Chapter 2

Potential Problems

2.1 Introduction

Since the publication of the first book on Boundary Elements in 1978 [1] many such works have appeared in the literature, some dealing with potential, others with elastostatics problems as described in Chapter 3 and many other engineering applications. The importance of 1978 is that on that date publication of the first book on boundary elements coincided with holding the first conference in which the method was established, although the name in the context of the now classical B.E.M. appears to have been used for the first time in two papers by Brebbia and Dominguez and dated 1977 [2], [3]. Up to that time boundary integral equations solutions were almost exclusively the domain of mathematicians and physicists, with very little work being done to apply them to realistic engineering problems. Efforts such as the pioneering work by Hess and Smith [4] remained as special cases rather than being interpreted as a way of generating a whole new method of solutions for general engineering problems. Nevertheless Hess and Smith developed many powerful programs for the solution of Laplace type boundary-value problems which were applied to potential flow and arbitrary bodies, using what is now called the indirect boundary element technique. They extended their formulation to analyse three dimensional objects as well as two dimensional and their codes are still popular in aerodynamics. Equally important although in a different field was the work of Harrington and his collaborators [5], [6] who applied the technique to solve electrical engineering problems using more general impedance boundary conditions, of the type now called Robin or mixed conditions. They were still using the indirect formulation.

Boundary elements are usually associated with the direct formulation which is the one discussed most of the time in this book. This formulation in potential problems can be traced to Jaswon. As early as 1963 Jaswon [7] and Symm [8] presented a numerical technique to solve Fredholm boundary integral equations, which consisted of discretizing the boundary into a series of small segments (elements) and assuming a constant source density within each segment. They employed collocation to obtain the governing system of equations and compute the influence coefficients using numerical – Simpson rule – techniques, with the exception of the singular coefficients which were computed either analytically or by the summation of off-diagonal terms. They even proposed a more general formulation through the application of Green's third identity with potentials and their derivatives as boundary unknowns and results for this formulation were represented in [7] and [9]. All the bases of boundary elements were there but somehow their work failed to attract the attention it deserved, probably due to the simultaneous emergence of the finite element method.

Since 1978 the boundary element method is seen as related to other numerical techniques such as finite elements and finite differences, mainly through the work of Brebbia and his collaborators. This relationship is sometimes highlighted by using weighted residual or variational type techniques.

In this chapter the formulation of boundary elements for potential problems will be discussed. The method is presented using the same consideration of weighted residuals as used in Chapter 1. The numerical implementation of the method will be described in detail, and used in simple computer codes which can be run in the type of PC used by engineers. The chapter presents formulations using different types of elements, i.e. those with constant and linear variations. Special consideration is given to the proper treatment of the corner points. Quadratic elements are also discussed and implemented in a computer code.

Other sections in this chapter deal with the Poisson equation and the treatment of distributed sources, problems with more than one surface, multizones applications, anisotropy and the Helmholtz equations. Computer codes are not provided for these cases but the existing programs can be easily extended and the boundary element student can do this as a computational exercise.

2.2 Basic Integral Equation

The starting boundary integral equation required by the method can be deduced in a simple way based on considerations of weighted residuals, Betti's reciprocal theorem, Green's third identity or fundamental principles such as virtual work. The advantage of using weighted residuals is its generality; it permits the extension of the method to solve more complex partial differential equations. It can also be used to relate boundary elements to other numerical techniques and can be easily understood by engineers.

Consider that we are seeking to find the solution of a Laplace equation in a Ω (two or three dimensional) domain, (figure 2.1)

$$\nabla^2 u = 0 \qquad \text{in } \Omega \tag{2.1}$$

with the following conditions on the Γ boundary

$$\begin{aligned} &\text{(i) `\textit{Essential}' \textit{Conditions} of the type } u = \bar{u} \text{ on } \Gamma_1 \\ &\text{(ii) `\textit{Natural}' \textit{Conditions} such as } q = \partial u/\partial n = \bar{q} \text{ on } \Gamma_2 \end{aligned} \tag{2.2}$$

where n is the normal to the boundary, $\Gamma = \Gamma_1 + \Gamma_2$ and the dashes indicate that those values are known. More complex boundary conditions such as combination of the above two, i.e.

$$\alpha u + \beta q = \gamma \tag{2.3}$$

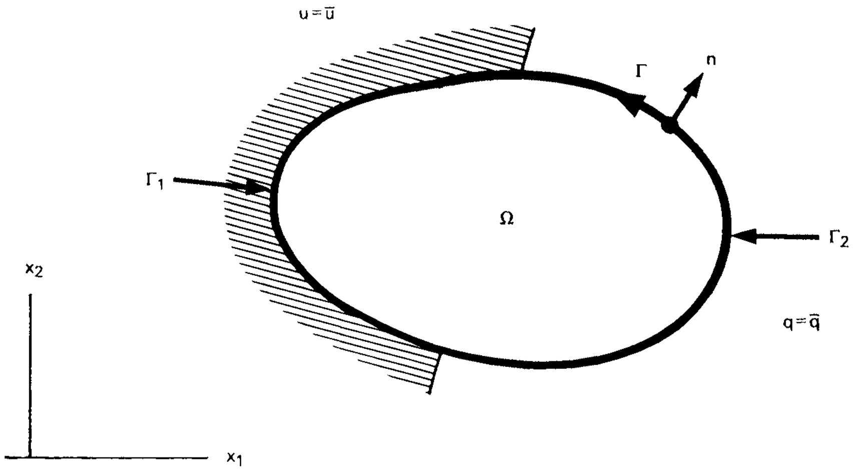


Figure 2.1 Geometrical definitions for Laplace equation

where α, β and γ are known parameters, can be easily included but they will not be considered now for simplicity's sake.

In principle the error introduced in the above equation if the exact (but unknown) values of u and q are replaced by an approximate solution can be minimized by orthogonalizing them with respect to a weighted function u^*, with derivatives on the boundary $q^* = \partial u^*/\partial n$.

In other words if R are the residuals, one can write in general that

$$\begin{aligned} R &= \nabla^2 u \neq 0 \\ R_1 &= u - \bar{u} \neq 0 \\ R_2 &= q - \bar{q} \neq 0 \end{aligned} \tag{2.4}$$

where u and q are approximate values. (The fact that one or more of the residuals may be identically zero does not detract from the generality of the argument.)

The weighting can now be carried out as shown in Chapter 1, i.e.

$$\int_\Omega R u^* \, d\Omega = \int_{\Gamma_2} R_2 u^* \, d\Gamma - \int_{\Gamma_1} R_1 q^* \, d\Gamma \tag{2.5}$$

or,

$$\int_\Omega (\nabla^2 u) u^* \, d\Omega = \int_{\Gamma_2} (q - \bar{q}) u^* \, d\Gamma - \int_{\Gamma_1} (u - \bar{u}) q^* \, d\Gamma \tag{2.6}$$

Integrating by parts the left hand side of this equation gives,

$$-\int_\Omega \left\{ \frac{\partial u}{\partial x_k} \frac{\partial u^*}{\partial x_k} \right\} d\Omega = -\int_{\Gamma_2} \bar{q} u^* \, d\Gamma - \int_{\Gamma_1} q u^* \, d\Gamma - \int_{\Gamma_1} u q^* \, d\Gamma + \int_{\Gamma_1} \bar{u} q^* \, d\Gamma \tag{2.7}$$

where $k = 1, 2, 3$ and the so-called Einstein's summation for repeated indexes has been used. Integrating by parts again the term on the left hand side one obtains,

$$\int_{\Omega} (\nabla^2 u^*) u \, d\Omega = -\int_{\Gamma_2} \bar{q} u^* \, d\Gamma - \int_{\Gamma_1} q u^* \, d\Gamma + \int_{\Gamma_2} u q^* \, d\Gamma + \int_{\Gamma_1} \bar{u} q^* \, d\Gamma \tag{2.8}$$

This is an important equation as it is the starting point for the application of the boundary element method. Notice that equation (2.8) is the same as Green's theorem (equation (1.18)) after substitution of equation (2.1) and once the boundary conditions are applied. Our aim is now to render formula (2.8) into a boundary integral equation. This is done by using a special type of weighting function u^* called the fundamental solution.

Fundamental Solution

The fundamental solution u^* satisfies Laplace's equation and represents the field generated by a concentrated unit charge acting at a point 'i'. The effect of this charge is propagated from i to infinity without any consideration of boundary conditions. Because of this the solution can be written

$$\nabla^2 u^* + \Delta^i = 0 \tag{2.9}$$

where Δ^i represents a Dirac Delta function which tends to infinity at the point $\mathbf{x} = \mathbf{x}^i$ and is equal to zero anywhere else. The integral of Δ^i however is equal to one. The use of Dirac delta function is an elegant way of representing unit concentrated charges as forces when dealing with differential equations.

The integral of a Dirac delta function multiplied by any other function is equal to the value of the latter at the point x^i. Hence

$$\int_{\Omega} u(\nabla^2 u^*) \, d\Omega = \int_{\Omega} u(-\Delta^i) \, d\Omega = -u^i \tag{2.10}$$

Equation (2.8) can now be written as,

$$u^i + \int_{\Gamma_2} u q^* \, d\Gamma + \int_{\Gamma_1} \bar{u} q^* \, d\Gamma = \int_{\Gamma_2} \bar{q} u^* \, d\Gamma + \int_{\Gamma_1} q u^* \, d\Gamma \tag{2.11}$$

It needs to be remembered that equation (2.11) applies for a concentrated charge at 'i' and consequently the values of u^* and q^* are those corresponding to that particular position of the charge. For each other $\mathbf{x}^i$ position one will find a new integral equation.

For an isotropic three dimensional medium the fundamental solution of equation (2.9) is

$$u^* = \frac{1}{4\pi r} \tag{2.12}$$

and for a two dimensional isotropic domain, it is

$$u^* = \frac{1}{2\pi} \ln\left(\frac{1}{r}\right) \tag{2.13}$$

where r is the distance from the point $\mathbf{x}^i$ of application of the delta function to any point under consideration.

It is easy to check that solution (2.12) and (2.13) satisfy the three and two dimensional Laplace equations. Consider for instance the three dimensional equation in terms of polar coordinates after neglecting terms which are zero due to symmetry of the solution, i.e.

$$\nabla^2 u^* \rightarrow \frac{\partial^2 u^*}{\partial r^2} + \frac{2}{r}\frac{\partial u^*}{\partial r} = -\Delta^i \tag{2.14}$$

Simply by substituting solution (2.12) into (2.14) we can check that the equation is satisfied for any value of r different from zero. For the case where $r \equiv 0$ we need to carry out the integration around a sphere of radius ε and then take ε to zero. Consider that the sphere has an Ω_ε domain, and integrate by parts to express the Laplacian in terms of boundary fluxes $\partial u^*/\partial n$, i.e.

$$\int_{\Omega_\varepsilon} (\nabla^2 u^*)\, d\Omega = \int_{\Gamma_\varepsilon} \left(\frac{\partial u^*}{\partial n}\right) d\Gamma = \int_{\Gamma_\varepsilon} \frac{\partial u^*}{\partial r}\, d\Gamma \tag{2.15}$$

Notice that $n \equiv r$ on the surface of the sphere.

Substituting now the fundamental solution (2.12) into (2.15) and making r (or ε) tend to zero gives

$$\begin{aligned} \lim_{\varepsilon\to 0}\left\{\int_{\Gamma_\varepsilon} \frac{\partial u^*}{\partial r}\, d\Gamma\right\} &= \lim_{\varepsilon\to 0}\left\{\int_{\Gamma_\varepsilon} -\frac{1}{4\pi\varepsilon^2}\, d\Gamma\right\} \\ &= \lim_{\varepsilon\to 0}\left\{-\frac{4\pi\varepsilon^2}{4\pi\varepsilon^2}\right\} = -1 \end{aligned} \tag{2.16}$$

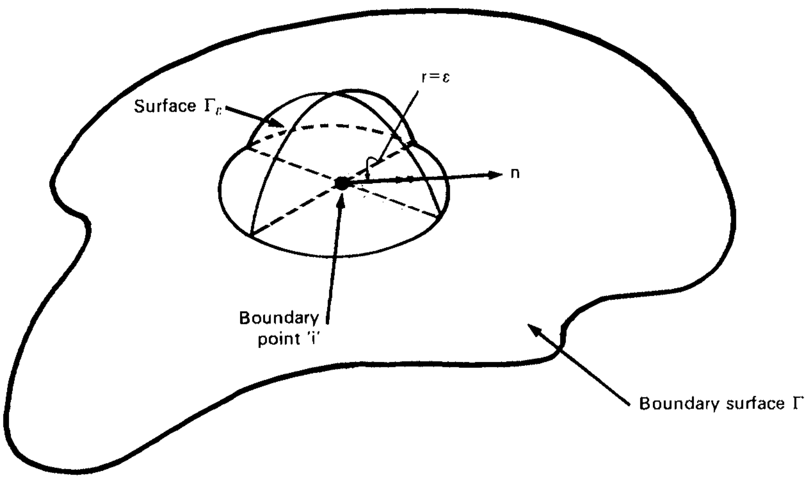
Notice that the surface of the sphere is $\Gamma_\varepsilon = 4\pi\varepsilon^2$. Similarly for the two dimensional case one can define a small circle of radius ε and then take the limit when $\varepsilon \rightarrow 0$, i.e.

$$\begin{aligned} \lim_{\varepsilon\to 0}\left\{\int_{\Gamma_\varepsilon} \frac{\partial u^*}{\partial n}\, d\Gamma\right\} &= \lim_{\varepsilon\to 0}\left\{\int_{\Gamma} -\frac{1}{2\pi r}\, d\Gamma\right\} \\ &= \lim_{\varepsilon\to 0}\left\{-\frac{2\pi\varepsilon}{2\pi\varepsilon}\right\} = -1 \end{aligned} \tag{2.17}$$

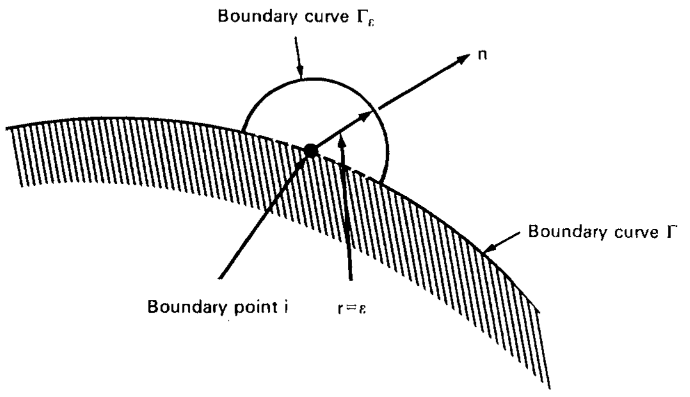
Here the perimeter of the small circle is $\Gamma_\varepsilon = 2\pi\varepsilon$.

Boundary Integral Equation

We have now deduced an equation (2.11) which is valid for any point within the Ω domain. In boundary elements it is usually preferable for computational reasons to apply equation (2.11) on the boundary and hence we need to find out what happens when the point $\mathbf{x}^i$ is on Γ. A simple way to do this is to consider that the point i is on the boundary but the domain itself is augmented by a hemisphere of radius ε (in 3D) as shown in figure 2.2 (for 2D the same applies but we will consider a semicircle instead). The point $\mathbf{x}^i$ is considered to be at the centre and then the radius ε is taken to zero. The point will then become a boundary point and the resulting expression the specialization of (2.11) for a point on Γ. At present we will only consider smooth surfaces as represented in figure 2.2 and discuss the case of corners in other sections.



(i) Three Dimensional Case. Hemisphere around i



(ii) Two Dimensional Case. Semicircle around i

Figure 2.2 Boundary points for two and three dimensional case, augmented by a small hemisphere or semicircle

It is important at this stage to differentiate between two types of boundary integrals in (2.11) as the fundamental solution and its derivative behave differently. Consider for the sake of simplicity equation (2.11) before any boundary conditions have been applied, i.e.

$$u^i + \int_\Gamma uq^*\, d\Gamma = \int_\Gamma u^*q\, d\Gamma \tag{2.18}$$

Here $\Gamma = \Gamma_1 + \Gamma_2$ and satisfaction of the boundary conditions will be left for later on.

Integrals of the type shown on the right hand side of (2.18) are easy to deal with as they present a lower order singularity, i.e. for three dimensional cases the integral around Γ_ε gives:

$$\begin{aligned}\lim_{\varepsilon\to 0}\left\{\int_{\Gamma_\varepsilon} qu^*\, d\Gamma\right\} &= \lim_{\varepsilon\to 0}\left\{\int_{\Gamma_\varepsilon} q\frac{1}{4\pi\varepsilon}\, d\Gamma\right\}\\ &= \lim_{\varepsilon\to 0}\left\{q\frac{2\pi\varepsilon^2}{4\pi\varepsilon}\right\} \equiv 0\end{aligned} \tag{2.19}$$

In other words nothing occurs to the right hand side integral when (2.11) or (2.18) are taken to the boundary. The left hand side integral however behaves in a different manner. Here we have around Γ_ε the following result,

$$\begin{aligned}\lim_{\varepsilon\to 0}\left\{\int_{\Gamma_\varepsilon} uq^*\, d\Gamma\right\} &= \lim_{\varepsilon\to 0}\left\{-\int_{\Gamma_\varepsilon} u\frac{1}{4\pi\varepsilon^2}\, d\Gamma\right\}\\ &= \lim_{\varepsilon\to 0}\left\{-u\frac{2\pi\varepsilon^2}{4\pi\varepsilon^2}\right\} = -\tfrac{1}{2}u^i\end{aligned} \tag{2.20}$$

They produce what is called a free term. It is easy to check that the same will occur for two dimensional problems in which case the right hand side integral around Γ_ε is also identically equal to zero and the left hand side integral becomes

$$\begin{aligned}\lim_{\varepsilon\to 0}\left\{\int_{\Gamma_\varepsilon} uq^*\, d\Gamma\right\} &= \lim_{\varepsilon\to 0}\left\{-\int_{\Gamma_\varepsilon} u\frac{1}{2\pi\varepsilon}\, d\Gamma\right\}\\ &= \lim_{\varepsilon\to 0}\left\{-u\frac{\pi\varepsilon}{2\pi\varepsilon}\right\} = -\tfrac{1}{2}u^i\end{aligned} \tag{2.21}$$

From (2.19) to (2.21) one can write the following expression for two or three dimensional problems

$$\tfrac{1}{2}u^i + \int_\Gamma uq^*\, d\Gamma = \int_\Gamma qu^*\, d\Gamma \tag{2.22}$$

where the integrals are in the sense of Caùchy Principal Value.

This is the boundary integral equation generally used as a starting point for boundary elements.

2.3 The Boundary Element Method

Let us now consider how expression (2.22) can be discretized to find the system of equations from which the boundary values can be found. Assume for simplicity that the body is two dimensional and its boundary is divided into N segments or elements as shown in figure 2.3. The points where the unknown values are considered are called 'nodes' and taken to be in the middle of the element for the so-called 'constant' elements (figure 2.3(a)). These are going to be the elements considered in this section, but later on we will also discuss the case of linear elements, i.e those elements for which the nodes are at the extremes or ends (figure 2.3(b)) and curved elements such as the quadratic ones shown in figure 2.3(c) and for which a further mid-element node is required.

For the constant elements considered here the boundary is assumed to be divided into N elements. The values of u and q are assumed to be constant over each element and equal to the value at the mid-element node. Equation (2.22) can be discretized for a given point 'i' before applying any boundary conditions, as follows,

$$\tfrac{1}{2}u^i + \sum_{j=1}^{N} \int_{\Gamma_j} uq^*\, d\Gamma = \sum_{j=1}^{N} \int_{\Gamma_j} qu^*\, d\Gamma \tag{2.23}$$

The point i is one of the boundary nodes. Note that for this type of element (i.e. constant) the boundary is always 'smooth' as the node is at the centre of the element, hence the multiplier of u^i is $\frac{1}{2}$. Γ_j is the boundary of the 'j' element.

The u and q values can be taken out of the integrals as they are constant over each element. They will be called u^j and q^j for element 'j'. Hence

$$\tfrac{1}{2}u^i + \sum_{j=1}^{N} \left(\int_{\Gamma_j} q^*\, d\Gamma\right)u^j = \sum_{j=1}^{N} \left(\int_{\Gamma_j} u^*\, d\Gamma\right)q^j \tag{2.24}$$

Notice that there are now two types of integrals to be carried out over the elements, i.e. those of the following types

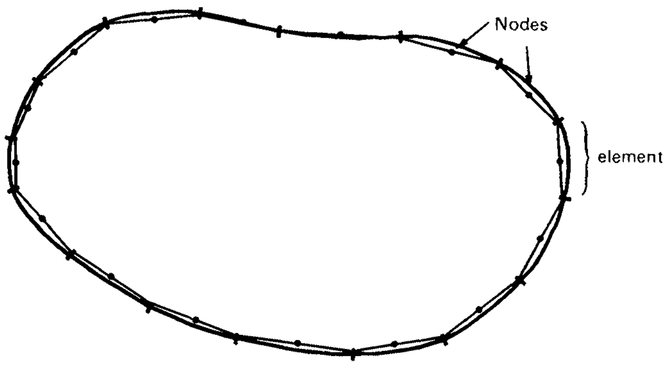
$$\int_{\Gamma_j} q^*\, d\Gamma \qquad \text{and} \qquad \int_{\Gamma_j} u^*\, d\Gamma$$

These integrals relate the 'i' node where the fundamental solution is acting to any other 'j' node. Because of this their resulting values are sometimes called influence coefficients. We will call them $\hat{H}^{ij}$ and G^{ij}, i.e.

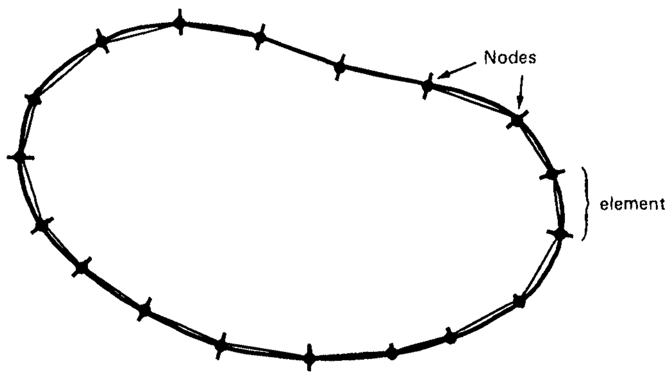
$$\hat{H}^{ij} = \int_{\Gamma_j} q^*\, d\Gamma; \qquad G^{ij} = \int_{\Gamma_j} u^*\, d\Gamma \tag{2.25}$$

Notice that we are assuming throughout that the fundamental solution is applied at a particular 'i' node, although this is not explicitly indicated in u^*, q^* notation to avoid proliferation of indexes. Hence for a particular 'i' point one can write,

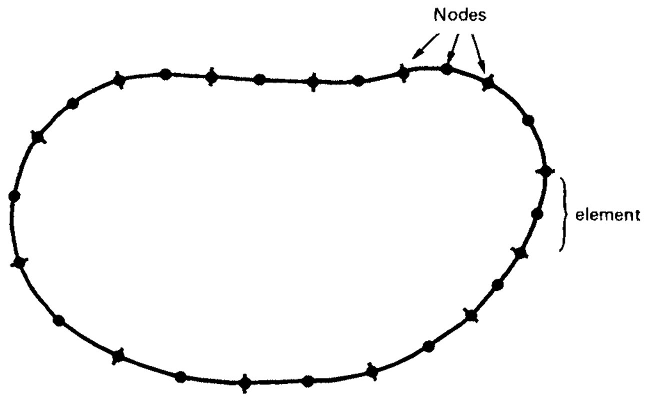
$$\tfrac{1}{2}u^i + \sum_{j=1}^{N} \hat{H}^{ij}u^j = \sum_{j=1}^{N} G^{ij}q^j \tag{2.26}$$



(a) Constant Elements



(b) Linear Elements



(c) Quadratic Elements

Figure 2.3 Different types of boundary elements

If we now assume that the position of i can also vary from 1 to N, i.e. we assume that the fundamental solution is applied at each node successively one obtains a system of equations resulting from applying (2.26) to each boundary point in turn.

Let us now call

$$H^{ij} = \begin{cases} \hat{H}^{ij} & \text{when } i \neq j \\ \hat{H}^{ij} + \frac{1}{2} & \text{when } i = j \end{cases} \tag{2.27}$$

hence equation (2.26) can now be written as

$$\sum_{j=1}^{N} H^{ij} u^j = \sum_{j=1}^{N} G^{ij} q^j \tag{2.28}$$

This set of equations can be expressed in matrix form as

$$\mathbf{HU} = \mathbf{GQ} \tag{2.29}$$

where **H** and **G** are two $N \times N$ matrices and **U**, **Q** are vectors of length N.

Notice that N_1 values of u and N_2 values of q are known on Γ_1 and Γ_2 respectively ($\Gamma_1 + \Gamma_2 = \Gamma$), hence there are only N unknowns in the system of equations (2.29). To introduce these boundary conditions into (2.29) one has to rearrange the system by moving columns of **H** and **G** from one side to the other. Once all unknowns are passed to the left-hand side one can write,

$$\mathbf{AX} = \mathbf{F} \tag{2.30}$$

where **X** is a vector of unknowns u's and q's boundary values. **F** is found by multiplying the corresponding columns by the known values of u's or q's. It is interesting to point out that the unknowns are now a mixture of the potential and its derivative, rather than the potential only as in finite elements. This is a consequence of the boundary element being a 'mixed' formulation and gives an important advantage to the method over finite elements.

Equation (2.30) can now be solved and all the boundary values are then known. Once this is done it is possible to calculate any internal value of u or its derivatives The values of u's are calculated at any internal point 'i' using formula (2.11) which can be written as,

$$u^i = \int_\Gamma q u^* \, d\Gamma - \int_\Gamma u q^* \, d\Gamma \tag{2.31}$$

Notice that now the fundamental solution is considered to be acting on an internal point 'i' and that all values of u and q are already known. The process is then one of integration (usually numerically). The same discretization is used for the boundary integrals, i.e.

$$u^i = \sum_{j=1}^{N} G^{ij} q^j - \sum_{j=1}^{N} \hat{H}^{ij} u^j \tag{2.32}$$

The coefficients G^{ij} and $\hat{H}^{ij}$ have been calculated anew for each different internal point.

The values of internal fluxes in the two directions, say x_1 and x_2, $q_{x_1} = \partial u/\partial x_1$ and $q_{x_2} = \partial u/\partial x_2$, are calculated by carrying out derivatives on (2.31), i.e.

$$
\begin{aligned}
(q_{x_1})^i &= \left(\frac{\partial u}{\partial x_1}\right)^i = \int_\Gamma q\left(\frac{\partial u^*}{\partial x_1}\right)^i d\Gamma - \int_\Gamma u\left(\frac{\partial q^*}{\partial x_1}\right)^i d\Gamma \\
(q_{x_2})^i &= \left(\frac{\partial u}{\partial x_2}\right)^i = \int_\Gamma q\left(\frac{\partial u^*}{\partial x_2}\right)^i d\Gamma - \int_\Gamma u\left(\frac{\partial q^*}{\partial x_2}\right)^i d\Gamma
\end{aligned}
\tag{2.33}
$$

Notice that the derivatives are carried out only on the fundamental solution functions u^* and q^* as we are computing the variations of flux around the 'i' point. The boundary integrals are discretized into integrals along the elements.

$$
\begin{aligned}
(q_{x1})^i &= \left(\frac{\partial u}{\partial x_1}\right)^i = \sum_{j=1}^{N}\left(\int_{\Gamma_j} \frac{\partial u^*}{\partial x_1} d\Gamma\right) q^j - \sum_{j=1}^{N}\left(\int_{\Gamma_j}\left(\frac{\partial q^*}{\partial x_1}\right) d\Gamma\right) u^j \\
(q_{x2})^i &= \left(\frac{\partial u}{\partial x_2}\right)^i = \sum_{j=1}^{N}\left(\int_{\Gamma_j} \frac{\partial u^*}{\partial x_2} d\Gamma\right) q^j - \sum_{j=1}^{N}\left(\int_{\Gamma_j}\left(\frac{\partial q^*}{\partial x_2}\right) d\Gamma\right) u^j
\end{aligned}
\tag{2.34}
$$

The kernels to be integrated along the elements are

$$
\left(\frac{\partial u^*}{\partial x_k}\right)^i = \frac{1}{2\pi}\frac{\partial}{\partial x_k}(-\ln r) = \frac{1}{2\pi r} r_{,k}
\tag{2.35}
$$

and

$$
\begin{aligned}
\left(\frac{\partial q^*}{\partial x_1}\right)^i &= \frac{1}{2\pi}\frac{\partial}{\partial x_1}\left(-\frac{1}{r}(r_{,1}n_1 + r_{,2}n_2)\right) \\
&= -\frac{1}{2\pi r^2}\left[(2r_{,1}^2 - 1)n_1 + 2r_{,1}r_{,2}n_2\right] \\
\frac{\partial q^*}{\partial x_2} &= -\frac{1}{2\pi r^2}\left[(2r_{,2}^2 - 1)n_2 + 2r_{,1}r_{,2}n_1\right]
\end{aligned}
\tag{2.36}
$$

where $r_{,k}$ indicates derivative at the integration point; i.e.

$$
\left(\frac{\partial r}{\partial x_k}\right)^i = -r_{,k}
$$

and n_1, n_2 are the components of the unit normal. The integration of the expressions given in (2.36) is done numerically using a standard Gaussian quadrature.

Evaluation of Integrals

Integrals like G^{ij} and $\hat{H}^{ij}$ in the above expressions can be calculated using numerical integration formulae (such as Gauss quadrature rules) for the case $i \neq j$. For the element $i = j$ however the presence on that element of the singularity due to the fundamental solution requires a more accurate integration. For these integrals it

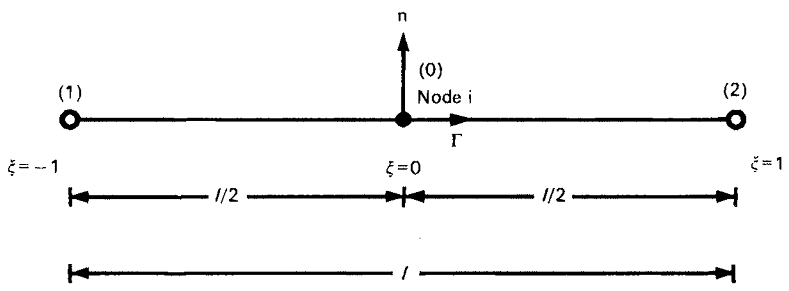


Figure 2.4 Element coordinate system

is recommended to use higher-order integration rules or a special formula (such as logarithmic and other transformations which will be discussed later on).

For the particular case of constant elements however the $\hat{H}^{ii}$ and G^{ii} integrals can be computed analytically. The $\hat{H}^{ii}$ terms for instance are identically zero, as the normal n and the element coordinate are always perpendicular to each other, i.e.

$$\hat{H}^{ii} = \int_{\Gamma_i} q^* \, d\Gamma = \int_{\Gamma_i} \frac{\partial u^*}{\partial r} \frac{\partial r}{\partial n} \, d\Gamma \equiv 0 \tag{2.37}$$

The integrals in G^{ii} require special handling. For a two dimensional element for instance they are

$$G^{ii} = \int_{\Gamma_i} u^* \, d\Gamma = \frac{1}{2\pi} \int_{\Gamma_i} \ln\left(\frac{1}{r}\right) d\Gamma \tag{2.38}$$

In order to integrate easily the above expression one can change coordinates to a homogeneous ξ coordinate over the element (figure 2.4) such that,

$$r = \left| \xi \frac{l}{2} \right| \tag{2.39}$$

where l is the element length.

Hence taking into account symmetry (2.35) can be written

$$\begin{aligned} G^{ii} &= \frac{1}{2\pi} \int_{\text{Point 1}}^{\text{Point 2}} \ln\left(\frac{1}{r}\right) d\Gamma = \frac{1}{\pi} \int_{\text{Node } i}^{\text{Point 2}} \ln\left(\frac{1}{r}\right) dr \\ &= \frac{1}{\pi}\left(\frac{l}{2}\right) \int_0^1 \ln\left(\frac{1}{\xi l/2}\right) d\xi \\ &= \frac{1}{\pi}\left(\frac{l}{2}\right)\left[\ln\left(\frac{1}{l/2}\right) + \int_0^1 \ln\left(\frac{1}{\xi}\right) d\xi \right] \end{aligned} \tag{2.40}$$

The last integral is equal to 1, i.e.

$$G^{ii} = \frac{1}{\pi}\left(\frac{l}{2}\right)\left[\ln\left(\frac{1}{l/2}\right) + 1\right] \tag{2.41}$$

For more complex cases special weighted formulae are used. The other integrals (i.e. for $i \neq j$) can be calculated using simple Gauss quadrature rules. In the two dimensional codes described in this chapter a 4 points rule has been used (see Appendix A).

2.4 Computer Code for Potential Problems using Constant Elements (POCONBE)

In what follows the above theory will be employed to produce a simple computer code written in FORTRAN for solving Laplace type problems. The code is valid for isotropic materials and uses constant elements. The program can be run in any IBM/PC type XT or AT or compatibles.

Boundary Element codes are substantially different from Finite Element programs. Their internal organization is somewhat simpler as they do not require an assembler. They also produce all the boundary values (u's and q's) and give generally very precise solutions.

Main Program

The macro-flow diagram for the POCONBE boundary element code is shown in figure 2.5. The main program defines the maximum dimensions of the system of equations, which in this case is 100 and allocates the input channel 5 and the output channel 6. It calls the following five routines

INPUTPC	– This routine reads the input to the program
GHMATPC	– It forms the system matrices **H** and **G** and rearranges them in accordance with the boundary conditions into a matrix **A**. It also creates the right hand side vector **F**.
SLNPD	– This is a subroutine for solving systems of equations, with pivoting.
INTERPC	– This routine computes the values of potentials and fluxes at internal points.
OUTPTPC:	Outputs the results.

The main routine also reads and opens files for input and output.

The general integer variables used by the program are defined as follows.

N:	Number of elements (equal number of nodes for constant elements).
L:	Number of internal points where the function is calculated.

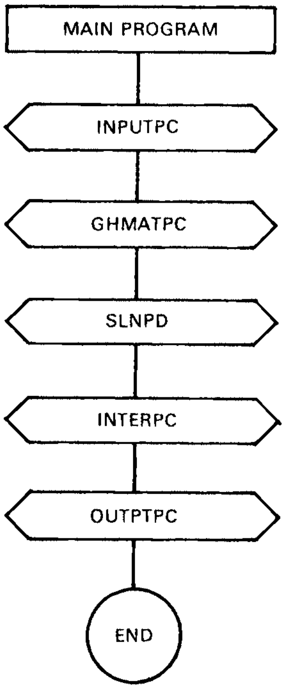


Figure 2.5 Macro flow diagram

KODE: One dimensional array indicating the type of boundary conditions at the nodes. KODE(J) = 0 means that the value of the potential is known at node J and KODE(J) = 1 signifies that the value of q is known at the corresponding boundary node.

The following real arrays are used to store data and results.

X:	One dimensional array of x_1 coordinates extreme point of boundary elements.
Y:	One dimensional array of x_2 coordinates extreme point of boundary elements.
XM:	x_1 coordinates of the nodes. XM(J) contains the x_1 coordinate of node J.
YM:	x_2 coordinates of the nodes. YM(J) contains the x_2 coordinates of node J.
G:	Matrix defined in equation (2.29). After application of boundary conditions the matrix **A** is stored in the same location.
H:	Matrix defined in equation (2.29).
FI:	Prescribed value of boundary conditions. FI(J) contains the prescribed value of the condition at node J. If KODE(J) = 0 it means that the potential is prescribed and if KODE(J) = 1 that the q is given for the element associated with the location in those vectors.

DFI:	Right hand side vector in equation (2.30). After solution it contains the values of the unknown u's and q's.
CX:	x_1 coordinate for internal point where the value of u is required.
CY:	x_2 coordinate for internal point where the value of u is required.
POT:	Vector of the potential values for internal points.
FLUX1, FLUX2:	Vectors of potential derivatives for internal points.

The listing of the MAIN program is as follows:

```
C
C-----------------------------------------------------------------------
C
      PROGRAM POCONBE
C
C   PROGRAM 1
C
C
C   THIS PROGRAM SOLVES TWO DIMENSIONAL (PO)TENTIAL PROBLEMS
C   USING (CON)STANT (B)OUNDARY (E)LEMENTS
C
C
      CHARACTER*10 FILEIN,FILEOUT
C
      DIMENSION X(101),Y(101),XM(100),YM(100),FI(100),DFI(100)
      DIMENSION KODE(100),CX(20),CY(20),POT(20),FLUX1(20),FLUX2(20)
C
      COMMON/MATG/ G(100,100)
      COMMON/MATH/ H(100,100)
      COMMON      N,L,INP,IPR
C
C     SET MAXIMUN DIMENSION OF THE SYSTEM OF EQUATIONS (NX)
C     (THIS NUMBER MUST BE EQUAL OR SMALLER THAN THE DIMENSION OF XM, ETC...)
C
      NX=100
C
C   ASSIGN NUMBERS FOR INPUT AND OUTPUT FILES
C
      INP=5
      IPR=6
C
C   READ NAMES AND OPEN  FILES FOR INPUT AND OUTPUT
C
      WRITE(*,' (A) ') ' NAME OF INPUT FILE (MAX. 10 CHART.)'
      READ(*,'  (A) ')FILEIN
      OPEN(INP,FILE=FILEIN,STATUS='OLD')
      WRITE(*,' (A) ') ' NAME OF OUTPUT FILE (MAX. 10 CHART.)'
      READ(*,' (A) ')FILEOUT
      OPEN(IPR,FILE=FILEOUT,STATUS='NEW')
C
C   READ DATA
C
      CALL INPUTPC(CX,CY,X,Y,KODE,FI)
C
C   COMPUTE H AND G MATRICES AND FORM SYSTEM (A X = F)
C
      CALL GHMATPC(X,Y,XM,YM,G,H,FI,DFI,KODE,NX)
C
C   SOLVE SYSTEM OF EQUATIONS
C
      CALL SLNPD(G,DFI,D,N,NX)
C
C   COMPUTE THE POTENTIAL AND THE FLUXES AT INTERNAL POINTS
C
      CALL INTERPC(FI,DFI,KODE,CX,CY,X,Y,POT,FLUX1,FLUX2)
C
C   PRINT RESULTS AT BOUNDARY NODES AND INTERNAL POINTS
C
      CALL OUTPTPC(XM,YM,FI,DFI,CX,CY,POT,FLUX1,FLUX2)
C
C    CLOSE INPUT AND OUTPUT FILES
C
      CLOSE (INP)
      CLOSE (IPR)
      STOP
      END
```

Routine INPUTPC

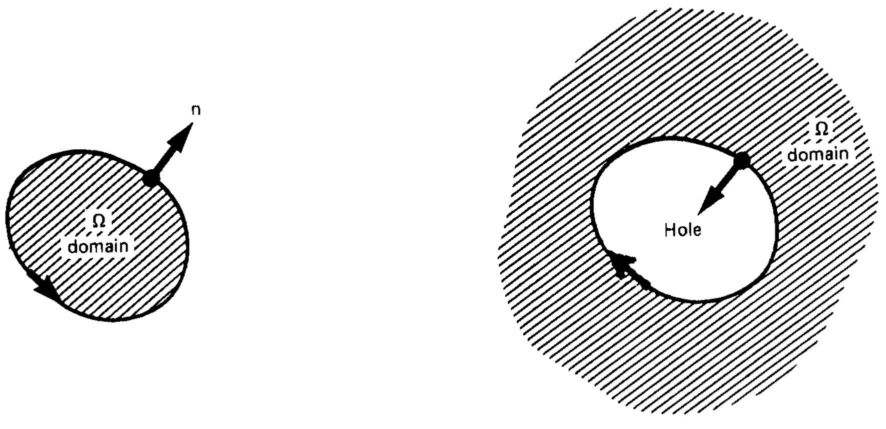
All the input required by the program is read in the program INPUTPC and contained in a file whose name is requested by the main program. The file should contain the following input lines (using free FORMAT):

(1) *Title Line* One line containing the title of the problem.
(2) *Basic Parameter Line* One line containing the number of boundary elements and the number of internal points where the function is required.
(3) *Extreme Points of Boundary Elements Lines* The coordinates of the extreme of the elements read in counterclockwise direction for the case shown in figure 2.6(a) and in clockwise direction for 2.6(b).
(4) *Boundary Conditions Lines* As many lines as nodes giving the values of KODE and the value of the potential at the node if KODE = 0 or the value of the potential derivative if KODE = 1.
(5) *Internal Points Coordinates Lines* The x_1x_2 coordinates of the internal points are read in free FORMAT in one or more lines.

This subroutine prints the title, the basic parameters the extreme points of the boundary elements and the boundary conditions. The internal point coordinates are printed in the OUTPTPC routine.

Notice that all input is written in free FORMAT.

```
C-----------------------------------------------------------------------
      SUBROUTINE INPUTPC(CX,CY,X,Y,KODE,FI)
C
C  PROGRAM 2
C
      CHARACTER*80 TITLE
      COMMON N,L,INP,IPR
      DIMENSION CX(1),CY(1),X(1),Y(1),KODE(1),FI(1)
C
C  N= NUMBER OF BOUNDARY NODES (=NUMBER OF ELEMENTS)
C  L= NUMBER OF INTERNAL POINTS WHERE THE FUNCTION IS CALCULATED
C
      WRITE(IPR,100)
  100 FORMAT(' ',79('*'))
C
C  READ JOB TITLE
C
      READ(INP,'(A)') TITLE
      WRITE(IPR,'(A)') TITLE
C
C  READ NUMBER OF BOUNDARY ELEMENTS AND INTERNAL POINTS
C
      READ(INP,*)N,L
      WRITE(IPR,300)N,L
  300 FORMAT(//' DATA'//2X,'NUMBER OF BOUNDARY ELEMENTS =',I3/2X,'NUMBER
     1 OF INTERNAL POINTS WHERE THE FUNCTION IS CALCULATED =',I3)
C
C
C  READ COORDINATES OF EXTREME POINTS OF THE BOUNDARY ELEMENTS
C  IN ARRAYS X AND Y
C
      WRITE(IPR,500)
  500 FORMAT(//2X,'COORDINATES OF THE EXTREME POINTS OF THE BOUNDARY ELE
     1MENTS',//1X,'POINT',7X,'X',15X,'Y')
      READ(INP,*) (X(I),Y(I),I=1,N)
      DO 10 I=1,N
   10 WRITE(IPR,700)I,X(I),Y(I)
  700 FORMAT(2X,I3,2(2X,E14.5))
```



(a) Numbering direction for external surface. CLOSE DOMAIN (anticlockwise) (b) Numbering direction for internal surface – OPEN DOMAIN (clockwise)

Figure 2.6 Numbering directions for external and internal surfaces

```
C
C  READ BOUNDARY CONDITIONS IN FI(I) VECTOR, IF KODE(I)=0 THE FI(I)
C  VALUE IS A KNOWN POTENTIAL;IF KODE(I)=1 THE FI(I) VALUE IS A
C  KNOWN POTENTIAL DERIVATIVE (FLUX).
C
      WRITE(IPR,800)
  800 FORMAT(//2X,'BOUNDARY CONDITIONS'//2X,'NODE',6X,'CODE',7X,'PRESCRI
     1BED VALUE')
      DO 20 I=1,N
      READ(INP,*) KODE(I),FI(I)
   20 WRITE(IPR,950)I,KODE(I),FI(I)
  950 FORMAT(2X,I3,9X,I1,8X,E14.5)
C
C  READ COORDINATES OF THE INTERNAL POINTS
C
      IF(L.EQ.0) GO TO 30
      READ(INP,*) (CX(I),CY(I),I=1,L)
   30 RETURN
      END
```

Routine GHMATPC

The routine GHMATPC forms the **G** and **H** matrices of equation (2.29) through its subroutines EXTINPC and LOCINPC. It then rearranges their columns to form **A** matrix and **F** vector of (2.30).

The subroutines EXTINPC and LOCINPC perform the following functions.

EXTINPC: This subroutine computes the **H** and **G** matrix elements by means of numerical integration along the boundary elements (using 4 points Gauss quadrature).
It calculates all elements except those on the diagonal.

LOCINPC: Only calculates the diagonal elements of **G** matrix, given by equation (2.41).

Notice that as $\hat{H}_{ii} \equiv 0$ the diagonal elements of **H** are simply $\frac{1}{2}$.

It is also important to point out that as the fundamental solution has been taken as $\ln\left(\frac{1}{r}\right)$ – without $\frac{1}{(2\pi)}$ – all terms in **G** and **H** are effectively multiplied by $1/(2\pi)$

Rearranging the columns of **G** and **H** produces the matrix **A**, which is stored in the original space used by **G**. The columns of this matrix are the columns of **H** or **G** which are multiplied by unknown values of u or q. The right hand side vector **F** is called DFI in the code and is obtained by multiplying the columns of **G** or **H** by the known values (i.e. boundary condition values) of q or u respectively.

```
C-----------------------------------------------------------------------
      SUBROUTINE GHMATPC(X,Y,XM,YM,G,H,FI,DFI,KODE,NX)
C
C  PROGRAM 3
C
C  THIS SUBROUTINE COMPUTES THE G AND H MATRICES
C  AND FORMS THE SYSTEM OF EQUATIONS A X = F
C
      COMMON N,L,INP,IPR
      DIMENSION X(1),Y(1),XM(1),YM(1),FI(1),KODE(1)
      DIMENSION DFI(1),G(NX,NX),H(NX,NX)
C
C  COMPUTE THE NODAL COORDINATES AND STORE IN ARRAYS XM AND YM
C
      X(N+1)=X(1)
      Y(N+1)=Y(1)
      DO 10 I=1,N
      XM(I)=(X(I)+X(I+1))/2
   10 YM(I)=(Y(I)+Y(I+1))/2
C
C  COMPUTE THE COEFFICIENTS OF G AND H MATRICES
C
      DO 30 I=1,N
      DO 30 J=1,N
      KK=J+1
      IF(I-J)20,25,20
   20 CALL EXTINPC(XM(I),YM(I),X(J),Y(J),X(KK),Y(KK),H(I,J),G(I,J)
     1,DQ1,DQ2,DU1,DU2,0)
      GO TO 30
   25 CALL LOCINPC(X(J),Y(J),X(KK),Y(KK),G(I,J))
      H(I,J)=3.1415926
   30 CONTINUE
C
C  REORDER THE COLUMNS OF THE SYSTEM OF EQUATIONS IN ACCORDANCE
C  WITH THE BOUNDARY CONDITIONS AND FORM SYSTEM MATRIX A WHICH
C  IS STORED IN G
C
      DO 55 J=1,N
      IF(KODE(J))55,55,40
   40 DO 50 I=1,N
      CH=G(I,J)
      G(I,J)=-H(I,J)
      H(I,J)=-CH
   50 CONTINUE
   55 CONTINUE
C
C  FORM THE RIGHT HAND SIDE VECTOR F WHICH IS STORED IN DFI
C
      DO 60 I=1,N
      DFI(I)=0.
      DO 60 J=1,N
      DFI(I)=DFI(I)+H(I,J)*FI(J)
   60 CONTINUE
      RETURN
      END
```

Routine EXTINPC

This subroutine computes the values of the off-diagonal coefficient of **H** and **G** using a 4-point Gauss integration formula (see Appendix A). It also computes, using the same numerical integration formula, the integrals of the fundamental solution and its derivatives required for the computation of potentials and fluxes at internal points (equations 2.35 and 2.36).

Consider now that instead of the system $x_1 - x_2$ we use an x–y system of coordinates. In this case $X1$, $X2$, $Y1$, $Y2$ are going to be the coordinates of the extreme points of each element considering them in clockwise (open domain) or anticlockwise manner (closed domain).

Using numerical integration and changing to a dimensionless system of coordinates the G^{ij} and H^{ij} terms for each element and collocation point can be written as,

$$G^{ij} = \sum_{k=1}^{4} \ln\left(\frac{1}{(RA)_k}\right) w_k \frac{\sqrt{(X1-X2)^2+(Y1-Y2)^2}}{2}$$

$$H^{ij} = \sum_{k=1}^{4} \frac{d}{dn}\left(\ln\frac{1}{(RA)_k}\right) w_k \frac{\sqrt{(X1-X2)^2+(Y1-Y2)^2}}{2}$$

$$= \sum_{k=1}^{4} -\frac{1}{(RA)_k}(\mathrm{RD1}*\mathrm{ETA1}+\mathrm{RD2}*\mathrm{ETA2})$$

$$\times\, w_k \frac{\sqrt{(X1-X2)^2+(Y1-Y2)^2}}{2}$$

where

$$\mathrm{RD1} = r_{,1} = \frac{(X)_k - XP}{(RA)_k}$$

$$\mathrm{RD2} = r_{,2} = \frac{(Y)_k - YP}{(RA)_k}$$

w_k is the weighting for each point, XP, YP are the coordinates of the collocation point and ETA1, ETA2 are the components of the unit normal. The values w_k and the location of the k points over the element are given in Appendix A.

```
C-----------------------------------------------------------------------
      SUBROUTINE EXTINPC(XP,YP,X1,Y1,X2,Y2,H,G,DQ1,DQ2,DU1,DU2,K)
C
C  PROGRAM 4
C
C  THIS SUBROUTINE COMPUTES THE INTEGRAL OF SEVERAL NON-SINGULAR
C  FUNCTIONS ALONG THE BOUNDARY ELEMENTS USING A FOUR POINTS
C  GAUSS QUADRATURE
C  WHEN K=0, THE OFF DIAGONAL COEFFICIENTS OF THE H AND G MATRICES ARE
C  COMPUTED
C  WHEN K=1, ALL THE COEFFICIENTS NEEDED FOR COMPUTATION OF THE POTENTIAL
C  AND FLUXES AT INTERNAL POINTS ARE COMPUTED (G,H,E1,E2,F1,F2)
C
C
C  RA= RADIUS   = DISTANCE FROM THE COLLOCATION POINT TO THE
C  GAUSS INTEGRATION POINTS ON THE BOUNDARY ELEMENT
C  ETA1,ETA2    = COMPONENTS OF THE UNIT NORMAL TO THE ELEMENT
C  RD1,RD2,RDN  = RADIUS DERIVATIVES
```

```
C
      DIMENSION XCO(4),YCO(4),GI(4),OME(4)
      DATA GI/0.86113631,-0.86113631,0.33998104,-0.33998104/
      DATA OME/0.34785485,0.34785485,0.65214515,0.65214515/
C
      AX=(X2-X1)/2.
      BX=(X2+X1)/2.
      AY=(Y2-Y1)/2.
      BY=(Y2+Y1)/2.
      SL=SQRT(AX**2+AY**2)
      ETA1=AY/SL
      ETA2=-AX/SL
      G=0.
      H=0.
      DU1=0.
      DU2=0.
      DQ1=0.
      DQ2=0.
C
C COMPUTE G, H, DQ1, DQ2, DU1 AND DU2 COEFFITIENTS
C
      DO 40 I=1,4
      XCO(I)=AX*GI(I)+BX
      YCO(I)=AY*GI(I)+BY
      RA=SQRT((XP-XCO(I))**2+(YP-YCO(I))**2)
      RD1=(XCO(I)-XP)/RA
      RD2=(YCO(I)-YP)/RA
      RDN=RD1*ETA1+RD2*ETA2
      IF(K) 30,30,10
   10 DU1=DU1+RD1*OME(I)*SL/RA
      DU2=DU2+RD2*OME(I)*SL/RA
      DQ1=DQ1-((2.*RD1**2-1.)*ETA1+2.*RD1*RD2*ETA2)*OME(I)*SL/RA**2
      DQ2=DQ2-((2.*RD2**2-1.)*ETA2+2.*RD1*RD2*ETA1)*OME(I)*SL/RA**2
   30 G=G+ALOG(1/RA)*OME(I)*SL
   40 H=H-RDN*OME(I)*SL/RA
      RETURN
      END
```

Routine LOCINPC

This routine simply computes equation (2.41) to obtain the diagonal elements of **G**. As we have used throughout the fundamental solution $\ln(1/r)$, the formula has to be multiplied by 2π, i.e. (2.41) becomes,

$$G^{ii} = l\left\{\ln\frac{1}{l/2} + 1\right\}$$

```
C-----------------------------------------------------------------------
      SUBROUTINE LOCINPC(X1,Y1,X2,Y2,G)
C
C  PROGRAM 5
C
C  THIS SUBROUTINE COMPUTES THE VALUES OF THE DIAGONAL
C  COEFFITIENTS OF THE G MATRIX
C
      AX=(X2-X1)/2.
      AY=(Y2-Y1)/2.
      SR=SQRT(AX**2+AY**2)
      G=2*SR*(1.-ALOG(SR))
      RETURN
      END
```

Routine SLNPD

This is a standard routine given in reference [10] which can solve the system of equations using pivoting if needed. If the matrix **A** has a zero in the diagonal it will interchange rows, deciding that the system matrix is singular only when no row interchange will produce a non-zero diagonal coefficient. If this happens it will give a message indicating a singularity in that row.

After elimination the results are stored in the same right hand side vector DFI.

```
C----------------------------------------------------------------------
      SUBROUTINE SLNPD(A,B,D,N,NX)
C
C  PROGRAM 6
C
C SOLUTION OF LINEAR SYSTEMS OF EQUATIONS
C BY THE GAUSS ELIMINATION METHOD PROVIDING
C FOR INTERCHANGING ROWS WHEN ENCOUNTERING A
C ZERO DIAGONAL COEFICIENT
C
C A : SYSTEM MATRIX
C B : ORIGINALLY IT CONTAINS THE INDEPENDENT
C     COEFFICIENTS. AFTER SOLUTION IT CONTAINS
C     THE VALUES OF THE SYSTEM UNKNOWNS.
C
C N : ACTUAL NUMBER OF UNKNOWNS
C NX: ROW AND COLUMN DIMENSION OF A
C
      DIMENSION B(NX),A(NX,NX)
C
      TOL=1.E-6
C
      N1=N-1
      DO 100 K=1,N1
      K1=K+1
      C=A(K,K)
      IF(ABS(C)-TOL)1,1,3
    1 DO 7 J=K1,N
C
C TRY TO INTERCHANGE ROWS TO GET NON ZERO DIAGONAL COEFFICIENT
C
      IF(ABS((A(J,K)))-TOL)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=B(K)
      B(K)=B(J)
      B(J)=C
      C=A(K,K)
      GO TO 3
    7 CONTINUE
      GO TO 8
C
C DIVIDE ROW BY DIAGONAL COEFFICIENT
C
    3 C=A(K,K)
      DO 4 J=K1,N
    4 A(K,J)=A(K,J)/C
      B(K)=B(K)/C
C
C ELIMINATE UNKNOWN X(K) FROM ROW I
C
      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 B(I)=B(I)-C*B(K)
  100 CONTINUE
C
C COMPUTE LAST UNKNOWN
C
      IF(ABS((A(N,N)))-TOL)8,8,101
  101 B(N)=B(N)/A(N,N)
C
C APPLY BACKSUBSTITUTION PROCESS TO COMPUTE REMAINING UNKNOWNS
C
      DO 200 L=1,N1
      K=N-L
      K1=K+1
      DO 200 J=K1,N
  200 B(K)=B(K)-A(K,J)*B(J)
```

```
C
C COMPUTE VALUE OF DETERMINANT
C
      D=1.
      DO 250 I=1,N
  250 D=D*A(I,I)
      GO TO 300
    8 WRITE(*,2) K
    2 FORMAT(' **** SINGULARITY IN ROW',I5)
      D=0.
  300 RETURN
      END
```

Routine INTERPC

Subroutine INTERPC reorders FI (boundary condition vector) and DFI (unknown vector) in such a way that all the values of the potential are stored in FI and all the values of the derivatives or fluxes in DFI.

This subroutine also computes the potential values for the internal points using formula (2.32) and the fluxes along x_1 and x_2 directions using formula (2.34). Note that because all the H and G terms appear multiplied by 2π the solution for the internal points is also multiplied by 2π.

```
C-----------------------------------------------------------------------
      SUBROUTINE INTERPC(FI,DFI,KODE,CX,CY,X,Y,POT,FLUX1,FLUX2)
C
C  PROGRAM 7
C
C  THIS SUBROUTINE COMPUTES THE VALUES OF THE POTENTIAL
C  AND THE POTENTIAL DERIVATIVES (FLUXES) AT INTERNAL POINTS
C
      COMMON N,L,INP,IPR
      DIMENSION FI(1),DFI(1),KODE(1),CX(1),CY(1),X(1),Y(1)
      DIMENSION POT(1),FLUX1(1),FLUX2(1)
C
C  REARRANGE THE FI AND DFI ARRAYS TO STORE ALL THE VALUES OF THE
C  POTENTIAL IN FI AND ALL THE VALUES OF THE DERIVATIVE IN DFI
C
      DO 20 I=1,N
      IF(KODE(I)) 20,20,10
   10 CH=FI(I)
      FI(I)=DFI(I)
      DFI(I)=CH
   20 CONTINUE
C
C  COMPUTE THE POTENTIAL AND THE FLUXES AT INTERNAL POINTS
C
      IF(L.EQ.0) GO TO 50
      DO 40 K=1,L
      POT(K)=0.
      FLUX1(K)=0.
      FLUX2(K)=0.
      DO 30 J=1,N
      KK=J+1
      CALL EXTINPC(CX(K),CY(K),X(J),Y(J),X(KK),Y(KK),A,B
     1,DQ1,DQ2,DU1,DU2,1)
      POT(K)=POT(K)+DFI(J)*B-FI(J)*A
      FLUX1(K)=FLUX1(K)+DFI(J)*DU1-FI(J)*DQ1
   30 FLUX2(K)=FLUX2(K)+DFI(J)*DU2-FI(J)*DQ2
      POT(K)=POT(K)/(2.*3.1415926)
      FLUX1(K)=FLUX1(K)/(2.*3.1415926)
   40 FLUX2(K)=FLUX2(K)/(2.*3.1415926)
   50 RETURN
      END
```

Routine OUTPTPC

This routine outputs the results. It first lists the coordinates of the boundary nodes and the corresponding values of potential and its derivatives (or fluxes). It also prints the values of potential and fluxes at internal points if any have been requested.

```
C-----------------------------------------------------------------------
      SUBROUTINE OUTPTPC(XM,YM,FI,DFI,CX,CY,POT,FLUX1,FLUX2)
C
C  PROGRAM 8
C
C  THIS SUBROUTINE PRINTS THE VALUES OF THE POTENTIAL AND ITS NORMAL
C  DERIVATIVE AT BOUNDARY NODES. IT ALSO PRINTS THE VALUES OF THE
C  POTENTIAL AND THE FLUXES AT INTERNAL POINTS
C
      COMMON N,L,INP,IPR
      DIMENSION XM(1),YM(1),FI(1),DFI(1),CX(1),CY(1)
      DIMENSION POT(1),FLUX1(1),FLUX2(1)
C
      WRITE(IPR,100)
  100 FORMAT(' ',79('*')//1X,'RESULTS'//2X,'BOUNDARY NODES'//8X,'X',15
     1X,'Y',13X,'POTENTIAL',3X,'POTENTIAL DERIVATIVE'/)
      DO 10 I=1,N
   10 WRITE(IPR,200) XM(I),YM(I),FI(I),DFI(I)
  200 FORMAT(4(2X,E14.5))
C
      IF(L.EQ.0) GO TO 30
      WRITE(IPR,300)
  300 FORMAT(//,2X,'INTERNAL POINTS',//8X,'X',15X,'Y',13X,'POTENTIAL',
     19X,'FLUX X',10X,'FLUX Y'/)
      DO 20 K=1,L
   20 WRITE(IPR,400)CX(K),CY(K),POT(K),FLUX1(K),FLUX2(K)
  400 FORMAT(5(2X,E14.5))
   30 WRITE(IPR,500)
  500 FORMAT(' ',79('*'))
      RETURN
      END
```

Example 2.1

The following example illustrates how the code can be used to analyse a simple potential problem. Consider the case of a square close domain of the type shown in figure 2.8, where the boundary has been discretized into 12 constant elements with 5 internal points.

The input statements are as follows:

HEAT FLOW EXAMPLE (DATA)

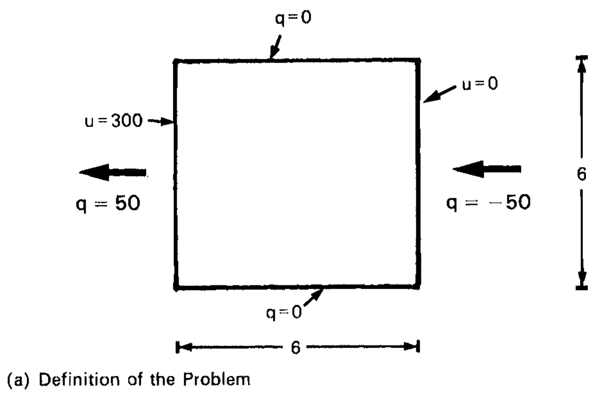
```
 HEAT FLOW EXAMPLE (12 CONSTANT ELEMENTS)
12 5
0. 0. 2. 0. 4. 0. 6. 0. 6. 2. 6. 4.
6. 6. 4. 6. 2. 6. 0. 6. 0. 4. 0. 2.
1 0
1 0
1 0
0 0
0 0
0 0
1 0
1 0
1 0
0 300
0 300
0 300
2. 2. 2. 4. 3. 3. 4. 2. 4. 4.
```

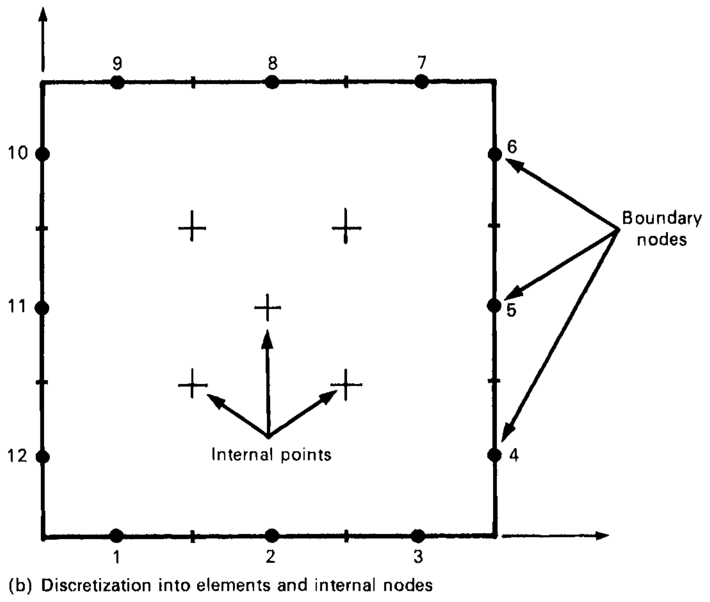
The results are printed out as follows.

HEAT FLOW EXAMPLE (OUTPUT)

```
*******************************************************************************
HEAT FLOW EXAMPLE (12 CONSTANT ELEMENTS)


DATA

 NUMBER OF BOUNDARY ELEMENTS = 12
 NUMBER OF INTERNAL POINTS WHERE THE FUNCTION IS CALCULATED =  5


 COORDINATES OF THE EXTREME POINTS OF THE BOUNDARY ELEMENTS

POINT        X                Y
   1      .00000E+00       .00000E+00
   2      .20000E+01       .00000E+00
   3      .40000E+01       .00000E+00
   4      .60000E+01       .00000E+00
   5      .60000E+01       .20000E+01
   6      .60000E+01       .40000E+01
   7      .60000E+01       .60000E+01
   8      .40000E+01       .60000E+01
   9      .20000E+01       .60000E+01
  10      .00000E+00       .60000E+01
  11      .00000E+00       .40000E+01
  12      .00000E+00       .20000E+01


 BOUNDARY CONDITIONS

 NODE       CODE        PRESCRIBED VALUE
   1          1             .00000E+00
   2          1             .00000E+00
   3          1             .00000E+00
   4          0             .00000E+00
   5          0             .00000E+00
   6          0             .00000E+00
   7          1             .00000E+00
   8          1             .00000E+00
   9          1             .00000E+00
  10          0             .30000E+03
  11          0             .30000E+03
  12          0             .30000E+03
*******************************************************************************

RESULTS

 BOUNDARY NODES

       X                Y               POTENTIAL    POTENTIAL DERIVATIVE

    .10000E+01       .00000E+00       .25225E+03       .00000E+00
    .30000E+01       .00000E+00       .15002E+03       .00000E+00
    .50000E+01       .00000E+00       .47750E+02       .00000E+00
    .60000E+01       .10000E+01       .00000E+00      -.52962E+02
    .60000E+01       .30000E+01       .00000E+00      -.48771E+02
    .60000E+01       .50000E+01       .00000E+00      -.52962E+02
    .50000E+01       .60000E+01       .47750E+02       .00000E+00
    .30000E+01       .60000E+01       .15002E+03       .00000E+00
    .10000E+01       .60000E+01       .25225E+03       .00000E+00
    .00000E+00       .50000E+01       .30000E+03       .52969E+02
    .00000E+00       .30000E+01       .30000E+03       .48737E+02
    .00000E+00       .10000E+01       .30000E+03       .52969E+02
 INTERNAL POINTS

       X                Y               POTENTIAL         FLUX X           FLUX Y

    .20000E+01       .20000E+01       .20028E+03      -.50303E+02      -.14976E+00
    .20000E+01       .40000E+01       .20028E+03      -.50303E+02       .14974E+00
    .30000E+01       .30000E+01       .15001E+03      -.50215E+02      -.40360E-05
    .40000E+01       .20000E+01       .99740E+02      -.50306E+02       .14564E+00
    .40000E+01       .40000E+01       .99740E+02      -.50306E+02      -.14564E+00
*******************************************************************************
```

Notice the excellent agreement of the results with the exact solution given in figure 2.7(a)), when the coarseness of the mesh and the simplicity of the model





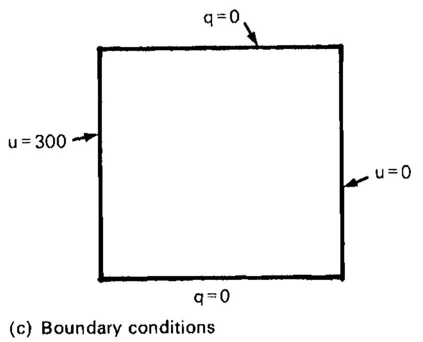


Figure 2.7 Simple potential problem

are considered. On the two vertical sides the fluxes are close to -50 and 50 as expected and on the horizontal sides the value of the potential is similar to the analytical solution which varies linearly from 300 on the left hand side to 0 on the right. The accuracy of the internal point results is however even more remarkable and this is due to the way in which these results are computed using formula (2.32), i.e. they are like a weighted average of the boundary values.

2.5 Linear Elements

Up to this section we have only considered the case of constant elements, i.e. those with the values of the variables assumed to be the same all over the element. Let us now consider a linear variation of u and q for which case the nodes are considered to be at the ends of the element as shown in figure 2.8.

The governing integral statement can now be written as,

$$c^i u^i + \int_\Gamma u q^* \, d\Gamma = \int_\Gamma u^* q \, d\Gamma \tag{2.42}$$

Notice that the $\frac{1}{2}$ coefficient of u^i has been replaced by an unknown c^i value. This is because $c^i = \frac{1}{2}$ applies only for a smooth boundary. The value of c^i for any other boundary can be proved to be,

$$c^i = \frac{\theta}{2\pi} \tag{2.43}$$

where θ is the internal angle of the corner in radians. This result can be obtained by defining a small spherical or circular region around the corners and then taking the radius of them to zero (similar to what has been shown in section 2.2). Another possibility is to determine the value of c^i implicitly (see section 2.6) and in this case it is not required to calculate the angle.

After discretizing the boundary into a series of N elements equation (2.42) can be written

$$c^i u^i + \sum_{j=1}^{N} \int_{\Gamma_j} u q^* \, d\Gamma = \sum_{j=1}^{N} \int_{\Gamma_j} u^* q \, d\Gamma \tag{2.44}$$

The integrals in this equation are more difficult to evaluate than those for the constant element as the u's and q's vary linearly over each Γ_j and hence it is not possible to take them out of the integrals.

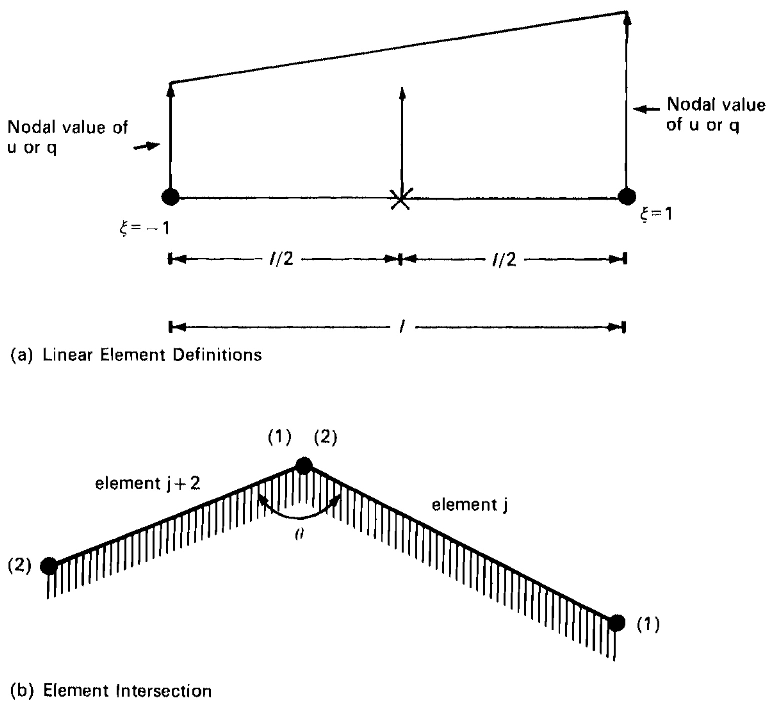


Figure 2.8 Linear element. Basic definitions

The values of u and q at any point on the element can be defined in terms of their nodal values and two linear interpolation functions ϕ_1 and ϕ_2, which are given in terms of the homogeneous coordinate ξ as shown in figure 2.8(a), i.e.

$$u(\xi)=\phi_1 u^1+\phi_2 u^2=[\phi_1 \phi_2]\begin{Bmatrix}u^1\\u^2\end{Bmatrix}$$
$$q(\xi)=\phi_1 q^1+\phi_2 q^2=[\phi_1 \phi_2]\begin{Bmatrix}q^1\\q^2\end{Bmatrix} \tag{2.45}$$

ξ is the dimensionless coordinate varying from -1 to $+1$ and the two interpolation functions are

$$\phi_1=\tfrac{1}{2}(1-\xi); \qquad \phi_2=\tfrac{1}{2}(1+\xi) \tag{2.46}$$

Let us consider the integrals over an element 'j'. Those on the left hand side can be written as,

$$\int_{\Gamma_j} uq^*\,d\Gamma=\int_{\Gamma_j}[\phi_1\phi_2]q^*\,d\Gamma\begin{Bmatrix}u^1\\u^2\end{Bmatrix}=[h_1^{ij}h_2^{ij}]\begin{Bmatrix}u^1\\u^2\end{Bmatrix} \tag{2.47}$$

where for each element 'j' we have the two terms,

$$h_1^{ij} = \int_{\Gamma_j} \phi_1 q^* \, d\Gamma \tag{2.48}$$

and

$$h_2^{ij} = \int_{\Gamma_j} \phi_2 q^* \, d\Gamma \tag{2.49}$$

Similarly the integrals on the right hand side give

$$\int_{\Gamma_j} q u^* \, d\Gamma = \int_{\Gamma_j} [\phi_1 \phi_2] u^* \, d\Gamma \begin{Bmatrix} q^1 \\ q^2 \end{Bmatrix} = [g_1^{ij} g_2^{ij}] \begin{Bmatrix} q^1 \\ q^2 \end{Bmatrix} \tag{2.50}$$

where

$$g_1^{ij} = \int_{\Gamma_j} \phi_1 u^* \, d\Gamma \tag{2.51}$$

and

$$g_2^{ij} = \int_{\Gamma_j} \phi_2 u^* \, d\Gamma \tag{2.52}$$

Treatment of Corners

A domain discretized using boundary elements will present a series of corners which require special attention as the conditions on both sides may not be the same.

When the boundary of the domain is discretized into linear elements, node 2 of element 'j' is the same point as node 1 of element '$j + 1$' (figure 2.8(b)). Since the potential is unique at any point of the boundary, u^2 of element 'j' and u^1 of element '$j + 1$' are both the same. However, this argument can not be applied as a general rule to the flux, as there are boundary points for which the flux does not have a unique value. This takes place at points where the normal to the boundary is not unique (corner points). It may also happen that the flux prescribed along a smooth boundary presents discontinuities at certain particular points. While corners with different values of the flux at both sides exist in many practical problems, discontinuous values of the flux along a smooth boundary are seldom prescribed.

To take into account the possibility that the flux of node 2 of an element may be different from the flux of node 1 of the next element, the fluxes can be arranged in a $2n$ array.

Substituting equations (2.47) and (2.50) for all 'j' elements into (2.44) one obtains the following equation for node 'i'.

$$c^i u^i + [\hat{H}^{i1} \hat{H}^{i2} \ldots \hat{H}^{iN}] \begin{Bmatrix} u^1 \\ u^2 \\ \vdots \\ u^N \end{Bmatrix} = [G^{i1} G^{i2} \ldots G^{i2N}] \begin{Bmatrix} q^1 \\ q^2 \\ \vdots \\ q^{2N} \end{Bmatrix} \tag{2.53}$$

where $\hat{H}^{ij}$ is equal to the h_1^{ij} term of element 'j' plus the h_2^{ij-1} term of element '$j-1$'. Hence formula (2.53) represents the assembled equation for node 'i'. Note the simplicity of this approach. Equation (2.53) can be written as,

$$c^i u^i + \sum_{j=1}^{N} \hat{H}^{ij} u^j = \sum_{j=1}^{2N} G^{ij} q^j \tag{2.54}$$

Similarly, as was previously shown (equation (2.28)), this formula can be written as

$$\sum_{j=1}^{N} H^{ij} u^j = \sum_{j=1}^{2N} G^{ij} q^j \tag{2.55}$$

and the whole set in matrix form becomes

$$\mathbf{HU} = \mathbf{GQ} \tag{2.56}$$

where $\mathbf{G}$ is now an $N \times 2N$ rectangular matrix.

Several situations may occur at a boundary node: First that the boundary be smooth at the node. In such a case both fluxes 'before' and 'after' the node are the same unless they are prescribed as different, but in any case, only one variable will be unknown either the potential or the unique flux. Second, that the node is at a corner point. In this case four different cases are possible depending on the boundary conditions:

(a) Known values: fluxes 'before' and 'after' the corner.
Unknown value: potential
(b) Known values: potential, and flux 'before' the corner.
Unknown value: flux 'after' the corner
(c) Known values: potential, and flux 'after' the corner.
Unknown value: flux 'before' the corner
(d) Known values: potential.
Unknown values: flux 'before' and 'after' the corner.

There is only one unknown per node for the first three cases, and two unknowns for case (d). As long as there is only one unknown per node, system (2.56) can be reordered in such a way that all the unknowns are taken to the left hand side and obtain the usual system of $N \times N$ equations, i.e.

$$\mathbf{AX} = \mathbf{F} \tag{2.57}$$

where $\mathbf{X}$ is the (N) vector of unknowns; $\mathbf{A}$ is the $(N \times N)$ matrix of coefficients which columns are columns of the matrix $\mathbf{H}$, and columns of the matrix $\mathbf{G}$ after a change of sign or sum of two consecutive columns of $\mathbf{G}$ with opposite sign when the unknown is the unique value of the flux at the corresponding node. $\mathbf{F}$ is the known vector computed by the product of the known boundary conditions and the corresponding coefficients of the $\mathbf{G}$ or $\mathbf{H}$ matrices.

When the number of unknowns at a corner node is two (case (d)), one extra equation is needed for the node. The problem can be solved using the idea of 'discontinuous' elements [11] presented in section 2.7.

2.6 Computer Code for Potential Problems using Linear Elements (POLINBE)

Although this code has many routines which are similar to those developed for the constant element case (POCONBE), there are some parts which require modification.

Main Program

The integer variables have the same meaning as in the constant elements program. The same can be said for the real arrays except for the mid-point coordinates XM and YM that are not needed as now the nodes are at the inter-element junction. Arrays FI and DFI now have a different meaning. The dimension of FI is (N) while the dimension of DFI is ($2N$). Prescribed boundary conditions are read in DFI (two per element). FI is used as the right hand side vector that after solution contains the values of the unknowns. Finally both vectors are reordered to put all the values of the potentials in FI and all the values of the fluxes in DFI as was done for constant elements. Now, however FI contains one potential and DFI two fluxes, per node.

The program allows for the flux 'before' and 'after' any node to be different. When two, equal or different, fluxes are prescribed at a node the potential is computed; if the potential and one flux are prescribed, the other flux is computed; and in the case that only the potential is prescribed, both fluxes are considered to be equal. It should be noticed that in problems with only one uniform region, the case of potential prescribed and two different unknown values of the flux will only take place in a corner where the potential is prescribed along the two elements that join at that corner. This situation is not frequent and since the potential would be known along two different directions emerging from the corner, the potential derivatives along these two directions would be known and consequently the flux along any direction would also be known. Thus, the three variables would be known at that corner and hence any two of them can be prescribed and the third will be computed. Notice that only for the case of a singularity on the corner would one require replacing the corner node by two different nodes inside each of the two adjacent elements.

The listing is as follows:

```
C-----------------------------------------------------------------------------
C
      PROGRAM POLINBE
C
C  PROGRAM 9
C
C  THIS PROGRAM SOLVES TWO DIMENSIONAL (PO)TENTIAL PROBLEMS
C  USING (LIN)EAR (B)OUNDARY (E)LEMENTS
C
      CHARACTER*10 FILEIN,FILEOUT
      COMMON/MATG/ G(80,160)
      COMMON/MATH/ H(80,80)
      COMMON N,L,INP,IPR
      DIMENSION X(81),Y(81),FI(80),DFI(160)
      DIMENSION KODE(160),CX(20),CY(20),POT(20),FLUX1(20),FLUX2(20)
C
C  SET MAXIMUN DIMENSION OF THE SYSTEM OF EQUATIONS (NX)
C  NX   = MAXIMUN NUMBER OF NODES = MAXIMUN NUMBER OF ELEMENTS
C
      NX=80
      NX1=2*NX
C
C  ASSIGN NUMBERS FOR INPUT AND OUTPUT FILES
C
      INP=5
      IPR=6
C
C  READ NAMES AND OPEN FILES FOR INPUT AND OUTPUT
C
      WRITE(*,' (A) ') ' NAME OF THE INPUT FILE (MAX. 10 CHART.)'
      READ(*,' (A) ') FILEIN
      OPEN(INP,FILE=FILEIN,STATUS='OLD')
      WRITE(*,' (A) ') ' NAME OF THE OUTPUT FILE (MAX.10 CHART.)'
      READ(*,' (A) ') FILEOUT
      OPEN(IPR,FILE=FILEOUT,STATUS='NEW')
C
C  READ DATA
C
      CALL INPUTPL(CX,CY,X,Y,KODE,DFI)
C
C  COMPUTE G AND H MATRICES AND FORM SYSTEM (A X = F)
C
      CALL GHMATPL(X,Y,G,H,FI,DFI,KODE,NX,NX1)
C
C  SOLVE SYSTEM OF EQUATIONS
C
      CALL SLNPD(H,FI,D,N,NX)
C
C  COMPUTE THE POTENTIAL AND THE FLUXES AT INTERNAL POINTS
C
      CALL INTERPL(FI,DFI,KODE,CX,CY,X,Y,POT,FLUX1,FLUX2)
C
C  PRINT RESULTS AT BOUNDARY NODES AND INTERNAL POINTS
C
      CALL OUTPTPL(X,Y,FI,DFI,CX,CY,POT,FLUX1,FLUX2)
C
      CLOSE (INP)
      CLOSE (IPR)
      STOP
      END
```

Routine INPUTPL

The input subroutine is similar to INPUTPC in POCONBE. Only the boundary conditions are prescribed in a different way. Now, two boundary conditions per element are read in array DFI. Thus, each node 'j' may have a different value of the flux, one as the end node of element '$j-1$' and the other as the start node of element 'j'.

```
C-----------------------------------------------------------------------
      SUBROUTINE INPUTPL(CX,CY,X,Y,KODE,DFI)
C
C  PROGRAM 10
C
C  N= NUMBER OF BOUNDARY ELEMENTS
C  L= NUMBER OF INTERNAL POINTS
C
      CHARACTER*80 TITLE
      COMMON N,L,INP,IPR
      DIMENSION CX(1),CY(1),X(1),Y(1),KODE(1),DFI(1)
      WRITE(IPR,100)
  100 FORMAT(' ',79('*'))
C
C  READ JOB TITLE
C
      READ(INP,'(A)') TITLE
      WRITE(IPR,'(A)') TITLE
C
C  READ NUMBER OF ELEMENTS AND INTERNAL POINTS
C
      READ(INP,*)N,L
      WRITE(IPR,300)N,L
  300 FORMAT(//' DATA'//2X,'NUMBER OF BOUNDARY ELEMENTS =',I3/2X,'NUMBER
     1 OF INTERNAL POINTS WHERE THE FUNCTION IS CALCULATED =',I3)
C
C  READ BOUNDARY NODES COORDINATES IN ARRAYS X AND Y
C
      WRITE(IPR,500)
  500 FORMAT(//2X,'COORDINATES OF THE EXTREME POINTS OF THE BOUNDARY ELE
     1MENTS',//2X,'POINT',10X,'X',18X,'Y')
      READ(INP,*) (X(I),Y(I),I=1,N)
      DO 10 I=1,N
   10 WRITE(IPR,700)I,X(I),Y(I)
  700 FORMAT(3X,I3,2(5X,E14.5))
C
C  READ BOUNDARY CONDITIONS IN DFI(I) VECTOR.IF KODE(I)=0
C  THE DFI(I) VALUE IS A KNOWN POTENTIAL; IF KODE(I)=1 THE
C  DFI(I) VALUE IS A KNOWN POTENTIAL DERIVATIVE (FLUX).
C  TWO BOUNDARY CONDITIONS ARE READ PER ELEMENT.
C  ONE NODE MAY HAVE TWO DIFFERENT VALUES OF THE
C  POTENTIAL DERIVATIVE BUT ONLY ONE VALUE OF THE POTENTIAL
C
      WRITE(IPR,800)
  800 FORMAT(//2X,'BOUNDARY CONDITIONS'//15X,'------FIRST NODE-----',
     19X,'-----SECOND NODE-----'/17X,'PRESCRIBED',20X,'PRESCRIBED'/
     2,1X,'ELEMENT',12X,'VALUE',7X,'CODE',14X,'VALUE',7X,'CODE')
      DO 20 I=1,N
      READ(INP,*) KODE(2*I-1),DFI(2*I-1),KODE(2*I),DFI(2*I)
   20 WRITE(IPR,950)I,DFI(2*I-1),KODE(2*I-1),DFI(2*I),KODE(2*I)
  950 FORMAT(2X,I3,2(10X,E14.7,5X,I1))
C
C  READ COORDINATES OF THE INTERNAL POINTS
C
      IF(L.EQ.0) GO TO 30
      READ(INP,*) (CX(I),CY(I),I=1,L)
   30 RETURN
      END
```

Routine GHMATPL

Notice that this routine is similar to the one described in POCONBE with the main difference that the g^{ij} elements are assembled in a $N \times 2N$ matrix instead of an $N \times N$ one as was previously the case. This is because two possible values of flux are considered at each node, one to the left and the other to the right of it. Then the boundary conditions are applied as described earlier to rearrange the system of equations and prepare it for solving.

The diagonal terms in **H** are computed implicitly. Assuming a constant potential over the whole boundary the flux must be zero and hence

$$\mathbf{H}\mathbf{I} = 0 \tag{2.58}$$

where **I** is a vector that for all nodes has a unit potential. Since (2.55) has to be satisfied

$$H^{ii} = -\sum_{j=1}^{N} H^{ij} \qquad (\text{for } j \neq i) \tag{2.59}$$

which gives the diagonal coefficients in terms of the rest of the terms of the **H** matrix.

The above considerations are strictly valid for close domains. When dealing with infinite or semi-infinite regions, equation (2.56) must be modified. If a unit potential is prescribed for a boundless domain the integral

$$\int_{\Gamma_\infty} p^* \, d\Gamma \tag{2.60}$$

over the external boundary Γ_∞ at infinity will not be zero and since p^* is due to a unit source, this integral must be (see equation (2.17) when $r \equiv \varepsilon \to \infty$)

$$\int_{\Gamma_\infty} p^* \, d\Gamma = -1 \tag{2.61}$$

The diagonal terms for this case are,

$$H^{ii} = 1 - \sum_{j=1}^{N} H^{ij} \qquad (\text{for } j \neq i) \tag{2.62}$$

Notice that as all the terms H^{ij} and G^{ij} are multiplied by 2π in the program, because the fundamental solution has been taken as $ln(1/r)$, equation (2.62) is also written in the program as

$$H^{ii} = 2\pi - \sum_{j=1}^{N} H^{ij} \qquad (\text{for } i \neq j) \tag{2.63}$$

```
C-----------------------------------------------------------------------
      SUBROUTINE GHMATPL(X,Y,G,H,FI,DFI,KODE,NX,NX1)
C
C  PROGRAM 11
C
C  THIS SUBROUTINE COMPUTES THE G AND H MATRICES
C  AND FORMS THE SYSTEM OF EQUATIONS A X = F
C  H IS A SQUARE MATRIX (N,N); G IS RECTANGULAR (N,2*N)
C
      COMMON N,L,INP,IPR
      DIMENSION X(1),Y(1),G(NX,NX1),H(NX,NX),FI(1),KODE(1),DFI(1)
      NN=2*N
      DO 10 I=1,N
      DO 6 J=1,N
    6 H(I,J)=0.
      DO 10 J=1,NN
   10 G(I,J)=0.
C
C  COMPUTE THE COEFFICIENTS OF G AND H MATRICES
```

```
C
      X(N+1)=X(1)
      Y(N+1)=Y(1)
      DO 100 I=1,N
      NF=I+1
      NS=I+N-2
      DO 50 JJ=NF,NS
      IF(JJ-N)30,30,20
   20 J=JJ-N
      GO TO 40
   30 J=JJ
   40 CALL EXTINPL(X(I),Y(I),X(J),Y(J),X(J+1),Y(J+1),A1,A2,B1,B2
     1,DQ1F1,DQ2F1,DQ1F2,DQ2F2,DU1F1,DU2F1,DU1F2,DU2F2,0)
      IF(J-N)42,43,43
   42 H(I,J+1)=H(I,J+1)+A2
      GO TO 44
   43 H(I,1)=H(I,1)+A2
   44 H(I,J)=H(I,J)+A1
      G(I,2*J-1)=B1
      G(I,2*J)=B2
   50 H(I,I)=H(I,I)-A1-A2
      NF=I+N-1
      NS=I+N
      DO 95 JJ=NF,NS
      IF(JJ-N)70,70,60
   60 J=JJ-N
      GO TO 80
   70 J=JJ
   80 CALL LOCINPL(X(J),Y(J),X(J+1),Y(J+1),B1,B2)
      IF(JJ-NF)82,82,83
   82 CH=B1
      B1=B2
      B2=CH
   83 G(I,2*J-1)=B1
   95 G(I,2*J)=B2
C
C   ADD ONE TO THE DIAGONAL COEFFICIENTS
C   FOR EXTERNAL PROBLEMS.
C
      IF(H(I,I)) 98,100,100
   98 H(I,I)=6.2831852+H(I,I)
  100 CONTINUE
C
C   REORDER THE COLUMNS OF THE SYSTEM OF EQUATIONS IN ACCORDANCE
C   WITH THE BOUNDARY CONDITIONS AND FORM THE SYSTEM MATRIX A
C   WHICH IS STORED IN H
C
      DO 155 I=1,N
      DO 150 J=1,2
      IF(KODE(2*I-2+J))110,110,150
  110 IF(I.NE.N .OR. J.NE.2) GO TO 125
      IF(KODE(1)) 115,115,113
  113 DO 114 K=1,N
      CH=H(K,1)
      H(K,1)=-G(K,2*N)
  114 G(K,2*N)=-CH
      GO TO 150
  115 DO 116 K=1,N
      H(K,1)=H(K,1)-G(K,2*N)
  116 G(K,2*N)=0.
      GO TO 150
  125 IF(I.EQ.1 .OR. J.GT.1 .OR. KODE(2*I-2).EQ.1) GO TO 130
      DO 129 K=1,N
      H(K,I)=H(K,I)-G(K,2*I-1)
  129 G(K,2*I-1)=0.
      GO TO 150
  130 DO 132 K=1,N
      CH=H(K,I-1+J)
      H(K,I-1+J)=-G(K,2*I-2+J)
  132 G(K,2*I-2+J)=-CH
  150 CONTINUE
  155 CONTINUE
C
C   FORM THE RIGHT HAND SIDE VECTOR F WHICH IS STORED IN FI
C
      DO 160 I=1,N
      FI(I)=0.
      DO 160 J=1,NN
      FI(I)=FI(I)+G(I,J)*DFI(J)
  160 CONTINUE
      RETURN
      END
```

Routine EXTINPL

This routine is similar to the one in POCONBE but instead of computing only one value per element for each coefficient as in POCONBE, it now computes two values per element, i.e. the parts of the coefficients corresponding to the adjacent nodes.

```
C-----------------------------------------------------------------------
      SUBROUTINE EXTINPL(XP,YP,X1,Y1,X2,Y2,A1,A2,B1,B2
     1,DQ1F1,DQ2F1,DQ1F2,DQ2F2,DU1F1,DU2F1,DU1F2,DU2F2,K)
C
C  PROGRAM 12
C
C  THIS SUBROUTINE COMPUTES THE INTEGRAL OF SEVERAL NON-SINGULAR
C  FUNCTIONS ALONG THE BOUNDARY ELEMENTS USING A FOUR POINTS
C  GAUSS QUADRATURE
C  WHEN K=0, THE OFF DIAGONAL COEFFICIENTS OF THE H AND G MATRICES
C  ARE COMPUTED
C  WHEN K=1, ALL THE COEFFICIENTS NEEDED FOR COMPUTATION
C  OF THE POTENTIAL AND FLUXES AT INTERNAL POINTS ARE
C  COMPUTED (A1,A2,B1,B2,F11,F21,F12,F22,E11,E21,E12,E22)
C
C  RA= RADIUS   = DISTANCE FROM THE COLOCATION POINT TO THE
C  GAUSS INTEGRATION POINTS ON THE BOUNDARY ELEMENTS
C  ETA1,ETA2    = COMPUNENTS OF THE UNIT NORMAL TO THE ELEMENT
C  RD1,RD2,RDN  = RADIUS DERIVATIVES

C
      DIMENSION XCO(4),YCO(4),GI(4),OME(4)
      DATA GI/0.86113631,-0.86113631,0.33998104,-0.33998104/
      DATA OME/0.34785485,0.34785485,0.65214515,0.65214515/
C
      AX=(X2-X1)/2
      BX=(X2+X1)/2
      AY=(Y2-Y1)/2
      BY=(Y2+Y1)/2
      SL=SQRT(AX**2+AY**2)
      ETA1= AY/SL
      ETA2=-AX/SL
      A1=0.
      A2=0.
      B1=0.
      B2=0.
      DQ1F1=0.
      DQ2F1=0.
      DQ1F2=0.
      DQ2F2=0.
      DU1F1=0.
      DU2F1=0.
      DU1F2=0.
      DU2F2=0.
C
C  COMPUTE THE TERMS TO BE INCLUDED IN THE G AND H MATRICES
C  OR THE TERMS NEEDED FOR COMPUTATION OF THE POTENTIAL
C  AND FLUXES AT INTERNAL POINTS
C
      DO 40 I=1,4
      XCO(I)=AX*GI(I)+BX
      YCO(I)=AY*GI(I)+BY
      RA=SQRT((XP-XCO(I))**2+(YP-YCO(I))**2)
      RD1=(XCO(I)-XP)/RA
      RD2=(YCO(I)-YP)/RA
      RDN=RD1*ETA1+RD2*ETA2
      F1=(1.-GI(I))/2.
      F2=(1.+GI(I))/2.
      IF(K) 30,30,10
   10 DU1=RD1*OME(I)*SL/RA
      DU2=RD2*OME(I)*SL/RA
      DQ1=-((2.*RD1**2-1.)*ETA1+2.*RD1*RD2*ETA2)*OME(I)*SL/RA**2
      DQ2=-((2.*RD2**2-1.)*ETA2+2.*RD1*RD2*ETA1)*OME(I)*SL/RA**2
      DQ1F1=DQ1F1+DQ1*F1
      DQ1F2=DQ1F2+DQ1*F2
      DQ2F1=DQ2F1+DQ2*F1
      DQ2F2=DQ2F2+DQ2*F2
      DU1F1=DU1F1+DU1*F1
      DU1F2=DU1F2+DU1*F2
      DU2F1=DU2F1+DU2*F1
      DU2F2=DU2F2+DU2*F2
```

```
30 H=RDN*OME(I)*SL/RA
   G=ALOG(1/RA)*OME(I)*SL
   A1=A1-F1*H
   A2=A2-F2*H
   B1=B1+F1*G
40 B2=B2+F2*G
   RETURN
   END
```

Routine LOCINPL

This routine computes now the part of the elements of the matrix **G** corresponding to the integrals along the elements which include the singularity. These integrals are:

$$B1 = \int_{\Gamma_j} \phi_1 \, ln \left(\frac{1}{r}\right) d\Gamma$$
$$B2 = \int_{\Gamma_j} \phi_2 \, ln \left(\frac{1}{r}\right) d\Gamma \tag{2.64}$$

Using the local system of coordinates in figure 2.9, the integrals can be written as

$$B1 = \int_{\text{Point (1)}}^{\text{Point (2)}} (1-\eta) \, ln \left(\frac{1}{r}\right) d\Gamma = l \int_0^1 (1-\eta) \, ln \left(\frac{1}{\eta l}\right) d\eta$$
$$B2 = \int_{\text{Point (1)}}^{\text{Point (2)}} \eta \, ln \left(\frac{1}{r}\right) d\Gamma = l \int_0^1 \eta \, ln \left(\frac{1}{\eta l}\right) d\eta \tag{2.65}$$

$$B1 = \frac{l}{2}\left[\frac{3}{2} - ln(l)\right]$$
$$B2 = \frac{l}{2}\left[\frac{1}{2} - ln(l)\right] \tag{2.66}$$

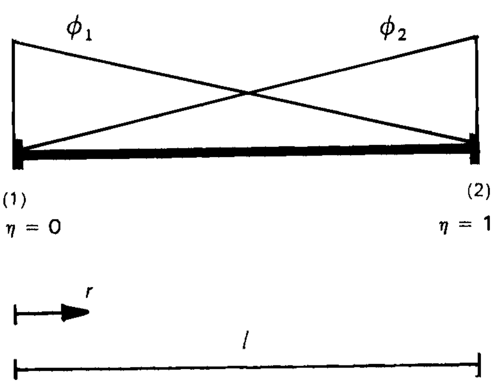


Figure 2.9 Linear element local coordinate system

```
C-----------------------------------------------------------------------
      SUBROUTINE LOCINPL(X1,Y1,X2,Y2,B1,B2)
C
C  PROGRAM 13
C
C  THIS SUBROUTINE COMPUTES THE PARTS OF THE G MATRIX
C  COEFFICIENTS CORRESPONDING TO INTEGRALS ALONG AN ELEMENT
C  THAT INCLUDES THE COLLOCATION POINT.
C
      SEP=SQRT((X2-X1)**2+(Y2-Y1)**2)
      B1=SEP*(1.5-ALOG(SEP))/2
      B2=SEP*(0.5-ALOG(SEP))/2
      RETURN
      END
```

Routine INTERPL

This routine replaces the INTERPC program used in POCONBE. It first arranges all potentials in FI and all their derivatives (or fluxes) in DFI and then computes the values of potentials and fluxes at the internal points if requested.

```
C-----------------------------------------------------------------------
      SUBROUTINE INTERPL(FI,DFI,KODE,CX,CY,X,Y,POT,FLUX1,FLUX2)
C
C  PROGRAM 14
C
C  THIS SUBROUTINE COMPUTES THE VALUES OF THE POTENTIAL
C  AND THE POTENTIAL DERIVATIVES (FLUXES) AT INTERNAL POINTS
C
      COMMON N,L,INP,IPR
      DIMENSION FI(1),DFI(1),KODE(1),CX(1),CY(1),X(1),Y(1)
      DIMENSION POT(1),FLUX1(1),FLUX2(1)
C
C  REARRANGE THE FI AND DFI ARRAYS TO STORE ALL THE VALUES OF THE
C  POTENTIAL IN FI AND ALL THE VALUES OF THE DERIVATIVE IN DFI
C
      DO 155 I=1,N
      DO 150 J=1,2
      IF(KODE(2*I-2+J))110,110,150
  110 IF(I.NE.N .OR. J.NE.2) GO TO 125
      IF(KODE(1)) 114,114,113
  113 CH=FI(1)
      FI(1)=DFI(2*N)
      DFI(2*N)=CH
      GO TO 150
  114 DFI(2*N)=DFI(1)
      GO TO 150
  125 IF(I.EQ.1 .OR. J.EQ.2 .OR. KODE(2*I-2).EQ.1) GO TO 130
      DFI(2*I-1)=DFI(2*I-2)
      GO TO 150
  130 CH=FI(I-1+J)
      FI(I-1+J)=DFI(2*I-2+J)
      DFI(2*I-2+J)=CH
  150 CONTINUE
  155 CONTINUE
C
C  COMPUTE THE POTENTIAL AND THE FLUXES AT INTERNAL POINTS
C
      IF(L.EQ.0) GO TO 50
      DO 40 K=1,L
      POT(K)=0.
      FLUX1(K)=0.
      FLUX2(K)=0.
      DO 30 J=1,N
      CALL EXTINPL(CX(K),CY(K),X(J),Y(J),X(J+1),Y(J+1),A1,A2,B1,B2
     1,DQ1F1,DQ2F1,DQ1F2,DQ2F2,DU1F1,DU2F1,DU1F2,DU2F2,1)
      IF(J-N)32,33,33
   32 POT(K)=POT(K)+DFI(2*J-1)*B1+DFI(2*J)*B2-FI(J)*A1-FI(J+1)*A2
      FLUX1(K)=FLUX1(K)+DFI(2*J-1)*DU1F1+DFI(2*J)*DU1F2
     1-FI(J)*DQ1F1-FI(J+1)*DQ1F2
      FLUX2(K)=FLUX2(K)+DFI(2*J-1)*DU2F1+DFI(2*J)*DU2F2
     1-FI(J)*DQ2F1-FI(J+1)*DQ2F2
      GO TO 30
   33 POT(K)=POT(K)+DFI(2*J-1)*B1+DFI(2*J)*B2-FI(J)*A1-FI(1)*A2
      FLUX1(K)=FLUX1(K)+DFI(2*J-1)*DU1F1+DFI(2*J)*DU1F2
     1-FI(J)*DQ1F1-FI(1)*DQ1F2
      FLUX2(K)=FLUX2(K)+DFI(2*J-1)*DU2F1+DFI(2*J)*DU2F2
     1-FI(J)*DQ2F1-FI(1)*DQ2F2
   30 CONTINUE
      POT(K)=POT(K)/(2.*3.1415926)
      FLUX1(K)=FLUX1(K)/(2.*3.1415926)
   40 FLUX2(K)=FLUX2(K)/(2.*3.1415926)
   50 RETURN
      END
```

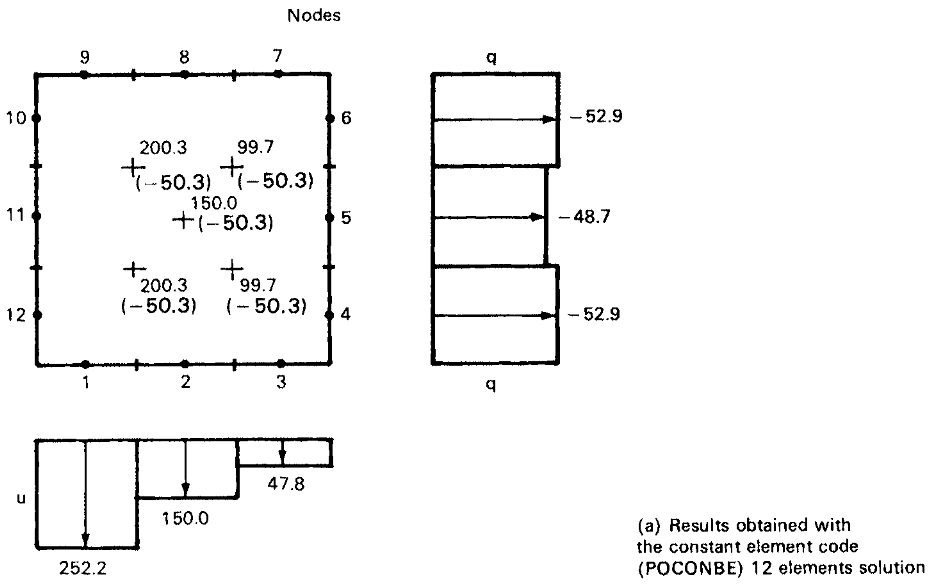
Routine OUTPTPL

This routine is similar to the one described in POCONBE but instead of printing the mid-point coordinates it now gives directly the values of the coordinates in the *X* and *Y* arrays. It also gives two values for the flux at each boundary node; one 'before' and the other 'after' the node.

```
C-----------------------------------------------------------------------
      SUBROUTINE OUTPTPL(X,Y,FI,DFI,CX,CY,POT,FLUX1,FLUX2)
C
C  PROGRAM 15
C
C  THIS SUROUTINE PRINTS THE VALUES OF THE POTENTIAL AND ITS
C  NORMAL DERIVATIVE AT BOUNDARY NODES. IT ALSO PRINTS THE
C  VALUES OF THE POTENTIAL AT INTERNAL POINTS.
C
      COMMON N,L,INP,IPR
      DIMENSION X(1),Y(1),FI(1),DFI(1),CX(1),CY(1)
      DIMENSION POT(1),FLUX1(1),FLUX2(2)
C
      WRITE(IPR,100)
  100 FORMAT(' ',79('*')//2X,'RESULTS'//2X,'BOUNDARY NODES'//
     156X,'POTENTIAL DERIVATIVE'/
     29X,'X',15X,'Y',12X,'POTENTIAL',6X,'BEFORE NODE',6X,'AFTER NODE'/)
      WRITE(IPR,200) X(1),Y(1),FI(1),DFI(2*N),DFI(1)
      DO 10 I=2,N
   10 WRITE(IPR,200) X(I),Y(I),FI(I),DFI(2*I-2),DFI(2*I-1)
  200 FORMAT(5(2X,E14.5))
C
      IF(L.EQ.0) GO TO 30
      WRITE(IPR,300)
  300 FORMAT(//,2X,'INTERNAL POINTS',//9X,'X',15X,'Y',12X,'POTENTIAL',
     19X,'FLUX X',10X,'FLUX Y'/)
      DO 20 K=1,L
   20 WRITE(IPR,400)CX(K),CY(K),POT(K),FLUX1(K),FLUX2(K)
  400 FORMAT(5(2X,E14.5))
   30 WRITE(IPR,500)
  500 FORMAT(' ',79('*'))
      RETURN
      END
```

Example 2.2

The potential problem solved with POCONBE will now be analysed using linear elements as shown in figure 2.10. The number of elements is still 12 for the linear (figure 2.10(b)) as well as the constant (figure 2.10(a)). Although the constant element solution gives reasonable agreement with the known results, the linear solution is identical to the analytical one within the precision limits of the computer. This result was to be expected since the exact solution varies linearly.



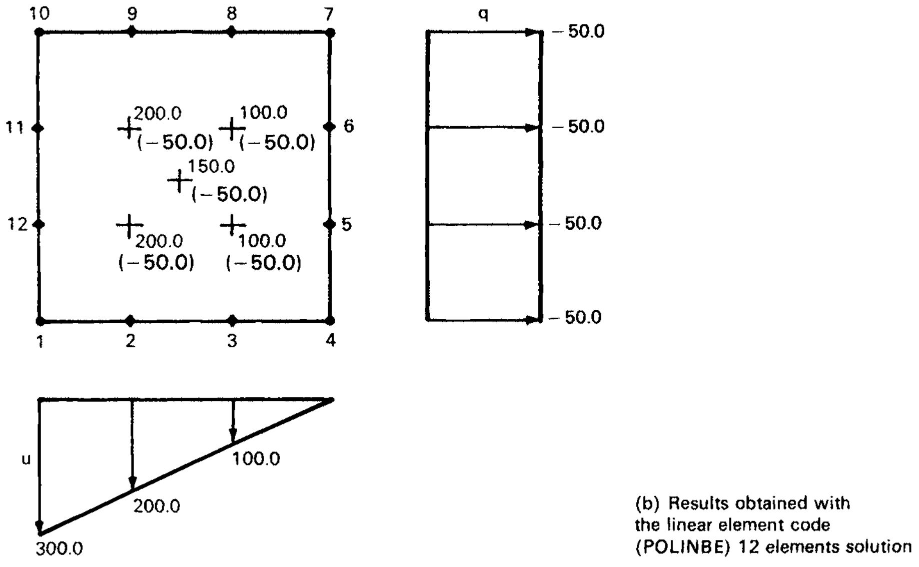
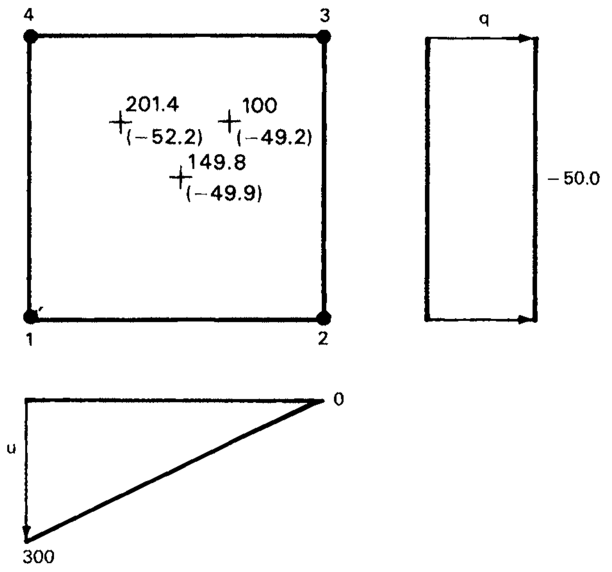


Figure 2.10 Results obtained using constant and linear elements for the heat flow example



(c) Results obtained with the linear element code (POLINBE) 4 elements solution

Figure 2.10 *continued*

The data corresponding to the 12 element problem is as follows.

HEAT FLOW EXAMPLE (DATA)

```
  HEAT FLOW EXAMPLE (12 LINEAR ELEMENTS)
12 5
0. 0. 2. 0. 4. 0. 6. 0. 6. 2. 6. 4. 6. 6.
4. 6. 2. 6. 0. 6. 0. 4. 0. 2.
1 0. 1 0.
1 0. 1 0.
1 0. 1 0.
0 0. 0 0.
0 0. 0 0.
0 0. 0 0.
1 0. 1 0.
1 0. 1 0.
1 0. 1 0.
0 300. 0 300.
0 300. 0 300.
0 300. 0 300.
2. 2. 2. 4. 3. 3. 4. 2. 4. 4.
```

and the output is given by

HEAT FLOW EXAMPLE (OUTPUT)

```
********************************************************************************
 HEAT FLOW EXAMPLE (12 LINEAR ELEMENTS)

DATA

 NUMBER OF BOUNDARY ELEMENTS = 12
 NUMBER OF INTERNAL POINTS WHERE THE FUNCTION IS CALCULATED =  5
```

```
COORDINATES OF THE EXTREME POINTS OF THE BOUNDARY ELEMENTS

POINT          X                    Y
  1        .00000E+00           .00000E+00
  2        .20000E+01           .00000E+00
  3        .40000E+01           .00000E+00
  4        .60000E+01           .00000E+00
  5        .60000E+01           .20000E+01
  6        .60000E+01           .40000E+01
  7        .60000E+01           .60000E+01
  8        .40000E+01           .60000E+01
  9        .20000E+01           .60000E+01
 10        .00000E+00           .60000E+01
 11        .00000E+00           .40000E+01
 12        .00000E+00           .20000E+01

BOUNDARY CONDITIONS

                ------FIRST NODE-----            -----SECOND NODE-----
                 PRESCRIBED                        PRESCRIBED
ELEMENT            VALUE         CODE                VALUE         CODE
  1            .0000000E+00       1             .0000000E+00        1
  2            .0000000E+00       1             .0000000E+00        1
  3            .0000000E+00       1             .0000000E+00        1
  4            .0000000E+00       0             .0000000E+00        0
  5            .0000000E+00       0             .0000000E+00        0
  6            .0000000E+00       0             .0000000E+00        0
  7            .0000000E+00       1             .0000000E+00        1
  8            .0000000E+00       1             .0000000E+00        1
  9            .0000000E+00       1             .0000000E+00        1
 10            .3000000E+03       0             .3000000E+03        0
 11            .3000000E+03       0             .3000000E+03        0
 12            .3000000E+03       0             .3000000E+03        0
*******************************************************************************

RESULTS

BOUNDARY NODES

                                                         POTENTIAL DERIVATIVE
       X                Y              POTENTIAL      BEFORE NODE      AFTER NODE

  .00000E+00       .00000E+00       .30000E+03       .50000E+02       .00000E+00
  .20000E+01       .00000E+00       .20000E+03       .00000E+00       .00000E+00
  .40000E+01       .00000E+00       .10000E+03       .00000E+00       .00000E+00
  .60000E+01       .00000E+00       .00000E+00       .00000E+00      -.50000E+02
  .60000E+01       .20000E+01       .00000E+00      -.50000E+02      -.50000E+02
  .60000E+01       .40000E+01       .00000E+00      -.50000E+02      -.50000E+02
  .60000E+01       .60000E+01       .00000E+00      -.50000E+02       .00000E+00
  .40000E+01       .60000E+01       .10000E+03       .00000E+00       .00000E+00
  .20000E+01       .60000E+01       .20000E+03       .00000E+00       .00000E+00
  .00000E+00       .60000E+01       .30000E+03       .00000E+00       .50000E+02
  .00000E+00       .40000E+01       .30000E+03       .50000E+02       .50000E+02
  .00000E+00       .20000E+01       .30000E+03       .50000E+02       .50000E+02

INTERNAL POINTS

       X                Y              POTENTIAL         FLUX X          FLUX Y

  .20000E+01       .20000E+01       .20000E+03      -.50001E+02      -.69996E-03
  .20000E+01       .40000E+01       .20000E+03      -.50001E+02       .69519E-03
  .30000E+01       .30000E+01       .15000E+03      -.50000E+02      -.99891E-06
  .40000E+01       .20000E+01       .10000E+03      -.50000E+02      -.34618E-03
  .40000E+01       .40000E+01       .10000E+03      -.50000E+02       .34851E-03
*******************************************************************************
```

Example 2.3

It is interesting to note that in this case even a simple four elements representation can give exact results using double value of the flux at the corners (figure 2.10(c)).

The input in this case is

HEAT FLOW EXAMPLE (DATA)

```
 HEAT FLOW EXAMPLE (4 LINEAR ELEMENTS)
4 3
0. 0. 6. 0. 6. 6. 0. 6.
1 0. 1 0.
0 0. 0 0.
1 0. 1 0.
0 300. 0 300.
2. 4. 3. 3. 4. 4.
```

The corresponding output is very accurate taking into consideration the simplicity of the mesh.

HEAT FLOW EXAMPLE (OUTPUT)

```
******************************************************************************
HEAT FLOW EXAMPLE (4 LINEAR ELEMENTS)


DATA

 NUMBER OF BOUNDARY ELEMENTS =  4
 NUMBER OF INTERNAL POINTS WHERE THE FUNCTION IS CALCULATED =  3


 COORDINATES OF THE EXTREME POINTS OF THE BOUNDARY ELEMENTS

 POINT           X                 Y
    1        .00000E+00        .00000E+00
    2        .60000E+01        .00000E+00
    3        .60000E+01        .60000E+01
    4        .00000E+00        .60000E+01


 BOUNDARY CONDITIONS

              ------FIRST NODE-----         -----SECOND NODE-----
                PRESCRIBED                    PRESCRIBED
ELEMENT            VALUE        CODE             VALUE        CODE
   1           .0000000E+00      1           .0000000E+00      1
   2           .0000000E+00      0           .0000000E+00      0
   3           .0000000E+00      1           .0000000E+00      1
   4           .3000000E+03      0           .3000000E+03      0
******************************************************************************

 RESULTS

 BOUNDARY NODES

                                                      POTENTIAL DERIVATIVE
        X               Y            POTENTIAL      BEFORE NODE      AFTER NODE

   .00000E+00      .00000E+00      .30000E+03      .50000E+02      .00000E+00
   .60000E+01      .00000E+00      .00000E+00      .00000E+00     -.50000E+02
   .60000E+01      .60000E+01      .00000E+00     -.50000E+02      .00000E+00
   .00000E+00      .60000E+01      .30000E+03      .00000E+00      .50000E+02


 INTERNAL POINTS

        X               Y            POTENTIAL        FLUX X          FLUX Y

   .20000E+01      .40000E+01      .20138E+03     -.52208E+02      .26160E+01
   .30000E+01      .30000E+01      .14979E+03     -.49931E+02      .50752E-06
   .40000E+01      .40000E+01      .10001E+03     -.49244E+02      .34782E+00
******************************************************************************
```

2.7 Discontinuous Elements

To avoid the problem of having two unknown fluxes at a corner node (for which only one boundary element equation can be written) the nodes of the two linear elements which meet at the corner can be shifted inside the two elements. The nodes remain as two distinct nodes (see figure 2.11) and one equation can be written for each node. The potential and the flux are represented by linear functions along the whole elements in terms of their nodal values and both of them are in principle discontinuous at the corner. Discontinuous elements are also useful for situations in which one of the variables takes an infinite value at the end of the element (for instance at a reentry corner or in fracture mechanics applications). In such cases the value of the variable at the node shifted from the end of the element is finite and can be computed from the system of equations without numerical difficulties.

The values of u and q at any point on a linear element have been defined in terms of their values at the extreme points by equation (2.42).

$$\begin{aligned} u(\xi) &= \phi_1 u^1 + \phi_2 u^2 = [\phi_1 \phi_2] \begin{Bmatrix} u^1 \\ u^2 \end{Bmatrix} \\ q(\xi) &= \phi_1 q^1 + \phi_2 q^2 = [\phi_1 \phi_2] \begin{Bmatrix} q^1 \\ q^2 \end{Bmatrix} \end{aligned} \qquad (2.67)$$

If the two nodes of an element have been shifted from the ends distances a and b respectively as shown in figure 2.11 any of the two equations (2.67) can be particularized for the nodes.

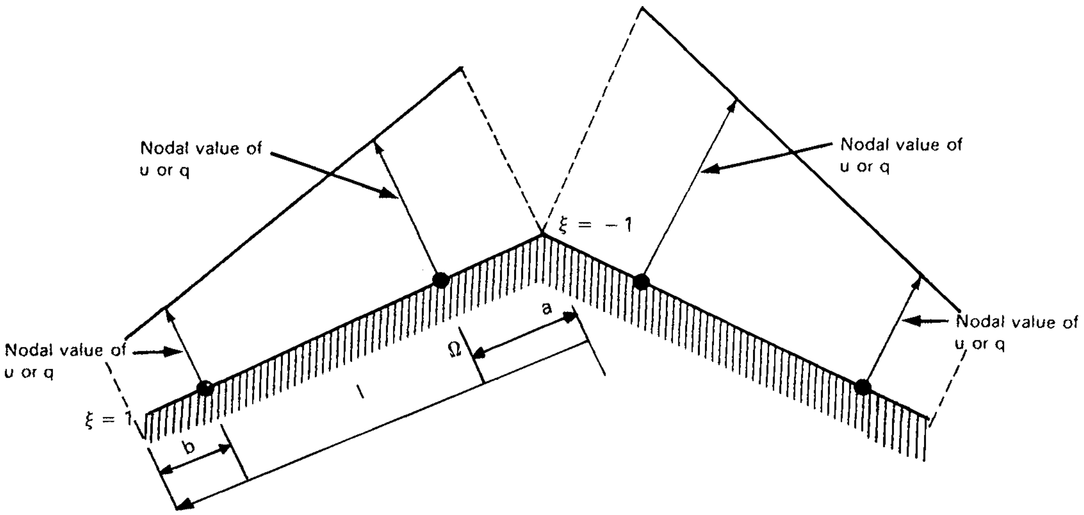


Figure 2.11 Discontinuous elements

$$\begin{Bmatrix} u^a \\ u^b \end{Bmatrix} = \begin{bmatrix} \phi_1(\xi_a) & \phi_2(\xi_a) \\ \phi_1(\xi_b) & \phi_2(\xi_b) \end{bmatrix} \begin{Bmatrix} u^1 \\ u^2 \end{Bmatrix} \tag{2.68}$$

where $\xi_a = (2a/l) - 1$ and $\xi_b = 1 - (2b/l)$ are the local coordinates for the nodal points.

Equation (2.68) can be inserted and after substitution into (2.67) yields the value of u at any point on the element in terms of the nodal values

$$u(\xi) = [\phi_1 \phi_2]\ \mathbf{Q} \begin{Bmatrix} u^a \\ u^b \end{Bmatrix} \tag{2.69}$$

where

$$Q = \frac{1}{l-a-b} \begin{bmatrix} l-b & -a \\ -b & l-a \end{bmatrix}$$

The same relation can be written for the flux

$$q(\xi) = [\phi_1 \phi_2]\ \mathbf{Q} \begin{Bmatrix} q^a \\ q^b \end{Bmatrix} \tag{2.70}$$

After discretizing the boundary into N elements the integral statement for a node "i" can be written as

$$c^i u^i + \sum_{j=1}^{N} \int_{\Gamma_j} u q^*\, d\Gamma = \sum_{j=1}^{N} \int_{\Gamma_j} u^* q\, d\Gamma \tag{2.71}$$

The integrals over a discontinuous element "j" are

$$\begin{aligned} \int_{\Gamma_j} u q^*\, d\Gamma &= \int_{\Gamma_j} [\phi_1 \phi_2]\ \mathbf{Q}\ q^*\, d\Gamma \begin{Bmatrix} u^a \\ u^b \end{Bmatrix} = [h_a^{ij}\ h_b^{ij}] \begin{Bmatrix} u^a \\ u^b \end{Bmatrix} \\ \int_{\Gamma_j} q u^*\, d\Gamma &= \int_{\Gamma_j} [\phi_1 \phi_2]\ \mathbf{Q}\ u^*\, d\Gamma \begin{Bmatrix} q^a \\ q^b \end{Bmatrix} = [g_a^{ij}\ g_b^{ij}] \begin{Bmatrix} q^a \\ q^b \end{Bmatrix} \end{aligned} \tag{2.72}$$

When solving a potential problem continuous and discontinuous elements can be used together in the same mesh. The total number of nodes will be equal to the total number of elements plus one additional node per each discontinuous element.

The coefficient c^i is equal to 0.5 for the nodes on discontinuous elements. The integrals h_a^{ij}, h_b^{ij}, g_a^{ij} and g_b^{ij} along the discontinuous elements given by equation (2.72) can be computed by the usual Gaussian numerical quadrature when the node "i" does not belong to the element. When "i" is one of the nodes of the discontinuous element $h_a^{ij} = h_b^{ij} = 0$ and g_a^{ij} and g_b^{ij} can be easily obtained by analytical integration. The element is subdivided into two parts one at each side of the node. The resulting integrals consist of the same basic integrals as those of the regular linear elements given by equation (2.66).

2.8 Quadratic and Higher Order Elements

It is usually more convenient for arbitrary geometries to implement some type of curvilinear elements. The simplest of these are the three noded quadratic elements which require working with transformations. Consider the curved boundary shown in figure 2.12 where Γ is defined along the boundary and the $\vec{r}$ position vector is a function of the cartesian system, x_1, x_2. The variables u or q can be written in terms of interpolation functions which are functions of the homogeneous coordinate ξ, i.e.

$$u(\xi) = \phi_1 u^1 + \phi_2 u^2 + \phi_3 u^3 = [\phi_1 \phi_2 \phi_3] \begin{Bmatrix} u^1 \\ u^2 \\ u^3 \end{Bmatrix}$$
$$q(\xi) = \phi_1 q^1 + \phi_2 q^2 + \phi_3 q^3 = [\phi_1 \phi_2 \phi_3] \begin{Bmatrix} q^1 \\ q^2 \\ q^3 \end{Bmatrix} \tag{2.73}$$

where the interpolation functions are

$$\phi_1 = \tfrac{1}{2}\xi(\xi - 1); \qquad \phi_2 = (1 - \xi)(1 + \xi); \qquad \phi_3 = \tfrac{1}{2}\xi(1 + \xi) \tag{2.74}$$

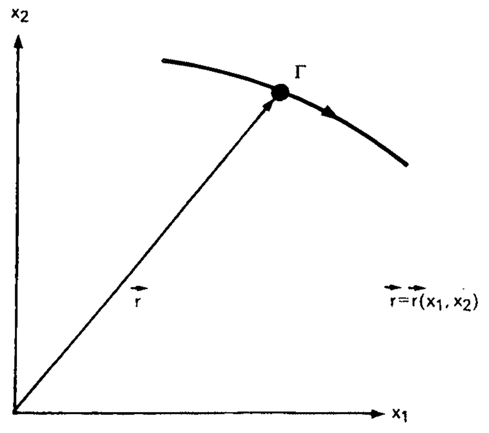


Figure 2.12 Curved boundary

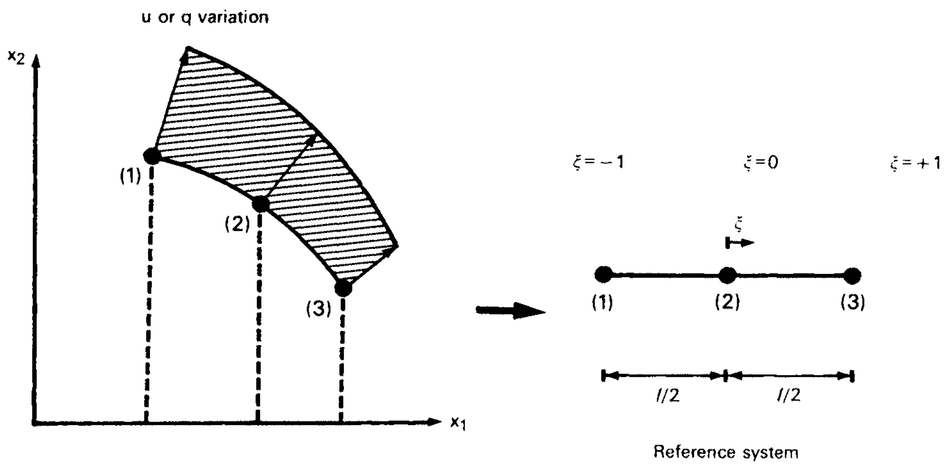


Figure 2.13 Quadratic element

These functions are quadratic in ξ and give the nodal values of the variable u or q when specialized for the nodes; i.e. with reference to figure 2.13

Node	ξ	ϕ_1	ϕ_2	ϕ_3
1	-1	1	0	0
2	0	0	1	0
3	$+1$	0	0	1

(2.75)

The integrals along any 'j' element are similar to those computed for the linear element, but there are now three nodal unknowns and the integration requires the use of a Jacobian. Consider for instance an integral for the H type terms, i.e.

$$\int_{\Gamma_j} u(\xi)q^*\,d\Gamma = \int_{\Gamma_j} [\phi_1\phi_2\phi_3]q^*\,d\Gamma \begin{Bmatrix} u^1 \\ u^2 \\ u^3 \end{Bmatrix}$$

$$= [h_1^{ij}h_2^{ij}h_3^{ij}] \begin{Bmatrix} u^1 \\ u^2 \\ u^3 \end{Bmatrix} \tag{2.76}$$

where

$$h_1^{ij} = \int_{\Gamma_j} \phi_1 q^*\,d\Gamma; \qquad h_2^{ij} = \int_{\Gamma_j} \phi_2 q^*\,d\Gamma; \qquad h_3^{ij} = \int_{\Gamma_j} \phi_3 q^*\,d\Gamma \tag{2.77}$$

The evaluation of these terms requires the use of a Jacobian as the ϕ_i functions are expressed in terms of ξ, but the integrals are functions of Γ. For a curve such as given in figure 2.12, the transformation is simple,

$$d\Gamma = \left\{ \sqrt{\left(\frac{dx_1}{d\xi}\right)^2 + \left(\frac{dx_2}{d\xi}\right)^2} \right\} d\xi = |G|\, d\xi \tag{2.78}$$

where $|G|$ is the Jacobian. Hence one can write,

$$h_k^{ij} = \int_{\Gamma_j} \phi_k(\xi) q^* \, d\Gamma = \int_{\text{Node 1}}^{\text{Node 2}} \phi_k(\xi) q^* |G| \, d\xi \tag{2.79}$$

Formulae such as (2.66) are generally too difficult to integrate analytically and numerical integration must be used in all cases, including those elements with a singularity.

Notice that in order to calculate the value of the Jacobian $|G|$ in (2.78) one needs to know the variation of the x_1 and x_2 coordinates in terms of ξ. This can be done by defining the geometrical shape of the element in the same way as the variables u and q are defined, i.e. using quadratic interpolation,

$$\begin{aligned} x_1 &= \phi_1 x_1^1 + \phi_2 x_1^2 + \phi_3 x_1^3 \\ x_2 &= \phi_2 x_2^1 + \phi_2 x_2^2 + \phi_2 x_2^3 \end{aligned} \tag{2.80}$$

where the superscript indicates the number of the node. This is a similar concept to the isoparametric elements commonly used in finite elements.

Cubic Elements

Elements of order higher than quadratic are seldom used in practice, but they may be interesting in some particular applications. Because of this we will briefly describe the case of elements with cubic variation of geometry, and u or q variables. In this case the functions are described by taking four nodes over each element (figure 2.14).

$$\begin{aligned} u &= \phi_1 u^1 + \phi_2 u^2 + \phi_3 u^3 + \phi_4 u^4 \\ q &= \phi_1 q^1 + \phi_2 q^2 + \phi_3 q^3 + \phi_4 q^4 \end{aligned} \tag{2.81}$$

and similarly

$$\begin{aligned} x_1 &= \phi_1 x_1^1 + \phi_2 x_1^2 + \phi_3 x_1^3 + \phi_4 x_1^4 \\ x_2 &= \phi_1 x_2^1 + \phi_2 x_2^2 + \phi_3 x_2^3 + \phi_4 x_2^4 \end{aligned} \tag{2.82}$$

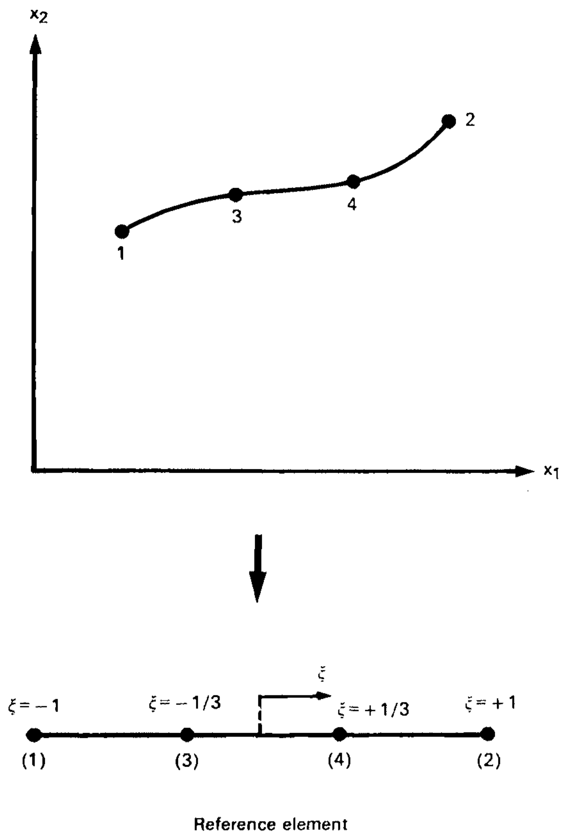


Figure 2.14 Cubic elements with four nodes

where the interpolation functions are,

$$\phi_1 = \tfrac{1}{16}(1-\xi)[-10+9(\xi^2+1)] \qquad \phi_2 = \tfrac{1}{16}(1+\xi)[-10+9(\xi^2+1)]$$
$$\phi_3 = \tfrac{9}{16}(1-\xi^2)(1-3\xi) \qquad \phi_4 = \tfrac{9}{16}(1-\xi^2)(1+3\xi) \qquad (2.83)$$

which can be specialized at the nodes as follows,

Node	ξ	ϕ_1	ϕ_2	ϕ_3	ϕ_4
1	-1	1	0	0	0
2	$-1/3$	0	1	0	0
3	$1/3$	0	0	1	0
4	1	0	0	0	1

(2.84)

Another possibility with cubic elements is to define the variation of u or q in terms of the function and its derivative along the element, at the two end points – i.e nodes 1 and 2 – as shown in figure 2.15. The corresponding function for u (same applies for q and $x_1 x_2$) is then given by

$$u = \phi_1 u^1 + \phi_2 \left(\frac{\partial u}{\partial \Gamma}\right)^1 + \phi_3 u^2 + \phi_4 \left(\frac{\partial u}{\partial \Gamma}\right)^2 \tag{2.85}$$

with

$$\begin{aligned} \phi_1 &= \tfrac{1}{4}(\xi - 1)^2(\xi + 2) & \phi_2 &= -\tfrac{1}{4}l(\xi - 1)^2(\xi + 1) \\ \phi_3 &= \tfrac{1}{4}(\xi + 1)^2(\xi - 2) & \phi_4 &= -\tfrac{1}{4}l(\xi + 1)^2(\xi - 1) \end{aligned} \tag{2.86}$$

where l is the element length.

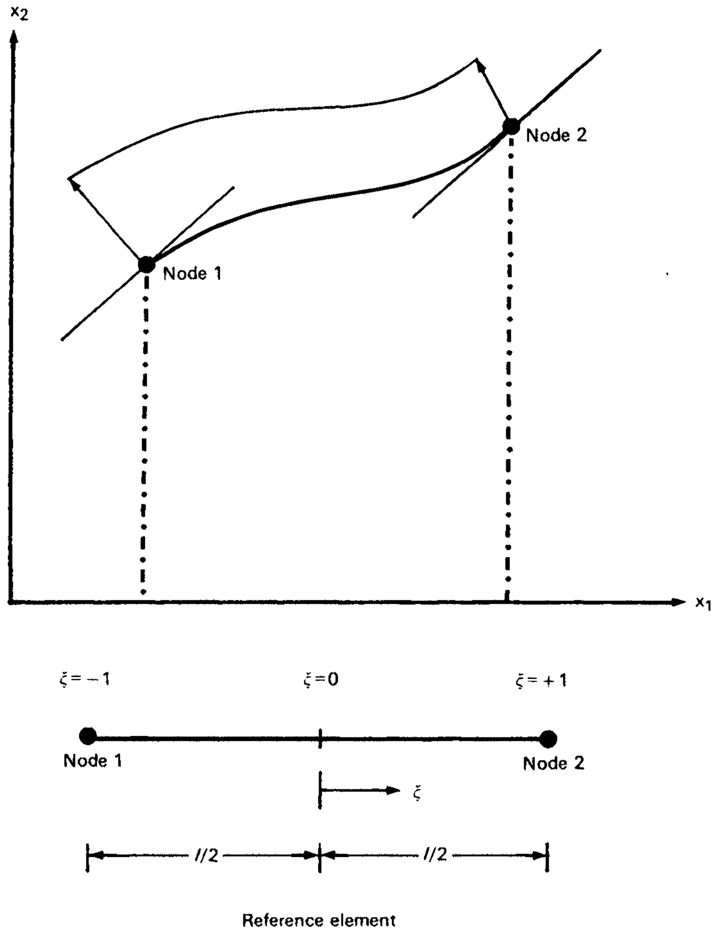


Figure 2.15 Cubic elements with only two nodes

This last type of cubic element could be used in cases where we wish to have a correct definition of the derivative along Γ, for instance to calculate fluxes in that direction, or if we prefer to reduce the number of nodes along the element. In some cases it may still be better to continue defining the geometry with four nodes as it is generally more difficult to have accurate results for the slopes.

2.9 Computer Code for Potential Problems using Quadratic Elements (POQUABE)

In what follows a FORTRAN code for potential problems using quadratic elements is described. The program has the same organization as the two previously studied.

All variables in the code have the same meaning as for the linear element program (POLINBE). FI and DFI have a slightly different form. The dimension of FI is (N), N being the number of nodes and that of DFI is $(3NE)$, where NE is the number of elements. The prescribed boundary conditions are read in DFI (three per element).

The program allows for the values of flux at both sides of the nodes connecting two elements to be different. Then, (i) when both fluxes are prescribed as different at both sides of the node, the potential is the only unknown; (ii) when the potential and one flux are prescribed, the other flux is the unknown and (iii) if only the potential is prescribed, one value of the flux is the unknown and will be the same on both sides of the node. Thus, the situation at corner nodes of quadratic elements is the same as for linear elements.

Main Program

The program follows the same structure of the constant and linear potential codes.

The listing is as follows.

```
C-------------------------------------------------------------------
C
      PROGRAM POQUABE
C
C  PROGRAM 16
C
C
C  THIS PROGRAM SOLVES TWO DIMENSIONAL (PO)TENTIAL PROBLEMS
C  USING (QUA)DRATIC (B)OUNDARY (E)LEMENTS.
C
      CHARACTER*10 FILEIN,FILEOUT
      COMMON/MATG/G(100,150)
      COMMON/MATH/H(100,100)
      COMMON N,L,INP,IPR
      DIMENSION X(101),Y(101)
      DIMENSION DFI(150),FI(100),KODE(150)
      DIMENSION CX(20),CY(20),POT(20),FLUX1(20),FLUX2(20)
C
C  SET MAXIMUN DIMENSION OF THE SYSTEM OF EQUATIONS (NX)
C  NX= MAXIMUN NUMBER OF NODES= 2*MAXIMUN NUMBER OF ELEMENTS
C  NX1= 3*MAXIMUN NUMBER ELEMENTS
C
      NX=100
      NX1=150
```

```
C
C  ASSIGN NUMBERS FOR INPUT AND OUTPUT FILES
C
      INP=5
      IPR=6
C
C  READ NAMES AND OPEN  FILES FOR INPUT AND OUTPUT
C
      WRITE(*,' (A) ') ' NAME OF INPUT FILE (MAX. 10 CHART.)'
      READ(*,'  (A) ')FILEIN
      OPEN(INP,FILE=FILEIN,STATUS='OLD')
      WRITE(*,' (A) ') ' NAME OF OUTPUT FILE (MAX. 10 CHART.)'
      READ(*,' (A) ')FILEOUT
      OPEN(IPR,FILE=FILEOUT,STATUS='NEW')
C
C  READ DATA
C
      CALL INPUTPQ(CX,CY,X,Y,KODE,DFI)
C
C  COMPUTE H AND G MATRICES AND FORM SYSTEM (A X = F)
C
      CALL GHMATPQ(X,Y,G,H,FI,DFI,KODE,NX,NX1)
C
C  SOLVE SYSTEM OF EQUATIONS
C
      CALL SLNPD(H,FI,D,N,NX)
C
C  COMPUTE POTENTIAL VALUES AT INTERNAL POINTS
C
      CALL INTERPQ(FI,DFI,KODE,CX,CY,X,Y,POT,FLUX1,FLUX2)
C
C  PRINT RESULTS AT BOUNDARY NODES AND INTERNAL POINTS
C
      CALL OUTPTPQ(X,Y,FI,DFI,CX,CY,POT,FLUX1,FLUX2)
C
C   CLOSE INPUT AND OUTPUT FILES
C
      CLOSE (INP)
      CLOSE (IPR)
      STOP
      END
```

Routine INPUTPQ

This subroutine reads all the input required by the program and requests a file from the user containing the following lines:

(i) *Title Line* Contains the title of the problem

(ii) *Basic Parameter Line* Contains the number of elements and the number of internal points.

(iii) *Boundary Nodes Coordinates Lines* Contains x_1x_2 coordinates read counterclockwise for external boundaries and clockwise for internal ones. The lines are organized in free format.

(iv) *Boundary Conditions Lines* As many lines as boundary elements. Three values of KODE and the known variables are read for each element, corresponding to the three nodes. In this way a value of the flux may be prescribed for an extreme node as part of one element and a different value as part of the other element. The potential however must be unique for any node and the flux must also be unique for the mid-node of any element. The known variables are the potential if KODE $= 0$ and the flux if KODE $= 1$. The order of reading is first KODE(I) and then value of the variable (I) for $I = 1, 2, 3$.

(v) *Internal Points Coordinates Lines* Contain x_1x_2 coordinates of the internal points organized in free format. There will be one or more lines if necessary.

This subroutine first prints the name of the run and the basic parameters. Then the coordinates of the nodes and the boundary conditions for each element, with codes and prescribed values are printed. The internal points coordinates are only printed in the subroutine OUTPTPQ.

The FORTRAN listing of INPUTPQ is as follows:

```
C-----------------------------------------------------------------------
      SUBROUTINE INPUTPQ(CX,CY,X,Y,KODE,DFI)
C
C  PROGRAM 17
C
C  NE= NUMBER OF BOUNDARY ELEMENTS
C  N = NUMBER OF BOUNDARY NODES = 2 * NE
C  L = NUMBER OF INTERNAL POINTS
C
      CHARACTER*80 TITLE
      COMMON N,L,INP,IPR
      DIMENSION KODE(1),X(1),Y(1),CX(1),CY(1),DFI(1)
      WRITE(IPR,100)
  100 FORMAT(' ',79('*'))
C
C  READ JOB TITLE
C
      READ(INP,'(A)') TITLE
      WRITE(IPR,'(A)') TITLE
C
C  READ NUMBER OF BOUNDARY ELEMENTS AND INTERNAL POINTS
C
      READ(INP,*)NE,L
      WRITE(IPR,210)NE,L
  210 FORMAT(//2X,'DATA'/2X,'NUMBER OF BOUNDARY ELEMENTS=',
     1I3/2X,'NUMBER OF INTERNAL POINTS=',I3)
      N=2*NE
C
C  READ BOUNDARY NODES COORDINATES IN ARRAYS X AND Y
C
      WRITE(IPR,500)
      READ(INP,*) (X(I),Y(I),I=1,N)
      DO 10 I=1,N
   10 WRITE(IPR,240) I,X(I),Y(I)
  500 FORMAT(//2X,'BOUNDARY NODES COORDINATES'///4X,
     1'NODE',10X,'X',18X,'Y'/)
  240 FORMAT(5X,I3,2(5X,E14.7))
C
C  READ BOUNDARY CONDITIONS IN DFI(I) VECTOR,IF KODE(I)=0
C  THE DFI(I) VALUE IS A KNOWN POTENTIAL; IF KODE(I)=1 THE
C  DFI(I) VALUE IS A KNOWN POTENTIAL DERIVATIVE (FLUX).
C  THREE BOUNDARY CONDITIONS ARE READ PER ELEMENT.
C  NODES BETWEEN TWO ELEMENTS MAY HAVE TWO DIFFERENT VALUES
C  OF THE POTENTIAL DERIVATIVE BUT ONLY ONE VALUE OF THE
C  POTENTIAL.
C
      WRITE(IPR,800)
  800 FORMAT(//2X,'BOUNDARY CONDITIONS'//11X,
     1'-----FIRST NODE------',3X,'-----SECOND NODE-----',3X,
     2'-----THIRD NODE------'/13X,'PRESCRIBED',14X,
     3'PRESCRIBED',14X,'PRESCRIBED'/1X,'ELEMENT',8X,'VALUE',
     47X,'CODE',8X,'VALUE',7X,'CODE',8X,'VALUE',7X,'CODE')
      DO 20 I=1,NE
      READ(INP,*) (KODE(3*I-3+J),DFI(3*I-3+J),J=1,3)
   20 WRITE(IPR,950)I,(DFI(3*I-3+J),KODE(3*I-3+J),J=1,3)
  950 FORMAT(3X,I3,2X,3(4X,E14.7,5X,I1))
C
C  READ COORDINATES OF THE INTERNAL POINTS
C
      IF(L.EQ.0) GO TO 30
      READ(INP,*) (CX(I),CY(I),I=1,L)
   30 RETURN
      END
```

Routine GHMATPQ

This subroutine computes the **G** and **H** system matrices by calling routines EXTINPQ and LOCINPQ.

EXTINPQ: Computes the **GW** and **HW** (3) submatrices which relate a collocation point with an element as defined by its three nodes. The collocation point is not any one of the element nodes.

LOCINPQ: Computes the **GW** (3) submatrix for the case when the collocation point is one of the nodes within the element under consideration (i.e. the singularity is in the same element). Notice that the corresponding **HW** (3) is computed using EXTINPQ because the singularity will occur only on the diagonal term and this is computed later on by adding the off-diagonal terms of the row.

The resulting **GW** and **HW** submatrices are assembled in the **G** and **H** system matrices. Matrix **G** is now rectangular since each extreme node of an element may have different fluxes, i.e. one 'before' and another 'after' the node. The diagonal terms in **H** are computed using constant potential considerations, which results in adding row coefficients together.

Once the matrices **H** and **G** are assembled, the system of equations needs to be reordered in accordance with the boundary conditions to form

$$\mathbf{AX} = \mathbf{F}$$

where **X** is a (N) vector of unknowns, N being the number of nodes; **A** is a $(N \times N)$ matrix whose columns are a combination of columns of **H** or **G** depending on the boundary conditions or of two consecutive columns of **G** when the unknown is the *unique* value of the tractions at both sides of the extreme node of an element; **F** is a known vector computed by multiplying the prescribed boundary conditions by the corresponding row terms of **G** or **H**.

At the end of the subroutine GHMATPQ and after rearranging **H** contains the matrix **A**, and **FI** the **F** vector.

The FORTRAN listing of GHMATPQ is as follows.

```
C-----------------------------------------------------------------------
      SUBROUTINE GHMATPQ(X,Y,G,H,FI,DFI,KODE,NX,NX1)
C
C  PROGRAM 18
C
C  THIS SUBROUTINE COMPUTES THE G AND H MATRICES AND FORMS
C  THE SYSTEM OF EQUATIONS A X = F
C  H IS A SQUARE MATRIX (2*NE,2*NE); G IS RECTANGULAR (2*NE,3*NE)
C
      DIMENSION X(1),Y(1),G(NX,NX1),H(NX,NX)
      DIMENSION HW(3),GW(3),DQ1W(3),DQ2W(3),DU1W(3),DU2W(3)
      DIMENSION FI(1),DFI(1),KODE(1)
      COMMON N,L,INP,IPR
      NE=N/2
      DO 20 I=1,N
      DO 11 J=1,N
   11 H(I,J)=0.
      DO 12 J=1,3*NE
   12 G(I,J)=0.
   20 CONTINUE
      X(N+1)=X(1)
      Y(N+1)=Y(1)
C
C  COMPUTE THE GW AND HW MATRICES FOR EACH COLLOCATION
C  POINT AND EACH BOUNDARY ELEMENT
C
      DO 40 LL=1,N
      DO 40 I=1,N-1,2
      IF((LL-I)*(LL-I-1)*(LL-I-2)*(LL-I+N-2)) 22,21,22
   21 NODO=LL-I+1
      IF((LL.EQ.1).AND.(I.EQ.N-1)) NODO=NODO+N
      CALL EXTINPQ(X(LL),Y(LL),X(I),Y(I),X(I+1),Y(I+1),X(I+2),Y(I+2)
     *,HW,GW,DQ1W,DQ2W,DU1W,DU2W,0)
      CALL LOCINPQ(X(I),Y(I),X(I+1),Y(I+1),X(I+2),Y(I+2),GW,NODO)
      GO TO 34
   22 CALL EXTINPQ(X(LL),Y(LL),X(I),Y(I),X(I+1),Y(I+1),X(I+2),Y(I+2)
     *,HW,GW,DQ1W,DQ2W,DU1W,DU2W,0)
C
C  PLUG THE GW AND HW MATRICES INTO THE GENERAL G AND H MATRICES.
C
   34 DO 38 J=1,3
      K=3*(I-1)/2
      G(LL,K+J)=G(LL,K+J)+GW(J)
      IF(I-N+1) 37,35,37
   35 IF(J-3) 37,36,36
   36 H(LL,1)=H(LL,1)+HW(J)
      GO TO 38
   37 H(LL,I-1+J)=H(LL,I-1+J)+HW(J)
   38 CONTINUE
   40 CONTINUE
C
C  COMPUTE THE DIAGONAL COEFFICIENTS OF THE H MATRIX
C
      DO 70 I=1,N
      H(I,I)=0.
      DO 60 J=1,N
      IF(I.EQ.J) GO TO 60
      H(I,I)=H(I,I)-H(I,J)
   60 CONTINUE
C
C  ADD ONE TO THE DIAGONAL COEFFICIENTS FOR
C  EXTERNAL PROBLEMS.
C
      IF(H(I,I)) 65,70,70
   65 H(I,I)=6.2831852+H(I,I)
   70 CONTINUE
C
C  REORDER THE COLUMNS OF THE SYSTEM OF EQUATIONS IN ACCORDANCE
C  WITH THE BOUNDARY CONDITIONS AND FORM SYSTEM MATRIX A WHICH
C  IS STORED IN H
C
      DO 180 I=1,NE
      DO 170 J=1,3
      IF(KODE(3*I-3+J)) 110,110,170
  110 IF((I-NE).NE.0 .OR. J.NE.3) GO TO 125
      IF(KODE(1)) 115,115,113
  113 DO 114 K=1,N
      CH=H(K,1)
      H(K,1)=-G(K,3*I)
  114 G(K,3*I)=-CH
      GO TO 170
  115 DO 116 K=1,N
      H(K,1)=H(K,1)-G(K,3*I)
  116 G(K,3*I)=0.
      GO TO 170
  125 IF(I.EQ.1 .OR. J.GT.1 .OR. KODE(3*I-3).EQ.1) GO TO 130
      DO 129 K=1,N
      H(K,2*I-1)=H(K,2*I-1)-G(K,3*I-2)
  129 G(K,3*I-2)=0.
      GO TO 170
  130 DO 132 K=1,N
      CH=H(K,2*I-2+J)
      H(K,2*I-2+J)=-G(K,3*I-3+J)
  132 G(K,3*I-3+J)=-CH
  170 CONTINUE
  180 CONTINUE
C
C  FORM THE RIGHT HAND SIDE VECTOR F WHICH IS STORED IN FI
C
      DO 190 I=1,N
      FI(I)= 0.
```

```
    DO 185 J=1,3*NE
185 FI(I)=FI(I)+G(I,J)*DFI(J)
190 CONTINUE
    RETURN
    END
```

Routine EXTINPQ

This subroutine computes using numerical integration, the (3) submatrices **GW** and **HW** that correspond to an element when the collocation point is at a node other than any of those 3 in the element. The correlation of the collocation points are XP and YP. The integrals are of the type (2.76), i.e.

$$\mathbf{HW} = \int_{\Gamma_j} \phi \mathrm{q}^* \, d\Gamma = \int_{-1}^{+1} \phi \mathrm{q}^* |G| \, d\xi \tag{2.87}$$

$$\mathbf{GW} = \int_{\Gamma_j} \phi \mathrm{u}^* \, d\Gamma = \int_{-1}^{+1} \phi \mathrm{u}^* |G| \, d\xi \tag{2.88}$$

This subroutine also computes the (3) submatrices **DU1W**, **DU2W**, **DQ1W** and **DQ2W** which are needed to obtain the x_1 and x_2 fluxes at internal points

$$\mathbf{DU1W} = \int_{\Gamma_j} \phi \left(\frac{\partial u^*}{\partial x_1}\right)^i d\Gamma \tag{2.89}$$

$$\mathbf{DU2W} = \int_{\Gamma_j} \phi \left(\frac{\partial u^*}{\partial x_2}\right)^i d\Gamma \tag{2.90}$$

$$\mathbf{DQ1W} = \int_{\Gamma_j} \phi \left(\frac{\partial q^*}{\partial x_1}\right)^i d\Gamma \tag{2.91}$$

$$\mathbf{DQ2W} = \int_{\Gamma_j} \phi \left(\frac{\partial q^*}{\partial x_2}\right)^i d\Gamma \tag{2.92}$$

The Jacobians are calculated by taking derivatives of the expressions for the x_1 and x_2 coordinates, which are defined as follows (equation (2.80)),

$$\begin{aligned} x_1 &= \phi_1 x_1^1 + \phi_2 x_1^2 + \phi_3 x_1^3 \\ x_2 &= \phi_1 x_2^1 + \phi_2 x_2^2 + \phi_3 x_2^3 \end{aligned} \tag{2.93}$$

After substituting the ϕ_i expression (equation (2.74)) the above relationships can be written as,

$$\begin{aligned} x_1 &= \tfrac{1}{2}\xi^2(x_1^1 - 2x_1^2 + x_1^3) + \tfrac{1}{2}\xi(x_1^3 - x_1^1) + x_1^2 \\ x_2 &= \tfrac{1}{2}\xi^2(x_2^1 - 2x_2^2 + x_2^3) + \tfrac{1}{2}\xi(x_2^3 - x_2^1) + x_2^2 \end{aligned} \tag{2.94}$$

The Jacobian is obtained by substituting (2.94) into (2.78) which gives

$$|G| = [\{(x_1^3 - 2x_1^2 + x_1^1)\xi + \tfrac{1}{2}(x_1^3 - x_1^1)\}^2 + \{(x_2^3 - 2x_2^2 + x_2^1)\xi + \tfrac{1}{2}(x_2^3 - x_2^1)^2\}^2]^{1/2} \tag{2.95}$$

A Gauss quadrature formula with ten points has been taken instead of the four points formula applied in the constant and linear element cases. The reason is twofold: (i) The variation of potential and flux is quadratic and hence more points should be taken and (ii) the element geometry is also quadratic.

```
C-----------------------------------------------------------------------
      SUBROUTINE EXTINPQ(XP,YP,X1,Y1,X2,Y2,X3,Y3,HW,GW
     1,DQ1W,DQ2W,DU1W,DU2W,K)
C
C  PROGRAM 19
C
C  THIS SUBRUOTINE COMPUTES THE HW AND GW MATRICES
C  WHICH RELATE A NODE (XP,YP) WITH A BOUNDARY
C  ELEMENT USING GAUSS QUADRATURE
C  IT ALSO COMPUTES (WHEN K=1) THE DQ1W, DQ2W, DU1W AND DU2W MATRICES
C  WHICH RELATE AN INTERNAL POINT WITH A BOUNDARY ELEMENT AND ARE
C  NEEDED FOR COMPUTATION OF THE INTERNAL FLUX VALUES
C
C  RA            = RADIUS
C  RD1,RD2,RDN   = RADIUS DERIVATIVES
C  ETA1,ETA2     = COMPONENTS OF THE UNIT NORMAL TO THE ELEMENT
C  XCO,YCO       = INTEGRATION POINT ALONG THE ELEMENT
C  XJA           = JACOBIAN
C
      COMMON N,L,INP,IPR
      DIMENSION F(3),GW(3),HW(3),DQ1W(3),DQ2W(3),DU1W(3),DU2W(3)
      DIMENSION GI(10),OME(10)
      DATA GI/0.9739065285,-0.9739065285,0.8650633666,-0.8650633666
     *,0.6794095683,-0.6794095682,0.4333953941,-0.4333953941,
     *0.1488743389,-0.1488743389/
      DATA OME/0.0666713443,0.0666713443,0.1494513491,0.1494513491
     *,0.2190863625,0.2190863625,0.2692667193,0.2692667193,
     *0.2955242247,0.2955242247/
      DO 10 J=1,3
      HW(J)=0.
      GW(J)=0.
      DQ1W(J)=0.
      DQ2W(J)=0.
      DU1W(J)=0.
   10 DU2W(J)=0.
      A=X3-2*X2+X1
      B=(X3-X1)/2
      C=Y3-2*Y2+Y1
      D=(Y3-Y1)/2
      DO 40 I=1,10
C
C  COMPUTE THE VALUES OF THE SHAPE FUNCTIONS AT THE
C  INTEGRATION POINTS
C
      F(1)=GI(I)*(GI(I)-1)*0.5
      F(2)=1.-GI(I)**2
      F(3)=GI(I)*(GI(I)+1)*0.5
C
C  COMPUTE GEOMETRICAL PROPERTIES AT THE INTEGRATION POINTS
C
      XCO=X1*F(1)+X2*F(2)+X3*F(3)
      YCO=Y1*F(1)+Y2*F(2)+Y3*F(3)
      XJA=SQRT((GI(I)*A+B)**2+(GI(I)*C+D)**2)
      ETA1=(GI(I)*C+D)/XJA
      ETA2=-(GI(I)*A+B)/XJA
      RA=SQRT((XP-XCO)**2+(YP-YCO)**2)
      RD1=(XCO-XP)/RA
      RD2=(YCO-YP)/RA
      RDN=RD1*ETA1+RD2*ETA2
```

```
C
C  COMPUTE GW, HW DQ1W, DQ2W, DU1W AND DU2W MATRICES
C
      DO 40 J=1,3
      IF(K) 30,30,20
   20 DU1W(J)=DU1W(J)+RD1*OME(I)*XJA*F(J)/RA
      DU2W(J)=DU2W(J)+RD2*OME(I)*XJA*F(J)/RA
      DQ1W(J)=DQ1W(J)-((2.*RD1**2-1.)*ETA1+2.*RD1*RD2*ETA2)*
     1OME(I)*XJA*F(J)/RA**2
      DQ2W(J)=DQ2W(J)-((2.*RD2**2-1.)*ETA2+2.*RD1*RD2*ETA1)*
     1OME(I)*XJA*F(J)/RA**2
   30 GW(J)=GW(J)+ALOG(1./RA)*OME(I)*XJA*F(J)
   40 HW(J)=HW(J)-RDN/RA*OME(I)*XJA*F(J)
      RETURN
      END
```

Routine LOCINPQ

This subroutine computes using numerical integration, the submatrix **GW** that corresponds to an element when the collocation point is one of those three in the element. The integrals are

$$\mathbf{GW} = \int_{\Gamma_j} \phi \mathrm{u}^* \, d\Gamma \tag{2.96}$$

Three cases are considered depending on the position of the collocation point, i.e. NODE = 1, 2 or 3 (see figure 2.16).

(i) *Collocation point at Node (1)*

First a change of coordinates from $x_1 x_2$ to ξ is defined in the same way that was done in subroutine EXTINPQ. Then, in order to integrate the singularity a new change of variables is carried out, i.e.

$$\eta = \frac{\xi + 1}{2} \tag{2.97}$$

The integral then gives two parts, one with a singular term ln $1/\eta$ and the other with no singularity. The first part is integrated by means of a special integration formula of the type (see Appendix A)

$$I = \int_0^1 \ln\left(\frac{1}{\eta}\right) f(\eta) \, d\eta \cong \sum_{i=1}^{n} w_i f(\eta_i) \tag{2.98}$$

(in the program $\eta_i \equiv \mathrm{GIL}(I)$, $w_i \equiv \mathrm{OMEL}(I)$). The second part is integrated by the standard Gauss quadrature formula in terms of the variable ξ (in the program $\xi_i = \mathrm{GI}(I)$, $w_i = \mathrm{OME}(I)$). The shape functions ϕ_1, ϕ_2, ϕ_3 are given by F1, F2, F3 in terms of ξ and by FL1, FL2, FL3 in terms of η.

XJA1 is the Jacobian for the special integration and XJA2 is the Jacobian for the standard Gauss quadrature.

(ii) *Collocation point at Node (2)*

In order to integrate the two singularities that appear at both sides of the nodes, the integral is divided into two parts,

$$\mathbf{GW} = \int_{(1)}^{(3)} \phi u^*|G|\, d\xi = \int_{(1)}^{(2)} \phi u^*|G|\, d\xi + \int_{(2)}^{(3)} \phi u^*|G|\, d\xi \qquad (2.99)$$

Then, the first part is changed to the variable (see figure 2.16) $\eta' = -\xi$ and the second part ot the variable $\eta = \xi$. Now each singular part of these two integrals is computed using the special integration formula, and the two non-singular parts together are integrated using standard 10 points Gauss quadrature.

In the program XJA1 and XJA11 are the Jacobians for the two special logarthmic integrations and XJA2 is the Jacobian for the standard Gauss quadrature. The functions ϕ_1, ϕ_2 and ϕ_3 in terms of the new variables η are FLN1, FLN2 and FLN3.

(iii) *Collocation point at Node (3)*

This case is similar to the first one with the logarithmic integration variable being now

$$\eta = \frac{1-\xi}{2} \qquad (2.100)$$

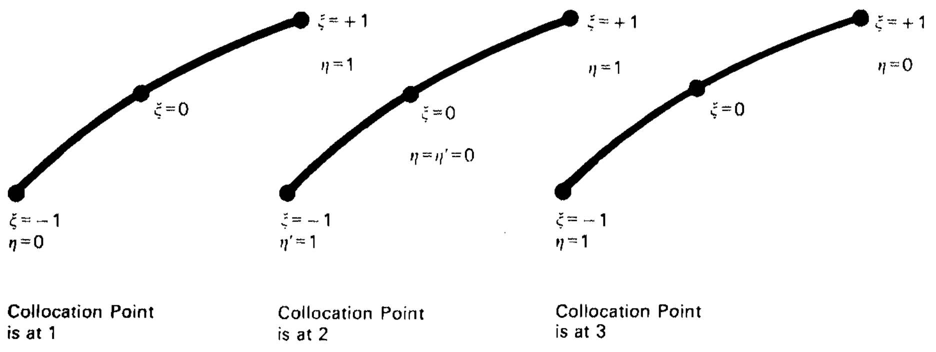


Figure 2.16 Geometrical coordinates systems for numerical integration

```
C-----------------------------------------------------------------------
      SUBROUTINE LOCINPQ(XG1,YG1,XG2,YG2,XG3,YG3,GW,NODO)
C
C  PROGRAM 20
C
C  THIS SUBROUTINE COMPUTES THE GW MATRIX WHEN THE COLLOCATION
C  POINT IS ONE OF THE NODES OF THE INTEGRATION ELEMENT.
C  THE COEFFICIENTS ARE COMPUTED BY NUMERICAL INTEGRATION:
```

```
C  THE NON SINGULAR PART IS COMPUTED USING STANDARD GAUSS QUADRATURE,
C  THE LOGARITHMIC PART IS COMPUTED USING A SPECIAL QUADRATURE FORMULA.
C
      COMMON N,L,INP,IPR
      DIMENSION GI(10),OME(10),GIL(10),OMEL(10),GW(3)
C
C  DATA FOR THE GAUSS QUADRATURE
C
      DATA GI/0.9739065285,-0.9739065285,0.8650633666,-0.8650633666
     @,0.6794095682,-0.6794095682,0.4333953941,-0.4333953941,
     @0.1488743389,-0.1488743389/
      DATA OME/0.0666713443,0.0666713443,0.1494513491,0.1494513491
     @,0.2190863625,0.2190863625,0.2692667193,0.2692667193,
     @0.2955242247,0.2955242247/
C
C  DATA FOR THE SPECIAL QUADRATURE
C
      DATA GIL/0.0090426309,0.0539712662,0.1353118246,0.2470524162
     @,0.3802125396,0.5237923179,0.6657752055,0.7941904160,
     @0.8981610912,0.9688479887/
      DATA OMEL/0.1209551319,0.1863635425,0.1956608732,0.1735771421
     @,0.1356956729,0.0936467585,0.0557877273,0.0271598109,
     @0.0095151826,0.0016381576/
C
C  SET A LOCAL COORDINATES SYSTEM
C
      GO TO(1,2,3),NODO
    1 X3=XG3-XG1
      Y3=YG3-YG1
      X2=XG2-XG1
      Y2=YG2-YG1
      A1=(X3-2*X2)*0.5
      B1=X2
      A2=(Y3-2*Y2)*0.5
      B2=Y2
      GO TO 4
    2 X3=XG3-XG2
      Y3=YG3-YG2
      X1=XG1-XG2
      Y1=YG1-YG2
      A1=X1+X3
      B1=X3-X1
      A2=Y1+Y3
      B2=Y3-Y1
      GO TO 4
    3 X2=XG2-XG3
      Y2=YG2-YG3
      X1=XG1-XG3
      Y1=YG1-YG3
      A1=(X1-2*X2)*0.5
      B1=-X2
      A2=(Y1-2*Y2)*0.5
      B2=-Y2
    4 CONTINUE
C
      DO 10 J=1,3
   10 GW(J)=0.
C
      DO 250 I=1,10
C
C  COMPUTE SHAPE FUNCTIONS FOR NUMERICAL INTEGRATIONS
C
      F3=0.5*GI(I)*(GI(I)+1.)
      F2=1.-GI(I)**2
      F1=0.5*GI(I)*(GI(I)-1.)
      FL3=GIL(I)*(2.*GIL(I)-1.)
      FL2=4.*GIL(I)*(1.-GIL(I))
      FL1=(GIL(I)-1.)*(2.*GIL(I)-1.)
      FLN3=0.5*GIL(I)*(GIL(I)+1.)
      FLN2=1.-GIL(I)**2
      FLN1=0.5*GIL(I)*(GIL(I)-1.)
```

```
C
C  COMPUTE GEOMETRICAL PROPERTIES
C
      GO TO(50,60,70) NODO
C
   50 XJA1=SQRT((4*A1*GIL(I)-2*A1+0.5*X3)**2+(4*A2*GIL(I)-2*A2+0.5*Y3)**
     @  2)*2
      XJA2=SQRT((A1*GI(I)*2+0.5*X3)**2+(A2*GI(I)*2+0.5*Y3)**2)
      XLO=-ALOG(2*SQRT((GI(I)*A1+B1)**2+(GI(I)*A2+B2)**2))
      S3=FL3*XJA1*OMEL(I)+F3*XJA2*XLO*OME(I)
      S2=FL2*XJA1*OMEL(I)+F2*XJA2*XLO*OME(I)
      S1=FL1*XJA1*OMEL(I)+F1*XJA2*XLO*OME(I)
      GO TO 200
   60 XJA1=SQRT((0.5*B1-A1*GIL(I))**2+(0.5*B2-A2*GIL(I))**2)
      XJA11=SQRT((0.5*B1+A1*GIL(I))**2+(0.5*B2+A2*GIL(I))**2)
      XJA2=SQRT((0.5*B1+A1*GI(I))**2+(0.5*B2+A2*GI(I))**2)
      XLO=-0.5*ALOG((GI(I)*A1*0.5+B1*0.5)**2+(GI(I)*A2*0.5+B2*0.5)**2)
      S3=(FLN1*XJA1+FLN3*XJA11)*OMEL(I)+F3*XJA2*XLO*OME(I)
      S2=FLN2*(XJA1+XJA11)*OMEL(I)+F2*XJA2*XLO*OME(I)
      S1=(FLN3*XJA1+FLN1*XJA11)*OMEL(I)+F1*XJA2*XLO*OME(I)
      GO TO 200
   70 XJA1=SQRT((2*A1-4*A1*GIL(I)-0.5*X1)**2+(2*A2-4*A2*GIL(I)-0.5*Y1)**
     @2)*2
      XJA2=SQRT((2*A1*GI(I)-0.5*X1)**2+(2*A2*GI(I)-0.5*Y1)**2)
      XLO=-ALOG(2*SQRT((A1*GI(I)+B1)**2+(A2*GI(I)+B2)**2))
      S3=FL1*XJA1*OMEL(I)+F3*XJA2*XLO*OME(I)
      S2=FL2*XJA1*OMEL(I)+F2*XJA2*XLO*OME(I)
      S1=FL3*XJA1*OMEL(I)+F1*XJA2*XLO*OME(I)
C
C  COMPUTE GW MATRIX
C
  200 GW(3)= GW(3)+S3
      GW(2)= GW(2)+S2
      GW(1)= GW(1)+S1
C
  250 CONTINUE
C
      RETURN
      END
```

Routine INTERPQ

This subroutine first reorders the vectors DFI and FI in such a way that all the boundary fluxes are stored in DFI and all the potentials in FI. It then computes the potentials and fluxes at internal points.

The potential at any interior point is given by

$$u^i = \sum_{j=0}^{NE} \left\{ \int_{\Gamma_j} u^* \phi \, \mathrm{d}\Gamma \right\} \mathbf{q}^j - \sum_{j=1}^{NE} \left\{ \int_{\Gamma_j} q^* \phi \, d\Gamma \right\} \mathbf{u}^j \tag{2.101}$$

The fluxes are given by

$$\begin{aligned} (q_{x1})^i &= \sum_{j=1}^{NE} \left\{ \int_{\Gamma_j} \left(\frac{\partial u^*}{\partial x_1} \right)^i \phi \, d\Gamma \right\} \mathbf{q}^j - \sum_{j=1}^{NE} \left\{ \int_{\Gamma_j} \left(\frac{\partial q^*}{\partial x_1} \right)^i \phi \, d\Gamma \right\} \mathbf{u}^j \\ (q_{x2})^i &= \sum_{j=1}^{NE} \left\{ \int_{\Gamma_j} \left(\frac{\partial u^*}{\partial x_2} \right)^i \phi \, d\Gamma \right\} \mathbf{q}^j - \sum_{j=1}^{NE} \left\{ \int_{\Gamma_j} \left(\frac{\partial q^*}{\partial x_2} \right)^i \phi \, d\Gamma \right\} \mathbf{u}^j \end{aligned} \tag{2.102}$$

where the integrals along the boundary elements are computed numerically by calling again the subroutine EXTINPQ.

The listing of INTERPQ is as follows:

```
C-----------------------------------------------------------------------
      SUBROUTINE INTERPQ(FI,DFI,KODE,CX,CY,X,Y,POT,FLUX1,FLUX2)
C
C  PROGRAM 21
C
C  THIS SUBROUTINE COMPUTES THE VALUES OF THE POTENTIAL AND THE FLUXES
C  AT INTERNAL POINTS.
C
      COMMON N,L,INP,IPR
      DIMENSION FI(1),DFI(1),KODE(1),CX(1),CY(1)
      DIMENSION X(1),Y(1),POT(1),FLUX1(1),FLUX2(1)
      DIMENSION HW(3),GW(3),DQ1W(3),DQ2W(3),DU1W(3),DU2W(3)
C
C  REARRANGE THE FI AND DFI ARRAYS TO STORE ALL THE VALUES OF THE
C  POTENTIAL IN FI AND ALL THE VALUES OF THE DERIVATIVE IN DFI
C
      NE=N/2
      DO 180 I=1,NE
      DO 170 J=1,3
      IF(KODE(3*I-3+J)) 110,110,170
  110 IF((I-NE).NE.0 .OR. J.NE.3) GO TO 125
      IF(KODE(1)) 114,114,113
  113 CH=FI(1)
      FI(1)=DFI(3*I)
      DFI(3*I)=CH
      GO TO 170
  114 DFI(3*I)=DFI(1)
      GO TO 170
  125 IF(I.EQ.1 .OR. J.GT.1 .OR. KODE(3*I-3).EQ.1) GO TO 130
      DFI(3*I-2)=DFI(3*I-3)
      GO TO 170
  130 CH=FI(2*I-2+J)
      FI(2*I-2+J)=DFI(3*I-3+J)
      DFI(3*I-3+J)=CH
  170 CONTINUE
  180 CONTINUE
C
C  COMPUTE THE VALUES OF THE POTENTIAL AND THE FLUXES AT
C  INTERNAL POINTS
C
      IF(L.EQ.0) GO TO 50
      DO 240 K=1,L
      POT(K)=0.
      FLUX1(K)=0.
      FLUX2(K)=0.
      DO 230 I=1,NE
      CALL EXTINPQ(CX(K),CY(K),X(2*I-1),Y(2*I-1),X(2*I),Y(2*I),X(2*I+1)
     1,Y(2*I+1),HW,GW,DQ1W,DQ2W,DU1W,DU2W,1)
      DO 220 J=1,3
      IJ2=2*I-2+J
      IF(IJ2.GT.(2*NE)) IJ2=J-2
      POT(K)=POT(K)+GW(J)*DFI(3*I-3+J)-HW(J)*FI(IJ2)
      FLUX1(K)=FLUX1(K)+DU1W(J)*DFI(3*I-3+J)-DQ1W(J)*FI(IJ2)
  220 FLUX2(K)=FLUX2(K)+DU2W(J)*DFI(3*I-3+J)-DQ2W(J)*FI(IJ2)
  230 CONTINUE
      POT(K)=POT(K)/(2.*3.1415926536)
      FLUX1(K)=FLUX1(K)/(2.*3.1415926536)
      FLUX2(K)=FLUX2(K)/(2.*3.1415926536)
  240 CONTINUE
   50 RETURN
      END
```

Routine OUTPTPQ

This subroutine prints the results in th following order.

(i) Potentials and fluxes at boundary nodes (fluxes 'before' and 'after' each node are printed. Mid-nodes always have the same flux at both sides).

(ii) Internal points potentials.

The listing is as follows:

```
C-----------------------------------------------------------------------
      SUBROUTINE OUTPTPQ(X,Y,FI,DFI,CX,CY,POT,FLUX1,FLUX2)
C
C  PROGRAM 22
C
C  THIS SUBROUTINE PRINTS THE VALUES OF THE POTENTIAL AND ITS NORMAL
C  DERIVATIVE AT BOUNDARY NODES. IT ALSO PRINTS THE VALUES OF THE
C  POTENTIAL AT INTERNAL POINTS
C
      COMMON N,L,INP,IPR
      DIMENSION X(1),Y(1),FI(1),DFI(1),CX(1),CY(1)
      DIMENSION POT(1),FLUX1(1),FLUX2(1)
C
      NE=N/2
      WRITE(IPR,100)
  100 FORMAT(' ',79('*')//2X,'RESULTS'//2X,'BOUNDARY NODES'//
     156X,'POTENTIAL DERIVATIVE'/
     29X,'X',15X,'Y',12X,'POTENTIAL',6X,'BEFORE NODE',6X,'AFTER NODE'/)
      WRITE(IPR,200) X(1),Y(1),FI(1),DFI(3*NE),DFI(1)
      WRITE(IPR,200) X(2),Y(2),FI(2),DFI(2),DFI(2)
      DO 10 I=2,NE
      WRITE(IPR,200) X(2*I-1),Y(2*I-1),FI(2*I-1),DFI(3*I-3),DFI(3*I-2)
   10 WRITE(IPR,200) X(2*I),Y(2*I),FI(2*I),DFI(3*I-1),DFI(3*I-1)
  200 FORMAT(5(2X,E14.5))
C
      IF(L.EQ.0) GO TO 30
      WRITE(IPR,300)
  300 FORMAT(//,2X,'INTERNAL POINTS',//9X,'X',15X,'Y',12X,'POTENTIAL',
     19X,'FLUX X',10X,'FLUX Y'/)
      DO 20 K=1,L
   20 WRITE(IPR,400)CX(K),CY(K),POT(K),FLUX1(K),FLUX2(K)
  400 FORMAT(5(2X,E14.5))
   30 WRITE(IPR,500)
  500 FORMAT(' ',79('*'))
      RETURN
      END
```

Example 2.4

The problem of an elliptical bar under torsion (figure 2.17(a)) is analysed using program POQUABE. Under Saint-Venant type torsion the displacements are given by

$$\begin{aligned} u_1 &= -\theta x_3 x_2 \\ u_2 &= \theta x_3 x_1 \\ u_3 &= \theta\phi \end{aligned} \tag{a}$$

where θ is the torsion angle per unit length and $\phi(x, y)$ is the warping function given by

$$\nabla^2\phi = 0 \tag{b}$$

The boundary conditions are as follows.

Tractions normal to the boundary are identically zero, hence

$$\frac{\partial\phi}{\partial n} = |r| \cos(\vec{r}, \vec{t}) \tag{c}$$

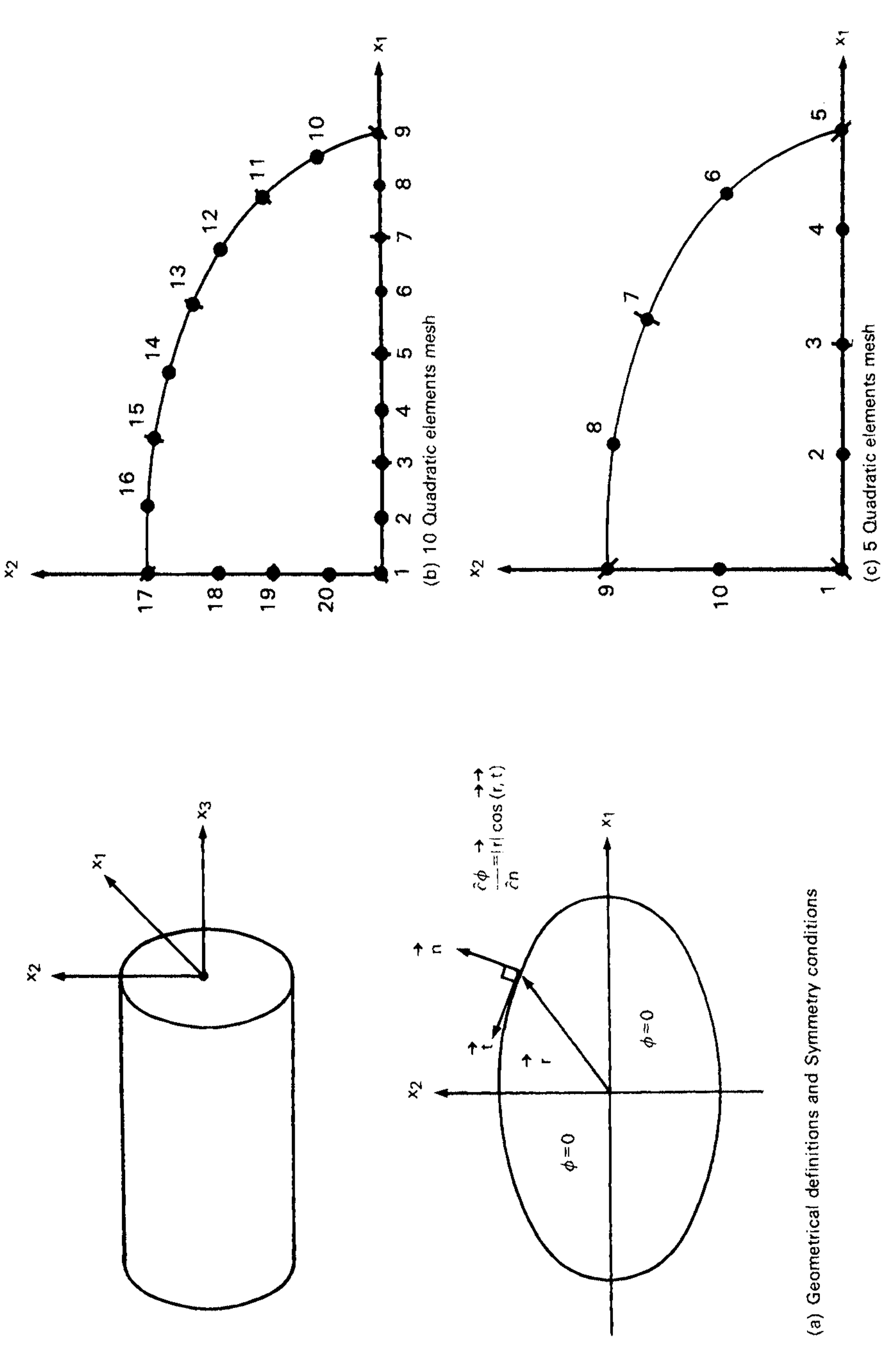

(a) Geometrical definitions and Symmetry conditions
x1
x2
x3
$\phi=0$
$\phi=0$
$\vec{r}$
$\vec{t}$
$\vec{n}$
$\frac{\partial\phi}{\partial n}=|\vec{r}|\cos(\vec{r},\vec{t})$
(b) 10 Quadratic elements mesh
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
(c) 5 Quadratic elements mesh
1 2 3 4 5 6 7 8 9 10

Figure 2.17 Torsion problem. Definition of the problem and boundary discretizations

For the case of an ellipse this becomes,

$$\frac{\partial\phi}{\partial n}=\frac{a^2-b^2}{\sqrt{a^4x_2^2+b^4x_1^2}}x_1x_2 \tag{d}$$

The dimensions were assumed to be $a = 10$ and $b = 5$ and values of ϕ on the boundary and at two selected internal points ($x_1 = 2, x_2 = 2$) and ($x_1 = 4, x_2 = 3.5$) were computed.

Because of symmetry $\phi \equiv 0$ along the two axes. Thus, only one quarter of the ellipse needs to be discretized. Ten quadratic elements were used here. Two for the short semi-axis, four for the long one and four for one quarter of the ellipse (see figure 2.17(b)).

The data for this case are as follows:

ELLIPTICAL SECTION (10) (DATA)

```
ELLIPTICAL SECTION UNDER TORSION (10 QUADRATIC ELEMENTS)
10  2
0. 0. 1.25 0. 2.5 0. 3.75 0. 5. 0. 6.25 0. 7.5 0. 8.75 0. 10. 0.
9.67 1.273  8.814  2.3617 7.7008 3.1898 6.174 3.933
4.7898  4.3891 3.3044 4.719 1.557 4.939
0. 5. 0. 3.375 0. 2.5  0. 1.25
0 0. 0 0. 0 0.
 0 0. 0 0. 0 0.
 0 0. 0 0. 0 0.
 0 0. 0 0. 0 0.
1 0. 1 -3.379  1 -4.8334
1 -4.8334 1 -4.9447  1 -4.3104
1 -4.3104 1 -3.4657  1 -2.4411
1 -2.4411 1 -1.1643  1 0.
0 0. 0 0. 0 0.
 0 0. 0 0. 0 0.
2. 2. 4. 3.5
```

and the output is given by

ELLIPTICAL SECTION (10) (OUTPUT)

```
******************************************************************************
ELLIPTICAL SECTION UNDER TORSION (10 QUADRATIC ELEMENTS)

 DATA
 NUMBER OF BOUNDARY ELEMENTS= 10
 NUMBER OF INTERNAL POINTS=  2

 BOUNDARY NODES COORDINATES

   NODE          X                   Y

     1      .0000000E+00        .0000000E+00
     2      .1250000E+01        .0000000E+00
     3      .2500000E+01        .0000000E+00
     4      .3750000E+01        .0000000E+00
     5      .5000000E+01        .0000000E+00
     6      .6250000E+01        .0000000E+00
```

```
      7        .7500000E+01        .0000000E+00
      8        .8750000E+01        .0000000E+00
      9        .1000000E+02        .0000000E+00
     10        .9670000E+01        .1273000E+01
     11        .8814000E+01        .2361700E+01
     12        .7700800E+01        .3189800E+01
     13        .6174000E+01        .3933000E+01
     14        .4789800E+01        .4389100E+01
     15        .3304400E+01        .4719000E+01
     16        .1557000E+01        .4939000E+01
     17        .0000000E+00        .5000000E+01
     18        .0000000E+00        .3375000E+01
     19        .0000000E+00        .2500000E+01
     20        .0000000E+00        .1250000E+01

 BOUNDARY CONDITIONS

          -----FIRST NODE------     -----SECOND NODE-----     -----THIRD NODE------
            PRESCRIBED                PRESCRIBED                PRESCRIBED
ELEMENT       VALUE        CODE         VALUE        CODE         VALUE        CODE
   1       .0000000E+00     0        .0000000E+00     0        .0000000E+00     0
   2       .0000000E+00     0        .0000000E+00     0        .0000000E+00     0
   3       .0000000E+00     0        .0000000E+00     0        .0000000E+00     0
   4       .0000000E+00     0        .0000000E+00     0        .0000000E+00     0
   5       .0000000E+00     1       -.3379000E+01     1       -.4833400E+01     1
   6      -.4833400E+01     1       -.4944700E+01     1       -.4310400E+01     1
   7      -.4310400E+01     1       -.3465700E+01     1       -.2441100E+01     1
   8      -.2441100E+01     1       -.1164300E+01     1        .0000000E+00     1
   9       .0000000E+00     0        .0000000E+00     0        .0000000E+00     0
  10       .0000000E+00     0        .0000000E+00     0        .0000000E+00     0
*********************************************************************************

 RESULTS

 BOUNDARY NODES

                                                        POTENTIAL DERIVATIVE
       X               Y            POTENTIAL       BEFORE NODE      AFTER NODE

   .00000E+00      .00000E+00      .00000E+00      -.28405E-03     -.28405E-03
   .12500E+01      .00000E+00      .00000E+00       .74965E+00      .74965E+00
   .25000E+01      .00000E+00      .00000E+00       .14996E+01      .14996E+01
   .37500E+01      .00000E+00      .00000E+00       .22502E+01      .22502E+01
   .50000E+01      .00000E+00      .00000E+00       .29999E+01      .29999E+01
   .62500E+01      .00000E+00      .00000E+00       .37536E+01      .37536E+01
   .75000E+01      .00000E+00      .00000E+00       .44914E+01      .44914E+01
   .87500E+01      .00000E+00      .00000E+00       .53030E+01      .53030E+01
   .10000E+02      .00000E+00      .00000E+00       .58638E+01      .00000E+00
   .96700E+01      .12730E+01     -.74619E+01      -.33790E+01     -.33790E+01
   .88140E+01      .23617E+01     -.12506E+02      -.48334E+01     -.48334E+01
   .77008E+01      .31898E+01     -.14746E+02      -.49447E+01     -.49447E+01
   .61740E+01      .39330E+01     -.14576E+02      -.43104E+01     -.43104E+01
   .47898E+01      .43891E+01     -.12616E+02      -.34657E+01     -.34657E+01
   .33044E+01      .47190E+01     -.93634E+01      -.24411E+01     -.24411E+01
   .15570E+01      .49390E+01     -.46001E+01      -.11643E+01     -.11643E+01
   .00000E+00      .50000E+01      .00000E+00       .00000E+00      .30154E+01
   .00000E+00      .33750E+01      .00000E+00       .20220E+01      .20220E+01
   .00000E+00      .25000E+01      .00000E+00       .14999E+01      .14999E+01
   .00000E+00      .12500E+01      .00000E+00       .74936E+00      .74936E+00

 INTERNAL POINTS

       X               Y            POTENTIAL         FLUX X           FLUX Y

   .20000E+01      .20000E+01     -.23990E+01      -.12001E+01     -.11990E+01
   .40000E+01      .35000E+01     -.84019E+01      -.21012E+01     -.24031E+01
*********************************************************************************
```

Results for some representative boundary nodes and the two internal points are compared with the known exact solution in the following table.

Boundary node	Potential using 10 quadratic elements	Exact potential solution
$\left.\begin{array}{l}x_1 = 8.814\\x_2 = 2.361\end{array}\right\}$	-12.506	-12.489
$\left.\begin{array}{l}x_1 = 6.174\\x_2 = 3.933\end{array}\right\}$	-14.576	-14.570
$\left.\begin{array}{l}x_1 = 3.304\\x_2 = 4.719\end{array}\right\}$	-9.363	-9.356
Internal points		
$\left.\begin{array}{l}x_1 = 2.\\x_2 = 2.\end{array}\right\}$	-2.399	-2.400
$\left.\begin{array}{l}x_1 = 4\\x_2 = 3.5\end{array}\right\}$	-8.402	-8.400

Example 2.5

This example is the solution of the same elliptical sections as described in figure 2.16(a)) and studied in Example 2.4 but the number of quadratic elements has been reduced to 5. It is interesting to see how the simple model gives results which are in good agreement with the theory.

The input for the five element model (figure 2.17(c)) is as follows:

ELLIPTICAL SECTION (5) (DATA)

```
ELLIPTICAL SECTION UNDER TORSION (5 QUADRATIC ELEMENTS)

5  2
0. 0.  2.5 0. 5. 0. 7.5 0. 10. 0.
  8.814  2.3617  6.174 3.933
 3.3044 4.719
0. 5.  0. 2.5
 0 0. 0 0. 0 0.
 0 0. 0 0. 0 0.
1 0.  1 -4.8334  1 -4.3104
1 -4.3104   1 -2.4411  1 0.
 0 0. 0 0. 0 0.
2. 2. 4. 3.5
```

The corresponding output is given below.

ELLIPTICAL SECTION (5) (OUTPUT)

```
******************************************************************************
ELLIPTICAL SECTION UNDER TORSION (5 QUADRATIC ELEMENTS)

 DATA
 NUMBER OF BOUNDARY ELEMENTS=  5
 NUMBER OF INTERNAL POINTS=  2

 BOUNDARY NODES COORDINATES

   NODE          X                  Y

     1       .0000000E+00       .0000000E+00
     2       .2500000E+01       .0000000E+00
     3       .5000000E+01       .0000000E+00
     4       .7500000E+01       .0000000E+00
     5       .1000000E+02       .0000000E+00
     6       .8814000E+01       .2361700E+01
     7       .6174000E+01       .3933000E+01
     8       .3304400E+01       .4719000E+01
     9       .0000000E+00       .5000000E+01
    10       .0000000E+00       .2500000E+01

 BOUNDARY CONDITIONS

          -----FIRST NODE------     -----SECOND NODE-----     -----THIRD NODE------
            PRESCRIBED                PRESCRIBED                PRESCRIBED
ELEMENT        VALUE        CODE        VALUE        CODE        VALUE        CODE
   1       .0000000E+00      0      .0000000E+00      0      .0000000E+00      0
   2       .0000000E+00      0      .0000000E+00      0      .0000000E+00      0
   3       .0000000E+00      1     -.4833400E+01      1     -.4310400E+01      1
   4      -.4310400E+01      1     -.2441100E+01      1      .0000000E+00      1
   5       .0000000E+00      0      .0000000E+00      0      .0000000E+00      0
******************************************************************************
 RESULTS

 BOUNDARY NODES

                                                        POTENTIAL DERIVATIVE
        X               Y             POTENTIAL      BEFORE NODE      AFTER NODE

    .00000E+00      .00000E+00      .00000E+00      -.10388E-01      -.10388E-01
    .25000E+01      .00000E+00      .00000E+00       .15215E+01       .15215E+01
    .50000E+01      .00000E+00      .00000E+00       .29757E+01       .29757E+01
    .75000E+01      .00000E+00      .00000E+00       .46612E+01       .46612E+01
    .10000E+02      .00000E+00      .00000E+00       .51625E+01       .00000E+00
    .88140E+01      .23617E+01     -.12779E+02      -.48334E+01      -.48334E+01
    .61740E+01      .39330E+01     -.14839E+02      -.43104E+01      -.43104E+01
    .33044E+01      .47190E+01     -.94350E+01      -.24411E+01      -.24411E+01
    .00000E+00      .50000E+01      .00000E+00       .00000E+00       .30004E+01
    .00000E+00      .25000E+01      .00000E+00       .15277E+01       .15277E+01

 INTERNAL POINTS

        X               Y             POTENTIAL         FLUX X           FLUX Y

    .20000E+01      .20000E+01     -.24305E+01      -.12130E+01      -.12163E+01
    .40000E+01      .35000E+01     -.84718E+01      -.21111E+01      -.23590E+01

******************************************************************************
```

The following table compares the 5 element solution with the exact solution.

Boundary node	Potential using 5 quadratic elements	Exact potential solution
$x_1 = 8.814$, $x_2 = 2.361$	-12.779	-12.489
$x_1 = 6.174$, $x_2 = 3.933$	-14.839	-14.570
$x_1 = 3.304$, $x_2 = 4.719$	-9.435	-9.356
Internal points		
$x_1 = 2.$, $x_2 = 2.$	-2.431	-2.400
$x_1 = 4$, $x_2 = 3.5$	-8.472	-8.400

As an exercise, the reader can run the same problem using programs POCONBE and POLINBE for constant and linear elements and compare results with those shown in the above table and in Example 2.4.

2.10 Computer Code for Multiboundary Problems (POMCOBE)

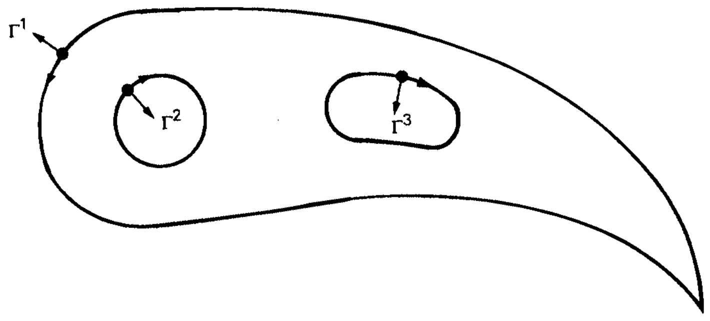
In engineering practice many problems have more than one surface as shown in figure 2.18, with internal and external boundaries. Both types of boundary can be differentiated by identifying the direction of the normals. This can easily be done in two dimensional problems by adopting the rule that the numbering on the external surface is done counterclockwise and the one on the internal surface is carried out in the clockwise direction. From these rules the normal will be well defined in the computer code.

The following computer code is based on the constant element code (POCONBE) of section 4, but while the previous code was applicable to problems with only one surface the following listing applies for multisurface cases.

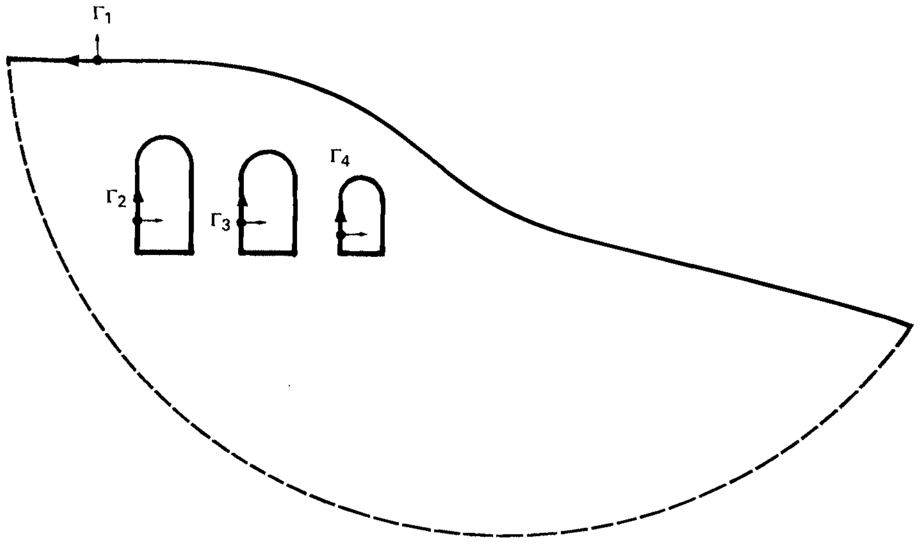
Main Program

Similar to program 1 already described in POCONBE, but now the Common statement is replaced by

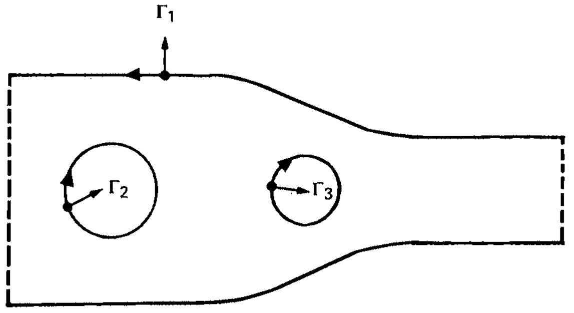
```
COMMON N,L,NC(5),M,LEC,IMP
```



(a) Turbine blade with cooling holes



(b) Set of tunnels



(c) Potential flow around several obstacles

Figure 2.18 Problems with more than one surface

where M defines the number of different surfaces and NC stores the last node of each different surface. The dimensions of NC allow for 5 different surfaces in the present listing but this can easily be extended by the user if required.

The listing is as follows:

```
C
C-------------------------------------------------------------------
C
      PROGRAM POMCOBE
C
C  PROGRAM 23
C
C
C  THIS PROGRAM SOLVES TWO DIMENSIONAL (PO)TENTIAL PROBLEMS
C  WITH (M)ULTIBOUNDARY DOMAINS USING (CO)NSTANT (B)OUNDARY (E)LEMENTS.
C
C
      CHARACTER*10 FILEIN,FILEOUT
C
      DIMENSION X(101),Y(101),XM(100),YM(100),FI(100),DFI(100)
      DIMENSION KODE(100),CX(20),CY(20),POT(20),FLUX1(20),FLUX2(20)
C
      COMMON/MATG/ G(100,100)
      COMMON/MATH/ H(100,100)
      COMMON       N,L,NC(5),M,INP,IPR
C
C     SET MAXIMUN DIMENSION OF THE SYSTEM OF EQUATIONS (NX)
C     (THIS NUMBER MUST BE EQUAL OR SMALLER THAN THE DIMENSION OF XM, ETC...)
C
      NX=100
C
C  ASSIGN NUMBERS FOR INPUT AND OUTPUT FILES
C
      INP=5
      IPR=6
C
C  READ NAMES AND OPEN  FILES FOR INPUT AND OUTPUT
C
      WRITE(*,' (A) ') ' NAME OF INPUT FILE (MAX. 10 CHART.)'
      READ(*,'  (A) ')FILEIN
      OPEN(INP,FILE=FILEIN,STATUS='OLD')
      WRITE(*,' (A) ') ' NAME OF OUTPUT FILE (MAX. 10 CHART.)'
      READ(*,' (A) ')FILEOUT
      OPEN(IPR,FILE=FILEOUT,STATUS='NEW')
C
C  READ DATA
C
      CALL INPUMPC(CX,CY,X,Y,KODE,FI)
C
C  COMPUTE H AND G MATRICES AND FORM SYSTEM (A X = F)
C
      CALL GHMAMPC(X,Y,XM,YM,G,H,FI,DFI,KODE,NX)
C
C  SOLVE SYSTEM OF EQUATIONS
C
      CALL SLNPD(G,DFI,D,N,NX)
C
C  COMPUTE  POTENTIAL VALUES AT INTERNAL POINTS
C
      CALL INTEMPC(FI,DFI,KODE,CX,CY,X,Y,POT,FLUX1,FLUX2)
C
C  PRINT RESULTS AT BOUNDARY NODES AND INTERNAL POINTS
C
      CALL OUTPMPC(XM,YM,FI,DFI,CX,CY,POT,FLUX1,FLUX2)
C
C   CLOSE INPUT AND OUTPUT FILES
C
    CLOSE (INP)
    CLOSE (IPR)
    STOP
    END
```

Routine INPUMPC

The input required is the same as in Program 2 of POCONBE with the exception of M and NC which are read in the same line as N and L (i.e. number of elements and number of internal points where the function *u* is required).

The listing is as follows:

```
C-----------------------------------------------------------------------
      SUBROUTINE INPUMPC(CX,CY,X,Y,KODE,FI)
C
C  PROGRAM 24
C
      CHARACTER*80 TITLE
      COMMON N,L,NC(5),M,INP,IPR
      DIMENSION CX(1),CY(1),X(1),Y(1),KODE(1),FI(1)
C
C  N= NUMBER OF BOUNDARY NODES (=NUMBER OF ELEMENTS)
C  L= NUMBER OF INTERNAL POINTS WHERE THE FUNCTION IS CALCULATED
C
      WRITE(IPR,100)
  100 FORMAT(' ',79('*'))
C
C  READ JOB TITLE
C
      READ(INP,'(A)') TITLE
      WRITE(IPR,'(A)') TITLE
C
C  READ NUMBER OF BOUNDARY ELEMENTS,NUMBER OF INTERNAL POINTS,
C  NUMBER OF DIFFERENT BOUNDARIES AND LAST NODE OF EACH BOUNDARY
C
      READ(INP,*)N,L,M,(NC(K),K=1,M)
      WRITE(IPR,300)N,L
  300 FORMAT(//' DATA'//2X,'NUMBER OF BOUNDARY ELEMENTS =',I3/2X,'NUMBER
     1 OF INTERNAL POINTS WHERE THE FUNCTION IS CALCULATED =',I3)
C
      IF(M)40,40,30
   30 WRITE(IPR,999)M,(NC(K),K=1,M)
  999 FORMAT(/2X,'NUMBER OF DIFFERENT BOUNDARIES =',I3/2X,
     1'LAST NODE OF EACH BOUNDARY =',5(I3,','))
C
C
C  READ COORDINATES OF EXTREME POINTS OF THE BOUNDARY ELEMENTS
C  IN ARRAYS X AND Y
C
   40 WRITE(IPR,500)
  500 FORMAT(//2X,'COORDINATES OF THE EXTREME POINTS OF THE BOUNDARY ELE
     1MENTS',//1X,'POINT',7X,'X',15X,'Y')
      READ(INP,*) (X(I),Y(I),I=1,N)
      DO 10 I=1,N
   10 WRITE(IPR,700)I,X(I),Y(I)
  700 FORMAT(2X,I3,2(2X,E14.5))
C
C  READ BOUNDARY CONDITIONS IN FI(I) VECTOR, IF KODE(I)=0 THE FI(I)
C  VALUE IS A KNOWN POTENTIAL;IF KODE(I)=1 THE FI(I) VALUE IS A
C  KNOWN POTENTIAL DERIVATIVE (FLUX).
C
      WRITE(IPR,800)
  800 FORMAT(//2X,'BOUNDARY CONDITIONS'//2X,'NODE',6X,'CODE',7X,'PRESCRI
     1BED VALUE')
      DO 20 I=1,N
      READ(INP,*) KODE(I),FI(I)
   20 WRITE(IPR,950)I,KODE(I),FI(I)
  950 FORMAT(2X,I3,9X,I1,8X,E14.5)
C
C  READ COORDINATES OF THE INTERNAL POINTS
C
      IF(L.EQ.0) GO TO 50
      READ(INP,*) (CX(I),CY(I),I=1,L)
   50 RETURN
      END
```

Routine GHMAMPC

In this routine the COMMON needs to be changed as in the MAIN program. In addition some extra commands have been included to differentiate the points on each of the surfaces. These are required in order to compute the mid-point coordinates XM and YM. Each surface has to close and the last node of each surface is in the mid-point between the last extreme point to the first point on that surface.

The listing of GHMAMPC is now as follows.

```
C-----------------------------------------------------------------------
      SUBROUTINE GHMAMPC(X,Y,XM,YM,G,H,FI,DFI,KODE,NX)
C
C  PROGRAM 25
C
C  THIS SUBROUTINE COMPUTES THE G AND H MATRICES
C  AND FORMS THE SYSTEM OF EQUATIONS A X = F
C
      COMMON N,L,NC(5),M,INP,IPR
      DIMENSION X(1),Y(1),XM(1),YM(1),FI(1),KODE(1)
      DIMENSION DFI(1),G(NX,NX),H(NX,NX)
C
C  COMPUTE THE NODAL COORDINATES AND STORE IN ARRAYS XM AND YM
C
      X(N+1)=X(1)
      Y(N+1)=Y(1)
      DO 10 I=1,N
      XM(I)=(X(I)+X(I+1))/2
   10 YM(I)=(Y(I)+Y(I+1))/2
      IF(M-1)15,15,12
   12 XM(NC(1))=(X(NC(1))+X(1))/2
      YM(NC(1))=(Y(NC(1))+Y(1))/2
      DO 13 K=2,M
      XM(NC(K))=(X(NC(K))+X(NC(K-1)+1))/2
   13 YM(NC(K))=(Y(NC(K))+Y(NC(K-1)+1))/2
C
C  COMPUTE THE COEFFICIENTS OF G AND H MATRICES
C
   15 DO 30 I=1,N
      DO 30 J=1,N
      IF(M-1)16,16,17
   17 IF(J-NC(1))19,18,19
   18 KK=1
      GO TO 23
   19 DO 22 K=2,M
      IF(J-NC(K))22,21,22
   21 KK=NC(K-1)+1
      GO TO 23
   22 CONTINUE
   16 KK=J+1
   23 IF(I-J)20,25,20
   20 CALL EXTINPC(XM(I),YM(I),X(J),Y(J),X(KK),Y(KK),H(I,J),G(I,J)
     1,DQ1,DQ2,DU1,DU2,0)
      GO TO 30
   25 CALL LOCINPC(X(J),Y(J),X(KK),Y(KK),G(I,J))
      H(I,J)=3.1415926
   30 CONTINUE
C
C  REORDER THE COLUMNS OF THE SYSTEM OF EQUATIONS IN ACCORDANCE
C  WITH THE BOUNDARY CONDITIONS AND FORM SYSTEM MATRIX A WHICH
C  IS STORED IN G
C
      DO 55 J=1,N
      IF(KODE(J))55,55,40
   40 DO 50 I=1,N
      CH=G(I,J)
      G(I,J)=-H(I,J)
      H(I,J)=-CH
   50 CONTINUE

   55 CONTINUE
C
C  FORM THE RIGHT HAND SIDE VECTOR F WHICH IS STORED IN DFI
C
```

```
      DO 60 I=1,N
      DFI(I)=0.
      DO 60 J=1,N
      DFI(I)=DFI(I)+H(I,J)*FI(J)
   60 CONTINUE
      RETURN
      END
```

Routine EXTINPC

As in POCONBE (Program 4).

Routine LOCINPC

As in POCONBE (Program 5).

Routine SLNPD

As in POCONBE (Program 6).

Routine INTEMPC

This routine varies from Program 7 by a few statements to take into account the different surfaces.

```
C---------------------------------------------------------------------
      SUBROUTINE INTEMPC(FI,DFI,KODE,CX,CY,X,Y,POT,FLUX1,FLUX2)
C
C  PROGRAM 26
C
C  THIS SUBROUTINE COMPUTES THE VALUES OF THE POTENTIAL
C  AND THE POTENTIAL DERIVATIVES (FLUXES) AT INTERNAL POINTS
C
      COMMON N,L,NC(5),M,INP,IPR
      DIMENSION FI(1),DFI(1),KODE(1),CX(1),CY(1),X(1),Y(1)
      DIMENSION POT(1),FLUX1(1),FLUX2(1)
C
C  REARRANGE THE FI AND DFI ARRAYS TO STORE ALL THE VALUES OF THE
C  POTENTIAL IN FI AND ALL THE VALUES OF THE DERIVATIVE IN DFI
C
      DO 20 I=1,N
      IF(KODE(I)) 20,20,10
   10 CH=FI(I)
      FI(I)=DFI(I)
      DFI(I)=CH
   20 CONTINUE
C
C  COMPUTE THE POTENTIAL AND THE FLUXES AT INTERNAL POINTS
C
      IF(L.EQ.0) GO TO 50
      DO 40 K=1,L
      POT(K)=0.
      FLUX1(K)=0.
      FLUX2(K)=0.
      DO 30 J=1,N
      IF(M-1)28,28,22
   22 IF(J-NC(1))24,23,24
   23 KK=1
      GO TO 29
```

```
 24 DO 26 LK=2,M
    IF(J-NC(LK))26,25,26
 25 KK=NC(LK-1)+1
    GO TO 29
 26 CONTINUE
 28 KK=J+1
 29 CALL EXTINPC(CX(K),CY(K),X(J),Y(J),X(KK),Y(KK),A,B
   1,DQ1,DQ2,DU1,DU2,1)
    POT(K)=POT(K)+DFI(J)*B-FI(J)*A
    FLUX1(K)=FLUX1(K)+DFI(J)*DU1-FI(J)*DQ1
 30 FLUX2(K)=FLUX2(K)+DFI(J)*DU2-FI(J)*DQ2
    POT(K)=POT(K)/(2.*3.1415926)
    FLUX1(K)= FLUX1(K)/(2.*3.1415926)
 40 FLUX2(K)= FLUX2(K)/(2.*3.1415926)
 50 RETURN
    END
```

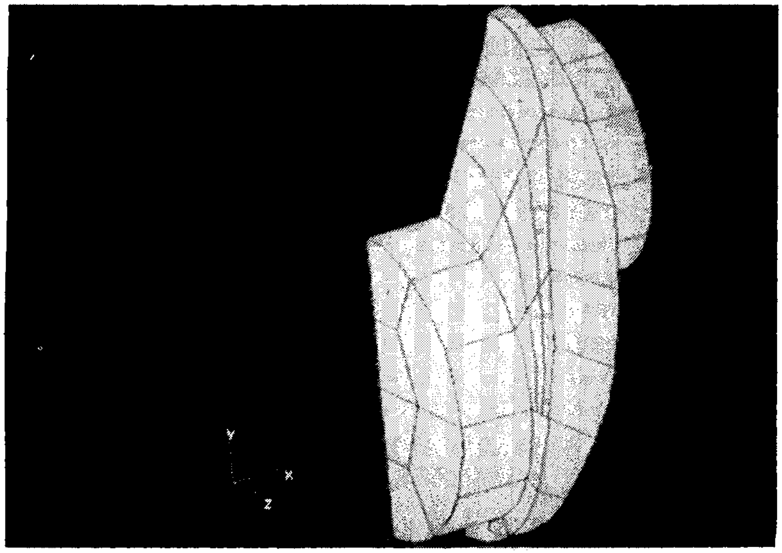
Routine OUTPMPC

Same as Program 8 but with the new COMMON.

```
C-----------------------------------------------------------------------
      SUBROUTINE OUTPMPC(XM,YM,FI,DFI,CX,CY,POT,FLUX1,FLUX2)
C
C  PROGRAM 27
C
C  THIS SUBROUTINE PRINTS THE VALUES OF THE POTENTIAL AND ITS NORMAL
C  DERIVATIVE AT BOUNDARY NODES. IT ALSO PRINTS THE VALUES OF THE
C  POTENTIAL AND THE FLUXES AT INTERNAL POINTS
C
      COMMON N,L,NC(5),M,INP,IPR
      DIMENSION XM(1),YM(1),FI(1),DFI(1),CX(1),CY(1)
      DIMENSION POT(1),FLUX1(1),FLUX2(1)
C
      WRITE(IPR,100)
  100 FORMAT(' ',79('*')//1X,'RESULTS'//2X,'BOUNDARY NODES'//8X,'X',15
     1X,'Y',13X,'POTENTIAL',3X,'POTENTIAL DERIVATIVE'/)
      DO 10 I=1,N
   10 WRITE(IPR,200) XM(I),YM(I),FI(I),DFI(I)
  200 FORMAT(4(2X,E14.5))
C
      IF(L.EQ.0) GO TO 30
      WRITE(IPR,300)
  300 FORMAT(//,2X,'INTERNAL POINTS',//8X,'X',15X,'Y',13X,'POTENTIAL',
     19X,'FLUX X',10X,'FLUX Y'/)
      DO 20 K=1,L
   20 WRITE(IPR,400)CX(K),CY(K),POT(K),FLUX1(K),FLUX2(K)
  400 FORMAT(5(2X,E14.5))
   30 WRITE(IPR,500)
  500 FORMAT(' ',79('*'))
      RETURN
      END
```

2.11 Boundary Elements for Three Dimensional Problems

The elements used in three dimensional problems are surface elements which cover the boundary of the body (figure 2.19). They are usually of two types; triangular or quadrilateral and both can be flat or curved. The functions u and q and those used to describe the geometry can be constant over the element, vary linearly, being second order functions and others which produce a curved element. While the development of constant or flat elements is comparatively simple, curved elements are more important in three dimensions as they can follow better the geometry of actual engineering components and hence they will be described in detail here.



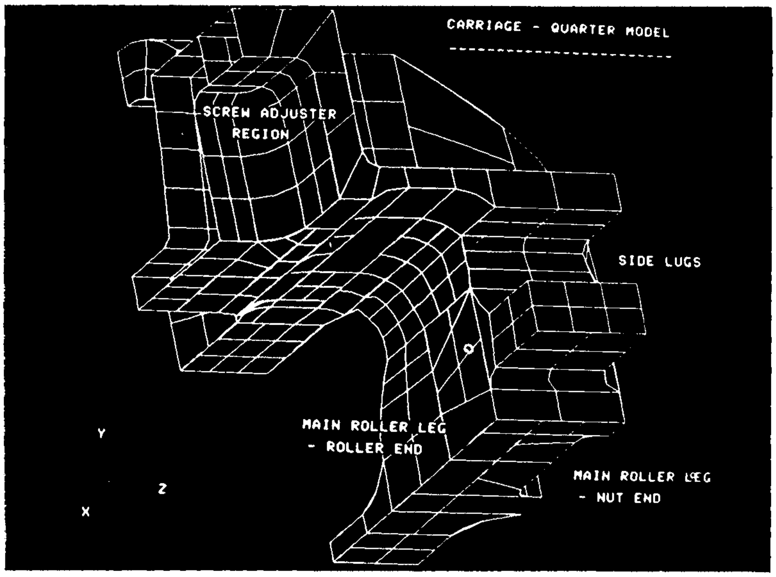


Figure 2.19 Some three dimensional applications (lower figure by courtesy of British Aerospace plc)

To study curved elements first we need to define the way in which we can pass from the $x_1x_2x_3$ global cartesian system to the $\xi_1\xi_2$, η system defined over the element, where $\xi_1\xi_2$ are oblique coordinates and η is in the direction of the normal (see figure 2.20).

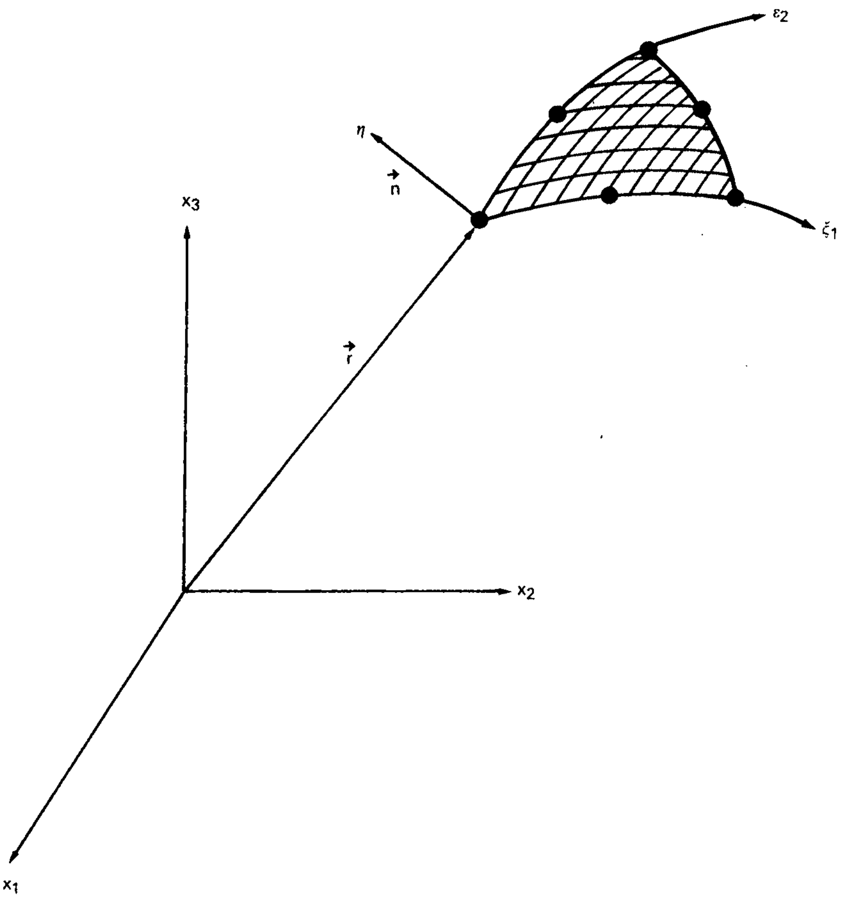


Figure 2.20 a) Triangular curved elements for three dimensional problems

The transformation for a given function – say u – is related through the following,

$$\begin{Bmatrix} \dfrac{\partial u}{\partial \xi_1} \\[2mm] \dfrac{\partial u}{\partial \xi_2} \\[2mm] \dfrac{\partial u}{\partial \eta} \end{Bmatrix} = \begin{bmatrix} \dfrac{\partial x_1}{\partial \xi_1} & \dfrac{\partial x_2}{\partial \xi_1} & \dfrac{\partial x_3}{\partial \xi_1} \\[2mm] \dfrac{\partial x_1}{\partial \xi_2} & \dfrac{\partial x_2}{\partial \xi_2} & \dfrac{\partial x_3}{\partial \xi_2} \\[2mm] \dfrac{\partial x_1}{\partial \eta} & \dfrac{\partial x_2}{\partial \eta} & \dfrac{\partial x_3}{\partial \eta} \end{bmatrix} \begin{Bmatrix} \dfrac{\partial u}{\partial x_1} \\[2mm] \dfrac{\partial u}{\partial x_2} \\[2mm] \dfrac{\partial u}{\partial x_3} \end{Bmatrix} \tag{2.103}$$

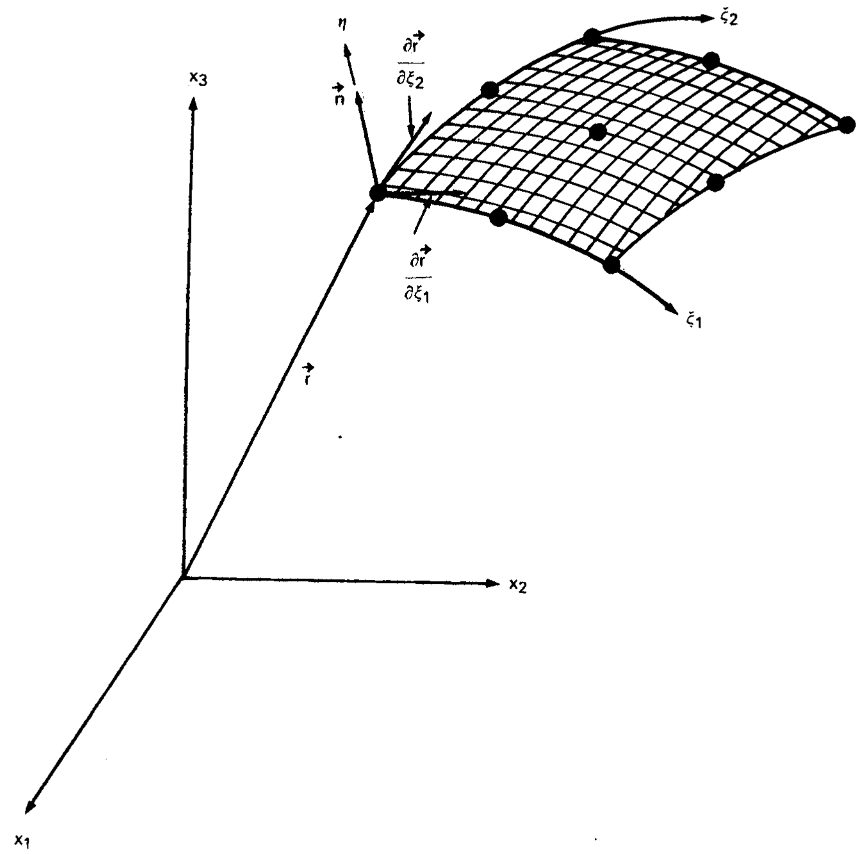

x_3
η
$\vec{n}$
$\frac{\partial \vec{r}}{\partial \xi_2}$
ξ_2
$\frac{\partial \vec{r}}{\partial \xi_1}$
ξ_1
$\vec{r}$
x_2
x_1

Figure 2.20 b) Quadrilateral curved elements for three dimensional problems

where the square matrix is the Jacobian or **J**. Hence

$$\begin{Bmatrix} \dfrac{\partial u}{\partial \xi_1} \\ \dfrac{\partial u}{\partial \xi_2} \\ \dfrac{\partial u}{\partial \eta} \end{Bmatrix} = \mathbf{J} \begin{Bmatrix} \dfrac{\partial u}{\partial x_1} \\ \dfrac{\partial u}{\partial x_2} \\ \dfrac{\partial u}{\partial x_3} \end{Bmatrix} \tag{2.104}$$

The inverse relationship is then given by

$$\begin{Bmatrix} \dfrac{\partial u}{\partial x_1} \\ \dfrac{\partial u}{\partial x_2} \\ \dfrac{\partial u}{\partial x_3} \end{Bmatrix} = \mathbf{J}^{-1} \begin{Bmatrix} \dfrac{\partial u}{\partial \xi_1} \\ \dfrac{\partial u}{\partial \xi_2} \\ \dfrac{\partial u}{\partial \eta} \end{Bmatrix} \tag{2.105}$$

Transformations of this type allow us to describe differentials of volume or surface in the cartesian system in terms of the curvilinear coordinates. For instance a differential of volume Ω can be written as,

$$\begin{aligned} d\Omega &= \text{Magnitude}\left(\frac{\partial \vec{r}}{\partial \xi_1} \times \frac{\partial \vec{r}}{\partial \xi_2} \cdot \frac{\partial \vec{r}}{\partial \eta}\right) d\xi_1\, d\xi_2\, d\eta \\ &= |\mathbf{J}|\, d\xi_1\, d\xi_2\, d\eta \end{aligned} \tag{2.106}$$

A differential of area instead will be given by

$$d\Gamma = \left|\frac{\partial \vec{r}}{\partial \xi_1} \times \frac{\partial \vec{r}}{\partial \xi_2}\right| d\xi_1\, d\xi_2 = |\mathbf{G}|\, d\xi_1\, d\xi_2 \tag{2.107}$$

where $\mathbf{G}$ is a reduced Jacobian and $|\mathbf{G}|$ is simply the magnitude of the normal vector $\vec{\eta}$; i.e.

$$\vec{\eta} = \frac{\partial \vec{r}}{\partial \xi_1} \times \frac{\partial \vec{r}}{\partial \xi_2} = (g_1, g_2, g_3) = \left(\frac{\partial x_1}{\partial \eta}, \frac{\partial x_2}{\partial \eta}, \frac{\partial x_3}{\partial \eta}\right) \tag{2.108}$$

where

$$\frac{\partial \vec{r}}{\partial \xi_1} = \left(\frac{\partial x_1}{\partial \xi_1}, \frac{\partial x_2}{\partial \xi_1}, \frac{\partial x_3}{\partial \xi_1}\right); \qquad \frac{\partial \vec{r}}{\partial \xi_2} = \left(\frac{\partial x_1}{\partial \xi_2}, \frac{\partial x_2}{\partial \xi_2}, \frac{\partial x_3}{\partial \xi_2}\right)$$

Notice that the values of g_1, g_2 and g_3 are given by

$$\begin{aligned} g_1 &= \left(\frac{\partial x_2}{\partial \xi_1}\frac{\partial x_3}{\partial \xi_2} - \frac{\partial x_2}{\partial \xi_2}\frac{\partial x_3}{\partial \xi_1}\right) \\ g_2 &= \left(\frac{\partial x_3}{\partial \xi_1}\frac{\partial x_1}{\partial \xi_2} - \frac{\partial x_1}{\partial \xi_1}\frac{\partial x_3}{\partial \xi_2}\right) \\ g_3 &= \left(\frac{\partial x_1}{\partial \xi_1}\frac{\partial x_2}{\partial \xi_2} - \frac{\partial x_2}{\partial \xi_1}\frac{\partial x_1}{\partial \xi_2}\right) \end{aligned} \tag{2.109}$$

Hence the magnitude of $|\mathbf{G}|$ is given by

$$|\mathbf{G}| = \sqrt{(g_1^2 + g_2^2 + g_3^2)} \tag{2.110}$$

These relationships can be used to integrate any of the boundary integrals terms such as,

$$\int_{\Gamma} u^* q \, d\Gamma \qquad \text{or} \qquad \int_{\Gamma} u q^* \, d\Gamma \tag{2.111}$$

which now become

$$\int_{\Gamma_\xi} u^* q |\mathbf{G}| \, d\xi_1 \, d\xi_2; \qquad \int_{\Gamma_\xi} u q^* |\mathbf{G}| \, d\xi_1 \, d\xi_2 \tag{2.112}$$

A whole range of quadrilateral and triangular elements can be defined. For quadrilateral cases one prefers to use Lagrangian type of elements (i.e. the ones which have in some cases nodes inside) as such elements give better numerical results when used with point collocation. This is simply because the collocation points are better distributed in these cases. Table 2.1 shows some of the elements which can be used.

2.12 Poisson's Equation

Many practical applications are governed by the Poisson rather than Laplace equations. In these cases sources are distributed in the Ω domain in accordance with a $b(\mathbf{x})$ function. This produces the following governing equation

$$\nabla^2 u = b \qquad \text{in } \Omega \tag{2.113}$$

where b is a known function of position.

Sometimes the above equation can be reduced to Laplace's simply by substituting a particular solution or a change of variables. Care should be taken in these cases also to transform the boundary conditions accordingly.

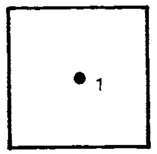
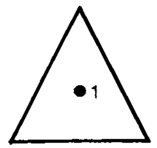
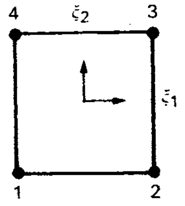
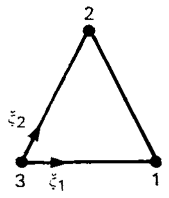
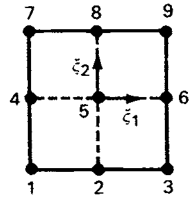
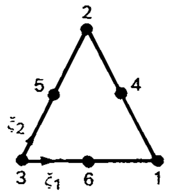
When the function $b(\mathbf{x})$ has a more complex formulation it may be difficult to find any suitable transformation and one needs to start with (2.113) plus the appropriate boundary conditions in order to deduce the basic integral equation, i.e.

$$\int_{\Omega} (\nabla^2 u - b) u^* \, d\Omega = \int_{\Gamma_2} (q - \bar{q}) u^* \, d\Gamma - \int_{\Gamma_1} (u - \bar{u}) q^* \, d\Gamma \tag{2.114}$$

which integrated by parts twice produces,

$$\int_{\Omega} (\nabla^2 u^*) u \, d\Omega - \int_{\Omega} b u^* \, d\Omega = -\int_{\Gamma_2} \bar{q} u^* \, d\Gamma - \int_{\Gamma_1} q u^* \, d\Gamma + \int_{\Gamma_2} u q^* \, d\Gamma + \int_{\Gamma_1} \bar{u} q^* \, d\Gamma \tag{2.115}$$

Table 2.1 Table of Triangular and Quadrilateral Surface Elements for 3D Problems

	Quadrilateral	Triangular
Zero Order	*Constant*  *Interpolation Functions* $\phi_1 = 1$	*Constant*  *Interpolation Functions* $\phi_1 = 1$
First Order	*Bilinear*  *Interpolation Functions* $\phi_1 = \frac{1}{4}(1-\xi_1)(1-\xi_2)$; $\phi_2 = \frac{1}{4}(1+\xi_1)(1-\xi_2)$ $\phi_3 = \frac{1}{4}(1+\xi_1)(1+\xi_2)$; $\phi_1 = \frac{1}{4}(1-\xi_1)(1+\xi_2)$	*Linear*  *Interpolation Functions* $\phi_1 = \xi_1$; $\phi_2 = \xi_2$; $\phi_3 = 1 - \xi_1 - \xi_2 = \xi_3$ where $\xi_3 = 1 - \xi_1 - \xi_2$
Second Order	*Biquadratic*  7 8 9 ξ_2 4 5 6 ξ_1 1 2 3 *Interpolation Functions* $\phi_1 = \frac{1}{4}\xi_1(\xi_1-1)\xi_2(\xi_2-1)$; $\phi_2 = \frac{1}{2}(1-\xi_1^2)\xi_2(\xi_2-1)$; $\phi_3 = \frac{1}{4}\xi_1(1+\xi_1)\xi_2(\xi_2-1)$ $\phi_4 = \frac{1}{2}\xi_1(\xi_1-1)(1-\xi_2^2)$; $\phi_5 = (1-\xi_1^2)(1-\xi_2^2)$; $\phi_6 = \frac{1}{2}\xi_1(1+\xi_1)(1-\xi_2^2)$ $\phi_7 = \frac{1}{4}\xi_1(\xi_1-1)\xi_2(1+\xi_2)$; $\phi_8 = \frac{1}{2}(1-\xi_1^2)\xi_2(1+\xi_2)$; $\phi_9 = \frac{1}{4}\xi_1(1+\xi_1)\xi_2(1+\xi_2)$	*Quadratic*  *Interpolation Functions* $\phi_1 = \xi_1(2\xi_1-1)$; $\phi_2 = \xi_2(2\xi_2-1)$; $\phi_3 = \xi_3(2\xi_3-1)$ $\phi_4 = 4\xi_1\xi_2$; $\phi_5 = 4\xi_2\xi_3$; $\phi_6 = 4\xi_3\xi_1$

After substituting the Laplace fundamental solution u^* and grouping all boundary terms together (i.e. in $\Gamma = \Gamma_1 + \Gamma_2$), one obtains

$$c^i u^i + \int_\Gamma u q^* \, d\Gamma + \int_\Omega b u^* \, d\Omega = \int_\Gamma q u^* \, d\Gamma \tag{2.116}$$

Notice that although the b functions are known and consequently the integrals in Ω do not introduce any new unknowns, the problem has changed in character as we need now to carry out integrals in the domain as well as on the boundary.

Regions of integration called cells can now be employed to compute the domain integral in (2.116) (figure 2.21). One usually applies a numerical integration scheme such as Gauss. In this case for each position of the singularity at a boundary point i, the integral in (2.116) can be written as,

$$D^i = \int_\Omega b u^* \, d\Omega = \sum_{e=1}^{M} \left(\sum_{k=1}^{r} w_k (b u^*)_k \right) \Omega_e \tag{2.117}$$

where e are the different cells (e varies from 1 to M, where M is the total number of cells describing Ω domain), w_k are the integration weights, the function (bu^*) needs to be evaluated at r integration points on each cell, Ω_e is the area of the cell 'e'. D^i is the result, different for each 'i' position of the fundamental solution, where i is one of the boundary nodes.

Hence equation (2.116) now becomes

$$c^i u^i + \sum_{j=1}^{N} \hat{H}^{ij} u^j + D^i = \sum_{j=1}^{N} G^{ij} q^j \tag{2.118}$$

or in matrix form

$$\mathbf{HU} + \mathbf{D} = \mathbf{GQ} \tag{2.119}$$

Notice that the domain integrals need to be computed as well when calculating any values of potentials or fluxes at internal points. Hence,

$$u^i = \sum_{j=1}^{N} G^{ij} q^j - \sum_{j=1}^{N} \hat{H}^{ij} u^j - D^i \tag{2.120}$$

where i is now an internal point, at which the singularity is applied.

Concentrated sources are very simple to handle in boundary elements. They are a special case for which the function b at the internal point 'l' becomes,

$$b = Q^l \Delta^l \tag{2.121}$$

where Q^l is the magnitude of the source and Δ^l is a Dirac delta function whose integral is equal to 1 at the point l and zero elsewhere. Assuming that a number of these functions exist one can write,

$$c^i u^i + \int_\Gamma uq^* \, d\Gamma + \int_\Omega bu^* \, d\Omega + \sum_{l=1}^{P} (Q^l u^{*l}) = \int_\Gamma qu^* \, d\Gamma \tag{2.122}$$

u^{*l} is the value of the fundamental solution at the point l. P are the number of concentrated forces within the domain.

Another way of dealing with volume sources is to transform the domain integral into equivalent boundary integrals. This is possible when the function b is harmonic (although the approach can in principle be extended to harmonic, or bi-harmonic, etc.). Harmonic means that b obeys the following equation,

$$\nabla^2 b = 0 \tag{2.123}$$

To effect the transformation – which is basically an integration by parts – we need to express the fundamental solution u^* in function of some derivatives. This is done by proposing a function v^* such that,

$$u^* = \nabla^2 v^* \tag{2.124}$$

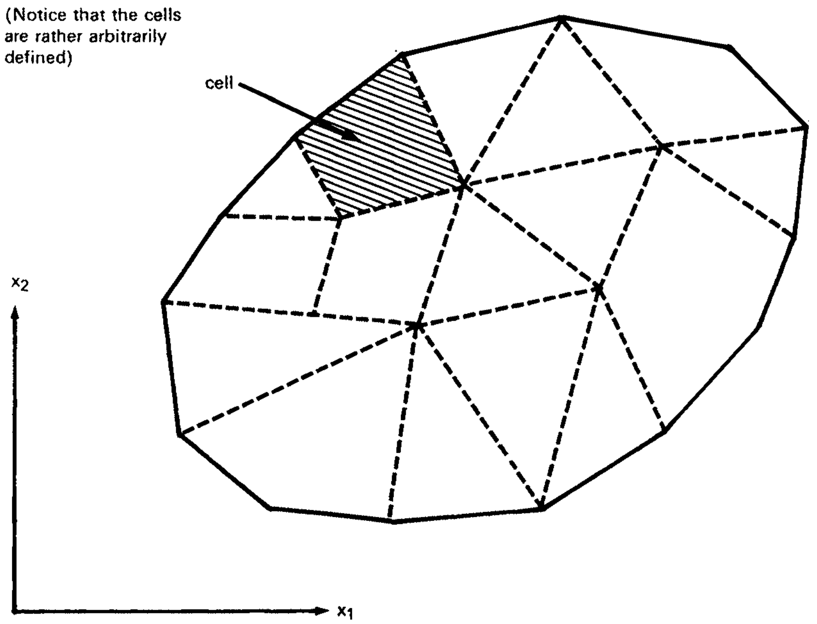


Figure 2.21 Boundary elements and internal cells

and then writing Green's identity, in the form

$$\int_\Omega (b\nabla^2 v^* - v^*\nabla^2 b)\, d\Omega = \int_\Gamma \left(b\frac{\partial v^*}{\partial n} - v^* \frac{\partial b}{\partial n}\right) d\Gamma \tag{2.125}$$

which if $\nabla^2 b \equiv 0$ reduces to,

$$\int_\Omega b u^*\, d\Omega = \int_\Gamma \left(b\frac{\partial v^*}{\partial n} - v^* \frac{\partial b}{\partial n}\right) d\Gamma \tag{2.126}$$

Hence we have effectively reduced the domain integrals to two different boundary integrals.

The function v^* required in this case is simply the fundamental solution of the biharmonic equation as,

$$\nabla^2 u^* = \nabla^2(\nabla^2 v^*) = \nabla^4 v^* = -\Delta^i \tag{2.127}$$

which for two dimensions is a well known fundamental solution used in plate bending, i.e.

$$v^* = \frac{r^2}{8\pi}\left[\ln\left(\frac{1}{r}\right) + 1\right] \tag{2.128}$$

and for three dimensions

$$v^* = \frac{1}{8\pi} r \tag{2.129}$$

Notice that for two dimensions v^* satisfies Laplace's equation as follows,

$$\nabla^2 v^* = \frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial v^*}{\partial r}\right) = \frac{1}{2\pi}\ln\frac{1}{r} = u^* \tag{2.130}$$

and for three dimensions one finds that,

$$\nabla^2 v^* = \frac{1}{r^2}\frac{\partial}{\partial r}\left(r^2\frac{\partial v^*}{\partial r}\right) = \frac{1}{4\pi r} = u^* \tag{2.131}$$

Example 2.6 Results for the Poisson Equation Using Internal Cells

The equation $\nabla^2 u = -2$ was solved for the geometry shown in figure 2.22 with the homogeneous boundary condition $\bar{u} = 0$ and using linear elements.

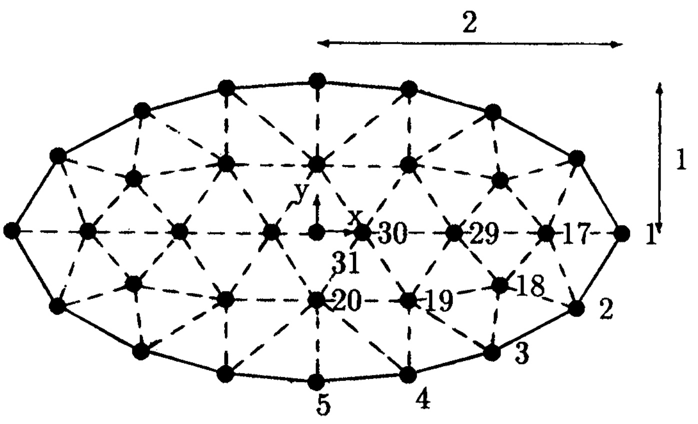


Figure 2.22 Elliptical section discretized with Boundary Elements and internal cells

Notice that the domain has been discretized with a total of 48 internal cells. Results presented in table A below demonstrate that the solution is accurate.

Table A Cell results for Poisson Problem

Node	X	Y	Cell	Exact
17	1.5	0.0	0.331	0.350
18	1.2	−0.35	0.401	0.414
19	0.6	−0.45	0.557	0.566
20	0.0	−0.45	0.629	0.638
29	0.9	0.0	0.626	0.638
30	0.3	0.0	0.772	0.782
31	0.0	0.0	0.791	0.800

The reader can verify these results by using program POLINBE, after having added an appropriate subroutine to compute the vectors **D** as described in the previous section.

2.13 Orthotropy and Anisotropy

Up to now we have considered problems with isotropic materials, i.e. those for which the properties are the same in all directions. We will study those materials for which the properties vary in different directions. If one finds the directions of orthotropy (figure 2.23) the governing equation for a two dimensional problem

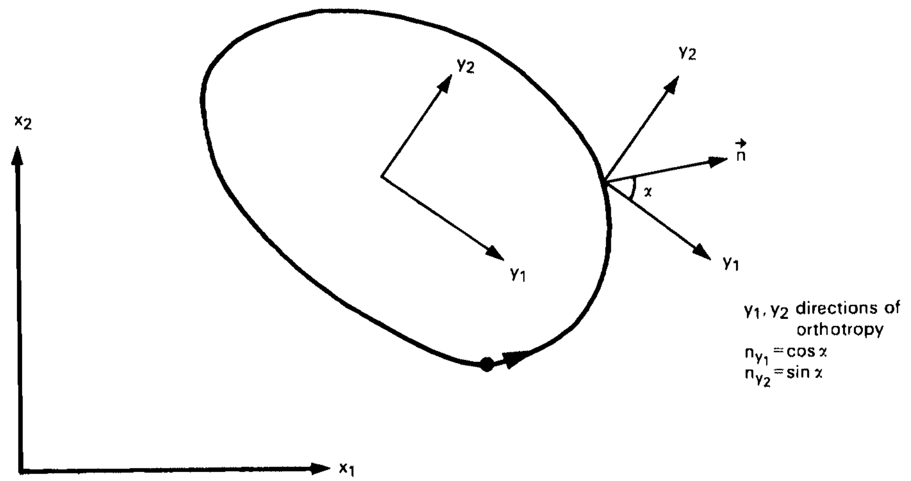


Figure 2.23 Orthotropic medium

can be written,

$$k_1 \frac{\partial^2 u}{\partial y_1^2} + k_2 \frac{\partial^2 u}{\partial y_2^2} = 0 \tag{2.132}$$

k_i is the medium property coefficient in the direction of orthotropy i. For three dimensional cases the equation is

$$k_1 \frac{\partial^2 u}{\partial y_1^2} + k_2 \frac{\partial^2 u}{\partial y_2^2} + k_3 \frac{\partial^2 u}{\partial y_3^2} = 0 \tag{2.133}$$

The fundamental solution corresponding to the above equations can be found by transforming the y_j system of coordinates into a z_j system in which the governing equations (2.132) or (2.133) become a Laplace equation without the k_j coefficients. This is achieved by using the following transformation,

$$z_j = \frac{y_j}{\sqrt{k_j}} \tag{2.134}$$

The fundamental solution of the Laplace equation in the z_j system is known and can then be backtransformed to the x_j coordinates. This gives for the two dimensional case the following solution:

$$u^* = \frac{1}{\sqrt{k_1 k_2}} \ln \frac{1}{r} \tag{2.135}$$

where r is now

$$r=\left\{\frac{1}{k_1}[y_1^i-y_1]^2+\frac{1}{k_2}[y_2^i-y_2]^2\right\}^{1/2} \tag{2.136}$$

where y_1, y_2 are the coordinates of the point under consideration and $y_1^i y_2^i$ are those of the node on which the fundamental solution is applied.

The corresponding normal flux is

$$q^*=k_1\frac{\partial u^*}{\partial y_1}n_{y1}+k_2\frac{\partial u^*}{\partial y_2}n_{y2} \tag{2.137}$$

where n_{y1} and n_{y2} are the direction cosines of the normal n to the surface with respect to y_1 and y_2 respectively. Similarly for the actual flux we have

$$q=k_1\frac{\partial u}{\partial y_1}n_{y1}+k_2\frac{\partial u}{\partial y_2}n_{y2} \tag{2.138}$$

For a fully anisotropic media, governing equation (2.132) becomes,

$$k_{11}\frac{\partial^2 u}{\partial x_1^2}+2k_{12}\frac{\partial^2 u}{\partial x_1 \partial x_2}+k_{22}\frac{\partial^2 u}{\partial x_2^2}=0 \tag{2.139}$$

where the k_{ij} coefficients represent the terms of the properties tensor. The fundamental solution can now be expressed

$$u^*=\frac{1}{2\pi\sqrt{|k_{ij}|}}\ln\frac{1}{r} \tag{2.140}$$

where $|k_{ij}|$ is the determinant of the medium conductivity and s is the inverse of the **k** matrix, ie the resistivity matrix

$$|k_{ij}|=k_{11}k_{22}-k_{12}^2;\ \mathbf{s}=\mathbf{k}^{-1}=\frac{1}{|k_{ij}|}\begin{bmatrix} k_{22} & -k_{12} \\ -k_{12} & k_{11}\end{bmatrix} \tag{2.141}$$

The r distance in equation (2.140) is now

$$r=\{s_{11}[x_1^i-x_i]^2+2s_{12}[x_1^i-x_1][x_2^i-x_2]+s_{22}[x_2^i-x_2]^2\}^{1/2} \tag{2.142}$$

The normal flux is

$$q^*=\left(k_{11}\frac{\partial u^*}{\partial x_1}+k_{12}\frac{\partial u^*}{\partial x_2}\right)n_{x1}+\left(k_{12}\frac{\partial u^*}{\partial x_1}+k_{22}\frac{\partial u^*}{\partial x_2}\right)n_{x2} \tag{2.143}$$

and similarly for q.

The fundamental solution for the three dimensional orthotropic case is

$$u^*=\frac{1}{2\pi\sqrt{k_1k_2k_3}}\left(\frac{1}{4\pi\ r}\right) \tag{2.144}$$

where r is now

$$r = \left\{ \frac{1}{k_1}[y_1^i - y_1]^2 + \frac{1}{k_2}[y_2^i - y_2]^2 + \frac{1}{k_3}[y_3^3 - y_3]^2 \right\} \tag{2.145}$$

Similar considerations as for two dimensional cases apply for the fluxes q^* and q.

2.14 Subregions

In certain cases the region under study may be piecewise homogeneous and then the boundary element procedure can be applied to each subregion in turn as if they were independent of each other. The final set of equations for the whole region can then be obtained by assembling the set of equations for each subregion using compatibility of potentials and fluxes between the common interfaces.

Consider for instance the two subregions shown in figure 2.24 one called Ω^1 and the other Ω^2. Over subregion Ω^1 we define,

$\mathrm{U}^1, \mathrm{Q}^1$;	Nodal potential and fluxes at the external boundary Γ^1
$\mathrm{U}_\mathrm{I}^1, \mathrm{Q}_\mathrm{I}^1$;	Nodal potential and fluxes at the interface Γ_I considering it belongs to Ω_1

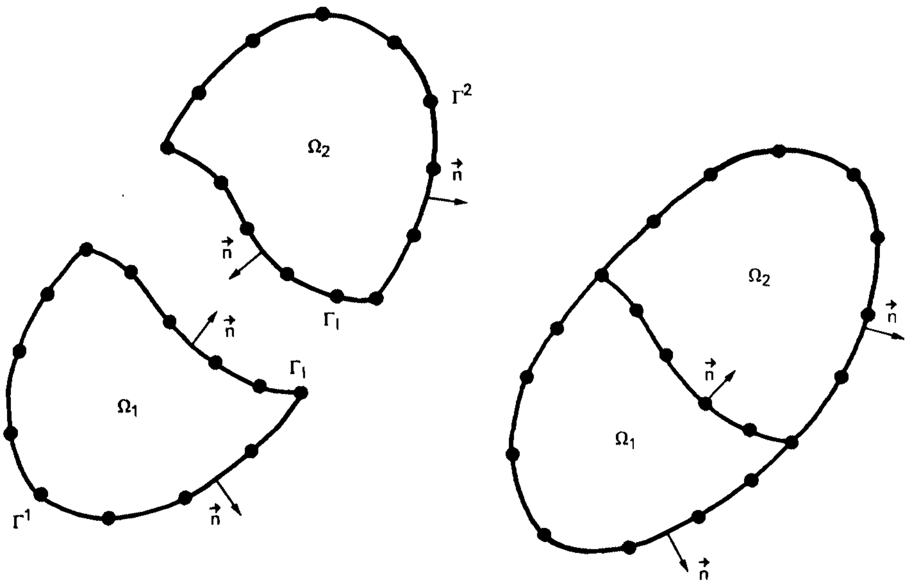


Figure 2.24 Piecewise homogeneous body consisting of two subregions

Over the other subregion, Ω_2, one can define,

$\mathbf{U}^2, \mathbf{Q}^2$; Nodal potential and fluxes at the external boundary Γ^2

$\mathbf{U}_I^2, \mathbf{Q}_I^2$; Nodal potential and fluxes at the interface Γ_I considering it belongs to Ω_2.

The system of equations corresponding to Ω_1 can be written as

$$[\mathbf{H}^1\mathbf{H}_I^1]\begin{Bmatrix}\mathbf{U}^1\\ \mathbf{U}_I^1\end{Bmatrix}=[\mathbf{G}^1\mathbf{G}_I^1]\begin{Bmatrix}\mathbf{Q}^1\\ \mathbf{Q}_I^1\end{Bmatrix}\tag{2.146}$$

and the one for subregion Ω_2 gives

$$[\mathbf{H}^2\mathbf{H}_I^2]\begin{Bmatrix}\mathbf{U}^2\\ \mathbf{U}_I^2\end{Bmatrix}=[\mathbf{G}^2\mathbf{G}_I^2]\begin{Bmatrix}\mathbf{Q}^2\\ \mathbf{Q}_I^2\end{Bmatrix}\tag{2.147}$$

The compatibility and equilibrium conditions on the interface Γ_I can be expressed as

$$\mathbf{U}_I^1=\mathbf{U}_I^2\tag{2.148}$$

and

$$\mathbf{Q}_I^1+\mathbf{Q}_I^2=\mathbf{0}\tag{2.149}$$

If one calls the potentials at the interface $\mathbf{U}_I$ and adapts on the same interface the fluxes of Ω_1 as reference values (which is equivalent to say that the normal on the interface is the normal to Ω_1) one has,

$$\mathbf{U}_I=\mathbf{U}_I^1=\mathbf{U}_I^2\tag{2.150}$$

and

$$\mathbf{Q}_I=\mathbf{Q}_I^1=-\mathbf{Q}_I^2\tag{2.151}$$

These conditions can be introduced in equations (2.146) and (2.147) which can now be written together as follows,

$$\begin{bmatrix}\mathbf{H}^1 & \mathbf{H}_I^1 & \mathbf{0}\\ \mathbf{0} & \mathbf{H}_I^2 & \mathbf{H}^2\end{bmatrix}\begin{Bmatrix}\mathbf{U}^1\\ \mathbf{U}_I\\ \mathbf{U}^2\end{Bmatrix}=\begin{bmatrix}\mathbf{G}^1 & \mathbf{G}_I^1 & \mathbf{0}\\ \mathbf{0} & -\mathbf{G}_I^2 & \mathbf{G}^2\end{bmatrix}\begin{Bmatrix}\mathbf{Q}^1\\ \mathbf{Q}_I\\ \mathbf{Q}^2\end{Bmatrix}\tag{2.152}$$

As $\mathbf{U}_I$ and $\mathbf{Q}_I$ are unknown at the interface the above system is frequently written as

$$\begin{bmatrix}\mathbf{H}^1 & \mathbf{H}_I^1 & -\mathbf{G}_I^1 & \mathbf{0}\\ \mathbf{0} & \mathbf{H}_I^2 & \mathbf{G}_I^2 & \mathbf{H}^2\end{bmatrix}\begin{pmatrix}\mathbf{U}^1\\ \mathbf{U}_I\\ \mathbf{Q}_I\\ \mathbf{U}^2\end{pmatrix}=\begin{bmatrix}\mathbf{G}^1 & \mathbf{0}\\ \mathbf{0} & \mathbf{G}^2\end{bmatrix}\begin{Bmatrix}\mathbf{Q}^1\\ \mathbf{Q}^2\end{Bmatrix}\tag{2.153}$$

Notice that we still have to apply the relevant boundary conditions on the external surface of the region, i.e. Γ^1 and Γ^2. It should also be noticed that the matrix on the left hand side of equation (2.153) is square and that the one on the right hand side is not.

It is interesting to point out that when more interfaces are included the system of equations may tend to be banded or have a large number of zero submatrices, which can result in an A matrix which is computationally more efficient. For this reason most boundary element codes offer nowadays subregions, particularly for analysing three dimensional problems.

2.15 Helmholtz Equation

Another useful equation in potential problems is the so called Helmholtz or wave equation. Its time dependent version is

$$\nabla^2 u + \lambda^2 u = 0 \qquad \text{in } \Omega \tag{2.154}$$

where ∇^2 is the two or three dimensional Laplacian and λ^2 is a positive and known parameter. Let us consider that the boundary conditions are the normal

$$\begin{array}{lll} \text{(i) Essential conditions} & u = \bar{u} & \text{on } \Gamma_1 \\ \text{(ii) Natural conditions} & q = \bar{q} & \text{on } \Gamma_2 \end{array} \tag{2.155}$$

Mixed boundary conditions can easily be incorporated in the formulation and are important in many practical problems. We will however restrict the discussion now to the above two conditions for simplicity.

The corresponding weighted residual statement for (2.154) and (2.155) is

$$\int_\Omega (\nabla^2 u + \lambda^2 u) u^* \, d\Omega = \int_{\Gamma_2} (q - \bar{q}) u^* \, d\Gamma - \int_{\Gamma_1} (u - \bar{u}) q^* \, d\Gamma \tag{2.156}$$

Integrating twice by parts one obtains.

$$\begin{aligned} \int_\Omega (\nabla^2 u^* + \lambda^2 u^*) u \, d\Omega = & - \int_{\Gamma_2} \bar{q} u^* \, d\Gamma - \int_{\Gamma_1} q u^* \, d\Gamma \\ & + \int_{\Gamma_1} \bar{u} q^* \, d\Gamma + \int_{\Gamma_2} u q^* \, d\Gamma \end{aligned} \tag{2.157}$$

or in more compact form

$$\int_\Omega (\nabla^2 u^* + \lambda^2 u^*) u \, d\Omega = -\int_\Gamma q u^* \, d\Gamma + \int_\Gamma u q^* \, d\Gamma \tag{2.158}$$

where $\Gamma = \Gamma_1 + \Gamma_2$.

The fundamental solution u^* for the Helmholtz equation now needs to satisfy

$$\nabla^2 u^* + \lambda^2 u^* + \Delta_i = 0 \tag{2.159}$$

For two dimensions this results in

$$u^* = \frac{i}{4} H_0^{(1)}(\lambda r) \tag{2.160}$$

and

$$q^* = \frac{\partial u^*}{\partial r} = -\frac{\lambda i}{4} H_1^{(1)}(\lambda r) \tag{2.161}$$

where r is as usual the distance from the source point to the point under consideration, $i = \sqrt{-1}$, $H_0^{(1)}$ is the Hankel function of the first kind and zero order and $H_1^{(1)}$ a Hankel function of the first kind and first order, i.e.

$$H_0^{(1)}(\lambda r) = J_0(\lambda r) + iY_0(\lambda r) \tag{2.162}$$

$$H_1^{(1)}(\lambda r) = J_1(\lambda r) + iY_1(\lambda r) \tag{2.163}$$

where J and Y are Bessel functions of first and second kind, with the subscripts indicating their order.

The three dimensional fundamental solution is

$$u^* = \frac{1}{4\pi r} e^{i\lambda r} \tag{2.164}$$

and

$$q^* = \frac{1}{4\pi r}\left(\frac{-1}{r} + i\lambda\right) e^{i\lambda r} \tag{2.165}$$

2.16 Axisymmetric Formulation

A way of dealing with the formulation for axisymmetric problems is by integrating the three dimensional solution.

Assume first that all boundary values have axial symmetry and consequently all domain values are axisymmetric. We can write the governing integral equation (2.42) in terms of the cylindrical polar coordinates (ρ, θ, Z) given in figure 2.25 as,

$$c^i u^i + \int_{\hat{\Gamma}} u \int_0^{2\pi} q^* \, d\theta \, \rho \, d\hat{\Gamma} = \int_{\hat{\Gamma}} q \int_0^{2\pi} u^* \, d\theta \, \rho \, d\hat{\Gamma} \tag{2.166}$$

where $\hat{\Gamma}$ is the generating body created by the intersection of the three dimensional Γ surface with the part of the plane defined by ρ and Z positive.

The three dimensional fundamental solution can be written in cylindrical coordinates

$$u^* = \frac{1}{4\pi r} = \frac{1}{4\pi r(\rho, \theta, Z)} \tag{2.167}$$

and then integrated with respect to θ, which gives the type of ring source solution required by axisymmetric problems, i.e.

$$\hat{u}^* = \int_0^{2\pi} u^* \, d\theta = \frac{4K(m)}{(a+b)^{1/2}} \tag{2.168}$$

where: $m = \dfrac{2b}{a+b} \qquad (0 \leqslant m \leqslant 1)$

$$a = \rho_i^2 + \rho^2 + (Z_i - Z)^2; \qquad b = 2\rho_i\rho \tag{2.169}$$

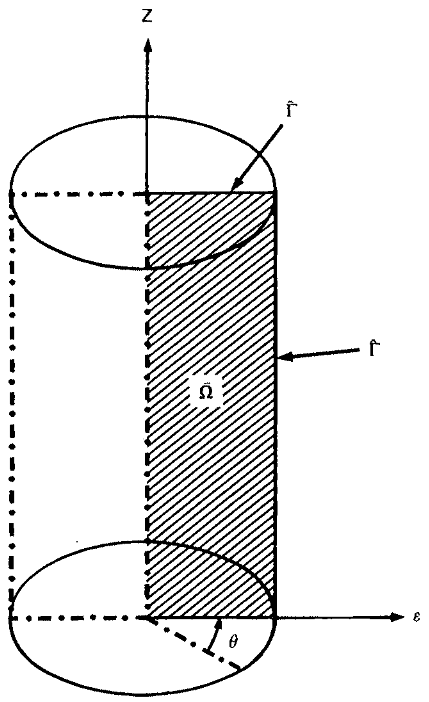


Figure 2.25 Geometrical definition, generating area and boundary contour of solid of revolution

$K(m) =$ is the complete elliptical integral of the first kind.

The subscript 'i' refers to the position of the fundamental solution. Notice that unlike the 2 or 3 dimensional cases, the axisymmetric fundamental solution can not be written simply as a function of the distance between two points but also depends on the distance from the source 'i', the point under consideration to the axis of revolution.

A different formulation of the axisymmetric problem will be presented in section 3.9 where the elliptical integral is avoided by direct numerical integration of the three-dimensional fundamental solution.

2.17 Indirect Formulation

Up to now we have been using the so-called direct formulation which is nowadays the one commonly associated with boundary elements. In the past it was frequent to associate boundary integral solutions with what is now called the 'indirect' approach. In this formulation the boundary solutions are obtained by using sources or sometimes dipoles and it is interesting to point out that this can be interpreted as a particular case of the more general direct approach.

Considering for instance the Poisson equation, i.e.

$$\nabla^2 u + b = 0 \qquad \text{in } \Omega \tag{2.170}$$

The weighted residual formulation applied on (2.170) produces the direct method integral statement, i.e. for any point in the body.

$$u^i + \int_\Gamma uq^* \, d\Gamma = \int_\Gamma u^* q \, d\Gamma - \int_\Omega bu^* \, d\Omega \tag{2.171}$$

If we now call the region external to Ω by Ω' and assume that u' is the solution to the Laplace equation, $\nabla^2 u' = 0$ within Ω', one can write the following statement,

$$\int_\Gamma q^* u' \, d\Gamma - \int_\Gamma u^* q' \, d\Gamma = 0 \tag{2.172}$$

where the point 'i' has been assumed as external to Ω' (internal to Ω) and the tractions q and q^* have been referred to the normal n to the internal domain so that the two values of q^* in (2.171) and (2.172) become identical. One can now specify u' as the solution in Ω' which generates potentials around Γ identical to those of our initial problem, in Ω, i.e.

$$u \equiv u' \qquad \text{on } \Gamma \tag{2.173}$$

We can now add equation (2.153) and (2.154) to give

$$u^i - \int_\Gamma u^*(q - q') \, d\Gamma + \int_\Omega bu^* \, d\Omega = 0 \tag{2.174}$$

or,

$$u^i = \int_\Gamma u^* \sigma \, d\Gamma - \int_\Omega b u^* \, d\Omega \tag{2.175}$$

$\sigma = q - q'$ represents the initially unknown density distribution of u^* over Γ necessary to generate u_i through equation (2.175). The physical interpretation of σ is the difference between the fluxes generated by the two solutions (internal and external).

We can also deduce another 'indirect' formulation from the same equations (2.171) and (2.172). This is the indirect approach in terms of dipoles. Instead of continuity of potentials between the two fields one can assume continuity of fluxes, i.e.

$$q - q' \equiv 0 \tag{2.176}$$

Then one can subtract the two equations to obtain

$$u^i + \int_\Gamma (u - u') q^* \, d\Gamma = - \int_\Omega b u^* \, d\Omega \tag{2.177}$$

or

$$u^i = \int_\Gamma \mu q^* \, d\Gamma - \int_\Omega b u^* \, d\Omega \tag{2.178}$$

$\mu = u' - u$ are called dipoles.

Notice that in order to apply material conditions on Γ, in both the source or dipole formulation one needs still to compute the derivatives of (2.175) or (2.178). This is rather cumbersome in the dipole formulation as it involves derivatives of the potential.

2.18 Other Approaches for the Treatment of Domain Integrals

In section 2.12 the case of Poisson's equation has already been discussed, i.e.

$$\nabla^2 u = b \qquad \text{in } \Omega \tag{2.179}$$

where b is a known function of position.

The boundary integral equation for this case can be expressed as Equation (2.116), i.e.

$$c^i u^i + \int_\Gamma u q^* \, d\Gamma + \int_\Omega b u^* \, d\Omega = \int_\Gamma q u^* \, d\Gamma \tag{2.180}$$

Notice that although the b function is known and the domain integrals do not introduce any unknowns, solution of (2.180) usually requires the discretization of the domain into a series of cells. Once this is done one can apply numerical integration schemes such as Gauss and this results in the following system after discretization

$$\sum_{j=1}^{N} H^{ij}u^j + D^i = \sum_{j=1}^{N} G^{ij}q^j \tag{2.181}$$

The drawback of that type of solution is that it requires the discretization of the domain and introduction of further approximations which deteriorates the accuracy of the method.

It was also mentioned in section 2.11 the possibility of applying a higher order fundamental solution v^* such that

$$u^* = \nabla^2 v^* \tag{2.182}$$

This effectively allows the integration by parts of the domain term in (2.180) and writing it in terms of boundary integrals only. The technique was defined as valid only for the case $\nabla^2 b \equiv 0$.

More recently researchers have proposed two new approaches for taking the domain integrals to the boundary, one based on the generalization concepts described in the paragraph above and called the Multiple Reciprocity Method [12] and the other on the idea of using a series of particular solutions and called Dual Reciprocity Method [13] in the literature. They will be described in the next sections, after a brief description of the use of particular solutions as an introduction to Dual Reciprocity.

(i) The Use of Particular Solutions

An obvious way of solving equation (2.180) without domain integrals is by changing the variables in such a manner that these integrals disappear. This can be attempted by adding a particular solution to a new variable.

To illustrate the procedure consider the Poisson equation (2.179) with boundary conditions such as

$$\left.\begin{array}{lll} \text{Essential conditions} & u = \bar{u} & \text{on } \Gamma_1 \\ \text{Natural conditions} & q = \bar{q} & \text{on } \Gamma_2 \end{array}\right\} \Gamma = \Gamma_1 + \Gamma_2 \tag{2.183}$$

Assume now that the potential u can be written as,

$$u = \tilde{u} + \hat{u} \tag{2.184}$$

where $\hat{u}$ is a particular solution of the Poisson equation, such that

$$\nabla^2 \hat{u} = b \tag{2.185}$$

One can now write the domain term in equation (2.180) as follows,

$$\int_\Omega bu^* \, d\Omega = \int_\Omega (\nabla^2 \hat{u}) u^* \, d\Omega \tag{2.186}$$

Integrating by parts this expression one finds the following relationships,

$$\begin{aligned} \int_\Omega bu^* \, d\Omega &= \int_\Omega (\nabla^2 \hat{u}) u^* \, d\Omega \\ &= \int_\Omega \hat{u}(\nabla^2 u^*) d\Omega + \int_\Gamma u^* \; \hat{q} \; d\Gamma - \int_\Gamma q^* \hat{u} \, d\Gamma \end{aligned} \tag{2.187}$$

where $\hat{q} = \frac{\partial \hat{u}}{\partial n}$.

Taking into consideration the special properties of the fundamental solution (see section 2.2) one can write the right hand side term in (2.187) as follows,

$$\int_\Omega bu^* \, d\Omega = -c^i \hat{u}^i + \int_\Gamma u^* \hat{q} \, d\Gamma - \int_\Gamma q^* \hat{u} \, d\Gamma \tag{2.188}$$

Substituting (2.188) into (2.180) one finds the following expression,

$$\begin{aligned} &c^i u^i + \int_\Gamma q^* u \, d\Gamma - \int_\Gamma q u^* \, d\Gamma \\ &= c^i \hat{u}^i + \int_\Gamma q^* \hat{u} \, d\Gamma - \int_\Gamma \hat{q} u^* \, d\Gamma \end{aligned} \tag{2.189}$$

Notice that now all integrals need to be computed only on the Γ boundary. Equation (2.189) can also be written in a more compact form as function of the new variable $\tilde{u}$, i.e.

$$c^i \tilde{u}^i + \int_\Gamma q^* \tilde{u} \, d\Gamma = \int_\Gamma \tilde{q} u^* \, d\Gamma \tag{2.190}$$

where $\tilde{u} = u - \hat{u}$ defines the new variable.

The main difficulty with particular solutions is that they are difficult to find in many cases and cannot usually be applied to time dependent or non-linear problems.

Example 2.7

This example describes how to apply particular solutions to the elliptical section described in example 2.6. (Figure 2.22).
Here the Poisson equation $\nabla^2 u = -2$ is solved using the solution procedure described above. *i.e.*

$$u = \tilde{u} + \hat{u} \qquad \text{(a)}$$

which is to say that the complete solution u will be expressed as the sum of the solution to the homogeneous Laplace equation, $\tilde{u}$, plus a particular solution to the Poisson equation, $\hat{u}$.

The reader can easily see that

$$\hat{u} = -\frac{1}{2}(x^2 + y^2) \qquad \text{(b)}$$

is such a solution. Results may now be obtained for the Poisson equation solving the Laplace equation, using POLIMBE, using the boundary conditions defined by (b) with a change of sign, in order that when the sum, (a) is carried out the net result will be the imposition of a homogeneous boundary condition on Γ.

The problem geometry will be the same as that used in section 2.12, figure 2.22, but without the internal cells, with which the results of this solution procedure may be compared.

The solution procedure may easily be understood examining table A

Table A Results for Poisson Problem using Particular Solutions

Node	X	Y	$\tilde{u}$	$\hat{u}$	$u = \tilde{u} + \hat{u}$	Cell	Exact
17	1.5	0.0	1.473	−1.125	0.348	0.331	0.350
18	1.2	−0.35	1.200	−0.781	0.419	0.401	0.414
19	0.6	−0.45	0.855	−0.281	0.574	0.557	0.566
20	0.0	−0.45	0.747	−0.101	0.646	0.629	0.638
29	0.9	0.0	1.049	−0.405	0.644	0.626	0.638
30	0.3	0.0	0.835	−0.045	0.790	0.772	0.782
31	0.0	0.0	0.808	0.000	0.808	0.791	0.800

In table A column $\tilde{u}$ is the solution to the Laplace equation with essential boundary conditions defined by equation (b) and with a change of sign. Column $\hat{u}$ is the result of evaluating (b) for the coordinates of the nodes. $u = \tilde{u} + \hat{u}$ is the sum of the two and the problem solution. Note that here a domain integral problem has been solved without the use of internal cells.

(ii) The Dual Reciprocity Method

The Dual Reciprocity Method (DRM) was invented by Nardini and Brebbia [13, 14] in 1982 for the solution of elastodynamic problems. It is essentially a generalized way of constructing particular solutions and it can be used to represent internal forces or body force distributions as well as for solving non-linear and time-dependent problems. The method can be applied to define body force distribution over the whole domain or only part [15].

It was realized early on that the approach could be used to solve a wide variety of parabolic as well as hyperbolic time dependent problems [16]. A complete description of how the method can be applied to vibrations has been presented by Nardini and Brebbia in reference [17]. The case of transient analysis has been studied in detail by Brebbia et al. [18–22] including the extension of the technique to deal with non-linear diffusion problems and axisymmetric problems and a whole book has been published dealing with the method [23] and its applications.

The Dual Reciprocity Method starts by representing any given function b as a combination of a series of known coordinate functions f^j where $f^j = f(\xi_j, x)$ is a function between a point ξ_j and any other point x, α^j are unknown coefficients associated with each of the f^j functions. The f^j are considered to originate at 'j' different points, many of them on the boundary (see figure 2.26). These points are called 'poles' and the f^j functions are the same type for all those points. The form of b is hence as follows,

$$b = \sum_{j=1}^{M} f^j \alpha^j \tag{2.191}$$

M is the number of poles equal to boundary nodes (N) plus internal nodes (L)

The next stage is to define the particular solution, corresponding to the generic function f^j, i.e.

$$\nabla^2 \hat{u}^j = f^j \tag{2.192}$$

The $\hat{u}^j$ field can be found by integrating the above equation. One can then compute the value of any associated variable such as $\hat{q}^j$ by differentiating the particular solution.

Equation (2.180) can now be written as follows,

$$c^i u^i + \int_\Gamma q^* u \, d\Gamma - \int_\Gamma q^* u^* \, d\Gamma = -\sum_{j=1}^{M} \left\{ \alpha^j \int_\Omega (\nabla^2 \hat{u}^j)) u^* \, d\Omega \right\} \tag{2.193}$$

Each term on the right hand side of the above equation can be integrated by parts resulting in only boundary integrals. (Notice that each 'j' term involves a particular solution localized at a 'j' pole). This gives

$$c^i u^i + \int_\Gamma q^* u \, d\Gamma - \int_\Gamma q^* u^* \, d\Gamma = \sum_{j=1}^{M} \left\{ \alpha^j \left[c^j \hat{u}^j + \int_\Gamma q^* \hat{u}^j \, d\Gamma - \int_\Gamma u^* \hat{q}^j \, d\Gamma \right] \right\} \tag{2.194}$$

This formula has only boundary integrals but the right hand side is not found by using only one particular solution but by the addition of a series of terms, each of them representing the effect of a particular solution localized at a 'j' pole.

After applying the usual boundary element discretization equation (2.194) gives rise to the following system,

$$\mathbf{HU} - \mathbf{GQ} = [\mathbf{H\hat{U}} - \mathbf{G\hat{Q}}]\alpha \tag{2.195}$$

or simply

$$\mathbf{HU} - \mathbf{GQ} = \mathbf{S}\alpha \tag{2.196}$$

where

$$\mathbf{S} = \mathbf{H\hat{U}} + \mathbf{G\hat{Q}} \tag{2.197}$$

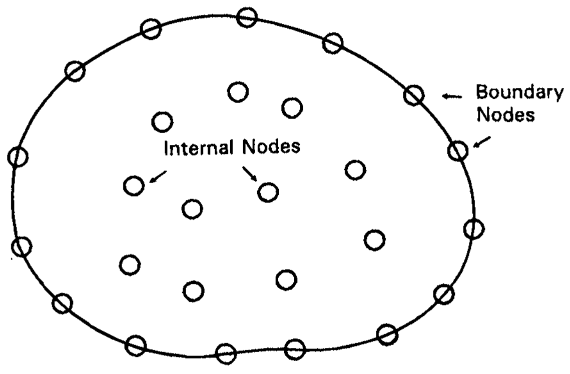


Figure 2.26 The total number of Poles given by the sum of all Boundary and Internal Nodes

$\mathbf{\hat{U}}$ and $\mathbf{\hat{Q}}$ are matrices $N \times M$ each for the boundary solution. (M is the number of points where the function 'j' has been applied or poles; N is the number of boundary unknowns as usual and L the number of internal nodes. Hence $M = N + L$). The columns of these matrices represent the values of the $\hat{u}^j$ and $\hat{q}^j$ functions at the different nodes for the case of the f^j function acting at a particular 'j' pole.

Notice that the α coefficients are different from the values of the b function at the points under consideration. Both are related through equation (2.191), which expressed in matrix form gives a $M \times M$ matrix F which can be inverted, i.e.

$$\mathbf{B} = \mathbf{F}\boldsymbol{\alpha} \tag{2.198}$$

where **B** vector represents the magnitudes of the b functions at the poles. **F** is a matrix whose elements are the values of f^j at all points for each position 'j' of the function f. Hence one can invert (2.198) to find α vector, i.e.

$$\boldsymbol{\alpha} = \mathbf{F}^{-1}\mathbf{B} = \mathbf{EB} \tag{2.199}$$

where $\mathbf{E} = F^{-1}$.
Equation (2.196) can now be written as,

$$\mathbf{HU} - \mathbf{GQ} = (\mathbf{SE})\mathbf{B} \tag{2.200}$$

The **H**, **G** and **S** matrices are functions of the geometry of the problem, the position of nodes and poles, type of interpolation and the $\hat{u}^j$, $\hat{q}^j$ fields of the particular solutions.

Choice of Functions

The most important consideration in the computational implementation of the DRM is the choice of appropriate interpolation functions f^j. After a series of numerical experiments, Brebbia and Nardini [13] [17] proposed using 'conical' functions of the type distance between the points of application of the function ξ, and any given point x, such as

$$f^j = r(\xi_j, x) \tag{2.201}$$

This gives a very simple type of u^j function for the Laplace operator, namely,

$$\hat{u}^j(x) = \frac{1}{9} r^3 \tag{2.202}$$

In addition those authors recommended adding a suitably chosen constant for completeness, which can then be incorporated through the following function,

$$f^j = 1 \quad (\text{or sometimes generalised as } f^j = 1 + r(\xi_j, x)) \tag{2.203}$$

This gives the following particular solution for the Laplace's equation

$$\hat{u}^j = \frac{r^2}{4} \tag{2.204}$$

The introduction of one more unknown will require setting up another equation in terms of α, which can be done by defining an internal degree of freedom or pole. In general the introduction of more degrees of freedom is always recommended to obtain better results when the b function is difficult to represent as a function of the boundary values only.

Other types of function proposed by Nardini and Brebbia can be seen in reference [13]. They include localized harmonic functions and polynomials in terms of the coordinates, but the best results were always reported using 'conical' functions.

For axisymmetric cases the solution depends not only on r but also on the distance from the source and field points to the axis of revolution. Because of this Wrobel, Telles and Brebbia [22] have proposed the following function,

$$f^j = r\left(1 - \frac{R_j}{4R}\right) \tag{2.205}$$

where R_j is the distance from the different poles to the axis of revolution and R is the distance from any point on Γ (or Ω) to the same axis. The $\hat{u}$ and $\hat{q}$ functions are the same type as for three dimensional analysis.

Example 2.8 Results for Poisson Equation Problem Using Dual Reciprocity Method [23]

The eliptical section described in figure 2.22 will be solved here again but now using Dual Reciprocity Methods. The body discretization consists of 16 linear elements as shown in the figure and the 17 internal nodes will be used as internal poles.

The results presented in the table A for the twelve poles shown in figure 2.22 describe the complete numerical solution due to symetry. The boundary element results are rather poor at node 1 due to the coarse discretization used there. Results obtained in example 2.6 using cells are also included for comparison.

Table A Results for Torsion of Ellipse for Different f functions

Variable	Node	X	Y	$f = r$	$f = 1 + r$	$f = 1$ or r	Cell	Exact
q	1	2.0	0.0	−0.680	−0.680	−0.682	−0.733	−0.8
	2	1.706	−0.522	−1.019	−1.020	−1.024	−1.046	−1.018
	3	1.179	−0.808	−1.357	−1.359	−1.363	−1.378	−1.322
	4	0.598	−0.954	−1.531	−1.532	−1.536	−1.549	−1.528
	5	0.0	−1.0	−1.587	−1.588	−1.592	−1.611	−1.6
u	17	1.5	0.0	0.349	0.349	0.350	0.331	0.350
	18	1.2	−0.35	0.418	0.418	0.418	0.401	0.414
	19	0.6	−0.45	0.574	0.573	0.573	0.557	0.566
	20	0.0	−0.45	0.646	0.646	0.646	0.629	0.638
	29	0.9	0.0	0.643	0.643	0.643	0.626	0.638
	30	0.3	0.0	0.789	0.789	0.789	0.772	0.782
	31	0.0	0.0	0.807	0.807	0.807	0.791	0.800

For the linear elements employed in the discretization, results using the three different f functions are seen to be very similar, thus the use of $f = 1 + r$ is recommended as this is the simplest to apply, requiring no special consideration. Note that the cell collocation results are much less accurate. In table B DRM results are presented for the same problem with $f = 1 + r$ considering different numbers of internal nodes.

Table B Results for Torsion Problem for Different Values of L

Variable	Node	X	Y	$L = 17$	$L = 13$	$L = 9$	$L = 5$	$L = 1$	Exact
q	1	2.0	0.0	−0.680	−0.678	−0.677	−0.676	−0.666	−0.8
	2	1.706	−0.522	−1.020	−1.017	−1.016	−1.013	−0.995	−1.018
	3	1.179	−0.808	−1.359	−1.357	−1.354	−1.349	−1.325	−1.322
	4	0.598	−0.954	−1.532	−1.530	−1.525	−1.517	−1.499	−1.528
	5	0.0	−1.0	−1.588	−1.585	−1.580	−1.573	−1.557	−1.6
u	31	0.0	0.0	0.807	0.806	0.803	0.798	0.788	0.800

It can be seen that in this case the sensitivity of the results to number of internal nodes is small. The total variation of results from $L = 1$ to $L = 17$ is approximately 2%.

(iii) The Multiple Reciprocity Method (MRM)

The Multiple Reciprocity Method is a generalization of the Galerkin technique described in section 2.12 for taking domain integrals to the boundary. It has some features which are similar to the DRM but instead of approximating the source terms by a set of coordinate functions it uses a sequence of functions related to the fundamental solution as will be shown. This sequence forms a set of higher order fundamental solutions which permits the application of Green's identity to each term of the sequence in succession. As a result the method can lead in the limit to the exact boundary only formulation of the problems.

The MRM was presented in its present form by Nowak and Brebbia [12] [24] who applied it to solve transient and Helmholtz type problems.

Consider again the case of Poisson's equation (2.179) but now with a subscript 'o' on the right hand side term to differentiate from other similar functions which will be generated during the solution, i.e.

$$\nabla^2 u = b_o \qquad \text{in } \Omega \tag{2.206}$$

where u and b_o are the potential and the source functions respectively.

The fundamental solution to Laplace's equation will also be called with a subscript 'o' for similar reasons as explained above, i.e.

$$\nabla^2 u_o^* + \Delta^i = 0 \tag{2.207}$$

Applying reciprocity to the above relations one obtains the same expression as before i.e. equation (2.180), which is written again below for completeness

$$c^i u^i + \int_\Gamma q_o^* u \, d\Gamma + \int_\Omega b_o u_o^* \, d\Omega = \int_\Gamma q u_o^* \, d\Gamma \tag{2.208}$$

where $q_o^* = \partial u_o^* / \partial n$.

The domain integral in (2.208) can now be transformed into a series of equivalent boundary integrals. In order to do this one introduces a new function u_1^* related to the fundamental solution u_o^* by the formula

$$\nabla^2 u_1^* = u_o^* \tag{2.209}$$

Thus the domain integral in (2.208) can be expressed as follows,

$$\begin{aligned}\int_\Omega b_o u_o^* \, d\Omega &= \int_\Omega b_o(\nabla^2 u_1^*) d\Omega \\ &= \int_\Gamma \left\{ b_o \frac{\partial u_1^*}{\partial n} - u_1^* \frac{\partial b_o}{\partial n} \right\} d\Gamma + \int_\Omega u_1^* \, \nabla^2 b_o \, d\Omega \end{aligned} \tag{2.210}$$

As the source function b_o is at present assumed to be a known function of space, one can obtain the function $\nabla^2 b_o$ analytically and hence one can define a new function b_1 such that

$$b_1 = \nabla^2 b_o \tag{2.211}$$

The domain integral on the right hand side of (2.210) can then be written in a form which is similar to the one for the previous domain integral and can be expanded in the same way, i.e.

$$\int_\Omega b_1 u_1^* \, d\Omega = \int_\Gamma \left\{ b_1 \frac{\partial u_2^*}{\partial n} - u_2^* \frac{\partial b_1}{\partial n} \right\} d\Gamma + \int_\Omega u_2^* (\nabla^2 b_1) d\Omega \tag{2.212}$$

Notice that one can now compute a b_2 function such that

$$b_2 = \nabla^2 b_1 \tag{2.213}$$

and continue carrying out this procedure as many times as desired.

The procedure can be generalized by introducing two sequences of functions defined by the following recurrence formula,

$$\begin{aligned} b_{j+1} &= \nabla^2 b_j \\ \nabla^2 u^*_{j+1} &= u^*_j \end{aligned} \qquad \text{for } j = 0, 1, 2 \ldots \tag{2.214}$$

The domain integral can then by expressed as

$$\int_\Omega b_o u^*_o \, d\Omega = \sum_{j=0}^{\infty} \int_\Gamma \left[b_j \frac{\partial u^*_{j+1}}{\partial n} - u^*_{j+1} \frac{\partial b_j}{\partial n} \right] d\Gamma \tag{2.215}$$

Introducing (2.215) into the original boundary integral expression one obtains the exact boundary only formulation of the problems, i.e.

$$\begin{aligned} c^i u^i + \int_\Gamma u q^*_o \, d\Gamma &= \int_\Gamma q u^*_o \, d\Gamma \\ &- \sum_{j=0}^{\infty} \int_\Gamma \left[b_j \frac{\partial u^*_{j+1}}{\partial n} - u^*_{j+1} \frac{\partial b_j}{\partial n} \right] d\Gamma \end{aligned} \tag{2.216}$$

The integrals can be evaluated numerically by subdividing the boundary Γ into elements as usual. As the functions b_j are known functions of space the integrals in the summation can be calculated directly. The same type of interpolations used for u and q can be applied for b_j. Thus equation (2.216) can be expressed in terms of the usual boundary element influence matrices now called $\mathbf{H}_o$ and $\mathbf{G}_o$ plus those matrices resulting from the use of higher order fundamental solutions, which are defined as $\mathbf{H}_{j+1}$ and $\mathbf{G}_{j+1}$ ($j = 0, 1, 2 \ldots$) i.e.

$$\mathbf{H}_o \mathbf{U} - \mathbf{G}_o \mathbf{Q} = \sum_{j=0}^{\infty} (\mathbf{H}_{j+1} \mathbf{P}_j - \mathbf{G}_{j+1} \mathbf{R}_j) \tag{2.217}$$

The vectors $\mathbf{P}_j$ and $\mathbf{R}_j$ contain the function b_j and its normal derivatives respectively at the boundary nodes.

Notice that the terms of the series in the right hand side of equation (2.217) vanish rapidly provided that the problem has been properly scaled, (i.e. all dimensions are divided by the maximum dimension of the problem). It has been shown that the convergence of the series is very rapid in a variety of practical cases. Furthermore this convergence can be easily calculated since all the terms are known and the contribution of each of them can be evaluated. It should also be pointed out that the functions u^*_j have no singularities for $j = 1, 2 \ldots$ and thus their integration does not require any special technique.

Equation (2.217) is solved using standard boundary element subroutines after taking into consideration the boundary conditions.

Higher Order Fundamental Solutions

The higher order fundamental solutions required here are defined by the recurrence formula (2.215). This equation can be easily solved analytically when the Laplace's operator is written in terms of a cylindrical (for 2-D problems) or spherical (for 3-D cases) coordinate system. For example, for 2-D problems the general form of the function u_j^* is given by the following expression,

$$u_j^* = \frac{1}{2\pi} r^{2j}(A_j \ln r - B_j) \tag{2.218}$$

where r represents the distance between points. The coefficients A_j and B_j are obtained for the following recurrence relationships

$$\begin{aligned} A_{j+1} &= \frac{A_j}{4(j+1)^2} \\ B_{j+1} &= \frac{1}{4(j+1)^2}\left\{\frac{A_j}{j+1} + B_j\right\} \end{aligned} \tag{2.219}$$

For $j = 0$ one obtains the classical fundamental solution, i.e. $A_o = 1$ and $B_o = 0$. Notice that formula (2.219) introduce factorials into the denominators of coefficients A_j and B_j and hence guarantee their rapid convergence.

References

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

[2] Brebbia, C. A. and Dominguez, J. Boundary Element Methods versus Finite Elements. *Proc. Int. Conference on Applied Numerical Modelling, Southampton University, 1977.* Ed. C. A. Brebbia, Pentech Press, London, 1978.

[3] Brebbia, C. A. and Dominguez, J. Boundary Element Methods for Potential Problems. *Applied Mathematical Modelling,* **1**, 7, December 1977.

[4] Hess, J. L. and Smith, A. M. O. Calculation of Potential Flow about Arbitrary Bodies. *Progress in Aeronautical Sciences Vol. 8.* Ed. D. Kuchemann, Pergamon Press, London, 1967.

[5] Harrington, R. F., Pontoppidan, K., Abramhamsen, P. and Albertsen, N. C. Computation of Laplacian Potentials by an Equivalent-source Method. *Proc. IEE,* **116**, 1715–1720 1969.

[6] Mautz, J. R. and Harrington, R. F. Computation of Rotationally Symmetric Laplacian Potentials. *Proc. IEE,* **117**, 850–852, 1970.

[7] Jaswon, M. Integral Equation Methods in Potential Theory, I. *Proc. Roy. Soc. Ser. A.*, **275**, 23–32, 1963.

[8] Symm, G. T. Integral Equation Methods in Potential Theory, II. *Proc. Roy. Soc. Ser. A.*, **275**, 33–46, 1963.

[9] Jaswon, M. and Ponter, A. R. An Integral Equation Solution of the Torsion Problem. *Proc. Roy. Soc. Ser. A.*, **273**, 237–246, 1963.

[10] Brebbia, C. A. and Ferrante, A. *Computational Methods for the Solution of Engineering Problems.* Pentech Press, London, Wiley, USA, 1986.

[11] Danson, D. J., Brebbia, C. A. and Adey, R. A. BEASY. A Boundary Element Analysis System. *Finite Element Systems Handbook.* Springer-Verlag and Computational Mechanics Publications, Southampton, 1982.

[12] Nowak, A. J. and Brebbia, C. A. The Multiple Reciprocity Method – A New Approach for Transforming BEM Domain Integrals to the Boundary. *Eng. Analysis*, **6**(3), 1989.

[13] Nardini, D. and Brebbia, C. A. A New Approach to Free Vibration Analysis using Boundary Elements, in Boundary Element Methods in Engineering (ed. C. A. Brebbia), Springer-Verlag, Berlin and New York, 1982.

[14] Brebbia, C. A. and Nardini, D. Dynamic Analysis in Solid Mechanics by an Alternative Boundary Element Approach, *Int. J. Soil Dynamics and Earthquake Engineering*, **2**(4), 228–233, 1983.

[15] Niku, S. M. and Brebbia, C. A. Dual Reciprocity Boundary Element Formulation for Potential Problems with Arbitrarily Distribution Forces. Technical Note, *Eng. Analysis*, **5**(1), 1988.

[16] Nardini, D. and Brebbia, C. A. The Solution of Parabolic and Hyperbolic Problems using an Alternative Boundary Element Formulation. In Proc. of VIIth Int. Conf. on BEM in Eng. Computational Mechanics Publications, Southampton and Boston, Springer-Verlag, Berlin, 1985.

[17] Nardini, D. and Brebbia, C. A. Boundary Integral Formulations of Mass Matrices for Dynamics Analysis. In Topics in Boundary Element Research, Vol. 2 (ed. C. A. Brebbia) Springer-Verlag, Berlin, and New York, 1985.

[18] Wrobel, L. C., Brebbia, C. A. and Nardini, D. The Dual Reciprocity Boundary Element Formulation for Transient Heat Conduction. In Proc. of the Vth Int. Conf. on FEM in Water Resources. Computational Mechanics Publications, Southampton and Boston, Springer-Verlag, Berlin, 1986.

[19] Wrobel, L. C. and Brebbia, C. A. The Dual Reciprocity Boundary Element Formulation for Non-Linear Diffusion Problems. Computer Methods in Applied Mechanics and Engg. Vol. 65, pp. 147–164, 1987.

[20] Wrobel, L. C., Brebbia, C. A. and Nardini, D. Analysis of Transient Thermal Problems in the BEASY system. In BETECH/86 (Eds. J. J. Connor and C. A. Brebbia), Computational Mechanics Publications, Southampton and Boston, 1986.

[21] Brebbia, C. A. and Wrobel, L. C. Non-Linear Transinet Thermal Analysis using the Dual Reciprocity Method. In Boundary Element Techniques: Applications in Stress Analysis and Heat Transfer, (Eds. C. A. Brebbia and W. Venturini), Computational Mechanics Publications, Southampton and Boston, 1987.

[22] Wrobel, L. C., Telles, J. C. F. and Brebbia, C. A. A Dual Reciprocity Boundary Element Formulation for Axisymmetric Diffusion Problems. In Proc. of VIIth Int. Conf. on B.E.M. in Engg., Tokyo, 1986. Computational Mechanics Publications, Southampton and Boston, Springer-Verlag, Berlin, 1986.

[23] Partridge, P, Brebbia, C. A. and Wrobel, L. C. 'The Dual Reciprocity Boundary Element Method'. Computational Mechanics Publications, Southampton and Boston, and Elsevier, London, 1991.

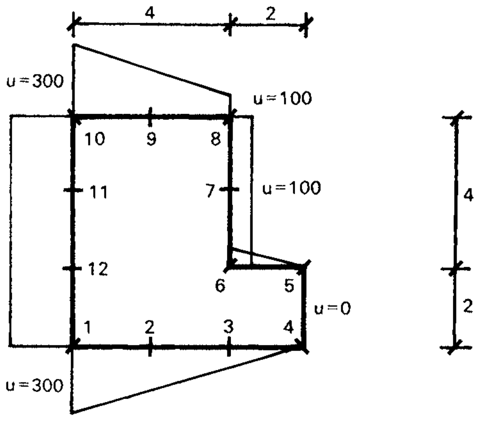
[24] Nowak, A. J. and Brebbia, C. A. Solving Helmholtz Equation by Boundary Elements using the Multiple Reciprocity Method. In Computers and Experiments in Fluid Flow, (Eds G. Carlomagno and C. A. Brebbia) Computational Mechanics Publications, Southampton and Boston Springer-Verlag, Berlin, 1989.

Exercises

2.1. Verify that the two-dimensional fundamental solution satisfies Laplace's equation for any point where $r \neq 0$.

2.2. The two-dimensional fundamental solution is $u^* = (1/2\pi)\ln(1/r)$. Since it includes a logarithm of a non-dimensionless quantity its value depends on the scale. Will this fact change the solution of a problem depending on its units?

2.3. Using program POLINBE, solve the example of the figure with 12 linear elements and prescribing values of the potential on the whole boundary: linear variation of u between nodes 1 and 4, $u = 0$ between nodes 4 and 5 linear variation of u between nodes 5 and 6, $u = 100$ between nodes 6 and 8, linear variation of u between nodes 8 and 10 and $u = 300$ between nodes 10 and 1. Check that the results for the fluxes at the corners are very poor.



2.4. Solve again the example of 2.3 prescribing values of the potential on the whole boundary and introducing six new very small elements at the corners in such a way that the domain has blunt corners. Try several sizes of the corner elements to check the accuracy of the approximation.

2.5. Using program POLINBE, solve again the example of 2.3 with 12 linear elements and assuming that the values of the potential are known on the whole boundary. Since the potential is known along two lines merging into each corner, determine the value of one potential derivative by differences along one of these lines and leave the other as unknown.

2.6. Derive, using the method of images, the fundamental solution for a semi-finite region with the condition that the potential on the free surface is zero.

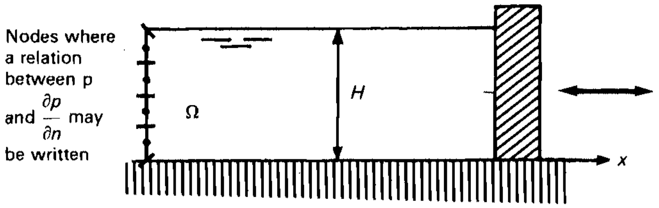
2.7. The same as 2.6 but with the condition that the flux on the free surface is zero.

2.8. Derive expressions for the derivatives of the fundamental solution $\left(\frac{\partial u^*}{\partial x_l}\right)^i$ and $\left(\frac{\partial q^*}{\partial x_l}\right)^i$ which should be used with equation (2.33) to compute internal fluxes.

2.9. Write a subroutine (DERINPC) which given an internal point XP, YP, computes the integrals $\int_{\Gamma_j}\left(\frac{\partial q^*}{\partial x_l}\right)^i d\Gamma$ and $\int_{\Gamma_j}\left(\frac{\partial u^*}{\partial x_l}\right)^i d\Gamma$ along a constant element defined by its extreme points.

2.10. Write a subroutine (IFLUXPC) that using DERINPC and the boundary potentials and fluxes of (FI and DFI) of the program POCONBE computes fluxes at internal points.

2.11. Assume an open channel section that extends from $x=0$ to $x=-\infty$. At the extreme $x=0$ there is a piston that moves harmonically with frequency ω. When the perturbations are small and the fluid is considered to be incompressible with zero viscosity, the equation that governs the motion is: $\nabla^2 p = 0$, p being the pressure, and $\frac{\partial p}{\partial n} = \rho\omega^2 u_n$, where u_n is the displacement along the n direction and ρ the density.



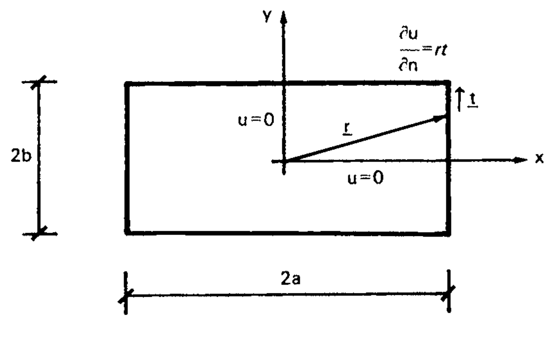
It is known that the solution that satisfies conditions at $z=H$ $(p=0)$ and $z=0$ $\left(\frac{\partial p}{\partial z}=0\right)$, is $P(z,x)=\sum_1^\infty A_n \cos K_n z\, e^{K_n x}$ where $K_n = \frac{2n-1}{2H}\pi$ $(n=1,2,\ldots)$. In order to solve a potential problem in a bounded region Ω for a prescribed motion of the piston, an artificial boundary with four constant elements is introduced at $X=-X_0$. Using as many terms of the series expansion of $p(z,x)$ as nodes exist on the artificial boundary, determine a matrix that relates the nodal values of p and $\frac{\partial p}{\partial n}$ along that boundary (Robin boundary condition).

2.12. Compute, using program POCONBE, the warping function of Saint Venant torsion of a rectangular cross section bar. Use several discretizations to verify the convergency towards the exact solution. Use symmetry to discretize only one quarter of the section.

Exact:

$$u = XY - \frac{32a^2}{\pi^3}\sum_{n=0}^{\infty}\frac{(-1)^n}{(2n+1)^3}\frac{1}{\cosh K_n b}\sin K_n X \sinh K_n Y$$

$$K_n = \frac{2n+1}{2a}\pi$$



2.13. Solve the problem of exercise 2.12 using linear (POLINBE) and quadratic (POQUABE) elements. Compare the solutions.