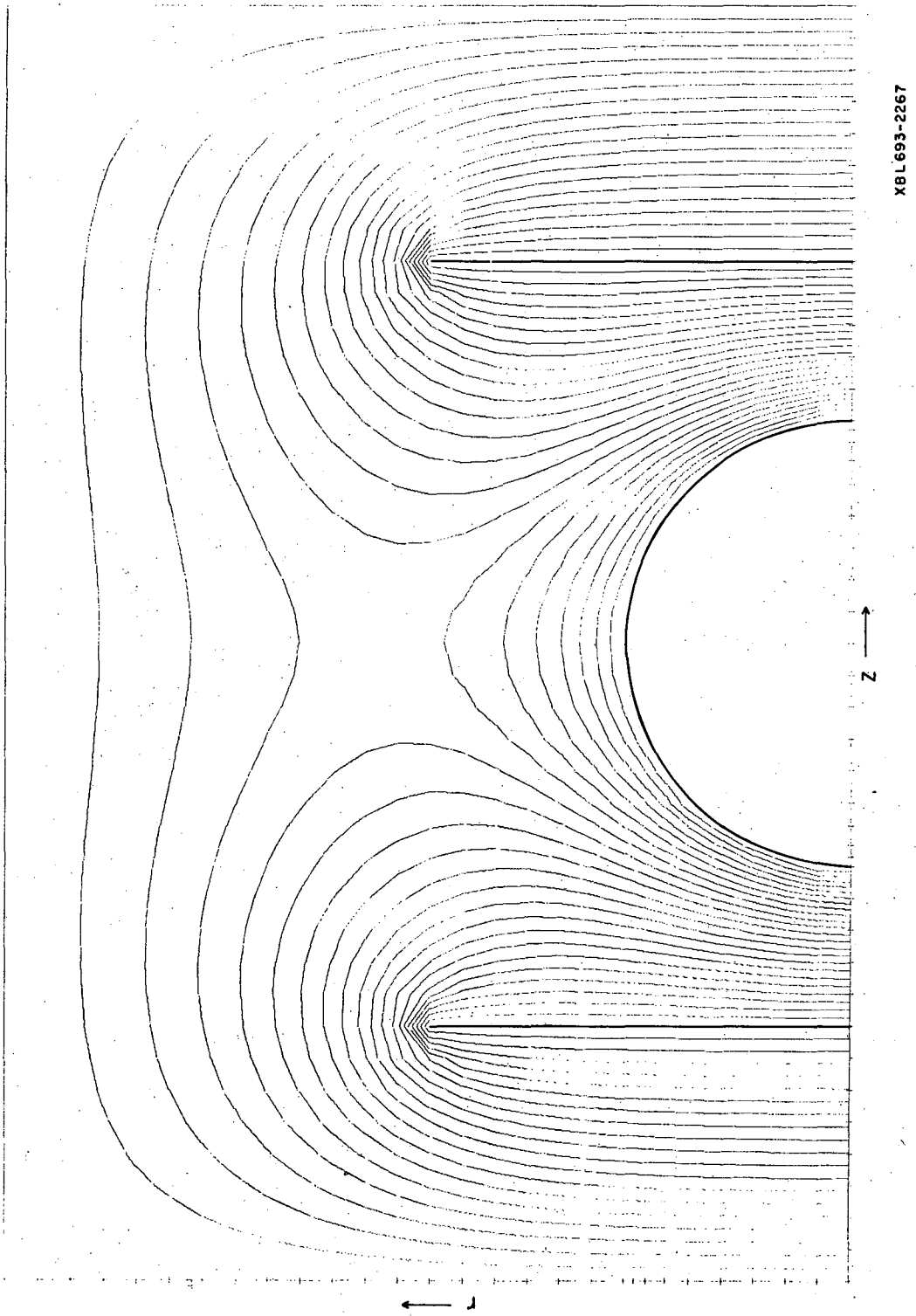


Fig. 3

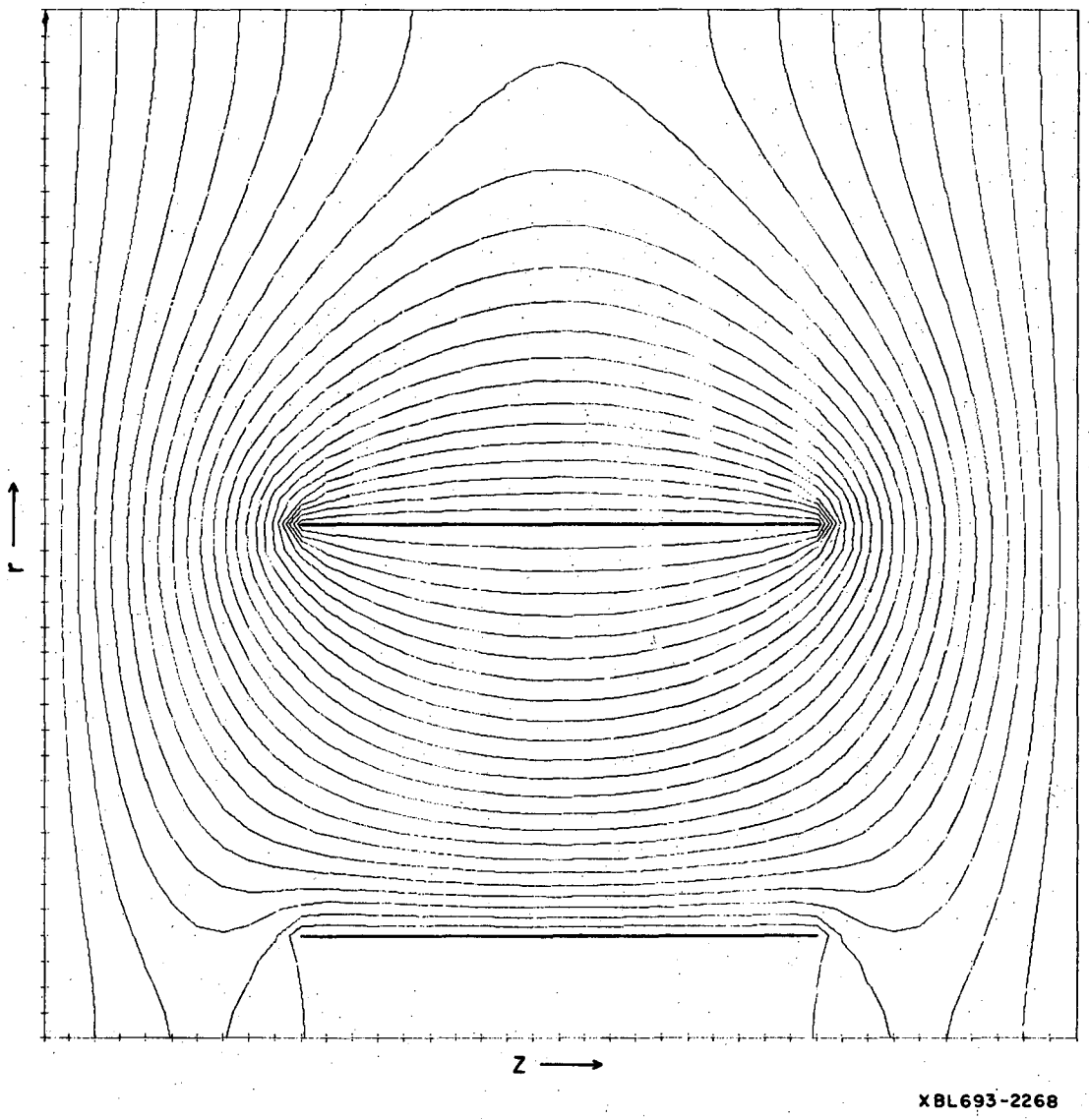


Fig. 4

LEGAL NOTICE

This report was prepared as an account of Government sponsored work. Neither the United States, nor the Commission, nor any person acting on behalf of the Commission:

A. Makes any warranty or representation, expressed or implied, with respect to the accuracy, completeness, or usefulness of the information contained in this report, or that the use of any information, apparatus, method, or process disclosed in this report may not infringe privately owned rights; or

B. Assumes any liabilities with respect to the use of, or for damages resulting from the use of any information, apparatus, method, or process disclosed in this report.

As used in the above, "person acting on behalf of the Commission" includes any employee or contractor of the Commission, or employee of such contractor, to the extent that such employee or contractor of the Commission, or employee of such contractor prepares, disseminates, or provides access to, any information pursuant to his employment or contract with the Commission, or his employment with such contractor.

TECHNICAL INFORMATION DIVISION
LAWRENCE RADIATION LABORATORY
UNIVERSITY OF CALIFORNIA
BERKELEY, CALIFORNIA 94720