1982

The Coupling of Monte Carlo Integration With the Boundary Integral Equation Technique to Solve Poisson Type Equations.

Gary Steven Gipson
Louisiana State University and Agricultural & Mechanical College

Follow this and additional works at: https://digitalcommons.lsu.edu/gradschool_disstheses

Recommended Citation
Gipson, Gary Steven, "The Coupling of Monte Carlo Integration With the Boundary Integral Equation Technique to Solve Poisson Type Equations." (1982). LSU Historical Dissertations and Theses. 3717.
https://digitalcommons.lsu.edu/gradschool_disstheses/3717
INFORMATION TO USERS

This was produced from a copy of a document sent to us for microfilming. While the most advanced technological means to photograph and reproduce this document have been used, the quality is heavily dependent upon the quality of the material submitted.

The following explanation of techniques is provided to help you understand markings or notations which may appear on this reproduction.

1. The sign or “target” for pages apparently lacking from the document photographed is “Missing Page(s)”. If it was possible to obtain the missing page(s) or section, they are spliced into the film along with adjacent pages. This may have necessitated cutting through an image and duplicating adjacent pages to assure you of complete continuity.

2. When an image on the film is obliterated with a round black mark it is an indication that the film inspector noticed either blurred copy because of movement during exposure, or duplicate copy. Unless we meant to delete copyrighted materials that should not have been filmed, you will find a good image of the page in the adjacent frame. If copyrighted materials were deleted you will find a target note listing the pages in the adjacent frame.

3. When a map, drawing or chart, etc., is part of the material being photographed the photographer has followed a definite method in “sectioning” the material. It is customary to begin filming at the upper left hand corner of a large sheet and to continue from left to right in equal sections with small overlaps. If necessary, sectioning is continued again—beginning below the first row and continuing on until complete.

4. For any illustrations that cannot be reproduced satisfactorily by xerography, photographic prints can be purchased at additional cost and tipped into your xerographic copy. Requests can be made to our Dissertations Customer Services Department.

5. Some pages in any document may have indistinct print. In all cases we have filmed the best available copy.
Gipson, Gary Steven

THE COUPLING OF MONTE CARLO INTEGRATION WITH THE
BOUNDARY INTEGRAL EQUATION TECHNIQUE TO SOLVE POISSON
TYPE EQUATIONS

The Louisiana State University and Agricultural and Mechanical Col. PH.D. 1982

University Microfilms International 300 N. Zeeb Road, Ann Arbor, MI 48106

Copyright 1982 by
Gipson, Gary Steven
All Rights Reserved
THE COUPLING OF MONTE CARLO INTEGRATION WITH
THE BOUNDARY INTEGRAL EQUATION TECHNIQUE
TO SOLVE POISSON TYPE EQUATIONS

A Dissertation

Submitted to the Graduate Faculty of the
Louisiana State University and
Agricultural and Mechanical College
in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

in

The Interdepartmental Program in Engineering Science

by

Gary Steven Gipson
B.S. Louisiana State University, 1975
M.S. Louisiana State University, 1978
May, 1982
ACKNOWLEDGEMENTS

There are many people who have contributed either directly or indirectly to the effort contained in these pages, and I would like to thank them here.

First, I would like to acknowledge the members of my committe, Dr. V.J. Gopu, Dr. V. Cundy, Dr. R.G. Hussey, Dr. W.N. Sharpe, and my major professor Dr. R.L. Thoms, for their vital part in my Ph.D. work. A special thanks goes to Dr. Thoms who suggested the topic presented here, made constructive criticisms, and generally bore the brunt of the seeminglyless problems associated with this particular degree program. He has done everything he could to help me through the past four years, whether it be to arrange much needed financial support, help with important references, or many times simply provide a sympathetic ear. Bob is my conception of what being a university professor is all about, and for this positive influence on my life I will always be grateful.

Also I would like to acknowledge the Institute for Environmental Studies for acting as my home department during my Ph.D. work, and thank the staff there for their help. Three of my friends there deserve special thanks. Judy Young actually deserves a medal more than a thank you. Her willingness to always chip in and help coupled with her corresponding talent for taking care of tedious details makes her one of the most valuable people I have been associated with at LSU. I will miss her greatly when I leave the Institute. I would like to thank Richard Gehle for helping with the 3-D graphics in Chapter VI; also, he has helped tremendously throughout the years by being the other half of one of my late night rap sessions over coffee and
donuts. It's fair to say that some of the better ideas to come out of this work were conceived at Mary Lee Donuts in my conversations with Richard. Also, thanks goes to Dr. Michael G. Waldon for helping me with his knowledge of statistics, and also for providing necessary intermission time with his "interesting" political philosophies. I wish him much success in his future research (of a nonpolitical nature) to make the earth a cleaner place in which to live.

The most special thanks goes to the most special person in my life, my dear wife Cherie. Her beautiful spirit, warmth and dedication have been a constant source of strength and inspiration for me throughout my academic career. Without her by my side there is no way I could have achieved the success that I have. This has been especially evident during the past few weeks when she stayed up nights without sleep in order to type this work. Beezie, I love you more every day, and am happy I have this opportunity to dedicate to you this piece of work which is a result of that love and your confidence in me.

Last, but not least, my heartfelt appreciation goes to the folks on Citizen Band frequencies 27.005, 27.0085, 26.965, and 27.305 MHz for providing me with free entertainment during the long nights over the past year that I spent doing the computer work. Thanks goes to Peon, Scrap and Granny, Cobra Six, Coon Hunter, Mother Goose, Ruth, Freebird, Mississippi Lady, Temporary, Shady Lady, Gator and a host of others for their late night conversation.
TABLE OF CONTENTS

ACKNOWLEDGEMENTS .......................................... ii
LIST OF TABLES .............................................. vii
LIST OF FIGURES ............................................. viii
ABSTRACT ................................................ xi

CHAPTER

I. INTRODUCTION .................................. 1
  Rationale for the Development of Boundary Element Methods .... 2
  Definition of the Problem ........................................ 5

II. THE BOUNDARY INTEGRAL EQUATION METHOD ............ 7
  The Source Approach ......................................... 7
  Weighted Residual Techniques ................................ 10
  Boundary Methods in Poisson Type Problems .................. 19
  Boundary Element Method ..................................... 25
  Discretization ................................................ 30
  Poisson Terms ............................................... 38
  Transient Problems .......................................... 39
  Conclusion ................................................... 41

III. FUNDAMENTAL SOLUTIONS .......................... 43

IV. DEALING WITH NONHOMOGENEOUS TERMS IN BOUNDARY METHODS ....................................................... 55
  Review of the Problem ........................................ 55
  Monte Carlo Methods ......................................... 57
  Application of Monte Carlo to Boundary Methods .......... 70
  Success of the Method ....................................... 73

V. COMPUTER IMPLEMENTATION .......................... 76
  Main Program MAIN .......................................... 77
  Subroutine INPUT ........................................... 77
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>D. COMPUTER LISTING</td>
<td>141</td>
</tr>
<tr>
<td>E. SAMPLE I/O</td>
<td>209</td>
</tr>
<tr>
<td>VITA</td>
<td>219</td>
</tr>
</tbody>
</table>
### LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>IV.1</td>
<td>Probability of the deviation between an estimand and an actual value exceeding a multiple of $\sqrt{V/N}$</td>
<td>64</td>
</tr>
<tr>
<td>VI.1</td>
<td>Exact and computed values for the temperature inside a rectangular plate with unit internal heat generation</td>
<td>93</td>
</tr>
<tr>
<td>VI.2</td>
<td>Finite difference vs. boundary element values of temperature computed inside heating duct insulation</td>
<td>97</td>
</tr>
<tr>
<td>VI.3</td>
<td>Exact and computed values for the temperature inside a square plate with internal heat generation and orthotropic conductivity</td>
<td>99</td>
</tr>
<tr>
<td>VI.4</td>
<td>Exact and computed values for the temperature inside a tube furnace</td>
<td>104</td>
</tr>
<tr>
<td>VI.6</td>
<td>Exact and computed values for the temperature on a unit disk with internal heat generation equal to $Q(r) = \cos \left(\frac{\pi r}{2}\right)$</td>
<td>108</td>
</tr>
<tr>
<td>VI.7</td>
<td>Exact and computed values of the torsion function for a circular shaft containing a radial crack</td>
<td>112</td>
</tr>
</tbody>
</table>
### LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>II-1</td>
<td>Relationship between source and observation points</td>
<td>8</td>
</tr>
<tr>
<td>II-2</td>
<td>Notation for well-posed boundary value problems</td>
<td>20</td>
</tr>
<tr>
<td>II-3</td>
<td>$\varepsilon$-neighborhood of a boundary point needed to construct the boundary form of the governing equation</td>
<td>27</td>
</tr>
<tr>
<td>II-4</td>
<td>Discretization of domain into boundary elements</td>
<td>31</td>
</tr>
<tr>
<td>II-5(a)</td>
<td>Constant boundary elements</td>
<td>34</td>
</tr>
<tr>
<td>II-5(b)</td>
<td>Linear boundary elements</td>
<td>34</td>
</tr>
<tr>
<td>II-6</td>
<td>Notation used for integration purposes</td>
<td>36</td>
</tr>
<tr>
<td>III-1</td>
<td>Relation of a 2-D boundary element to an axisymmetric ring element</td>
<td>50</td>
</tr>
<tr>
<td>III-2</td>
<td>Notation for an orthotropic 2-D body</td>
<td>52</td>
</tr>
<tr>
<td>IV-1</td>
<td>Very general 2-D body</td>
<td>71</td>
</tr>
<tr>
<td>V-1</td>
<td>Macro-flow diagram for computer program</td>
<td>78</td>
</tr>
<tr>
<td>V-2</td>
<td>Outward normal and contour directions for external and internal surfaces</td>
<td>79</td>
</tr>
<tr>
<td>V-3</td>
<td>Numerical integration of boundary elements</td>
<td>82</td>
</tr>
<tr>
<td>VI-1</td>
<td>Boundary element model of one-quarter of a square plate</td>
<td>92</td>
</tr>
<tr>
<td>VI-2(a)</td>
<td>Heating duct</td>
<td>95</td>
</tr>
<tr>
<td>VI-2(b)</td>
<td>Finite difference model and results for heating duct</td>
<td>95</td>
</tr>
<tr>
<td>Figure</td>
<td>Description</td>
<td>Page</td>
</tr>
<tr>
<td>---------</td>
<td>------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>VI-3</td>
<td>Boundary element model of heating duct in Figure VI-2(a)</td>
<td>96</td>
</tr>
<tr>
<td>VI-4</td>
<td>Boundary element model for a square plate with orthotropic material properties</td>
<td>98</td>
</tr>
<tr>
<td>VI-5</td>
<td>Boundary element model for studying potential distribution between two spheres</td>
<td>101</td>
</tr>
<tr>
<td>VI-6</td>
<td>Axisymmetric boundary element model of a hollow cylinder</td>
<td>103</td>
</tr>
<tr>
<td>VI-7</td>
<td>Planar boundary element model of a hollow cylinder</td>
<td>105</td>
</tr>
<tr>
<td>VI-8</td>
<td>Boundary element discretization of one-quarter of a circular plate</td>
<td>109</td>
</tr>
<tr>
<td>VI-9</td>
<td>Boundary element model of a shaft containing a radial crack</td>
<td>111</td>
</tr>
<tr>
<td>VI-10</td>
<td>3-D plots of the torsion function for a shaft with a radial crack</td>
<td>114</td>
</tr>
<tr>
<td>VI-11</td>
<td>3-D plot of the torsion function for a shaft with a radial crack and the corresponding contour plot</td>
<td>115</td>
</tr>
<tr>
<td>VI-12</td>
<td>Circular shaft with an eccentrically drilled hole</td>
<td>116</td>
</tr>
<tr>
<td>VI-13</td>
<td>Boundary element model of circular shaft with eccentric hole</td>
<td>118</td>
</tr>
<tr>
<td>VI-14</td>
<td>Plots of the analytical and computed results for the shaft with an eccentrically-drilled hole</td>
<td>119</td>
</tr>
<tr>
<td>Figure</td>
<td>Page</td>
<td></td>
</tr>
<tr>
<td>--------</td>
<td>------</td>
<td></td>
</tr>
<tr>
<td>B-1</td>
<td>137</td>
<td></td>
</tr>
</tbody>
</table>

Sample element and nodal point numbering system for a boundary element model
ABSTRACT

The primary topic of this work is a method for dealing with boundary element formulations of nonhomogeneous terms in Poisson type equations. A concise presentation of boundary integral techniques and the corresponding fundamental solutions plays an important role in this development. Using Monte Carlo quadrature theory, an algorithm for the construction of a two-dimensional BEM Poisson equation analyzer is derived. A FORTRAN program based upon this algorithm is presented as a novel device that solves the general Poisson equation in two-dimensional and axisymmetric geometries without domain discretization. In the axisymmetric case, Monte Carlo integration is also used to effectively compute the integrals of the singular functions corresponding to some diagonal terms of the assembly matrix. Sample analyses of several engineering problems are performed with the computer program and the results are compared with solutions obtained by other means. The fine quality of the results implies that the program is generally viable for obtaining solutions to the Poisson equation. It is concluded that while the theory is applicable to transient analyses, the technique is not practical in such cases because of the large amounts of computer time needed to assemble the matrices. It is also concluded that extensions of the theory to general three-dimensional geometries pose no special problems; these are possible by drawing a simple analogy with the two-dimensional algorithm.
CHAPTER I

INTRODUCTION

Due to the advent of high-speed computing equipment, many engineering and scientific problems beyond the realm of analytical solution have been successfully solved. In most cases, the theoretical apparatus necessary to perform the numerical solutions was developed long before the computer was invented. This is particularly true in boundary value problems of the potential type in which complex boundaries and/or boundary conditions may prevent closed-form solutions. Classical mathematics alone is inadequate for many of these situations. However, some outstanding computational products have resulted when applied mathematics has been coupled with the power of the computer.

The present work deals with an old theoretical concept that has only recently been applied in numerical work. Called traditionally by the name "boundary integral equation methods" and more recently by the term "boundary element" techniques (Brebbia, 1978), the subject is receiving much attention in the current literature as an innovative engineering tool. It is a method by which the external surface of a domain is discretized into a series of elements connected together at their nodal points. The values of a desired function inside the domain and on the surface are governed by some relationship that can be cast as a boundary integral equation. The technique produces a system of simultaneous equations whose solution is an approximation to the function that satisfies the governing equation when the function is subjected to
prescribed boundary conditions. While similar to the more conventional finite element and finite difference techniques, the boundary element method (BEM) differs in that a considerable reduction in data preparation and solution time usually results. Success has been reported in the application of BEM in such diverse fields as stress analysis (Cruse, 1973; Rizzo, 1967; Alarcon et al, 1978; Patterson, 1981), electromagnetism (Symm, 1978; Trowbridge, 1978), fracture mechanics (Cruse, 1975, 1978), and geomechanics (Wood, 1981; Dredeleanu, 1981), as well as potential and fluid flow type problems which will be discussed in this work.

Rationale for the Development of Boundary Element Methods

One can reasonably question the need for the development of yet another device to solve boundary value problems. In order to answer the question and define the thesis topic it is necessary to first discuss the present state of the art and make some critical comparisons. As previously stated, numerical techniques must be relied upon to solve problems too complicated to be resolved by analytical means. Despite the myriad of numerical methods that have been developed to accomplish solutions, practical use is usually limited to just two — finite differences and finite elements. The reasons why these two methods are so popular serve to illustrate a point. Finite differences is easy to grasp, simple to program, and has existed long enough to have had its own formalism developed (finite difference calculus). The simplicity of finite differences is such that many problems are solvable with a hand calculator; a computer is often not necessary. Finite element
analysis originated at about the same time the computer became a practical machine (see Turner, 1956, one of the early papers). The power of the method over finite differences eventually became universally recognized (circa 1960) and the FEM has been under constant refinement and expansion ever since. Because the computer is nearly always necessary for a finite element problem, it is not surprising to see the progress of the method, and the complexity of the problems with which it is capable of dealing, parallel the development of the computer. In the author's opinion, the FEM's popularity is largely a result of the timing of its origination and its practical advantages over the only other viable alternative, finite differences. The suitability of finite elements for a great range of problems has been overstated, and is most probably a result of zealous overreaction on the part of finite element researchers. While FEM may be popular, it is very often inefficient, even for some of its more popular applications.

Because finite elements and finite differences are domain-type formulations they suffer from a common awkward unwieldiness when it comes time to actually perform an analysis; that is, for all but the simplest problems, many man-hours are necessary to establish a working model. This is due to the fact that the entire solution domain must be discretized and serially indexed. Not only is the situation unfortunate in circumstances where it is known that the boundary conditions completely define the solution (i.e. potential theory and elastostatics), but it is also unnecessarily expensive. The cost of the time needed to model a region can be considerable.

Most of the effort toward relieving this problem has been applied toward the development of automatic mesh generation software.
However, the efficiency of such programs is often hampered by the cause of the problem in the first place, that being the complex geometrical boundaries and varying boundary conditions. Indeed, one can choose a current copy of almost any engineering journal dedicated to numerical methods and see an "improved" mesh generation technique presented (i.e. the *International Journal of Numerical Methods in Engineering*). The ultimate ambition of authors of such software is probably unattainable since there is no limit to the complexities of possible solution domains in any general FEM formulation.

Whether or not mesh generation techniques are finally perfected, it will always be less efficient, from a modelling standpoint at least, to use a mesh when the mathematical formulation of the problem does not require it. It is with this realization that the boundary integral equation method is presented as a more efficient alternative to FEM in many cases. It should be noted that the BEM will probably never be appropriate for problems involving time-dependent or rapidly varying material laws, or some configuration-dependent non-linear problems because of the unpredictable domain influence (Brebbia, 1980). In these problems finite elements are more suitable. However, for the common problems of elastostatics, potential theory, and some others for which finite elements are highly touted, boundary elements are usually superior.

The reason BEM has not taken the forefront is probably for the same reason finite differences lingers on, tradition. Implementation of the BEM requires a rethinking of the formulation of modern numerical techniques. The mathematics is not the sort normally associated with engineering problems. What is modern about BEM is not its foundations, but the point of view that may be taken toward classical mathematics
in the presence of computers. Rather than discretizing everything at the outset, BEM relates boundary data, proceeds as far as possible analytically, and then introduces approximations that are straightforward and effective (Rizzo, 1975).

**Definition of the Problem**

As alluded to previously, the lack of the need for a mesh is one of the most appealing features of BEM. In some problems, however, this advantage is not realized to its full extent.

To illustrate this, consider the application of BEM to Laplace's equation. It is necessary to reduce the governing equation to quadratures on the boundary solution domain. As will be derived in a later section, Laplace's equation $\nabla^2 u = 0$ in a region $V$ bounded by a surface $S$ has a solution at point $i$ given by:

$$u_i = \int_{S_1} (qu^* - uq^*)ds + \int_{S_2} (qu^* - uq^*)ds .$$  \hspace{1cm} (1.1)

Here, $u = \overline{u}$ on $S_1$ and $q = \frac{\partial u}{\partial n} = \overline{q}$ on $S_2$. The total surface is given by $S = S_1 + S_2$. $u^*$ is the fundamental solution defined by $\nabla^2 u^* = -\delta(r-r')$. The important feature to notice in the above equation for $u_i$ is that the solution is entirely dependent upon the boundary conditions and boundary geometry. This is as it should be since solutions to Laplace's equation are analytic functions (Churchill, 1960). However, if Poisson's equation $\nabla^2 u = b$ is analyzed, another term is added to the right-hand side of Equation I.1 that cannot be cast in surface integral form:
\[ \int_{V} u*b \, dV \]

This integral has been handled in previous work in a rather amusing manner. To quote Brebbia (1978):

"The integration can be performed by dividing the domain into a series of cells or 'elements' similar in shape to those used in the finite element method, but conceptually different."

While he is correct about the conceptual difference, in that there are no internal unknowns associated with the volume integration, the internal cells must be numbered and indexed; that is, the boundary integral method loses its modelling advantage over the domain type techniques. Similar circumstances occur in time-dependent and non-linear problems.

The problem to be studied is the elimination of the need for a grid in some of these non-homogeneous problems involving the Laplacian operator. Also, a general system program incorporating the results is developed and some sample problems demonstrating the capabilities of the program are presented. Extensive use of Monte Carlo integration techniques are used to accomplish these goals.
CHAPTER XI

THE BOUNDARY INTEGRAL EQUATION METHOD

There are at least two distinct formulations of boundary integral equation methods. One is the "source" approach, a highly mathematical theory, that would in fact be more appealing to a theoretician than to one interested in applying the method. Applications based upon this technique do occur in the literature, however, and the theory will be discussed briefly. The second method is derived from weighted residual techniques and is more in harmony with classical engineering analysis. This approach will be discussed in detail starting with weighted residual theory.

The Source Approach

The source approach to BIE theory has its roots in the work of Fredholm (1903) who demonstrated the existence of solutions to equations of the type to be discussed here. Fredholm equations are a direct result of the representation of harmonic potentials by distributions of point or dipole potentials.

Consider a three dimensional unit point source (Figure II-1)

\[ g = \frac{1}{4\pi \left| \mathbf{r} - \mathbf{r}' \right|} \]  

(ii.1)
Figure II-1

The vector relationship between the source and observation points in potential theory.
where $\mathbf{r}'$ locates the source point and $\mathbf{r}$ is the position of the field point. The function $g$ satisfies Laplace's equation $\nabla^2 g = 0$, is continuous, and is differentiable everywhere except at $\mathbf{r} = \mathbf{r}'$. A distribution of sources with strengths $q_1, q_2, \ldots, q_i$ respectively generates a potential

$$U(\mathbf{r}) = \sum_{i=1}^{N} g(\mathbf{r}, \mathbf{r}_i) q_i \quad (II.2)$$

Given continuous values of $U$ over the entire boundary $S$ of a region $V$, Equation (II.2) has the continuum form

$$U(\mathbf{r}) = \int_{S} g(\mathbf{r}, \mathbf{r}') \sigma(\mathbf{r}') d\mathbf{q} \quad (II.3)$$

This defines a Dirichlet problem in terms of the unknown surface charge density $\sigma(\mathbf{r})$. Equation (II.3) is a Fredholm equation of the first kind, and cannot be solved analytically in the general case. However, unique solutions do exist in a well-posed problem (Fredholm, 1903). A numerical approximation to the solution can be obtained by dividing $S$ into discrete elements and assuming that over each element $\sigma$ is a constant. Making appropriate limiting adjustments to Equation (II.3) and writing an equation for each $\sigma_i$ in terms of all the other $\sigma_j$ produces a set of simultaneous equations in the $\sigma_i$'s. The equations can be solved with the help of known values of $U$ on the boundary, and a unique solution set is obtained.
A similar approach is used in Neumann problems and mixed boundary value formulations. There is also a double-layer or dipole source approach which is theoretically more elegant but is cumbersome to apply, and, consequently, rarely seen in practical applications.

The interested reader who desires to know more about the source approach and its applications is referred to the monograph of Jaswon and Symm (1977).

Weighted Residual Techniques

The most instructive and widely propounded theory of boundary elements emanates from an interpretation of the equations as weighted residual approximations. The method of weighted residuals is a general analysis technique that embraces several other approximation devices common to engineers (Crandall, 1956). For instance, suppose it is desired to obtain the solution to the differential equation

\[ y'(x) = x - y \quad \text{for} \quad 0 \leq x \leq 1 \]

knowing that \( y = 0 \) when \( x = 0 \). Suppose further that for some reason the exact solution is beyond our grasp or otherwise too inconvenient to obtain. As a trial function to approximate our solution we may try

\[ y_i = ax^2 + bx^3 \quad \text{(II.4)} \]

where \( a \) and \( b \) are undetermined. Note that this is a truncated form of the complete function

\[ y = \sum_{n=0}^{N} a_n x^n \]

and that the boundary condition
at $x = 0$ is satisfied. Substitution of Equation (II.4) into the differential equation yields an equation for the "residual" $R$:

$$R = \frac{dy}{dx} + y - x$$

$$= a(2x + x^2) + b(3x^2 + x^3) - x \tag{II.5}$$

If the trial function was exact, $R$ would be zero, but the best that can be hoped for is that $R$ approximate zero in the desired range. There are several common ways to obtain approximate solutions, and hence several ways to proceed from here.

We might try collocation where it is assumed that $R$ does not vary considerably in between points where it vanishes. Picking a number of collocation points that correspond to the number of undetermined coefficients achieves an approximate solution. Arbitrarily picking $x = 0.5$ and $x = 1.0$ gives

$$R(0.5) = 1.25a + 0.875b - 0.5$$

$$R(1.0) = 3.0a + 4.0b - 1.0$$

and consequently,

$$y_1 = 0.474x^2 - 0.105x^3 \tag{II.6}$$

Another way to approximate the solution is through the method of moments. This involves a weighting function which is normally a linear combination of a linearly independent set of functions. For this example we will use $1, x, x^2 \ldots$ and demand that
\[
\int_0^1 R \cdot 1 \, dx = 0
\]
\[
\int_0^1 R \cdot x \, dx = 0
\]

In general there will be one moment equation for each undetermined coefficient. Here we get

\[
1.333a + 1.25b - 0.5 = 0,
\]
\[
0.9166a + 0.95b - 0.333 \cdot 0 = 0,
\]
yielding

\[
y_2 = 0.483x^2 - 0.115x^3 \tag{II.7}
\]

If the weighting functions are taken to be from the same family of functions of \( x \) that our trial function was constructed from, Galerkin's method results (Duncan, 1938). The idea here is that the weighted averages of the residual vanish. The criterion in our example is

\[
\int_0^1 R \cdot x^2 \, dx = 0
\]
\[
\int_0^1 R \cdot x^3 \, dx = 0
\]

Performing the integrations and solving the simultaneous equations gives
\[ y_3 = 0.475x^2 - 0.107x^3 \] \hspace{1cm} (II.8)

Also, there is the subdomain method (Biezeno and Koch, 1923) in which the desired interval is broken into as many subdomains as there are adjustable coefficients and then the average residual in each subdomain is forced to be zero. In this example we might set

\[ \frac{1}{h} \int_0^h R \, dx = 0 \]

and

\[ \frac{1}{h} \int_0^h R \, dx = 0 \]

This would yield

\[ y_4 = 0.485x^2 - 0.118x^3 \] \hspace{1cm} (II.9)

As a last approximation, the method of least squares can be used. This technique minimizes the integral of the square of the residual over the interval. The normal equations are

\[ \frac{1}{2} \frac{\partial}{\partial a} \int_0^1 R^2 \, dx = 0 \]

and

\[ \frac{1}{2} \frac{\partial}{\partial b} \int_0^1 R^2 \, dx = 0 \]
The resulting approximation is

\[ y_5 = 0.478x^2 - 0.110x^3 \]  \hspace{1cm} (II.10)

Five different approximations to the same function have just been obtained. However, the techniques of approximation are remarkably similar. In every case there is a set of simultaneous equations to solve; the main difference between the methods being the amount of effort required to obtain the equations.

All five of the techniques can be considered as special cases of the general criterion that the weighted averages of the residual should vanish. This is the basis for the weighted residual technique. In the example just considered, all the criteria can be reduced to the two statements

\[
\int_0^1 W_1 R \, dx = 0 \\
\int_0^1 W_2 R \, dx = 0,
\]

where the \( W \)'s are general weighting functions. In our example, the \( W \)'s necessary to achieve the approximations of Equations (II.6-10) are as follows:

1. Collocation \( W_1 = \delta(x - 0.5) \)  
\[ W_2 = \delta(x - 1.0) \]
2. Moments - \( W_1 = 1 \)
\( W_2 = x \)

3. Galerkin - \( W_1 = x^2 \)
\( W_2 = x^3 \)

4. Subdomain - \( W_1 = H(x - 0.5) - H(0.0) \)
\( W_2 = H(x - 1.0) - H(x - 0.5) \)

5. Least Squares - \( W_1 = \frac{\partial R}{\partial a} = 2x + x^2 \)
\( W_2 = \frac{\partial R}{\partial b} = 3x^2 + x^3 \)

where \( \delta \) indicates the Dirac delta and \( H \) is the Heaviside step function.

(It might be said that the weighted residual technique is to numerical approximation as virtual work is to energy methods.)

With the weighted residual method, it is possible to derive the numerical techniques discussed in Chapter I, and see the common roots of finite differences, finite elements, and boundary elements. Suppose we have a function \( u \) which approximates the solution to \( \nabla^2 \phi = 0 \) inside a domain \( V \). Then substituting the function \( u \) for \( \phi \) yields

\[
\nabla^2 u = R
\]  \text{ (II.11)}

where \( R \) is the residual or error. The error can be distributed in accordance with a weighting function \( W \) by integrating both sides of Equation (II.11) over the domain; i.e.

\[
\int_V (\nabla^2 u) W \, dv = \int_V RW \, dv \quad . \text{ (II.12)}
\]
If the function $W$ is picked to be

$$W = a_1 \delta_1 + a_2 \delta_2 + \ldots + a_N \delta_N$$

where $\delta_i$ is the Dirac delta centered at point $i$, and we apply the weighted residual criterion, i.e., the right hand side of Equation (II.12) must vanish; the result is $N$ equations of the form

$$\int_V (\nabla^2 u) \delta_i \, dV = 0$$

or $$\nabla^2 u_i = 0.$$

This is immediately recognized as the finite difference approximation.

Finite elements comes from weighted residuals by means of a more sophisticated definition of the residual. Again assume that $\nabla^2 u = R$ in the domain, but also that some of the boundary conditions possibly cannot be accurately accounted for. Define

$$u = \bar{u} \text{ on } S_1$$

and $$\frac{\partial u}{\partial n} = \bar{q} \text{ on } S_2$$

as the exact essential and natural boundary conditions, respectively. $S = S_1 + S_2$ is the entire surface of the domain. If these cannot be satisfied exactly, we have
\[ u - \bar{u} = R_1 \text{ on } S_1 \]

and
\[ q - \bar{q} = R_2 \text{ on } S_2 \]

In a so-called "weak" formulation of the weighted residual statement, the FEM follows from taking \( R_1 \) to be zero, but assuming that the distribution of error in approximating \( u \) can be weighted in the same manner as the error \( R_2 \) on the boundary \( S_2 \). This gives

\[
\int_V R_w \, dV - \int_{S_2} R_2 w \, dS = 0 \quad \text{(II.13)}
\]

as the weak weighted residual statement. This formulation requires approximating functions \( u \) that satisfy the boundary conditions on \( S_1 \) and weighting functions \( w \) that satisfy the homogeneous conditions on \( S_2 \).

Explicitly, Equation (II.13) is

\[
\int_V (\nabla^2 u) w \, dV = \int_{S_2} (q - \bar{q}) w \, dS \quad . \quad \text{(II.14)}
\]

We now apply a multidimensional integration by parts on the left-hand side of Equation (II.14). This procedure is common in the boundary element method and the details of the procedure by which it is done are in Appendix A. Here, just the result of that discussion is presented:

\[
\int_V (\nabla^2 u) w \, dV = \int_S q w \, dS - \int_V \frac{\partial u}{\partial x_i} \frac{\partial w}{\partial x_i} \, dV \quad \text{(II.15)}
\]
where it is assumed that the \( w \) function is well-behaved enough to be

differentiated, and the Einstein summation convention is invoked.

This yields the governing relationship

\[
\int_V \frac{\partial u}{\partial x_i} \frac{\partial w}{\partial x_i} \, dV = \int_{S_2} q w \, dS .
\]  \quad (II.16)

Notice that this equation can be used (by serendipity) by only assuming

that \( u \) and \( w \) are once differentiable, although \( u \) must be twice differ­

entiable in order to get to this stage. If a mesh of nodal points and

elements is created to simulate the domain \( V \), the \( u \) function is given by

\[
u = u_1 N_1 + u_2 N_2 + \ldots + u_N N_N
\]

where the \( u_i \) are the unknowns and the \( N_i \) are shape functions. The

original weighting function is interpreted as a linear combination of

the virtual increments of \( u \). Explicitly,

\[
w = N_1 \delta u_1 + N_2 \delta u_2 + \ldots + N_N \delta u_N
\]

\[
= \delta u .
\]

With these definitions Equation (II.16) takes on the familiar form

\[
\int_V \frac{\partial u}{\partial x_i} \frac{\partial \delta u}{\partial x_i} \, dV = \int_{S_2} q \delta u \, dS
\]
which is usually written in matrix form (Zienkiewicz, 1977) as

\[
\{\delta u\}^T [K] \{u\} = \{\delta u\}^T [q],
\]

or simply

\[
[K] \{u\} = \{q\}.
\]

These are only a few of the interesting results that can be obtained from the theory of weighted residuals. For a definitive work on the subject, Finlayson's text (1972) is recommended.

With the framework laid down in this section, the governing equations for the boundary integral technique can now be derived.

Boundary Methods in Poisson-Type Problems

In the previous section, the weighted residual technique was shown to be a very general principle with a broad range of applications. In this section, the method will be used to derive the governing equations for boundary elements in the context of Poisson-type problems of heat conduction, groundwater flow, etc.

Referring to Figure II-2, consider the following governing equation in a domain \( V \),

\[
\nabla^2 u_c = b \tag{II.17}
\]

where \( b \) is some known function of the coordinates, but for the immediate discussion is time independent. The boundary conditions
General well-posed boundary value problem for Laplacian-Poisson type problems of the form
\[ \nabla^2 u = b \]

Figure II-2
on the problem are of two types:

\[ u^e = \bar{u} \text{ on } S_1 \text{ (forced)} \]

and

\[ q^e = \frac{\partial u^e}{\partial n} = \bar{q} \text{ on } S_2 \text{ (natural)}. \]

The partial with respect to \( n \) denotes the directional derivative with respect to the outward normal \( \bar{n} \). We assume that \( u^e \) can be approximated by another function \( u \). As in the last section, residuals are defined by the use of this approximate function in the following manner:

\[
\nabla^2 u - b = R \\
\quad u - \bar{u} = R_1 \\
\quad \text{and} \quad q - \bar{q} = R_2
\]

Now, we introduce a weighting function \( u^* \) and postulate that it has continuous first derivatives in order to justify forthcoming operations. The weighted residual statement is cast as

\[
\int_V Ru^* \, dV = \int_{S_2} R_2 u^* \, dS - \int_{S_1} R_1 q^* \, dS \tag{II.18}
\]

where \( q^* \) is the normal derivative of \( u^* \). Notice here that all the residuals are assumed to be non-zero at the outset. The seemingly arbitrary way in which the last term was constructed can be justified
in different ways, none of which are particularly rigorous. It is simplest to accept that this is an approximation method and justify the validity of the approximation by the end result. This requires accepting the presence of $q^*$ on dimensional grounds and proving the minus sign preceding the last integral is proper from integration by parts. Equation (II.18) is explicitly

$$\int (V^2 u - b) u^* \, dV = \int_{S_2} (q - \bar{q}) u^* \, dS - \int_{S_1} (u - \bar{u}) q^* \, dS. \quad (II.19)$$

With the help of multidimensional integration by parts (see Equation (II.15) and Appendix A), the left hand side of Equation (II.19) becomes (the sum is implied)

$$-\int_{V} bu^* \, dV + \int_{S} qu^* \, dS - \int_{V} \frac{\partial u}{\partial x_i} \frac{\partial u^*}{\partial x_i} \, dV,$$

and upon simplification, that equation is

$$\int_{V} \frac{\partial u}{\partial x_i} \frac{\partial u^*}{\partial x_i} \, dV + \int_{V} bu^* \, dV = \int_{S_1} qu^* \, dS + \int_{S_2} \bar{q} u^* \, dS + \int_{S_1} u q^* \, dS$$

$$-\int_{S_1} \bar{u} q^* \, dS. \quad (II.20)$$
Integrating by parts again yields an interesting statement:

\[ \int (\nabla^2 u^*) u \, dV - \int b u^* \, dV = -\int q u^* \, dS - \int q u^* + \int u q^* \, dS \]

\[ + \int \tilde{u} q^* \, dS \quad . \]  

(II.21)

The original expression, Equation (II.19) has been transformed into the so-called "inverse" problem (Brebbia, 1980), Equation (II.21). It is remarkable in that nothing has been assumed about \( u^* \) except the minimum necessary to perform the two integrations by parts; that is, that \( u^* \) be twice differentiable. Also note the character of Equation (II.21). The domain integral involving the non-homogeneous integrand \( b u^* \) cannot be worked with further. However, in the case of Laplace's equation, \( b = 0 \), and the term disappears. Now, if a function \( u^* \) can be found such that \( \nabla^2 u^* = 0 \), the possibility of a totally boundary integral formulation exists. The function must exist for two reasons. First, it is known from the theory of analytic functions that the boundary values of such a function define its value everywhere (Churchill, 1960). Secondly, one of the elementary techniques described in the preceding section can be used to find trial functions for \( u \) (by hypothesis, \( u \) exists) and the Galerkin method would produce suitable values of \( u^* \).

In a Galerkin formulation, assuming \( u^* \) has been found such that \( \nabla^2 u^* = 0 \), we might set \( u^* = \delta u \) as defined in the last section and
obtain from Equation (II.21)

$$\int_V b \delta u \, dV + \int_{S_2} u \frac{\partial u}{\partial n} \, dS + \int_{S_1} \frac{\partial u}{\partial n} \, dS = \int_{S_1} q \delta u \, dS + \int_{S_2} q \delta u \, dS \quad . \quad (\text{II.22})$$

Noting the convenient coupling of known and unknown values of the function and the flux on the boundary allows us to write this as:

$$\int_V b \delta u \, dV + \int_S u \delta q \, dS = \int_S q \delta u \, dS \quad (\text{II.23})$$

in which $\delta q = \frac{\partial u}{\partial n}$; $u = \bar{u}$ on $S_1$, and $q = \bar{q}$ on $S_2$ is implied.

As a more rigorous way to achieve Equation (II.23) and as a device to justify the previous procedure, Green's theorem can be used. It states that

$$\int_V (u \nabla^2 \delta u - \delta u \nabla^2 u) \, dV = \int_S \left( q \delta u - u \delta q \right) \, dS \quad .$$

Since $\nabla^2 \delta u = 0$ and $\nabla^2 u = b$, this gives

$$\int_V b \delta u \, dV = \int_S u \delta q \, dS - \int_S q \delta u \, dS$$

which is identical to Equation (II.23).
Equation (11.21) represents the traditional starting point for the boundary element method. This is the subject of the next section.

**Boundary Element Method**

The term "boundary elements" was invented in the Civil Engineering Department of the University of Southampton in Great Britain (Brebbia, 1978). The expression applies to the result of a particular type of weighting function being applied in Equation (11.21). It is the function defined by the equation

\[ \nabla^2 u^* = -\delta(\mathbf{r} - \mathbf{x}_1) \]  

where \( u^* \) is the potential in an infinite region at a point \( \mathbf{r} \) due to a source at point \( \mathbf{x}_1 \). For a two-dimensional isotropic medium,
\[ u^* = \frac{1}{2\pi} \ln |\vec{x} - \vec{x}_1| \]

In three-dimensions, this is

\[ u^* = \frac{1}{4\pi |\vec{x} - \vec{x}_1|} \]

The reason for the choice of \( u^* \) is the effect that it has on Equation (11.21). Substituting for \( \nabla^2 u^* \) as defined by Equation (11.24) yields:

\[ u + \int_{V} bu^* dV + \int_{S_2} uq^* dS + \int_{S_1} \overline{uq}^* dS = \int_{S_2} \overline{qu}^* dS + \int_{S_1} qu^* dS \quad (11.25) \]

The potential at any point "i" inside the domain is related to the boundary values of the function and flux, and to a domain integral involving the nonhomogeneous term. To make Equation (11.25) entirely dependent upon boundary values of \( u \), a limiting procedure must be used in taking point 'i' to the boundary. There are several ways in which the limiting procedure can be performed. A simple expedient for the task is to consider the boundary to be made up of the two types corresponding to \( S_1 \) and \( S_2 \). Considering the two-dimensional case for simplicity (Figure II-3), a small circular sector of radius \( \varepsilon \) can be constructed around a boundary point 'i'. The surrounding boundary on either side of the point is assumed to be flat. Considering the boundary point to be of type \( S_2 \), the following statement can be written:
Figure II-3

$\varepsilon$-neighborhood of a boundary point needed to construct the boundary form of the governing equation
\[ \int_{S_2} uq^* \, dS = \int_{S_{(2-\varepsilon)}} uq^* \, dS + \int_{S_\varepsilon} uq^* \, dS . \]  

On the \( \varepsilon \)-circle \( q^* = -1/(2\pi\varepsilon) \). Thus

\[ \int_{S_\varepsilon} uq^* \, dS = \varepsilon \int_{0}^{\frac{\pi}{2\varepsilon}} \frac{-u}{2\pi} \, d\theta = \frac{-au}{2\pi} . \]  

Now taking the limit as \( \varepsilon \to 0 \), \( S_{(2-\varepsilon)} \to S_2 \); the point 'i' is forced to the boundary on the right-hand side and the form of the left-hand side, taken to the boundary is found. The presence of the term found in Equation (II.27) represents the result of an integration through the singularity in the sense of a Cauchy principal value. For the \( S_1 \) portion of the boundary, the formulation is the same,

\[ \int_{S_1} qu^* \, dS = \int_{S_1-\varepsilon} qu^* \, dS + \int_{S_\varepsilon} qu^* \, dS . \]

On \( S_\varepsilon \),

\[ \int_{S_\varepsilon} qu^* \, dS = \frac{-\varepsilon}{2\pi} \int_{0}^{\frac{\pi}{\varepsilon}} q \ln \varepsilon \, d\theta . \]

which clearly vanishes as \( \varepsilon \to 0 \). Similar results can be obtained for three-dimensional problems (Brebbia, 1980). Incorporating Equations
(II.26) and (II.27) into Equation (II.25) yields the result:

\[ c_i u_i + \int \frac{1}{V} \left( \int_{S_2} u_{q*} dS + \int_{S_1} u_{q*} dS \right) = \int_{S_2} q_{u*} dS + \int_{S_1} q dS, \quad (II.28) \]

where \( c_i = 1 - (\alpha/2\pi) \) in a two-dimensional problem. The value of \( c_i \) has an interesting physical interpretation if we take a less restrictive view of the concept of the point source. That is, if the point source is taken as not occupying a mathematical point, but rather a very small, bounded neighborhood of that point, \( c_i \) represents the fraction of that volume of the source occupying the inside of the domain to the total volume of the source. This interpretation applies in any number of dimensions.

For more complicated surfaces, \( c_i \) cannot be computed easily with this method (Hartmann, 1980). However, as will be shown later, it is not necessary to explicitly calculate these terms in potential problems.

Equation (II.28) is typically written in brief form as:

\[ c_i u_i + \int \frac{1}{V} \left( \int_{S} u_{q*} dS \right) = \int_{S} q_{u*} dS \quad (II.29) \]

where the boundary conditions \( u = \bar{u} \) and \( q = \bar{q} \) are to be applied on the appropriate portions of \( S \). Equation (II.29) is a formula that can be used for domain or boundary points. It is the device that allows the development of a matrix formulation for numerical work.
Consider for the moment the case where \( b = 0 \) in Equation (11.29). This reduces the governing equation to a form that only involves boundary integrals. In such a case one would discretize the boundary into 'N' elements (Figure II-4) and this is all that would be necessary. As in finite element theory, boundary elements can have different shape functions, the result being different degrees of approximation for the boundary conditions and for the geometry of a body.

In the simplest case, each element would have one node associated with it, typically at the midpoint of the element. Equation (II.29) would take on the discrete form

\[
\frac{1}{2} u_i + \sum_{j=1}^{N} \int_{S_j} u q^* \, dS = \sum_{j=1}^{N} \int_{S_j} q u^* \, dS \quad (11.30)
\]

for a particular node 'i'. Note \( c_i = \frac{1}{2} \) for every node since each node lies in the middle of an element and, hence, on a flat surface. Because 'u' and 'q' are assumed to be constant over an element, they can be factored out of the integrals. Equation (II.30) evolves into

\[
\frac{1}{2} u_i + \sum_{j=1}^{N} u_j \left( \int_{S_j} q^* \, dS \right) = \sum_{j=1}^{N} q_j \left( \int_{S_j} u^* \, dS \right) \quad .
\]

Each of the terms in parentheses relates the source node 'i' with the object node 'j'. This means that there are two \( N \times N \) matrices
\[ \frac{1}{2} u_i + \sum_{j=1}^{N} u_j \int_{S_j} q^* dS = \sum_{j=1}^{N} q_j \int_{S_j} u^* dS \]

**WHERE**

\[ H_{ij} = \hat{H}_{ij} \text{ when } i \neq j \]

\[ H_{ij} = \hat{H}_{ij} + \frac{1}{2} \text{ when } i = j \]

**ABOVE EQUATION BECOMES**

\[ \sum_{j=1}^{N} H_{ij} u_j = \sum_{j=1}^{N} G_{ij} q_j \]

**Figure II-4**

Discretization of boundary into elements and the corresponding constant element equations for Laplace's equation.
involved in the formulation. They are defined by

\[ H_{ij} = \int_S q^*(\mathbf{x}_i, \mathbf{x}) \, dS + \frac{1}{2} \delta_{ij} \]

and

\[ G_{ij} = \int_S u^*(\mathbf{x}_i, \mathbf{x}) \, dS \]

where \( \mathbf{x} \) locates the differential element \( dS \). \( \delta_{ij} \) is the ordinary Kronecker delta. With these definitions we have a set of equations that can be written as:

\[
\sum_{j=1}^{N} H_{ij} u_j = \sum_{j=1}^{N} G_{ij} q_j \quad (II.32)
\]

or

\[ [H]\{u\} = [G]\{q\} \]

In a well-posed problem, there are \( N_1 \) values of \( u \) and \( N_2 \) values of \( q \) specified on the boundary. A known value of \( u \) is paired with an unknown value of \( q \) in such a way that the number of nodes \( N = N_1 + N_2 \). This being the case, the equations can be reordered into the system

\[ [A]\{y\} = \{F\} \quad (II.33) \]

where \( \{y\} \) contains all the unknowns and \( \{F\} \) holds known values.

Once the equations are solved, values of \( u \) on the interior of the
domain can be computed from the discrete form of Equation (11.25); i.e.,

\[ u_i^I = \sum_{j=1}^{N} G_{ij} q_j - \sum_{j=1}^{N} H_{ij} u_j^I . \]  

where the superscript I's denote the source point taken from inside the domain. The internal fluxes are obtained by differentiation and discretization. Experience has shown that a standard Gaussian quadrature is sufficient for the evaluation of the \( H_{ij} \) and \( G_{ij} \) for any element except the one containing the node 'i' (Wrobel and Brebbia, 1981). The integrand of \( G_{ii} \) is singular, but integrable. These terms are normally computed manually; i.e., for the two-dimensional isotropic case

\[ G_{ii} = \frac{\ell}{2\pi} \left[ 1 - \ln(\ell/2) \right] \]

where \( \ell \) is the length of the element. In a constant element \( H_{ii} = 0 \) because the surface element and unit normal are orthogonal.

The variation of \( u \) and \( q \) is obviously better represented by a linear function on each element (see Figure II-5 for an aesthetic comparison). The nodes are now at the interelement connections.

Reverting back to Equation (11.29), we write the equivalent of Equation (11.30) for linear elements:

\[ c_1 u_i + \sum_{j=1}^{N} \int_{S_j} uq^* \, ds = \sum_{j=1}^{N} \int_{S_j} qu^* \, ds . \] 

\[ (II.35) \]
Figure II-5
Boundary element discretizations into (a) constant & (b) linear elements
Now, however, 'u' and 'q' cannot be factored out of the integral as before because they are no longer assumed as constant over the element. To continue, some preprocessing of 'u' and 'q' must be done. Referring to Figure II-6, we can define shape functions \( N_A \) and \( N_B \) in terms of the dimensionless variable \( s = x/\ell \) and use them to define 'u' and 'q'. Just as in finite elements define

\[
N_A = s \quad \text{and} \quad N_B = 1 - s;
\]

in terms of the unknowns at the endnodes \( u_A \) and \( u_B \),

\[
u(s) = N_A u_A + N_B u_B
\]

and

\[
q(s) = N_A q_A + N_B q_B
\]

With these the \( G_{ij} \) and \( H_{ij} \) each have two components. For example,

\[
\int_{S_j} q u^* \, ds = q_A \int_{S_j} N_A u^* \, ds + q_B \int_{S_j} N_B u^* \, ds
\]

where \( q_A \) and \( q_B \) are constants for the integration. Define the integrals as

\[
G_{iA} = \int_{S_j} N_A u^* \, ds
\]
SHAPE FUNCTIONS:

\[ N_A = S \]
\[ N_B = 1 - S \]

LINEAR ELEMENT NOTATION

Figure II-6
Notation used for integration purposes
and
\[ g_{iB} = \int_{S_j} N_B u^* \, dS \]

Since each nodal point is node A of one element, say element \( j \), and node B of the preceding adjacent element, \( j - 1 \), each \( G_{ij} \) is composed the following way:

\[ G_{ij} = g_{iA} + g_{iB} \]

The formulation for \( H_{ij} \) is analogous.

\[ h_{iA} = \int_{S_j} N_A q^* \, dS \]

and
\[ h_{iB} = \int_{S_j} N_B q^* \, dS \]

The set of \( N \) equations can be written as before in the constant element case

\[ [H][U] = [G][Q] \quad (II.36) \]

where the \( c_{i,j} \) term is absorbed into the left-hand side of the equation.

The diagonal terms of \( H_{ii} \) (including the \( c_{i,i} \) ) can be computed explicitly but there is a simpler way using constant potential considerations. Knowing that a uniform potential applied on the boundary will
cause a zero flux at the boundary can be used to advantage. The form of $H_{ij}$ does not depend on the boundary conditions, so we can apply a unit potential to obtain

$$[H] \{1\} = \{0\} .$$

This equation dictates that the sum of the terms in any row of $H$ are zero. Therefore the diagonal term can be computed from the off-diagonal terms with the formula

$$H_{ii} = - \sum_{j \neq i} H_{ij} .$$  \hspace{1cm} (II.37)

It is possible to define even higher order elements. The reader is referred to the text of Brebbia and Walker (1980) for further discussion of this subject.

**Poisson Terms**

Everything that has been derived in the previous section is directly applicable to the original Poisson equation and the integral representation of Equation (II.29). To perform the Poisson analysis, another term must be added to Equation (II.36) that accounts for the volume integral in Equation (II.29). Defining

$$B_i = \int_V b u^*(\vec{x}_i, \vec{x}) \, dV, \hspace{1cm} (II.38)$$
Equation (II.36) with the Poisson term is

$$\{B\} + [K]\{U\} = [G]\{Q\}$$

Values of $u$ at internal points are given by

$$u_i^I = \sum_{j=1}^{N} (G_{ij} q_j^I - H_{ij} u_j^I) - B_i^I$$

In order to obtain the $B_i$ terms, some sort of quadrature over the domain $V$ must be done. In past work (i.e., Brebbia and Walker, 1980) domain discretization such as that done in finite elements was resorted to. While this gives accurate results, it is expensive in terms of modelling time. One can reasonably argue that a more common method, finite elements, (which has the advantage in that the assembly matrix is symmetric and positive definite) can be used just as effectively. The possibility of eliminating the domain discretization in BEM is the subject of Chapter IV.

**Transient Problems**

The boundary element method can be applied to transient problems such as the diffusion equation

$$\nabla^2 u = b^2 \frac{\partial u}{\partial t}$$  \hspace{1cm} (II.39)

One obvious way to solve this problem is to consider the right-hand side of the equation to be a Poisson term at any instant in time and
use a finite difference time-stepping procedure. This method has been reported to cause error accumulation and to be too time consuming in practice (Brebbia and Walker, 1980).

A more reasonable approach is to use a time dependent fundamental solution. This method follows from the weighted residual statement

\[ t_2 \int \int \left( \nabla^2 u - b^2 \frac{\partial u}{\partial t} \right) u^* \, dV \, dt = \int \int \left( q - q^* \right) u^* \, dS \, dt - \int \int \left( u - u^* \right) q^* \, dS \, dt \]

\[ t_1 \quad t_1 \quad S_2 \quad t_1 \quad S_2 \]

(II.40)

in the same manner as Equation (II.19) yielded the Poisson-type formulation. It is assumed that time begins at \( t = t_1 \). Two integrations by parts give the inverse relationship desired:

\[ t_2 \int \int \left( \nabla^2 u^* + b^2 \frac{\partial u^*}{\partial t} \right) u \, dV \, dt + \int \int \left( qu^* - uq^* \right) \, dS \, dt = \int _V uu^* \, dV \left. \right|_{t=t_1}^{t=t_2} \]

(II.41)

A fundamental solution that solves

\[ \nabla^2 u^* + b^2 \frac{\partial u^*}{\partial t} = \delta(x - x_1) \delta(t - t_1) \]

is called for. This equation is solved in detail for three spatial dimensions in Chapter III. The result can be found in Equation (III.7). With the fundamental solution, the first domain integral is removed and yields in rough terms:
\[ c_1 u_{11} + \int_{t_1}^{t_2} \int_S (u q^* - q u^*) \, dS \, dT = \int_V u u^* \, dV \right] \tau = t_1 \] (II.41)

The remaining temporal integration can be accomplished analytically. However, the form of the matrix equations can be seen at this stage. In the notation of the last Poisson term discussion,

\[ [H] \{U\}_B^{t_2} = [G] \{Q\}_B^{t_2} - [D] \{U\}_B^{t_1} \] (II.42)

The matrix \([D]\) is the result of the domain integral in Equation (II.41). The last equation defines an iteration scheme that can be used to solve for \(u\) and \(q\) as functions of times. The subscripts 'B' and 'I' denote the boundary and internal nodes, respectively. The sizes of \([H]\) and \([G]\) are both \(N_B \times N_B\) while \([D]\) is \(N_B \times (N_B + N_I)\). The size of the matrices is a nontrivial consideration that will be touched upon in Chapter IV again.

Conclusion

A concise discussion of boundary element fundamentals in the context of potential-type problems has been presented in this chapter. It has been shown that the boundary element method has many appealing features over domain type formulations. Among these are the reduction of the dimensions of the problem by one, the implied reduction in input data as a consequence, and the probable reduction in simultaneous equations to solve.
Major disadvantages of the technique are the generally non-symmetric assembly matrices that must be reduced, the size of the problem in a transient case with many nodes, and the need for domain discretization in nonhomogeneous problems.

It is with the boundary element theory presented in this section, and with the awareness of the pros and cons of BEM that a computer program is developed in this work. In a later section, the program will be presented with a new technique of dealing with the nonhomogeneous problem as easily as with its homogeneous counterpart.

The next chapter deals with the all-important technique of finding fundamental solutions. It may be of little interest to some readers familiar with the techniques. If this is the case, the next chapter may be skipped with no loss of continuity.
CHAPTER III

FUNDAMENTAL SOLUTIONS

The construction of fundamental solutions is a necessary peripheral subject one must be familiar with in order to understand and apply the BIE method. The fundamental solution corresponding to an isolated source at the point $\mathbf{r}'$ in infinite space is defined by the equation

$$Lu^* (\mathbf{r}, \mathbf{r}') = -\delta (\mathbf{r} - \mathbf{r}')$$  \hspace{1cm} (III.1)$$

where $L$ is a linear operator with eigenvalues $\lambda (\mathbf{r})$; i.e.

$$Lw = \lambda w$$

Both $u^*$ and $\delta$ can be expanded in terms of eigenfunctions normalized over all space. The set of eigenfunctions will be dependent upon a continuously varying parameter $\mathbf{k}$ and will be denoted here by $w (\mathbf{k}, \mathbf{r})$. $\mathbf{k}$ is a vector in $k$-space with the same number of components as $\mathbf{r}$ in real space. Since the eigenfunctions are orthonormal, we have the inner product relationship (denoted by brackets $\langle, \rangle$)

$$\int w (\mathbf{k}, \mathbf{r}) w (\mathbf{k}', \mathbf{r}) \, d\mathbf{r} = \langle w (\mathbf{k}, \mathbf{r}), w (\mathbf{k}', \mathbf{r}) \rangle$$

$$= \delta (\mathbf{k} - \mathbf{k}')$$  \hspace{1cm} (III.2)$$
In terms of the complete set of eigenfunctions, any bounded function $u^*$ in the space can be written as

$$u^*(\vec{r}, \vec{r}') = \langle w(\vec{k}, \vec{r}), \hat{a}(\vec{k}, \vec{r}') \rangle \quad \text{(III.3)}$$

where the $\hat{a}(\vec{k}, \vec{r}')$ must be determined. The symbol $'$ denotes the compulsory complex conjugation necessary to define the inner product.

With the relationships (II.2) and (III.2) and (III.3), Equation (III.1) becomes

$$\delta(\vec{r} - \vec{r}') = Lu^*(\vec{r}, \vec{r}')$$

$$= L \langle w(\vec{k}, \vec{r}), \hat{a}(\vec{k}, \vec{r}') \rangle$$

$$= \langle Lw(\vec{k}, \vec{r}), \hat{a}(\vec{k}, \vec{r}') \rangle$$

$$= \langle \lambda(\vec{k})w(\vec{k}, \vec{r}), \hat{a}(\vec{k}, \vec{r}') \rangle .$$

Using the eigenfunction expansion of $\delta$, we can say further:

$$\langle w(\vec{k}, \vec{r}), w(\vec{k}, \vec{r}') \rangle = \langle \lambda(\vec{k})w(\vec{k}, \vec{r}), \hat{a}(\vec{k}, \vec{r}') \rangle$$

$$= \langle w(\vec{k}, \vec{r}), \lambda(\vec{k})\hat{a}(\vec{k}, \vec{r}') \rangle .$$

Because this must hold for all $\vec{r}$ and $\vec{r}'$, we can conclude that

$$a(\vec{k}, \vec{r}') = \frac{\hat{a}(\vec{k}, \vec{r}')}{\lambda(\vec{k})}$$

and, hence, Equation (III.3) becomes
Equation (III.4) represents a general formula for determining fundamental solutions. Although there are infinitely many sets of eigenfunctions that span an n-dimensional space, the usual choice for the $w$'s are the waves

$$w(k,x) = \frac{-\exp(ikx)}{2\pi}$$

in the 1-D case and superpositions thereof in higher dimensions. The denominator is a consequence of the normalization procedure. With Equations (III.4) and (III.5), fundamental solutions for the BIE method can be derived. More detailed analysis of Green's functions and explanation of details in the above discussion with which the reader may not be familiar can be found in most advanced applied mathematics texts. Specifically excellent are Byron (1970) and Stakgold (1979).

To apply Equation (III.4) to a practical equation usually requires some effort. For example, the general diffusion or heat transfer problem is defined by

$$\nabla^2 u = \frac{a^2}{b^2} \frac{\partial u}{\partial t}.$$  \hspace{1cm} (III.6)

The eigenfunction is a product of three terms like Equation (III.5) and another similar term in time; i.e.,

$$w(k,x,t) = \frac{-\exp[ik \cdot (\vec{r} - \vec{r}')] \exp[ip(t - t')]}{(2\pi)^4}.$$
The eigenvalues are given by the operation

\[
\left( \nabla^2 - b^2 \frac{\partial^2}{\partial t^2} \right) w = (k^2 + b^2 \pi) w
\]

Now, using Equation (III.4), with the notation \( \vec{s} = \vec{r} - \vec{r}' \), we must perform the four-dimensional integral

\[
u^*(r, r', t, t') = \frac{1}{(2\pi)^4} \int \frac{dk}{k^2 + b^2 \pi} \exp(i\vec{k} \cdot \vec{s}) \exp[ip(t - t')]
\]

In order to do this we will utilize spherical coordinates in \( k \)-space, and for convenience let \( \vec{s} \) be aligned with the \( k_z \) axis. Also, let \( q = \cos \theta \) where \( \theta \) is the angle between \( k \) and \( k_z \). Then:

\[
u^* = \frac{1}{(2\pi)^3 i b^2} \int_0^\infty k^2 dk \int_{-1}^1 d(cos\theta) \int_{-\infty}^{\infty} \frac{\exp[i(k \cdot \vec{s})]\exp[ip(t - t')]}{p - ik^2/b^2} dp
\]

where the azimuthal integration is already done. Continuing,

\[
u^* = \frac{1}{(2\pi)^3 i b^2} \int_0^\infty k \ dk \int_{-\infty}^{\infty} \frac{\exp(iks) - \exp(-iks)}{is(p - ik^2/b^2)} \exp[ip(t - t')] dp
\]

\[
= \frac{-1}{(2\pi)^3 b^2} \int_{-\infty}^{\infty} \frac{k \exp(iks) \ dk}{s} \int_{-\infty}^{\infty} \frac{\exp[ip(t - t')]}{p - ik^2/b^2} dp
\]
Using the residue theorem on the inside integral, we have

$$u^* = \frac{-2\pi i}{(2\pi)^3 b^{2s}} \int_{-\infty}^{\infty} k \exp(iks) \exp[-(k^2/b^2)(t - t')] dk$$

where it is understood that the function makes no sense for $t < t'$.

Now, expanding the exponential and noting that the imaginary portion of the integral must vanish from symmetry; we obtain

$$u^* = \frac{1}{4\pi^2 b^{2s}} \int_{-\infty}^{\infty} k \sin(ks) \exp[-(k^2/b^2)(t - t')] dk$$

$$= \frac{-2}{4\pi^2 b^{2s}} \frac{\partial}{\partial s} \int_{0}^{\infty} \cos(ks) \exp[-(k^2/b^2)(t - t')] dk$$

$$= \frac{-2}{4\pi^2 b^{2s}} \frac{\partial}{\partial s} \left[ \frac{b^{1/2}}{2(t' - t')^{1/2}} [\exp(-s^2/4)(b^2/(t - t'))] \right] .$$

Finally,

$$u^* = \frac{b}{[4\pi(t' - t')]^{3/2}} \exp \left[ \frac{-s^2}{4(t - t')} \right] . \quad \text{(III.7)}$$

In fact, it can be shown (Carslaw and Jaeger, 1959) that this equation is true in n dimensions if the "3" is replaced by "n" in the above.

In less complicated problems a simpler procedure may be convenient if the interpretation of the fundamental solution as a point charge is used. For instance, the two-dimensional Laplace's equation for a point source is obtained from
\[ \frac{1}{r} \frac{d}{dr} \left( r \frac{du^*}{dr} \right) = 0 \]

\[ \Rightarrow u^* = A \ln r + B \]

B represents an arbitrary potential datum and can be set to zero.

To determine A, we can use the fact that the source at \( r = 0 \) is a delta-function type singularity and integrate the small region of space around it. That is, integrating both sides of \( \nabla^2 u^* = \delta(r) \) over an \( \varepsilon \)-circle around the point \( r = 0 \) gives (by the divergence theorem)

\[ A \int_0^{2\pi} \frac{1}{\varepsilon} \cdot \varepsilon \, d\theta = 2\pi A = -1. \]

Therefore,

\[ u^* = \frac{-1}{2\pi} \ln r \quad \text{(III.8)} \]

for the two-dimensional Laplace's equation. The three dimensional Laplacian fundamental solution is obtained by the same procedure (using spherical coordinates) and is:

\[ u^* = \frac{1}{4\pi r} \]

From a general source point 'i',

\[ u^* = \frac{1}{4\pi \left[ (x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2 \right]^{\frac{3}{2}}} \quad \text{(III.9)} \]
The fundamental solutions derived thus far can be used directly in the relations presented in Chapter II. However, from time to time, a situation arises where it is preferable to manipulate the fundamental solution somewhat before using it. A case in point is the analysis of an axisymmetric region which is actually a two-dimensional problem, mathematically speaking, but requires the three-dimensional fundamental solution. In cylindrical coordinates \((r, \theta, z)\), Equation (III.9) appears as

\[
 u^* = \frac{1}{4\pi \left[ r^2 + r_1^2 - 2rr_1 \cos(\theta - \theta_1) + (z - z_1)^2 \right]^2}. \tag{III.10}
\]

In the boundary integral formulation, \(u^*\) denotes a ring element (Figure III-1). Since the solution cannot depend on \(\theta\), a modified fundamental solution can be obtained by doing part of the boundary integral in advance; i.e.,

\[
 u^* = \int_{0}^{2\pi} u^*_{ax} \, d(\theta - \theta_1) \tag{III.11}
\]

where \(u^*\) is dependent upon only \(r\) and \(z\). The fundamental solution \(u^*\) can be used directly in the governing equations of Chapter II. Substituting Equation (III.10) into Equation (III.11) gives

\[
 u^* = \frac{m}{4\pi rr_1} \int_{0}^{\pi/2} \frac{d\phi}{\left[ 1 - m \sin^2 \phi \right]^{1/2}} \tag{III.12}
\]

in which \(\phi = (\theta - \theta_1)/2\) and
Figure III-1

Relationship between two dimensional boundary elements and axisymmetric ring elements
The integral in Equation (III.12) is recognized as an elliptic integral of the first kind \( K(m) \). Therefore:

\[
\frac{4 \pi r_1}{(r + r_1)^2 + (z - z_1)^2} = \frac{K(m)}{\pi [(r + r_1)^2 + (z - z_1)^2]^{1/2}}.
\]  

(III.13)

With this form of \( u^* \), an axisymmetric body can be modelled as easily as a two-dimensional body.

Fundamental solutions for orthotropic problems can be obtained from their isotropic counterparts by using simple coordinate transformations. The two-dimensional diffusion equation with orthotropic material properties is

\[
k_1 \frac{\partial^2 u}{\partial x_1^2} + k_2 \frac{\partial^2 u}{\partial x_2^2} = b^2 \frac{\partial u}{\partial t}.
\]  

(III.14)

Let the axes of orthotropy be \( x_1, x_2 \) (Fig. III.2) and now define new coordinates

\[
\begin{align*}
Y_1 &= x_1 / \sqrt{k_1}, \\
Y_2 &= x_2 / \sqrt{k_2}.
\end{align*}
\]

With this transformation, the Dirac delta function at the field
Figure III-2

Notation for an orthotropic body
point can be written as

\[
\delta(\vec{R} - \vec{R}') = \delta(x_1 - x_1')\delta(x_2 - x_2')
\]

\[
= \delta(\sqrt{k_1}(y_1 - y_1'))\delta(\sqrt{k_2}(y_2 - y_2'))
\]

\[
= \delta(y_1 - y_1')\delta(y_2 - y_2')/\sqrt{k_1k_2}
\]

Hence, Equation (III.10) becomes

\[
\frac{\partial^2 u}{\partial y_1^2} + \frac{\partial^2 u}{\partial y_2^2} = b^2 \frac{\partial u}{\partial t}
\]

and the corresponding form of Equation (III.1) can be written

\[
Lu^* (\vec{x}, \vec{x}'; t, t') = \frac{-\delta(\vec{x} - \vec{x}')\delta(t - t')}{\sqrt{k_1k_2}}
\]

Here,

\[
L = \frac{\partial^2}{\partial y_1^2} + \frac{\partial^2}{\partial y_2^2} - b^2 \frac{\partial}{\partial t}
\]

Thus, the fundamental solution for this operator is immediately deduced from Equation (III.7):

\[
u^* = \frac{b}{4\pi(t - t')\sqrt{k_1k_2}} \exp \left[ -\frac{(x_1^2/k_1) - (x_2^2/k_2)}{4(t - t')} \right]
\]
The same transformation applied to Equation (III.8) yields the D'Arcy fundamental solution

\[ u^* = \frac{-1}{4\pi \sqrt{k_1 k_2}} \ln \left( \frac{x_1^2}{k_1} + \frac{x_2^2}{k_2} \right) \quad (III.15) \]

These techniques illustrate some of the devices available to the practitioner for the development of fundamental solutions. However, it will not always be necessary to formally derive a particular relationship as was done in this work. Most of the Green's functions for the popular linear operators are readily accessible in literature on the physical process the operators define.
In Chapter II, it was shown that the boundary element technique was suitable for solution of several common problems involving Laplacian operators. It was also demonstrated that in nonhomogeneous problems, the governing equations cannot be reduced solely to boundary integrals. This circumstance is unfortunate since many common engineering problems are of the Poisson variety; it would be highly convenient to be able to model these problems as easily as their nonhomogeneous counterparts. In this chapter, a method by which this can be done is presented.

Review of the Problem

In the current literature only one way to solve the general case of

\[ \nabla^2 u = b \]

by boundary elements is presented (i.e. Brebbia, 1980). It requires discretizing the governing relationship, Equation (II.29) (repeated here for convenience):

\[ c_i u_i + \int b u^* \, dv = \int (q u^* - u q^*) \, ds \quad (II.29) \]
on the surface of the region and in the domain. The domain discretization involves no unknowns but still requires considerable time and effort to number and index the domain elements. This drawback reduces the appeal of the method over domain-type formulations.

A review of the literature has shown no efforts to confront the problem of discretization other than the wide variety of mesh generation techniques presented. While automatic generation is suitable for some special problems, its validity for a general domain involving cavities, reentrant corners, and "tight" spots has not been forthcoming. The tactical problems are magnified when the dimensionality of the domain being modeled is increased.

Of course, standard analytical techniques are out of the question, because the Poisson problem would not be a problem at all if one could use classical mathematics to handle the nonhomogeneous terms. Also, standard numerical quadrature formulas cannot be applied to highly irregular domains; if the domains are sub-divided, we are back to the mesh approach and the considerations of the previous paragraph apply.

What is needed is a numerical integration device that is fast, efficient (by comparison with the alternative), controllable, easy to implement, and reasonably accurate; it should be applicable to a wide class of problems, and should not be highly sensitive to dimensionality, complexity of the region, or to the class of functions for which it applies, and most importantly, should not require discretization of the domain. The demands are great, and might even be considered excessive. This is perhaps why the discretization
problem has not been confronted in the past.

In preparation for the assault on this problem, one does well to adopt the attitude (discussed in Chapter I) that is responsible for the development of boundary elements. That is, in light of the computer's ever-increasing power, some methods that were once considered impractical can now be utilized advantageously.

It is with this philosophy that the solution to the discretization problem is attempted by a relatively old technique, Monte Carlo quadrature. The Monte Carlo methods have been used successfully in some practical problems, but this work represents the only attempt (to the author's knowledge) to apply the method to boundary element techniques. Monte Carlo methods and their applications to integration techniques are the subject of the next section.

**Monte Carlo Methods**

Monte Carlo methods comprise a branch of mathematics dealing with experiments on random numbers. The name "Monte Carlo" is taken after the famous casino because of the analog between random numbers and the throwing of dice, drawing of a card hand, etc. Normally, problems handled by these methods are of the probabilistic type because they can be used directly to model some random behavior in a physical process. For instance, a sociologist might wish to use Monte Carlo in order to simulate lifespans of people residing in a particular city. Another application would be to a nuclear reactor, where the random motions of the neutrons could be modelled and their
resultant effects on the performance of the reactor inferred. The utility of random numbers in cases such as these is apparent. Less obvious, however, is why one would resort to a random number technique in a deterministic type problem such as we have in this work. The justification lies in an inherent weakness of an abstract mathematical formula to adequately compensate in practicality for its own generality. That is, the more general a theory, the less simple it is to obtain a numerical solution. The rationale behind applying Monte Carlo is to exploit some apparently non-related random process in order to reduce the aforementioned theoretical weaknesses, while using the strengths of the formula to full advantage. A relevant example of a deterministic problem in which Monte Carlo has been used effectively is in the solution of Laplace's equation; an electromagnetic potential can be determined by guiding charged particles by means of random numbers until they are absorbed by barriers chosen to represent the prescribed boundary conditions (Hammersley and Handscomb, 1964).

The problem at hand is performing numerical quadrature for an arbitrary function over an arbitrary geometrical region. Normally this is done by subdividing the domain into smaller regions and performing a deterministic formula such as Gaussian quadrature. When feasible, such a method usually produces excellent results and the Monte Carlo method is not competitive in any sense. However, if the function fails to be regular, or the region is geometrically very complex, it is often more convenient to use Monte Carlo, especially in multidimensional integrals.
Numerical integration by use of random numbers can be likened to a reasonable game of chance. The expected score of a player, no matter how complex the game, can be estimated by averaging the results of a large number of previous plays. The calculation can often be made more efficient by simulating the game with another one known to have the same expectation value. Monte Carlo technique described in the following is called the "hit or miss" method, and is based upon the interpretation of an integral as an area under a curve. Consider the evaluation of

\[ I = \int_{a}^{b} f(x) \, dx \]

where \( 0 \leq f(x) \leq h \) for \( a \leq x \leq b \). Call the rectangle described by these inequalities \( R \). Let \((X,Y)\) denote a random vector uniformly distributed over \( R \) with probability density

\[ p(x,y) = \frac{1}{h(b-a)} \]

We now proceed with the following algorithm to compute \( I \):

1. A random point \((x,y,) \in (X,Y,)\) is chosen as a trial point.
2. If this point lies beneath the curve \( y = f(x) \), a "hit" is scored. Otherwise it is a "miss".
3. The process above is repeated a large number of times and the proportion of "hits" to the total number of trials is
calculated.

The probability that a point lies beneath \( y = f(x) \) is given by

\[
P = \frac{\text{area under } f(x)}{\text{area of } R} = \frac{\int_a^b f(x) \, dx}{h(b - a)} = \frac{I}{h(b - a)}.
\]

For a sufficiently large number of trials, we get an intuitively obvious result,

\[
P \approx \frac{\text{number of hits}}{\text{number of trials}} = \frac{N_H}{N} = \hat{P};
\]

and therefore, we can estimate the value of the integral by

\[I \approx Ph(b - a).
\]

That is, the probability of landing in the area beneath \( y = g(x) \) is numerically equal to the normalized area. Although presented in a non-rigorous fashion, we can see two interesting qualitative features of Monte Carlo in the example. First, the efficiency of the process does not at all depend on the detailed character of \( f(x) \), but only on a very gross feature of the problem, the total area. Secondly, the practical extension of the theory to any number of dimensions is easy to invoke (Kahn, 1960).
In order to justify the Monte Carlo method on a general class of problems, and place the theory on a rigorous basis we need the following two theorems.

THEOREM (THE STRONG LAW OF LARGE NUMBERS): If a sequence of \( N \) random variables \( U_1 \ldots U_N \) is chosen from a population with probability density function \( p(x) \) and a random variable \( f_N \) defined by the equation

\[
 f_N = \frac{1}{N} \sum_{i=1}^{N} f(x_i)
\]

where \( f(x_i) \) is an integrable function, and if the expectation value

\[
 \bar{f} = \int_{-\infty}^{\infty} f(x) p(x) \, dx
\]

exists, then \( f_N \) will approach \( \bar{f} \) as \( N \to \infty \).

THEOREM (CENTRAL LIMIT THEOREM): For large \( N \) the probability that \( \bar{f} - \delta \leq f_N \leq \bar{f} + \delta \) occurs is asymptotically independent of both \( f(x) \) and \( p(x) \); it depends upon \( N \) and the statistical variance \( V \). In fact,

\[
 \text{Prob}(\bar{f} - \delta \leq f_N \leq \bar{f} + \delta) = \frac{1}{\sqrt{2\pi V}} \int_{-\delta\sqrt{\frac{N}{V}}}^{\delta\sqrt{\frac{N}{V}}} e^{-x^2/2} \, dx + o\left(\frac{1}{\sqrt{N}}\right).
\]

(Here, '\( \delta \)' indicates a small deviation.)
The variance $V$ is defined by

$$\begin{align*}
V &= \frac{(\bar{f} - \bar{f})^2}{\int (\bar{f} - \bar{f})^2 \ p(x) \ dx} \\
&= \int (f^2 - 2f\bar{f} + \bar{f}^2) \ p(x) \ dx \\
&= \bar{f}^2 - 2\bar{f}^2 + \bar{f}^2 = \bar{f}^2 - \bar{f}^2.
\end{align*}$$ (IV.2)

A more efficient way than the "hit or miss" method is outlined in the strong law of large numbers. The technique is commonly called "crude Monte Carlo"; however, we shall refer to it by a more modern name, "the sample-mean method." Again, the idea is to compute

$$I = \int_a^b f(x) \ dx$$

which can be rewritten as

$$I = \int_a^b \frac{f(x)}{p(x)} \ p(x) \ dx$$

where, as before, $p(x)$ is given by

$$p(x) = \frac{1}{b-a} \text{ for } a \leq x \leq b.$$ (IV.3)

According to the first theorem above we should be able to estimate the value of $I$ by
The algorithm for performing this calculation is more simple than before:

1. Generate a sequence of $N$ random numbers $\{u_i\}$ normalized to the interval $[0,1]$.
2. Compute $x_i = a + U_i (b-a)$, $i = 1, \ldots, N$.
3. Compute $f(x_i)$, $i = 1, \ldots, N$.
4. Finally, compute the sample mean $I_N = \frac{(b-a)}{N} \sum_{i=1}^{N} f(x_i)$.

As often occurs in applied mathematics, a tactical problem occurs in the implementation of the algorithm because of the wording of the first theorem. The result $I_N$ holds only "for sufficiently large $N". In most applications there is no way to determine a priori how large $N$ should be. Most times, this does not cause trouble since $N$ can be made large enough by intuitive means. The successful application of the algorithm is clearly a case of engineering judgement.

In regard to bounding the error we must look at the central limit theorem. The probability that the deviation of $f_N$ from $\bar{f}$ is greater than $\pm \epsilon \sqrt{N}$ is given by substituting $\delta = \epsilon \sqrt{N}$ in Equation (IV.1) and then subtracting it from one. That is,

$$
\text{Pr} \{ |f_N - \bar{f}| \geq \epsilon \sqrt{N} \} = 1 - \text{Pr} \{ \bar{f} - \delta \leq f_N \leq \bar{f} + \delta \} = \sqrt{\frac{2}{\pi}} \int_{\epsilon}^{\infty} e^{-x^2/2} \, dx.
$$
We can get a qualitative idea of how bad the deviations are by looking at the table below.

TABLE IV.1

| $\epsilon$ | $\Pr \{ |\bar{f}_N - \bar{f}| \geq \varepsilon \sqrt{V/N} \}$ |
|------------|-------------------------------------------------------------|
| 0.0        | 1.000                                                       |
| 0.5        | 0.492                                                       |
| 1.0        | 0.317                                                       |
| 2.0        | 0.045                                                       |
| 3.0        | 0.003                                                       |

It is apparent that deviations in excess of $\pm 3\sqrt{V/N}$ are rare. The quantity $\sigma = \sqrt{V/N}$ is defined as the standard deviation and is used by statisticians to measure sampling error. Since $V$ is a constant for a given problem, the standard deviation can be lessened by increasing $N$. It should be noted that making $N$ larger results in diminished returns since, for instance, halving the error requires quadrupling the number of points.

To apply the table, we need the variance explicitly. Using Equations (IV.2) and (IV.3),

$$V(I_N) = V \left[ \frac{b - a}{N} \sum_{i=1}^{N} f(x_i) \right]$$

$$= \left( b - a \right)^2 \int_{a}^{b} \frac{[f(x)]^2}{b - a} \, dx - 1^2$$
\[ = \left[ (b - a) \int_{a}^{b} [\xi(x)]^2 \, dx - I^2 \right] . \quad (IV.4) \]

We see that in order to calculate the variance explicitly, we must know,

\[ \int_{a}^{b} [f(x)]^2 \, dx \quad \text{and} \quad I. \]

However, if we know these values, we do not need Monte Carlo. Normally, the terms are approximated by some other calculations, probably $\overline{f}_N^2$ and $\overline{f}_N$, respectively. This is again acceptable for "N sufficiently large", but frequently disturbs practitioners because the error bound is calculated with possibly erroneous results. However, the problem is not as serious as one might initially think, because errors in estimates of errors can be fairly severe without destroying one's confidence in the results. For instance, if an error of 3% is computed and it happens to be miscomputed by a factor of 2 (that is, the error estimate is 100% off!), the result can probably be trusted; in a practical problem where a 3% error would be considered acceptable, a 6% error would also probably be acceptable. Again, a degree of engineering judgment must be injected into the practical application of Monte Carlo integration.

Neither of the two theorems presented depends upon the dimensionality of the integral involved. This is one of the main strengths of the Monte Carlo approach over other quadrature methods. Although
it is true in most techniques that the variance increases with the number of dimensions, it is also usually true that the number of points required for the quadrature increases geometrically. There is no reason in principle why one would need more integration points for a multidimensional quadrature and the Monte Carlo method exploits this circumstance.

The "hit or miss" method was claimed to be not as efficient as the sample mean method just described. This is an important point since the "hit or miss" method is responsible in large part for the sometimes bad reputation of the Monte Carlo method (Hammersley and Handscomb, 1964). To support this claim, the idea of efficiency must be defined. Suppose two estimates exist for computing the same integral. Call these $I_1$ and $I_2$. Let $t_1$ and $t_2$ be the units of computing time necessary to arrive at these estimates. The first method is said to be more efficient than the second if

$$t_1 \cdot V(I_1) < t_2 \cdot V(I_2) \ .$$

Comparing "hit or miss" with sample-mean integration proceeds as follows. First, using the notation of the "hit or miss" example

$$V(I_1) = [h(b - a)]^2 \frac{V(N_H/N)}{N}$$

$$= \frac{h^2(b - a)^2}{N} V(N_H)$$

$$= h^2(b - a)^2 P(1 - P)$$

$$= I [h(b - a) - I] \quad (IV.5)$$
Subtracting Equation (IV.3) from (IV.5), we have

\[ V(I_1) - V(I_2) = b - a[hI - \int_a^b f(x)^2 \, dx]. \]

Note that necessarily

\[ f(x) \leq h. \]

Therefore

\[ hI - \int f(x)^2 \, dx \geq 0. \]

Thus,

\[ V(I_1) - V(I_2) \geq 0. \]

Reasonably assuming that \( t_1 \) and \( t_2 \) are approximately the same, we have the sample mean method being more efficient than the "hit or miss" technique.

The sample mean method is far from the most efficient of all random number-based devices, but it is the best for the general type of geometry in which we will be interested. Refinement of the methods with so-called variance reduction techniques requires more specific known information about the problem. (Variance reduction may be desirable in particular applications of the computer program where a permanent modification of the code is in order.)

In the previous paragraphs the term "random number" has been used frequently without explanation as to what the expression
means. A random sequence of numbers is ideally a set of numbers randomly distributed across the set of all possible numbers in a range. A device which would generate such a sequence might be the throwing of a ten-sided die numbered from zero to nine where each throw gives a digit. A throw or group of throws gives a random number. This method cannot be simulated exactly on a computer since a computer program with given data is deterministic and its results repeatable upon demand. However, it isn't likely that one would want a truly random sequence, since the numbers cannot be duplicated in order to check a calculation. What is done in practice is to generate a "pseudorandom" sequence of numbers with the computer. A pseudorandom sequence of numbers is a deterministic sequence of numbers defined mathematically as

"a vague notion embodying the idea of sequence in which each item is unpredictable to the uninitiated, and whose digits pass a certain number of tests traditional with statisticians and depending somewhat on the uses to which the sequence is to be put"

(Davis and Rabinowitz, 1975). The word "random" is still used in the literature and will be used here.

A great deal of research has gone into devising random sequences on computers. A common and effective set of methods is collectively described as "multiplicative." The generated sequence defined by a multiplicative random number generator is started with a seed value $J_0$. The integers $J_1, J_2, \ldots J_N$, are defined by the recursive formula

$$J_{n+1} = aJ_n + c \mod m$$
Here $a, c,$ and $m$ are integers and the above notation means that $J_{n+1}$ is the remainder of $(aJ_n + c)/m$. Care must be taken in the selection of $a, c,$ and $m$ since division by $m$ can produce at most $m$ different remainders. Proper selection of the constants should produce a period that is large relative to the number of random numbers required for a computation. Normally, $m$ is chosen as the word size of the machine; $a$ and $c$ are selected by statistical tests to give a uniform distribution.

The random number generator used in this work is a variation on RANDU, the subroutine used to generate random numbers in the IBM 360 Scientific Subroutine Package (Hughes, et al, 1977). With appropriate values of $a$ and $c$ given in the program (see the Computer Implementation section, Chapter V), the code will produce uniformly distributed random numbers on a specified interval approximately $2^{29}$ times before repeating. It should be noted that the random number generator is appropriate only for IBM 360-370 type machines. On another computer, the user will have to provide his own routine.

It will be rare when an arbitrary domain will coincide with the rectangular distributions of random numbers that have been discussed here. This presents no problem from a theoretical standpoint since it can be shown that an even distribution of random numbers over a rectangular domain is also evenly distributed over any subregion of that domain (Rubinstein, 1981).
Application of Monte Carlo to Boundary Methods

We have just examined a powerful technique of integration that offers the possibility of eliminating explicit domain discretization in nonhomogeneous boundary integral formulations. The Monte Carlo theory can be put into practice in this context with a remarkably simple algorithm.

Consider for example a two-dimensional region \( V \) such as that in Figure IV-1; two dimensions are considered for simplicity although the generalization of the following idea is valid in any number of dimensions. The region can be multiply connected to any order; the boundaries can contain cusps or indentations, and any number of reentrant corners. Suppose further that we wish to integrate some function \( f(x,y) \) over this region. The futility of attempting a mesh generation procedure or some analytical integration technique is evident. To apply Monte Carlo, one actually requires only one assumption — that the function \( f \) be integrable. However, for the purposes of this study, an additional restriction will be invoked. We will require that the region have a well-defined external boundary; i.e. the boundary cannot go to infinity. Another way of stating this is to declare that the limits of integration must be bounded.

The following algorithm can be applied to obtain the value of the integral:

1. Circumscribe the exterior of the region with a rectangle \( R \) defined by vertices \((x_{\text{min}},y_{\text{min}}), (x_{\text{min}},y_{\text{max}}), (x_{\text{max}},y_{\text{min}}), \) and \((x_{\text{max}},y_{\text{max}})\); the minima and maxima denote the extreme values
Figure IV-1

'Very' general two-dimensional body
of the boundaries of $V$. (The rectangle touches the boundary at at least four points.) This guarantees $V \subseteq R$.

2. Choose a uniformly distributed random coordinate $(x^*_i, y^*_i) \in R$.

3. Test $(x^*_i, y^*_i)$ with respect to location. If $(x^*_i, y^*_i) \in V$, continue. Otherwise, return to step 2.

4. Compute $f(x^*_i, y^*_i) = f_i$ and add this value to the running sum $\sum f_i$. Do the same with $f^2_i$.

5. Repeat steps 2 - 4 until the desired number $N$ of observations is obtained.

6. Denote by $A_V$ the area of $V$. Compute

$$I = \frac{A_V}{N} \sum_{i=1}^{N} f_i$$

as the value of the integral, with variance

$$V = \frac{A_V}{N} \sum_{i=1}^{N} f_i^2 - I^2 .$$

The variance can be used to estimate the error in the calculation.

With this algorithm and the theory of Chapter II, a linear boundary element computer program was developed to solve nonhomogeneous problems involving the Laplacian operator in two-dimensional planar and axisymmetric geometries. In the course of constructing the program another interesting use for Monte Carlo integration was found. The axisymmetric problem requires the integration of elliptic
integrals to form the system matrices (see Equation (III.13) and the explicit fundamental solution). All integrations $G_{ij}$ and $H_{ij}$ can be performed using a standard Gaussian quadrature except for the diagonal terms which have singular integrands. Unlike the two dimensional case, the diagonal terms cannot be integrated analytically unless some limiting form of the integrand is used instead of the actual integrand. A complicated procedure ensues where it is necessary to integrate a portion of the element analytically when the approximation is valid, and the rest of it numerically. The analytical form of the $G_{ii}$ term is given for a constant element in Wrobel and Brebbia (1981). The expression occupies three printed lines in that paper and involves computation of five natural logarithms. For linear elements, each $G_{ii}$ would be composed of two similar, more involved terms. Clearly, this is an expensive term to compute.

The Monte Carlo integration procedure was used to compute these terms instead of the explicit procedure. It was found that all the terms could be computed with small standard deviations by using a few hundred random points. This formulation makes use of Equations (IV.3) and (IV.4) directly, with the fundamental solution $u^*$ from Equation (II.16) replacing $f(x)$.

Success of the Method

As the results in Chapter VI will indicate, the Monte Carlo technique can be used effectively in computing the domain terms
in the Poisson formulation. However, it is sometimes necessary to use several thousand points to get sufficient accuracy with this approach. Even with an advanced computer, this many points (multiplied by the number of nodal points) uses a lot of central processing time. For instance, an analysis of a cracked circular shaft necessitated the evaluation of fifty $B_I$ terms with three thousand random points. This required a minimum of one hundred-fifty thousand random-based computations in addition to the effort required to form and solve the equations. The actual run time for this problem was seven minutes and five seconds. The time required was not excessive, especially in light of the very small amount of time required to model the system.

Success was achieved with several Poisson-type problems, presented in Chapter VI, and it is concluded that the method works well in these problems.

The lengthy run time requirements for the generation of many domain terms forbid the use of the technique in problems where iteration must be used or large matrices of domain-dependent terms are required. For instance, the diffusion equation discussed in Chapter II requires the iterative scheme of Equation (II.42).

The $[D]$ matrix is of the size $N_B \times (N_B + N_L)$, where each term is the result of a domain integral. A moment's inspection reveals that even a small problem will require an excessive amount of time to
form the matrices. In the cracked shaft example, with forty-five boundary points and five internal points, an equivalent size transient formulation would require a [D] matrix with 2250 terms and require at least forty-five times as much CPU time. The result would be a run time requirement in excess of five hours just to form the matrices! Therefore, it is concluded that Monte Carlo is unsuitable for any reasonably sized transient analysis. For smaller problems, the modelling time would not be a major concern and discretization may as well be done. This issue will be addressed again in the final section of this work.

The next chapter deals with the construction and description of the computer program.
CHAPTER V

COMPUTER IMPLEMENTATION

Based upon the theory developed in Chapter II, and the Monte Carlo methods of Chapter IV, several computer programs which use boundary element data were written and tested. The result is a general double-precision FORTRAN program to solve the orthotropic Laplace and Poisson equations by a combination of linear boundary elements and Monte Carlo procedures. The program is designed to accommodate general two-dimensional and axisymmetric geometries with potential value and flux boundary conditions specified. Each subroutine is examined separately. A user's guide and the Job Control Language necessary to implement the program are in Appendices B and C respectively; the actual program and a sample of the output are in Appendices D and E. A careful reading of this chapter, along with a study of the relevant appendices will provide the user with enough information to run the program on his own.

The program is purposely designed so that additional modules can be added without unnecessary branching or excessive rewriting. In a program modification project, the programmer will find the carefully commented documentation at the head of each program helpful. This documentation contains the calling arguments of the programs and describes what each argument is.
Main Program MAIN

The main program does very few actual computations. Its purpose is to serve as a central organizer. In this module, the I/O device numbers are defined, as are the limits on the problem size (200 nodes in this version). The program calls each subroutine as necessary and then terminates the execution. Figure V-1 shows the flow of the entire program as controlled by MAIN.

Subroutine INPUT

INPUT reads and echoes the input data to the program. The first program input parameters consist of boundary node numbers along with their coordinates and the values of the specified potential or flux, whichever is relevant. Up to 200 nodes can be used to define a body's surface.

Also input are the material properties of the region under analysis. The material properties consist of the orientation of the orthotropic axes with respect to the global axis system, and the thermal conductivities of the material along each axis.

In a Poisson analysis, the user must specify the number of random points for the domain integration. In principle this can be any number. The elements surrounding the material are defined by their endnodes and are input in that manner. One thousand of these are allowed. Element cards are necessary in order to define the direction of the normal by numbering the elements clockwise (inward) as shown in Figure V-2.
Figure V-1

Macro-flow diagram for computer program
Figure V-2
Outward normal and contour directions for external and internal surfaces
The last data group is the set of internal coordinates at which the user desires results to be calculated. Up to 500 of these are allowed.

Subroutine FORMGH

FORMGH is the workhorse of the solution package. It forms the \([G]\) and \([H]\) matrices as described in Chapter II, sorts them according to the boundary conditions and adds in the Poisson vector \([B]\). The routine creates the system \([A]\) \([Y]\) = \([F]\) ready for final solution.

Subroutine SELFI

This routine computes the diagonal terms of the \([G]\) matrix by direct analytical computation in the two-dimensional case and by Monte Carlo integration in the axisymmetric problem (see Chapter IV).

Each call to the subroutine results in the term associated with one of the endnodes to be computed. The terms are given by

\[
g_{1A} = \int_0^\lambda \frac{x}{\ell} u^* \, dS \\
g_{1B} = \int_0^\lambda u^* \, dS - g_{1A}
\]

where \(u^*\) is the appropriate orthotropic fundamental solution, and \(x\) measures distance along the element in the \(x_1, x_2\)
plane of the model. The form of $dS$ depends upon whether or not the
gometry is two-dimensional or axisymmetric. In 2-D, $dS = dx$. For
axial symmetry $dS = rdx$ where $r$ is the radial coordinate measured
in the global coordinate system.

Subroutine INTEG

This is the equivalent of SELF for the nondiagonal terms of the
$[G]$ and $[H]$ matrices. It computes

$$
g_{iA} = \int_{S_j} N_A u^* \, dS \quad \text{and} \quad g_{iB} = \int_{S_j} N_B u^* \, dS ,
$$
as well as

$$
h_{iA} = \int_{S_j} N_A q^* \, dS \quad \text{and} \quad h_{iB} = \int_{S_j} N_B q^* \, dS
$$

using a standard four point Gaussian quadrature. Conceptually the
integration appears as shown in Figure V-3 where the variable of
integration has been transformed in such a way that the limits of
integration run from $-1$ to $1$. Each integral is of the form

$$
I = \int_{-1}^{1} f(t) \, dt = \sum_{i=1}^{4} w_i f(t_i) .
$$
Figure V-3

Numerical integration of boundary elements
The $t_i$ and $w_i$ are abscissae and weighting factors, respectively, given in Stroud and Secrest (1966) or almost any numerical methods handbook.

**Subroutine EXTREM**

This is a simple routine that sorts the coordinates of the nodes in order to determine the extreme values of the boundary curve. The subprogram is used in the Poisson analysis to generate a rectangularly uniform, evenly distributed set of random numbers. The necessity of doing this is described in the program algorithm in Chapter IV.

**Subroutine RANDOM**

RANDOM computes a random number on the interval (0,1) and linearly transforms it into a range of the user's choice. After a large number of calls to this program, an essentially uniform set of random numbers is obtained. The program will generate approximately $10^{29}$ numbers before repeating. It is a modified version of the program RANDU found in IBM's Scientific Siboutine Package and also in Hughes, et al (1977).

**Subroutine OUTIN**

OUTIN is one of the most generally useful programs developed in the course of this research. It performs the task of determining whether a point resides inside or outside a given region. The module accomplishes this by a simple application of the residue theorem.
A first-order pole is introduced at the test point, and a line integral is performed around the boundary of the region. According to the residue theorem,

\[ \oint_{\mathbb{R}} \frac{dz}{z - z_0} = 2\pi i \text{ for } z_0 \subset \mathbb{R} \]

\[ = 0 \text{ for } z_0 \not\subset \mathbb{R}. \]

The routine tests each trial point in the rectangle (defined by EXTREM) to determine if it is also contained inside the region of analysis. For a modification of the program to accommodate three dimensions the relevant analogous procedure would be Gauss' Law.

**Subroutine OUTPUT**

OUTPUT is a routine which, as its name implies outputs the results of the analysis in the form of potentials and fluxes at the boundary, and potentials at the specified internal points. Also, in the case of any Monte Carlo integration, the results of that integration as well as the standard deviation are reported.

**Subroutine POISN**

Subroutine POISN computes the terms \( \{b\} \) given by

\[ \int_{\mathbb{V}} \mathbf{b}^* \, d\mathbf{V} \]

where \( \mathbf{V} \) is the domain of consideration. The integration is performed
with the Monte Carlo technique described in Chapter IV; it requires a user-specified number of random points and the function \( b \) (also specified by the user). After a call to POISN, FORMGH receives a value \( B_1 \). This subprogram makes extensive use of RANDOM and OUTIN to obtain sample points for the integration. The function \( b \) is computed by a yet to be described routine, FUNCTION \( F \).

**Subroutine SOLVE**

This is a standard routine to solve a system of equations with a non-positive definite coefficient matrix. The method is Gaussian elimination with partial pivoting. It is effectively the same program as that used in the text by Brebbia (1978).

**Subroutine TERNAL**

After the values of the potential and the flux are computed on the boundary, this program computes the potential at user-specified internal points. The formula was derived in Chapter II. It is

\[
u^I_i = \sum_{j=1}^{N} (G^I_{ij} - H^I_{ij}u_j) - B^I_i .
\]

TERNAL uses the Poisson-term generator POISN to generate the \( B^I_i \) terms needed to compute \( u^I_i \).
**Function ELLPTC**

ELLPTC is a utility program for the computation of the complete elliptic integrals that occur in the axisymmetric fundamental solution and its derivative. The method by which this is done is a polynomial approximation in constants \(a_{ji}\) and \(b_{ji}\) given by

\[
K_j (m) = \sum_{i=1}^{4} (a_{ji} x_i^i + b_{ji} x_i^i \ln x)
\]

where \(x = 1 - m\) and \(j\) denotes the type of the integral. \(j = 1\) implies the elliptic integral is of the first kind; \(j = 2\) implies the integral is of the first kind. The formula has a tolerance of \(10^{-8}\) for \(0 < m \leq 1\) (Abramowitz and Stegun, 1964). Since the routine is used only for Monte Carlo integration, the isolated case for which \(m = 0\) \((K_j \to \infty)\) is dealt with by setting \(K_j = 0\). This technique is known as "ignoring the singularity" (Squire, 1970).

**Subroutine RECALL**

The element data in the program consists of the element number and the endnode numbers. In order to save space, all three parameters are stored in one integer location in the form of

\[
LM(I) = 1000 \times N1 + N2
\]

where \(I\) is the element number, \(N1\) is the node number 1, and \(N2\) is node number 2. RECALL reconstructs \(N1\) and \(N2\) from the element data.
by doing the integer arithmetic:

\[ N_1 = \frac{LM(I)}{1000} \]

\[ N_2 = LM(I) - 1000 \times N_1 \]

**Subroutine PIKSHR**

For the purposes of checking the input data and providing the user with a hardcopy representation of the boundary element model, PIKSHR is provided. It is called at the user's option, and when invoked, creates two frames of the boundary grid. One frame consists of boundary elements, nodal points, node numbers, and a boundary condition legend. The other frame is the same, except for the omission of the node numbers.

Since PIKSHR is a plotting routine, it exhibits a lot of idiosyncrasies unique to the particular computer operating system that is being used (in this case, the LSU IBM 3033). The non-LSU user can either convert it to his system or eliminate it completely with little trouble.

**Function F**

This is a user-definable function that corresponds to the nonhomogeneous term in the Poisson equation. The user simply writes his own definition of F according to the rules of FORTRAN. At his disposal are the coordinates of the point, the transformed coordinates (in orthotropic system) and all the material in the common blocks of
the program. For instance, if the user wishes to analyze the steady-state temperature in a region with internal heat generation defined by \( Q(x, y) = \cos xy \), he simply codes in

\[ F = DCOS (X \ast Y) \]

on the appropriate line. It must be remembered that the routine is in double precision.

**Concluding Statements**

The computer code presented is simple, but novel in that many problems that could not be easily solved before are now very simple to deal with. Several programming innovations such as the Monte Carlo integrator and the user-definable function make it applicable to a wide class of problems governed by the Poisson equation. Additionally, the modular form of the routine makes it easy to modify. The next chapter deals with some sample analyses that were performed with this program.
CHAPTER VI

SAMPLE ANALYSES

The purpose of this section is to demonstrate some of the capabilities of the program developed in Chapter V, and also to illustrate some of the modelling considerations that are used in practical boundary element analysis.

Discussed here are sample problems from a variety of applications in which Laplace's and Poisson's equations occur. The examples range in complexity from very simple to nearly intractable. In each case, the governing equation is related to a relevant physical process, and the solution is compared with a result obtained by other means (if available).

The Poisson analyses offered here are novel in that they are performed with neither domain discretization nor analytical preprocessing. As a result, these examples probably represent the simplest device to date for solving the general Poisson equation in more than one dimension.

Square Plate with Internal Heat Generation

The first case to be studied is a simple application of Poisson's equation in two dimensions. The problem is an isotropic square plate occupying the region \(-6 < x < 6; \ -6 < y < 6\), subject to internal heat generation of unit value while its edges are held at
zero temperature. The governing equation for two-dimensional steady-state temperature is generally

\[ \frac{\partial}{\partial x} \left( k_x \frac{\partial u}{\partial x} \right) + \frac{\partial}{\partial y} \left( k_y \frac{\partial u}{\partial y} \right) = -Q(x,y) \]  \hspace{1cm} (VI.1)

where \( Q(x,y) \) is the rate of internal heat generation and \( k_x, k_y \) are thermal conductivities along the \( x \) and \( y \) axes.

For this case, Equation (VI.1) reduces to

\[ \nabla^2 u = -1 \]  \hspace{1cm} (VI.2)

in nondimensional terms. Other interpretations of \( u \) in Equation (VI.2) include the torsion function for a square shaft (Timoshenko and Goodier, 1951) and the deflection of a uniformly loaded membrane (Crandall, 1956).

An analytical solution can be obtained by expansion of \( u \) in eigenfunctions and using the orthonormal properties of the function to construct an open form series solution (Carslaw and Jaeger, 1959).

Besides explicit domain discretization there is another possibility for numerical solution that is sometimes mentioned (Jaswon and Symm, 1977). In principle, Poisson's equation can always be reduced to Laplace's equation if a particular solution \( u_p \) for

\[ \nabla^2 u_p = b \]
is known. In the case of Equation (VI.2), a particular solution can be obtained by inspection. It is just

$$u_p = -(x^2 + y^2)/4$$

If we let $$u = u_0 + u_p$$, the problem is reduced to solving

$$\nabla^2 u_0 = 0$$

with the boundary condition $$u_0 = -u_p$$ on all surfaces. Jaswonn and Symm (1977) have used this method effectively.

The boundary element model used for this analysis is shown in Figure VI-1; it accounts for symmetry by consisting of only the negative quadrant $$-6 < x < 0, 0 < y < 6$$. Boundary conditions are specified as zero flux on the surfaces $$x = 0, y = 0$$, and as zero temperature on the surfaces $$x = -6, y = 6$$. Sixteen elements and nodal points are used for the boundary mesh. An interesting modeling feature occurs in this discretization. The corner points are actually two nodes placed very close together. This is an expedient to eliminate the ill-defined "normal" direction of the flux at the corner; i.e. one node at the corner would create a conflict. In this manner, the normal derivative is empirically split into two components, one for each adjacent surface.

The solution was computed with 500, 1000 and 3000 random integration points. These results are given in Table VI.1 and are indicated by subscripts denoting the appropriate number of integration points. Also, the essentially exact solution obtained by series summation
Figure VI-1
Discretization of one quadrant of a square plate. Note the points where constant fluxes and potentials are specified. Also note the "doubling" of the nodal points at the corners in order to handle the ambiguous boundary condition. Potentials at five internal points are desired.
(Lebedev et al, 1965) is offered for comparison.

**TABLE VI.1**

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
<th>(u_{500})</th>
<th>(u_{1000})</th>
<th>(u_{3000})</th>
<th>(u_{\text{exact}})</th>
</tr>
</thead>
<tbody>
<tr>
<td>-2.0</td>
<td>2.0</td>
<td>8.985</td>
<td>8.543</td>
<td>8.537</td>
<td>8.690</td>
</tr>
<tr>
<td>-4.0</td>
<td>2.0</td>
<td>5.802</td>
<td>5.645</td>
<td>5.736</td>
<td>5.748</td>
</tr>
<tr>
<td>-2.0</td>
<td>4.0</td>
<td>5.718</td>
<td>5.634</td>
<td>5.633</td>
<td>5.748</td>
</tr>
<tr>
<td>-4.0</td>
<td>4.0</td>
<td>3.961</td>
<td>3.987</td>
<td>3.981</td>
<td>3.928</td>
</tr>
<tr>
<td>-2.0</td>
<td>0.0</td>
<td>9.761</td>
<td>9.607</td>
<td>9.718</td>
<td>9.588</td>
</tr>
<tr>
<td>-4.0</td>
<td>0.0</td>
<td>6.628</td>
<td>6.161</td>
<td>6.234</td>
<td>6.286</td>
</tr>
</tbody>
</table>

For 1000 and 3000 integration points, the results are within a four percent tolerance. In an engineering application the accuracy is excellent.

It is interesting to note the diminished returns that occur with the increase in the number of random points. The analysis demonstrates a practical consequence of the central limit theorem, Equation (IV.2); that is, in order to halve the tolerance, it is necessary to square the number of observations. Although the accuracy is generally better with 3000 points, some of the individual results obtained with 1000 points are superior.

As a matter of peripheral concern, run time requirements varied from a little more than forty-five CPU seconds for the 3000 point problem to about five seconds for 500 random points.
Insulated Heating Duct

As a nontrivial example of the use of the program in solving Laplace's equation, consider the case of a 2' x 2' metal heating duct insulated \( (k = 0.1 \text{ Btu/hr ft °F}) \) as shown in Figure VI-2(a). The temperature distribution inside the insulation will be found for a duct temperature of 1000°F and an outside temperature of 0°F.

This problem is used as an example in the text by Gebhardt (1961), where the analysis is done by a finite difference relaxation procedure. The finite difference mesh using thirty-nine points and the results obtained from that analysis are shown in Figure VI-2(b). Note that the use of symmetry to eliminate most of the domain from the analysis.

In order to form a comparable boundary element model with the same mesh size, the twenty-four node boundary model of Figure VI-3 was made. Referring to that figure, the specified potentials on the upper and lower surfaces are 1000°F and 0°F respectively. The flux is specified as zero on the other surfaces.

No analytical solution to this problem is available. However, the results of the two numerical solutions at some typical points are compared in Table VI.2.

The agreement between the results is excellent. This example points out other advantages of the BEM besides the reduced problem size and modelling effort. Note that to get results at a particular point by the finite difference method, it is necessary to place a node at that point and then adjust the grid accordingly. Also, to get more accurate results, it is necessary to subdivide
Temperature
= 1000°C inside duct
= 0°C outside insulation

Figure VI-2
(a) Insulated heating duct. The dashed lines indicate the region to be modeled. b) Finite difference solution to the problem (after Gebhardt)
Figure VI-3
Boundary element model of the heating duct in Figure VI-2. The coordinate system is the same. Values are desired at five internal points.
the domain into more cells. In the BEM formulation, the solution is obtained at any internal point desired. Since the domain is not being discretized, adjustments to the grid size can be made without creating a completely new model.

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
<th>( u_{FD} )</th>
<th>( u_{BEM} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.354</td>
<td>0.354</td>
<td>182</td>
<td>178.9</td>
</tr>
<tr>
<td>1.060</td>
<td>1.060</td>
<td>426</td>
<td>420.3</td>
</tr>
<tr>
<td>1.060</td>
<td>0.708</td>
<td>281</td>
<td>278.0</td>
</tr>
<tr>
<td>0.708</td>
<td>1.416</td>
<td>686</td>
<td>681.4</td>
</tr>
<tr>
<td>2.124</td>
<td>0.354</td>
<td>57</td>
<td>56.4</td>
</tr>
</tbody>
</table>

**Orthotropic Material Properties**

As a variation on the first example, consider again the case of a square region with internal heat generation. In this case, however, the thermal conductivity in the vertical direction is only one quarter of that in the horizontal. The region of analysis is now the area \( 0 < x < 6, 0 < y < 6 \) and symmetry is not accounted for in the model. This being the case, the boundary element simulation (Figure VI-4) is nearly identical to that of Figure VI-1. Boundary conditions are specified as zero temperature at all nodes. Internal heat generation occurs at unit value.
Figure VI-4

Boundary element model for a square plate with internal heat generation and orthotropic material properties. The entire plate is modeled in this analysis.
With the orthotropic fundamental solution in two dimensions (Equation (II.15)), the problem is easily solved by boundary elements without special handling. The vertical conductivity is simply input as a non-dimensional value of 1.0 and the horizontal is input as 0.25. Although not used in this problem, the program has the capability of handling orthotropic axes that are not parallel with global coordinate axes.

Results for the analysis at the five internal points indicated in Figure VI-4 are calculated. These results and an analytical solution from Carslaw and Jaeger (1959) are indicated in Table VI.3.

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
<th>$u_{2000}$</th>
<th>$u_{\text{exact}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.0</td>
<td>2.0</td>
<td>3.56</td>
<td>3.44</td>
</tr>
<tr>
<td>4.0</td>
<td>2.0</td>
<td>3.61</td>
<td>3.44</td>
</tr>
<tr>
<td>3.0</td>
<td>3.0</td>
<td>4.11</td>
<td>4.10</td>
</tr>
<tr>
<td>2.0</td>
<td>4.0</td>
<td>3.43</td>
<td>3.44</td>
</tr>
<tr>
<td>4.0</td>
<td>4.0</td>
<td>3.50</td>
<td>3.44</td>
</tr>
</tbody>
</table>

As indicated, because of symmetry the values of $u$ at four of the five points should be identical. The results obtained with 2000 random points are slightly different at these coordinates because different random integration points are used. However, the stability of the solution is still remarkable, even in light of the Strong Law of Large Numbers predicting such a result.
Axially Symmetric Problem; Concentric Spheres

To demonstrate the validity of the program in dealing with axial symmetry, the simple problem of steady-state potential between two concentric spheres is presented. The spheres have radii of 0.5 and 1.0. The inside surface is held at a potential of 100 while the outside is maintained at zero potential.

There are a variety of ways in which this problem can be modelled since symmetry appears in an infinite number of ways. Here we will consider the model of Figure VI-5, which consists of a 30° sector of the region to be revolved around the vertical axis of symmetry. Nodes are doubled at the corners as in the first example. Fluxes are specified as zero on nodes 4 - 8 and on nodes 13 - 15. Nodes 1 - 3 are specified to be at potential value 100 while nodes 12 - 16 are at zero.

Although this problem is governed by Laplace's equation, Monte Carlo integration is used as described in Chapter IV to integrate the diagonal terms of the [G] matrix. The analytical solution obtained from simple integration and application of boundary conditions is

\[ u = \frac{100}{r} - 100 \]

where \( r \) is the spherically radial coordinate. Results are compared at nodes 5 - 7 in Table VI.4.
INDICATES FLUX SPECIFIED

INDICATES POTENTIAL SPECIFIED

Figure VI-5

Boundary element model for studying the potential distribution between two concentric spheres. Much use is made of symmetry considerations.
TABLE VI.4

<table>
<thead>
<tr>
<th>Node</th>
<th>( r )</th>
<th>( u_{\text{program}} )</th>
<th>( u_{\text{exact}} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>0.625</td>
<td>58.5</td>
<td>60.0</td>
</tr>
<tr>
<td>6</td>
<td>0.750</td>
<td>31.8</td>
<td>33.3</td>
</tr>
<tr>
<td>7</td>
<td>0.875</td>
<td>14.9</td>
<td>14.3</td>
</tr>
</tbody>
</table>

This analysis illustrates one drawback of an axisymmetric analysis by this formulation. The region of analysis is purposely kept away from the axis of symmetry. This is because the diagonal terms of \([G]\) are necessarily small when the radial coordinate of the point they represent is small. This is no fault of Monte Carlo, but simply a consequence of the formulation. To keep the assembly matrix well-conditioned, it is advised not to locate any nodes near the axis of symmetry.

**Hollow Cylinder**

Another axisymmetric problem that can be handled easily with the boundary element formulation is the rough arrangement that appears in a tube furnace (Figure VI-6). A similar analysis was used by Wrobel and Brebbia (1980) for much the same reason it is presented here, as a validity check for the program. In idealized form, neglecting end effects, the problem is actually the cylindrical analog to the concentric sphere problem.
Figure VI-6

Axisymmetric model of a hollow cylinder simulating a tube furnace
The region of analysis is the annulus $2.0 < r < 6.0$, $0.0 < z < 6.0$. Zero flux boundary conditions are specified on the $z$-surfaces; a value of $u = 100$ is specified as the temperature on the inner radial surface. Again the computed solution is compared at various points against the easily obtained analytical solution:

$$u = 100 \frac{\ln(r/r_2)}{\ln(r_1/r_2)}.$$

<table>
<thead>
<tr>
<th>$x$</th>
<th>$y$</th>
<th>$u_{\text{program}}$</th>
<th>$u_{\text{exact}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.0</td>
<td>2.0</td>
<td>63.3</td>
<td>63.1</td>
</tr>
<tr>
<td>4.0</td>
<td>2.0</td>
<td>37.1</td>
<td>36.9</td>
</tr>
<tr>
<td>3.0</td>
<td>3.0</td>
<td>63.2</td>
<td>63.1</td>
</tr>
<tr>
<td>5.0</td>
<td>4.0</td>
<td>16.5</td>
<td>16.6</td>
</tr>
<tr>
<td>4.0</td>
<td>4.0</td>
<td>36.9</td>
<td>36.9</td>
</tr>
</tbody>
</table>

**Multiply-Connected Region; Hollow Cylinder**

The program can also be used to analyze multiply-connected regions (to any order). As an illustration, we can find the temperature distribution between two hollow cylinders as in the previous example, except from a different point of view (Figure VI-7). The region of analysis here is the same as in the previous example and the results are comparable. Modelling features that should be noted in Figure VI-7 are the large number of nodal points (48) required to represent the geometry accurately and the numbering system used.
Figure VI-7

Planar boundary element model of a hollow cylinder. This model illustrates the program's capability to handle a multiply-connected region.
On the internal surface, either node numbers must be specified in counterclockwise order, or element cards (not shown) must indicate the direction of the normal to the boundary by the way they are punched (see Appendix B). The normal to the surface always points outward from the region. For nodes 1 - 24 in this model, outward is away from the center, whereas for nodes 25 - 48 the direction is toward the center.

Again the boundary element solution is excellent. The success of the Monte Carlo BEM formulation in the axially symmetric cases is evident from this and the previous example.

**Variable Nonhomogeneous Function**

This example illustrates one of the more powerful features of the program. The user may specify any non-homogeneous function he wishes by modifying the subprogram FUNCTION F. In the previous examples, the function was a simple constant. We will now consider a case where the nonhomogeneous term varies continuously. The problem is heat generation inside a circular region of unit radius. The heat is generated at a rate of

\[ Q(r) = \cos \left( \pi r/2 \right) \]

where \( r = (x^2 + y^2)^{1/2} \). The governing relationship is Equation (VI.1) where \( k_x = k_y = 1 \) is assumed for simplicity. Because the problem is radially symmetric, polar coordinates can be used to solve for \( u \) as a function of the single variable \( r \). The Poisson equation is
\[ \frac{1}{r} \frac{d}{dr} \left( r \frac{du}{dr} \right) = -\cos(\pi r/2) \]

with boundary conditions

\[ u(1) = 0 \]

\[ u'(0) \text{ must be bounded.} \]

One integration gives

\[ u' = -(4/\pi^2 r) \cos(\pi r/2) - (2/\pi) \sin(\pi r/2) + C/r. \]

In order to make \( u'(0) \) finite, we must have \( C = 4/\pi^2 \). Integrating again, using the principal value of the first term, and applying \( u(1) = 0 \), gives ultimately:

\[ u = \frac{4}{\pi^2} \left[ \cos \frac{\pi r}{2} - \sum_{n=1}^{\infty} \frac{(-1)^n (\pi/2)^{2n} [r^{2n} - 1]}{2n(2n)!} \right] \]  \hspace{1cm} (VI.3)

The boundary element solution procedure is instituted by first changing the statement (see Appendix B)

\[ F = 0.0 \]

in the subprogram FUNCTION F to the two statements:

\[ R = \text{DSQRT} \left( XP*XP + YP*YP \right) \]

\[ F = -\text{DCOS} \left( R*1.57 \right). \]
(This is one possibility.) XP and YP represent the coordinates of a random point inside the circular region.

The boundary element discretization, which accounts for symmetry, is the positive quadrant shown in Figure VI-8. Zero fluxes are specified on the flat surfaces while zero temperatures are specified on the circular face. The results for 1000 random points are given in TABLE VI.6 along with values computed from Equation (VI.3).

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
<th>u_{1000}</th>
<th>u_{exact}</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.2</td>
<td>0.2</td>
<td>.162</td>
<td>.160</td>
</tr>
<tr>
<td>0.4</td>
<td>0.4</td>
<td>.105</td>
<td>.107</td>
</tr>
<tr>
<td>0.5</td>
<td>0.5</td>
<td>.073</td>
<td>.074</td>
</tr>
<tr>
<td>0.7</td>
<td>0.7</td>
<td>.004</td>
<td>.003</td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>.180</td>
<td>.180</td>
</tr>
<tr>
<td>0.0</td>
<td>0.1</td>
<td>.172</td>
<td>.177</td>
</tr>
<tr>
<td>0.0</td>
<td>0.3</td>
<td>.163</td>
<td>.158</td>
</tr>
<tr>
<td>0.0</td>
<td>0.5</td>
<td>.122</td>
<td>.122</td>
</tr>
<tr>
<td>0.0</td>
<td>0.7</td>
<td>.075</td>
<td>.075</td>
</tr>
</tbody>
</table>

Not only are the results outstanding, but the ease with which they were obtained is also remarkable. For an equivalent finite element formulation, each element would have to contain a lumped value of the function Q over the entire element. This would probably generate inaccuracies in the final solution that would not be commensurate
Figure VI-8

Discretization used for a circular plate with internal heat generation.

Symmetry is used to reduce the size of the model.
with the effort required to obtain that solution. This consideration is in addition to the previous remarks made in this paper concerning modelling effort.

As a matter that may be of concern, the total CPU time to run this problem with eighteen boundary nodes and four internal nodes was fifty-three seconds. Total modelling, data input, and run time totalled a little less than six minutes. By comparison, the total time required to derive Equation (VI.3), program it, and get the results was more than forty-five minutes.

The power of the boundary element-Monte Carlo couple is further exemplified by the following problem concerning torsion.

Circular Shaft with a Radial Crack

A very interesting problem governed by Poisson's equation is torsion of an elastic shaft. In nondimensional form, the equation appears as

$$\nabla^2 u = -2$$

where \( u \) is a nondimensional torsion function from which the stresses in the shaft can be derived (Murphy, 1946).

Consider a circular shaft of unit radius with a radial crack running from its center to its outside edge. Since the edges are free, no shearing stresses may exist there. This implies that \( u \) on the boundary is a constant which may be taken to be zero, arbitrarily. A boundary element discretization into 45 elements was performed as shown in Figure VI-9 where the crack was approximated as a gap with
Boundary element model of a circular shaft containing a radial crack. Values of the torsion function are desired in this model. The crack is modelled as a gap of clearance one one-hundredth the radius of the shaft.
0.01 clearance. (One who is acquainted with finite element modeling can truly appreciate the simplicity of the boundary discretization.)

An exact open-form solution does exist (Lebedev et al, 1965). It is obtained from eigenfunction expansion in cylindrical coordinates and integration of the orthonormal eigenfunctions.

The problem was run with 1000 and 3000 random points, respectively. The solutions obtained with the computer program and the analytical solution are presented in Table VI.7.

<table>
<thead>
<tr>
<th>$X$</th>
<th>$Y$</th>
<th>$u_{1000}$</th>
<th>$u_{3000}$</th>
<th>$u_{exact}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>0.2</td>
<td>0.1927</td>
<td>0.1982</td>
<td>0.2083</td>
</tr>
<tr>
<td>0.0</td>
<td>0.5</td>
<td>0.2094</td>
<td>0.2504</td>
<td>0.2412</td>
</tr>
<tr>
<td>0.0</td>
<td>0.7</td>
<td>0.1902</td>
<td>0.1776</td>
<td>0.1842</td>
</tr>
<tr>
<td>0.0</td>
<td>-0.5</td>
<td>0.2360</td>
<td>0.2423</td>
<td>0.2412</td>
</tr>
<tr>
<td>0.0</td>
<td>-0.7</td>
<td>0.1654</td>
<td>0.1802</td>
<td>0.1842</td>
</tr>
</tbody>
</table>

The general improvement of the solution in going from 1000 to 3000 points is apparent from the table. It should be noted that the Monte Carlo procedure dominates the run-time requirement. This is evident from the increase in CPU time needed to run the 1000 node problem (two minutes, forty-three seconds) to that of the 3000 node problem (seven minutes, five seconds). The CPU time ratio of the two problems is approximately equal to the corresponding random point ratio. Also it should be noted that both of the numerical
solutions were constructed, run, and back in the author's hands in less than forty-five minutes. By contrast, the time required to program the known analytical solution for this problem exceeded twenty-five minutes. To derive the solution from the basic differential equation would have taken much longer. The boundary element solution is clearly an advantageous method in this example.

With the potential values obtained at more internal points, qualitative plots of $u$ can be made through surface fitting. These are shown in Figures VI-10 and VI-11. With these, the stress distribution in the shaft can be judged on a relative basis.

**Shaft with an Eccentric Hole**

As a further illustration of boundary elements applied to complex torsion problems, we now consider a circular shaft containing an eccentrically drilled hole. In addition to being a demonstration of the Poisson capabilities of the program, this problem places special demands upon the routine, in that the domain is multiply connected. The region of analysis appears as shown in Figure VI-12. The problem can be solved by separation of variables in dipolar coordinates and has been done so by Weinel (1932). The exact solution, containing infinite series of bipolar coordinates, is very complicated and exceptionally difficult to work with; results were obtained with the analytical solution, however, in order to make a comparison with the boundary element solution.

The boundary element model is a simple modification of the two-dimensional concentric cylinder analysis performed earlier. The
Figure VI-10

Two views of the torsion function corresponding to a circular shaft with a radial crack.
Another view of the torsion function for the cracked shaft and a rough contour plot of the same.
Figure VI-12

End view of a shaft containing an eccentrically drilled hole. The exact solution is possible using a bipolar coordinate solution.
forty-eight node problem has an outer surface of radius 1.0 (nondimensional) and an inner surface of radius 0.5; the centers of the circles are offset by a distance of 0.25 (Figure VI-13). Twenty-four nodes were used to describe each surface. In order to make the solution compare with that of Weinel, boundary conditions were prescribed as $u = 0.208$ on the inner surface and $u = 0.176$ on the outer boundary.

Functional values were computed at a series of internal nodes on the horizontal axis of symmetry. The results are graphed in Figure VI-14 along with the values obtained from the expression by Weinel (1932).

As the graph indicates, boundary elements have again given excellent results. The run time for this analysis using 3000 points was seven minutes eight seconds.

Concluding Remark

Consistently throughout the examples in this chapter, boundary element analysis has accurately predicted solutions to problems of various complexity. With the instruction offered by these examples, the user's guide in Appendix B, and the program in Appendix C, the engineer now has available a tool of great power (yet simplicity) to solve the Poisson equation.

The next chapter summarizes and gives some general conclusions.
Figures VI-13

Boundary element model corresponding to the shaft in Figure VI-12

- INDICATES FLUX SPECIFIED
- INDICATES POTENTIAL SPECIFIED
RESULTS FOR A CIRCULAR SHAFT WITH AN ECCENTRICALLY DRILLED HOLE

Figure VI-14

Plots of the analytical and computed results for the shaft with an eccentrically drilled hole
CHAPTER VII

SUMMARY AND CONCLUSIONS

In the previous chapters, the boundary element method has been developed and compared with other numerical methods. BEM represents an advance over other techniques because of the boundary value nature of the formulation; i.e. the boundary is the predominant influence on what happens in the domain. The practical side of the theory has been enlarged upon, with the aid of Monte Carlo techniques to eliminate a troublesome term in the general Poisson equation formulation. It has been demonstrated that a program developed from this theory can be used to solve a wide variety of practical Poisson type problems. The program provides accurate, efficient results with a minimum of modelling time. The program's validity for the examples in Chapter VI implies its appropriateness for other similar problems.

Although the computer code is limited to two-dimensional and axisymmetric geometries, both the boundary element theory of Chapter II and the Monte Carlo methods of Chapter IV are not subject to these limitations. Therefore, since a program can be developed that solves two-dimensional problems, it is reasonable to assume that a general three-dimensional solution package can be implemented.

For the present, limitations on computer time exclude the use of software based upon the theory developed in this work for transient analysis. This circumstance is not as discouraging as it may seem. Because the problem is purely technical in nature, and because this
work represents a first attempt to confront the nonhomogeneous terms from the nondiscretization attitude, it is premature to conclude that the general transient problem cannot be handled by a variation on this theory. As a suggestion for future work toward this end, one might try experimenting with variance reduction techniques. They have been shown to significantly reduce the number of observations necessary to perform an accurate quadrature and thus would relieve the CPU time requirement.

The work presented in the previous chapters represents a beginning, not an end to research in the area of model reduction time. It is hoped that by using this dissertation as groundwork, future researchers will see the day when the engineer will no longer be required to spend large amounts of valuable time on discretization efforts.
REFERENCES


APPENDIX A

INTEGRATION BY PARTS IN SEVERAL VARIABLES

Many of the fruitful results obtainable by the weighted residual process are by-products of an integration by parts. The technique by which this is done is straightforward but the result is not obvious. Consider a two-dimensional Laplace's equation weighted against another function which is twice differentiable and integrated over a region \( R \). The problem appears as

\[
I = \int \int_R \left( \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right) w \, dR = 0 .
\]  

(A.1)

Consider the integral over the first term in Equation (A.1) alone. It can be written as

\[
I_1 = \int \int_R w \frac{\partial^2 u}{\partial x^2} \, dx \, dy
\]

Using ordinary one-dimensional integration by parts gives

\[
I_1 = \int_S w \frac{\partial u}{\partial x} \bigg|_{x=B(y)} \, dy - \int_S \int_R \frac{\partial w}{\partial x} \frac{\partial u}{\partial x} \, dx \, dy
\]

Here \( A \) and \( B \) are variable extremes of the region on lines parallel to the \( x \)-axis. These are not necessarily single-valued functions
of $y$. In terms of the differential $dS$ of surface length, $dy = \ell_x \, dS$
where $\ell_x$ is the direction cosine between the $x$-axis and the normal to
the surface. The expression above becomes

$$I_1 = \int_S w \frac{\partial u}{\partial x} \ell_x \, dS - \int_R \left( \int_R \frac{\partial w}{\partial x} \frac{\partial u}{\partial x} \, dx \right) \, dy .$$

The exact same procedure may be used for the second term in Equation
(A.1). The result can be drawn by analogy ($\ell_y$ is the direction cosine
of the normal with the $y$-axis):

$$I_2 = \int_S w \frac{\partial u}{\partial y} \ell_y \, dS - \int_R \left( \int_R \frac{\partial w}{\partial y} \frac{\partial u}{\partial y} \, dy \right) \, dx .$$

Now, since $I = I_1 + I_2$,

$$I = \int_S w \left( \frac{\partial u}{\partial x} \ell_x + \frac{\partial u}{\partial y} \ell_y \right) \, dS - \int_R \left( \int_R \left( \frac{\partial w}{\partial x} \frac{\partial u}{\partial x} + \frac{\partial w}{\partial y} \frac{\partial u}{\partial y} \right) \, dx \right) \, dy . \quad \text{(A.2)}$$

However, the term in parentheses in the first integral is just the
directional derivative of $u$ with respect to the outwardly directed
normal. By induction, Equation (A.2) can be generalized to any
number of dimensions. The general result for a Laplacian operator is

$$\int_V (V^2 u) \, dV = \int_S w \frac{\partial u}{\partial n} \, dS - \int_V \frac{\partial w}{\partial x_i} \frac{\partial u}{\partial x_i} \, dV \quad \text{(A.3)}$$
in which the Einstein summation convention is used.
APPENDIX B

USER'S GUIDE

The following pages may be used as an independent guide for the user desiring to run the program.
BOUNDARY INTEGRAL EQUATION PROGRAM FOR SOLUTION OF
POISSON EQUATION IN PLANAR AND AXISYMMETRIC GEOMETRIES

by
Gary S. Gipson
May, 1982

PURPOSE

Determination of the solution of the Poisson type equation
when subjected to specified boundary conditions.

METHOD OF ANALYSIS

The boundary integral equation technique coupled with a Monte
Carlo integration procedure is used as the method of analysis.
The theory and solution procedure are described in the disserta-
tion:

The Coupling of Monte Carlo Integration with the Boundary
Integral Equation Technique to Solve Poisson Type Equations,

by
Gary S. Gipson
College of Engineering
Louisiana State University
**INPUT DATA**

The first step is to create a boundary element mesh of the region or regions of interest. The nodal points are then numbered in sequence beginning with one. The elements are numbered in ascending order around the region of interest. See the diagram note (a) for a sample numbering scheme.

The following group of punched cards (b) numerically define the problem to be analyzed:

A. **Identification Card - (18A4)**

Columns 1 to 72 of this card contain information which the user desires to be printed with the output results.

B. **Control Card - (515, F10.0, 215)**

Columns 1 - 5  Problem type
  = 0 for planar analysis
  = 1 for axisymmetric analysis
6 - 10  Number of nodal points (200 maximum)
11 - 15  The integer 1
16 - 20  Plot code
  = 0 means no plots will be produced
  = 1 invokes the plotting package

C. **Nodal Point Cards (215, 4F10.0)**

Columns 1 - 5  Nodal point number
10  Boundary condition code
= 0 if normal flux specified
= 1 if functional value is specified

11 - 20 x-coordinate (r in axisymmetric case)
21 - 30 y-coordinate (z in axisymmetric case)
31 - 40 Boundary value at node specified by column 10

In case of an axisymmetric body the total heat flow on a one radian segment must be supplied in columns 31 - 40. For insulated nodal points, this value is zero.

Nodal point cards are input in numerical sequence starting with one. If cards are omitted, the omitted nodal points are generated at equal intervals along a straight line between the specified nodal points. For the generated points, the information in columns 10 and 31 - 40 is set equal to that on the last card in the sequence.

D. Material Property Cards

For each material two cards are necessary:

First Card (215, 3F10.0)

Columns 1 - 5 Material identification number
6 - 10 Number of elements surrounding the material or region
11 - 20 Thermal conductivity in direction of v-axis (c,d)
31 - 40 Angle of u-axis with respect to global x-axis (c,d)
Second Card (2I5, 4F10.0)

Columns 1 - 5  Number of internal points belonging exclusively to the region
6 - 10  Number of random points to be generated for Poisson analysis

E. For each region the following sets of cards must be supplied:

Internal coordinates cards (2F10.0)
-one card for each coordinate pair where the user wishes the function to be calculated:
Columns 1 - 10  x-coordinate
11 - 20  y-coordinate

Element cards (4I5)
-the elements are numbered sequentially in counterclockwise order around the periphery of the region (or clockwise for an external region). The endnodes 1 and 2 defining the elements are also in counterclockwise sequence.
Columns 1 - 5  Element number
6 - 10  Node 1
11 - 15  Node 2

Omitted element cards are generated. Nodes 1 and 2 are generated in evenly spaced increments consistent with the first and last cards in the generation sequence.

F. Poisson function
- The program contains a function subprogram which defines the right-hand side of a Poisson type equation; \( \nabla^2 u = f(x,y) \) for instance. It normally appears as:
FUNCTION F (XP, YP, XT, YT, MTYPE)

{COMMON BLOCKS}
F = 0.0
RETURN
END

which simulates Laplace's equation. If the user desires a different right-hand side, he must change the F = 0.0 statement to the appropriate wording. In fact, the function subprogram may be modified in any way consistent with FORTRAN rules for subprograms. The routine will be compiled and linked into the rest of the program at execution time. Here, XP and YP are coordinates of a point inside the domain. XT and YT are the transformed coordinates in the material axis system. In this version of the program MTYPE = 1 must be specified.

NOTES
(a) Figure B - 1 demonstrates the mesh numbering and input data.
(b) The I/O device numbers are variables defined near the beginning of the other program. If devices other than unit 5 are wanted for input and unit 6 for output, the statements can be changed appropriately.
(c) The u,v-axis is a local coordinate system that defines the axes of orthotropy; see the diagram in Figure B - 1.
(d) Input parameters are described in terms of heat transfer terminology for brevity.
Figure B-1

Sample element and nodal point numbering scheme. θ denotes the possible angle of orthotropy.
APPENDIX C

JOB CONTROL LANGUAGE

The computer program in Appendix D requires the following JCL setups for operation on the IBM 3033 computer at the LSU SNCC. The format is current as of April, 1982.

The first setup assumes that both the source deck and data are on cards.

Notes
(a) //Jobname JOB (pppp,dddd,cpu,lines), user name'
(b) //A EXEC FORTXCLG, REGION = 1000K
   //FORT.SYSIN DD *
       Source deck
   //GO.SYSIN DD *
       Data cards

If the program already exists on the system in the form of a load module, and it is only desired that the user-defined function F be modified, the following setup might be used:
(a)  //Jobname JOB (pppp,ddddd,cputme,lines),'user name'
(b)  //A EXEC FORTXCLG, REGION = 1000K
     //FORT.SYSIN DD *
     
     Source deck for FUNCTION F
     
     //LKED.IN DD DSN = dsname, DISP = SHR
(c)  //LKED.SYSIN DD *
(d)  INCLUDE IN (memname)
     
     ENTRY MAIN
     //GO.SYSIN DD *
     
     Data cards
     
//

NOTES:

(a)  Current LSU jobcard format. User dependent parameters are:
  Jobname = name of the job
  pppp = dept. project number
  dddddd = user i.d. number
  cputme = minutes of CPU time required
  lines = number of lines of output required (in thousands)
(b)  This setup uses the H-extended compiler. If the Gl-compiler or
     VS-compiler are desired instead, substitute G or V for X in
     FORTXCLG.
(c)  'dsname' is the name of the data set containing the load module
to be modified.
(d) 'memname' is the member of 'dsname' corresponding to the load module of the boundary element program.
This section contains a listing of the program described in Chapter V. The user will find it well-documented and easy to follow, but possibly a bit tedious; the program contains a variety of features necessary for future expansion of its capabilities. Therefore, a moderate amount of what may appear to be unnecessary complication will be evident.

Other than these isolated instances of overkill, the code is written in a manner that conforms to current ideas of modular structure (Brebbia, 1978), and every attempt has been made to design and structure it efficiently. However, experience has taught the author that no program is beyond improvement, and rarely is one beyond repair. Any suggestions, problems, helpful modifications, etc. will be welcomed by this author as a compliment from the user by means of his interest in the program.
PROGRAM

PURPOSE --- TO DETERMINE THE SOLUTION OF THE POISSON AND
LAPLACE'S EQUATIONS FOR PLANAR AND AXISYM-
METRIC GEOMETRIES

METHOD OF ANALYSIS --- THE BOUNDARY INTEGRAL EQUATION
TECHNIQUE COUPLED WITH A MONTE CARLO INTEGRATION SCHEME

ORIGIN --- LOUISIANA STATE UNIVERSITY, BY G.S. GIPSON,
MAY, 1982

IMPLICIT REAL*8(A-H,O-Z)

DIMENSION GKERN(200,200), HKERN(200,200), X(201), Y(201), SD(200),
FIXBND(200), FIXDER(200), XCENT(500), YCENT(500),
SOLV(200), SOLUT(200), BNDINT(200), B(200), KODE(200)

COMMON/TEMP/DELTA, NUMDLT, INTER
COMMON/IO/INDEV, IOUTDV, IPLOT, ISTOR1, ISTOR2, ISTOR3, ISTOR4, ISTOR5
COMMON/CNTRL/ITYPE, NUMEL(20), NUMINT(20), NUMMAT, NTTOT, NUMNP
COMMON/MTRL/UCOND(20), VCOND(20), SPHT(20), DENS(20), QU(20), QV(20),
UCOS(20), VCOS(20), QX(20), QY(20), NUMRAN(20)
COMMON/IMENT/LM(1000)
C
C = DEFINITION OF I/O DEVICE NUMBERS AND LIMITS ON PROBLEM =
C = SIZE =
C = NTOT = MAXIMUM NUMBER OF EQUATIONS =
C = INDEV = INPUT DEVICE NUMBER =
C = IOUTDV = OUTPUT DEVICE NUMBER =
C = ISTOR1 THROUGH ISTOR5 = AUXILIARY STORAGE FILES =
C =
C
INDEV=5
IOUTDV=6
NTOT=200
ISTOR1=1
ISTOR2=2
ISTOR3=3
ISTOR4=4
ISTOR5=8
C
C =
C =
C = INPUT OF PROGRAM DATA =
C =
C ===========================
C
CALL INPUT(XCENT,YCENT,X,Y,FIXBND,KODE,BNDINT)
C
C ===========================
C
C =
C = PLOT BOUNDARY MESH =
C =
C ===========================
C
IF(IPLT.EQ.1)CALL PIKSHR(X,Y,XCENT,YCENT,KODE)
C
C ===========================
C
C =
C = FORMATION OF SYSTEM EQUATIONS =
C =
C ===========================
C
CALL FORMGH(X,Y,GKERN,HKERN,FIXBND,KODE,SOLV,B,NUMRAN,SD)
C
C ===========================
C
C =
C = SOLVE SYSTEM OF EQUATIONS =
C =
C ===========================
CALL SOLVE(GKERN,SOLV,NUMNP,DET,OUTDV)

C

C = COMPUTATION OF INTERNAL POTENTIALS
C

CALL TERNAL(FIXBND,SOLV,KODE,XCENT,YCENT,SOLUT,X,Y,B)

C

C = OUTPUT OF COMPUTATIONS
C

CALL OUTPUT(X,Y,SOLV,FIXBND,XCENT,YCENT,SOLUT,BNDINT,TIME)

C

STOP

END
SUBROUTINE INPUT(XCENT,YCENT,X,Y,FIXBND,KODE,BNDINT)

C
C ===============================================================:
C = SUBROUTINE INPUT
C =
C =
C = THIS SUBROUTINE READS THE INPUT DATA AND INITIALIZES THE
C = PLOTTING DEVICES.
C =
C = CALLING ARGUMENTS -
C =
C = XCENT,YCENT - VECTORS CONTAINING COORDINATES OF INTERNAL
C = POINTS WHERE THE USER DESIRES THE FUNCTION
C = TO BE CALCULATED
C =
C =
C = FIXBND - VECTOR OF FIXED BOUNDARY CONDITIONS IN ONE-TO-ONE
C = CORRESPONDENCE WITH THE X,Y(200 MAXIMUM)
C =
C = KODE - VECTOR CONTAINING BOUNDARY CONDITION CODES IN A
C = ONE-TO-ONE CORRESPONDENCE WITH FIXBND
C =
C = IMPLICIT REAL*8 (A-H,O-Z)
C IMPLICIT REAL*8 (A-H,O-Z)
DIMENSION TITLE(18),XCENT(1),YCENT(1),X(1),Y(1),FIXBND(1),
BNDINT(1),KODE(1)
COMMON/I0/INDEV, IOUTDV, IPILOT, ISTOR1, ISTOR2, ISTOR3, ISTOR4, ISTOR5
COMM0N/CNTRL/ITYPE, NUMEL(20), NUMINT(20), NUMMAT, NTO T, NUMNP
COMMON/MTRL/UCOND(20), VCOND(20), SPHT(20), DENS(20), QU(20),
$ QV(20), UCOS(20), VCOS(20), QX(20), QY(20), NUMRAN(20)
COMMON/TEMP/DDELTA, NUMDLT, INTER
COMMON/LMENT/LM(1000)

EQUIVALENCE (NUMMAT, NREG)

C
C  = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
C  =
C  = TITLE AND CONTROL CARDS
C  =
C  = TITLE(18A4) - HEADING CARD CONTAINING USER'S
C  =
C  = DESCRIPTIVE INFORMATION
C  =
C  =
C  = CONTROL CARD(4I5, F10.0, 215) - CONTAINS THE FOLLOWING =
C  =
C  =
C  = ITYPE = 0 FOR PLANAR ANALYSIS =
C  =
C  = 1 FOR AXISYMMETRIC ANALYSIS =
C  =
C  = NUMNP = TOTAL NUMBER OF NODAL POINTS( 200 MAXIMUM ) =
C  =
C  = NUMMAT = TOTAL NUMBER DIFFERENT MATERIALS( 1 MAXIMUM ), =
C  =
C  = THERE IS ONLY ONE MATERIAL IN THIS VERSION. =
C  =
C  = IPILOT = 0 MEANS NO PLOTS WILL BE PRODUCED =
C = INVOKES THE PLOTTING PACKAGE

C =

C =

C =

C = READ(INDEV,1000)TITLE
READ(INDEV,1100)ITYPE,NUMNP,NUMMAT,
$ IPLOT,DELTA,NUMDLT,INTER
C
WRITE(IOUTDV,2000)TITLE,ITYPE,NUMNP,NUMMAT,IPLOT
IF(ITYPE.EQ.0)WRITE(IOUTDV,2110)
IF(ITYPE.EQ.1)WRITE(IOUTDV,2120)
C
C =

C = INPUT OF BOUNDARY COORDINATES(EXTREME POINTS OF
C = BOUNDARY ELEMENTS) AND BOUNDARY CONDITIONS(215,4F10.0) =
C = OMITTED NODES ARE GENERATED IN A STRAIGHT LINE
C =
C = - NUMBERING OF NODES PROCEEDS
C = IN ASCENDING ORDER
C =
C = NODE(I) = NODE NUMBER
C =
C = KODE(I) = BOUNDARY CONDITION CODE
C = 0 IF FLUX IS SPECIFIED AT NODE
C = 1 IF TEMPERATURE(OR ANALAGOUS QUANTITY)
C = IS SPECIFIED AT NODE
C = X(I) = GLOBAL X-COORDINATE OF NODE
C = Y(I) = GLOBAL Y-COORDINATE OF NODE
C = FIXBND(I) = BOUNDARY CONDITION AT NODE
C = GENERATED BOUNDARY CONDITIONS
C = ARE SET EQUAL TO INFO ON FIRST
C = CARD IN SEQUENCE
C
C ================================================================

C IERROR=0
WRITE(IOUTDV,533)
L=1
60 READ(INDEV,540)N,KODE(N),X(N),Y(N),FIXBND(N),BNDINT(N)
DIFF=DFLOAT(N+1-L)
IF(N-L)80,110,90
80 WRITE(IOUTDV,760)N
IERROR=1
C
GO TO 60
C
90 DX=(X(N)-X(L-1))/DIFF
DY=(Y(N)-Y(L-1))/DIFF
C
100  KODE(L)=KODE(N)
     X(L)=X(L-1)+DX
     Y(L)=Y(L-1)+DY
     BNDINT(L)=BNDINT(N)
     FIXBND(L)=FIXBND(N)
110  WRITE(IOUTDU,610)L,KODE(L),X(L),Y(L),FIXBND(L)
     L=L+1
120  IF(N-L)120,110,100
     IF(NUMNP+1.GT.L)GO TO 60
C
C ============================================================
C = MATERIAL PROPERTY CARDS - TWO CARDS FOR EACH MATERIAL
C = ( 1 MATERIAL MAXIMUM )
C = - FIRST CARD(2I5,3F10.0)
C = MAT = MATERIAL NUMBER
C = NUMEL(MAT) = NUMBER OF ELEMENTS SURROUNDING THE MATERIAL
C = UCOND(MAT) = THERMAL CONDUCTIVITY IN DIRECTION OF U-AXIS
C = VCOND(MAT) = THERMAL CONDUCTIVITY IN DIRECTION OF V-AXIS
C = ANGLE = ANGLE IN DEGREES CCW OF U-AXIS WITH RESPECT TO
C = GLOBAL X-AXIS
C = -SECOND CARD(215)
C = NUMINT(MAT) = NUMBER OF INTERNAL POINTS BELONGING
C = EXCLUSIVELY TO THE REGION
C = NUMRAN(MAT) = NUMBER OF RANDOM POINTS FOR POISSON
C = OR TRANSIENT ANALYSIS
C =
C
WRITE(IOUTDV,2150)
C
DO 10 M=1,NUMMAT
C
READ(INDEV,1200)MAT, NUMEL(M),UCOND(M),VCOND(M),ANGLE,
$ NUMINT(M),NUMRAN(M),SPHT(M),DENS(M),QU(M),QV(M)
WRITE(IOUTDV,2200)MAT, NUMEL(M),UCOND(M),VCOND(M),ANGLE,
$ NUMINT(M),NUMRAN(M)
UCOS(M)=DCOS(ANGLE*0.017453293)
VCOS(M)=DCOS((ANGLE+90.0)*0.017453293)
C
10 CONTINUE
C
ISTART=1
ISTOP=NUMINT(1)
NSTART=1
**Sample Code**

```
NSTOP=NUMEL(1)

DO 432 M=1,NUMMAT

WRITE(IOUTDV,2250)M

DO 20 I=ISTART,ISTOP

READ(INDEV,1300)XCENT(I),YCENT(I)
WRITE(IOUTDV,2300)XCENT(I),YCENT(I)

20 CONTINUE
ISTART=ISTART+NUMINT(M)
ISTOP=ISTOP+NUMINT(M+1)
```

---

**Comments**

- NSTOP is set to NUMEL(1).
- A loop DO 432 M=1,NUMMAT is initialized.
- A WRITE statement is executed.
- Another loop DO 20 I=ISTART,ISTOP is initiated.
- READ and WRITE statements are included to read and write XCENT(I) and YCENT(I) coordinates.
- Loop continues with a CONTINUE statement.
- ISTART and ISTOP are updated.

---

**Explanation**

- The code snippet appears to be part of a larger program, possibly related to a computational or simulation task.
- It involves input and output operations, as well as iterative calculations.
- The use of loops and specific variables (ISTART, ISTOP, XCENT, YCENT) suggests a structured approach to processing data or coordinates.
C = - ELEMENTS ARE NUMBERED IN ASCENDING ORDER COUNTERCLOCKWISE
C = AROUND INTERIOR OF REGION(CLOCKWISE FOR AN EXTERIOR REGION)
C =
C = IEL = ELEMENT NUMBER( 1000 MAXIMUM )
C =
C = N1,N2 = NODE NUMBERS BORDERING THE ELEMENT;
C = N2 IS LOCATED IN COUNTERCLOCKWISE POSITION WITH RESPECT TO N1(OR CLOCKWISE FOR AN EXTERIOR REGION)
C =
C =
C = OMITTED ELEMENT CARDS ARE GENERATED; N1 AND N2 ARE GENERATED IN EVENLY SPACED INCREMENTS CONSISTENT WITH THE FIRST AND LAST CARDS IN THE GENERATION SEQUENCE
C =
C =
C =
C
WRITE(IOUTDV,2450)M
L=NSTART
C
160 READ(INDEV,1450)IEL,N1,N2,IKOD
   NDIF=IEL+1-L
   IF(IEL-L)180,1110,190
180 WRITE(IOUTDV,760)IEL
   IERROR=1
   GO TO 160
190 ND1=(N1-N11)/NDIF
   ND2=(N2-N22)/NDIF
1001 N1=N11+ND1
   N2=N22+ND2
1110 WRITE(IOUTDV,1610)L,N1,N2
   LM(L)=N1*10000+N2*10+IKOD
   N11=N1
   N22=N2
   L=L+1
C
   IF(IEL-L)1120,1001,1001
C
1120 IF(NSTOP+1.GT.L)GO TO 160
   NSTART=NSTART+NUMEL(M)
   NSTOP=NSTOP+NUMEL(M+1)
C
432 CONTINUE
C
RETURN
C
C ================
C =
C = FORMAT STATEMENTS =
C =
C ================
C
1000 FORMAT(18A4)
1100 FORMAT(4I5,F10.0,2I5)
540 FORMAT(2I5,4F10.0)
610 FORMAT(I5,5X,18,2X,5E16.6)
2150 FORMAT(/81('_')/17X,'MATERIAL PROPERTY AND '
$       ,'SUBREGIONAL PARAMETERS'/81('_')//)
1200 FORMAT(2I5,3F10.0/215,4F10.0)

C

1450 FORMAT(4I5)
2000 FORMAT (1X,120('*')/25X,18A4//1X,120('*')//
1  10X,'PROBLEM TYPE----',I4/
2  10X,'NUMBER OF NODAL POINTS----',I4/
3  10X,'NUMBER OF MATERIALS----',I4/
4  10X,'PLOT CODE----',I4//)
533 FORMAT (20H0 N.P. NO. CODE 11X,1HX,14X,1HY,14X,1HT,7X
$/IX,81( '_')/)
2450 FORMAT(/81('_')/17X,'ELEMENT DATA FOR REGION NUMBER',I4/
$     81('<')//10X,'ELEMENT NO. N1 N2',
$     /10X,58('<')//)
630 FORMAT(1X,10I5)
2120 FORMAT (24H1AXISYMMETRIC SOLID BODY )
2110 FORMAT (27H1TWO DIMENSIONAL PLANE BODY )
740 FORMAT(6X,I3,7X,I3,7X,I3,7X,I3,13X,E13.6,10X,E13.6)
750 FORMAT(20H0N.P. NO. CODE,11X,1HT)
760 FORMAT (10H0CARD NO. 14, 13H OUT OF ORDER )
770 FORMAT (13H0BAD CARD NO. 14)
2250 FORMAT(/10X,'INTERNAL POINTS OF MATERIAL NUMBER',I4,' ----'//
$     23X,'X',14X,'Y'/14X,35('<')//)
1610 FORMAT(8X,I10,2X,I13,112)
1300 FORMAT(2F10.0)
2300 FORMAT(15X,2E15.6)

2200 FORMAT(10X,'MATERIAL NUMBER------',I4//
   $  15X,'NUMBER OF ELEMENTS DEFINING REGION------',I4/
   $  15X,'THERMAL CONDUCTIVITY ALONG U-AXIS------',D15.6/
   $  15X,'THERMAL CONDUCTIVITY ALONG V-AXIS------',D15.6/
   $  15X,'ANGLE OF U-AXIS W/R TO GLOBAL X-AXIS------',D15.6/
   $  15X,'NUMBER OF INTERNAL POINTS SPECIFIED------',I4/
   $  15X,'NUMBER OF RANDOM POINTS SPECIFIED------',I5///)

C

END
SUBROUTINE SELFI(XI,YI,XJ,YJ,GTERM,ITYPE,MTYPE,K)
IMPLICIT REAL*8 (A-H,O-Z)
COMMON/TEMP/DELTA,NUMDLT,INTER
COMMON/IO/INDEV,IOUT,IPL0T,ISTOR1,ISTOR2,ISTOR3,ISTOR4,ISTOR5
COMMON/MTRL/UCO(20),VC0(20),SPHT(20),DENS(20),QU(20),
$QV(20),UC(20),VC(20),QX(20),QY(20),NUMRAN(20)
DATA NCALL/0/
C
C = SUBROUTINE SELFI
C = THIS SUBROUTINE COMPUTES THE TERMS OF THE
C = 'GKERN' MATRIX THAT CORRESPOND TO ELEMENTS THAT
C = INCLUDE THE SOURCE NODE.
C = CALLING ARGUMENTS -
C = XI,YI,XJ,YJ - COORDINATES OF THE ENDNODES
C = GTERM - TERM OF 'GKERN' COMPUTED AND RETURNED
C = ITYPE - CODE FOR PROBLEM TYPE
C = = 0 FOR PLANAR GEOMETRY
C = = 1 FOR AXISYMMETRIC PROBLEM
C = MTYPE - MATERIAL IDENTIFICATION NUMBER

C =

C = K - CODE THAT INDICATES WHETHER NODE 'I' OR 'J' IS THE NODE UNDER CONSIDERATION

C = = 1 FOR NODE 'I'

C = = 2 FOR NODE 'J'

C =

UCOND=UCO(MTYPE)
VCOND=VCO(MTYPE)
UCOS=UC(MTYPE)
VCOS=VC(MTYPE)

I=ITYPE+1

C =

C = TRANSFORM COORDINATES TO MATERIAL AXIS SYSTEM

C =

C =

X1=XI*UCOS+YI*VCOS
X2=XJ*UCOS+YJ*VCOS
Y1=-XI*VCOS+YI*UCOS
Y2=-XJ*VCOS+YJ*UCOS

GO TO (10,20),I
C

C = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
C =
C = TWO-DIMENSIONAL PLANAR ANALYSIS =
C =
C = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =

C

10 DX=X2-X1

DY=Y2-Y1

ALENTH=DSQRT(DX*DX+DY*DY)

ELCOS=DX/ALENTH

ELSIN=DY/ALENTH

IF(DELTA.NE.0.0)GO TO 15

C

C = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
C =
C = STEADY-STATE TERM COMPUTATION =
C =
C = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =

C

CONS=0.5

IF(K.EQ.1)CONS=1.5

FACT=ALENTH*0.5/DSQRT(UCOND*VCOND)

ARG=(ELCOS*ELCOS/UCOND)+(ELSIN*ELSIN/VCOND)

GTERM=FACT*(CONS-DLOG(ALENTH*DSQRT(ARG)))

RETURN
15 CONTINUE

C

20 CONTINUE

C

C = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
C
C = AXISYMMETRIC STEADY-STATE TERM =
C
C = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
C

NCALL=NCALL+1
NRAN=500
IF(NCALL.EQ.1)WRITE(IOUT,200)NRAN
IF(NUMDLT.NE.0)GO TO 40
CONFAC=2./DSQRT(UCOND*VCOND)
CONRAT=UCOND/VCOND
DX=X2-X1
DY=Y2-Y1
ALENTH=DSQRT(DX*DX+DY*DY)
ELCOS=(X1-X2)/ALENTH
ELSIN=(Y2-Y1)/ALENTH
G=0.0
G2=0.0
N=0

C

61 N=N+1

CALL RANDOM(0.0,ALENTH)
DEN=(2.0*X1-B*ELCOS)**2+CONRAT*B*B*ELSIN*ELSIN
DEN2=DSQRT(DEN)
PHI=B/ALENTH
IF(K.EQ.1)PHI=1.0-PHI
ARG=4.0*X1*(X1-B*ELCOS)/DEN
GT=(X1-B*ELCOS)*ELLPTC(ARG,1)*PHI/DEN2
G=G+GT
G2=G2+GT*GT
IF(N.NE.NRAN)GO TO 61
GTERM=ALENTH*G*CONFAC/NRAN
GSQUAR=ALENTH*G2*CONFAC*CONFAC/NRAN
SD=DSQRT((ALENTH*GSQUAR-GTERM*GTERM)/NRAN)
WRITE(IOUT,100)GTERM,SD
100 FORMAT(10X,2E15.7)
200 FORMAT//(10X,'DIAGONAL TERMS',3X,'STNDRD. DEV.',/10X,30('—'))
RETURN
40 CONTINUE
RETURN
C
END
SUBROUTINE FORMGH(X,Y,GKERN,HKERN,FIXBND,KODE,SOLV,B,NUMRAN,SD)
C
IMPLICIT REAL*8 (A-H,O-Z)
DIMENSION GKERN(NUMNP,NUMNP),HKERN(NUMNP,NUMNP),SOLV(NUMNP),
$ X(1),Y(1),FIXBND(1),KODE(1),B(1),NUMRAN(1),SD(1)
COMMON/CNTRL/ITYPE,NUMEL(20),NUMINT(20),NUMMAT,NTOT,NUMNP
COMMON/IO/INDEV,IOUTDV,IPLOT,ISTOR1,ISTOR2,ISTOR3,ISTOR4,ISTOR5
COMMON/LMENT/LM(1000)
C
C  = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
C  = SUBROUTINE FORMGH
C  =
C  =
C  =
C  = THIS SUBROUTINE FORMS THE SYSTEM MATRICES AND ARRANGES
C  = THE EQUATIONS TO BE SOLVED
C  =
C  = CALLING ARGUMENTS -
C  =
C  = X,Y - VECTORS CONTAINING THE NODAL COORDINATES
C  =
C  = GKERN,HKERN - ARRAYS CONTAINING THE SYSTEM COEFFICIENTS
C  = TO BE CALCULATED IN THIS ROUTINE
C  =
C  = FIXBND - VECTOR OF BOUNDARY CONDITIONS
C  =
C  = KODE - VECTOR OF BOUNDARY CONDITION CODES
C  =
C = SOLV - VECTOR WHICH WILL CONTAIN SOLUTION SET
C = B - VECTOR TO CONTAIN POISSON TERMS
C = NUMRAN - ARRAY CONTAINING NUMBERS OF RANDOM POINTS IN ONE-TO-ONE CORRESPONDENCE WITH MATERIAL NUMBERS
C = SD - VECTOR TO CONTAIN STANDARD DEVIATIONS OF B-TERMS

DO 10 I=1,NUMNP
   B(I)=0.0
   SD(I)=0.0
   DO 10 J=1,NUMNP
      GKERN(I,J)=0.0
      HKERN(I,J)=0.0
10 CONTINUE

NSTART=1
NSTOP=NUMEL(1)
NCOUNT=0
C = COMPUTATION OF SYSTEM COEFFICIENT MATRICES =
C =
C==============================================================================

DO 999 M=1,NUMMAT
    NRAN=NUHRAN(M)
    NEL=NUMEL(M)
    IF(NRAN.EQ.0)GO TO 22

    CALL EXTREM(X,Y,XMIN,XMAX,YMIN,YMAX,NEL,LM)

22 CONTINUE

DO 899 NN=NSTART,NSTOP

    CALL RECALL(LM(NN),N1,N2,IKOD)
    XP=X(N1)
    YP=Y(N1)

    IF(NRAN.EQ.0)GO TO 66
    CALL POISN(XP,YP,X,Y,STERM,NRAN,NEL,BTERM,$
        XMIN,XMAX,YMIN,YMAX,ITYPE,M)
    B(N1)=B(N1)+BTERM
    SD(N1)=SD(N1)+STERM

66 DO 899 N=NSTART,NSTOP
C
CALL RECALL(LM(N),N3,N4,IKOD)

SGN=1.0

IF(IKOD.EQ.1)SGN=-1.0

IF(N1.NE.N3)GO TO 19

CALL SELFI(X(N3),Y(N3),X(N4),Y(N4),GI,ITYPE,M,1)
CALL SELFI(X(N3),Y(N3),X(N4),Y(N4),GJ,ITYPE,M,2)
GKERN(N1,N3)=GKERN(N1,N3)+GI*SGN
GKERN(N1,N4)=GKERN(N1,N4)+GJ*SGN

GO TO 899

C

19 IF(N1.NE.N4)GO TO 89

C

CALL SELFI(X(N3),Y(N3),X(N4),Y(N4),GI,ITYPE,M,1)
CALL SELFI(X(N3),Y(N3),X(N4),Y(N4),GJ,ITYPE,M,2)
GKERN(N1,N3)=GKERN(N1,N3)+GJ*SGN
GKERN(N1,N4)=GKERN(N1,N4)+GI*SGN

GO TO 899

C

89 CONTINUE

C

CALL INTEG(XP,YP,X(N3),Y(N3),X(N4),Y(N4),HI,HJ,GI,GJ,ITYPE,M)
GKERN(N1,N3)=GKERN(N1,N3)+GI*SGN
GKERN(N1,N4)=GKERN(N1,N4)+GJ*SGN
HKERN(N1,N3)=HKERN(N1,N3)+HI
HKERN(N1,N4)=HKERN(N1,N4)+HJ

C

HKERN(N1,N1)=HKERN(N1,N1)-HI-HJ
899 CONTINUE

NCOUNT = NCOUNT + NEL
NSTART = NSTART + NEL
NSTOP = NSTOP + NUMEL(M+1)

999 CONTINUE

C = ARRANGEMENT OF EQUATIONS FOR SOLUTION

DO 161 J = 1, NUMNP
   IF (KODE(J)) 140, 140, 161
140 DO 161 I = 1, NUMNP
   GIJ = GKERN(I, J)
   GKERN(I, J) = -HKERN(I, J)
   HKERN(I, J) = -GIJ

161 CONTINUE

C = INITIALIZE THE SOLUTION VECTOR
IF(NUMRAN(NUMMAT).NE.0)WRITE(IOUTDV,501)

DO 180 I=1,NUMNP
IF(NUMRAN(NUMMAT).NE.0)WRITE(IOUTDV,101)I,B(I),SD(I)
   SOLV(I)=B(I)
DO 180 J=1,NUMNP
   SOLV(I)=SOLV(I)+HKERN(I,J)*FIXBND(J)
C
180 CONTINUE
C
501 FORMAT(/10X,'POISSON TERMS',5X,'NODE',5X,'POISSON TERM',
     $ 9X,'STANDARD DEVIATION')
101 FORMAT(29X,I4,5X,E15.7,5X,E15.7)
RETURN
END
SUBROUTINE EXTREM(X,Y,XMIN,XMAX,YMIN,YMAX,NUMEL,LM)

REAL*8 X,Y,XMIN,XMAX,YMIN,YMAX

C

C =========================================================
C = SUBROUTINE EXTREM
C = THIS SUBROUTINE COMPUTES THE EXTREME COORDINATES OF
C = A PLANE CURVE.
C = CALLING ARGUMENTS -
C = X,Y - ARRAYS OF COORDINATE PAIRS DEFINING CURVE
C = XMIN,XMAX,
C = YMIN,YMAX - EXTREME VALUES RETURNED BY PROGRAM
C = NUMEL - NUMBER OF ELEMENTS DEFINING CURVE
C = LM - INTEGER ARRAY
C = CONTAINING ELEMENT DATA
C
C =========================================================

C

DIMENSION X(1),Y(1),LM(1)

C

CALL RECALL(LM(1),I,J,KOD)
\[
X_1 = X(I) \\
X_2 = X_1 \\
Y_1 = Y(I) \\
Y_2 = Y_1
\]

C

DO 10 K = 2, NUMEL

C

CALL RECALL(LM(K), I, J, KOD)

XI = X(I) \\
YI = Y(I) \\
X_1 = \text{A}MIN(X_1, XI) \\
X_2 = \text{A}MAX(X_2, XI) \\
Y_1 = \text{A}MIN(Y_1, YI) \\
Y_2 = \text{A}MAX(Y_2, YI)

C

10 CONTINUE

C

XMIN = X_1 \\
XMAX = X_2 \\
YMIN = Y_1 \\
YMAX = Y_2

C

RETURN

END
SUBROUTINE INTEG(XP,YP,XI,YI,XJ,YJ,HI,HJ,GI,GJ,ITYPE,MTYPE)

C

C = SUBROUTINE INTEG

C = THIS SUBROUTINE INTEGRATES THE TERMS OF THE ASSEMBLY MATRICES EXCEPT FOR THE DIAGONAL ELEMENTS

C = CALLING ARGUMENTS -

C = XP,YP = COORDINATES OF SOURCE POINT

C = XI,YI, XJ,YJ = COORDINATES OF ENDNODES OF ELEMENT BEING INTEGRATED IN CCW ORDER

C = HI,HJ = MATRIX TERMS OF 'HKERN' CORRESPONDING TO ENDNODE COORDINATES

C = GI,GJ = MATRIX TERMS OF 'GKERN' CORRESPONDING TO ENDNODE COORDINATES

C = ITYPE = 0 FOR 2-D PLANAR PROBLEM = 1 FOR AXISYMMETRIC PROBLEM

C = MTYPE = MATERIAL IDENTIFICATION NUMBER
IMPLICIT REAL*8(A-H,O-Z)

COMMON/MTRL/UCOND(20),VCOND(20),SPHT(20),DENS(20),QU(20),QV(20),
                  UC0S(20),VC0S(20),QX(20),QY(20),NUMRAN(20)

COMMON/TEMP/DELTA,NUMDLT,INTER

DIMENSION GK(4),GW(4)

DATA GK/0.86113631,-0.86113631,0.33998104,-0.33998104/,
     GW/0.34785485,0.34785485,0.65214515,0.65214515/

C

C = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =

C = TRANSFORM COORDINATES TO ORTHOTROPIC AXES =

C = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =

C

U=UCOS(MTYPE)
V=VCOS(MTYPE)
UC=UCOND(MTYPE)
VC=VCOND(MTYPE)
COND=D SQRT(UC*VC)

C

XIT=XI*U+YI*V
XJT=XJ*U+YJ*V
YIT=-XI*V+YI*U
YJT=-XJ*V+XJ*U
\[
XPT = XP \times U + YP \times V \\
YPT = -XP \times V + YP \times U
\]

\[
I = ITYPE + 1
\]

\[
\begin{align*}
DX &= XJT - XIT \\
DY &= YJT - YIT \\
AX &= DX \times 0.5 \\
AY &= DY \times 0.5 \\
BX &= (XJT + XIT) \times 0.5 \\
BY &= (YJT + YIT) \times 0.5
\end{align*}
\]

\[
AMET = DSQRT(DY \times DY + DX \times DX) \\
ANCOS = DY / AMET \\
ANSIN = -DX / AMET
\]

\[
\begin{align*}
GI &= 0.0 \\
GJ &= 0.0 \\
HI &= 0.0 \\
HJ &= 0.0
\end{align*}
\]

\[
\text{GO TO (10,20),} \ I
\]

\[
\begin{align*}
C \quad \text{====================================} \\
C = = \text{TWO-DIMENSIONAL PLANAR ANALYSIS} = \\
C = . \quad =
\end{align*}
\]
10 IF(NUMDLT.NE.0)GO TO 16
   DO 25 I=1,4
   
   GKI=GK(I)
   XG=AX*GKI+BX
   YG=AY*GKI+BY
   DXR=XG-XPT
   DYR=YG-YPT
   R2=(DXR*DXR/UC)+(DYR*DYR/VC)
   
   G=-GW(I)*DLOG(R2)*AMET*0.25/COND
   GI=GI-(GKI-1.0)*G*0.5
   GJ=GJ+(GKI+1.0)*G*0.5
   
   H=GW(I)*0.5*(DXR*ANCOS+DYR*ANSIN)*AMET/(COND*R2)
   HI=HI+(GKI-1.0)*H*0.5
   HJ=HJ-(GKI+1.0)*H*0.5
   
   25 CONTINUE
   RETURN
   
   CONTINUE
   20 IF(NUMDLT.NE.0)GO TO 26
   
   ===============
C = AXISYMMETRIC ANALYSIS
C =
C ===============================

CONRAT=UC/VC
CON2=DSQRT(CONRAT)
UCRT=DSQRT(UC)

DO 35 I=1,4

GKI=GK(I)
RG=AX*GKI+BX
ZG=AY*GKI+BY
DZR=ZG-YPT
RADD=RG+XPT
DRR=RG-XPT
DEN=RADD*RADD+CONRAT*DZR*DZR
DENRT=DSQRT(DEN)
ARG=4.0*XPT*RG/DEN
ELL1=ELLPTC(ARG,1)
ELL2=ELLPTC(ARG,2)

G=GW(I)*RG*ELL1*AMET/(COND*DENRT)
GI=GI-(GKI-1.0)*G*0.5
GJ=GJ+(GKI+1.0)*G*0.5
BOT = DRR * DRR + CONRAT * DZR * DZR

ZTERM = VC * UCRT * CON2 * DZR * ELL2 / BOT

RTERM = 0.0

IF (RG .EQ. 0.0) GO TO 23

R1 = (RG * RG - XPT * XPT - CONRAT * DZR * DZR) * ELL2 / BOT

RTERM = (0.5 * UC * UCRT / RG) * (ELL1 + R1)

23 CONTINUE

QSTAR = 2.0 * (RTERM * ANCOS + ZTERM * ANSIN) / (COND * DENRT)

H = -GW(I) * RG * QSTAR * AMET * 0.5

HI = HI - (GKI - 1.0) * H * 0.5

HJ = HJ + (GKI + 1.0) * H * 0.5

35 CONTINUE

RETURN

C

26 CONTINUE

RETURN

END
SUBROUTINE RANDOM(ZP,ZMIN,ZMAX)

REAL*8 ZP,ZMIN,ZMAX

DATA IXSEED/137462873/

C = SUBROUTINE RANDOM
C = THIS SUBROUTINE COMPUTES A PSEUDO-RANDOM
C = COORDINATE WITHIN A LINEAR DOMAIN.
C = THE SEED VALUE, WHICH MUST BE AN ODD INTEGER,
C = MAY BE ALTERED BY CHANGING THE DATA
C = STATEMENT ABOVE.
C = CALLING ARGUMENTS
C = ZP - RANDOM COORDINATE RETURNED
C = BY PROGRAM
C = ZMIN,ZMAX - DEFINE THE EXTREMITIES OF THE
C = LINEAR REGION TO WHICH THE
C = RANDOM POINT IS CONFINED
C =

IYSEED=IXSEED*65539
IF(IYSEED)1,1,2

1  IYSEED=IYSEED+2147483647+1
2  ZP=IYSEED
   IXSEED=IYSEED
   ZP=ZP*0.4656613E-9

C
C
C
C
ZP=(ZMAX-ZMIN)*ZP+ZMIN
C
RETURN
C
END
SUBROUTINE OUTIN(X,Y,XP,YP,NUMEL,FACTR)
REAL*8 X,Y,XP,YP,FACTR
COMMON/LMEN/LM(1000)

C

C = SUBROUTINE OUTIN
C = THIS SUBROUTINE DETERMINES WHETHER OR NOT A POINT
C = RESIDES WITHIN A TWO-DIMENSIONAL DOMAIN
C = CALLING ARGUMENTS -
C = X,Y - ARRAYS CONTAINING BOUNDARY
C = COORDINATE PAIRS
C = XP,YP - COORDINATES OF POINT IN QUESTION
C = NUMEL - NUMBER OF BOUNDARY ELEMENTS
C = FACTR - FACTOR COMPUTED BY SUBROUTINE
C = 0.0 IF POINT IS NOT IN DOMAIN
C = 1.0 IF POINT IS INSIDE DOMAIN
C = 
C = DIMENSION X(1),Y(1)
SUM = 0.0

DO 10 N = 1, NUMEL

CALL RECALL(LM(N), I, J, IKOD)

AX = X(I) - XP
BX = X(J) - XP
AY = Y(I) - YP
BY = Y(J) - YP

C = FORM CROSS PRODUCT OF THE VECTORS TO
C = DETERMINE SIGN OF ANGULAR SEGMENT
CP=AX*BY-AY*BX
SGN=SIGN(1.0,CP)
IF(CP.EQ.0.0)SGN=0.0

C
C = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
C = COMPUTE DOT PRODUCT OF TWO VECTORS FOR =
C = PURPOSE OF OBTAINING THE ACTUAL ANGLE =
C = SUBTENDED BY THE BOUNDARY INCREMENT =
C =
C = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
C

DOT=AX*BX+AY*BY
A2=AX*AX+AY*AY
B2=BX*BX+BY*BY
PROD=A2*B2

C
C = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
C
C ANG=0.0
IF(PROD.NE.0.0)ANG=ACOS(DOT/SQRT(PROD))
SUM = SUM + ANG * SGN

10 CONTINUE

FCTR = SUM * 0.159154943
IF (ABS(1.0 - FCTR).LE.1.E-02) FCTR = 1.0
IF (ABS(FCTR).LE.1.E-02) FCTR = 0.0

RETURN

END
SUBROUTINE OUTPUT(X,Y,SOLV,FIXBND,XCENT,YCENT,SOLUT,BNDINT,TIME)

C
C ==============
C = SUBROUTINE OUTPUT
C =
C = THIS SUBROUTINE WRITES OUT THE RESULTS OF THE COMPUTATION
C =
C = CALLING ARGUMENTS -
C =
C = X,Y - VECTORS OF BOUNDARY COORDINATES
C =
C = FIXBND - VECTOR CONTAINING POTENTIALS AT NODES
C =
C = SOLV - VECTOR CONTAINING FLUXES AT NODES
C =
C = XCENT,YCENT - VECTORS CONTAINING INTERNAL COORDINATES
C =
C = SOLUT - VECTOR CONTAINING POTENTIALS AT INTERNAL NODES
C =
C =
C ==============
C
IMPLICIT REAL*8(A-H,O-Z)
DIMENSION X(1),Y(1),FIXBND(1),SOLV(1),XCENT(1),YCENT(1),
SOLUT(1),BNDINT(1)
COMMON/CNTRL/ITYPE,NUMEL(20),NUMINT(20),NUMMAT,NTOT,NUMNP
COMMON/TEMP/DELTA,NUMDLT,INTER
COMMON/IO/INDEV,IOUT,IPLLOT,I1,I2,I3,I4,I5

WRITE(IOUT,100)
IF(NUMDLT.NE.0)WRITE(IOUT,500)TIME
DO 5 I=1,NUMNP
   WRITE(IOUT,200)I,X(I),Y(I),FIXBND(I),SOLV(I)
5 CONTINUE

NSTART=1
NSTOP=NUMINT(1)
DO 10 M=1,NUMMAT

WRITE(IOUT,350)M

DO 20 I=NSTART,NSTOP

WRITE(IOUT,300)I,XCENT(I),YCENT(I),SOLUT(I)

20 CONTINUE
NSTART=NSTART+NSTOP
NSTOP=NSTOP+NUMINT(M+1)
10 CONTINUE

RETURN

C = = = = = = = = = = = = =
C =
C = FORMAT STATEMENTS =

C =

C = = = = = = = = = = = = = = = = = = = = = = = = = = =

C

100 FORMAT(10X//100('-')//11X,'SOLUTION'//12X,
        $ 'BOUNDARY NODE',11X,'X',23X,'Y',
        $ 19X,'POTENTIAL',10X,'POTENTIAL DERIVATIVE'/)

200 FORMAT(13X,I3,4(10X,E14.7))

350 FORMAT(//13X,'INTERNAL POTENTIAL COMPUTATION FOR REGION',I4//
        $ 11X,'NODE',11X,'X',23X,'Y',19X,'POTENTIAL'//)

300 FORMAT(11X,I4,3(10X,E14.7))

500 FORMAT(//'TIME=',E15.7/)
SUBROUTINE POISN(XP, YP, X, Y, SD, NRAN, NEL, BTERM, XMIN, XMAX, YMIN, YMAX, ITYPE, MTYPE)

IMPLICIT REAL*8 (A-H, O-Z)

C
C ==============================================================
C = SUBROUTINE POISN
C = THIS SUBROUTINE COMPUTES THE POISSON TERMS OF THE ASSEMBLY
C = MATRIX AND THE CORRESPONDING STANDARD DEVIATIONS -
C = CALLING ARGUMENTS -
C = XP, YP = COORDINATES OF BOUNDARY POINT IN QUESTION
C = IDEV = INPUT CHANNEL CONTAINING ELEMENT DATA
C = X, Y = VECTORS OF BOUNDARY COORDINATE PAIRS
C = SD = STANDARD DEVIATION CORRESPONDING TO POISSON INTEGRATION TERM
C = NRAN = NUMBER OF RANDOM INTEGRATION POINTS
C = NEL = NUMBER OF BOUNDARY ELEMENTS SURROUNDING REGION
C = BTERM = TERM OF POISSON MATRIX RETURNED BY PROGRAM
C =

C = XMIN, XMAX,

C = YMIN, YMAX = EXTREMITIES OF PLANE REGION

C =

C = ITYPE = PROBLEM TYPE CODE

C = = 0 FOR PLANAR ANALYSIS

C = = 1 FOR AXISYMMETRIC ANALYSIS

C =

C = MTYPE = MATERIAL IDENTIFICATION NUMBER

C =

C =-------------------------------------------------------------

C

COMMON/TEMP/DELTA, NUMDLT, INTER

COMMON/MTRL/UCO(20), VCO(20), SPHT(20), DENS(20), QU(20),

$ QV(20), UC(20), VC(20), QX(20), QY(20), NUMRAN(20)

DIMENSION X(1), Y(1)

COMMON/LMENT/LM(1000)

C

I = ITYPE + 1

UCOND = UCO(MTYPE)

VCOND = VCO(MTYPE)

UCOS = UC(MTYPE)

VCOS = VC(MTYPE)

C

SUM = 0.0

SUMV = 0.0

C
C = TRANSFORM COORDINATES =
C =
C ==========================

X1 = XP * UCOS + YP * VCOS
Y1 = -XP * VCOS + YP * UCOS

N = 0
98 N = N + 1
    IF (N.GT.NRAN) GO TO 66
C
C = COMPUTE RANDOM POINT IN =
C = PARENT RECTANGLE =
C =
C ===========================
C
18 CALL RANDOM(XR, XMIN, XMAX)
    CALL RANDOM(YR, YMIN, YMAX)
    CALL OUTIN(X, Y, XR, YR, NEL, FACTR)
    IF (FACTR .NE. 1.0) GO TO 18
C
C =
C =
C = TRANSFORM COORDINATES & =
C = COMPUTE METRIC
C =
C = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
C
C XR1=XR*UCOS+YR*VCOS
YR1=-XR*VCOS+YR*UCOS
C
DX=X1-XR1
DY=Y1-YR1
ALENTH=(DX*DX/UCOND)+(DY*DY/VCOND)
C
GO TO (10,20),I
10  IF(INTER.NE.0)GO TO 15
C
C = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
C = TWO-DIMENSIONAL PLANAR COMPUTATION
C =
C = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
C
C USTAR=-0.5*DLOG(ALENTH)/DSQRT(UCOND*VCOND)
GO TO 30
15  CONTINUE
20  CONTINUE
25  CONTINUE
30  TERM=USTAR*F(XR,YR,XR1,YR1,MTYPE)
SUM = SUM + TERM
SUMV = SUMV + TERM * TERM

C
GO TO 98
C

66 CONTINUE
C
C
C == = = = = = = = = = = = = = = = = = = = = = =
C

A = 0.0
DO 77 J = 1, NEL
   CALL RECALL(LM(J), I, K, IKOD)
   RX = XR - X(I)
   RY = YR - Y(I)
   DX = X(K) - X(I)
   DY = Y(K) - Y(I)
   A = A + RY * DX - RX * DY
C
77 CONTINUE
C

AREA = 0.5 * A
ASUM = AREA * SUM
BTERM = ASUM / NRAN
SDTERM = AREA * SUMV / NRAN

VAR1 = (AREA * SDTERM - BTERM * BTERM) / NRAN

SD = DSQRT(VAR1)

RETURN

END
SUBROUTINE SOLVE(A,S,N,DET,IOUT)

IMPLICIT REAL*8(A-H,O-Z)

C
C
C
C
C
C
C
C
C
C
C
C
C
C
C
C
C
C
C
C
C
C
C
C
C
C
C
C
C
C
C
C
C = IMPLICIT REAL*8(A-H,O-Z)

C = SUBROUTINE SOLVE

C = THIS SUBROUTINE SOLVES A SYSTEM OF LINEAR EQUATIONS BY

C = GAUSSIAN ELIMINATION USING PARTIAL PIVOTING

C = CALLING ARGUMENTS -

C =

C = A = COEFFICIENT MATRIX

C =

C = S = VECTOR OF UNKNOWNS ,TO CONTAIN SOLUTION VECTOR LATER

C =

C = N = NUMBER OF NODAL POINTS IN PROBLEM

C =

C = DET = VALUE OF DETERMINANT

C =

C = IOUT = OUTPUT CHANNEL NUMBER

C =

C = IMPLICIT REAL*8(A-H,O-Z)

C =

C = DIMENSION A(N,N),S(1)

C1=N-1

C =

C = DO 100 K=1,N1

C =
K1 = K + 1
C = A(K, K)
IF(DABS(C) - 0.000001) 1, 1, 3
1 DO 7 J = K1, N
     C
     =
     =
     = ATTEMPT INTERCHANGE OF ROWS IF DIAGONAL IS WEAK =
     =
     =
     =
     =
     =
     IF(DABS(A(J, K)) - 0.000001) 7, 7, 5
5 DO 6 L = K, N
     C = A(K, L)
     A(K, L) = A(J, L)
6 A(J, L) = C
     C = S(K)
     S(K) = S(J)
     S(J) = C
     C = A(K, K)
     GO TO 3
7 CONTINUE
8 WRITE (IOUT, 2) K
2 FORMAT(10X, ' SINGULARITY IN ROW', I5)
     DET = 0.0
     GO TO 300
C
=DIVIDE ROW BY DIAGONAL COEFFICIENT=

3  

DO 4 J=K1,N

4

A(K,J)=A(K,J)/C

S(K)=S(K)/C

=ELIMINATE UNKNOWN X(K) FROM ROW I=

DO 10 I=K1,N

C=A(I,K)

DO 9 J=K1,N

9

A(I,J)=A(I,J)-C*A(K,J)

10  S(I)=S(I)-C*S(K)

100 CONTINUE

=COMPUTE LAST UNKNOWN=

==
DO 200 L=1,N1
  K=N-L
  K1=K+1
  DO 200 J=K1,N
  200  S(K)=S(K)-A(K,J)*S(J)

DET=1.0
  DO 250 I=1,N
  250  DET=DET*A(I,I)

IF (DABS(A(N,N))-0.000001)8,8,101

101 S(N)=S(N)/A(N,N)
300 RETURN
C
END
SUBROUTINE TERNAL(FIXBND,SOLV,KODE,XCENT,YCENT,SOLUT,X,Y,B)

IMPLICIT REAL*8(A-H,O-Z)

DIMENSION FIXBND(1),SOLV(1),KODE(1),XCENT(1),YCENT(1),
$ SOLUT(1),X(1),Y(1),B(1)

C
C =============================================================
C = SUBROUTINE TERNAL
C = THIS SUBROUTINE COMPUTES THE POTENTIAL VALUES AT
C = THE INTERNAL POINTS SPECIFIED BY THE USER.
C = CALLING ARGUMENTS -
C = FIXBND - VECTOR OF FIXED BOUNDARY CONDITIONS
C = SOLV - VECTOR OF CONSTANTS CORRESPONDING TO THE
C = CURRENT STATE OF THE ASSEMBLY MATRIX
C = KODE - VECTOR OF BOUNDARY CONDITION CODES
C = = 0 IF FLUX IS SPECIFIED
C = = 1 IF TEMPERATURE IS SPECIFIED
C = XCENT,YCENT - COORDINATE VECTORS CONTAINING
C = INTERNAL POINTS
C =
\[ \mathbf{C} = \text{SOLUT} - \text{EVENTUAL SOLUTION VECTOR} \]

\[ \mathbf{C} = \mathbf{X}, \mathbf{Y} - \text{VECTORS OF BOUNDARY COORDINATES} \]

\[ \mathbf{C} = \mathbf{B} - \text{VECTOR OF POISSON TERMS} \]

---

\[ \text{DATA 002PI/0.1591549/} \]

\[ \text{COMMON/MTRL/UCO(20),VO(20),SPHT(20),DENS(20),QU(20),} \]

\[ \text{QV(20),UC(20),VC(20),QX(20),QY(20),NUMRAN(20)} \]

\[ \text{COMMON/CNTRL/ITYPE,NUMEL(20),NUMINT(20),NUMMAT,NTOT,NUMNP} \]

\[ \text{COMMON/IO/INDEV,IOUT,IPLOT,ISTOR1,ISTOR2,ISTOR3,ISTOR4,ISTOR5} \]

\[ \text{COMMON/LEMNT/LM(1000)} \]

\[
\text{BTERM} = 0.0 \\
\text{DO 20 I=1,NUMNP} \\
\quad \text{IF(KODE(I))10,10,20} \\
10 \quad \text{C=FIXBND(I)} \\
\quad \text{FIXBND(I)=SOLV(I)} \\
\quad \text{SOLV(I)=C} \\
20 \quad \text{CONTINUE} \\
\]

---

\[ \mathbf{C} = \text{COMPUTATION OF INTERNAL POTENTIALS} \]
C

C

ISTART = 1
ISTOP = NUMINT(1)
NSTART = 1
NSTOP = NUMEL(1)

C

DO 50 M = 1, NUMMAT

NRAN = NUMRAN(M)
NEL = NUMEL(M)

IF(NRAN .NE. 0) WRITE(IOUT, 501) M

C

CALL EXTREM(X, Y, XMIN, XMAX, YMIN, YMAX, NEL, LM)

C

DO 40 I = ISTART, ISTOP

SOLUT(I) = 0.0
XP = XCENT(I)
YP = YCENT(I)

IF(NRAN .EQ. 0) GO TO 28

CALL POISN(XP, YP, X, Y, SD, NRAN, NEL, BTERM,
$ XMIN, XMAX, YMIN, YMAX, ITYPE, M)

WRITE(IOUT, 101) I, BTERM, SD

28 CONTINUE

DO 30 J = NSTART, NSTOP

CALL RECALL(LM(J), N1, N2, IKOD)

CALL INTEG(XP, YP, X(N1), Y(N1), X(N2), Y(N2), HI, HJ,
$ GI, GJ, ITYPE, M)
C

30  SOLUT(I) = SOLUT(I) + SOLV(N1) * GI + SOLV(N2) * GJ

$ -FIXBND(N1) * H1 - FIXBND(N2) * HJ

40  SOLUT(I) = (SOLUT(I) - BTERM) * 002PI

ISTART = ISTART + ISTOP

ISTOP = ISTOP + NUMINT(M+1)

NSTART = NSTART + NSTOP

NSTOP = NSTOP + NUMEL(M+1)

C

50  CONTINUE

C

RETURN

101  FORMAT(29X, I4, 9X, E15.7, 5X, E15.7)

501  FORMAT(/10X, 'POISSON TERMS FOR REGION ---', 13, '/23X,

$ 'INTERNAL NODE', 5X, 'POISSON TERM',

$ 9X, 'STANDARD DEVIATION')

END
SUBROUTINE RECALL(LM,N1,N2,IKOD)

C
C  = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
C  =
C  = SUBROUTINE
C  =
C  = RECALL
C  =
C  = THIS SUBROUTINE RECOVERS
C  = THE ELEMENT DATA FROM
C  = THE ELEMENT CARDS.
C  =
C  =
C  = CALLING ARGUMENTS -
C  =
C  = LM - ENCODED ELEMENT DATA -
C  =
C  = N1,N2 - ELEMENT ENDNODES -
C  =
C  = = = = = = = = = = = = = = = = = = = = = = = = = = = = =

N1=LM/10000
I=LM-N1*10000
N2=I/10
RETURN
END
SUBROUTINE PIKSHR(X,Y,XCENT,YCENT,KODE)
REAL*8 X,Y,XCENT,YCENT,XMIN,XMAX,YMIN,YMAX
DIMENSION X(1),Y(1),XCENT(1),YCENT(1),KODE(1)

C
C    =SUBROUTINE PIKSHR
C
C    THIS SUBROUTINE PLOTS THE BOUNDARY MESH.
C
C    CALLING ARGUMENTS -
C
C    X,Y - ARRAYS OF BOUNDARY COORDINATES
C
C    XCENT,YCENT - ARRAYS OF INTERNAL POINTS
C
C    KODE - ARRAY OF BOUNDARY CONDITION CODES
C
C    COMMON/CNTRL/ITYPE,NUMEL(20),NUMINT(20),NUMMAT,NTOT,NUMNP
C    COMMON/LMEN/LM(1000)
C
ORGN=10.0
CALL IDENT
CALL VTHICK(3)
CALL PLOT(ORGN,5.0,-3)
DO 5 ILOOP = 1, NUMMAT

C

NEL = NUMEL(ILOOP)
NINT = NUMINT(ILOOP)
CALL EXTREM(X, Y, XMIN, XMAX, YMIN, YMAX, NEL, LM)

C

XD = XMAX - XMIN
YD = YMAX - YMIN
XM = XMIN
YM = YMIN
FAC = 5./AMAX1(XD, YD)

C

C

DO 5 M = 1, 2

C

CALL PLOT(ORGN, 0.0, -3)
CALL SYMBOL(0.0, -0.5, .14, 5, 0.0, -1)
CALL SYMBOL(0.28, -0.5, .14, 'INDICATES FLUX SPECIFIED', 0.0, 24)
CALL SYMBOL(0.0, -1.0, .14, 5, 0.0, -1)
CALL SYMBOL(0.0, -1.0, .14, 4, 0.0, -1)
CALL SYMBOL(0.28, -1.0, .14, 'INDICATES POTENTIAL SPECIFIED', 0.0, 29)

C

DO 10 L = 1, NEL

C

CALL RECALL(LM(L), I, J, KOD)
X1 = FAC*(X(I) - XM)
X2=FAC*(X(J)-XM)
Y1=FAC*(Y(I)-YM)
Y2=FAC*(Y(J)-YM)
CALL VECTOR(X1,Y1,X2,Y2)
CALL SYMBOL(X2,Y2,.14,5,0.0,-1)
FJ=J
DX=X2-X1
DY=Y2-Y1
AMET=SQRT(DX*DX+DY*DY)
ANCOS=DY/AMET
ANSIN=-DX/AMET
XN=0.1414*ANCOS+X2
YN=0.1414*ANSIN+Y2
IF(M.EQ.1)CALL NUMBER(XN,YN,.07,FJ,0.0,-1)
IF(KODE(J).EQ.1)CALL SYMBOL(X2,Y2,.14,4,0.0,-1)

10 CONTINUE

IF(NINT.EQ.0)GOTO 5
DO 20 K=1,NINT

XI=FAC*(XCENT(K)-XM)
YI=FAC*(YCENT(K)-YM)
FK=K
XN=XI+.1
YN=I+.1
CALL SYMBOL(XN,YN,.14,4,0.0,-1)
IF(M.EQ.1)CALL NUMBER(XN,YN,.07,FK,0.0,-1)

20 CONTINUE

5 CONTINUE

C

CALL EOPLLOT

RETURN

END
FUNCTION ELLPTC(AK2,IORDER)
IMPLICIT REAL*8 (A-H,O-Z)
DIMENSION A(10,2)
DATA A/1.38629436112,.09666344259,.03590092383,.03742563713,
$.01451196212,0.,.12498593597,.06880248576,.03328355346,
$.00441787012,1.,.44325141463,.06260601220,.04757383546,
$.01736506451,0.,.24998363310,.09200180037,.04069697526,
$.00526449639/
C
C = FUNCTION ELLPTC
C = THIS FUNCTION COMPUTES COMPLETE ELLIPTIC INTEGRALS OF THE
C = FIRST AND SECOND KINDS.
C = AK2 = 'K-SQUARED', THE ARGUMENT OF THE INTEGRAL
C = IORDER = INTEGRAL TYPE
C = 1 IMPLIES FIRST KIND
C = 2 IMPLIES SECOND KIND
C
ELLPTC=0.0
IF(AK2.EQ.1.0)RETURN
AKM=1.0-AK2
AK=1.

C

ASUM=0.0
BSUM=0.0

C

DO 10 I=1,5
   J=I+5
   ASUM=ASUM+A(I,IORDER)*AK
   BSUM=BSUM+A(J,IORDER)*AK
10   AK=AK*AKM

C

ELLPTC=ASUM-BSUM*DLOG(AKM)

C

RETURN

END

END OF DATA
FUNCTION F(XP,YP,XT,YT,MTYPE)

C
C    ================================================================
C
C
C    FUNCTION  F
C
C
C    THIS FUNCTION COMPUTES THE NON-HOMOGENEOUS PORTION OF THE
C    POISSON EQUATION AT A POINT AND IS CHANGEABLE BY THE USER.
C
C
C    CALLING ARGUMENTS -
C
C
C    XP,YP - COORDINATES OF POINT IN QUESTION
C
C
C    XT,YT - SAME POINT AS ABOVE EXCEPT TRANSFORMED TO ORTHO-
C    TROPIC AXES
C
C
C    ================================================================
C
C
IMPLICIT REAL*8(A-H,O-Z)
COMMON/TEMP/DELTA,NUMDLT,INTER
COMMON/IO/INDEV,IOUTDV,IPL0T,ISTOR1,ISTOR2,ISTOR3,ISTOR4,ISTOR5
COMMON/CNTRL/ITYPE,NUMEL(20),NUMINT(20),NUMMAT,NTOT,NUMNP
COMMON/MTRL/UCOND(20),VCOND(20),SPHT(20),DENS(20),QU(20),QV(20),
                 UCOS(20),VCOS(20),QX(20),QY(20),NUMRAN(20)

C

  F=0.0
C
RETURN
END
APPENDIX E

SAMPLE I/O

The analysis of the square plate, with internal heat generation and orthotropic material properties, presented in Chapter VI (Figure VI-4) is used here to demonstrate the input and output formats of the program in Appendix D. On the following page is the sample input data. Starting on the page following it is the output from that data. For neatness, some of the output lines have been abbreviated in order to fit on 8½" by 11" paper (as opposed to the usual large computer forms).
(THESE SAMPLE INPUT DATA CARDS PRODUCE
THE OUTPUT ON THE FOLLOWING PAGES)

RECTANGULAR PLATE; ORTHOTROPIC CONDUCTIVITY

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>16</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>6.0</td>
<td>0.0</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>6.0</td>
<td>0.001</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>6.0</td>
<td>6.0</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>6.0</td>
<td>6.001</td>
</tr>
<tr>
<td>12</td>
<td>1</td>
<td>0.0</td>
<td>6.001</td>
</tr>
<tr>
<td>13</td>
<td>1</td>
<td>0.0</td>
<td>6.0</td>
</tr>
<tr>
<td>16</td>
<td>1</td>
<td>0.0</td>
<td>0.001</td>
</tr>
<tr>
<td>1</td>
<td>16</td>
<td>1.0</td>
<td>.25</td>
</tr>
<tr>
<td>5</td>
<td>1000</td>
<td>2.0</td>
<td>2.0</td>
</tr>
<tr>
<td>4.0</td>
<td>2.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.0</td>
<td>3.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.0</td>
<td>4.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.0</td>
<td>4.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>15</td>
<td>16</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>16</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>
**RECTANGULAR PLATE; ORTHOTROPIC CONDUCTIVITY**

**PROBLEM TYPE----  0**
**NUMBER OF NODAL POINTS---- 16**
**NUMBER OF MATERIALS---- 1**
**PLOT CODE----  0**

**TWO DIMENSIONAL PLANE BODY**

<table>
<thead>
<tr>
<th>N.P. NO.</th>
<th>CODE</th>
<th>X</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| 1        | 1    | 0.0      | 0.0      |
| 2        | 1    | 0.200000D+01 | 0.0     |
| 3        | 1    | 0.400000D+01 | 0.0     |
| 4        | 1    | 0.600000D+01 | 0.0     |
| 5        | 1    | 0.600000D+01 | 0.100000D-02 | 0.0 |
| 6        | 1    | 0.600000D+01 | 0.200067D+01 | 0.0 |
| 7        | 1    | 0.600000D+01 | 0.400033D+01 | 0.0 |
| 8        | 1    | 0.600000D+01 | 0.600000D+01 | 0.0 |
| 9        | 1    | 0.600000D+01 | 0.600100D+01 | 0.0 |
| 10       | 1    | 0.400000D+01 | 0.600100D+01 | 0.0 |
MATERIAL PROPERTY AND SUBREGIONAL PARAMETERS

MATERIAL NUMBER------ 1

NUMBER OF ELEMENTS DEFINING REGION------ 16
THERMAL CONDUCTIVITY ALONG U-AXIS------ 0.100000D+0

1

THERMAL CONDUCTIVITY ALONG V-AXIS------ 0.250000D+0

0

ANGLE OF U-AXIS W/R TO GLOBAL X-AXIS------ 0.0
NUMBER OF INTERNAL POINTS SPECIFIED------ 5
NUMBER OF RANDOM POINTS SPECIFIED------ 1000
INTERNAL POINTS OF MATERIAL NUMBER 1

\begin{align*}
\begin{array}{cc}
X & Y \\
0.200000D+01 & 0.200000D+01 \\
0.400000D+01 & 0.200000D+01 \\
0.300000D+01 & 0.300000D+01 \\
0.200000D+01 & 0.400000D+01 \\
0.400000D+01 & 0.400000D+01 \\
\end{array}
\end{align*}

ELEMENT DATA FOR REGION NUMBER 1

\begin{align*}
\begin{array}{ccc}
\text{ELEMENT NO.} & N1 & N2 \\
1 & 1 & 2 \\
2 & 2 & 3 \\
3 & 3 & 4 \\
4 & 4 & 5 \\
5 & 5 & 6 \\
6 & 6 & 7 \\
7 & 7 & 8 \\
8 & 8 & 9 \\
\end{array}
\end{align*}
<table>
<thead>
<tr>
<th>POISSON TERMS</th>
<th>NODE</th>
<th>POISSON TERM</th>
<th>STA</th>
</tr>
</thead>
<tbody>
<tr>
<td>NDARD DEVIATION</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0.1326000D+03</td>
<td>0.12</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>0.1218869D+03</td>
<td>0.15</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>0.1228449D+03</td>
<td>0.15</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>0.1314545D+03</td>
<td>0.12</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>0.1318694D+03</td>
<td>0.13</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>0.1068111D+03</td>
<td>0.12</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>0.1047664D+03</td>
<td>0.12</td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>0.1325312D+03</td>
<td>0.12</td>
</tr>
</tbody>
</table>
POISSON TERMS FOR REGION 1

INTERNAL NODE  POISSON TERM

STANDARD DEVIATION

1  0.8911257D+02
0.1392306D+01
2  0.8880195D+02
0.1408965D+01
### SOLUTION

<table>
<thead>
<tr>
<th>BOUNDARY NODE</th>
<th>X</th>
<th>Y</th>
<th>POTENTIAL</th>
<th>POTENTIAL DERIVATIVE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.0</td>
<td>-0.1666411D+01</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0.2000000D+01</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>-0.1152216D+01</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0.4000000D+01</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>-0.8029084D+00</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0.6000000D+01</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>-0.2178998D+01</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>0.6000000D+01</td>
<td>0.0</td>
<td>0.1000000D</td>
<td></td>
</tr>
<tr>
<td>-02</td>
<td>0.0</td>
<td>0.1062082D+01</td>
<td>0.2000667D</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>0.6000000D+01</td>
<td>0.2000667D</td>
<td></td>
<td></td>
</tr>
<tr>
<td>+01</td>
<td>0.0</td>
<td>-0.3086309D+01</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NODE</td>
<td>X</td>
<td>Y</td>
<td>POTENTIAL</td>
<td></td>
</tr>
<tr>
<td>------</td>
<td>---------</td>
<td>---------</td>
<td>-----------</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>0.6000000D+01</td>
<td>-0.326957D+01</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>0.6000000D+01</td>
<td>0.6000000D</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>0.6000000D+01</td>
<td>-0.6059912D+00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>0.4000000D+01</td>
<td>0.60001000D</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>0.2000000D+01</td>
<td>0.6001000D</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>0.0</td>
<td>0.7536601D-01</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>0.0</td>
<td>-0.1187139D+02</td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>0.0</td>
<td>0.1431033D+02</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>0.0</td>
<td>-0.5664001D+01</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>0.0</td>
<td>-0.2808841D+01</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

INTERNAL POTENTIAL COMPUTATION FOR REGION 1
<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.2000000D+01</td>
<td>0.2000000D+01</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0.3558668D+01</td>
<td>0.2000000D+01</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0.3617713D+01</td>
<td>0.2000000D+01</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0.3000000D+01</td>
<td>0.2000000D+01</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>0.3503096D+01</td>
<td>0.2000000D+01</td>
<td></td>
</tr>
</tbody>
</table>
VITA

Gary Steven Gipson was born in Jackson, Mississippi on January 4, 1952. After graduating from Byram High School in 1970, he attended Louisiana State University in Baton Rouge where he received the degree of B.S. in Physics in 1975. He received a teaching and research assistantship in the LSU Department of Engineering Science, which awarded him the degree of Master of Science in 1978. He worked as Instructor in the Department of Civil Engineering from 1977-80, and as a Research Associate in the Institute of Environmental Studies from 1980-82.

Mr. Gipson is the author of several scientific and technical papers. He also holds memberships in the American Physical Society, the American Society of Mechanical Engineers, the National Society of Professional Engineers, the Sigma Xi Scientific Research Society, the Society of Engineering Science, the American Academy of Mechanics, the Louisiana Engineering Society, and MENSA. He is presently a candidate for the degree of Doctor of Philosophy in the Interdepartmental Program in Engineering Science at LSU.

He has been married to the former Cheryl Ann Martinez of Baton Rouge since 1973.
EXAMINATION AND THESIS REPORT

Candidate: Gary Steven Gipson

Major Field: Engineering Science

Title of Thesis: The Coupling of Monte Carlo Integration with the Boundary Integral Equation Technique to Solve Poisson Type Equations

Approved:

[Signatures]

Major Professor and Chairman
Dean of the Graduate School

EXAMINING COMMITTEE:

[Signatures]

Date of Examination:
April 29, 1982