Appendix A

Established structural approximation methods

A.1 Governing equations

The theory of elasticity is clearly set out in the book by Timoshenko [53]. The basic equations are summarized here for the sake of completeness and basic comprehension.

Consider the partial differential equations

$$\nabla^2 u = 0 \tag{A.1}$$

This is known as Laplace's equation. If we only consider the two-dimensional case, we obtain

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = b \tag{A.2}$$

(A.2) is a special form of (2.7). If b is not equal to zero, it is known as Poisson's equation which governs many types of engineering problems such as seepage and aquifer analysis, heat conduction, diffusion processes, torsion, fluid motion and others [54]. This equation is generally associated with equilibrium or steady state problems and is known as an elliptic partial differential equation as discussed in [28].

We can define a basic coordinate system (x_1, x_2, x_3) that corresponds to displacements (u_1, u_2, u_3). We also define our main stresses σ_{ij} on a body in the same directions and the surface tractions (p_1, p_2, p_3). This is shown in Figure A.1.

For the stresses of the body volume in Figure A.1 we have the equilibrium equations

$$\frac{\partial \sigma_{ij}}{\partial x_j} + b_i = 0 \tag{A.3}$$

Where the b_i term represents the body force per unit volume (internal). We consider (n_1, n_2, n_3) as the direction cosines of the outward normal vector $\vec{n}$ as indicated in Figure A.1. We can now relate the surface tractions p_i to the stresses with the relationship

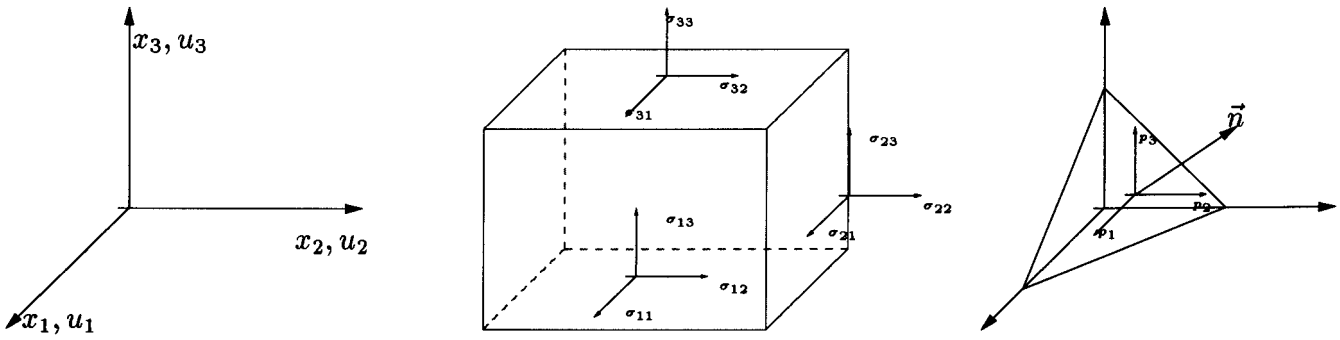


Figure A.1: The main coordinates, displacements, stresses and tractions in three dimensions

$$p_i = \sigma_{ij} n_j \tag{A.4}$$

The deformations of the boundary are a function of the displacements which have the components (u_1, u_2, u_3) at every point on the boundary. The strain is a description of the deformation per unit length; thus we can write the strains in terms of the deformations.

$$\epsilon_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) \tag{A.5}$$

The strain-displacement equations are also called the kinematic equations of deformation equations; they yield the strain field given the displacement field. The constitutive equations connect the stress and strain fields. For linear elasticity however, a considerable simplification occurs because the relation becomes algebraic, linear and homogeneous. Thus for this case the stress-strain relations may be written in component notation as

$$\sigma_{ij} = E_{ijkl}\epsilon_{kl} \tag{A.6}$$

The E_{ijkl} module of elasticity in this case does not presume the same mechanical properties of the material in all directions. They are components of a fourth order tensor $\mathbf{E}$ called the elasticity tensor. The elasticity module generally satisfies the following symmetries

$$E_{ijkl} = E_{jikl} = E_{ijlk} \tag{A.7}$$

This relation reduces the number from $3^4 = 81$ constants to $6^2 = 36$ constants. Furthermore, it is a requirement that the elastic energy that is stored must be positive and if the body admits strain energy[1] the elastic module must satisfy the additional symmetries

$$E_{ijkl} = E_{klij} \tag{A.8}$$

This reduces the number of independent constants to 21. Further symmetries occur if the material is isotropic and the module of elasticity can be expressed in terms of Young's modulus E and Poisson's ratio v.

[1]The material is not only elastic but hyperelastic.

The states of stresses and strains in a body are related through the constitutive relationship for the material as shown in [54] generally they are PDEs. This is shown to be

$$\sigma_{ij} = \frac{E}{1+v}[\frac{v}{1-2v}\delta_{ij}\epsilon_{kk} + \epsilon_{ij}] \tag{A.9}$$

where v is Poisson's ratio, E the Modulus of elasticity and δ_{ij} is Kronecher delta ($\delta_{ij} = 1$ when $i = j$ and 0 if $i \neq j$).

A.1.1 One dimension

The only possibility in one dimension is a stress σ along the direction x and its corresponding strain ϵ. The stress strain relationship reduces to

$$\sigma = E\epsilon \tag{A.10}$$

A.1.2 Two dimensions

Plane stress

A thin planar body subjected to in-plane loading on its edge surface is said to be in plane stress [40]. The state of stress is then specified by only σ_x, σ_y, τ_{xy} [53]. It may be assumed that these three components are independent of z and thus they do not vary through the thickness [53].

Plane strain

Plane strain occurs when a body of uniform cross section is subjected to transverse loading along its length [40]. This simplification usually occurs when the dimension of the body in the z direction is very large [53]. Here ϵ_z, γ_{yz} and γ_{yz} are taken to be zero [40].

A.1.3 Mechanics of the boundary value problem

Consider an elastic bar of length l, constant cross sectional area A, Young's modulus E and a temperature coefficient α. A point load P is applied to the end of the bar and an arbitrarily distributed load is applied to the length of the bar Figure A.2. The problem is purely one-dimensional and will have only one Euclidean coordinate, namely $x_1 \in [0, l]$ [55]. An infinitesimal section of the bar is now chosen. The equilibrium is shown schematically in Figure A.3.

By physical reasoning, the state of equilibrium can be written as

$$-F + (F + dF) + \rho dx_1 = 0$$
$$\rho + \frac{dF}{dx_1} = 0$$

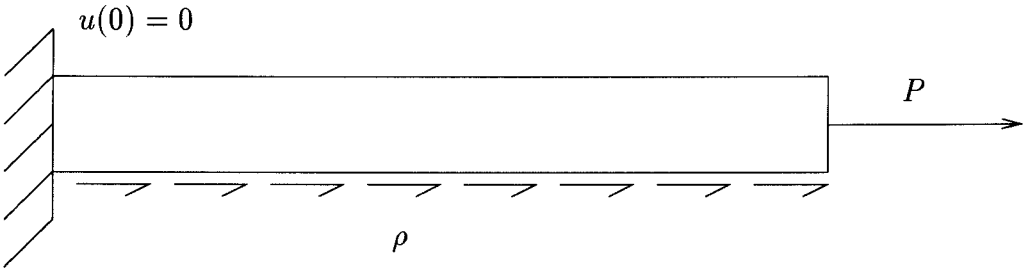


Figure A.2: One-dimensional bar problem

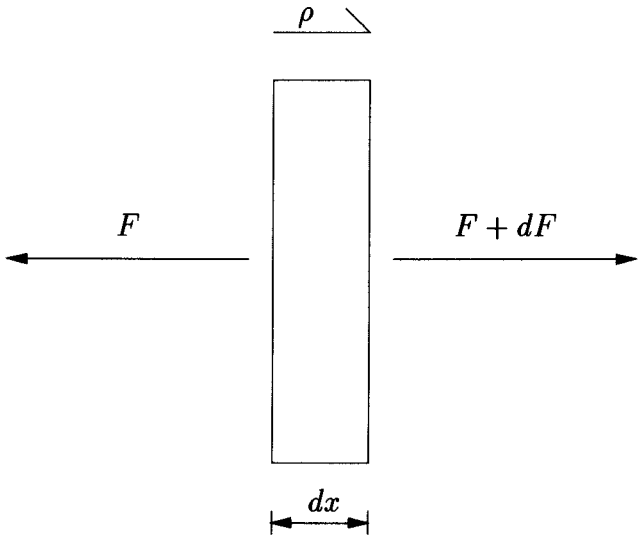


Figure A.3: Equilibrium of an infinitesimal bar element

$$\rho + F_{,1} = 0 \tag{A.11}$$

The strain-displacement relationship for this problem can be written as

$$\epsilon_{11} = \frac{du}{dx_1} = u_{,1} \tag{A.12}$$

The constitutive law (material law) for this problem is obtained as

$$\begin{aligned} \sigma &= E(\epsilon_{11} - \alpha \triangle T) \\ F &= A\sigma = AE(\epsilon_{11} - \alpha \triangle T) \end{aligned} \tag{A.13}$$

The boundary conditions that are applied to the problem are described by

$$u(0) = 0 \tag{A.14}$$

$$F(x_1 = l) = P \tag{A.15}$$

(A.14) is the so-called *essential boundary condition.*

(A.15) is the so-called *natural boundary condition.* It is important to make this distinction since their implementation into finite element equations are different. To define the variational form of the equilibrium equations, we need to characterize two classes of functions [55].

1. The candidate or trial solutions to u
   a) These solutions must satisfy the essential boundary conditions [55].
   b) They must also satisfy $\int_0^l u_{,\alpha}u_{,\alpha}dx_1 < \infty$; $\|\alpha\| \leq m$. (We say $u \in H^m$ where H^m is a *Sobolev* space [55].) In this case $m = 1$ [55].
2. Weighting functions, test functions or variation funtions, U [55].
   a) We require U to satisfy the homogeneous part of the essential boundary conditions [55]. $U \in H_0^m$, where the 0 subscript refers to the *homogeneous* part [55].

The boundary value problem as defined above is known as the strong form of the boundary value problem. In the finite element displacement method, the equilibrium equation can be written in its variational form as (A.16) where U are the weighting functions

$$\int_0^l U(\rho + F_{,1})dx = 0 \tag{A.16}$$

(A.16) corresponds to weighting the error or residual over the domain with respect to a weighting function U which may be arbitrary. The solution of this form is identical to the solution of the equilibrium equation if $U(x_1)$ is allowed to be arbitrary with the constraint that U is required to satisfy the homogeneous part of the essential boundary conditions (Point 2a above) [55].

The weak form of the boundary value problem

Integration by parts of (A.16), to reduce the order of the derivatives of u

$$\int_0^l U(\rho + F_{,1})dx_1 = FU|_0^l - \int_0^l FU_{,1}dx_1 + \int_0^l U\rho dx_1 = 0 \tag{A.17}$$

Now we need to satisfy exactly

$$\left.\begin{array}{l}\text{Strain-displacement}\\ \text{Constitutive law}\\ \text{Boundary conditions: } F(l) = P\end{array}\right\} F = AEu_{,1} - AE\alpha\Delta T \tag{A.18}$$

This is called the *variational equation* which is equivalent to the equation of virtual work. The solution is the weak or generalized solution. $U(x_i)$ is the same as the virtual displacement and is sometimes also written in the form $\delta u(x_1)$. The weak form is also called the

variational boundary value problem. The above formulation leads to a *symmetric* form and to a weakening of the smoothness requirements on u since we are only working with the first derivative [55].

The Galerkin method: discretization

Our first step in the direction of a solution method is to construct finite dimensional approximations to the trial solutions u and the weighting functions U. u^h is denoted as a member of the collection of functions used to approximate u and U^h as a member of the functions used to approximate U. In fact u^h is an approximate solution to the weak form [55].

To approximate the trial functions we use a set of basis functions ϕ^k, $k = 1 \ldots N$ and we obtain

$$u^h = \phi^k q^k \qquad \text{sum on } k \tag{A.19}$$

(Note: k is not a power.) The trial function is therefore made up of the accumulation of chosen basis functions multiplied by unknown coefficients, q^k, $k = 1 \ldots N$ [55].

There are various ways in which the weighting function can be chosen, but we will use the *Bubnov-Galerkin* method which assumes the same basis functions as are used for the trial solutions [55]. Hence we obtain

$$U^h = \phi^k Q^k \tag{A.20}$$

We can also write both equations in matrix form

$$u^h = \underbrace{N}_{1\times N}\underbrace{q}_{N\times 1} \qquad U^h \underbrace{N}_{1\times N}\underbrace{Q}_{N\times 1} \tag{A.21}$$

These are substituted into the weak form to give an approximate solution in

$$\int_0^l U^h_{,1} A E u^h_{,1} dx_1 = \int_0^l U^h p(x_1) dx_1 + \int_0^l AE\alpha\triangle T U^h_{,1} dx_1 \tag{A.22}$$

(A.22) is sometimes referred to as the Galerkin equation. Approximate methods of the type considered are examples of so-called weighted residual methods. We are in fact weighting the residual of the equilibrium equations $\rho + F_{,1}$. By substitution of U^h and u^h, (A.22) becomes (A.23) [55].

$$Q^T \int_0^l N^T_{,1} AE N_{,1} dx_1 = Q^T N_l^T P + Q^T \int_0^l N^T \rho(x_1) dx_1 + Q^T \int_0^l N^T_{,1} AE\alpha\triangle T dx_1 \tag{A.23}$$

Since U^h can be arbitrary, Q can also be arbitrary and (A.23) becomes (A.24),

$$\int_0^l N^T_{,1} AE N_{,1} dx_1 q = N_l^T P + \int_0^l N^T \rho dx_1 + \int_0^l N^T_{,1} AE\alpha\triangle T dx_1 \tag{A.24}$$

or

$$[\int_0^l N_{,1}^T AEN_{,1}dx_1]q = N_l^T P + \int_0^l N^T \rho dx_1 + \int_0^l N_{,1}^T AE\alpha\triangle T dx_1 \tag{A.25}$$

or

$$\mathbf{Kq} = \mathbf{R_f} + \mathbf{R_p} + \mathbf{R_t} \tag{A.26}$$

$\mathbf{K}$ designates the stiffness matrix $(N \times N)$
$\mathbf{q}$ designates the unknown displacement coefficients $(N \times 1)$
$\mathbf{R_F}$ designates the generalized or equivalent point loads $(N \times 1)$
$\mathbf{R_p}$ designates the generalized or equivalent distributed loads $(N \times 1)$
$\mathbf{R_j}$ designates the generalized or equivalent thermal loads $(N \times 1)$

The *Bubnov-Galerkin method* leads to symmetry of the stiffness matrix and the requirement of only one set of basis functions [55]. The only task remaining to find u^h is the solution of the generalized equilibrium equations of (A.26).

A.2 Finite difference method

A.2.1 Introduction

The finite difference method (FDM) is a very general way of solving PDEs for all types of problems. Smith [28] deals with this in great detail. Since the structural problems under consideration are governed by the Laplace equations, (A.2), we will only consider finite difference methods for solving this PDE. It is valuable to be familiar with this approach because such knowledge will reinforce our understanding of finite element procedures [56]. The approach is also very similar to approaches that could be used for CA.

A.2.2 Methodology

If we divide a function $y(x)$ into equally distant spaces δ so that we obtain discrete values $y_1, y_2, y_3 \ldots$ for $x = 0, x = \delta, x = 2\delta \ldots$, we can approximate the first derivatives of $y(x)$ at corresponding points

$$\left(\frac{dy}{dx}\right)_{x=0} \approx \frac{y_1 - y_0}{\delta} \qquad \left(\frac{dy}{dx}\right)_{x=\delta} \approx \frac{y_2 - y_1}{\delta} \tag{A.27}$$

In the same manner we can obtain the second derivatives as shown

$$\left(\frac{d^2y}{dx^2}\right)_{x=\delta} \approx \left(\frac{dy}{dx}\right)_{x=\delta} - \left(\frac{dy}{dx}\right)_{x=0} = \frac{y_2 - 2y_1 + y_0}{\delta^2} \tag{A.28}$$

Suppose we have a smooth function $f(x,y)$ we can use approximate solutions to partial derivatives similar to (A.27) and (A.28) for the points specified in Figure A.4.

$$\frac{\partial f}{\partial x} \approx \frac{f_1 - f_0}{\delta} \qquad \frac{\partial f}{\partial y} \approx \frac{f_2 - f_0}{\delta}$$
$$\frac{\partial^2 f}{\partial x^2} \approx \frac{f_1 - 2f_0 + f_3}{\delta^2} \qquad \frac{\partial^2 f}{\partial x^2} \approx \frac{f_1 - 2f_0 + f_4}{\delta^2} \tag{A.29}$$

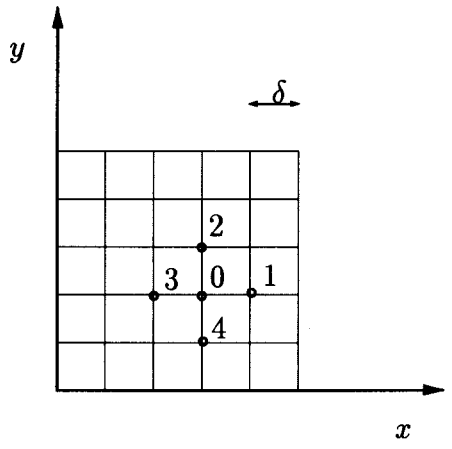


Figure A.4: Finite difference two-dimensional grid

To gain better insight into how this can be used, we can consider the shear stresses within a long uniform cylinder in torsion. The governing PDE is shown to reduce to

$$\frac{\partial^2 \phi}{\partial x^2} + \frac{\partial^2 \phi}{\partial y^2} = -2\mu\theta \tag{A.30}$$

ϕ is the stress function which must be constant along the boundary of the cross section, θ is the angle of twist per unit length of the bar, and μ is the modulus of shear [53].Using (A.29), we can transform (A.30) into a finite difference equation

$$\frac{1}{\delta^2}(\phi_1 + \phi_2 + \phi_3 + \phi_4 - 4\phi_0) = -2\mu\theta \tag{A.31}$$

In this way every torsional problem reduces to a set of numerical values of the stress function ϕ which satisfy (A.31) at every nodal point within the boundary of the cross section and become constant along the boundary [53]. As the simplest example, a bar of square cross section $a \times a$ will be considered and a square net with a mesh side $\delta = \frac{1}{4}a$ will be used as shown in Figure A.5.

From symmetry we can conclude that it is sufficient to consider only one eighth of the cross section shown in Figure A.5 [53]. If we determine values for the stress function ϕ at points α, β and γ we shall know ϕ at all nodal points on the net within the boundary [53]. We can now rewrite (A.31) as (A.32) by using symmetry.

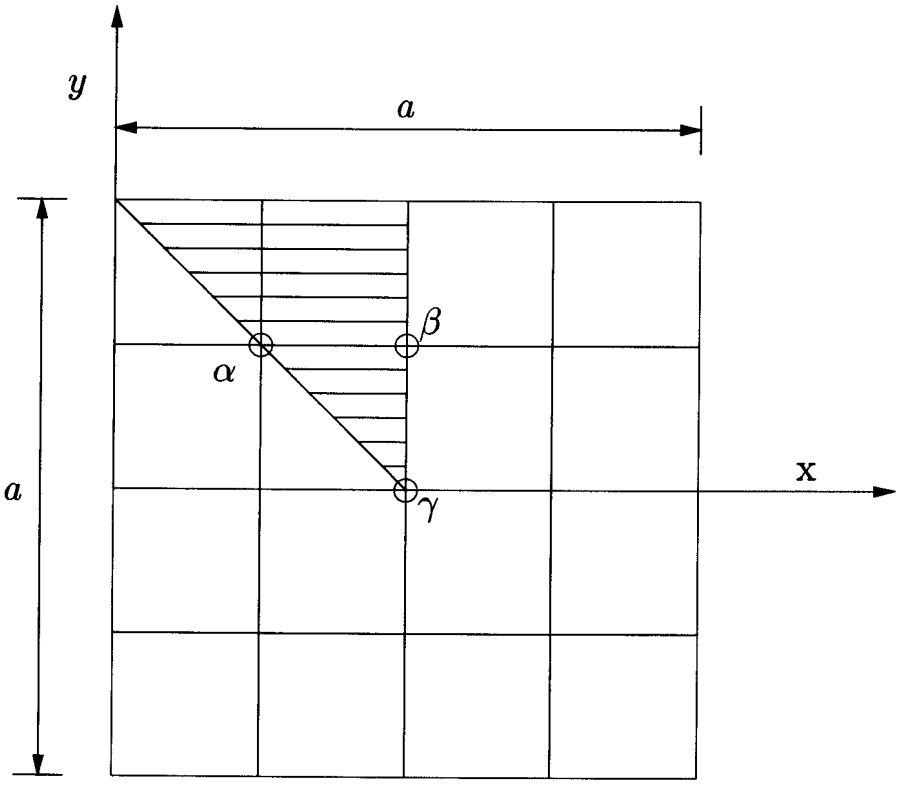


Figure A.5: Finite difference torsion bar example cross section

$$\begin{aligned} 2\beta - 4\alpha &= -2\mu\theta\delta^2 \\ 2\alpha + \gamma - 4\beta &= -2\mu\theta\delta^2 \\ 4\beta - 4\gamma &= -2\mu\theta\delta^2 \end{aligned} \tag{A.32}$$

(A.32) can now be solved to obtain $\alpha = 1.375\mu\theta\delta^2$, $\beta = 1.750\mu\theta\delta^2$ and $\gamma = 2.250\mu\theta\delta^2$ [53]. The required stress function is thus determined by these numerical values at all nodal points within the boundary and by zero values at the boundary [53].

Methods of solution belong essentially to either a class of direct methods or a class of iterative methods [28]. Direct methods solve the system of equations in a known number of arithmetic operations and errors in the solution arise entirely from rounding errors introduced during computation [28]. These direct methods are elimination methods of which the best known examples are the Gaussian elimination method and the triangular decomposition method which factorizes the matrix $\mathbf{A}$ of the equation $\mathbf{Ax} = \mathbf{B}$ into $\mathbf{A} = \mathbf{LU}$, where $\mathbf{L}$ and $\mathbf{U}$ are upper and lower triangular matrices respectively [28]. In the latter method, once the decomposition has been determined, the solution is calculated from $\mathbf{LUx} = \mathbf{B}$ by setting $\mathbf{Ux} = \mathbf{y}$ and then solving $\mathbf{Ly} = \mathbf{b}$ for $\mathbf{y}$ by forward substitution and $\mathbf{Ux} = \mathbf{y}$ for $\mathbf{x}$ by back substitution [28]. With both methods it is usually necessary to employ partial pivoting with scaling to control the growth of rounding errors [28].

A.2.3 Methods of successive approximations

To increase the accuracy of the solution, it is necessary to refine the mesh used. But the number of equations that must be solved becomes larger and larger. The solution can be greatly simplified by using methods of successive approximations [53]. An iterative method for solving equations is one in which a first approximation is used to calculate a second approximation which is in turn used to calculate a third and so on. The iterative procedure is said to be convergent when the differences between the exact solution and the successive approximation tend to zero as the number of iterations increase. In general, an exact solution is never obtained in a finite number of steps, but this does not matter. What is important is that the successive approximation converges very rapidly to values that are correct to a specific accuracy. One would consider using iterative methods when a direct method requires faster computer storage space than is available and the matrix coefficients are sparse but well conditioned. This situation often arises with the difference equations approximating elliptic boundary value problems [28], as is the case with Laplace's equation (A.1) governing elastostatic problems.

To illustrate this, we return to the two-dimensional case with Poisson's equation, (A.2). The corresponding finite difference equation is given by.

$$\frac{1}{4}(\phi_1 + \phi_2 + \phi_3 + \phi_4) = \phi_0 \tag{A.33}$$

(A.33) implies that the approximate value of the function ϕ at the nodal point 0 of the square net is equal to the average value of the function at the four adjacent nodal points. We suppose that for a simple system we obtain the system of equations in (A.34) where $a_{ii} \neq 0$ for $i = 1 \ldots 4$

$$\begin{aligned}
a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + a_{14}x_4 &= b_1 \\
a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + a_{24}x_4 &= b_2 \\
a_{31}x_1 + a_{32}x_2 + a_{33}x_3 + a_{34}x_4 &= b_3 \\
a_{41}x_1 + a_{42}x_2 + a_{43}x_3 + a_{44}x_4 &= b_4
\end{aligned} \tag{A.34}$$

Rewriting (A.34) in the form of the unknowns gives

$$\begin{aligned}
x_1 &= \frac{1}{a_{11}}(b_1 - a_{12}x_2 - a_{13}x_3 - a_{14}x_4) \\
x_2 &= \frac{1}{a_{22}}(b_2 - a_{21}x_1 - a_{23}x_3 - a_{24}x_4) \\
x_3 &= \frac{1}{a_{33}}(b_3 - a_{31}x_1 - a_{32}x_2 - a_{43}x_4) \\
x_4 &= \frac{1}{a_{44}}(b_4 - a_{41}x_1 - a_{42}x_2 - a_{43}x_3)
\end{aligned} \tag{A.35}$$

Jacobi method

We denote our first approximation to x_i by $x_i^{(1)}$ and the second by $x_i^{(2)}$ etc., and assume that n of them have been calculated, i.e. $x_i^{(n)}$ is known for $i = 1 \ldots 4$. The Jacobi iterative method expresses the $(n+1)$th iterative values exclusively in terms of the nth iterative values and the iterations corresponding to (A.35) are defined by (A.36).

$$\begin{aligned}
x_1^{(n+1)} &= \frac{1}{a_{11}}\left(b_1 - a_{12}x_2^{(n)} - a_{13}x_3^{(n)} - a_{14}x_4^{(n)}\right) \\
x_2^{(n+1)} &= \frac{1}{a_{22}}\left(b_2 - a_{21}x_1^{(n)} - a_{23}x_3^{(n)} - a_{24}x_4^{(n)}\right) \\
x_3^{(n+1)} &= \frac{1}{a_{33}}\left(b_3 - a_{31}x_1^{(n)} - a_{32}x_2^{(n)} - a_{43}x_4^{(n)}\right) \\
x_4^{(n+1)} &= \frac{1}{a_{44}}\left(b_4 - a_{41}x_1^{(n)} - a_{42}x_2^{(n)} - a_{43}x_3^{(n)}\right)
\end{aligned} \tag{A.36}$$

Gauss-Seidel method

In this method the $(n+1)$th iterative values are used as soon as they become available. Iterations corresponding to (A.35) are defined by (A.37)

$$\begin{aligned}
x_1^{(n+1)} &= \frac{1}{a_{11}}\left(b_1 - a_{12}x_2^{(n)} - a_{13}x_3^{(n)} - a_{14}x_4^{(n)}\right) \\
x_2^{(n+1)} &= \frac{1}{a_{22}}\left(b_2 - a_{21}x_1^{(n+1)} - a_{23}x_3^{(n)} - a_{24}x_4^{(n)}\right) \\
x_3^{(n+1)} &= \frac{1}{a_{33}}\left(b_3 - a_{31}x_1^{(n+1)} - a_{32}x_2^{(n+1)} - a_{34}x_4^{(n)}\right) \\
x_4^{(n+1)} &= \frac{1}{a_{44}}\left(b_4 - a_{41}x_1^{(n+1)} - a_{42}x_2^{(n+1)} - a_{43}x_3^{(n+1)}\right)
\end{aligned} \tag{A.37}$$

This increases the rate at which the solution converges, which means the number of iterations required to obtain a solution is reduced.

Successive over-relaxation method

If $x_i^{(n)}$ is added to and subtracted from the right side of the Gauss-Seidel equation, (A.37) is rewritten as

$$\begin{aligned}
x_1^{(n+1)} &= x_1^{(n)} + \left[\frac{1}{a_{11}}\left(b_1 - a_{11}x_1^{(n)} - a_{12}x_2^{(n)} - a_{13}x_3^{(n)} - a_{14}x_4^{(n)}\right)\right] \\
x_2^{(n+1)} &= x_2^{(n)} + \left[\frac{1}{a_{22}}\left(b_2 - a_{21}x_1^{(n+1)} - a_{22}x_2^{(n)} - a_{23}x_3^{(n)} - a_{24}x_4^{(n)}\right)\right]
\end{aligned}$$

$$
\begin{aligned}
x_3^{(n+1)} &= x_3^{(n)} + \left[\frac{1}{a_{33}}\left(b_3 - a_{31}x_1^{(n+1)} - a_{32}x_2^{(n+1)} - a_{33}x_3^{(n)} - a_{34}x_4^{(n)}\right)\right] \\
x_4^{(n+1)} &= x_4^{(n)} + \left[\frac{1}{a_{44}}\left(b_4 - a_{41}x_1^{(n+1)} - a_{42}x_2^{(n+1)} - a_{43}x_3^{(n+1)} - a_{44}x_4^{(n)}\right)\right]
\end{aligned} \tag{A.38}
$$

It is seen that the expressions in the square brackets are the corrections or changes made to $x_i^{(n)}$, $i = 1 \ldots 4$, by one Gauss-Seidel iteration [28]. If successive corrections are all one-signed as they usually are for the approximating difference equation of elliptic problems, it would be reasonable to expect convergence to be accelerated if each equation of (A.38) were given a larger correction term than is defined by (A.38) [28]. This idea leads to the successive over-relaxation (SOR) which is defined by (A.39).

$$
\begin{aligned}
x_1^{(n+1)} &= x_1^{(n)} + \left[\frac{\omega}{a_{11}}\left(b_1 - a_{11}x_1^{(n)} - a_{12}x_2^{(n)} - a_{13}x_3^{(n)} - a_{14}x_4^{(n)}\right)\right] \\
x_2^{(n+1)} &= x_2^{(n)} + \left[\frac{\omega}{a_{22}}\left(b_2 - a_{21}x_1^{(n+1)} - a_{22}x_2^{(n)} - a_{23}x_3^{(n)} - a_{24}x_4^{(n)}\right)\right] \\
x_3^{(n+1)} &= x_3^{(n)} + \left[\frac{\omega}{a_{33}}\left(b_3 - a_{31}x_1^{(n+1)} - a_{32}x_2^{(n+1)} - a_{33}x_3^{(n)} - a_{34}x_4^{(n)}\right)\right] \\
x_4^{(n+1)} &= x_4^{(n)} + \left[\frac{\omega}{a_{44}}\left(b_4 - a_{41}x_1^{(n+1)} - a_{42}x_2^{(n+1)} - a_{43}x_3^{(n+1)} - a_{44}x_4^{(n)}\right)\right]
\end{aligned} \tag{A.39}
$$

The factor ω, called the acceleration parameter or relaxation factor, generally lies in the range $1 < \omega < 2$ [28]. The determination of the optimum value of ω for maximum rate of convergence is discussed in [28] but is beyond the scope of this research.

A.2.4 Practicality for structural analysis

In the problem of the torsion bar in Figure A.5 the boundary conditions are specified directly in the form of the stress function. This is, however, not possible with most problems since the boundary conditions applied are specified displacements (*essential boundary conditions*) and forces applied (*natural boundary conditions*). For a general problem it is possible to obtain either displacements or stresses where the boundary conditions are applied. This fact makes finite differences inapplicable in common problems. Table A.1 summarizes some widely used finite difference approximations, also called finite difference stencils or molecules [56]. These stencils are derived from the partial derivatives to be used for successive approximations.

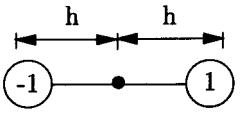
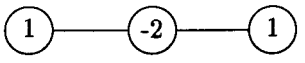
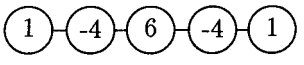
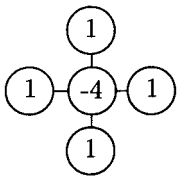
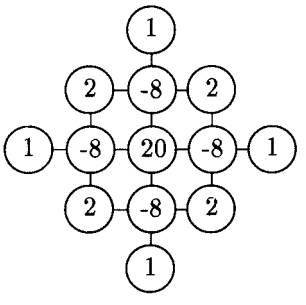
Differentiation	Finite difference approximation	Molecules
$\frac{dw}{dx}\|_i$	$\frac{w_{i+1}-w_{i-1}}{2h}$	
$\frac{d^2w}{dx^2}\|_i$	$\frac{w_{i+1}-2w_i+w_{i-1}}{h^2}$	
$\frac{d^3w}{dx^3}\|_i$	$\frac{w_{i+2}-2w_{i+1}+2w_{i-1}-w_{i-2}}{2h^3}$	
$\frac{d^4w}{dx^4}\|_i$	$\frac{w_{i+2}-4w_{i+1}+6w_i-4w_{i-1}+w_{i-2}}{h^4}$	
$\nabla^2 w\|_{i,j}$	$\frac{-4w_{i,j}+w_{i+1,j}+w_{i,j+1}+w_{i-1,j}+w_{i,j-1}}{h^2}$	
$\nabla^4 w\|_{i,j}$	$[20w_{i,j} - 8(w_{i+1,j} + w_{i-1,j} + w_{i,j+1} + w_{i,j-1}) + 2(w_{i+1,j+1} + w_{i-1,j+1} + w_{i-1,j+1} + w_{i+1,j-1}) + w_{i+2,j} + w_{i-2,j} + w_{i,j+2} + w_{i,j-2}]/h^4$	

Table A.1: Established finite difference computational molecules

A.3 Finite element method

A.3.1 Introduction

In a 1941 mathematics lecture, published in 1943, Courant suggested piece-wise polynomial interpolation over triangular sub-regions as a way to obtain approximate numerical solutions. He recognized this approach as a Rayleigh-Ritz solution of a variation problem. This is the finite element method (FEM) as we know it today. Courant's work was forgotten until engineers had independently developed it [55].

None of the preceding work was practical at the time because there were no computers to perform the calculations. By 1953 engineers were writing stiffness equations in matrix notation and solving the equations with digital computers. The classical paper *Stiffness and deflection analysis of complex structures* by Turner, Clough and Topp appeared in 1956. With this paper and others, explosive development of FEM in engineering began. The name "finite element" was coined in 1960. By 1963 the method was recognized as rigorously sound, and it became a respected area of study for academics. As late as 1967, engineers and mathematicians worked with finite elements in apparent ignorance of one another.

Currently, finite elements are the most generally used method in structural analysis. FEM has the capability of dealing with both linear and non-linear problems and can be constructed using various types of formulations.

There are various ways of introducing the FEM. It can be introduced rather physically, as is done with the displacement formulation as found in most introductory courses. The approach works well for teaching basics but is somewhat superficial and does not allow the reader to think of new possibilities, an attitude that has to be developed in further analyses. One can also proceed along purely mathematical lines, the approach taken by mathematicians. For the engineer, a purely mathematical course should serve to broaden his or her knowledge base and not as an introduction to finite element analysis [55].

The vast amount of finite element formulations available designed for specific tasks makes it impossible to describe a full definition of finite elements in this document. Instead, the approach will be to simply explain enough of the method to allow us to analyse the problems with which the CA analysis will be compared. The formulation will adopt a mathematical point of view but with practical implementation since it is an existing method that is most often used to solve the governing equation in structural analyses. This also allows for comparison to boundary element methods which consist of a more complicated formulation and implementation. Finite elements are probably by far the most used and the most well-known method in structural analyses and should thus be considered as an extremely suitable standard for comparison.

A.3.2 Methodology

One dimension

In the FEM domain of a bar as discussed in section A.1.3, the bar is divided into intervals and the displacements field is assumed to be constructed from piece-wise linear functions. Some degree of continuity is imposed at the boundaries between the pieces or elements. The degree of continuity in the field variable must be the same as one order lower than its differential order in the weak form [55].

If we use the same notation as in section A.1.3, we can divide the bar into $n-1$ intervals.

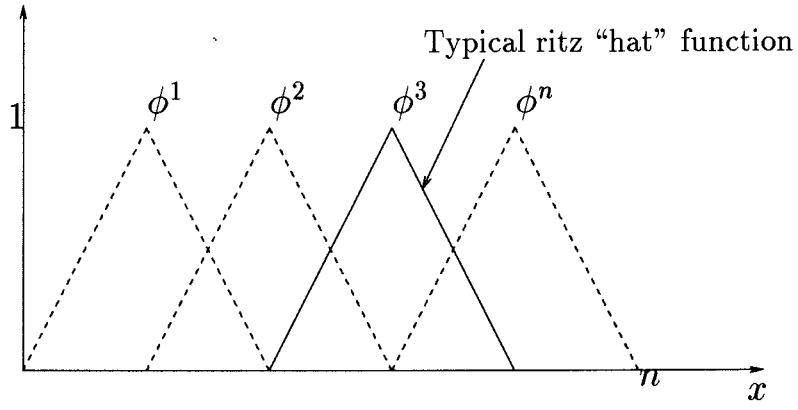


Figure A.6: Typical ritz function used in the analysis of one-dimensional bar problem

A piece-wise linear function can be made up as the linear combination of the Ritz functions ϕ shown in Figure A.7. The ϕ^k shown in Figure A.6 are functions that equal 1 at a particular node k and vanish at all others [55].

$$u = \phi^k q^k \qquad k = 1, 2, 3 \ldots n \tag{A.40}$$

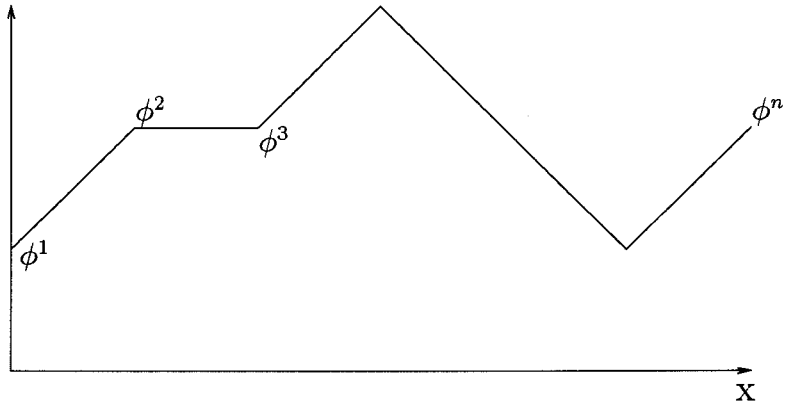


Figure A.7: Linear combination of the ritz functions

We notice that ϕ^k forms a local basis for the displacement variable as it is identically zero except in elements adjoining node k. Only the adjacent elements will be coupled. The

piece-wise linear finite element functions are the most widely used finite element functions for one-dimensional models [55].

We now consider the one-dimensional bar problem shown in Figure A.2. We consider the solution to this problem to be obtained using two finite elements as shown in Figure A.8.

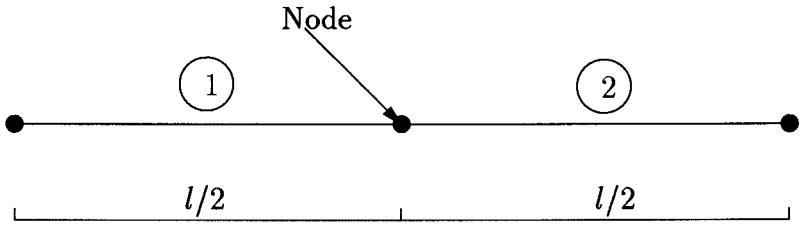


Figure A.8: Simple bar FEM mesh

Since the bar is constrained at the left hand side, the first shape function ϕ^1 will not influence the solution since the displacement is specified to be zero in that point. The second ϕ^2 and third ϕ^3 shape functions are defined by

$$\phi^2 = \left\{ \begin{array}{ll} \frac{2x}{l} & 0 < x < \frac{l}{2} \\ 2 - \frac{2x}{l} & \frac{l}{2} < x < l \end{array} \right\}$$
$$\phi^3 = \left\{ \begin{array}{ll} \frac{2x}{l} - 1 & \frac{l}{2} < x < l \\ 0 & 0 < x < \frac{l}{2} \end{array} \right\} \tag{A.41}$$

To solve the displacement u we can rewrite (A.40) as

$$u = \phi^2 q^2 + \phi^3 q^3 = [\phi^2 \phi^3] \left\{ \begin{array}{c} q^2 \\ q^3 \end{array} \right\} \tag{A.42}$$

The strains at the represented nodes are defined by

$$\epsilon_{11} = u_{,1} = [\phi^2_{,1} \phi^3_{,1}] \left\{ \begin{array}{c} q^2 \\ q^3 \end{array} \right\} \tag{A.43}$$

To obtain the strain, we need to find the partial derivatives of our shape function as defined by (A.41). The derivatives are obtained as

$$\phi^2_{,1} = \left\{ \begin{array}{ll} \frac{2}{l} & 0 < x < \frac{l}{2} \\ -\frac{2}{l} & \frac{l}{2} < x < l \end{array} \right\}$$
$$\phi^3_{,1} = \left\{ \begin{array}{ll} \frac{2}{l} & \frac{l}{2} < x < l \\ 0 & 0 < x < \frac{l}{2} \end{array} \right\} \tag{A.44}$$

We use this known function in the weak form of our governing equation, (A.25). The evaluation of the left hand side of (A.26) can be shown to be

$$
\begin{aligned}
& AE\int_0^l \begin{Bmatrix} \phi^2_{,1} \\ \phi^3_{,1} \end{Bmatrix} [\phi^2_{,1}\phi^3_{,1}]dx \\
&= AE \begin{bmatrix} \left(\underbrace{\int_0^{\frac{l}{2}} \frac{2}{l}\frac{2}{l}dx}_{\text{Element 1}} + \underbrace{\int_{\frac{l}{2}}^{l} -\frac{2}{l}-\frac{2}{l}dx}_{\text{Element 2}}\right) & \left(\underbrace{\int_0^{\frac{l}{2}} \frac{2}{l}0}_{\text{Element 1}} + \underbrace{\int_{\frac{l}{2}}^{l} -\frac{2}{l}\frac{2}{l}dx}_{\text{Element 2}}\right) \\ \left(\underbrace{\int_0^{\frac{l}{2}} \frac{2}{l}0dx}_{\text{Element 1}} + \underbrace{\int_{\frac{l}{2}}^{l} -\frac{2}{l}\frac{2}{l}dx}_{\text{Element 2}}\right) & \left(\underbrace{\int_0^{\frac{l}{2}} 00dx}_{\text{Element 1}} + \underbrace{\int_{\frac{l}{2}}^{l} \frac{2}{l}\frac{2}{l}dx}_{\text{Element 2}}\right) \end{bmatrix} \\
&= AE \begin{bmatrix} \left(\frac{4}{l^2}\frac{l}{2}+\frac{4}{l^2}\frac{l}{2}\right) & \left(0-\frac{4}{l^2}\frac{l}{2}\right) \\ \left(0-\frac{4}{l^2}\frac{l}{2}\right) & \left(0+\frac{4}{l^2}\frac{l}{2}\right) \end{bmatrix} \\
&= \frac{AE}{l}\begin{bmatrix} 4 & -2 \\ -2 & 2 \end{bmatrix}
\end{aligned} \tag{A.45}
$$

Now we can consider the solution to the distributed load as set out in (A.26) as

$$
\begin{aligned}
& \rho\int_0^l \begin{Bmatrix} \phi^2 \\ \phi^3 \end{Bmatrix} dx \\
&= \rho \left[\underbrace{\int_0^{\frac{l}{2}} \begin{Bmatrix} \frac{2x}{l} \\ 0 \end{Bmatrix} dx}_{\text{Element 1}} + \underbrace{\int_{\frac{l}{2}}^{l} \begin{Bmatrix} 2-\frac{2x}{l} \\ \frac{2x}{l}-1 \end{Bmatrix} dx}_{\text{Element 2}} \right] \\
&= \rho \begin{bmatrix} \frac{x^2}{l}|_0^{\frac{l}{2}} & + & 2x-\frac{x^2}{l}|_{\frac{l}{2}}^{l} \\ 0 & & \frac{x^2}{l}-x|_{\frac{l}{2}}^{l} \end{bmatrix} \\
&= \rho \begin{bmatrix} \frac{l}{4} & + & l-\frac{3l}{4} \\ 0 & & \frac{3l}{4}-\frac{l}{2} \end{bmatrix} \\
&= \rho \begin{bmatrix} \frac{l}{2} \\ \frac{l}{4} \end{bmatrix}
\end{aligned} \tag{A.46}
$$

The nodal load P is only applied at the third node. It can be added to the right hand side as

$$
P\begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{A.47}
$$

We can now write out our solution which is a combination of (A.45), (A.46) and (A.47) as

$$\frac{AE}{l}\begin{bmatrix} 4 & -2 \\ -2 & 2 \end{bmatrix}\begin{Bmatrix} q^2 \\ q^3 \end{Bmatrix} = \rho\begin{bmatrix} \frac{l}{2} \\ \frac{l}{4} \end{bmatrix} + P\begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{A.48}$$

With ρ and P known, the only unknowns to solve in this equation are the displacements q^2 and q^3. Once the displacements are known, we use (A.10) and (A.43) to calculate the stress as

$$\sigma = E[\phi^2_{,1}\phi^3_{,1}]\begin{Bmatrix} q^2 \\ q^3 \end{Bmatrix} \tag{A.49}$$

Two dimensions

We will now consider a two-dimensional problem under plane stress as described in section A.1.3 and its governing equations described by (A.3), (A.5) and (A.9). In Figure A.9 we define a rectangular element used to model the membrane.

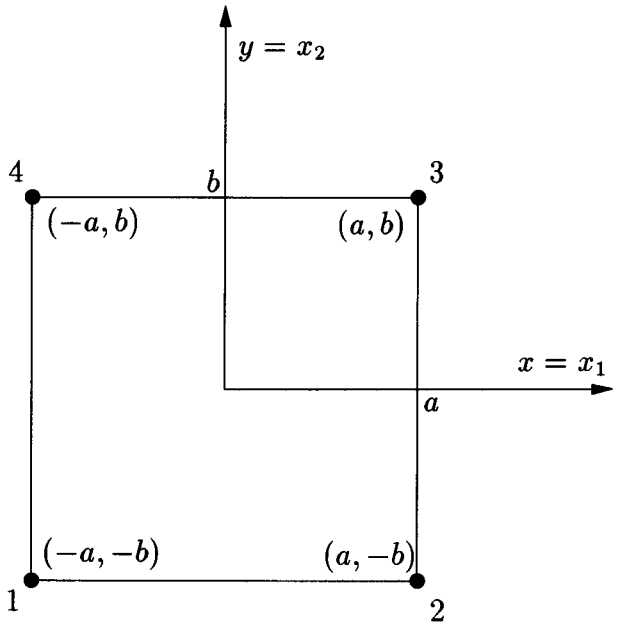


Figure A.9: Simple membrane FEM element

The displacements of each node can be approximated in a similar fashion as with our one-dimensional element. We can therefore rewrite (A.40) into a form where i denotes the node under consideration as

$$u_i = \phi^k q_i^k \qquad k = 1, \ldots 4 \tag{A.50}$$

A node is defined at each corner and similar to the ritz hat function in our one-dimensional problem, we want a shape function ϕ such that it has a value of 1 at the node under consideration and zero at all other nodes. These functions should be linearly independent and

provide continuity of displacement with reference to Figure A.9. The four shape functions can be written as

$$\begin{aligned}
\phi^1 &= \frac{1}{4}\left(1-\frac{x_1}{a}\right)\left(1-\frac{x_2}{a}\right)\\
\phi^2 &= \frac{1}{4}\left(1+\frac{x_1}{a}\right)\left(1-\frac{x_2}{a}\right)\\
\phi^3 &= \frac{1}{4}\left(1+\frac{x_1}{a}\right)\left(1+\frac{x_2}{a}\right)\\
\phi^4 &= \frac{1}{4}\left(1-\frac{x_1}{a}\right)\left(1+\frac{x_2}{a}\right)
\end{aligned} \tag{A.51}$$

Since different finite elements have different shapes, we need to transform the x_1 and x_2 axis to normalized coordinates with axes ζ_1 and ζ_2 where our element is always square in these normalized coordinates. In order to determine the shape functions in these normalized coordinates the values of a and b in A.9 are set equal to one, resulting in the following shape functions.

$$\begin{aligned}
\phi^1 &= \frac{1}{4}(1-\zeta_1)(1-\zeta_2)\\
\phi^2 &= \frac{1}{4}(1+\zeta_1)(1-\zeta_2)\\
\phi^3 &= \frac{1}{4}(1+\zeta_1)(1+\zeta_2)\\
\phi^4 &= \frac{1}{4}(1-\zeta_1)(1+\zeta_2)
\end{aligned} \tag{A.52}$$

Suppose that the finite elements occupy a domain Ω, and are surrounded by a boundary Γ. Suppose too that a distributed load p is applied along the boundary between nodes 1 and 2. We know that the displacements are described by (A.50). We define our finite element formulation from (A.25) as

$$\underbrace{\int_\Omega \mathbf{B}^T\mathbf{C}\mathbf{B}d\Omega}_{\text{Stiffness}}\, q = \underbrace{\int_\Gamma \mathbf{N}^T p^* d\Gamma}_{\text{Consistent Nodal Loads}} \tag{A.53}$$

$\mathbf{C}$ is the constitutive matrix and derived from basic continuum mechanics. $\mathbf{B}$ contains derivatives of the shape functions matrix which are contained in $\mathbf{N}$. As with the one-dimensional case, we will consider the left and right hand side of (A.53) separately. The left hand side is expressed in (A.54) for a two-dimensional membrane element with a thickness t.

$$k^m = \int_A \mathbf{B}^T\mathbf{C}\mathbf{B}dx_1dx_2t \tag{A.54}$$

In (A.54) we reduced our domain Ω to an area A with our two axis (x_1 x_2) as defined in Figure A.9. We can now translate the entire equation to our normalized axis as set out with (A.52) rewriting (A.54) with $dx_1 = ad\zeta_1$, $dx_2 = bd\zeta_2$. This yields

$$k^m = \int_{-1}^{1}\int_{-1}^{1} \mathbf{B}^T\mathbf{C}\mathbf{B}d\zeta_1 d\zeta_2 abt \tag{A.55}$$

We can now move the constants a, b and t in (A.55) out of the integral and use numerical integration to solve the problem as

$$k^m = abt\sum_{i=1}^{m}\sum_{j=1}^{n} \mathbf{B}^T(\zeta_i,\zeta_j)\mathbf{C}\mathbf{B}(\zeta_i,\zeta_j)w_i w_j \tag{A.56}$$

The only aspect of (A.56) that has not been dealt with is the strain-displacement operator $\mathbf{B}$. Consider the displacements of the first two nodes only since this is sufficient to describe this operator. We can write the displacements of the nodes as a function of the shape functions equation. By doing this (A.40) becomes (A.52) in matrix form as

$$\left\{\begin{array}{c} u_1 \\ u_2 \end{array}\right\} = \left[\begin{array}{cccccc} \phi^1 & 0 & \phi_1 & 0 & \phi_1 & 0 \\ 0 & \phi_2 & 0 & \phi_2 & 0 & \phi_2 \end{array}\right] \left\{\begin{array}{c} q_1^1 \\ q_2^1 \\ q_1^2 \\ q_2^2 \\ q_1^3 \\ q_2^3 \\ q_1^4 \\ q_2^4 \end{array}\right\} \tag{A.57}$$

Strain ϵ if difined as the derivative of our displacement equation, (A.12). Thus, we can write the strain as

$$\epsilon = \mathbf{L}u = \underbrace{\mathbf{L}\mathbf{N}}_{\mathbf{B}} q = \left\{\begin{array}{cc} \frac{\partial}{\partial x_1} & 0 \\ 0 & \frac{\partial}{\partial x_2} \\ \frac{\partial}{\partial x_1} & \frac{\partial}{\partial x_2} \end{array}\right\} \left[\begin{array}{cccccccc} \phi^1 & 0 & \phi^2 & 0 & \phi^3 & 0 & \phi^4 & 0 \\ 0 & \phi^1 & 0 & \phi^2 & 0 & \phi^3 & 0 & \phi^4 \end{array}\right] q \tag{A.58}$$

The strain-displacement operator $\mathbf{B}$ can be written as

$$\mathbf{B} = \left[\begin{array}{cccccccc} \phi^1_{,1} & 0 & \phi^2_{,1} & 0 & \phi^3_{,1} & 0 & \phi^4_{,1} & 0 \\ 0 & \phi^2_{,2} & 0 & \phi^2_{,2} & 0 & \phi^3_{,2} & 0 & \phi^4_{,2} \\ \phi^1_{,2} & \phi^1_{,1} & \phi^2_{,2} & \phi^2_{,1} & \phi^3_{,2} & \phi^3_{,1} & \phi^4_{,2} & \phi^4_{,1} \end{array}\right] \tag{A.59}$$

Where $\phi_{,1} = \frac{\partial\phi}{\partial x_1}$ and $\phi_{,2} = \frac{\partial\phi}{\partial x_2}$, the strain-displacement operator $\mathbf{B}$ can be written in the normalized coordinates by making use of the relation

$$\frac{\partial\phi}{\partial\zeta_\alpha} = \frac{\partial\phi}{\partial x_i}\frac{\partial x_i}{\partial\zeta_\alpha} \tag{A.60}$$

In a more general form, (A.60) can be written as

$$\begin{aligned} & & \phi_{,\alpha} &= \phi_{,i}\mathbf{J}_{\alpha i} \\ & \text{or} & \phi_{,\alpha} &= \mathbf{J}\phi_{,i} \\ \text{where} & & \mathbf{J} &= \begin{bmatrix} \frac{\partial x_1}{\partial \zeta_1} & \frac{\partial x_2}{\partial \zeta_1} \\ \frac{\partial x_1}{\partial \zeta_2} & \frac{\partial x_2}{\partial \zeta_2} \end{bmatrix} \end{aligned} \tag{A.61}$$

$\mathbf{J}$ is known as the Jacobian matrix. For the rectangular element in Figure A.9 it is obvious that $\frac{\partial x_1}{\partial \zeta_1} = a$ and $\frac{\partial x_2}{\partial \zeta_2} = b$.

The edge tractions can be calculated in accordance with (A.53) in the normalized coordinates as

$$\underbrace{R^m}_{\text{Consistent nodal load}} = \int_{-1}^{1} \begin{Bmatrix} \phi^1(\zeta_1, \zeta_2) \\ \phi^2(\zeta_1, \zeta_2) \end{Bmatrix} p^* d\zeta_1 d\zeta_2 \tag{A.62}$$

(A.62) results in equivalent nodal loads for the distributed force applied. If the force had any other form besides that of a fixed traction value p^*, the traction would have to be written in the form of the generalized coordinates $p^*(\zeta_1, \zeta_2)$. It is, however, clear that the integral being evaluated in (A.62) is dependent on the form functions. This will give rise to the following system of equations:

$$\mathbf{Kq} = \mathbf{R}$$

or in a more general numerical form

$$\mathbf{AX} = \mathbf{F} \tag{A.63}$$

Thus the finite element method creates a system of equations that solves the displacements at the nodes under consideration. With the displacements known, the strains are known for the problem and from the stress-strain relationship, (A.9) the stresses can be calculated. The strains are the derivatives of the displacements and are thus less accurate than the displacements [55].

In the previous example only one element was discussed. For more than one element, the principles remain the same, the only difference being that most nodes except for those on the boundary will form part of more than one element. Each element is set up in the normalized coordinates and the stiffness is translated to the global coordinates. The stiffness component of each element must be added together to form the global stiffness matrix. The size of the system of equations under consideration is dependent on the number of nodes in the system m, the number of degrees of freedom per node n and the number of nodes restrained by the *essential boundary conditions* e. Thus the total number N in the system to be solved can be calculated as

$$N = (m \times n) - e \tag{A.64}$$

It is clear that to better approximate the governing equations, the finite elements used should decrease in size resulting in more elements. One way in which the accuracy of each element is improved is by using more nodes per element. Thus far we have discussed isoparametric formulation where it is assumed that the interpolation function (shape function) is of the same order as the assumed displacement function. To be able to use higher order elements with such functions, the same principles apply. The shape function must be equal to one in the node under consideration and be zero in all other nodes. If we consider one nine-noded element (the equivalent of four connected four-noded elements) in the normalized local coordinates (ζ_1, ζ_2), the element is still square with a maximum of one and a minimum of minus one. This occurs if we consider the first point at $(1, 1)$ and its corresponding shape function can be set up by logic. If the line exists where $\zeta_1 = -1$, the function is required to be zero so a term $\times(\zeta_1 + 1)$ is added to the equation. This will make all the terms on the line zero for the form function. The next line can be drawn at $\zeta_1 = 0$ (middle nodes on ζ_1). For this a term $\times\zeta_1$ is added. Now on the ζ_2 there are two more that must be set equal to zero where $\zeta_2 = -1$ and where $\zeta_2 = 0$. In order to do this, $\times(\zeta_2 + 1)\zeta_2$ is added. Now the first form function

$$\phi = \frac{a}{b}(\zeta_1 + 1)\zeta_1(\zeta_2 + 1)\zeta_2 \tag{A.65}$$

By substituting the values of $\zeta_1 = 1$ and $\zeta_2 = 1$ into (A.65) which must be equal to one at this point, the fraction $\frac{a}{b} = \frac{1}{4}$ is obtained. The remaining form functions can be constructed in the same manner.

A.3.3 Implementation

Thus far, the implementation of the problem to be solved has been considered using analytical solutions. We use computer programs to solve the finite element equations. Kwon [57] offers a good explanation on how to become acquainted with the implementation. He uses Matlab as the programming language which makes the code easy to change and use. The code provided will be used to calculate the displacements of the single element problem shown in Figure A.10. This problem cannot be considered to be truly representative, but it has to deliver a constant stress in the direction of the applied load.

Using modified code from [57], the displacements as shown in Table A.2 and then the stresses as shown in Table A.3 were calculated.

Node	Displacement x	Displacement y
1	-1.0069e-016	-9.6507e-017
2	5.0000e-001	8.4198e-017
3	5.0000e-001	-1.5000e-001
4	0	-1.5000e-001

Table A.2: Displacements calculated by FEM (single membrane element)

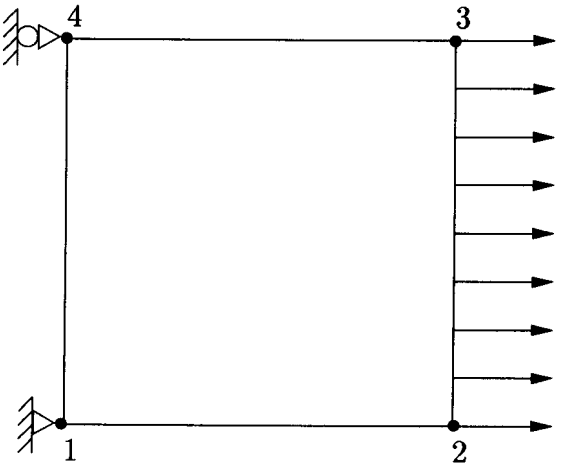


Figure A.10: FEM element patch test

Node	σ_x	σ_y	$\tau_x y$
1	5.0000e+005	1.1642e-010	9.0739e-011
2	5.0000e+005	5.8208e-011	6.9389e-011
3	5.0000e+005	5.8208e-011	6.4051e-011
4	5.0000e+005	0	4.2701e-011

Table A.3: Stresses Calculated by FEM (single membrane element)

A.4 Boundary element method

A.4.1 Introduction

Boundary element methods (BEM) are approximations made only on the boundary or surface of a domain. The approximation is based on a solution of a boundary integral equation. Many important engineering problems can be reduced to mathematical models that belong to a class of problems known as **boundary value problems** [58]. It is easy to understand that for most geometries, boundary elements which only use a mesh on the surface will require less elements than the same structure meshed with finite elements. The method also entails that the dimension of the elements are always one dimension less than the dimension of the defined problem. This entails that for a three-dimensional problem only two-dimensional elements are required to mesh the surface. The discretization is even simpler when using discontinuous elements [54]. The use of elements which sometimes do not meet at corners and are consequently discontinuous in terms of their variables are possible [54]. However, BEM can only be applied if the fundamental solution to the governing partial differential is known. BEM entails two steps in the solving of the problem. The solution is first calculated on the boundary and then internal points are calculated in two separate steps.

A.4.2 Basic principles

Before BEM can be applied to a problem, a fundamental solution must be obtained. This section will handle some basic concepts which will allow us to obtain a fundamental solution later. The fundamental solution to Poisson's equation, (A.2) in two dimensions is known to be

$$u(p,q) = \frac{1}{2\pi} ln(\frac{1}{r}) \tag{A.66}$$

In (A.66), p can be seen as the source point and q the observation point. Each of these two points consists of two Cartesian coordinates which can be used to calculate the radius r in (A.66). Differentiating (A.66) with respect to x at point q we obtain

$$\frac{\partial u}{\partial x} = \frac{x_0 - x}{2\pi r} \tag{A.67}$$

Similarly, an expression can be obtained with respect to y. An approximate solution u of the form in (A.68) is now considered:

$$u = \alpha_1\phi_1 + \alpha_2\phi_2 + \alpha_3\phi_3 + ... \tag{A.68}$$

α_i are unknown coefficients and the ϕ_is are a set of linearly independent functions which are known. In general engineering problems, α_i's are considered as nodal values as they have a clear physical meaning. This is the manner of implementation for finite elements, finite differences and boundary elements. Introducing the approximation of u into the governing differential equation, it is clear that except for the case where enough coefficients and functions are present in (A.68), the approximation produces an error or residual function [54]. The residual function can be defined as

$$R = \nabla^2 u - b \tag{A.69}$$

With approximations, errors also occur in the boundary conditions. The numerical methods used in engineering try to reduce these errors to a minimum by applying different techniques. This reduction is carried out by forcing the errors to be zero at certain points, regions or in a mean sense [54]. This operation can generally be interpreted as *distributing* these errors [54]. Forcing these errors to be zero is generally carried out by weighted residual techniques. If we presume that we have a domain Ω surrounded by a boundary Γ, one can introduce the idea of multiplying (A.2) with an arbitrary weight function ω to obtain

$$\int_\Omega (\nabla^2 u - b) w d\Omega = 0 \tag{A.70}$$

The integration of (A.70) is done by integration of parts. This can be seen in [54]. The result is (A.71) which is known as Green's theorem. Although this theorem is in many cases the starting point for many engineering applications, including boundary element formulations, it is much more illuminating to use the concept of distribution as it illustrates the degree

of continuity required of the function and the importance of the accurate treatment of the boundary conditions [54]:

$$\int_{\Omega}[(\nabla^2 u)w - (\nabla^2 w)u]d\Omega = \int_{\Gamma}(\frac{\partial u}{\partial n}w - u\frac{\partial w}{\partial n})d\Gamma \tag{A.71}$$

The boundary Γ can now be divided into two segments Γ_1 and Γ_2 such that $\Gamma = \Gamma_1 + \Gamma_2$ which surround the domain Ω. We furthermore define a unit vector n so that it has the properties that it exists on every point on the boundary and at each point it is perpendicular to the boundary. We also define the boundary conditions that must be satisfied.

$$\begin{aligned} u &= \bar{u} \quad \text{on} \quad \Gamma_1 \\ q = \frac{\partial u}{\partial n} &= \bar{q} \quad \text{on} \quad \Gamma_2 \end{aligned} \tag{A.72}$$

Rewriting (A.71) to separate the boundary and using the boundary conditions (A.72) we obtain

$$\int_{\Omega}[(\nabla^2 u - b)w]d\Omega - \int_{\Gamma_2}(q-\bar{q})wd\Gamma + \int_{\Gamma_1}(u-\bar{u})\frac{\partial w}{\partial n}d\Gamma = 0 \tag{A.73}$$

This equation shows that one is trying to satisfy a differential equation in the domain Ω plus two types of boundary conditions, the *essential boundary conditions* $u = \bar{u}$ on Γ_1 plus the *natural boundary conditions* $q = \bar{q}$ on Γ_2 [54]. (A.74) defines the residual functions (A.69) on the segmented boundary.

$$\begin{aligned} R_1 &= u - \bar{u} \quad \text{on} \quad \Gamma_1 \\ R_2 &= q - \bar{q} \quad \text{on} \quad \Gamma_2 \end{aligned} \tag{A.74}$$

If we write (A.73) in terms of our residual functions we obtain

$$\int_{\Omega} Rwd\Omega - \int_{\Gamma_2} R_2 wd\Gamma + \int_{\Gamma_1} R_1\frac{\partial w}{\partial n}d\Gamma = 0 \tag{A.75}$$

Considering a special case of this function where the function u exactly defines the *essential boundary conditions*, $u = \bar{u}$ on Γ_1, which means $R_1 = 0$, (A.71) becomes

$$\int_{\Omega} Rwd\Omega = \int_{\Gamma} R_2 wd\Gamma \tag{A.76}$$

Again using integration by parts, (A.77) is obtained.

$$\int_{\Omega}(\nabla^2 w)ud\Omega = -\int_{\Gamma_2}\bar{q}wd\Gamma - \int_{\Gamma_1} qwd\Gamma + \int_{\Gamma_1}\bar{u}\frac{\partial w}{\partial n}d\Gamma + \int_{\Gamma_2} u\frac{\partial w}{\partial n}d\Gamma \tag{A.77}$$

(A.77) can be seen as the starting point for boundary element formulation of the Laplace equation. This equation is also seen as the weak formulation on which it is presumed that the *essential boundary conditions* have been satisfied.

A.4.3 Fundamental solution

Now we will attempt to find a fundamental solution by making use of the principles described in section A.4.2. If (A.9) and (A.5) are substituted into (A.3), the result obtained is the Navier equation in terms of the displacements.

$$(\frac{1}{1-2v})u_{j,jl} + u_{l,jj} + \frac{1}{\mu}b_l = 0 \tag{A.78}$$

μ is the shear modulus which is usually donated by G but in this case G will be used for Galerkin's vector.

$$\mu = \frac{E}{2(1+v)} \tag{A.79}$$

Kelvin's solution of (A.78) when a unit contributed load applied at a point i in the direction of the unit vector e_l is given by

$$b_l = \Delta^i e_l \tag{A.80}$$

An easy way of computing the fundamental solution is by using the representation of the displacements in terms of Galerkin's vector [54]. Galerkin's method implies that the same weight function as the interpolating function is used. Therefore one assumes a vector G from which the displacement components may be obtained as [54].

$$u_j = G_{j,mm} - \frac{1}{2(1-v)}G_{m,jm} \tag{A.81}$$

Substitution of (A.80) and (A.81) into (A.78) gives

$$\nabla^2(\nabla^2 G_l) + \frac{1}{\mu}\Delta^i e_l = 0 \tag{A.82}$$

Through a series of substitutions and integrations shown in [54] we can obtain an expression for G

$$G = \frac{1}{8\pi\mu}r^2 ln(\frac{1}{r}) \tag{A.83}$$

This equation is valid for three dimensions. If we take each load as independent, one can write

$$G_{lk} = G\delta_{lk} \tag{A.84}$$

G_{lk} is the k component of Galerkin's vector at any point when a unit load is applied at 'i' in the l direction [54]. If we now return to our equation obtained from Galerkin's vector

(A.81) and we presume at any point that we have u^*_{lk} as the displacement at any point in the k direction when a unit load is applied at i in the l direction, we obtain

$$u^*_{lk} = G_{lk,mm} - \frac{1}{2(1-v)} G_{lm,km} \tag{A.85}$$

By substituting (A.83) and (A.84) into (A.85) and simplifying it for two dimensions (A.86) is obtained.

$$u^*_{lk} = \frac{1}{8\pi\mu(1-v)} \left((4v-3)(ln(r))\delta_{lk} + \frac{r_k r_l}{r^2} \right) \tag{A.86}$$

Stresses at any point can be written using the strain-displacement relation (A.5) and the stress-strain equation (A.9). By rearranging these equations they can be expressed as

$$\sigma^*_{kj} = S^*_{lkj} e_l \tag{A.87}$$

Where we need to obtain S^*_{lkj} for (A.86) this is shown clearly in [54]. The tractions of surface forces on the Γ boundary with

$$p^*_k = p^*_{lk} e_l \tag{A.88}$$

Now we can write the traction components p^*_{lk} in two dimensions as

$$p^*_{lk} = -\frac{1}{4\pi(1-v)r} \left[\frac{r_l n_l + r_k n_k}{r} \left[(1-2v)\delta_{lk} + 2\frac{r_k r_l}{r^2} \right] + (1-2v) \left(n_l \frac{r_k}{r} - n_k \frac{r_l}{r} \right) \right] \tag{A.89}$$

To better explain the results obtained from (A.89) and (A.86), we will now consider a unit load placed at point P along the 1 2 direction as shown in Figure A.11 from [59].

We now find that (A.86) and (A.89) can be written as matrices shown in (A.90) and (A.91).

$$[u^*] = \begin{bmatrix} u^*_{11} & u^*_{12} \\ u^*_{21} & u^*_{22} \end{bmatrix} \tag{A.90}$$

$$[p^*] = \begin{bmatrix} p^*_{11} & p^*_{12} \\ p^*_{21} & p^*_{22} \end{bmatrix} \tag{A.91}$$

In (A.90) and (A.91), u^*_{11}, u^*_{12} are the deformations and p^*_{11}, p^*_{12} are the tractions. In this case, a unit-concentrated load along direction 1 is placed at any point P inside or on the surface of the two-dimensional body [59].

A.4.4 Boundary integral formulation

We now consider that one needs to minimize the errors involved in the numerical approximation of the governing equations in elastostatics (A.3). We know that we have to satisfy the

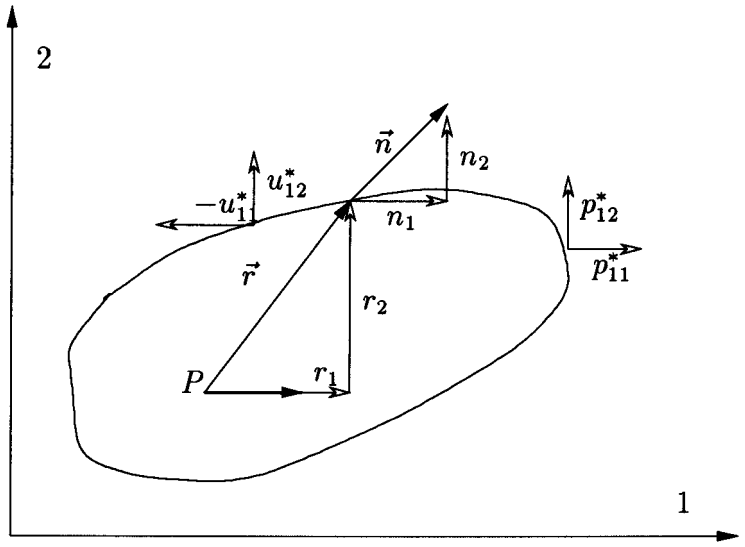


Figure A.11: The fundamental solution in two dimensions with unit force along direction 1

prescribed displacements (*Essential boundary conditions*) and tractions (*Natural boundary conditions*). If we rewrite (A.70) in terms of our governing equation, (A.3), we obtain

$$\int_{\Omega}(\sigma_{kj,j} + b_k)u_k^* d\Omega = 0 \tag{A.92}$$

In this equation we define our weight function as a displacement type function u_k^*. We now use integration by parts similar to section A.4.2 (details shown in [54]) to obtain

$$u_l^i + \int_{\Gamma} p_{lk}^* u_k d\Gamma = \int_{\Gamma} u_{lk}^* p_k d\Gamma + \int_{\Omega} u_{lk}^* b_k d\Omega \tag{A.93}$$

This equation is known as Somigliana's identity and gives the value of the displacements at any internal points in terms of the boundary values u_k and p_k, the forces throughout the domain and the known fundamental solution [54]. (A.93) is valid for any particular point *'i'* where forces are applied [54]. However, for the solution to a boundary element problem we are interested in the values at the boundary Γ and not over the whole domain Ω. A singularity occurs when solving the integral equation (A.93) at the boundary. The solution to this problem is obtained by defining the boundary Γ in the sense of Cauchy Principal values c_{lk}^i. This is described in detail in [54] and (A.94) results.

$$c_{lk}^i u_l^i + \int_{\Gamma} p_{lk}^* u_k d\Gamma = \int_{\Gamma} u_{lk}^* p_k d\Gamma + \int_{\Omega} u_{lk}^* b_k d\Omega \tag{A.94}$$

In two dimensions we can define c_{lk}^i as (A.95) when it is inside Ω.

$$[c] = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{A.95}$$

When c_{lk}^i is on the boundary and the boundary Γ is smooth, we obtain (A.96).

$$[c] = \begin{bmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{2} \end{bmatrix} \tag{A.96}$$

When c^i_{lk} is not smooth, this matrix becomes very difficult to calculate. In two dimensions we will later define the general expression in terms of the angles of the elements.

A.4.5 Boundary element formulation

To be able to obtain a numerical solution we must discretize the boundary into a series of elements over which displacements and tractions are calculated at specific nodal points. We begin by defining the u and p functions which apply over each element 'j' as (A.97) and (A.98) respectively.

$$u = \Phi u^j \tag{A.97}$$

$$p = \Phi p^j \tag{A.98}$$

The values of the matrices u^j and p^j have a dimension of $[N \times 1]$ where N is equal to the number of nodes per element n, multiplied by m degrees of freedom per node. The matrix Φ then has the dimension of $[m \times N]$. NE elements now exist into which the boundary Γ is divided. We can now write (A.94) in terms of a discretized system using (A.90), (A.91), (A.97) and (A.98) into (A.99)

$$c^i u^i + \sum_{j=1}^{NE} \left(\int_{\Gamma_j} p^* \Phi d\Gamma \right) u^j = \sum_{j=1}^{NE} \left(\int_{\Gamma_j} u^* \Phi d\Gamma \right) p^j + \sum_{s=1}^{M} \left(\int_{\Omega_s} u^* b d\Omega \right) \tag{A.99}$$

We notice that our body force term b on the domain Ω is divided into M internal cells over which the boundary force integrals are computed [54]. These body force terms can usually be avoided by taking the body force integrals to the boundary [54]. We can now also use the transformation of coordinates exactly similar to that used in finite elements. For the internal domain we write

$$d\Omega = \|J\| d\zeta_1 d\zeta_2 \tag{A.100}$$

And similarly for the boundary we obtain

$$d\Gamma = \|G\| d\zeta_1 \tag{A.101}$$

J is the full Jacobian matrix in two dimensions. G is the reduced Jacobian matrix.

$$\|G\| = \frac{d\Gamma}{d\zeta} = \left(\left[\frac{dx_1}{d\zeta} \right] \left[\frac{dx_2}{d\zeta} \right] \right)^{\frac{1}{2}} \tag{A.102}$$

With (A.101) and (A.100), we can write (A.99) as (A.103) by making use of numerical integration.

$$c^i u^i + \sum_{j=1}^{NE} \left(\sum_{k=1}^{l} w_k (p^* \Phi)_k \|G\| \right) u^j = \sum_{j=1}^{NE} \left(\sum_{j=1}^{l} w_k (u^* \Phi)_k \|G\| \right) p^j + \sum_{s=1}^{M} \left(\sum_{p=1}^{r} w_p (u^* b^*)_p \|J\| \right) \tag{A.103}$$

l is the number of integration points on the surface elements and w_k is the weight at those points [54]. Once (A.103) has been numerically integrated to correspond to a particular node $'i'$ it can be written as

$$c^i u^i + \sum_{j=1}^{N} \hat{H}^{ij} u^j = \sum_{j=1}^{N} G^{ij} p^j + \sum_{s=1}^{M} B^{is} \tag{A.104}$$

N is the number of nodes, u^j and p^j are the displacements and tractions at node $'j'$ [54]. $\hat{H}^{ij}$,G^{ij} and B^{is} are the matrices obtained after numerically integrating (A.103). We can now simplify the left hand side of (A.104) by considering

$$\begin{aligned} H^{ij} &= \hat{H}^{ij} \quad \text{if} \quad i \neq j \\ H^{ij} &= \hat{H}^{ij} + c^i \quad \text{if} \quad i = j \end{aligned} \tag{A.105}$$

If we consider the contribution of all the $'i'$ nodes we obtain a global system of equations in

$$\mathbf{HU} = \mathbf{GP} + \mathbf{B} \tag{A.106}$$

The vectors $\mathbf{U}$ and $\mathbf{P}$ represent all the values of the displacements and tractions before applying the boundary conditions [54]. These conditions can be introduced by rearranging the columns in $\mathbf{H}$ and $\mathbf{G}$ and passing all the unknowns to a vector $\mathbf{X}$ on the left hand side [54]. This gives the final system of equations

$$\mathbf{AX} = \mathbf{F} \tag{A.107}$$

We also note that the $\mathbf{B}$ vector has been incorporated into $\mathbf{F}$. By solving the linear system of equations all the unknown displacements and tractions are obtained.

A.4.6 Implementation

We can now consider a boundary Γ divided into elements and at first we supply each element with a single node in the middle of the element. It follows that we have a smooth boundary and we can use (A.96) in (A.104) to obtain a system of equations to solve the unknowns. It is clear in Figure A.12 that this results in a constant element formulation causing large discontinuities between elements since it assumes a single solution is valid for the entire element.

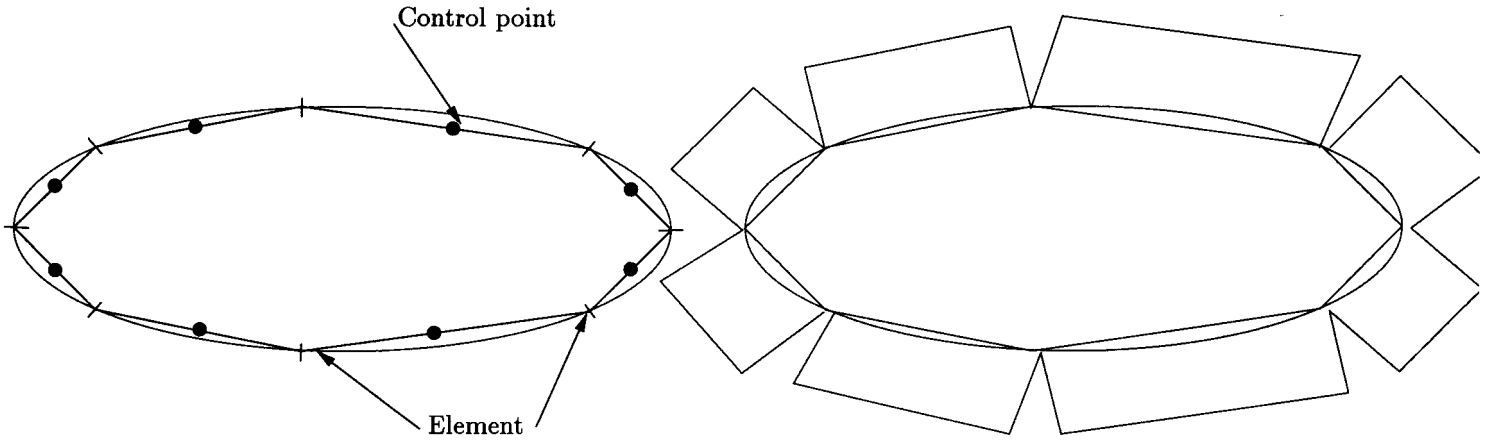


Figure A.12: Constant element solution BEM

Various methods such as "Saw tooth" correction exist to improve the solution of the constant element formulation. But in general it is a well-known fact that low order BEM perform very poorly in structural analysis. To improve this situation we have to put nodes at the edge of elements which in turn entails that we have to obtain a new $[c]$ matrix for the boundary which is not smooth. To obtain this matrix we consider the limit of the fundamental solution of tractions in (A.108). This is more clearly illustrated in Figure A.13

$$[I] = \lim_{\epsilon \to 0} \left(\int_{\Gamma_\epsilon} p^* d\Gamma \right) \tag{A.108}$$

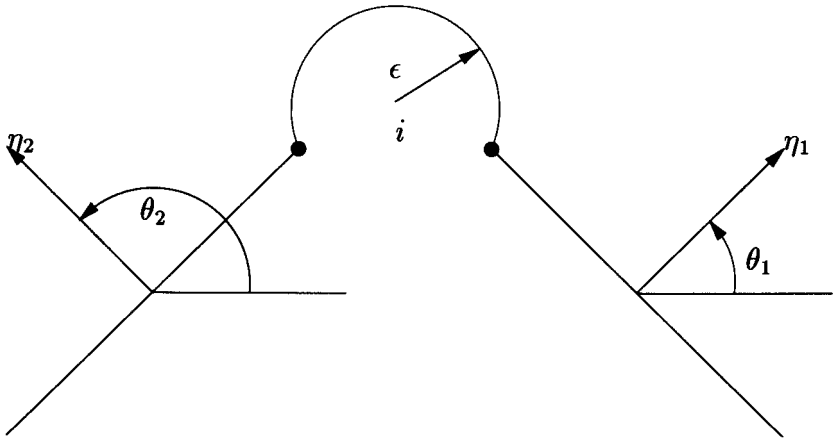


Figure A.13: Corner point

(A.109) gives the equation to describe the two-dimensional case. If we use the conventions described in Figure A.13 we obtain the limit

$$[I] = D \begin{bmatrix} 4(1-v)(\pi + \theta_2 - \theta_1) + \sin 2\theta_1 - \sin 2\theta_2 & \cos 2\theta_2 - \cos 2\theta_1 \\ \cos 2\theta_2 - \cos 2\theta_1 & 4(1-v)(\pi + \theta_2 - \theta_1) + \sin 2\theta_2 - \sin 2\theta_1 \end{bmatrix} \tag{A.109}$$

$$D = \frac{-1}{8\pi(1-v)} \tag{A.110}$$

We can now calculate the new $[c]$ matrix

$$c_{lk} = \delta_{lk} + I_{lk} \tag{A.111}$$

This now allows us to use elements with nodes on the end of each element. But in order to stimulate traction discontinuities over the boundary, the concept of double nodes can be employed [60]. This means that we have two nodes with exactly the same coordinates connected to two different elements. If we have one node at each end of the element we obtain linear elements but, however, this causes problems with curved geometries. It would thus be better to employ an element with two nodes at its corners and one in the middle. For these quadratic elements we can write (A.97) and (A.98) as (A.112) and (A.113) respectively

$$u = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} \phi_1 & 0 & \phi_1 & 0 & \phi_1 & 0 \\ 0 & \phi_2 & 0 & \phi_2 & 0 & \phi_2 \end{bmatrix} \begin{pmatrix} u_1^1 \\ u_2^1 \\ u_1^2 \\ u_2^2 \\ u_1^3 \\ u_2^3 \end{pmatrix} = \mathbf{\Phi u^j} \tag{A.112}$$

$$p = \begin{bmatrix} p_1 \\ p_2 \end{bmatrix} = \begin{bmatrix} \phi_1 & 0 & \phi_1 & 0 & \phi_1 & 0 \\ 0 & \phi_2 & 0 & \phi_2 & 0 & \phi_2 \end{bmatrix} \begin{pmatrix} p_1^1 \\ p_2^1 \\ p_1^2 \\ p_2^2 \\ p_1^3 \\ p_2^3 \end{pmatrix} = \mathbf{\Phi p^j} \tag{A.113}$$

The ϕ_is are the quadratic interpolation functions as set out in (A.114) in the local coordinate system.

$$\begin{aligned} \phi_1 &= \frac{1}{2}\zeta(\zeta - 1) \\ \phi_2 &= (1 - \zeta^2) \\ \phi_3 &= \frac{1}{2}\zeta(\zeta + 1) \end{aligned} \tag{A.114}$$

By using the same interpolation functions ϕ_is as we used in (A.112) and (A.113) for the transformation of the coordinates of the nodes we obtain

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} \phi_1 & 0 & \phi_1 & 0 & \phi_1 & 0 \\ 0 & \phi_2 & 0 & \phi_2 & 0 & \phi_2 \end{bmatrix} \begin{pmatrix} x_1^1 \\ x_2^1 \\ x_1^2 \\ x_2^2 \\ x_1^3 \\ x_2^3 \end{pmatrix} = \mathbf{\Phi x^j} \tag{A.115}$$

We can now consider a simple example of a square geometry with four elements, one on each side and three nodes per element, when we obtain the setup as shown in Figure A.14.

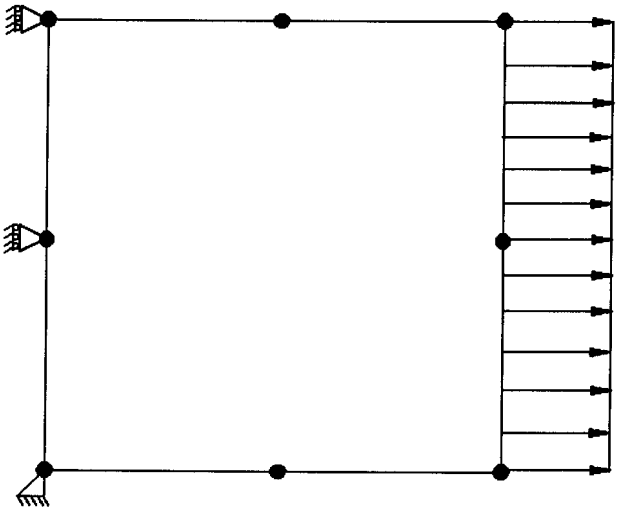


Figure A.14: Simple boundary element problem

Treatment of the boundary conditions are shown in Figure A.15. This clearly shows how the double node concept is employed to satisfy traction discontinuities over the boundary.

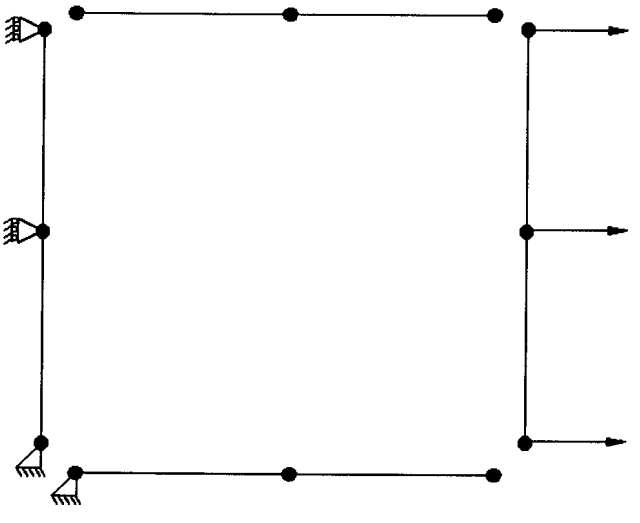


Figure A.15: Applying boundary conditions

We note that the distributed load is treated by only using the value of the force at the point at which it is applied since we are only interested in the tractions applied. This can be very useful when considering a complex distributed load as seen in Figure A.16. This problem would involve quite a complicated integration to be applied to a finite element. The boundary element method allows us to use only the immediate value at the node. This, however, entails that enough nodes have to be placed on the side with the complex load to describe the load accurately.

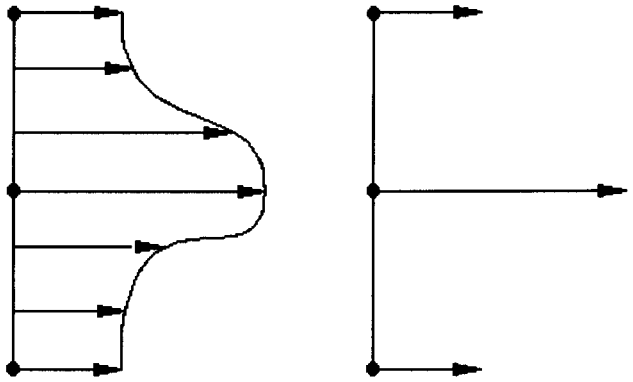


Figure A.16: Applying complex load

When solving the problem in Figure A.14, a modulus of elasticity $E = 1e6$, Poisson's ratio $v = 0.3$ and a distributed load of 500000 was used. For this setup, the BEM should be able to represent a constant deformation and stress Figure A.17 shows the solution to the problem. The values used in this problem are not realistic but it does enable us to see if the correct results were obtained. The plot shows the original geometry as well as the deformed geometry.

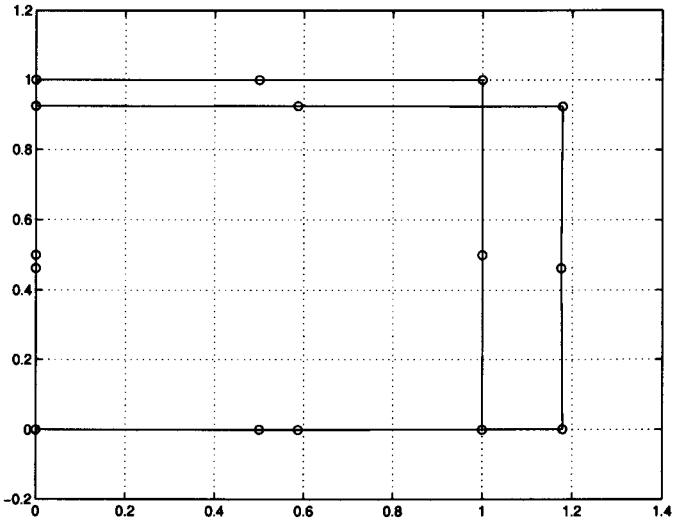


Figure A.17: Constant deformation of a simple geometry

In order to see that the answer obtained has converged, the mesh has to be refined. In Figure

A.18 the solution is plotted for a refined mesh which shows the same deformation as Figure A.17.

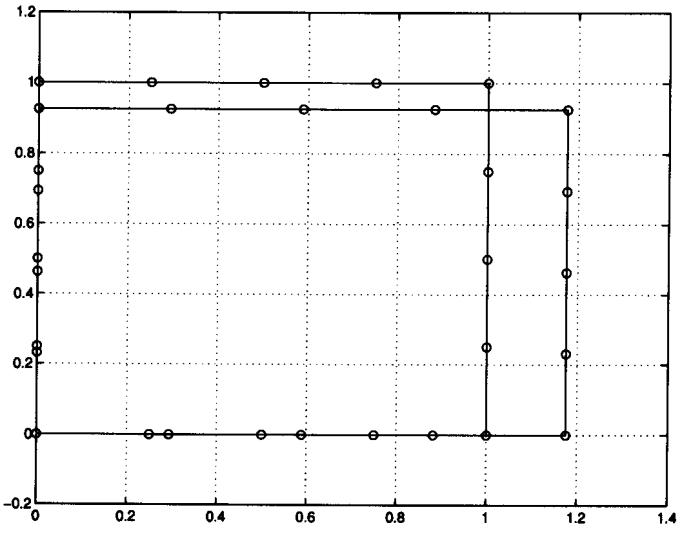


Figure A.18: Constant deformation of a refined mesh

The problem considered should also show a constant stress over the entire geometry. BEM solves the tractions and displacements on the boundary and then solves the internal displacements and stresses at the specified internal points by making use of Somigliana's identity (A.93). Table A.4 gives the internal values calculated by BEM using four internal points. This is done using the mesh in Figure A.17.

Coordinates				
x	y	σ_x	τ_{xy}	σ_y
0.2500000E+00	0.2500000E+00	0.4871852E+06	-0.3897214E+04	0.3686987E+04
0.7500000E+00	0.2500000E+00	0.5055677E+06	0.1617945E+04	0.2063524E+04
0.2500000E+00	0.7500000E+00	0.4879818E+06	0.3100705E+04	0.2811925E+04
0.7500000E+00	0.7500000E+00	0.5063640E+06	-0.8215335E+03	0.1188476E+04

Table A.4: Internal stresses (BEM)

Clearly the stress representation is not very accurate for the internal points since we know that for this problem the stress in the x direction has to be constant over the whole surface. Table A.5 gives the stress at the same internal points calculated using the refined mesh.

The x stresses in Table A.5 are clearly more accurate than for the single elements. But this result is still poor in comparison with the results obtained in finite elements. We know that in our boundary element formulation A.103, the errors are only minimized on the boundary itself. The prediction of the displacements on the internal points are shown in Table A.6.

The displacement in the x direction is definitely more accurate than the stresses calculated at the same internal point. The tractions on displacements on the boundary are, however, calculated very accurately.

Coordinates				
x	y	σ_x	τ_{xy}	σ_y
0.2500000E+00	0.2500000E+00	0.4994989E+06	-0.5405129E+04	0.3858706E+04
0.7500000E+00	0.2500000E+00	0.5002098E+06	-0.9422518E+03	-0.1184704E+04
0.2500000E+00	0.7500000E+00	0.4994980E+06	0.5405965E+04	0.3859719E+04
0.7500000E+00	0.7500000E+00	0.5002088E+06	0.9414868E+03	-0.1183872E+04

Table A.5: Internal stresses refined mesh (BEM)

Coordinates			
x	y	Displacement x	Displacement y
0.2500000E+00	0.2500000E+00	0.4550087E-01	-0.1889137E-01
0.7500000E+00	0.2500000E+00	0.1323639E+00	-0.1938061E-01
0.2500000E+00	0.7500000E+00	0.4549616E-01	-0.5655904E-01
0.7500000E+00	0.7500000E+00	0.1323686E+00	-0.5606983E-01

Table A.6: Internal displacements refined mesh (BEM)

A.5 Finite element method and boundary element method comparison

The boundary element formulation is mathematically complicated and its implementation just as tedious. BEM and FEM have major differences. In a comparison it can be seen as either an advantage or a disadvantage. These differences will be put out more clearly in this section [38].

1 FEM: An entire domain mesh is required.
BEM: A mesh of the boundary only is required.
Comment: Because of the reduction in size of the mesh, one often hear people saying that the problem size has been reduced by one dimension. This is one of the major advantages of BEM - construction of meshes for complicated objects, particularly in 3D, is a very time-consuming exercise.

2 FEM: Entire domain solution is calculated as part of the solution.
BEM: Solution on the boundary is calculated first, and then the solution at domain points (if required) is found as a separate step.
Comment: There are many problems where the details of interest occur on the boundary, or are localized to a particular part of the domain, and hence an entire domain solution is not required.

3 FEM: Reactions on the boundary are typically less accurate than the dependent variables.

BEM: Both u and q are of the same accuracy.

4 FEM: Differential equation is being approximated.
BEM: Only boundary conditions are being approximated.
Comment: The use of the Green-Gauss theorem and a fundamental solution in the formulation means that the BEM involves no approximations of the differential Equation in the domain - only in its approximations of the boundary conditions.

5 FEM: Sparse symmetric matrix is generated.
BEM: Fully populated asymmetric matrices generated.
Comment: The matrices are generally of different sizes due to the differences in size of the domain mesh compared to the surface mesh. There are problems where either method can give rise to the smaller system and quickest solution - it depends partly on the volume to surface ratio. For problems involving infinite or semi-infinite domains, BEM is to be favored.

6 FEM: Element integrals are easy to evaluate.
BEM: Integrals are more difficult to evaluate, and some contain integrands that become singular.
Comment: BEM integrals are far more difficult to evaluate. Also the integrals that are the most difficult (those containing singular integrands) have a significant effect on the accuracy of the solution, so these integrals need to be evaluated accurately.

7 FEM: Is widely applicable, and handles non-linear problems well.
BEM: Cannot handle all linear problems.
Comment: A fundamental solution must be found (or at least an approximate one) before the BEM can be applied. There are many linear problems (e.g. virtually any non-homogeneous equation) for which fundamental solutions are not known. There are certain areas in which the BEM is clearly superior, but it can be rather restrictive in its applicability.

8 FEM: Is relatively easy to implement.
BEM: Is much more difficult to implement.
Comment: The need to evaluate integrals involving singular integrands makes the BEM at least an order of magnitude more difficult to implement than a corresponding finite element procedure.

Appendix B

Numerical optimization

B.1 Optimization formulation

B.1.1 Mathematical definition

Mathematical optimization is the process by which we attempt to minimize a problem of the general mathematical form

$$\text{Minimize} \qquad f(x), x = (x_1, x_2, \ldots, x_n) \in R^n \tag{B.1}$$

Subject to the constraints

$$g_j(x) \leq 0 \qquad j = 1, 2, \ldots, m \tag{B.2}$$

$$h_j(x) = 0 \qquad j = 1, 2, \ldots, r \tag{B.3}$$

where the design variables are $x_1, x_2, \ldots, x_n$
the objective function $f(x)$
the inequality constraints $g_j(x)$ and
the equality constraints $h_j(x)$

The vector x that solves the problem (B.1) is denoted by x^*. This is called the optimal solution and $f(x^*)$ the optimal function value.

B.1.2 Solution methodology

When the objective function $f(x)$ does not exist in terms of a simple analytical function determining the optimal solution x^* becomes a more complicated procedure. It may not be possible to differentiate the equation and determine the minimum of all the turning points. In engineering, the objective function usually has to be obtained via a simulation. This is a timely process and thus it is not practical to evaluate all functions to determine which

solution is the optimum solution. Instead numerical optimization consists of techniques which allows us to find the optimum solution x^* by evaluating few functions. The algorithms can be divided into different classes and each class is based on certain principles. Since each problem is different, some algorithms will perform better that others on different problems.

B.2 Genetic algorithm

B.2.1 Introduction

Genetic algorithms (GAs) make use of the survival of the fittest strategy found in nature to search the solution space of a function. The principle of the survival of the fittest results in the fittest individuals of any population reproducing and surviving to the next generation. These species are well adapted to their environment. Inferior individuals also have the possibility of surviving [61, 62].

Many optimization algorithms are limited to convex regular functions. Many functions are, however, multi-modal, discontinuous and non-differentiable. These functions have been optimized, using stochastic search techniques. The stochastic techniques do not make use of traditional search techniques such as gradients, Hessians, linearity and continuity. Decision rules or stochastic sampling is used instead to determine the next sampled point. GAs can therefore be used to solve functions that do not possess properties such as continuity, differentiability and satisfaction of the Lipschitz Condition [62].

GAs require the determination of six issues: solution representation (representation of design variables), selection, genetic operators that make up the reproductive function (including crossover and mutation), the creation of the initial population, termination criteria and the evaluation function. Each of these issues will be discussed [62].

B.2.2 Solution representation

A chromosome representation function determines how the problem is structured in the GA and the genetic operators that are used. The function is required to describe each individual in the population of interest. An initial design population is made up a of a sequence of genes. Each chromosome represents a design variable. The design vectors in the design population are improved in subsequent generations by means of the selection, crossover and mutation operators [61, 62].

B.2.3 Selection function

A probabilistic selection operation is performed based upon an individual's fitness. The operation is such that fitter individuals have a greater chance of being selected to form the mating pool. Individuals may be selected more than once and all individuals in the pool have a chance of being selected [61, 62].

A common selection method uses the following approach:

1. Assign a probability of selection P_j to each individual j based on its fitness value.
2. Compare the series of random numbers N that is generated against the cumulative probability of the population $C_i = \sum_{j=1}^{i} P_j$.
3. Select the appropriate individual i and copy to the new population if $C_{i-1} < U(0,1) \leq C_i$.

There are a number of different methods of selection determining how probabilities are assigned to individuals: roulette wheel, tournament selection and ranking selection.

Roulette wheel

The roulette wheel selection is also known as the expected value selection [63]. The objective function is converted from a maximization to a minimization problem by multiplying the objective function with -1. A constant must be added to functions with negative values as function values must be positive. The probability P_i for each design is calculated as follows:

$$P_i = \frac{F_i}{\sum_{j=1}^{Popsize} F_j} \qquad i = 1, 2, 3, \ldots \tag{B.4}$$

where F_i denotes the fitness of individual i.

Tournament selection

Tournament selection selects j individuals randomly and inserts the best j into the new population. The procedure is repeated until N individuals have been selected. Tournament selection simulates the process whereby individuals in the population compete for mating rights [61, 62].

Ranking methods

In ranking method, after ranking in descending order, the relative member's fitness P_i is expressed as the relative fitness of each member i.

$$P_i = \frac{t_i}{\sum_{i=1}^{e} t_i} \tag{B.5}$$

where

$$t_i = 2 \cdot \frac{(e + 1 - i)^c}{(e^2 + e)} \tag{B.6}$$

c is any value between 1 and 10 and e is the population size. The cumulative probability space g_j is constructed using the relative fitness p_i [61, 62].

Normalized geometric ranking defines P_i as:

$$P_i = q'(1 - q)^{r-1} \tag{B.7}$$

where q is the probability of selecting the best individual, r is the rank of the individual where 1 is the best, P is the population size and

$$q' = \frac{q}{1-(1-q)^P} \tag{B.8}$$

B.2.4 Genetic operators

Genetic Operators are used to create new solutions based on solutions that already exists in the population. Crossover and mutation are the two basic types of operators. Crossover produces a new individual from two individuals. Crossover allows selected individuals to trade characteristics. Mutation alters one individual in the production of a new solution. Mutation protects against complete loss of genetic diversity by randomly changing characteristics of the design variable [61, 62].

Binary mutation and simple crossover are defined for $\overline{X}$ and $\overline{Y}$ binary. Binary mutation flips each bit in every individual in the population with probability p_m as shown in (B.9) [62].

$$x_i' = \begin{cases} 1 - x_i & \text{if } U(0,1) < p_m \\ x_i & \text{otherwise} \end{cases} \tag{B.9}$$

Simple crossover generates a random number r from a uniform distribution from 1 to m and creates two new individuals $\overline{X}'$ and $\overline{Y}'$ according to (B.10) and (B.11) [62].

$$x_i' = \begin{cases} x_i & \text{if } i < r \\ y_i & \text{otherwise} \end{cases} \tag{B.10}$$

$$y_i' = \begin{cases} x_i & \text{if } i < r \\ y_i & \text{otherwise} \end{cases} \tag{B.11}$$

Uniform mutation, non-uniform mutation, multi-non-uniform mutation, boundary mutation, simple crossover, arithmetic crossover and heuristic crossover are defined for X and Y real. Uniform mutation randomly selects one variable j and sets it equal to a uniform random number $U(a_i b_i)$ as shown by (B.12) [62].

$$x_i' = \begin{cases} U(a_i, b_i) & \text{if } i = j \\ x_i & \text{otherwise} \end{cases} \tag{B.12}$$

Boundary mutation randomly selects one variable j and sets it equal to either its lower or upper bound where $r = U(0,1)$ shown by (B.13) [62].

$$x_i' = \begin{cases} a_i & \text{if } i = j, r < 0.5 \\ b_i & \text{if } i = j, r \geq 0.5 \\ x_i & \text{otherwise} \end{cases} \tag{B.13}$$

Non-uniform mutation randomly selects one variable j and sets it equal to a non-uniform random number as seen in (B.14) and (B.15) [62].

$$x_i' = \begin{cases} x_i + (b_i - x_i)f(G) & \text{if } r_1 < 0.5 \\ x_i - (x_i + a_i)f(G) & \text{if } r_1 \geq 0.5 \\ x_i & \text{otherwise} \end{cases} \tag{B.14}$$

where

$$f(G) = \left(r_2\left(1 - \frac{G}{G_max}\right)\right)^b \tag{B.15}$$

r_1, r_2 = a uniform random number between $(0, 1)$
G = current generation
G_max = the maximum number of generations
b = a shape parameter

The multi-non-uniform mutation operator applies the non-uniform operator to all of the variables in the parent $\overline{X}$ [62].

Real-valued crossover is the same as the binary simple crossover shown in (B.10) and (B.11). Arithmetic crossover results in two complimentary linear combinations of the parents, where $r = U(0, 1)$ [62].

$$\overline{X}' = r\overline{X} + (1 - r)\overline{Y} \tag{B.16}$$

$$\overline{Y}' = (1 - r)\overline{X} + r\overline{Y} \tag{B.17}$$

Heuristic crossover makes use of the fitness of the individuals to produce a linear extrapolation of the two individuals. $\overline{X}'$ is created where $r = U(0, 1)$ and $\overline{X}$ has a better fitness than $\overline{Y}$. If $\overline{X}'$ is unfeasible, a new random number r must be generated and a new solution created. After t failures, the children should be made to equal the parents and the procedure should stop [62].

$$\overline{X}' = \overline{X} + r(\overline{X} - \overline{Y}) \tag{B.18}$$

$$\overline{Y}' = \overline{X} \tag{B.19}$$

$$\text{feasibility} = \begin{cases} 1 & \text{if } x_i' \geq a_i, x_i' \leq b_i \quad \forall i \\ 0 & \text{otherwise} \end{cases} \tag{B.20}$$

B.2.5 Initialization function

An initial population must be supplied to the GA. A randomly generated solution is commonly chosen. GAs can, however, improve on existing solutions and the beginning population can therefore consist of a mixture of randomly generated individuals seeded with potentially good solutions [62].

B.2.6 Termination criteria

The GA moves from one generation to the next generation and parents are selected and reproduced until the criterion for termination has been met. This criterion is often specified as a maximum number of generations. Termination can also be chosen as population convergence, when the sum of the deviations among individuals becomes smaller than a certain specification. Other criteria may be lack of improvement in the best solution over a specified number of generations or the achievement of some arbitrarily "acceptable" threshold [62].

B.2.7 Objective function

Many forms of evaluation functions may be used in a GA. The minimal requirement of the function is that it maps the population into a partially ordered set. Stochastic decision rules make for the evaluation function being independent of the GA [62].

Appendix C

Flow diagrams for selected programs

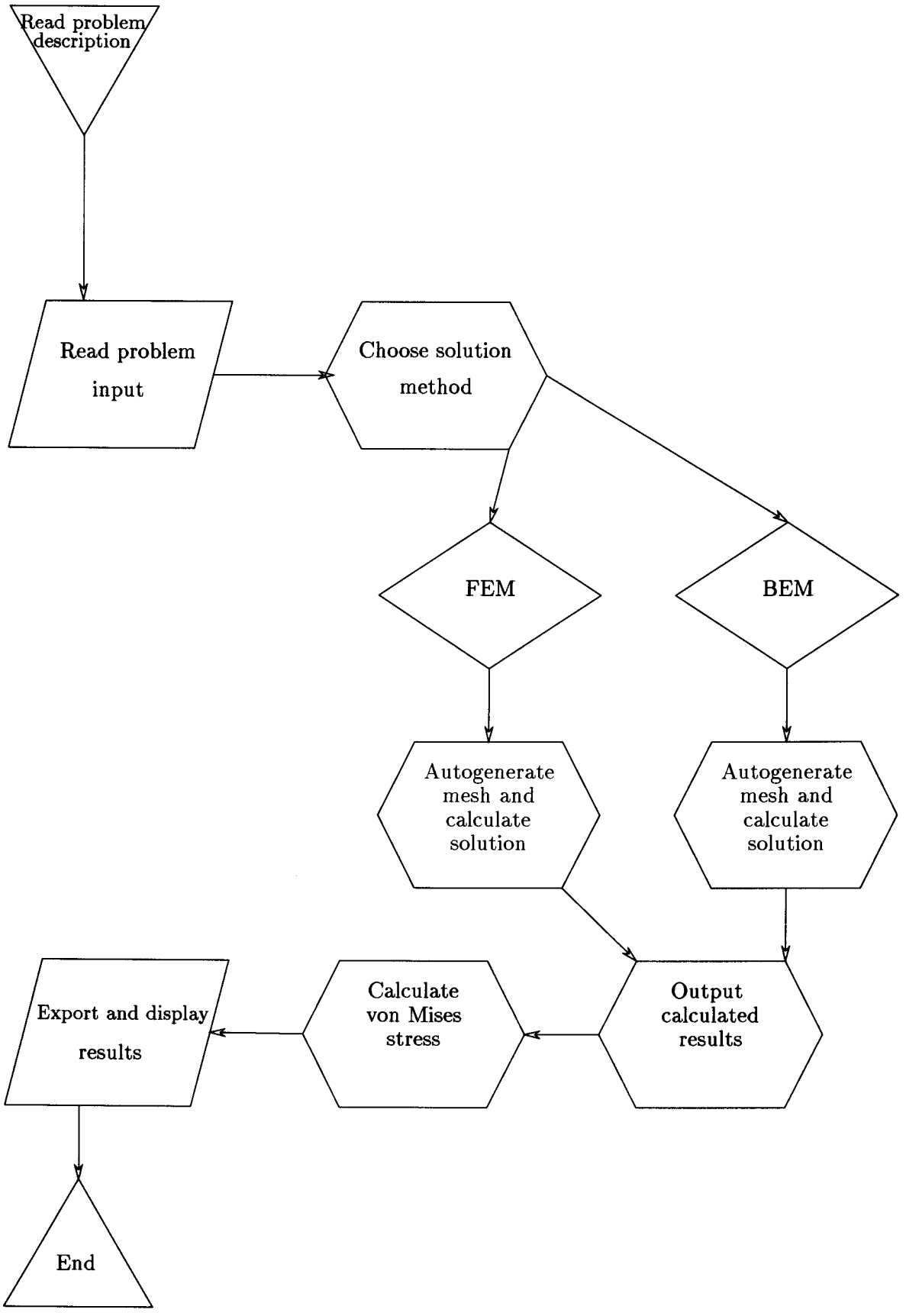


Figure C.1: Boundary cell calculation

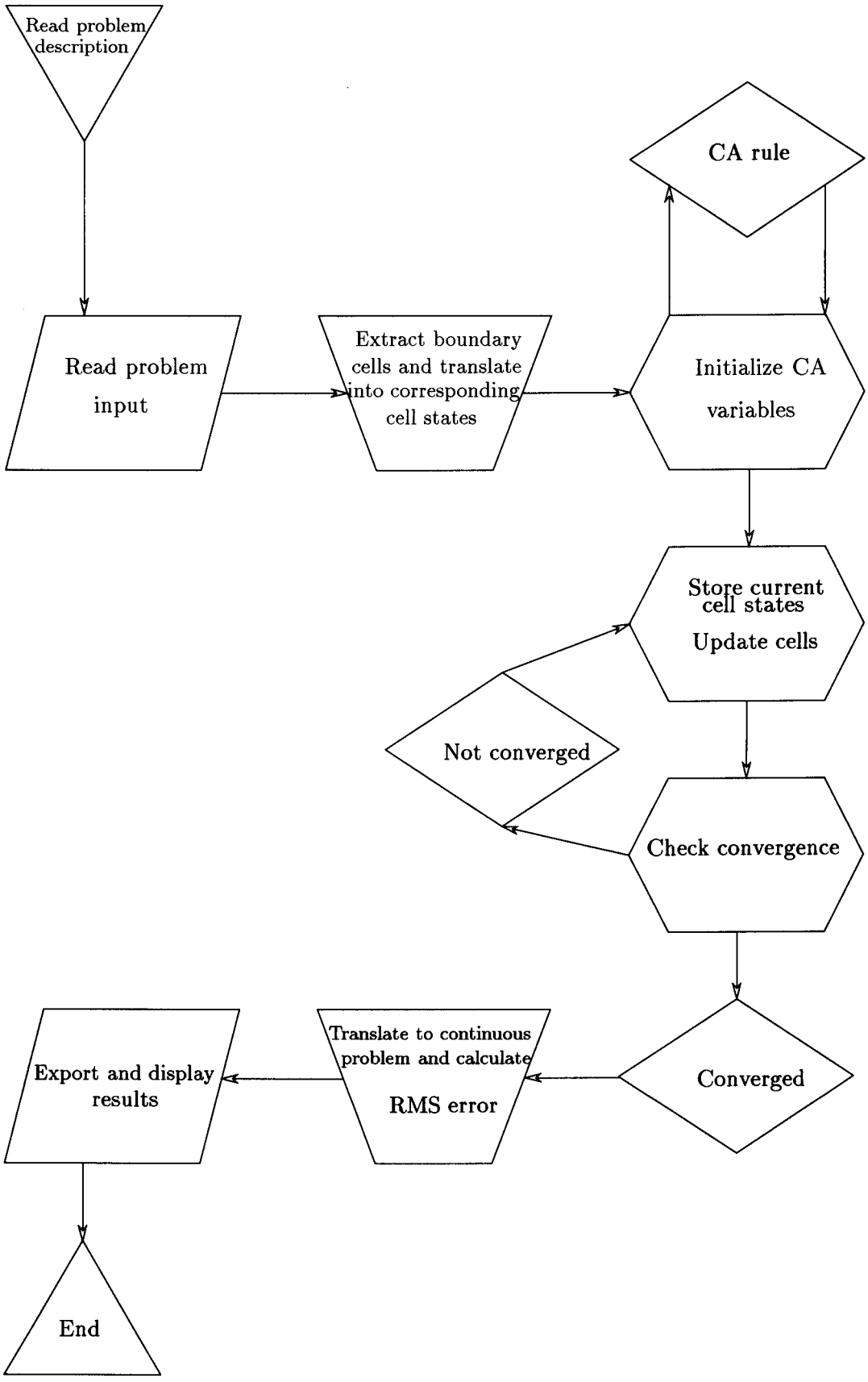


Figure C.2: CA simulation main

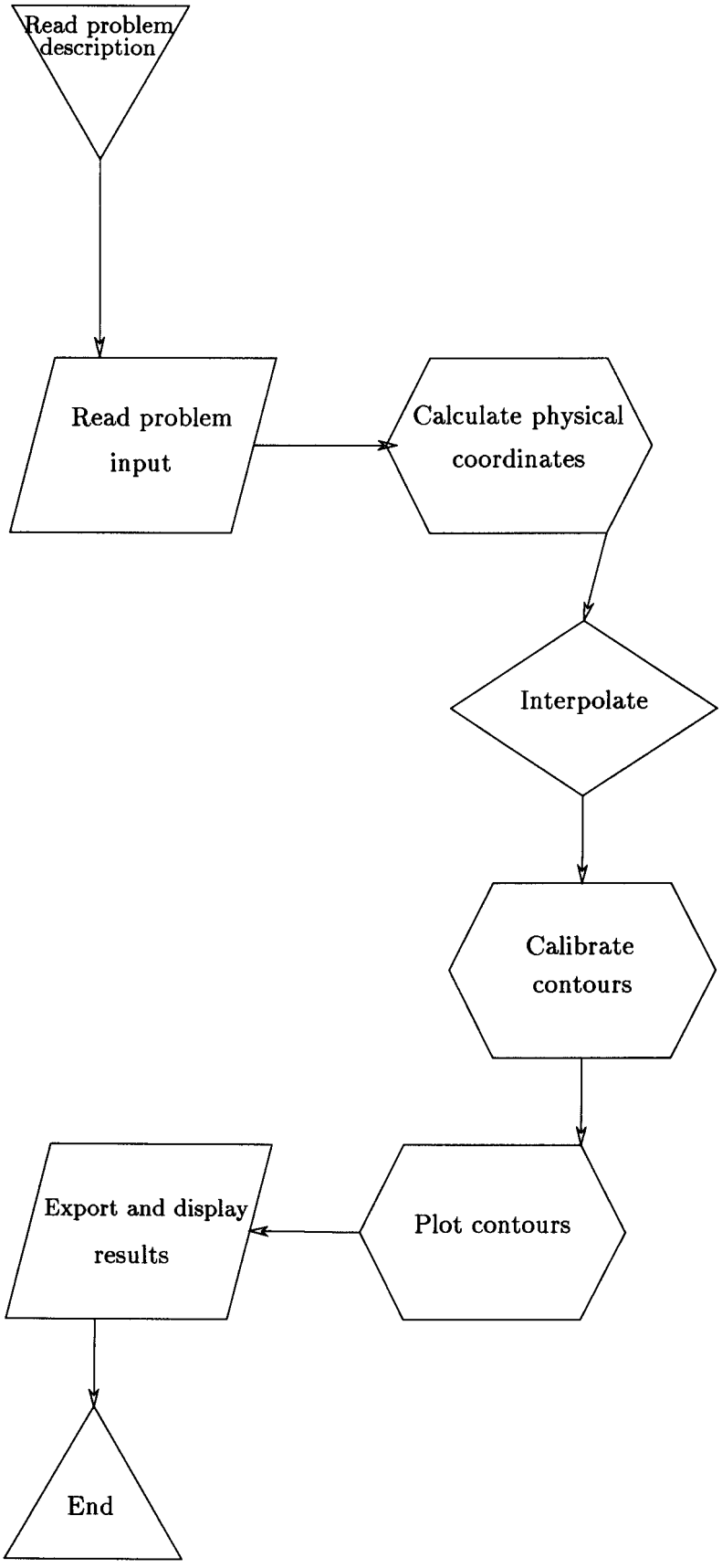


Figure C.3: Stress contour plot

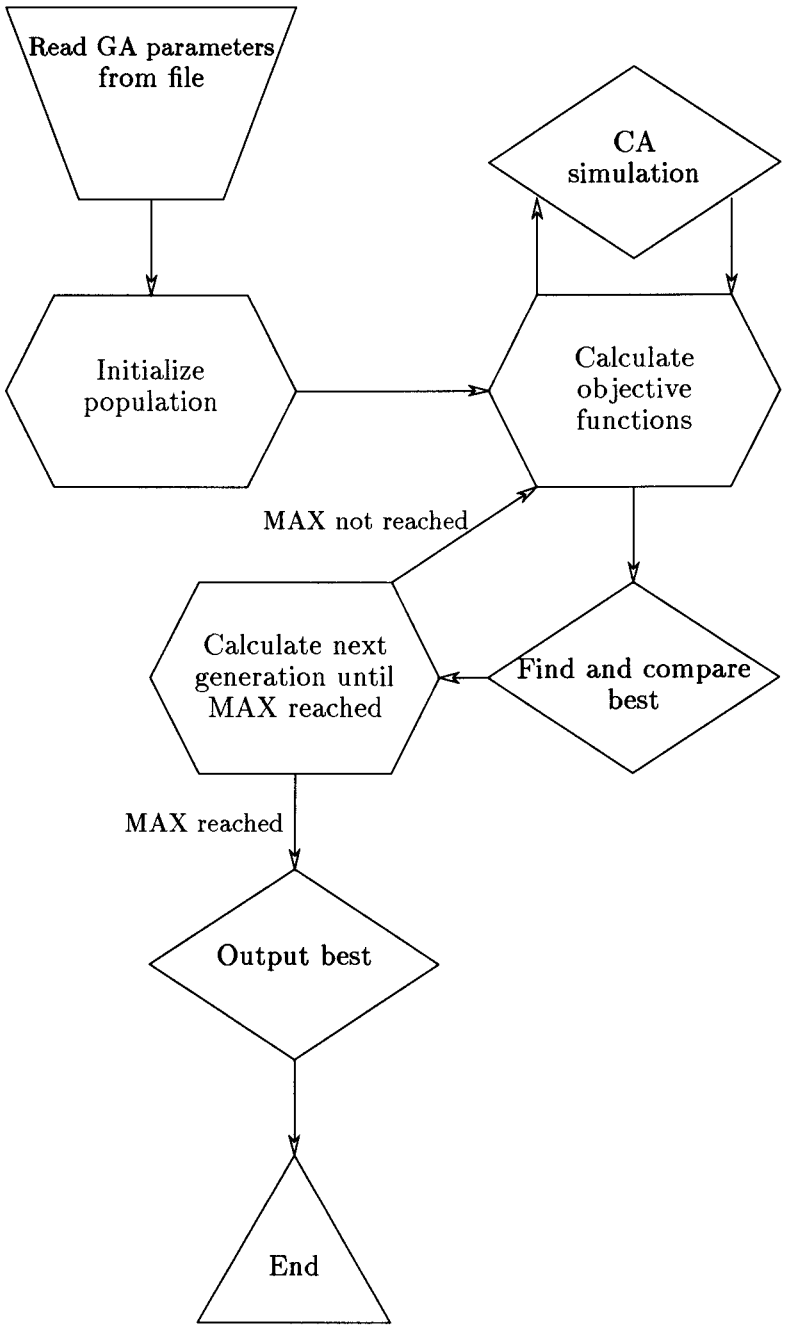


Figure C.4: Genetic algorithm

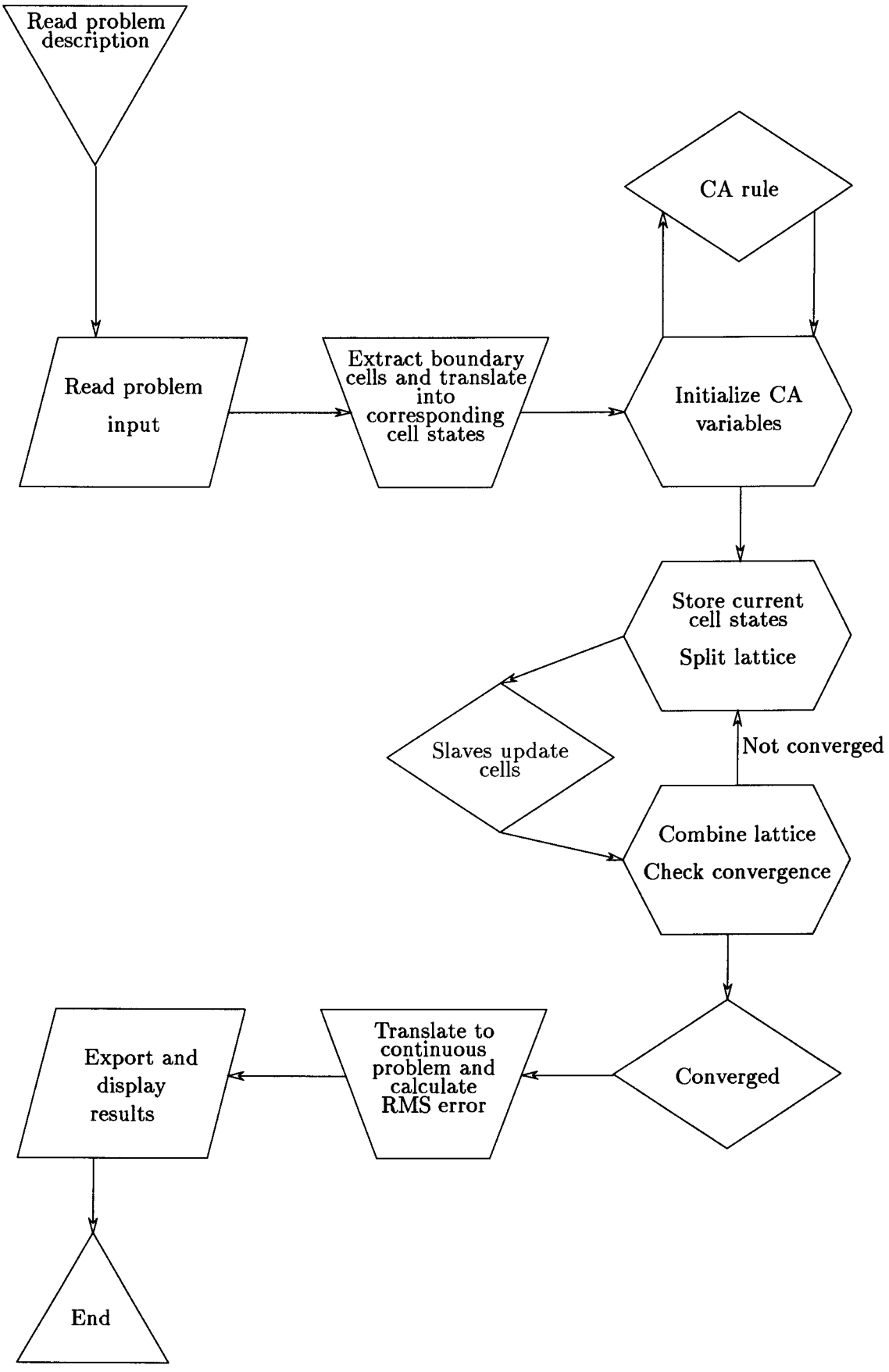


Figure C.5: CA parallel implementation number one

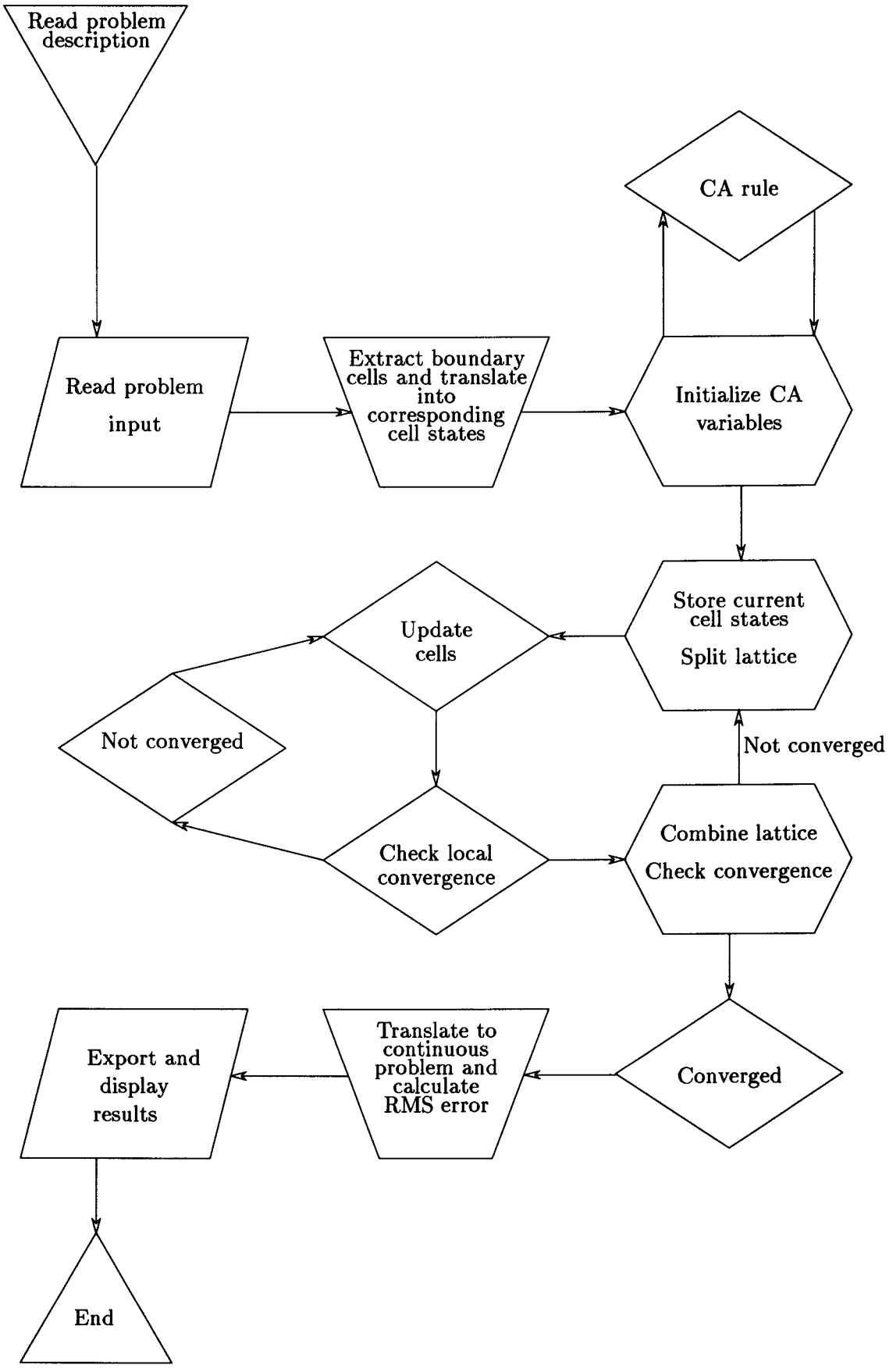


Figure C.6: CA parallel implementation number two

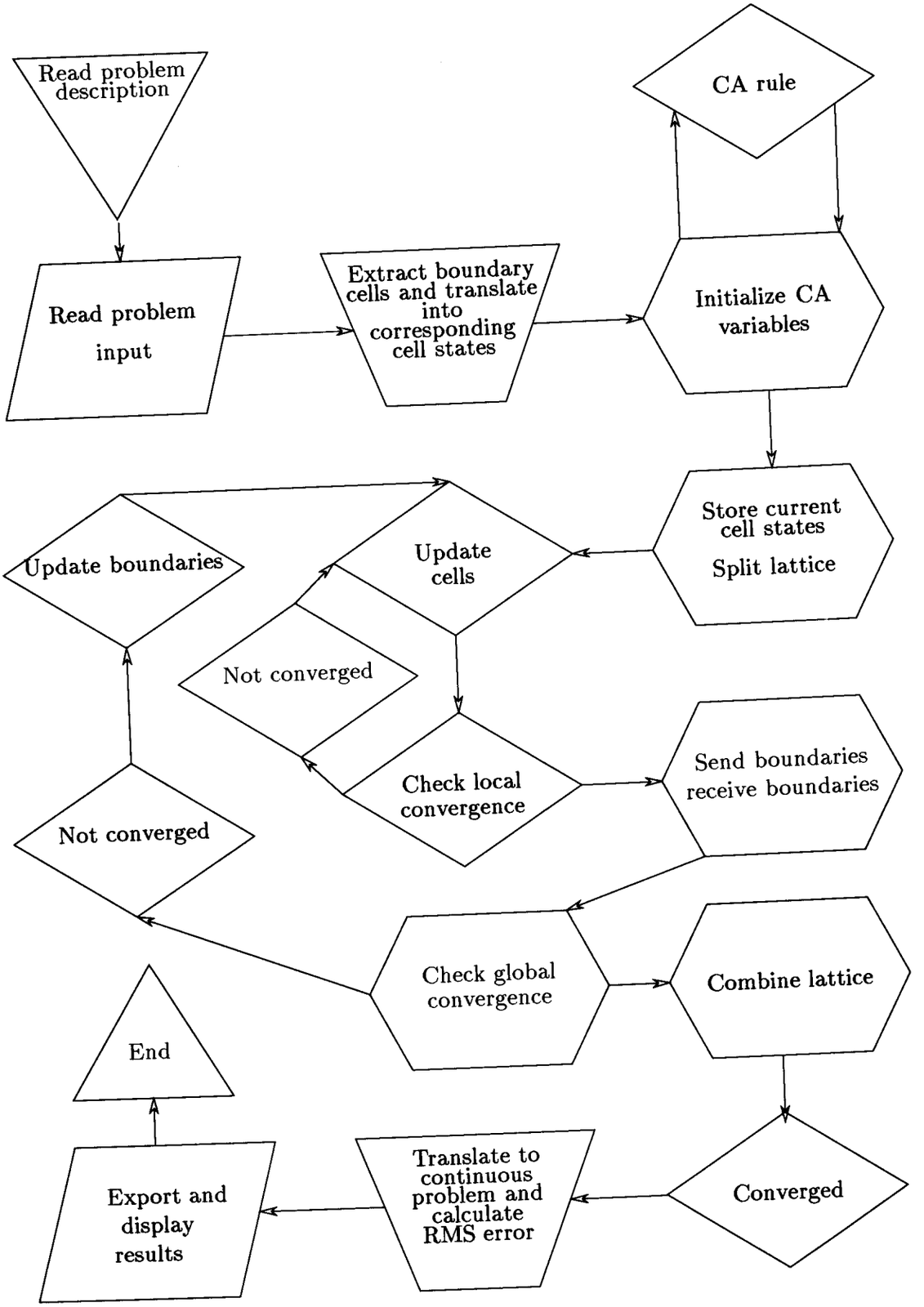


Figure C.7: CA parallel implementation number three

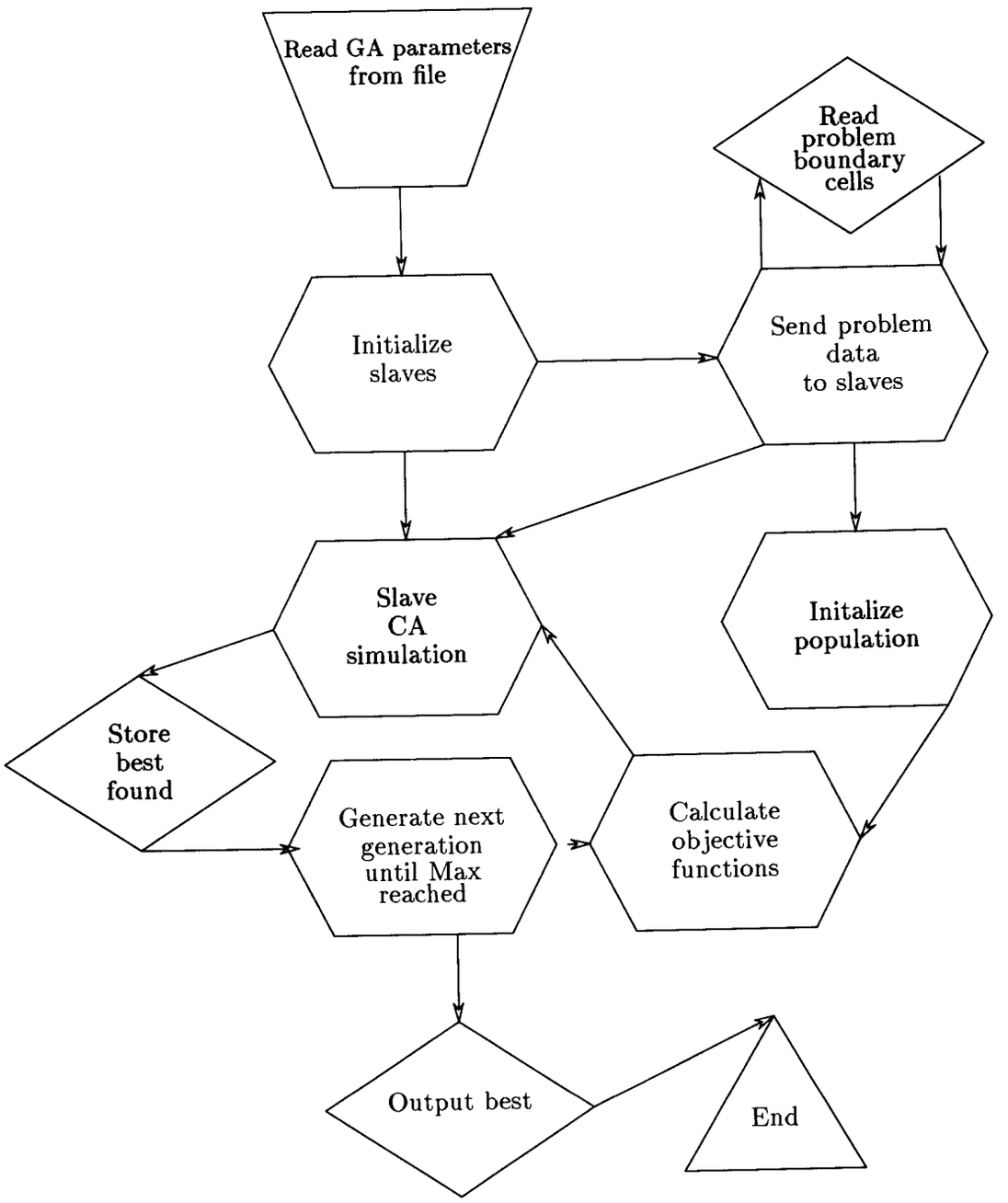


Figure C.8: GA parallel implementation