TRANSIENT HEAT CONDUCTION PROBLEMS USING B.I.E.M.

VICENTE ROURES
Civil Engineering Division, INITEC, Madrid, Spain

and

ENRIQUE ALARCON
Polytechnical University, Madrid, Spain

Abstract—A method, using boundary elements, is presented as a solution to plane transient heat conduction. The proposed method considers the governing equation to be a Helmholtz's equation and solves the problem of time variation using step by step integration. A numerical procedure is developed and its effectiveness verified. Several examples are provided and their results compared with the theoretical ones.

1. INTRODUCTION

The application of Boundary Integral Equation Method (B.I.E.M.) to problems of transient heat conduction started in 1969 with a paper by Rizzo and Shippy[1] in which the properties of Laplace transform were applied; later, in 1970, Shaw[2] and, in 1972, Chang *et al.*[3] solved the problem by using a time-dependent solution. In 1977 Combescure and Lachat[4] repeated the use of integral transforms and in 1979 Wrobel and Brebbia[5] used a time-dependent fundamental solution and a weighted residual formulation.

This paper presents the formulation for two-dimensional isotropic continuous solid using the Helmholtz's equation as well as the Green's theorem, with step by step computation in the time domain.

In the context of boundary technique, the idea was proposed by Brebbia and Walker[8] but they did not present any result.

As there appears a domain term in the formulation it is necessary to evaluate some integrals inside the volume by establishing an internal cell subdivision. However only known quantitites are involved in these integrals with the unknown values limited at the boundary, and the cell system does not increment the size of the system of equations to be solved.

A procedure to eliminate the domain integration in connection with the method described in (5) has been recently published[9].

2. FORMULATION

The equation that governs the transient heat conduction phenomena in a domain D with boundary ∂D is

$$k\nabla^2\phi = \rho c\,\frac{\partial\phi}{\partial t}$$

or

$$K\nabla^2\phi = \frac{\partial\phi}{\partial t} \tag{1}$$

where ϕ is the temperature; t is the time; k is the thermal conductivity; ρ is the density; c is the specific heat; and K is the thermometric conductivity $K = k/\rho c$.

The boundary conditions are known in every moment

$$\phi = \bar{\phi} \quad \text{on } \partial D_1$$

$$q = \frac{\partial\phi}{\partial n} = \bar{q} \quad \text{on } \partial D_2 \tag{2}$$

and so are the initial conditions inside the domain

$$\phi = \phi_0 \quad \text{in } D \tag{3}$$

q is the heat flux; n is the normal to the boundary; and $\partial D = \partial D_1 + \partial D_2$.

The basic equation is obtained from (1) through a finite difference approach in the time domain so that the equation

$$K\nabla^2\phi = \frac{\phi\Delta_t - \phi_0}{\Delta t} \tag{4}$$

can be solved at every time step.

The boundary formulation can be obtained looking for a ψ such that

$$K\nabla^2\psi - \frac{\psi}{\Delta t} + \Delta = 0 \tag{5}$$

where Δ is the Dirac function presenting a singularity at the point where (4) is being applied.

The eqn (5) can be compared with the Helmholtz's equation in the form of a modified Bessel equation. The solution of (5), called fundamental solution, is—in two dimensions—the following.

$$\psi = \frac{1}{2\pi K}\,K_0\left[\frac{r}{\sqrt{(K\Delta t)}}\right] \tag{6}$$

$$\frac{\partial\psi}{\partial r} = \frac{-1}{2\pi K}\,\frac{1}{\sqrt{(K\Delta t)}}\,K_1\left[\frac{r}{\sqrt{K\Delta t)}}\right] \tag{7}$$

$$\frac{\partial\psi}{\partial n} = \frac{-1}{2\pi K}\,\frac{d}{r}\,\frac{1}{\sqrt{k(\Delta t)}}\,K_1\left[\frac{r}{\sqrt{(k\Delta t)}}\right] \tag{8}$$

where K_0 is the modified Bessel function of second kind and zero order; K_1 is the modified Bessel function of second kind and first order; r is the distance between two points; and d is the distance between the point under study and the observed boundary element.

Applying the 2nd Green's theorem it is possible to write.

$$\phi(J) - \int_D \frac{1}{\Delta t}\psi(H, J)\phi_0(H)\mathrm{d}V(H) + \int_{\partial D} K\phi(G)\frac{\partial \psi}{\partial n}(G, J)\mathrm{d}S(G) = \int_{\partial D} K\psi(G, J)\frac{\partial \phi}{\partial n}(G)\mathrm{d}S(G)$$

$$J, H \in D \quad G \in \partial D. \tag{9}$$

When the point J is on ∂D there appears a singularity ($r \to 0$) which can be elucidated by the usual procedure of isolation of the point under study by a circle—two dimensions—of radius $\zeta \to 0$.

If the point J is on the boundary and ∂D is the boundary of the circle, (9) can be written as

$$\phi(J) - \int_D \frac{1}{\Delta t}\psi(H, J)\phi_0(H)\mathrm{d}V(H) + \int_{\partial D - \partial D_\epsilon} K\phi(G)\frac{\partial \psi}{\partial n}(G, J)\,\mathrm{d}S(G) + \int_{\partial D_\epsilon} K\phi(G)\frac{\partial \psi}{\partial n}(G, J)\mathrm{d}S(G) = \int_{\partial D - \partial D_\epsilon} K\psi(G, J)\frac{\partial \phi}{\partial n}(G)\mathrm{d}S(G) + \int_{\partial D_\epsilon} K\psi(G, J)\frac{\partial \phi}{\partial n}(G)\mathrm{d}S(G)$$

$$H \in D \quad J, G \in \partial D \tag{10}$$

but for smooth boundaries

$$\lim_{\epsilon \to 0}\int_{\partial D_\epsilon} \phi(G)\frac{\partial \psi}{\partial n}(G,J)\,\mathrm{d}S(G) = \frac{-1}{2}\phi(J)$$

$$\lim_{\epsilon \to 0}\int_{\partial D_\epsilon} \psi(G, J)\frac{\partial \phi}{\partial n}(G)\,\mathrm{d}S(G) = 0. \tag{11}$$

If a sharp corner exists in the boundary the first integral will have a value $T^*(J)$ and putting

$$T(J) = 1 - T^*(J) \tag{12}$$

it is possible to write the fundamental equation to be discretized as below.

$$T(J)\phi(J) - \int_D \frac{1}{\Delta t}\psi(H.\ J)\phi(H)\mathrm{d}V(H) + \int_{\partial D} K(\phi)G\frac{\partial \psi}{\partial n}(G,J)\,\mathrm{d}S(G) \int_{\partial D} K\psi(G, J)\frac{\partial \phi}{\partial n}(G)\,\mathrm{d}S(G)$$

$$H \in D \quad J, G \in \partial D \tag{13}$$

(13) and (9) are the basic pair of equations that allows the solution of the problem on the boundary and inside the domain respectively.

3. DISCRETIZATION

The isoparametric formulation described elsewhere [6] is used when discretizing the boundary as well as the field variables.

That is, the geometry is interpolated by

$$X(\xi) = Ng(\xi)X_g \qquad Y(\xi) = Ng(\xi)Y_g \tag{14}$$

while the potential and flux are assumed to vary according to

$$\phi(\xi) = Ng(\xi)\phi_g \qquad q(\xi) = Ng(\xi)q_g \tag{15}$$

inside each element; where, as usual,

$$N_1(\xi) = \frac{-1}{2}(\xi - 1)$$

$$N_2(\xi) = \frac{1}{2}(\xi + 1) \tag{16}$$

and

$$\mathrm{d}S_g = \frac{1g}{2}\mathrm{d}\xi \tag{17}$$

where $1g$ is the length of the g element.

Inside the domain the integral is evaluated assuming quadrilateral cells.

$$X(\xi, \eta) = M_h(\xi, \eta)X_h \qquad Y(\xi, \eta) = M_h(\xi, \eta)Y_h \tag{18}$$

where the interpolation functions are the bilinear ones.

$$M_1(\xi, \eta) = \frac{1}{4}(1 - \xi)(1 - \eta)$$

$$M_2(\xi, \eta) = \frac{1}{4}(1 + \xi)(1 - \eta)$$

$$M_3(\xi, \eta) = \frac{1}{4}(1 + \xi)(1 + \eta)$$

$$M_4(\xi, \eta) = \frac{1}{4}(1 - \xi)(1 + \eta) \tag{19}$$

and

$$\mathrm{d}V_h = \frac{1}{8}\begin{vmatrix} X_1 & Y_1 & 1 \\ X_3 & Y_3 & 1 \\ X_4 - X_2 & Y_4 - Y_2 & 0 \end{vmatrix}\mathrm{d}\xi\mathrm{d}\eta \tag{20}$$

On the contrary the function ϕ_0 is assumed to have a constant value inside each cell ϕ_{0h}, so that the system of equations can be put in the form.

$$T(J)\phi(J) - \sum_{h=1}^{m} C + \sum_{g=1}^{n}(A_1 A_2)\begin{pmatrix}\phi_g(G)\\ \phi_g(G+1)\end{pmatrix} = \sum_{g=1}^{m}(B_1 B_2))\begin{pmatrix}q_g(G)\\ q_g(G+1)\end{pmatrix} \tag{21}$$

where

$$A_1 = K \int_{\partial D_g} N_1(\xi) \frac{\partial \psi_g}{\partial n} (J) \frac{1_g}{2} \, d\xi$$

$$A_2 = K \int_{\partial D_g} N_2(\xi) \frac{\partial \psi_g}{\partial n} (J) \frac{1_g}{2} \, d\xi$$

$$B_1 = K \int_{\partial D_g} N_1(\xi) \psi_g (J) \frac{1_g}{2} \, d\xi$$

$$B_2 = K \int_{\partial D_g} N_2(\xi) \psi_g (J) \frac{1_g}{2} \, d\xi$$

$$C = \frac{1}{\Delta t} \phi_{oh} \int_{D_h} \psi_h (J) \frac{1}{8} \begin{vmatrix} X_{1h} & Y_{1h} & 1 \\ X_{3h} & Y_{3h} & 1 \\ X_{4h} - X_{2h} & Y_{4h} - Y_{2h} & 0 \end{vmatrix} d\xi d\eta. \tag{22}$$

The computation of $A_1 A_2 B_1 B_2 C$ can be done either in closed form or by numerical procedures. This paper uses the second type except when an element is viewed from itself. The expressions are the following.

Numerically

$$A_1 = \frac{d}{8\pi} \frac{1_g}{\sqrt{(K\Delta t)}} \left[\sum_{i=1}^{j} (\xi_i - 1) \frac{1}{r_i} K_1 \left(\frac{r_i}{\sqrt{(K\Delta t)}} \right) \omega_i \right]$$

$$A_2 = \frac{-d}{8\pi} \frac{1_g}{\sqrt{(K\Delta t)}} \left[\sum_{i=1}^{j} (\xi_i + 1) \frac{1}{r_i} K_1 \left(\frac{r_i}{\sqrt{(K\Delta t)}} \right) \omega_i \right]$$

$$B_1 = \frac{-1_g}{8\pi} \left[\sum_{i=1}^{j} (\xi_i - 1) K_0 \left(\frac{r_i}{\sqrt{(K\Delta t)}} \right) \omega_i \right]$$

$$B_2 = \frac{1_g}{8\pi} \left[\sum_{i=1}^{j} (\xi_i + 1) K_0 \left(\frac{r_i}{\sqrt{(K\Delta t)}} \right) \omega_i \right]$$

$$C = \frac{1}{\Delta t} \phi_{oh} \frac{1}{16\pi K} \begin{vmatrix} x_{1h} & y_{1h} & 1 \\ x_{3h} & y_{3h} & 1 \\ x_{4h} - x_{2h} & y_{4h} - y_{2h} & 0 \end{vmatrix} \left[\sum_{i=1}^{j} K_0 \left(\frac{r_i}{\sqrt{(K\Delta t)}} \right) \omega_i \right]. \tag{23}$$

In closed form

$$B = \frac{-1_g}{2\pi} \left[\frac{1}{\alpha^2} - \frac{1}{\alpha} K_1(\alpha) - \frac{\pi}{2} \times [K_0(\alpha) L_{-1}(\alpha) - L_0(\alpha) K_{-1}(\alpha)] \right]$$

$$B = \frac{1_g}{2\pi} \left[\frac{1}{\alpha^2} - \frac{1}{\alpha} K_1(\alpha) \right] \tag{24}$$

where K_{-1} is the modified Bessel function of second Kind and order -1; L_0 is the modified Struve function of zero order; and L_{-1} is the modified Struve function of order -1.

4. EXAMPLES

In order to show some of the properties of the method several examples have been run. The first is (Fig. 1) a square plate with zero initial temperature inside and unity at the boundary for every instant.

The problem has been solved on a quarter of the plate because of the symmetries in which case we had a mixed type problem.

Considering $K = 1.$, the discretization has 20 elements at the boundary and 25 cells inside the domain.

The results have been compared with the theoretical ones presented for instance by Carslaw and Jaeger [7], and it is important to note the clear influence of the discretization time interval. In order to improve the results we have considered three zones; the first from $t = 0.0$ to $t = 0.1$ includes the initial rising of the temperature while the second reaches $t = 0.75$ and the third $t = 3.0$; the considered time intervals have been 0.005, 0.05 and 0.25 respectively. The accuracy has been good except in the first zone where it was finally necessary to use a time interval $\Delta t = 0.002$ in order to model the temperature variation.

Another classical example is a disc (Fig. 2) with the same boundary conditions as before. The discretization contains 18 boundary elements and 23 cells. The results shown correspond to $K = 1.$ and $\Delta t = 0.05$.

A more complex domain is shown in Fig. 3 where the data are the boundary temperatures at the beginning of the process. Again the figure has an axis of symmetry which has been used to reduce the area to be discretized and solved, to a mixed type problem. Discretization has 42 boundary elements and 49 cells. $K = 1$ and $\Delta t = 0.5$.

Finally a turbine-like shape has been analysed (Fig. 4) using 68 boundary elements and 114 cells and with the same step as before.

5. GENERAL COMMENTS

As has been seen, the procedure is simple and efficient and there is the possibility of introducing several improvements.

Fig. 1.

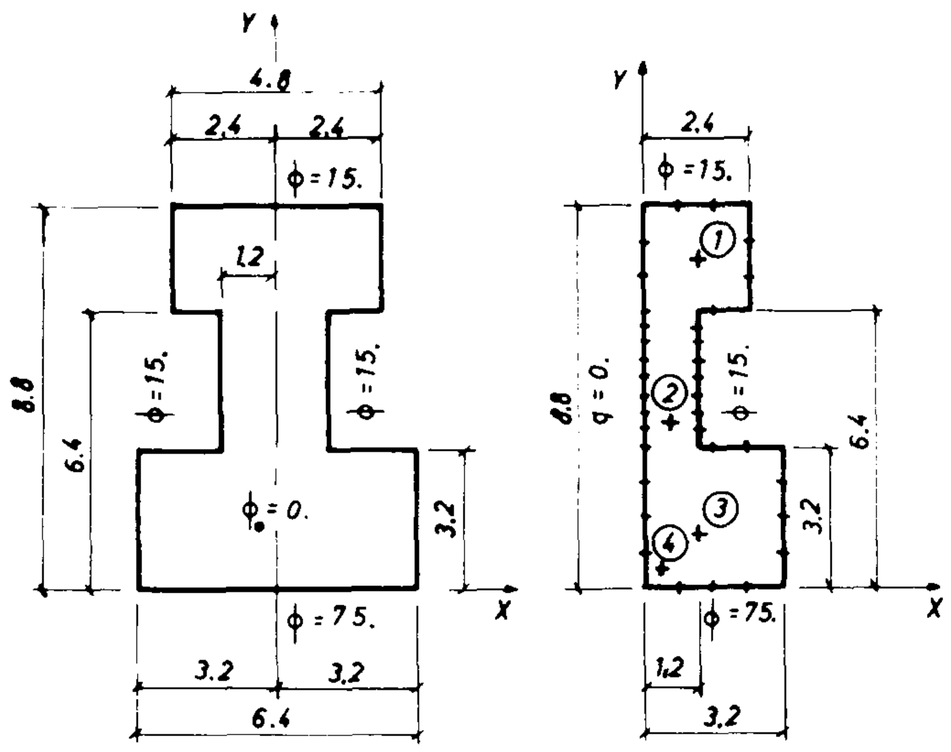
In problems with symmetric geometry subjected to symmetric or antisymmetric actions, it is in general possible to compute and group the corresponding coefficients in an automatic fashion and then elements are not required on the lines of symmetry. Whether or not this technique is advantageous is a matter of the user's taste (see, for instance [10]) although in some cases it produces more accurate results [13].

The most expensive part of the method is the numerical integration. We have not intended to use an adaptive procedure such as explained by Doblare *et al.* [13] or Lachat *et al.* [12]. In general the integration in the boundary is made using a standard Gauss rule with 4 points, except in neighbour elements where the results have been found analytically.

The domain integration has always been done numerically using a rule of 36 or 16 points inside each quadrilateral cell, depending on whether or not the cell is in contact with the boundary. The enclosed computer program only uses the first type of integration.

The integration constants have to be computed for each *different* Δt. When a unique Δt is chosen the integration constants are computed only once during the first iteration, and then stored for subsequent use. The problem then is the election of the interval size in order to maintain an acceptable accuracy without increasing the computation cost by too much.

We have not found any mathematical rule establishing



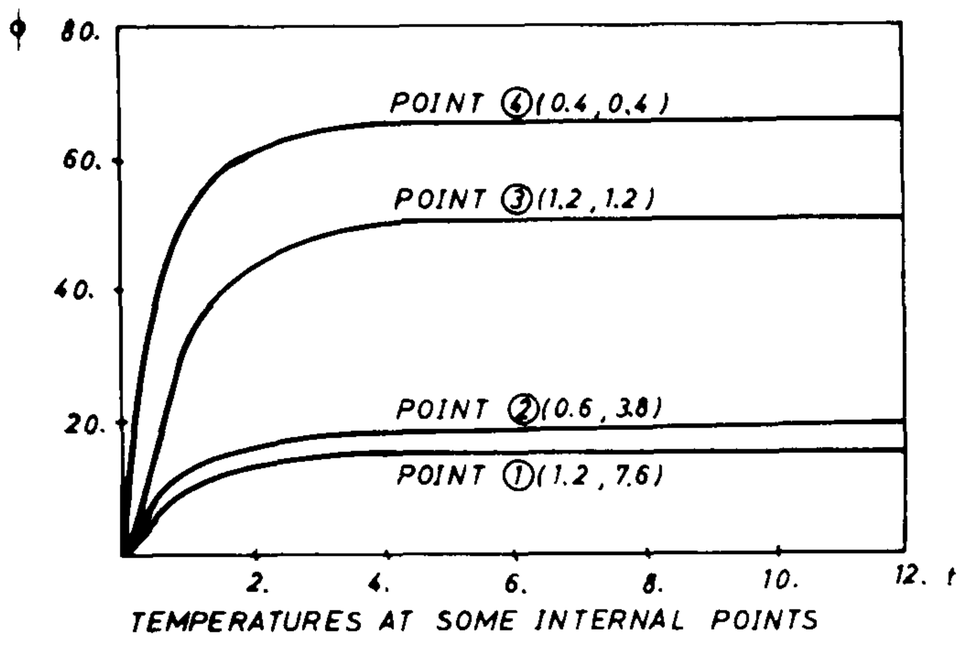


Fig. 3.

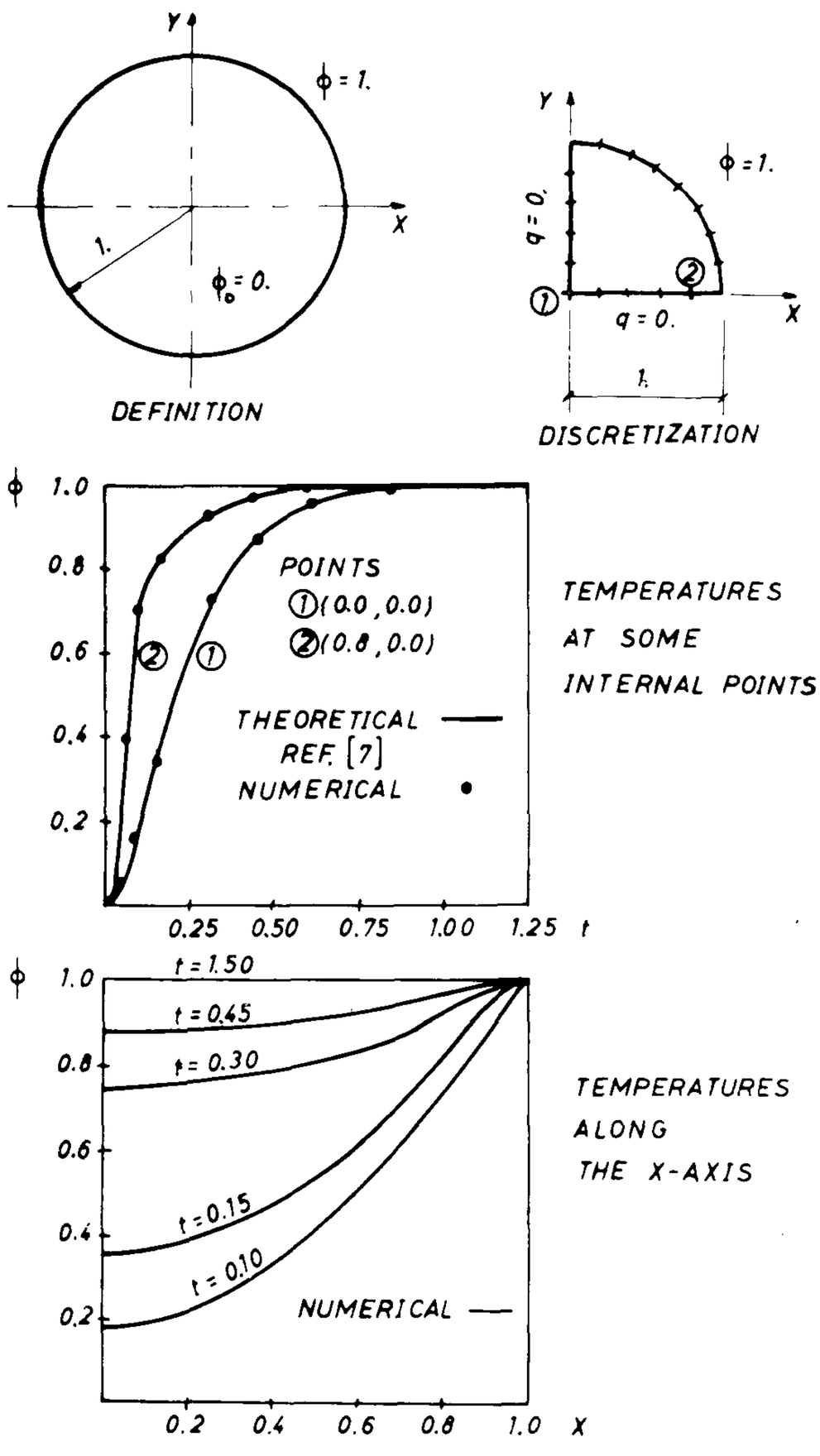


Fig. 2.

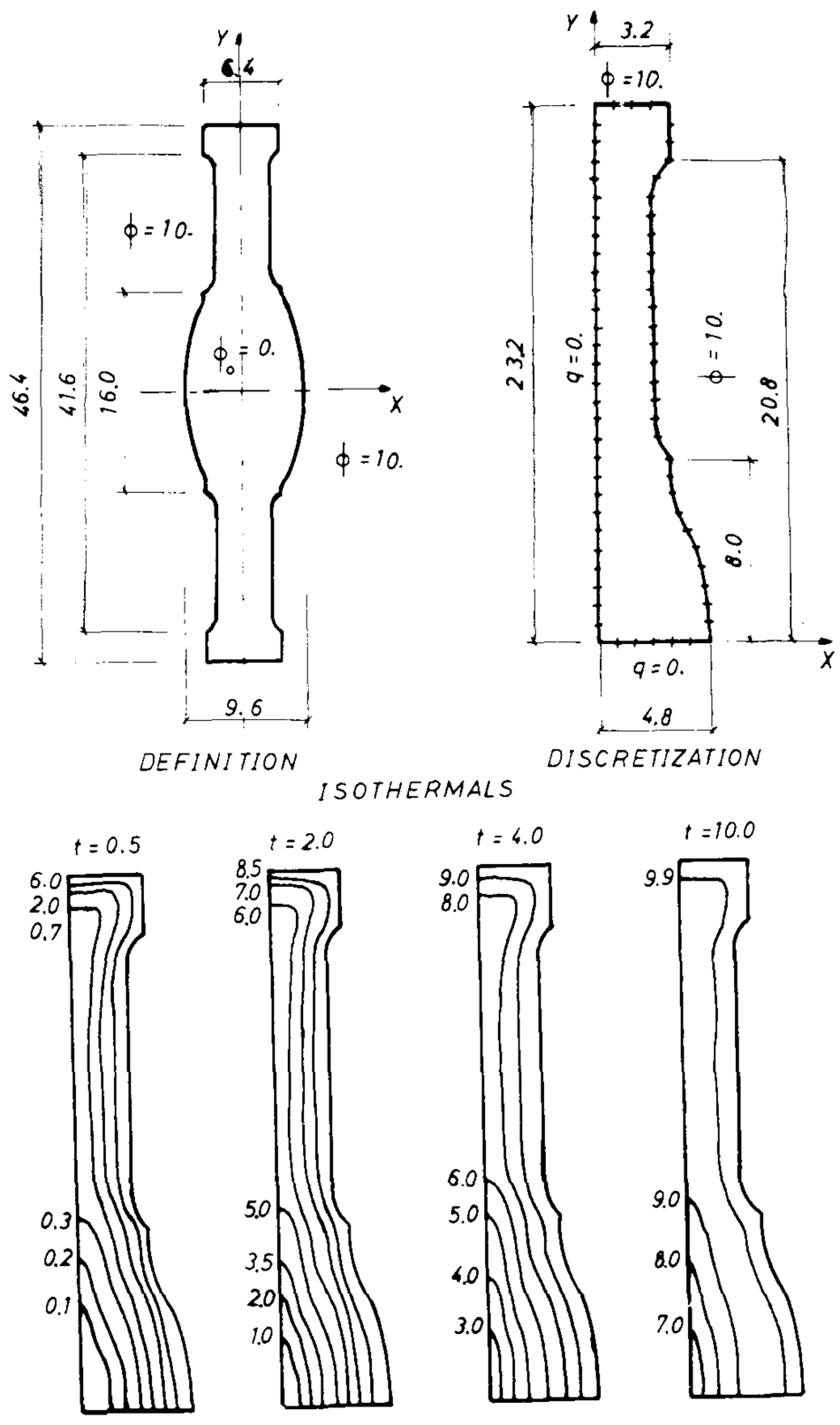


Fig. 4.

a rigorously founded criterion, but our experience with the method suggests that a good choice is usually

$$\Delta t = 0.0125 \frac{A}{K} \tag{25}$$

where A is the area enclosed by the domain; $K = k/pc$. With this rule the initial results are a bit polluted but they stabilize after a time of about

$$t \approx 0.028 \frac{A}{K}. \tag{26}$$

If more precision is needed it is possible to define three ranges of integration as in example 1. The examples we have run seem to agree well with the following rules:

If

$$0 \leqslant t \leqslant 0.025 \frac{A}{K}$$

take

$$\Delta t = 0.0050 \frac{A}{K} \tag{a}$$

If

$$0.025 \frac{A}{K} \leqslant t \leqslant 0.20 \frac{A}{K}$$

take

$$\Delta t = 0.0125 \frac{A}{K} \tag{b}$$

If

$$t \geqslant 0.20 \frac{A}{K}$$

take

$$\Delta t = 0.0600 \frac{A}{K}. \tag{c}$$

(27)

The first two rules imply 19 intervals. We have found that one more, of the (c) type, is usually enough to reach 95% of the stationary conditions. Then a good rule with which to start the study is to fix the step number in, say, 21 and to compare the values of the last two steps. These rules are, of course, intended for the transient situation after a sudden application of thermal loads. If the temperature varies it should be necessary to study the problem under consideration. A possible way of attack could be the Fourier analysis of the load and the determination of the step through the velocity of the temperature wave

$$v = \sqrt{(2kw)}$$

where w is the characteristic frequency. The comparison of the wave lengths with the size of the domain will give us an idea of the necessary time step.

6. CONCLUSIONS

The examples show the capability of the B.I.E.M. to treat transient potential problems with a discretization of the boundary and the use of integration cells inside the domain.

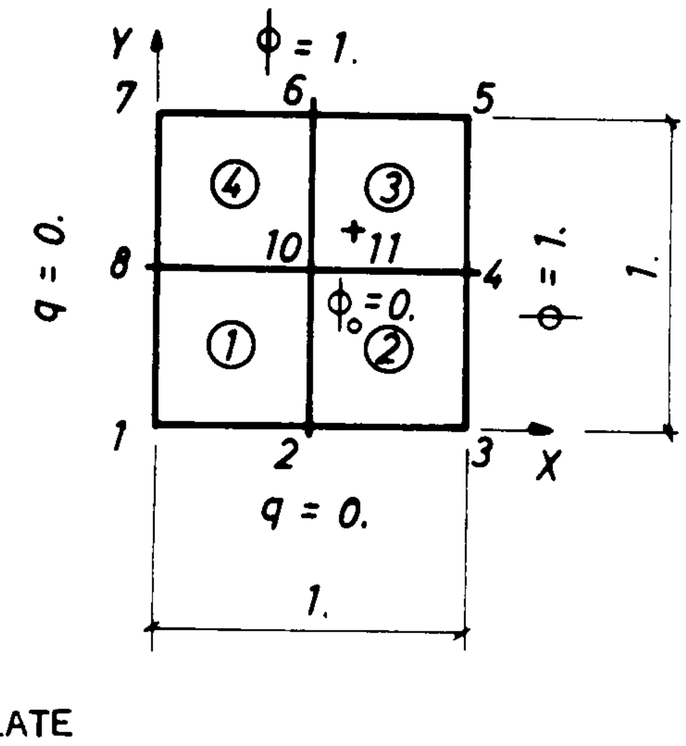
The time step to be used is problem dependent and a judicious choice has to be made in order to get sound results.

Although the problems analysed present steady-state boundary temperatures, the computer program prepared has the possibility of treating time dependent boundary-conditions with a very simple instruction.

REFERENCES

1. F. J. Rizzo and D. J. Shippy, A method of solution of certain problems of transient heat conduction. *AIAA J.* **11**, 2004–2009 (1970).
2. R. P. Shaw, An integral equation approach to diffusion. *Int. J. Heat Mass Transfer* **17**, 693–699 (1974).
3. Y. P. Chang, C. S. Kang and D. J. Chen, The use of fundamental Green's functions for the solution of problems of heat conduction in anisotropic media. *Int. J. Heat Mass Transfer* **16**, 1905–1918 (1973).
4. J. C. Lachat and A. Combescure, Laplace transform and boundary integral equations: application to transient heat conduction problems. 1*st Int. Conf. on Innovative Numerical Analysis in Appl. Engng.* Versailles (1977).
5. L. C. Wrobel and C. A. Brebbia, The boundary element method for steady-state and transient heat conduction. 1*st Int. Conf. on Num. Meth. in Thermal Problems*, Swansea (1979).
6. E. Alarcon, A. Martin and F. Paris, Boundary elements in potential and elasticity theory. *Comput. Structures* **10**, 351–362 (1979).
7. H. S. Carslaw and J. C. Jaeger, *Conduction of Heat in Solids*, 2nd Edn. Clarendon Press, Oxford (1969).
8. C. A. Brebbia and S. Walker, *Boundary Element Techniques in Engineering.* Newnes–Butterworths (1980).
9. L. Wrobel and C. Brebbia, A boundary element solution for time-dependent problems. 2*nd Int. Cong. on Numerical Meth. Engng*, Dunod. (1980).
10. J. O. Watson, Advanced implementation of the boundary element for two and three-dimensional elastostatics. *Developments in Boundary Element Methods*, Vol. 1, Chap. 3. Banerjee & Butterfield. Applied Science Publishers. (1979).
11. M. Boblaré and E. Alarcón, A three-dimensional BEM system. *Int. Seminar on Finite Element Systems.* Computational Mechanics Center. (1981).
12. J. C. Lachat and J. O. Watson, Effective numerical treatment of boundary integral equations. *Int. J. Num. Meth. Engng* **10**, 991–1105.
13. J. J. Anza and E. Alarcón, The effect of boundary conditions in the solution with B.I.E.M. of 3-D thermoelastic problems. To be published in *Software Engineering.*

APPENDIX




```
 1:    SQUARE PLATE
 2:     8   10   11    4      0      1.0       0.05       0.1
 3:     1     0.        0.
 4:     3     1.        0.
 5:     5     1.        1.
 6:     7     0.        1.
 7:     9     0.        0.
 8:     1     1         0.          0.
 9:     3     31.       0.
10:     5     41.
11:     7     21.                   0.
12:     9     1         0.          0.
13:    10      0.5      0.5
14:    11      0.7      0.7
15:     1     1    2    10     8     0.
16:     2     2    3     4    10     0.
17:     3    10    4     5     6     0.
18:     4     8   10     6     7     0.
19:     0     0   01.
```

Fig. A1.

```
 1:C
 2:C PROGRAM MECCA
 3:C
 4:       DIMENSION X(151),Y(151),XTD(80),YTD(80),V1(80),V2(80),V3(80),
 5:      1V4(80),NCOD(51),FII(51),DFII(51,2),FI(50,30),DFI(50,2,30),
 6:      2FINT(20,30),FUI(80),FPCT(80,30),IPIVO(50),ZZ(6),ZO(6),SI(36),
 7:      3YI(36),ZI(36),VI(36),AMEG(36),DFF(50,2),COEF(50,50),CARGO(50),
 8:      4CARGA(50),V(2),A(2),B(2),TA(80,50,4),DA(20,50,4),D(20,80),
 9:      5E(80,80),F(50,80),ITIT(40),TIME(30)
10:       INTEGER V1,V2,V3,V4,D1,D2,D3,D4
11:       DATA ZZ/0.23861918,-0.23861918,0.66120938,-0.66120938,0.93246951,-
12:      10.93246951/,ZO/0.46791393,0.46791393,0.36076157,0.36076157,0.17132
13:      2449,0.17132449/
14:       DIS(X1,Y1,X2,Y2)=SQRT((X1-X2)**2+(Y1-Y2)**2)
15:C
16:C ASSIGN DATA SET NUMBERS FOR INPUT,LEC AND OUTPUT,IMP
17:C
18:       LEC=5
19:       IMP=6
20:       NPMAX=50
21:C
22:C READ PROBLEM TITLE
23:C
24:       READ(LEC,1000)ITIT
25: 1000 FORMAT(40A2)
26:       WRITE(IMP,1001)ITIT
27: 1001 FORMAT(1X,40A2)
28:C
29:C READ BASIC PARAMETERS
30:C NPC =NUMBER OF BOUNDARY NODES
31:C NPD =MAX. NUMBER OF INTERNAL POINTS FOR DISCRETISATION
32:C NPI =MAX. NUMBER OF INTERNAL POINTS WHERE THE FUNCTION
33:C       WILL BE CALCULATED
```

```
C NTD =NUMBER OF INTERNAL CELLS
C NCOA=SUPPORT CONDITION FOR NEUMANN PROBLEM
C CTER=TERMOMETRIC CONDUCTIVITY
C TINC=TIME INTERVAL
C TFIN=FINAL TIME
C
      READ(LEC,1005)NPC,NPD,NPI,NTD,NCOA,CTER,TINC,TFIN
 1005 FORMAT(5I5,5X,3F10.0)
      WRITE(IMP,1006)NPC,NPD,NPI,NTD,NCOA,CTER,TINC,TFIN
 1006 FORMAT(2X,5I5,3F10.5)
      DO 10 I=1,NPC
      FII(I)=0.
      DFII(I,1)=0.
   10 DFII(I,2)=0.
      L=0
C
C READ BOUNDARY NODES COORDINATES
C
   15 READ(LEC,1010)I,X(I),Y(I)
 1010 FORMAT(I5,5X,2F10.0)
      IF(L)20,25,20
   20 NINT=I-L
      AX=(X(I)-X(L))/NINT
      AY=(Y(I)-Y(L))/NINT
   25 L=L+1
      IF(I-L)500,35,30
   30 X(L)=X(L-1)+AX
      Y(L)=Y(L-1)+AY
      GO TO 25
   35 IF(NPC-I)45,40,15
   40 X(NPC+1)=X(1)
      Y(NPC+1)=Y(1)
   45 L=0
C
C READ BOUNDARY CONDITIONS
C NCOD(I)=CODE OF THE NODE I ACCORDING TO THE NEIGHBOURG CONDITIONS
C CODE 1 =THE VALUES OF THE FUNCTION DERIVATIVE BEFORE AND AFTER OF
C           THE NODE ARE KNOWN
C CODE 2 =THE VALUES OF THE FUNCTION AND
C           THE FUNCTION DERIVATIVE AFTER ARE KNOWN
C CODE 3 =THE VALUES OF THE FUNCTION DERIVATIVE BEFORE AND
C           THE FUNCTION ARE KNOWN
C CODE 4 =THE VALUES OF THE FUNCTION IN
C           A SHARP CORNER ARE KNOWN
C CODE 5 =US 5 BUT IN CONTINUOUS BOUNDARY
C FII(I) =THE VALUE OF THE FUNCTION IN THE NODE I IF IT IS KNOWN
C DFII(I,1),DFII(I,2) =THE VALUES OF THE FUNCTION DERIVATIVES
C                       BEFORE AND AFTER IF THEY ARE KNOWN
C
   85 READ(LEC,1020)I,NCOD(I),FII(I),DFII(I,1),DFII(I,2)
 1020 FORMAT(2I5,3F10.0)
      IF(I-L-1)525,55,50
   50 NINT=I-L
      AX=(FII(I)-FII(L))/NINT
      AY=(DFII(I,1)-DFII(L,2))/NINT
   55 L=L+1
      IF(I-L)525,70,60
   60 IF(NCOD(I).LE.3.AND.NCOD(I).NE.2) GO TO 65
      NCOD(L)=5
      FII(L)=FII(L-1)+AX
      GO TO 55
   65 NCOD(L)=1
      DFII(L,1)=DFII(L-1,2)+AY
      DFII(L,2)=DFII(L,1)
      GO TO 55
   70 IF(NPC-I)80,75,85
   75 FII(NPC+1)=FII(1)
      DFII(NPC+1,1)=DFII(1,1)
      DFII(NPC+1,2)=DFII(1,2)
      NCOD(NPC+1)=NCOD(1)
   80 WRITE(IMP,1025)
 1025 FORMAT(1H1,5X,'BOUNDARY NODES-BOUNDARY CONDITIONS
     1'/3X,'NODE',12X,'COOR X',9X,'COOR Y',10X,'CODE',11X,'POTENTIAL',8
     2X,'FLUX BEFORE',9X,'FLUX AFTER')
      WRITE(IMP,1030)(I,X(I),Y(I),NCOD(I),FII(I),(DFII(I,K),K=1,2),I=1,N
     1PC)
 1030 FORMAT(2X,I5,3X,2F15.4,8X,I5,1X,3F20.5)
  500 CONTINUE
  525 CONTINUE
```

```
113:C
114:C READ INTERNAL POINTS(DISCRESITATION)COORDINATES
115:C
116:       READ(LEC,1026)(X(IPD),Y(IPD),IPD=NPC+2,NPD)
117: 1026 FORMAT(10X,2F10.0)
118:       WRITE(IMP,1011)
119: 1011 FORMAT(1H1,5X,'INTERNAL POINTS - DISCRETISATION'/3X,'NODE',12X,
120:      1'COOR X',9X,'COOR Y')
121:       WRITE(IMP,1012)(IPD,X(IPD),Y(IPD),IPD=NPC+2,NPD)
122: 1012 FORMAT(2X,I5,3X,2F15.4)
123:       IF(NPI.EQ.NPD)GO TO 1032
124:C
125:C READ INTERNAL POINTS(WHERE THE FUNCTION WILL BE CALCULATED)COORDINATES
126:C
127:       READ(LEC,1026)(X(IPI),Y(IPI),IPI=NPD+1,NPI)
128:       WRITE(IMP,1031)
129: 1031 FORMAT(1H1,5X,'INTERNAL POINTS - POTENTIAL CALCULATION'/3X,'NODE',
130:      112X,'COOR X',9X,'COOR Y')
131:       WRITE(IMP,1012)(IPI,X(IPI),Y(IPI),IPI=NPD+1,NPI)
132:C
133:C READ INTERNAL CELLS INCIDENCES AND INITIAL POTENTIAL
134:C
135: 1032 READ(LEC,1040)(V1(ITD),V2(ITD),V3(ITD),V4(ITD),FUI(ITD),ITD=1,NTD)
136: 1040 FORMAT(5X,4I5,5X,1F10.0)
137:       WRITE(IMP,1050)
138: 1050 FORMAT(1H1,5X,'CELLS-INCIDENCES'/7X,'CELL',16X,'VER 1'
139:      1,9X,'VER 2',9X,'VER 3',9X,'VER 4',9X,'INITIAL POTENTIAL')
140:       WRITE(IMP,1060)(ITD,V1(ITD),V2(ITD),V3(ITD),V4(ITD),FUI(ITD),ITD=1
141:      1,NTD)
142: 1060 FORMAT(6X,I5,14X,I5,9X,I5,9X,I5,9X,I5,13X,1F10.5)
143:       DO 90 ITD=1,NTD
144:       D1=V1(ITD)
145:       D2=V2(ITD)
146:       D3=V3(ITD)
147:       D4=V4(ITD)
148:       XTD(ITD)=(X(D1)+X(D2)+X(D3)+X(D4))*0.25
149:    90 YTD(ITD)=(Y(D1)+Y(D2)+Y(D3)+Y(D4))*0.25
150:       WRITE(IMP,1070)
151: 1070 FORMAT(1H1,5X,'CELLS CENTER POINTS'/3X,'NODE',12X,'COOR X'
152:      1,9X,'COOR Y')
153:       WRITE(IMP,1075)(ITR,XTD(ITR),YTD(ITR),ITR=1,NTD)
154: 1075 FORMAT(2X,I5,3X,2F15.4)
155:C
156:C READ CODES
157:C
158:       READ(LEC,1090)NCODT,NCODD,NCOTT,CC
159: 1090 FORMAT(3I5,1F5.0)
160:       IF(NCODT)92,92,91
161:    91 WRITE(IMP,1091)
162: 1091 FORMAT(5X,'NOT IMPRESION POTENTIAL OF CENTER POINTS OF CELLS'
163:      1)
164:       GO TO 93
165:    92 WRITE(IMP,1092)
166: 1092 FORMAT(5X,'IMPRESION POTENTIAL OF CENTER POINTS OF CELLS')
167:    93 IF(NCODD)94,94,95
168:    94 WRITE(IMP,1093)
169: 1093 FORMAT(5X,'NOT CALCULATION CELLS VERTEX POTENTIAL')
170:       GO TO 99
171:    95 WRITE(IMP,1094)
172: 1094 FORMAT(5X,'CALCULATION CELLS VERTEX POTENTIAL')
173:    99 IF(NCOTT)101,101,102
174:   101 WRITE(IMP,1095)
175: 1095 FORMAT(5X,'DOMAIN INITIAL POTENTIAL EQUAL ZERO')
176:       GO TO 103
177:   102 WRITE(IMP,1096)
178: 1096 FORMAT(5X,'DOMAIN INITIAL POTENTIAL NOT EQUAL ZERO')
179:C
180:C LOAD FACTOR FOR UNIFORM VARIABLE BOUNDARY CONDITIONS
181:C
182:   103 WRITE(IMP,1097)CC
183: 1097 FORMAT(5X,'LOAD FACTOR=',1F5.2)
184:       WRITE(IMP,1200)ITIT
185: 1200 FORMAT(1H1,10X,40A2/10X,80('*'))
186:       NTIT=1+TFIN/TINC
187:       II=1
188:C
189:C CALCULATION INTERPOLATION FUNCTIONS
190:C
191:       DO 900 J=1,6
```

```
192:        DO 905 K=1,6
193:        SI(II)=0.25*(1-ZZ(J))*(1-ZZ(K))
194:        YI(II)=0.25*(1+ZZ(J))*(1-ZZ(K))
195:        ZI(II)=0.25*(1+ZZ(J))*(1+ZZ(K))
196:        VI(II)=0.25*(1-ZZ(J))*(1+ZZ(K))
197:        AMEG(II)=ZO(J)*ZO(K)
198:        II=II+1
199:    905 CONTINUE
200:    900 CONTINUE
201:        FC=1.
202:C
203:C INITIALISATION TIME INTEGRATION
204:C
205:        DO 333 IT=1,NTIT
206:        DO 96 J=1,NPC
207:        FI(J,IT)=FII(J)*FC
208:        DFI(J,1,IT)=DFII(J,1)*FC
209:     96 DFI(J,2,IT)=DFII(J,2)*FC
210:C
211:C INTEGRATION ON THE BOUNDARY
212:C
213:        DO 200 NODO=1,NPC
214:        IF(IT-1)601,601,602
215:    601 DDIAG=0.
216:        DO 605 IDD=1,ND
217:        D1=V1(IDD)
218:        D2=V2(IDD)
219:        D3=V3(IDD)
220:        D4=V4(IDD)
221:        FPPT=1.
222:C
223:C CALCULATION INTEGRATION CONSTANT C FOR BOUNDARY NODES
224:C
225:        CALL CONUM(X(NODO),Y(NODO),X(D1),Y(D1),X(D2),Y(D2),X(D3),Y(D3),X(D
226:       14),Y(D4),FPPT,CTER,TINC,SI,YI,ZI,VI,AMEG,G)
227:        F(NODO,IDD)=G
228:    605 DDIAG=DDIAG+G
229:        DO 100 I=1,NPC
230:    100 COEF(NODO,I)=0.
231:        CDIAG=0.
232:        C=0.
233:        CARGO(NODO)=0.
234:C
235:C CALCULATION INTEGRATION CONSTANTS A,B FOR BOUNDARY NODES
236:C
237:        DO 190 IELEM=1,NPC
238:        IF(IELEM.EQ.NODO)GO TO 152
239:        IF((IELEM+1),EQ.NODO)GO TO 151
240:        IF(IELEM.EQ.NPC.AND.NODO.EQ.1)GO TO 151
241:        CALL ABNUM(X(NODO),Y(NODO),X(IELEM),Y(IELEM),X(IELEM+1),
242:       1Y(IELEM+1),CTER,TINC,A(1),A(2),B(1),B(2))
243:        GO TO 153
244:    152 CALL ABANA(X(NODO),Y(NODO),X(IELEM+1),Y(IELEM+1),CTER,TINC,A(1),A(
245:       12),B(1),B(2))
246:        GO TO 153
247:    151 CALL ABANA(X(IELEM),Y(IELEM),X(NODO),Y(NODO),CTER,TINC,A(1),A(2),B
248:       1(1),B(2))
249:        AX=B(1)
250:        B(1)=B(2)
251:        B(2)=AX
252:    153 CDIAG=CDIAG+A(1)+A(2)
253:        DO 179 K=1,2
254:        KN=IELEM+K-1
255:        IF(KN.EQ.NPC+1)KN=1
256:C
257:C THE INTEGRATION RESULTS ARE PLACED IN THE COEFFICIENTS MATRIX
258:C OR IN THE LOAD VECTOR ACCORDING TO THE CODE OF THE NODES
259:C
260:        GO TO(1,2,3,4,6),NCOD(KN)
261:      1 CARGO(NODO)=CARGO(NODO)+B(K)*DFI(KN,3-K,IT)
262:        COEF(NODO,KN)=COEF(NODO,KN)+A(K)
263:        GO TO 179
264:      2 IF(K.EQ.2)GO TO 160
265:    165 CARGO(NODO)=CARGO(NODO)+B(K)*DFI(KN,3-K,IT)-A(K)*FI(KN,IT)
266:        GO TO 179
267:      3 IF(K.EQ.2)GO TO 165
268:    160 CARGO(NODO)=CARGO(NODO)-A(K)*FI(KN,IT)
269:        COEF(NODO,KN)=COEF(NODO,KN)-B(K)
270:        GO TO 179
```

```
271:     4 DEN=DIS(X(IELEM),Y(IELEM),X(IELEM+1),Y(IELEM+1))
272:       XNUE=(Y(IELEM+1)-Y(IELEM))/DEN
273:       YNUE=(X(IELEM)-X(IELEM+1))/DEN
274:       KNMA=KN+1
275:       KNME=KN-1
276:       IF(KN.EQ.1)KNME=NPC
277:       IF(KN.EQ.NPC)KNMA=1
278:       DEN=(X(KNMA)-X(KN))*(Y(KNME)-Y(KN))-(X(KNME)-X(KN))*(Y(KNMA)-Y(KN)
279:      1)
280:C
281:C COMPUTE GRADIENT COMPONENTS IN THE NODE UNDER STUDY
282:C
283:       DFX=((Y(KN)-Y(KNMA))*FI(KNME,IT)+(Y(KNMA)-Y(KNME))*FI(KN,IT)+(Y(KN
284:      1ME)-Y(KN))*FI(KNMA,IT))/DEN
285:       DFY=((X(KNMA)-X(KN))*FI(KNME,IT)+(X(KNME)-X(KNMA))*FI(KN,IT)+(X(KN
286:      1)-X(KNME))*FI(KNMA,IT))/DEN
287:       COSE=0.7071
288:       IF(DFX.NE.0.)GO TO 170
289:       IF(DFY.EQ.0.)GO TO 175
290:  170  COSE=(DFX*XNUE+DFY*YNUE)/SQRT(DFX**2+DFY**2)
291:  175 KK=3-K
292:       DFF(KN,KK)=COSE
293:       COEF(NODO,KN)=COEF(NODO,KN)-B(K)*COSE
294:       CARGO(NODO)=CARGO(NODO)-A(K)*FI(KN,IT)
295:       GO TO 179
296:     6 COSE=1
297:       GO TO 175
298:  179 CONTINUE
299:  190 CONTINUE
300:C
301:C MODIFY THE ELEMENTS OF THE PRINCIPAL DIAGONAL IN
302:C THE COEFFICIENTS MATRIX OF THE FUNCTIONS
303:C
304:       TDIAG=CDIAG-DDIAG
305:       IF(NCOD(NODO)-1)130,181,180
306:  180 CARGO(NODO)=CARGO(NODO)+TDIAG*FI(NODO,IT)
307:       GO TO 702
308:  181 COEF(NODO,NODO)=COEF(NODO,NODO)-TDIAG
309:  702 IF(NCOTT)650,650,705
310:C
311:C INTRODUCTION OF CELLS POTENTIAL AND LOAD FACTOR
312:C
313:  705 DO 700 ITT=1,NTD
314:  700 C=C+F(NODO,ITT)*FUI(ITT)
315:       GO TO 650
316:  602 C=0.
317:       DO 113 ITD=1,NTD
318:  113 C=C+F(NODO,ITD)*FPCT(ITD,IT-1)
319:  650 CARGA(NODO)=CARGO(NODO)*FC+C
320:  200 CONTINUE
321:C
322:C INTRODUCTION OF SUPPORT CONDITION FOR NEUMANN PROBLEM
323:C
324:       IF(NCOA.EQ.0)GO TO 250
325:       DO 210 J=1,NPC
326:       COEF(J,NCOA)=0.
327:  210 COEF(NCOA,J)=0.
328:       COEF(NCOA,NCOA)=1.
329:       CARGA(NCOA)=0.
330:  250 IF(IT-1)800,600,805
331:  800 V(1)=1.
332:C
333:C SOLUTION OF THE SYSTEM OF EQUATIONS
334:C GJR IS A STANDARD SUBROUTINE FOR MATRIX INVERSION
335:C
336:       CALL GJR(COEF,NPMAX,NPMAX,NPC,NPC,*888,IPIVO,V)
337:       GO TO 805
338:  888 WRITE(IMP,1400)
339: 1400 FORMAT(1X,'ERROR....INVERSE MATRIX')
340:       STOP
341:  805 DO 300 NODO=1,NPC
342:       SOL=0.
343:       DO 810 L=1,NPC
344:  810 SOL=SOL+COEF(NODO,L)*CARGA(L)
345:       NC=NCOD(NODO)
346:C
347:C ASSEMBLE AND MODIFY THE RESULTANT VALUES OF
348:C THE SYSTEM OF EQUATIONS
349:C
```

```
350:       GO TO(260,265,265,270,275),NC
351:   260 FI(NODO,IT)=SOL
352:       GO TO 300
353:   265 NC=NC-1
354:       DFI(NODO,NC,IT)=SOL
355:       GO TO 300
356:   270 DO 280 I=1,2
357:       IF(ABS(DFF(NODO,I)),LE,1,E-10)GO TO 280
358:       DFI(NODO,I,IT)=SOL*DFF(NODO,I)
359:   280 CONTINUE
360:       GO TO 300
361:   275 DO 285 I=1,2
362:   285 DFI(NODO,I,IT)=SOL
363:   300 CONTINUE
364:       IF(CC,EQ,0,)CC=1,
365:       FC=CC**IT
366:       IF (NCODD)411,411,412
367:   411 IF(NPI,EQ,NPD)GO TO 399
368:       IPR=NPD+1
369:       GO TO 413
370:   412 IPR=NPC+2
371:   413 INT=IPR
372:C
373:C COMPUTE FUNCTION VALUE IN INTERNAL POINTS
374:C
375:       DO 420 MMM=1,NPI-IPR+1
376:       IF(IT-1)421,421,422
377:   421 C=0,
378:C
379:C CALCULATION INTEGRATION CONSTANTS A,B
380:C
381:       DO 610 J=1,NPC
382:       CALL ABNUM(X(INT),Y(INT),X(J),Y(J),X(J+1),Y(J+1),CTER,TINC,A(1),A(
383:      12),B(1),B(2))
384:       DA(MMM,J,1)=A(1)
385:       DA(MMM,J,2)=A(2)
386:       DA(MMM,J,3)=B(1)
387:   610 DA(MMM,J,4)=B(2)
388:C
389:C CALCULATION INTEGRATION CONSTANT C
390:C
391:       DO 710 JJD=1,NTD
392:       D1=V1(JJD)
393:       D2=V2(JJD)
394:       D3=V3(JJD)
395:       D4=V4(JJD)
396:       FPPT=1,
397:       CALL CDNUM(X(INT),Y(INT),X(D1),Y(D1),X(D2),Y(D2),X(D3),Y(D3),X(D4)
398:      1,Y(D4),FPPT,CTER,TINC,SI,YI,ZI,VI,AMEG,G)
399:   710 D(MMM,JJD)=G
400:       IF(NCOTT)424,424,706
401:   706 DO 777 JDJ=1,NTD
402:   777 C=C+D(MMM,JDJ)*FUI(JDJ)
403:       GO TO 424
404:   422 C=0,
405:       DO 423 JTD=1,NTD
406:   423 C=C+D(MMM,JTD)*FPCT(JTD,IT-1)
407:   424 HPCT=0,
408:       DO 430 J=1,NPC
409:       JJ=J+1
410:       IF(J,EQ,NPC)JJ=1
411:   430 HPCT=HPCT-DA(MMM,J,1)*FI(J,IT)-DA(MMM,J,2)*FI(JJ,IT)+DA(MMM,J,3)*D
412:      1FI(J,2,IT)+DA(MMM,J,4)*DFI(JJ,1,IT)
413:       FINT(MMM,IT)=HPCT+C
414:   420 INT=INT+1
415:C
416:C COMPUTE FUNCTION VALUE IN CENTER NODES OF CELLS
417:C
418:   399 DO 400 ITD=1,NTD
419:       IF(IT-1)401,401,402
420:   401 C=0,
421:C
422:C CALCULATION INTEGRATION CONSTANTS A,B
423:C
424:       DO 615 J=1,NPC
425:       CALL ABNUM(XTD(ITD),YTD(ITD),X(J),Y(J),X(J+1),Y(J+1),CTER,TINC,A(1
426:      1),A(2),B(1),B(2))
427:       TA(ITD,J,1)=A(1)
428:       TA(ITD,J,2)=A(2)
```

```
      TA(ITD,J,3)=B(1)
  615 TA(ITD,J,4)=B(2)
C
C CALCULATION INTEGRATION CONSTANT C
C
      DO 715 IID=1,NTD
      D1=V1(IID)
      D2=V2(IID)
      D3=V3(IID)
      D4=V4(IID)
      FPPT=1.
      CALL CONUM(XTD(ITD),YTD(ITD),X(D1),Y(D1),X(D2),Y(D2),X(D3),Y(D3),X
     1(D4),Y(D4),FPPT,CTER,TINC,SI,YI,ZI,VI,AMEG,G)
  715 E(ITD,IID)=G
      IF(NCOTT)403,403,707
  707 DO 779 IDI=1,NTD
  779 C=C+E(ITD,IDI)*FUI(IDI)
      GO TO 403
  402 C=0.
      DO 440 JTD=1,NTD
  440 C=C+E(ITD,JTD)*FPCT(JTD,IT-1)
  403 HPCT=0.
      DO 450 J=1,NPC
      JJ=J+1
      IF(J.EQ.NPC)JJ=1
  450 HPCT=HPCT-TA(ITD,J,1)*FI(J,IT)-TA(ITD,J,2)*FI(JJ,IT)+TA(ITD,J,3)*D
     1FI(J,2,IT)+TA(ITD,J,4)*DFI(JJ,1,IT)
  400 FPCT(ITD,IT)=HPCT+C
      TIME(IT)=TINC*IT
      WRITE(IMP,1250) TIME(IT)
 1250 FORMAT(1H1,10X,1F10.5/5X,'BOUNDARY NODES'/6X,'NODE',10X,
     1'POTENTIAL',9X,'FLUX BEFORE',9X,'FLUX AFTER')
      WRITE(IMP,1260)(I,FI(I,IT),(DFI(I,J,IT),J=1,2),I=1,NPC)
 1260 FORMAT(5X,I5,3F20.5)
      IF(NCODT)502,502,501
  502 WRITE(IMP,1270)TIME(IT)
 1270 FORMAT(1H1,10X,1F10.5/5X,'CELLS CENTER POINTS'/6X,'NODE',1
     10X,'POTENTIAL')
      WRITE(IMP,1280)(I,FPCT(I,IT),I=1,NTD)
 1280 FORMAT(5X,I5,1F20.5)
  501 IF(NPI.EQ.NPD.AND.NCODD.EQ.0)GO TO 333
      WRITE(IMP,1290)TIME(IT)
 1290 FORMAT(1H1,10X,1F10.5/5X,'INTERNAL POINTS'/6X,'NODE',10X,
     1'POTENTIAL')
      WRITE(IMP,1300)((I-1+IPR),FINT(I,IT),I=1,NPI-IPR+1)
 1300 FORMAT(5X,I5,1F20.5)
  333 CONTINUE
      STOP
      END
      SUBROUTINE ABNUM(XP,YP,X1,Y1,X2,Y2,CTER,TINC,A1,A2,B1,B2)
C
C THIS SUBROUTINE COMPUTE NUMERICALLY THE INTEGRATION CONSTANTS A,B
C
      DIMENSION XI(4),OMEG(4)
      DATA XI/0.86113631,-0.86113631,0.33998104,-0.33998104/,
     1   OMEG/0.34785485,0.34785485,0.65214515,0.65214515/
      AX=(X2-X1)/2.
      AY=(Y2-Y1)/2.
      BX=(X2+X1)/2.
      BY=(Y2+Y1)/2.
      IF(AX)10,20,10
   10 DIS=ABS((AY*XP/AX-YP+Y1-AY*X1/AX)/SQRT((AY/AX)**2+1))
      GO TO 30
   20 DIS=ABS(XP-X1)
   30 SIG=(X1-XP)*(Y2-YP)-(X2-XP)*(Y1-YP)
      IF(SIG.LT.0) DIS=-DIS
      A1=0.
      A2=0.
      B1=0.
      B2=0.
      DO 40 I=1,4
      XC=AX*XI(I)+BX
      YC=AY*XI(I)+BY
      R=SQRT((XP-XC)**2+(YP-YC)**2)
      H=DIS*OMEG(I)*BE1(R,CTER,TINC)*SQRT(AX**2+AY**2)/R
      G=OMEG(I)*BE0(R,CTER,TINC)*SQRT(AX**2+AY**2)
      A1=A1+(XI(I)-1)*H/(4.*3.141592*SQRT(CTER*TINC))
      A2=A2-(XI(I)+1)*H/(4.*3.141592*SQRT(CTER*TINC))
      B1=B1-(XI(I)-1)*G/(4.*3.141592)
```

```
508:    40 B2=B2+(XI(I)+1)*G/(4.*3.141592)
509:       RETURN
510:       END
511:       SUBROUTINE ABANA(X1,Y1,X2,Y2,CTER,TINC,A1,A2,B1,B2)
512:C
513:C THIS SUBROUTINE COMPUTE ANALITICALLY THE INTEGRATION CONSTANT B
514:C
515:       A1=0.
516:       A2=0.
517:       DIS=SQRT((X2-X1)**2+(Y2-Y1)**2)
518:       P=BE0(DIS,CTER,TINC)*ST1(DIS,CTER,TINC)
519:       Q=(P+BE1(DIS,CTER,TINC)*ST0(DIS,CTER,TINC))*3.1415926/2
520:       ALF=DIS/SQRT(CTER*TINC)
521:       S=(ALF*BE1(DIS,CTER,TINC)-1)/ALF**2
522:       B1=(Q+S)*DIS/(2.*3.141592)
523:       B2=-S*DIS/(2.*3.141592)
524:       RETURN
525:       END
526:       SUBROUTINE CDNUM(XP,YP,X1,Y1,X2,Y2,X3,Y3,X4,Y4,FPCT,CTER,TINC,SI,Y
527:      1I,ZI,VI,AMEG,G)
528:C
529:C THIS SUBROUTINE COMPUTE NUMERICALLY THE INTEGRATION CONSTANT C
530:C
531:       DIMENSION SI(36),YI(36),ZI(36),VI(36),AMEG(36)
532:       H=0.
533:       AACO=0.125*(Y1*(X4-X2)+X3*(Y4-Y2)-Y3*(X4-X2)-X1*(Y4-Y2))
534:       AACO=ABS(AACO)
535:       DO 30 I=1,36
536:       XC=X1*SI(I)+X2*YI(I)+X3*ZI(I)+X4*VI(I)
537:       YC=Y1*SI(I)+Y2*YI(I)+Y3*ZI(I)+Y4*VI(I)
538:       R=SQRT((XP-XC)**2+(YP-YC)**2)
539:       S=1/10.**6
540:       IF (R-S) 30,30,10
541:    10 D=BE0(R,CTER,TINC)*AMEG(I)
542:       H=H+D
543:    30 CONTINUE
544:       G=(H/(6.283184*CTER*TINC))*AACO
545:       RETURN
546:       END
547:       FUNCTION BE0(DIS,CTER,TINC)
548:C
549:C THIS FUNCTION COMPUTE THE VALUE OF
550:C THE MODIFIED BESSEL FUNCTION SECOND KIND ORDER 0
551:C
552:       DOUBLE PRECISION DP,XZ,XT,TZ,CO,FK0,E
553:       X=DIS/SQRT(CTER*TINC)
554:       IF(X-4.)5,3,3
555:     3 DP=X
556:       FK0=(1.253314*DEXP(-DP)/SQRT(DP))*(1.-0.125/DP+0.0703125/DP**2-0.0
557:      1732421875/DP**3+0.1121520996/DP**4)
558:       GO TO 30
559:     5 DP=X/2.
560:       FK0=-0.577216649-DLOG(DP)
561:       E=0.
562:       XZ=1.
563:       DO 20 K=1,60
564:       E=E+1./K
565:       XT=E-0.577216649-DLOG(DP)
566:       XZ=XZ*(X**2/4.)/K**2
567:       TZ=XT*XZ
568:       CO=FK0/10.**10
569:       IF(DABS(TZ)-DABS(CO))30,10,10
570:    10 FK0=FK0+TZ
571:    20 CONTINUE
572:       WRITE(6,100)
573:   100 FORMAT(1X,'ERROR..SERIES BE0 IS NOT FINISHED')
574:       STOP
575:    30 BE0=FK0
576:       RETURN
577:       END
578:       FUNCTION BE1(DIS,CTER,TINC)
579:C
580:C THIS FUNCTION COMPUTE THE VALUE OF
581:C THE MODIFIED BESSEL FUNCTION SECOND KIND ORDER 1
582:C
583:       DOUBLE PRECISION DP,E,XZ,XT,TZ,FK1,CO
584:       X=DIS/SQRT(CTER*TINC)
585:       IF(X-4.)5,3,3
586:     3 DP=X
```

```
587:       FK1=(1.253314*DEXP(-DP)/SQRT(DP))*(1.+0.375/DP-0.1171875/DP**2+0.1
588:      1025390625/DP**3-0.1441955566/DP**4)
589:       GO TO 70
590:     5 DP=X/2.
591:       FK1=DLOG(DP)+0.577216649-0.5
592:       E=0.
593:       XZ=1.
594:       DO 20 I=1,60
595:       E=E+1./I
596:       XT=DLOG(DP)+0.577216649-E-1./(2.*(I+1.))
597:       XZ=XZ*(X**2/4.)/(I*(I+1.))
598:       TZ=XT*XZ
599:       CO=FK1/10.**10
600:       IF(DABS(TZ)-DABS(CO))60,40,40
601:    40 FK1=FK1+TZ
602:    20 CONTINUE
603:       WRITE(6,300)
604:   300 FORMAT(1X,'ERROR..SERIES BE1 IS NOT FINISHED')
605:       STOP
606:    60 FK1=FK1*X/2+1./X
607:    70 BE1=FK1
608:       RETURN
609:       END
610:       FUNCTION STO(DIS,CTER,TINC)
611:C
612:C THIS FUNCTION COMPUTE THE VALUE OF
613:C THE MODIFIED STRUVE FUNCTION ORDER 0
614:C
615:       DOUBLE PRECISION A,XZ,FK0,XA,XT,TZ,CO
616:       A=DIS/SQRT(CTER*TINC)
617:       XZ=1.5D0
618:       FK0=(1./DGAMMA(XZ)**2)*A/2.
619:       XA=1.
620:       DO 20 K=1,60
621:       XT=A/2.
622:       XA=XA*(A**2/4.)/((2*(K+1)-1.)/2.)**2
623:       TZ=XT*XA/0.7854
624:       CO=FK0/10.**10
625:       IF(DABS(TZ)-DABS(CO))30,10,10
626:    10 FK0=FK0+TZ
627:    20 CONTINUE
628:       WRITE(6,300)
629:   300 FORMAT(1X,'ERROR..SERIES ST0 IS NOT FINISHED')
630:       STOP
631:    30 ST0=FK0
632:       RETURN
633:       END
634:       FUNCTION ST1(DIS,CTER,TINC)
635:C
636:C THIS FUNCTION COMPUTE THE VALUE OF
637:C THE MODIFIED STRUVE FUNCTION ORDER -1
638:C
639:       DOUBLE PRECISION A,XZ,FK0,XA,XT,TZ,CO
640:       A=DIS/SQRT(CTER*TINC)
641:       XZ=1.5D0
642:       FK0=(1./(DGAMMA(XZ)*1.77245))+((1./(DGAMMA(XZ)*DGAMMA(XZ+1)))*A**2
643:      1/4.)
644:       XA=1.
645:       DO 20 K=1,60
646:       XT=A**2/4.
647:       XA=XA*(A**2/4.)/(((2*(K+1)-1.)/2.)*((2*(K+2)-1.)/2.))
648:       TZ=XT*XA/1.1781
649:       CO=FK0/10.**10
650:       IF(DABS(TZ)-DABS(CO))30,10,10
651:    10 FK0=FK0+TZ
652:    20 CONTINUE
653:       WRITE(6,300)
654:   300 FORMAT(1X,'ERROR..SERIES ST1 IS NOT FINISHED')
655:       STOP
656:    30 ST1=FK0
657:       RETURN
658:       END
```