MARTIN MARIETTA ENERGY SYSTEMS LIBRARIES

3 4456 0069914 4

ORNL/TM-9816

ornl

OAK RIDGE NATIONAL LABORATORY

MARTIN MARIETTA

A Program for Solving the 3-Dimensional LaPlace Equation via the Boundary Element Method

L. J. Gray

OAK RIDGE NATIONAL LABORATORY
CENTRAL RESEARCH LIBRARY
CIRCULATION SECTION
4500N ROOM 175
LIBRARY LOAN COPY
DO NOT TRANSFER TO ANOTHER PERSON
If you wish someone else to see this report, send in name with report and the library will arrange a loan.

OPERATED BY
MARTIN MARIETTA ENERGY SYSTEMS, INC.
FOR THE UNITED STATES
DEPARTMENT OF ENERGY

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
NTIS price codes—Printed Copy: A03; Microfiche A01

This report 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.

ORNL/TM-9816

Engineering Physics and Mathematics Division

Mathematical Sciences Section

A PROGRAM FOR SOLVING THE 3-DIMENSIONAL LAPLACE EQUATION VIA THE BOUNDARY ELEMENT METHOD

L. J. Gray

Date Published - September 1986

Research sponsored by the
Y-12 Development Division
Y-12 Plant
U.S. Department of Energy

Prepared by the
Oak Ridge National Laboratory
Oak Ridge, Tennessee 37831
operated by
Martin Marietta Energy Systems, Inc.
for the
U.S. Department of Energy
under Contract DE-AC05-84OR21400

MARTIN MARIETTA ENERGY SYSTEMS LIBRARIES

3 4456 0069914 4

CONTENTS

Abstract 1

I. Introduction 1

II. Boundary Element Method 2

III. Test Results 5

IV. Program Listing 7

References 16

Appendix A 23

Appendix B 25

Appendix C 26

A PROGRAM FOR SOLVING THE 3-DIMENSIONAL LAPLACE EQUATION VIA THE BOUNDARY ELEMENT METHOD

L. J. Gray

ABSTRACT

A subroutine for solving the interior three dimensional LaPlace equation $\nabla^2 V = 0$ by the boundary element method is described. The bounding surface is defined by triangular elements, each element defined by three nodes (vertices) in appropriate order for calculating the outward normal. The boundary data at a boundary node P is either the value of the potential $V(P)$ or the normal derivative $\frac{\partial V}{\partial \vec{n}}(P)$. The required integrals over the triangular elements are computed using linear interpolation from the nodal values. This routine has been especially tailored in several ways for use in modeling electrochemical processes.

I. INTRODUCTION

The three dimensional LaPlace Equation

$$\nabla^2 V = 0 \tag{1.1}$$

together with specified boundary conditions, occurs in many areas of science and engineering, and there are many numerical techniques for obtaining the solution. Although finite differences and finite elements are the most widely known, the boundary element method [2] is particularly effective in the context of computer modeling of electrochemical deposition or machining processes. In these simulations, LaPlace's equation must be solved repeatedly in regions with possibly irregular boundaries which are evolving in time; thus speed of execution and ability to handle arbitrary geometries are important considerations in the choice of a numerical scheme. The boundary element method, which replaces the differential equation for the potential V by an equivalent integral equation for the unknown boundary values, is well suited to complicated boundaries; furthermore, the algorithm can be tailored to accommodate the special circumstances that arise in electrochemical applications, thereby reducing execution time. A more complete discussion of the advantages of the boundary element method with regards to electrochemical modeling can be found in reference [3].

The program described herein is an implementation of the boundary element approach for solving the three dimensional LaPlace equation, specifically designed for electrochemical applications. There are two principal differences between this algorithm and a completely general procedure for solving an interior problem. First, when the boundary conditions specify the value of the current density $\frac{\partial V}{\partial \vec{n}} \equiv V_{\vec{n}}$ on a node, this value is assumed to be zero; in plating applications, a nonpotential surface is always either an insulated or symmetry surface, so this assumption is valid. Incorporating this knowledge can greatly reduce the number of integrations required and significantly decrease the execution time. Second, in order to calculate the electrode polarization efficiently, the finite linear system of equations which replaces the integral equation is accumulated in two matrices rather than one. Although this procedure nearly doubles the storage required, it results in tremendous time savings during the course of a simulation.

The outline of this report is as follows. Section II contains a description of the method, along with a discussion of the special modifications. Section III describes several sample calculations, along with an explanation of the use of "double nodes", and Section IV contains a listing of the program. The details involved in computing the required integrals and a sample main program for calling the subroutine can be found in the appendices.

II. BOUNDARY ELEMENT METHOD

Let D be a closed bounded region in R^3 with boundary ∂D; a point in R^3 will be denoted by $X = (x, y, z)$. The potential function $V(X)$ satisfies LaPlace's Equation inside D, and the boundary conditions supply, for each $X \in \partial D$, either the potential $V(X)$ or the current density $V_{\vec{n}}(X)$ (the more general case, wherein a linear relationship between the potential and the current density is specified, can also be treated but this situation does not arise in electrochemical problems). The fundamental solution or Green's function $G(X, X_0)$ is defined by

$$G(X, X_0) = \frac{1}{4\pi \| X - X_0 \|} , \tag{2.1}$$

where $\| X - X_0 \|$ is the distance between X and X_0. Since G is the potential of a point charge at X_0, it satisfies LaPlace's equation everywhere except X_0, and in fact,

$$\nabla^2 G(X, X_0) = -\tfrac{1}{2}\, \delta\, (X, X_0) , \tag{2.2}$$

where the differentiation is with respect to X and $\delta\, (X, X_0)$ is the Dirac delta function. Using Green's Theorem, it can be shown that LaPlace's equation becomes

$$\phi(X_0)V(X_0) - \int\!\!\int_{\partial D} [G(X, X_0)\nabla V(X) - V(X)\nabla G(X, X_0)] \bullet \vec{n}\, dS = 0 \tag{2.3}$$

where again the gradient and the surface integration are with respect to the X variable, and X_0 is an arbitrary point on ∂D. The function $\phi(P)$ denotes the interior solid angle of the surface at P divided by 4π. For example, if the surface is smooth at P, $\phi(P) = \tfrac{1}{2}$, whereas if P is the corner of a cube, $\phi(P) = 0.125$. Eq. (2.3) provides, for each X_0, a linear relationship between the values of V and $V_{\vec{n}}$ on ∂D, and is the starting point for the boundary element method.

A closed form solution of the integral equation (2.3) usually cannot be achieved. An approximate numerical solution can be obtained by replacing the boundary surface with a union of M triangles, $\bigcup_{m=1}^{M} T_m$, where the K vertices $\{Y_k\}$ of the triangles (henceforth called nodes) lie on ∂D. By using "double nodes" if necessary (to be defined below), it can be assumed that for each triangle T_m, either V or $V_{\vec{n}}$ is specified at all three nodes of T_m. When integrating V(X) and $V_{\vec{n}}(X)$ over T_m, the function values are approximated by linear interpolation from the nodal values. Using this approximation and rewriting Eq. (2.3) in the form

$$\phi(X_0)V(X_0) + \sum_{m=1}^{M} \iint_{T_m} V(X)\nabla G(X, X_0)\cdot\vec{n}dS$$
$$= \sum_{m=1}^{M} \iint_{T_m} G(X, X_0)\nabla V(X)\cdot\vec{n}dS \ , \tag{2.4}$$

a linear equation in the K unknown boundary values of V and $V_{\vec{n}}$ is obtained, provided X_0 is chosen to be one of the node points. Letting X_0 be each of the nodes yields a system of K linear equations which, following Eq. (2.4), can be expressed as

$$HU = LW \tag{2.5}$$

Here, H and L are K by K matrices and

$$U = \begin{pmatrix} v_1 \\ \cdot \\ \cdot \\ v_K \end{pmatrix} \qquad W = \begin{pmatrix} w_1 \\ \cdot \\ \cdot \\ w_K \end{pmatrix} \tag{2.6}$$

where $v_k = V(Y_k)$ and $w_k = V_{\vec{n}}(Y_k)$. Depending upon which values are unknown, the equations can be rearranged in the usual form $AX = B$. Although the matrices A and B can be calculated directly without first forming H and L (and this would in fact reduce the computer storage required), for electrochemical applications it is more desirable to calculate H and L separately, and then do the rearrangement; the reasons for this will be discussed below. Having put the equations in the form $AX = B$, X is solved for by an LU factorization algorithm.

The calculation of the matrix H will now be described in detail, the matrix L following along similar lines. By definition, the (k, l) matrix element of H is the coefficient of $V(Y_l)$ on the left side of Eq. (2.4) when $X_0 = Y_k$. In order to conveniently evaluate the surface integrals over the triangles T_m, a local coordinate system will be used. Label the nodes of T_m as 1, 2, and 3, subject to the following two conditions: the node labeled as 1 is arbitrary unless one of the nodes is Y_k, in which case this point is designated to be node 1; second, if necessary interchange nodes 2 and 3 in order to have the distance from 1 to 3 greater than or equal to the distance from 1 to 2. Node 1 is taken as the origin of the coordinate system, the plane of T_m as the $z=0$ plane, and the line from 2 to 3 as the positive y direction (see Fig. 1). There is no harm in doing this, with the proviso that one keeps track of the outward normal, which is now either the plus or minus z direction.

In accordance with the linear interpolation assumption, the value of the potential at any point (r, θ) of T_m is taken to be

$$V(r, \theta) = c + ar\cos\theta + br\sin\theta, \tag{2.7}$$

where (r, θ) denotes polar coordinates in the x-y plane. A simple calculation shows that the coefficients a, b, c are linear combinations of V_1, V_2, V_3, the values of V at the nodes 1, 2, and 3:

$$c = V_1$$

$$\begin{bmatrix} a \\ b \end{bmatrix} = \lambda \begin{bmatrix} r_3 \sin\theta_3 & -r_2 \sin\theta_2 \\ -r_3 \cos\theta_3 & r_2 \cos\theta_2 \end{bmatrix} \begin{bmatrix} V_2 - V_1 \\ V_3 - V_1 \end{bmatrix} \tag{2.8}$$

where $\lambda = (r_2 r_3 sin(\theta_3 - \theta_2))^{-1}$ and (r_2, θ_2), (r_3, θ_3) are the coordinates of the nodes 2, 3.

Note that

$$\nabla G(X, Y_k) \bullet \vec{n} = \frac{\hat{z}}{4\pi} \rho^{-3} \tag{2.9}$$

where

$$\rho = [(r\cos\theta - \hat{x})^2 + (r\sin\theta - \hat{y})^2 + \hat{z}^2]^{1/2} \tag{2.10}$$

is the distance from $X = (r\cos\theta, r\sin\theta, 0)$ to Y_k, $(\hat{x}, \hat{y}, \hat{z})$ being the coordinates of Y_k in the local system. Thus, if Y_k is a node of T_m, $\hat{z} = 0$, and the integral is zero (this can be seen directly: since $G(X, Y_k)$ is only a function of the distance from Y_k, $\nabla G(X, Y_k)$ must be in the direction from Y_k to X). For $\hat{z} \neq 0$, the surface integral can be written as

$$\int\int_{T_m} V(X) \nabla G(X, Y_k) \bullet \vec{n}\, dS = \tag{2.11}$$

$$\frac{\hat{z}}{4\pi} \left[c \int\int_{T_m} r\rho^{-3} dr d\theta + a \int\int_{T_m} r^2 \rho^{-3} \cos\theta\, dr d\theta + b \int\int_{T_m} r^2 \rho^{-3} \sin\theta\, dr d\theta \right].$$

For each of these integrals, the r integration can be performed in closed form, the details of which can be found in Appendix A. The θ integration is evaluated numerically using a Gaussian quadrature scheme. Eq. (2.11) together with Eq. (2.8) determine the matrix elements of H, with the exception of the term $\phi(Y_k)V(Y_k)$ of Eq. (2.4). The value of $\phi(Y_k)$ could be quite troublesome to compute for nonsmooth surfaces, so fortunately there is an easy method due to Brebbia. Note that this is the only term which contributes to the diagonal matrix element $h_{k,k}$ (if $Y_k \in T_m$ then the surface integral vanishes), and that the matrices H and L, while dependent upon the geometry of ∂D, are independent of the boundary data. Thus, if V is chosen to be identically 1 on the boundary, the solution is $V_{\vec{n}} = 0$ and

$$H \begin{bmatrix} 1 \\ 1 \\ \bullet \\ \bullet \\ 1 \end{bmatrix} = L \begin{bmatrix} 0 \\ 0 \\ \bullet \\ \bullet \\ 0 \end{bmatrix} = 0; \tag{2.12}$$

i.e., H has zero row sums. Consequently,

$$h_{k,k} = -\sum_{j \neq k} h_{k,j} \tag{2.13}$$

The analysis of the integrals

$$\int\int_{T_m} G(X, Y_k) \nabla V(X) \cdot \vec{n} \, dS \tag{2.14}$$

for the computation of the L matrix elements proceeds in exactly the same fashion (the details of the r integration for this case can also be found in Appendix A), with one exception. If Y_k is a vertex of T_m, then the integral is not necessarily zero and must be computed. This integral can be evaluated in closed form (see [1] and Appendix A), thus avoiding the possibility of numerical problems caused by the singularity of G lying on T_m.

There are two basic modifications of this algorithm which have been incorporated for electrochemical modeling purposes. First, as mentioned in the introduction, when the value of the current density is specified on the boundary, it is assumed to be zero. Thus, for any element T_m with $V_{\vec{n}}$ known, the integral in Eq. (2.14) (contributing to the matrix elements of L) is eventually multiplied by zero in the computation of B; hence, this calculation may be dispensed with. For many practical problems a significant portion of the bounding surface is insulated and a great many integrations can be avoided.

Second, taking into account polarization at the electrodes requires an iterative calculation in which LaPlace's Equation is solved several times with the same geometry, but with different values of the potential on the boundary (the insulated segments remaining insulated). As mentioned above, H and L, and thus also A are unaffected by such alterations in the boundary conditions; only the right hand side matrix B changes. Furthermore, since $V_{\vec{n}}$ always remains zero, B is determined by H and the new boundary values of the potential. Since L does not enter into the computation of B, the program computes H and L separately and then overwrites L with A, which in turn is overwritten by its LU factorization. Having saved H and the LU factors, the iterative process used to obtain the polarization can proceed with little additional work: a new right hand side B matrix is computed from the previous solution and the matrix H, and then a new solution is determined from the LU factors (the cycle continuing until convergence). Thus, at each iteration, obtaining the required solution of LaPlace's equation is essentially reduced to an inexpensive forward and back substitution. This can result in substantial savings of time during the course of a simulation.

III. TEST RESULTS

Two sample problems for which the numerical answers can be compared to an exact solution will be described. In the first test case, the region D is the unit cube; the specified boundary data has $V = 0$ on one face, $V = 1$ on the opposing face, and all other faces insulated (i.e. $V_{\vec{n}} = 0$). This problem was chosen to illustrate the use of "double nodes" at the edges of the cube where the boundary conditions change from potential to insulated. Because of the discontinuity in the normal derivative at this edge, special treatment is required in order to obtain an accurate result. In the boundary element method, this problem is easily handled by making each node on this intersection into two nodes [1].

Two calculations were performed for this insulated cube, one with a minimal number of nodes and elements and one with a significantly finer decomposition of the cube surface. The data for the crude mesh is small enough to construct by hand and can be completely displayed, thus serving as a convenient example of the use of double nodes. The second calculation comes closer to an actual problem and available mesh generation software (PIGS [4]) was employed to obtain the data; as computer generated geometry is the likely method for most practical applications, this calculation is an illustration of the automatic assignment of double nodes and boundary data.

The coarse discretization of the unit cube is shown in Figure 2. There are 14 nodal points, the corners together with the center of each face, and each face is divided into four elements. The $y = 0$ and $y = 1$ surfaces have the potential specified, with $V = 1$ and $V = 0$ respectively, and the four remaining faces are insulated. Note that the boundary condition at the eight corner nodes (labeled 1 through 8) is ambiguous, as they could either be potential or insulated points. This problem is dealt with by making these nodes into double nodes; that is, introducing eight additional nodes (labeled 9 through 16) with the same coordinates as the original points (i.e. node 9 is the same point as node 1). Which node number is used in the definition of a particular element depends upon the boundary condition for that element. For example, focusing on the double node [3,11], the element on the $y = 0$ face is given by nodes {17,3,4}, while the adjoining element on the insulated $z = 1$ face is specified by nodes {20,12,11}. Similarly, a neighboring element on the $x = 1$ surface is given by {19,11,10} because it too is insulated. (A complete list of the nodes and elements is given in Table 1; the parameter NODDEF is 0 for a potential node and 1 for an insulated node). The boundary element algorithm considers each node to be a distinct point, and thus the resulting matrix equation is of order 22.

The results of the calculation, shown in Table 1, agree with the exact answer, $V(x,y,z) = 1 - y$. If the boundary condition is the potential at the node, then the solution gives the current density $V_{\vec{n}}$ at this point and vice versa. Note, for example, that the computed value at node 7 (input as a potential node) is -1.0, which is the outward normal derivative on the $y = 1$ plane, whereas the value at its companion node 15 is 0.0, which is indeed the potential on this surface.

The subdivision of the cube surface for the second calculation is shown in Figure 3. Each edge was divided into five equal subintervals, and these points were used to generate a square grid on each face. Every small square was divided into two triangles via a diagonal, resulting in a total of 300 elements and 152 nodes. In the input data, each element was identified as being either a potential or insulated surface. Before executing the boundary element subroutine, the program first constructed the double nodes by checking if any node appeared on both types of surface elements; if so, a new node number was generated for that point and the element definitions were changed accordingly. The program listing for this part of the calculation can be found in Appendix C.

One further comment about the input data is required. The program assumes that the order of the node numbers defining the triangular elements specifies the positive orientation (i.e. outward normal) of the element. (Actually, as long as the surface is consistently defined as either inward or outward normal, everything is correct, with the sign of the computed current density depending upon which normal is chosen; this follows from examining Eq. (2.4), recalling that the term $\phi(X_0)$ is calculated from Eq. (2.13), and that all supplied values of the current density are zero.) One possible way to assure that the boundary is oriented correctly is to compute the inner product of the calculated normal with a specific vector; the sign of this inner product will indicate whether or not the

orientation of the element is the proper one. An example of this procedure is given in the program listing in Appendix C. As with the coarse discretization, the results for this simple problem were very accurate (ten digits were printed out and these agreed with the known solution).

The second problem consists of two concentric spheres centered at the origin, the inner sphere of radius $r = 1$ having a potential $V = 1$ and the outer sphere, $r = 2$, having a potential $V = 0$. Instead of solving the problem in this form, the symmetry will be utilized in order to reduce the size of the region (also, as indicated above, symmetry surfaces require less computation). The region to be considered (see Figure 4) is the 45° wedge formed by cutting the top half ($z \geqslant 0$) with the planes $y = 0$ and $x = y$. Thus, in addition to the spherical potential segments, the boundary is composed of (pieces of) annular regions in the $z = 0, y = 0$, and $x = y$ planes with boundary condition $V_{\vec{n}} = 0$. The exact solution is $V = -1 + 2/r$, $1 \leqslant r \leqslant 2$, and $V_{\vec{n}}(r) = 2(-1)^{r+1}/r^2$, for $r = 1, 2$.

As with the cube, the double nodes and element orientation were generated automatically by the main program which calls the LaPlace subroutine. The elements were generated starting with eight equally spaced nodes along each edge of the boundary. The results of the calculation at several representative points (indicated in Figure 4) are shown in Table 2. As might be expected, the answers are less accurate than for the cube; the approximation of a spherical surface by planar triangles introduces a geometrical approximation not present in the previous example. The calculated potential on the plane of symmetry (point 9) is reasonably accurate, as are the current densities away from the edges of the region (points 7 and 8). However, note that the current density solutions at the double nodes (along the intersection of a potential and insulated surface, points 1-6) are significantly worse than the other results, with the sharp corners (points 1 and 2) exceptionally bad. Aside from the geometry, the other basic approximation in the calculation is the linear interpolation, Eq. (2.7), employed in the evaluation of the integrals given in Eq. (2.14). This functional form does not take into account all of the known information about the behavior of the current density at this edge: the function must achieve either a maximum or a minimum when crossing an insulated surface. This explains why the corners, where there is clearly more constraint on the function, are the least accurate points. If this behavior is incorporated into the approximation by using a higher order polynomial interpolation, the results can be significantly improved with little additional computation. The details of the method will be reported elsewhere.

IV. PROGRAM LISTING

```
C*********************************************************************
C*THIS SUBROUTINE SOLVES LAPLACE'S EQUATION IN 3 DIMENSIONS USING
C*A BOUNDARY INTEGRAL METHOD. EITHER THE POTENTIAL OR THE NORMAL
C*DERIVATIVE OF THE POTENTIAL MUST BE KNOWN FOR EACH NODE ON THE
C*BOUNDARY, AND THIS ROUTINE SOLVES FOR THE UNKNOWN VALUES ON THE
C*BOUNDARY. IF THE POTENTIAL IS UNKNOWN, THE NORMAL DERIVATIVE IS
C*ASSUMED TO BE ZERO.
C*********************************************************************
      SUBROUTINE D3LAPL(KEL,MNODES,XYZ,NODES,NODDEF,BV,B,IPOLE)
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION XYZ(601,3),NODES(700,3),NODDEF(500),BV(500),P(500)
      DIMENSION H(400,400),ZL(400,400),IP(4),XNORM(3),B(400,1)
      DIMENSION IIP(3)
```

```
      DIMENSION XY(3,3)
      COMMON XH, YH, ZH, BETASQ, D14, BETA
      EXTERNAL D01BAZ,F1,F2,F3,G1,G2,G3
      IFAIL = 0
      DO 3 I=1,MNODES
    3 B(I,1) = 0.D0
      IF(IPOLE.NE.0) GO TO 121
      NPOINT = 12
      EPS = X02AAF(XXX)
      NDIM = 400
      IR = 1
      PI = X01AAF(PQR)
      FPI = 4.D0 * PI
      IP(4) = MNODES + 1
      DO 2 I=1,MNODES
      DO 2 II=1,MNODES
      H(I,II) = 0.D0
    2 ZL(I,II) = 0.D0
C*********************************************************************
C*DIMENSION 500 IS THE MAXIMUM NUMBER OF NODES, 600 THE MAXIMUM
C*NUMBER OF ELEMENTS. AN EXTRA ROW IS INCLUDED IN XYZ.
C*KEL = NUMBER OF ELEMENTS
C*MNODES = NUMBER OF NODES
C*XYZ = COORDINATES OF THE NODES
C*NODES = NODE NUMBERS, IN PROPER SEQUENCE FOR COMPUTING OUTWARD
C*       NORMAL, FORMING KTH TRIANGULAR ELEMENT
C*NODDEF = 0 IF POTENTIAL IS SPECIFIED
C*        1 IF NORMAL DERVIATIVE IS ZERO
C*BV = ARRAY CONTAINING KNOWN BOUNDARY VALUES
C*B = ARRAY CONTAINING SOLVED FOR BOUNDARY VALUES
C*IPOLE = 0 INDICATES NEW GEOMETRY..ENTIRE ALGORITHM TO BE EXECUTED
C*      = 1 INDICATES POLARIZATION LOOP..CHANGE RIGHT HAND SIDE
C*NPOINT IS THE NUMBER OF POINTS IN THE GAUSSIAN QUADRATURE
C*********************************************************************
C*CONTRUCT H AND L MATRICES
C*K LOOP INTEGRATES OVER EACH ELEMENT
C*********************************************************************
      DO 101 K=1,KEL
      DO 4 I=1,3
    4 IP(I) = NODES(K,I)
      SIGN = 1.D0
C*********************************************************************
C*MAKE 1 TO 3 LARGER THAN 1 TO 2
C*********************************************************************
      D13 = DIST(XYZ,IP(1),IP(3))
      D12 = DIST(XYZ,IP(1),IP(2))
      IF(D13.GE.D12) GO TO 20
      TEMP = D12
      D12 = D13
      D13 = TEMP
      IT = IP(2)
      IP(2) = IP(3)
```

```
      IP(3) = IT
      SIGN = -SIGN
C*********************************************************************
C*TRANSLATE POINT 1 TO THE ORIGIN, MOVING POINTS 2,3
C*              TO BE RESTORED LATER
C*********************************************************************
   20 DO 21 I=2,3
      DO 21 II=1,3
   21 XY(I,II) = XYZ(IP(I),II) - XYZ(IP(1),II)
C*********************************************************************
C*FAC IS LAMBDA
C*D14 IS SMALL H
C*********************************************************************
      SUM = 0.D0
      DO 25 I=1,3
   25 SUM = SUM + (XY(2,I))*(XY(3,I)-XY(2,I))
      D23 = DIST(XYZ,IP(2),IP(3))
      FAC = -SUM/(D23*D23)
      SUM = 0.D0
      DO 26 I=1,3
      XYZ(IP(4),I) = XY(2,I)+FAC*(XY(3,I)-XY(2,I))
   26 SUM = SUM + XYZ(IP(4),I)*XYZ(IP(4),I)
      D14 = DSQRT(SUM)
      C2 = D14/D12
      C3 = D14/D13
      THETA2 = DACOS(C2)
      THETA3 = DACOS(C3)
      IF(FAC.GT.0.D0) THETA2 = -THETA2
      S3 = DSIN(THETA3)
      S2 = DSIN(THETA2)
      D = D12*D13*(C2*S3 - S2*C3)
      D = 1.D0/D
C*********************************************************************
C*CALCULATE UNIT NORMAL XNORM
C*********************************************************************
      XNORM(1)=XY(2,2)*XY(3,3)-XY(3,2)*XY(2,3)
      XNORM(2)=XY(2,3)*XY(3,1)-XY(3,3)*XY(2,1)
      XNORM(3)=XY(2,1)*XY(3,2)-XY(3,1)*XY(2,2)
      DO 30 I=1,3
   30 XNORM(I) = SIGN*XNORM(I)*D
C*********************************************************************
C*M LOOP MOVES SINGULARITY AROUND THE BOUNDARY
C*********************************************************************
      DO 100 M=1,MNODES
C*********************************************************************
C*IP(I) IS THE NODE NUMBER OF POINT I. IF THE SINGULARITY IS AMONG
C*THE ELEMENT NODES, THIS MUST BE POINT 1.
C*CHECK FOR MULTIPLE NODES
C*********************************************************************
      KSING = 0
      DO 15 I=1,3
      J = I
```

```
      DT = DIST(XYZ,IP(J),M)
      IF(DT.GT.1.D-12) GO TO 15
      KSING = I
      GO TO 16
   15 CONTINUE
      GO TO 10
C*******************************************************************
C*IF SINGULARITY ON THIS ELEMENT AND NORMAL DERIVATIVE IS SPECIFIED
C*THEN BOTH INTEGRALS ARE ZERO AND THERE IS NOTHING TO COMPUTE
C*******************************************************************
   16 IF(NODDEF(IP(2)).NE.0) GO TO 100
C*******************************************************************
C*I
C*COMPUTE ZL MATRIX ELEMENTS
C*IF KSING NOT EQUAL TO ONE, CHANGE GEOMETRY
C*******************************************************************
      DO 74 I=1,3
   74 IIP(I) = IP(I)
      IF(KSING.NE.1) GO TO 62
      SD12 = D12
      SD13 = D13
      SD14 = D14
      SD = D
      SS2 = S2
      SS3 = S3
      SC2 = C2
      SC3 = C3
      GO TO 50
   62 IIP(1) = IP(KSING)
      IIP(KSING) = IP(1)
C*******************************************************************
C*MAKE 1 TO 3 LARGER THAN 1 TO 2
C*******************************************************************
      SD13 = DIST(XYZ,IIP(1),IIP(3))
      SD12 = DIST(XYZ,IIP(1),IIP(2))
      IF(SD13.GE.SD12) GO TO 201
      TEMP = SD12
      SD12 = SD13
      SD13 = TEMP
      IT = IIP(2)
      IIP(2) = IIP(3)
      IIP(3) = IT
C*******************************************************************
C*TRANSLATE POINT 1 TO THE ORIGIN, MOVING POINTS 2,3
C*            TO BE RESTORED LATER
C*******************************************************************
  201 DO 211 I=2,3
      DO 211 II=1,3
  211 XY(I,II) = XYZ(IIP(I),II) - XYZ(IIP(1),II)
C*******************************************************************
C*FAC IS LAMBDA
C*D14 IS SMALL H
```

```
C*****************************************************************
      SUM = 0.D0
      DO 251 I=1,3
  251 SUM = SUM + (XY(2,I))*(XY(3,I)-XY(2,I))
      SD23 = DIST(XYZ,IIP(2),IIP(3))
      FAC = -SUM/(SD23*SD23)
      SUM = 0.D0
      DO 262 I=1,3
      Q = XY(2,I)+FAC*(XY(3,I)-XY(2,I))
  262 SUM = SUM + Q*Q
      SD14 = DSQRT(SUM)
      SC2 = SD14/SD12
      SC3 = SD14/SD13
      ETA2 = DACOS(SC2)
      ETA3 = DACOS(SC3)
      IF(FAC.GT.0.D0) ETA2 = -ETA2
      SS3 = DSIN(ETA3)
      SS2 = DSIN(ETA2)
      SD = SD12*SD13*(SC2*SS3 - SS2*SC3)
      SD = 1.D0/SD
C*****************************************************************
C*ARRIVAL HERE INDICATES SINGULARITY IS POINT 1 AND POTENTIAL IS
C*       SPECIFIED ON THIS ELEMENT, ERGO:
C*              INTEGRATE EXACTLY
C*****************************************************************
   50 XI3 = SD14*DLOG((SS3*SC2+SC2)/(SC3+SC3*SS2))/FPI
      XI1 = SD14*XI3/2.D0
      XI2 = SD14*SD14*((1.D0/SC3)-(1.D0/SC2))/(2.D0*FPI)
      ZL(M,IIP(1))=ZL(M,IIP(1))+XI3 + SD*((SD12*SS2-SD13*SS3)*XI1
     &                  + (SD13*SC3-SD12*SC2)*XI2)
      ZL(M,IIP(2)) = ZL(M,IIP(2)) + SD*SD13*(SS3*XI1 - SC3*XI2)
      ZL(M,IIP(3)) = ZL(M,IIP(3)) + SD*SD12*(SC2*XI2 - SS2*XI1)
      GO TO 100
C*****************************************************************
C*SINGULARITY NOT ON THIS ELEMENT
C*MOVE SINGULAR POINT
C*****************************************************************
   10 DO 41 I=1,3
   41 XYZ(M,I) = XYZ(M,I) - XYZ(IP(1),I)
C*****************************************************************
C*CALCULATE COORDINATES OF SINGULAR POINT IN LOCAL COORDINATE SYSTEM
C*****************************************************************
      XH = (XYZ(IP(4),1)*XYZ(M,1) + XYZ(IP(4),2)*XYZ(M,2) +
     &     XYZ(IP(4),3)*XYZ(M,3))/D14
      YH = ((XYZ(IP(3),1)-XYZ(IP(2),1))*XYZ(M,1) +
     &     (XYZ(IP(3),2)-XYZ(IP(2),2))*XYZ(M,2) +
     &     (XYZ(IP(3),3)-XYZ(IP(2),3))*XYZ(M,3))/D23
      ZH = XNORM(1)*XYZ(M,1) + XNORM(2)*XYZ(M,2) +
     &    XNORM(3)*XYZ(M,3)
      DO 22 I=1,3
   22 XYZ(M,I) = XYZ(M,I) + XYZ(IP(1),I)
      BETASQ = XH*XH + YH*YH + ZH*ZH
```

```
      BETA = DSQRT(BETASQ)
C*********************************************************************
C*IF ZH IS ZERO, SINGULAR POINT ON THE PLANE AND SO GRAD(G)*N = 0
C*********************************************************************
      IF(DABS(ZH).LT.1.D-15) GO TO 777
      AZH = 1.D0/FPI
      F3INT = D01BAF(D01BAZ,THETA2,THETA3,NPOINT,F3,IFAIL)
      F1INT = D01BAF(D01BAZ,THETA2,THETA3,NPOINT,F1,IFAIL)
      F2INT = D01BAF(D01BAZ,THETA2,THETA3,NPOINT,F2,IFAIL)
      XI1 =  AZH*F1INT
      XI2 =  AZH*F2INT
      XI3 = -AZH*F3INT
      H(M,IP(1)) = H(M,IP(1)) + XI3 + D*((D12*S2-D13*S3)*XI1
     &                    + (D13*C3-D12*C2)*XI2)
      H(M,IP(2)) = H(M,IP(2)) + D*D13*(S3*XI1 - C3*XI2)
      H(M,IP(3)) = H(M,IP(3)) + D*D12*(C2*XI2 - S2*XI1)
C*********************************************************************
C*CALCULATE ZL MATRIX
C*ARRIVAL HERE INDICATES SINGULARITY NOT ON THIS ELEMENT:
C*            INTEGRATE NUMERICALLY
C*IF NORMAL DERIVATIVE KNOWN, INTEGRAL IS 0
C*********************************************************************
  777 IF(NODDEF(IP(1)).NE.0) GO TO 100
      G1INT = D01BAF(D01BAZ,THETA2,THETA3,NPOINT,G1,IFAIL)
      G2INT = D01BAF(D01BAZ,THETA2,THETA3,NPOINT,G2,IFAIL)
      G3INT = D01BAF(D01BAZ,THETA2,THETA3,NPOINT,G3,IFAIL)
      XI1 = G1INT/FPI
      XI2 = G2INT/FPI
      XI3 = G3INT/FPI
      ZL(M,IP(1)) = ZL(M,IP(1)) + XI3 + D*((D12*S2-D13*S3)*XI1
     &                    + (D13*C3-D12*C2)*XI2)
      ZL(M,IP(2)) = ZL(M,IP(2)) + D*D13*(S3*XI1 - C3*XI2)
      ZL(M,IP(3)) = ZL(M,IP(3)) + D*D12*(C2*XI2 - S2*XI1)
  100 CONTINUE
  101 CONTINUE
C*********************************************************************
C*CALCULATE DIAGONAL MATRIX ELEMENTS OF H
C*REARRANGE EQUATIONS IN FORM AX = B
C*OVERWRITE ZL WITH MATRIX A
C*********************************************************************
      DO 110 M=1,MNODES
      SUM = 0.D0
      DO 111 I=1,MNODES
  111 SUM = SUM + H(M,I)
  110 H(M,M) = -SUM
      WRITE(6,7080) (I,H(I,I), I=1,MNODES)
 7080 FORMAT(1X, 3(I4,2X,F13.6))
C*********************************************************************
C*********************************************************************
C*********************************************************************
      DO 120 M=1,MNODES
      IF(NODDEF(M).EQ.0) GO TO 120
```

```
      DO 122 I=1,MNODES
  122 ZL(I,M) = -H(I,M)
  120 CONTINUE
C*********************************************************************
C*ALTHOUGH THE FOLLOWING LOOP COULD BE INCLUDED IN THE ABOVE LOOP,
C*IT IS SPLIT OFF FOR PURPOSE OF RECALCULATING NEW RIGHT HAND SIDE
C*ORDINARILY EXECUTE 127 ONLY WHEN NODDEF(M) = 0 (I.E. BOUNDARY
C*VALUE IS SPECIFIED) BUT SINCE WE HAVE ASSUMED NORMAL DERIVATIVE
C*IS ZERO WHEN IT IS GIVEN, BV(M)=0 ANYWAY
C*********************************************************************
  121 DO 130 M=1,MNODES
      IF(NODDEF(M).NE.0) GO TO 130
      DO 127 I=1,MNODES
  127 B(I,1) = B(I,1) + H(I,M)*BV(M)
  130 CONTINUE
C*********************************************************************
C*SOLVE SYSTEM OF EQUATIONS (ZL)X = B
C*IF SOLVING FOR POLARIZATION, USE OLD FACTORIZATION
C*********************************************************************
      IF(IPOLE.NE.0) GO TO 140
      CALL F03AFF(MNODES,EPS,ZL,NDIM,D1,IID,P,IFAIL)
  140 CALL F04AJF(MNODES,IR,ZL,NDIM,P,B,NDIM)
      WRITE(6,7080) (I,B(I,1), I=1,MNODES)
      RETURN
      END
      FUNCTION DIST(XYZ,M1,M2)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION XYZ(601,3)
      SUM = 0.D0
      DO 10 I=1,3
      Q = XYZ(M1,I)-XYZ(M2,I)
   10 SUM = SUM + Q*Q
      DIST = DSQRT(SUM)
      RETURN
      END
      FUNCTION F1(THETA)
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON XH, YH, ZH, BETASQ, D14, BETA
      F = FF(THETA)
      F1 = F*DCOS(THETA)
      RETURN
      END
      FUNCTION F2(THETA)
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON XH, YH, ZH, BETASQ, D14, BETA
      F = FF(THETA)
      F2 = F*DSIN(THETA)
      RETURN
      END
      FUNCTION F3(THETA)
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON XH, YH, ZH, BETASQ, D14, BETA
```

```
      SS = DSIN(THETA)
      CC = DCOS(THETA)
      ALPHA = XH*CC + YH*SS
      QQ = XH*SS - YH*CC
      BA = ZH*ZH + QQ*QQ
      SEC = 1.D0/CC
      R = D14*SEC
      X = (R-ALPHA)*(R-ALPHA) + BA
      X = DSQRT(X)
      F3 = ((BETASQ-R*ALPHA)*(ZH/X) - (ZH*BETA))/BA
      RETURN
      END
      FUNCTION FF(THETA)
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON XH, YH, ZH, BETASQ, D14, BETA
      SS = DSIN(THETA)
      CC = DCOS(THETA)
      ALPHA = XH*CC + YH*SS
      QQ = XH*SS - YH*CC
      BA = ZH*ZH + QQ*QQ
      SEC = 1.D0/CC
      R = D14*SEC
      X = (R-ALPHA)*(R-ALPHA) + BA
      X = DSQRT(X)
      FF = (((2.D0*ALPHA*ALPHA-BETASQ)*R - BETASQ*ALPHA)*(ZH/X)
     &          + ZH*BETA*ALPHA)/BA
      FF = FF + ZH*DLOG((X+R-ALPHA)*(BETA+ALPHA)/BA)
      RETURN
      END
      FUNCTION G1(THETA)
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON XH, YH, ZH, BETASQ, D14, BETA
      G = GG(THETA)
      G1 = G*DCOS(THETA)
      RETURN
      END
      FUNCTION G2(THETA)
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON XH, YH, ZH, BETASQ, D14, BETA
      G = GG(THETA)
      G2 = G*DSIN(THETA)
      RETURN
      END
      FUNCTION G3(THETA)
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON XH, YH, ZH, BETASQ, D14, BETA
      CC = DCOS(THETA)
      SS = DSIN(THETA)
      ALPHA = XH*CC + YH*SS
      QQ = XH*SS - YH*CC
      BA = ZH*ZH + QQ*QQ
      SEC = 1.D0/CC
```

```
      R = D14*SEC
      X = (R-ALPHA)*(R-ALPHA) + BA
      X = DSQRT(X)
      G3 = X - BETA + ALPHA*DLOG((X+R-ALPHA)*(BETA+ALPHA)/BA)
      RETURN
      END
      FUNCTION GG(THETA)
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON XH, YH, ZH, BETASQ, D14, BETA
      CC = DCOS(THETA)
      SS = DSIN(THETA)
      ALPHA = XH*CC + YH*SS
      QQ = XH*SS - YH*CC
      BA = ZH*ZH + QQ*QQ
      SEC = 1.D0/CC
      R = D14*SEC
      X = (R-ALPHA)*(R-ALPHA) + BA
      X = DSQRT(X)
      GG = (R+3.D0*ALPHA)*X - 3.D0*ALPHA*BETA
      GG = GG + (3.D0*ALPHA*ALPHA-BETASQ)*
     &        DLOG((X+R-ALPHA)*(BETA+ALPHA)/BA)
      GG = GG/2.D0
      RETURN
      END
```

Acknowledgements: It is a pleasure to thank G. E. Giles, J. S. Bullock, and R. E. Funderlic for useful discussions and G. E. Giles for generating the discretizations for the test problems. This work was sponsored by the Y-12 Development Division.

REFERENCES

[1] P. L. Betts and J. S. Kang, "On the Accuracy of the Boundary Element Method for Three Dimensional Conduction Problems", in *Numerical Methods in Heat Transfer, V. II*, eds. R. W. Lewis, K. Morgan, and B. A. Schrefler, Wiley, New York 1983.

[2] C. A. Brebbia, *The Boundary Element Method* for *Engineers*, Pentech Press London 1978.

[3] J. S. Bullock, G. E. Giles, and L. J. Gray, "Simulation of an Electrochemical Plating Process", in *Research in Boundary Element Methods*, ed. C. A. Brebbia, in press.

[4] PAFEC Ltd., Strelley Hall, Strelley Nottingham NG8 6PE UK

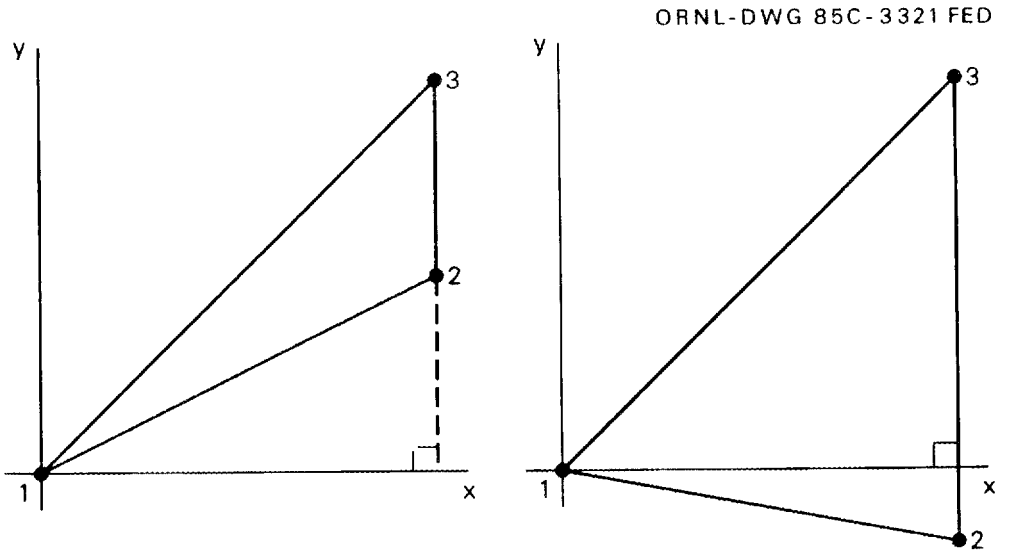

Figure 1. The arrangement of node numbers on an element used for computing the surface integrals.

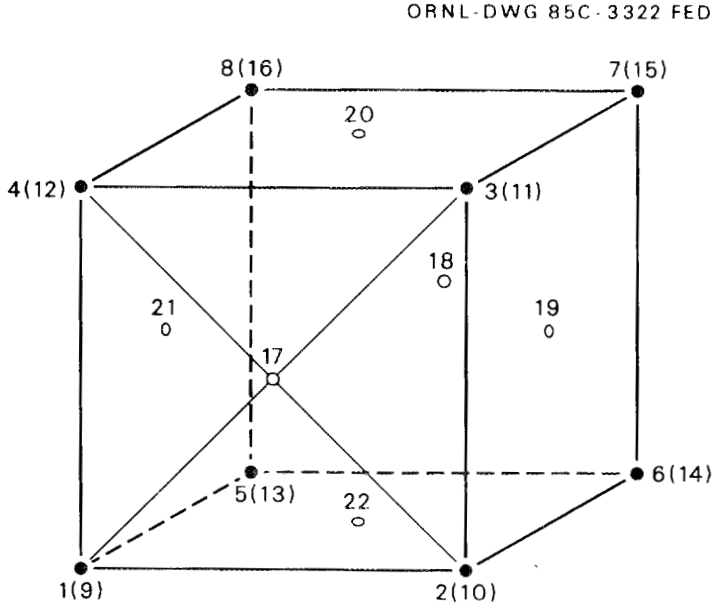

Figure 2. The coarse discretization of the unit cube surface.

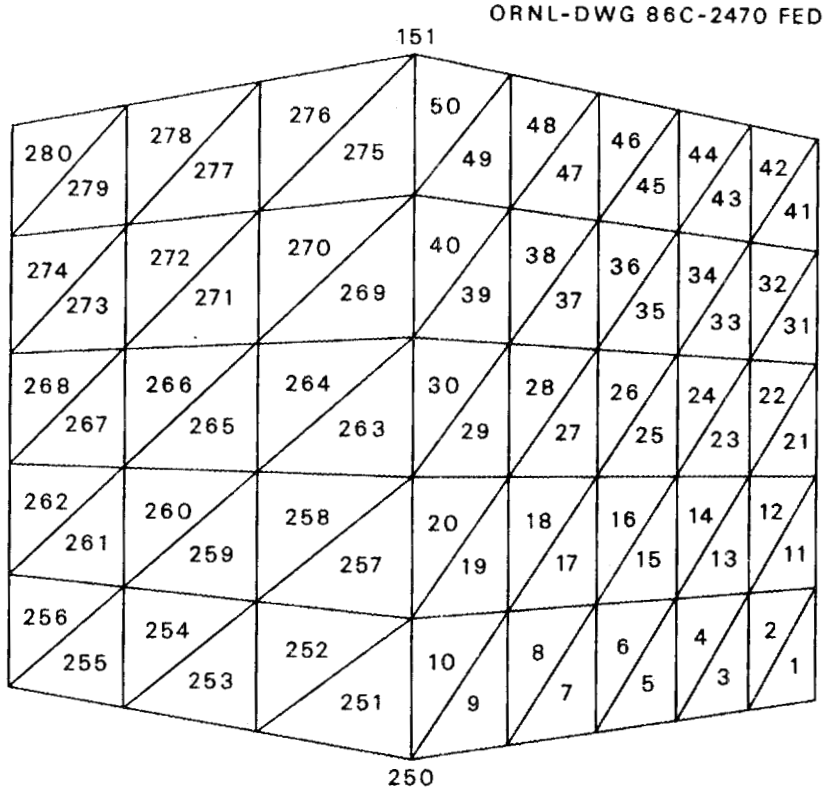

Figure 3. The fine discretization of the unit cube surface.

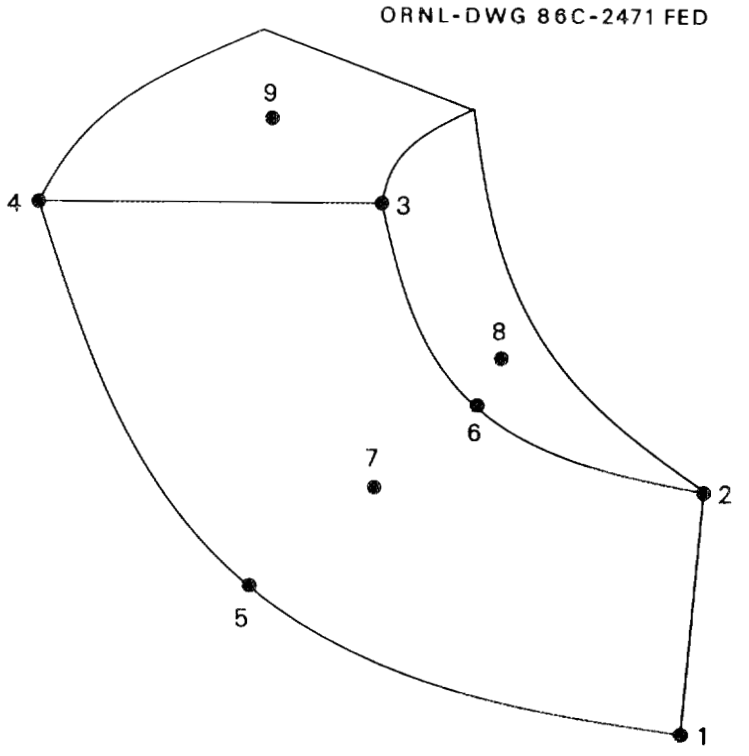

Figure 4. The concentric sphere wedge employed in the test calculation. The values obtained at the numbered nodes are reported in Table 3.

TABLE 1: INPUT DATA AND RESULTS FOR THE INSULATED CUBE

NODE	NODDEF	X	Y	Z	INPUT BOUNDARY VALUE	COMPUTED BOUNDARY VALUE
1	0	0.0	0.0	0.0	1.0	1.00
2	0	1.0	0.0	0.0	1.0	1.00
3	0	1.0	0.0	1.0	1.0	1.00
4	0	0.0	0.0	1.0	1.0	1.00
5	0	0.0	1.0	0.0	0.0	-1.00
6	0	1.0	1.0	0.0	0.0	-1.00
7	0	1.0	1.0	1.0	0.0	-1.00
8	0	0.0	1.0	1.0	0.0	-1.00
9	1	0.0	0.0	0.0	0.0	1.00
10	1	1.0	0.0	0.0	0.0	1.00
11	1	1.0	0.0	1.0	0.0	1.00
12	1	0.0	0.0	1.0	0.0	1.00
13	1	0.0	1.0	0.0	0.0	0.00
14	1	1.0	1.0	0.0	0.0	0.00
15	1	1.0	1.0	1.0	0.0	0.00
16	1	0.0	1.0	1.0	0.0	0.00
17	0	0.5	0.0	0.5	1.0	1.00
18	0	0.5	1.0	0.5	0.0	-1.00
19	1	1.0	0.5	0.5	0.0	0.50
20	1	0.5	0.5	1.0	0.0	0.50
21	1	0.0	0.5	0.5	0.0	0.50
22	1	0.5	0.5	0.0	0.0	0.50

ELEMENT DEFINITIONS

1 2 17	2 3 17	17 3 4	17 4 1	10 14 19	14 15 19
19 15 11	19 11 10	6 5 18	6 18 7	18 8 7	18 5 8
21 16 13	21 13 9	21 9 12	21 12 16	12 11 20	11 15 20
20 15 16	20 16 12	10 9 22	9 13 22	13 14 22	14 10 22

TABLE 2: RESULTS FOR THE CONCENTRIC SPHERES

POINT*	COMPUTED RESULT	EXACT RESULT
1	-0.552	-0.500
2	1.878	2.000
3	1.924	2.000
4	-0.536	-0.500
5	-0.511	-0.500
6	1.985	2.000
7	0.228	0.231
8	2.013	2.000
9	0.227	0.231

* Point numbers refer to Figure 4.

APPENDIX A

Obtaining the elements of the matrix H requires evaluating the integrals in Eq. (2.11). Integration of the r variable can be done in closed form, leaving just the θ integration to be performed numerically. Setting $h = \| P_3 - P_1 \| \cos\theta_3$ (see fig. 1), these integrals are

$$\int_0^{h\sec\theta} r\rho^{-3}dr \quad \text{and} \quad \int_0^{h\sec\theta} r^2\rho^{-3}dr \tag{A.1}$$

where, from Eq. (2.10),

$$\rho = [r^2 - 2\alpha r + \beta^2]^{1/2} \tag{A.2}$$

and

$$\begin{aligned} \alpha &= \hat{x}\cos\theta + \hat{y}\sin\theta \\ \beta^2 &= \hat{x}^2 + \hat{y}^2 + \hat{z}^2. \end{aligned} \tag{A.3}$$

From integral tables,

$$\int \frac{rdr}{(r^2 - 2\alpha r + \beta^2)^{3/2}} = -\frac{(\beta^2 - 2\alpha r)}{2(\beta^2 - \alpha^2)(r^2 - 2\alpha r + \beta^2)^{1/2}} \tag{A.4}$$

and

$$\begin{aligned} \int \frac{r^2dr}{(r^2 - 2\alpha r + \beta^2)^{3/2}} &= \frac{(2\alpha^2 - \beta^2)r - \beta^2\alpha}{(\beta^2 - \alpha^2)(r^2 - 2\alpha r + \beta^2)^{1/2}} \\ &\quad + \log\left[(r^2 - 2\alpha r + \beta^2)^{1/2} + r - \alpha\right] \end{aligned} \tag{A.5}$$

We therefore have

$$\int_0^{h\sec\theta} r\rho^{-3}dr = \frac{-1}{\beta^2 - \alpha^2}\left[\frac{-\alpha h\sec\theta + \beta^2}{(h^2\sec^2\theta - 2\alpha h\sec\theta + \beta^2)^{1/2}} - \beta\right] \tag{A.6}$$

$$\begin{aligned} \int_0^{h\sec\theta} r^2\rho^{-3}dr &= \frac{(2\alpha^2 - \beta^2)h\sec\theta - \beta^2\alpha}{(\beta^2 - \alpha^2)(h^2\sec^2\theta - 2\alpha h\sec\theta + \beta^2)^{1/2}} \\ &\quad + \log\left[(h^2\sec^2\theta - 2\alpha h\sec\theta + \beta^2)^{1/2} + h\sec\theta - \alpha\right] + \frac{\beta\alpha}{\beta^2 - \alpha^2} - \log(\beta - \alpha). \end{aligned}$$

The corresponding integrals required for evaluating L, when the singularity is not on the element, are given by

$$\int_0^{h\sec\theta} r\rho^{-1}dr \quad \text{and} \quad \int_0^{h\sec\theta} r^2\rho^{-1}dr \tag{A.7}$$

and can also be computed directly. Thus,

$$\int_0^{h\sec\theta} r\rho^{-1}dr = (h^2\sec^2\theta - 2\alpha h\sec\theta + \beta^2)^{1/2} - \beta + \tag{A.8}$$

$$\alpha \log\left[\frac{(h^2\sec^2\theta - 2\alpha h\sec\theta + \beta^2)^{1/2} + h\sec\theta - \alpha}{\beta - \alpha}\right].$$

and

$$\int_0^{h\sec\theta} r^2\rho^{-1}dr = \tfrac{1}{2}\left[(h\sec\theta + 3\alpha)(h^2\sec^2\theta - 2\alpha h\sec\theta + \beta^2)^{1/2} - 3\alpha\beta\right] + \tag{A.9}$$

$$\tfrac{1}{2}(3\alpha^2 - \beta^2)\log\left[\frac{(h^2\sec^2\theta - 2\alpha h\sec\theta + \beta^2)^{1/2} + h\sec\theta - \alpha}{\beta - \alpha}\right].$$

When the singularity is on the element, things are considerably simplified and the complete surface integral over the triangle can easily be accomplished. These integrals are:

$$\int_{\theta_2}^{\theta_3} d\theta \int_0^{h\sec\theta} r\cos\theta\, dr = \frac{h^2}{2}\log\left[\frac{\sec\theta_3 + \tan\theta_3}{\sec\theta_2 + \tan\theta_2}\right] \tag{A.10}$$

$$\int_{\theta_2}^{\theta_3} d\theta \int_0^{h\sec\theta} r\sin\theta\, dr = \frac{h^2}{2}\left[\sec\theta_3 - \sec\theta_2\right]$$

$$\int_{\theta_2}^{\theta_3} d\theta \int_0^{h\sec\theta} dr = h\log\left[\frac{\sec\theta_3 + \tan\theta_3}{\sec\theta_2 + \tan\theta_2}\right]$$

APPENDIX B

INPUT VARIABLES FOR D3LAPL

NAME	TYPE	DESCRIPTION
KEL	INTEGER	The number of triangular elements
MNODES	INTEGER	The number of nodes
XYZ	REAL ARRAY	XYZ(I,J) is the Jth coordinate of node I
NODES	INTEGER ARRAY	NODES(I,J) is the Jth node of element I
NODDEF	INTEGER ARRAY	NODDEF=0 for a potential node NODDEF=1 for an insulated node
BV	REAL ARRAY	BV(I) is the boundary value at node I
B	REAL ARRAY	B(I), on exit, is the computed boundary value at node I
IPOLE	INTEGER	IPOLE=0 indicates new geometry IPOLE=1 indicates same geometry, new boundary values

APPENDIX C

The following is a listing of the main program used to call D3LAPL for the concentric spheres problem discussed in section III. It was found from experience that the data obtained from the mesh generating software PIGS often left in nodes that were not used by any element, and that the nodes were not numbered consecutively. The first part of the program corrects the data if either of these problems is present. The next step is to identify the double nodes and generate the appropriate boundary condition (BV) and node identification (NODDEF) for every node. The last task before calling D3LAPL is to align the normal of each element correctly by adjusting the order in NODES. For the concentric spheres, the center of the spheres was chosen as a convenient point to align the normal on the spherical segments [PT(1,*) and PT(3,*)], whereas a point inside the region [PT(2,*)] is employed to check the normal on the insulated surfaces.

```
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION XYZ(601,3),NODES(700,3),NODDEF(500),BV(500),B(400,1)
      DIMENSION IPROP(700),XNORM(3),Q(3,3)
      DIMENSION PT(3,3)
      DIMENSION NORD(1000), JORD(600)
      READ(5,70) KEL,MNODES
   70 FORMAT(2I3)
C*NORD IS USED TO RENUMBER THE NODES FROM 1 TO MNODES*****************
      DO 60 I=1,MNODES
      READ(5,14) II, (XYZ(I,J), J=1,3)
   14 FORMAT(10X,I3,3E13.5)
   60 NORD(II) = I
      DO 20 I=1,KEL
      READ(5,13) IPROP(I),NODES(I,1),NODES(I,2),NODES(I,3)
   13 FORMAT(50X,I2,10X,I3,10X,I3/10X,I3)
      DO 21 J=1,3
   21 NODES(I,J) = NORD(NODES(I,J))
   20 CONTINUE
      IPOLE = 0
C*REMOVE UNUSED NODES AND RENUMBER*************************************
      KNODES = 0
      DO 80 M=1,MNODES
      DO 81 I=1,KEL
      K = (NODES(I,1)-M)*(NODES(I,2)-M)*(NODES(I,3)-M)
      IF(K.EQ.0) GO TO 82
   81 CONTINUE
      GO TO 80
   82 KNODES = KNODES + 1
      NORD(M) = KNODES
      JORD(KNODES) = M
   80 CONTINUE
      MNODES = KNODES
      DO 83 M=1,MNODES
      DO 83 J=1,3
   83 XYZ(M,J) = XYZ(JORD(M),J)
      DO 84 I=1,KEL
```

```
      DO 84 J=1,3
   84 NODES(I,J) = NORD(NODES(I,J))
C*INITIALIZE BV(I) AND NODDEF(I)****************************************
      DO 400 I=1,MNODES
      BV(I) = 0.D0
      NODDEF(I) = 0
  400 CONTINUE
C*CHECK FOR DOUBLE NODES AND ASSIGN NODDEF AND BOUNDARY VALUES*********
      MN = MNODES + 1
      DO 100 M=1,MNODES
      IFLAG = 0
      DO 200 I=1,KEL
      J = I
      K = (NODES(I,1)-M)*(NODES(I,2)-M)*(NODES(I,3)-M)
      IF(K.NE.0) GO TO 200
      JPROP = IPROP(I)
      IF(JPROP.EQ.1) BV(M) = 1.D0
      IF(JPROP.EQ.3) NODDEF(M) = 1
      IF(JPROP.EQ.3) GO TO 250
      IF(DABS(XYZ(M,3)).GT.1.D-3) GO TO 987
      IF(DABS(XYZ(M,2)-XYZ(M,1)).LT.1.D-6) GO TO 250
      IF(JPROP.EQ.1) XYZ(M,1)=DSQRT(1.D0-XYZ(M,2)*XYZ(M,2))
      IF(JPROP.EQ.2) XYZ(M,1)=DSQRT(4.D0-XYZ(M,2)*XYZ(M,2))
      GO TO 250
  987 IF(JPROP.EQ.1) XYZ(M,3)=DSQRT(1.D0-XYZ(M,1)*XYZ(M,1)-
     &                           XYZ(M,2)*XYZ(M,2))
      IF(JPROP.EQ.2) XYZ(M,3)=DSQRT(4.D0-XYZ(M,1)*XYZ(M,1)-
     &                           XYZ(M,2)*XYZ(M,2))
      GO TO 250
  200 CONTINUE
      WRITE(6,801) M
  801 FORMAT(1X, 'NODE',I4,' DOES NOT BELONG TO ANY TRIANGLE')
      GO TO 100
  250 J1 = J + 1
      DO 300 I=J1,KEL
      K = (NODES(I,1)-M)*(NODES(I,2)-M)*(NODES(I,3)-M)
      IF(K.NE.0) GO TO 300
      IF(IPROP(I).EQ.JPROP) GO TO 300
      KS = 1
      IF((NODES(I,2)-M).EQ.0) KS = 2
      IF((NODES(I,3)-M).EQ.0) KS = 3
      NODES(I,KS) = MN
      IF(IFLAG.EQ.1) GO TO 300
      WRITE(6,700) MN, M
  700 FORMAT(1X,'NODE',I4,' IS NODE',I4)
      DO 275 II=1,3
  275 XYZ(MN,II) = XYZ(M,II)
      BV(MN) = 0.D0
      IF(IPROP(I).EQ.1) BV(MN) = 1.D0
      NODDEF(MN) = 0
      IF(IPROP(I).EQ.3) NODDEF(MN) = 1
      IFLAG = 1
```

```
  300 CONTINUE
      MN = MN + IFLAG
  100 CONTINUE
      MNODES = MN - 1
C*END CHECK FOR DOUBLE NODES*****************************************
C*ORIENT NORMAL CORRECTLY
      PT(1,1) = 0.D0
      PT(1,2) = 0.D0
      PT(1,3) = 0.D0
      PT(3,1) = 0.7D0
      PT(3,2) = 0.1D0
      PT(3,3) = 1.32287D0
      PT(2,1) = 0.0D0
      PT(2,2) = 0.0D0
      PT(2,3) = 0.0D0
      DO 610 I=1,KEL
      Q(2,2) = XYZ(NODES(I,2),2)-XYZ(NODES(I,1),2)
      Q(3,3) = XYZ(NODES(I,3),3)-XYZ(NODES(I,1),3)
      Q(3,2) = XYZ(NODES(I,3),2)-XYZ(NODES(I,1),2)
      Q(2,3) = XYZ(NODES(I,2),3)-XYZ(NODES(I,1),3)
      Q(3,1) = XYZ(NODES(I,3),1)-XYZ(NODES(I,1),1)
      Q(2,1) = XYZ(NODES(I,2),1)-XYZ(NODES(I,1),1)
      XNORM(1)=Q(2,2)*Q(3,3)-Q(3,2)*Q(2,3)
      XNORM(2)=Q(2,3)*Q(3,1)-Q(3,3)*Q(2,1)
      XNORM(3)=Q(2,1)*Q(3,2)-Q(3,1)*Q(2,2)
      DIR = 0.D0
      DO 640 L=1,3
  640 DIR = DIR - (PT(IPROP(I),L) - XYZ(NODES(I,1),L))*XNORM(L)
      IF(IPROP(I).EQ.1) DIR = -DIR
      IF(DIR .GE. 0.D0) GO TO 610
      IT = NODES(I,2)
      NODES(I,2) = NODES(I,3)
      NODES(I,3) = IT
  610 CONTINUE
C*END NORMAL CHECK*****************************************************
C*OUTPUT: ****************************************************
      WRITE(6,51) (I, (XYZ(I,J), J=1,3), I=1,MNODES)
      WRITE(6,25) (I,IPROP(I),(NODES(I,J), J=1,3), I=1,KEL)
      WRITE(6,65) (I,NODDEF(I),BV(I), I=1,MNODES)
   25 FORMAT(10X,I3,2X,I1,7X,3I6)
   65 FORMAT(2(1X,I3,2X,I1,2X,F10.4,6X))
   51 FORMAT(10X,I3,3E13.5)
      CALL D3LAPL(KEL,MNODES,XYZ,NODES,NODDEF,BV,B,IPOLE)
C     WRITE(6,67) (I,B(I,1), I=1,MNODES)
C  67 FORMAT(3(1X,I3,2X,F13.6,5X))
      STOP
      END
```

INTERNAL DISTRIBUTION

1. V. Alexiades
2. J. S. Bullock
3. C. W. Coates
4. M. V. Denson
5. J. B. Drake
6. G. E. Giles
7-11. L. J. Gray
12-13. R. F. Harbison
14. M. T. Heath
15. J. K. Ingersoll
16. F. C. Maienschein
17. T. J. Mitchell
18. M. D. Morris
19. E. G. Ng
20. A. D. Solomon
21. S. Thompson
22. V. R. R. Uppuluri
23. R. C. Ward
24. M. A. Williams
25. D. G. Wilson
26. A. Zucker
27. P. W. Dickson, Jr. (Consultant)
28. G. H. Golub (Consultant)
29. R. M. Haralick (Consultant)
30. D. Steiner (Consultant)
31. Central Research Library
32. K-25 Plant Library
33. ORNL Patent Office
34. Y-12 Technical Library /Document Reference Section
35. Laboratory Records - RC
36-37. Laboratory Records Department

EXTERNAL DISTRIBUTION

38. Office of Assistant Manager for Energy Research and Development, U.S. Department of Energy, Oak Ridge Operations Office, Oak Ridge, Tennessee 37830

39-66. Technical Information Center.