Continuum Modeling using Finite Elements

1 Introduction

Materials modeling at the continuum level can generally be described as the solution of boundary value problems. For a given physical system, a set of governing equations is derived based on fundamental principles (equilibrium, conservation of mass, Newton's Laws, etc.). Next, a constitutive law is chosen to describe how the material in question responds to its environment. Depending on the problem of interest, this constitutive law may include such things as the relationship between applied stress and resulting strain, or between temperature gradient and thermal flux. Finally, initial and boundary values of the variables are prescribed to match the physical system of interest, and the equations are solved.

There are a variety of techniques available to achieve these solutions, and various physical phenomena often result in governing equations of a common form. Thus, many of the techniques can be applied to a variety of physical systems. Such techniques range from closed-form analytical solutions, to series expansions and other approximate forms, to computational approaches in which the continuous physical system is discretized in some way such that the integro-differential equations of the continuous domain are reduced to matrix equations that can be solved on a computer. The finite element method (FEM) is one such discretization technique.

The FEM is well-developed and it is quite naturally suited to interfacing with smaller-scale (*i.e.* atomistics and dislocation dynamics) modeling techniques. The FEM formulation, whereby the continuous physical system is represented by a grid of discrete nodes, leads to a computational scheme that is very close in its structure to that found in the methods at smaller length scales.

The FEM is a well-established technique for solving continuum models of complex engineering systems. Several commercial packages exist that solve linear and non-linear problems, static and dynamics problems, solid mechanics, heat and mass transfer, and more. These packages often have elaborate graphical user interfaces for pre- and post- processing and multiple modules and subroutines. The software is extremely useful for the solution of classical continuum problems, and their use is recommended for such purposes[1].

The FEM serves as the basis for the QC method. For this reason, this part of the tutorial is an attempt to shed light on the basic steps in formulating and implementing the FEM. Initially, we will focus on static, linear elastic problems undergoing small strains, and then generalize to the static, hyper-elasticity formulation, which is the form the FEM takes in the QC implementation.

2 Formal Description of the Continuum Mechanics Problem

By now, you will have been through the continuum mechanics part of the tutorial. However, to clarify the problem we are trying to solve with the FEM, we re-state it here.

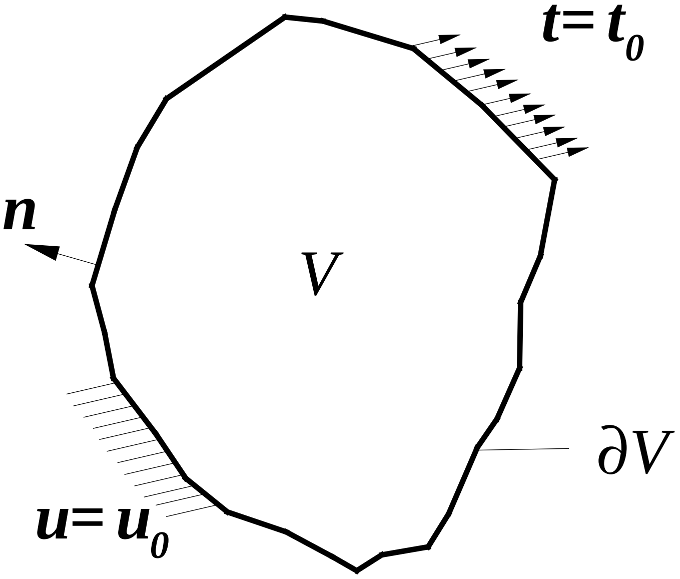

Figure 1: General Continuum Mechanics Boundary Value Problem.

The problem considered is shown in fig. 1. A Continuous body occupies volume V and has boundary ∂V. Each material point is defined by a reference coordinate $\boldsymbol{x}$. Initially, the body is stress and strain free, but the body is subjected to fixed displacements on some portion of its boundary, and prescribed loading in the form of tractions on the remainder of the boundary. Thus,

$$\boldsymbol{u} = \boldsymbol{u}_0, \quad \boldsymbol{x} \in \partial V_u \tag{1}$$

[1] There are even relatively good free packages available. See, for example [1].

$$\boldsymbol{t} = \boldsymbol{t}_0, \quad \boldsymbol{x} \in \partial V_s, \tag{2}$$

where $\boldsymbol{u}$ is the displacment and the traction, $\boldsymbol{t}$, applied to a surface with outward normal vector $\boldsymbol{n}$ is related to the stress, $\boldsymbol{P}$, at the surface of the body by

$$\boldsymbol{t} = \boldsymbol{P}^T \boldsymbol{n}, \tag{3}$$

and the superscript T indicates the transpose of a matrix. Note that $\partial V_u + \partial V_s = \partial V$, and free surfaces are therefore part of ∂V_s with $\boldsymbol{t}_0 = 0$.

Stresses and strains arise in the body due to these boundary conditions, leading to a displacement field $\boldsymbol{u}$ and a deformed body described by

$$\boldsymbol{\phi} = \boldsymbol{x} + \boldsymbol{u}, \tag{4}$$

where $\boldsymbol{\phi}$, $\boldsymbol{x}$ and $\boldsymbol{u}$ are vector fields with D components in D dimensions. The stresses which arise are dictated by the constitutive behaviour of the material, which we assume here to be hyperelastic. Thus, there is an elastic strain energy function, $W(\boldsymbol{F})$, which determines the energy per unit volume at each point in the body given the deformation gradient, $\boldsymbol{F}$, at that point. This deformation gradient is found from the displacement field as

$$\boldsymbol{F} = \boldsymbol{\nabla}\boldsymbol{\phi} = \frac{\partial \boldsymbol{\phi}^T}{\partial \boldsymbol{x}} = \boldsymbol{I} + \frac{\partial\left(\boldsymbol{u}^T\right)}{\partial \boldsymbol{x}} \tag{5}$$

where $\boldsymbol{I}$ is the identity matrix. Recall that the Lagrangian stress (first Piola-Kirchoff stress), $\boldsymbol{P}$, and the Lagrangian tangent stiffness tensor, $\boldsymbol{C}$ are obtained from this strain energy density as

$$\boldsymbol{P} = \frac{\partial W}{\partial \boldsymbol{F}} \tag{6}$$

$$\boldsymbol{C} = \frac{\partial^2 W}{\partial \boldsymbol{F} \partial \boldsymbol{F}}. \tag{7}$$

The goal is to "solve" this problem, in the sense that we seek the equilibrium displacement field $\boldsymbol{u}$ everywhere in the body given the prescribed boundary conditions and the material properties. Once we have $\boldsymbol{u}$, the stress $\boldsymbol{P}$ follows from eqn. (6).

To obtain this solution, we will use the principal of minimum potential energy. The energy, ψ_C, of the continuous system (body plus external loads) is [2]

$$\psi_C = \int_V W(\boldsymbol{x}) dV - \int_{\partial V_s} \boldsymbol{t}_0^T \boldsymbol{u} dA, \tag{8}$$

Our goal is to find a displacement field $\boldsymbol{u}$ which, for a given set of boundary conditions from eqns. (1) and (2), minimizes this energy functional.

[2] For simplicity, we have neglected the possibility of body forces in this derivation.

2.1 Linear Elastic Finite Elements

In the QC method, the strain energy functional, W, will be nonlinear and deformations can, in general, be large. Thus, finite strain hyper-elasticity is the FEM formulation that is required. It is instructive, however, to first consider the simplified case of linear elastic deformation and small strains, so as to remove certain distractions until later in the presentation. Thus, we will now simplify the model to the linear elastic, small strain case. In section 5, we will return to the fully general form used in QC.

We assume that the strain energy takes the form

$$W = \frac{1}{2} c_{ijkl} \epsilon_{ij} \epsilon_{kl} \tag{9}$$

where $\boldsymbol{\epsilon}$ is the small strain tensor defined as

$$\boldsymbol{\epsilon} = sym \boldsymbol{\nabla} \boldsymbol{u}. \tag{10}$$

Because the deformation is assumed to be small, the difference between the reference and deformed configuration is negligible and we needn't distinguish between the Lagrangian and Eulerian measures of stress. Indeed we can show that

$$\begin{aligned} P_{ij} &= \sigma_{ij} = c_{ijkl} \epsilon_{kl} & (11) \\ C_{ijkl} &= c_{ijkl}, & (12) \end{aligned}$$

where $\boldsymbol{\sigma}$ is the Cauchy stress.

Because the matrices $\boldsymbol{\sigma}$ and $\boldsymbol{\epsilon}$ are symmetric, it is convenient to represent them in a more compact vector form as follows

$$\boldsymbol{\sigma} = \begin{bmatrix} \sigma_{xx} \\ \sigma_{yy} \\ \sigma_{zz} \\ \sigma_{yz} \\ \sigma_{xz} \\ \sigma_{xy} \end{bmatrix} \quad \boldsymbol{\epsilon} = \begin{bmatrix} \epsilon_{xx} \\ \epsilon_{yy} \\ \epsilon_{zz} \\ \gamma_{yz} \\ \gamma_{xz} \\ \gamma_{xy} \end{bmatrix}, \tag{13}$$

where $\gamma_{xy} = 2\epsilon_{xy}$. This form will be useful later in the derivation of the FEM equations. When the vector forms of stress and strain are used, the linear elastic constitutive law becomes

$$\begin{aligned} W &= \frac{1}{2} \boldsymbol{\epsilon}^T \boldsymbol{D} \boldsymbol{\epsilon}, & (14) \\ \boldsymbol{\sigma} &= \boldsymbol{D} \boldsymbol{\epsilon}, & (15) \end{aligned}$$

where $\boldsymbol{D}$ is the second order (6×6) elastic stiffness matrix containing appropriate entries obtained from the tensor $\boldsymbol{c}$.

Using these simplifications, the energy functional for the linear elastic model becomes

$$\psi_C = \int_V \frac{1}{2}\boldsymbol{\epsilon}^T\boldsymbol{\sigma}dV - \int_{\partial V_s} \boldsymbol{u}^T\boldsymbol{t}_0 dA, \tag{16}$$

3 Discretization

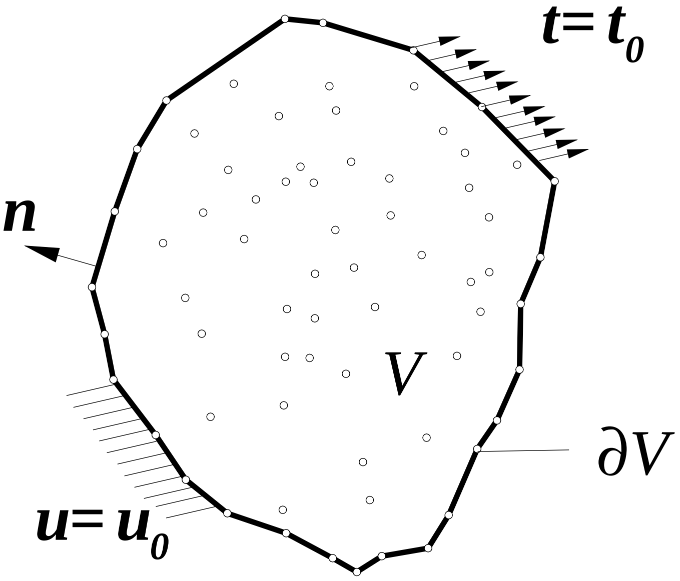

Figure 2: Discretized general continuum mechanics boundary value problem.

Displacement fields which minimize eqn. (16) are difficult to obtain for complex geometries or material behaviour if we work in the continuous domain. However, we can replace the continuous field $\boldsymbol{u}$ with a discrete representation by selecting a set of nodes throughout the body and solving explicitly for only the displacements at these nodes. The displacement fields away from the nodes will be determined by a suitable interpolation from the nodal values. The selection of nodes is completely arbitrary, and can be a non-uniform distribution of points as illustrated in fig. 2, but of course the positions and density of the nodes will effect the accuracy of the approximate solution we ultimately obtain. For a collection of n

nodes we define two vectors, $\boldsymbol{X}$ and $\boldsymbol{U}$

$$\boldsymbol{X} = \begin{bmatrix} \boldsymbol{X}_1 \\ \boldsymbol{X}_2 \\ \vdots \\ \boldsymbol{X}_n \end{bmatrix}, \qquad \boldsymbol{U} = \begin{bmatrix} \boldsymbol{U}_1 \\ \boldsymbol{U}_2 \\ \vdots \\ \boldsymbol{U}_n \end{bmatrix} = \begin{bmatrix} u_1 \\ v_1 \\ w_1 \\ u_2 \\ v_2 \\ w_2 \\ \vdots \\ u_n \\ v_n \\ w_n \end{bmatrix}, \tag{17}$$

where $\boldsymbol{X}$ contains the coordinates of the nodes, and $\boldsymbol{U}$ contains their displacements, represented for node i by $\boldsymbol{X}_i$ and $\boldsymbol{U}_i$ respectively. The last equation above shows each scalar entry for $\boldsymbol{U}$ in the case of a three-dimensional problem, with u, v and w indicating the x, y and z components of the each node's displacement vector.

Between the nodes, an appropriate set of interpolation functions (often called *shape functions*) can be used to interpolate values from the nodes to any point in the body. An expression for the displacement field everywhere in the body can then be written as:

$$\boldsymbol{u}(\boldsymbol{x}) = \boldsymbol{N}\boldsymbol{U} = \begin{bmatrix} \boldsymbol{N}_1(\boldsymbol{x}) & \boldsymbol{N}_2(\boldsymbol{x}) & \ldots & \boldsymbol{N}_n(\boldsymbol{x}) \end{bmatrix} \begin{bmatrix} \boldsymbol{U}_1 \\ \boldsymbol{U}_2 \\ \vdots \\ \boldsymbol{U}_n \end{bmatrix}, \tag{18}$$

where the matrix $\boldsymbol{N}$ contains the chosen shape functions. Normally, the same shape function is used for each degree of freedom, and it is not physically sensible to use information from one degree of freedom to interpolate the other. Thus, for the case of a three-dimensional problem, eqn. (18) becomes

$$\boldsymbol{u}(\boldsymbol{x}) = \begin{bmatrix} u(\boldsymbol{x}) \\ v(\boldsymbol{x}) \\ w(\boldsymbol{x}) \end{bmatrix} = \begin{bmatrix} N_1 & 0 & 0 & N_2 & 0 & 0 & & N_n & 0 & 0 \\ 0 & N_1 & 0 & 0 & N_2 & 0 & \ldots & 0 & N_n & 0 \\ 0 & 0 & N_1 & 0 & 0 & N_2 & & 0 & 0 & N_n \end{bmatrix} \begin{bmatrix} u_1 \\ v_1 \\ w_1 \\ u_2 \\ v_2 \\ w_2 \\ \vdots \\ u_n \\ v_n \\ w_n \end{bmatrix}, \tag{19}$$

where $N_i(\boldsymbol{x})$ is the shape function associated with node i. It should be emphasized that the shape functions are all functions of position, as are the interpolated fields, $\boldsymbol{u}(\boldsymbol{x})$. The vectors of nodal displacements and positions, $\boldsymbol{U}$ and $\boldsymbol{X}$, are not functions of position however, as they are fixed by the selection of the nodal points. The vector $\boldsymbol{U}$ is the set of unknowns for which we are solving.

> *The isoparametric formulation.* It is often convenient to introduce a change of coordinates, and write both $\boldsymbol{u}$ and $\boldsymbol{x}$ as interpolated fields of a new coordinate variable $\boldsymbol{\xi}$. In such a formulation, the shape functions are explicitly functions of $\boldsymbol{\xi}$ and both $\boldsymbol{u}(\boldsymbol{\xi})$ and $\boldsymbol{x}(\boldsymbol{\xi})$ are interpolated using equations analogous to eqn. (18). Usually, the shape functions for $\boldsymbol{x}$ and $\boldsymbol{u}$ are chosen to be the same, and hence the formulation is referred to as *isoparametric*. While this formulation offers significant efficiencies in terms of assembling the final equations, there are no conceptual differences between the formulation presented here, in which we continue to work in the physical coordinate system $\boldsymbol{x}$, and the isoparametric one. For simplicity, the details of the isoparametric formulation are left to section 5.

The strain matrix at any point in the body is obtained by a suitable linear operator (*cf.* eqn. (10)) on the displacement fields

$$\boldsymbol{\epsilon} = \boldsymbol{S}\boldsymbol{u} = \boldsymbol{S}\boldsymbol{N}\boldsymbol{U} = \boldsymbol{B}\boldsymbol{U}, \tag{20}$$

where we have defined the matrix

$$\boldsymbol{B} = \boldsymbol{S}\boldsymbol{N}. \tag{21}$$

Finally, the stresses in a linear elastic material are found from eqn. (15) to be

$$\boldsymbol{\sigma} = \boldsymbol{D}\boldsymbol{\epsilon} = \boldsymbol{D}\boldsymbol{B}\boldsymbol{U}. \tag{22}$$

We are now at a point where, given a vector of nodal displacements, we can obtain an interpolated field of the displacements, strains and stresses everywhere in the body. Making use of these interpolated fields in eqn. (16) will allow us to solve for the nodal displacements.

3.1 Solution of the Discretized System

Starting from the total energy of eqn. (16), we make use of the discretized displacement to write an approximate total energy as

$$\psi = \int_V \frac{1}{2}\boldsymbol{U}^T\boldsymbol{B}^T\boldsymbol{D}\boldsymbol{B}\boldsymbol{U}dV - \int_{\partial V_s} \boldsymbol{U}^T\boldsymbol{N}^T\boldsymbol{t}_0 dA, \tag{23}$$

where we have made use of eqns. (18), (20) and (22). We seek the nodal displacements such that ψ is minimized. Minimization of a function requires that derivatives with respect to the solution variables are zero, *i.e.*,

$$\frac{\partial\psi}{\partial\boldsymbol{U}} = \left(\int_V \boldsymbol{B}^T\boldsymbol{D}\boldsymbol{B}dV\right)\boldsymbol{U} - \int_{\partial V_s} \boldsymbol{N}^T\boldsymbol{t}_0 dA = 0 \tag{24}$$

where we have moved $\boldsymbol{U}$ outside of the integration because it is not a function of position. Close examination of this equation shows that it is of the form

$$\boldsymbol{K}\boldsymbol{U} - \boldsymbol{f} = 0, \tag{25}$$

where $\boldsymbol{f}$ is the *applied force vector* of length Dn (D is the dimensionality of the problem, n is the number of nodes) and $\boldsymbol{K}$ is the *stiffness matrix* of dimension $Dn \times Dn$. In this equation, we have defined

$$\boldsymbol{f} = \int_{\partial V_s} \boldsymbol{N}^T \boldsymbol{t}_0 dA \tag{26}$$

$$\boldsymbol{K} = \int_V \boldsymbol{B}^T \boldsymbol{D}\boldsymbol{B} dV. \tag{27}$$

Thus, a set of nodal displacements that satisfies the required equilibrium conditions is simply

$$\boldsymbol{U} = \boldsymbol{K}^{-1}\boldsymbol{f}, \tag{28}$$

and the approximate solution of the original boundary problem is found from the interpolated fields, namely eqns. (18), (20) and (22).

3.1.1 Comments

Notice that we have not yet introduced the concept of an "element", something normally considered the essential building block of the FEM. In fact, the "elements" are hidden in the appropriate choice of the shape functions, $\boldsymbol{N}$, but this formulation is not limited to the conventional finite element approach. Recent developments of so-called "meshless methods" (see, for example, [2] and [3]) begin from a formulation similar to this, and make use of shape functions that do not require the explicit choice of elements that will follow in the next section. There are pros and cons to both the meshless and meshed methods, but the meshed methods like FEM are certainly more widely used and developed.

The devil, as they say, is in the details. In this case, the devil is in choosing shape functions such that the evaluation of the force vector and stiffness matrix is a computationally efficient, but still physically accurate, endeavor. In the next section, some of these details are elucidated.

4 Implementation

In this section, the details of implementing a computationally efficient scheme for the solution of eqn. (28) will be discussed. The presentation will be general, but it is often difficult to understand the details in a fully generalized context. Thus, to aid in understanding, a simple one-dimensional example will be developed fully during the discussion, with some additional information provided as a starting point if the participant chooses to explore more complex problems at a later date.

4.1 Choosing Shape Functions and Elements

Solution of eqn. (28) requires the computation of the stiffness matrix, $\boldsymbol{K}$, which in general involves $Dn \times Dn$ integrations over the entire domain V. Without some consideration of ways to simplify this calculation, it is easy to see that the computational demand could quickly become much too great.

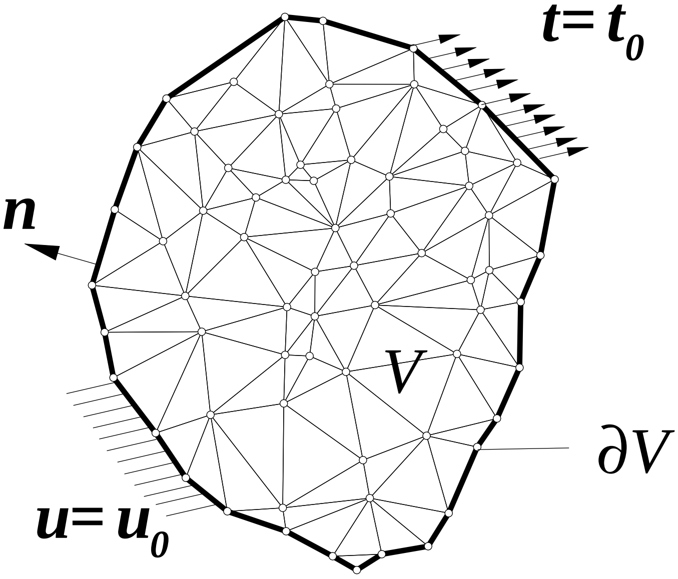

Figure 3: Elements connecting the nodes to form a finite element mesh.

In the FEM, this simplification is achieved by connecting the nodes of the discretized problem by elements (shown schematically in fig. 3), and defining the shape functions with respect to these elements. Specifically, the shape functions are given the following 3 attributes:

- *Compact support.* N_i is defined to be identically zero in any element not touching node i.

- *The delta function.* N_i must satisfy:

$$N_i(\boldsymbol{X}_j) = \delta_{ij} = \begin{cases} 1, & \text{when } i = j \\ 0, & \text{when } i \neq j \end{cases} . \tag{29}$$

This ensures that the value of the interpolated displacement field at the position of node i is equal to the nodal value, $\boldsymbol{u}(\boldsymbol{x}_i) = \boldsymbol{U}_i$.

- *The interpolation property.* For the special case when the displacements are equal at every node in the mesh, the interpolated field should be exactly uniform. To guarantee this behaviour, the shape functions must satisfy

$$\sum_{i=1}^{n} N_i(\boldsymbol{x}) = 1 \tag{30}$$

for all $\boldsymbol{x} \in V$.

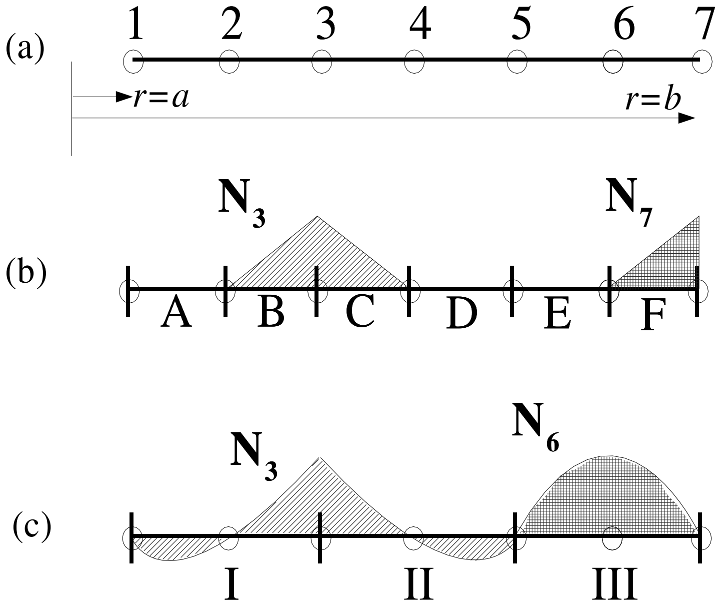

Figure 4: Shape functions for a one-dimensional domain. (a) Discretized domain. (b) Linear Elements A-F. (c) Quadratic Elements I-III.

As a simple example, consider fig. 4. The one-dimensional domain from $r = a$ to $r = b$ is represented by 7 nodes as shown in (a). In fig. 4(b), these nodes are connected by 6 elements, and the shape functions for nodes 3 and 7 are shown. It is clear that the interpolated displacement field will only depend on the position of node 3 within elements B and C. Conversely, node 3 is only directly affected by the fields within these elements. These elements are described as *linear*, since they linearly interpolate the displacements between each pair of nodes.

In fig. 4(c), an alternative choice of elements in shown, where each element includes 3 nodes along the domain. Shape functions for nodes 3 and 6 are shown, and are parabolic within each element. Notice that the shape functions for node 3 are identically zero at all other nodes, including the nodes internal to elements I and II. These elements are described as *quadratic*, since they interpolate the displacements within each element with quadratic functions.

Restricting the shape functions to have compact support and to satisfy eqn. (29) leads to an important simplification in the evaluation of integrals like the stiffness matrix. First, it allows this integration to be done on an element-by-element basis, with the important property that the integral within a given element depends only on the displacements of the nodes which make up that element. Second, it ensures that entry (i, j) in the global stiffness matrix K_{ij} will only be non-zero if nodes i and j are connected through an element. This allows for the definition of an *elemental* stiffness matrix (as will be discussed next) and further simplifies the calculation of $\boldsymbol{K}$.

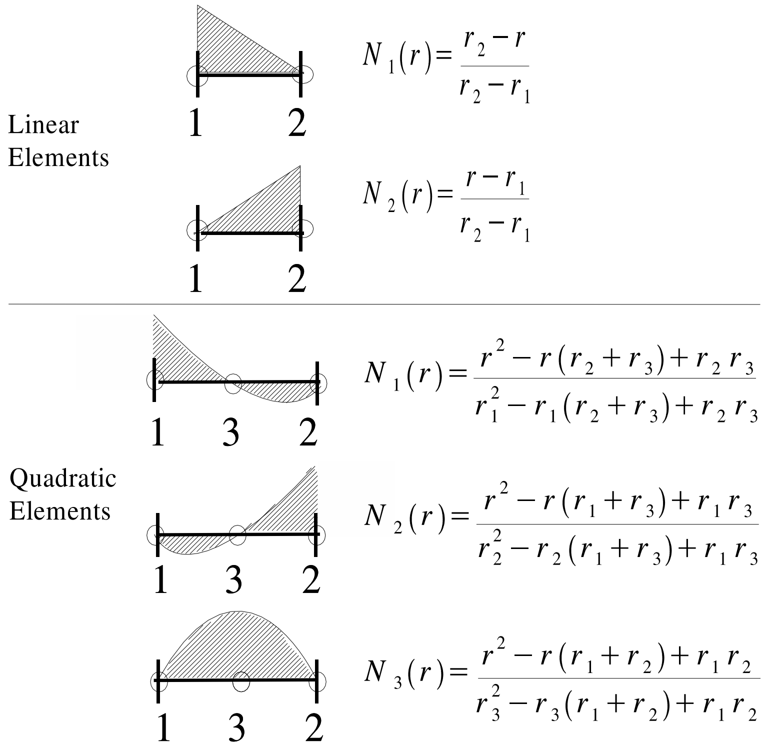
Linear Elements
1 2
$N_1(r) = \frac{r_2 - r}{r_2 - r_1}$
1 2
$N_2(r) = \frac{r - r_1}{r_2 - r_1}$
Quadratic Elements
1 3 2
$N_1(r) = \frac{r^2 - r(r_2 + r_3) + r_2 r_3}{r_1^2 - r_1(r_2 + r_3) + r_2 r_3}$
1 3 2
$N_2(r) = \frac{r^2 - r(r_1 + r_3) + r_1 r_3}{r_2^2 - r_2(r_1 + r_3) + r_1 r_3}$
1 3 2
$N_3(r) = \frac{r^2 - r(r_1 + r_2) + r_1 r_2}{r_3^2 - r_3(r_1 + r_2) + r_1 r_2}$

Figure 5: Linear and Quadratic Elemental Shape functions for a one-dimensional domain.

Shape functions within a single element for the one-dimensional case are shown in fig. 5 along with their functional form. Reference [4] contains a collection of shape functions for 1, 2 and 3 dimensional continuum problems.

An important point about standard FEM formulations is that the shape functions provide continuous interpolations of the primary variables (*i.e.* the displacements) but *derivatives*

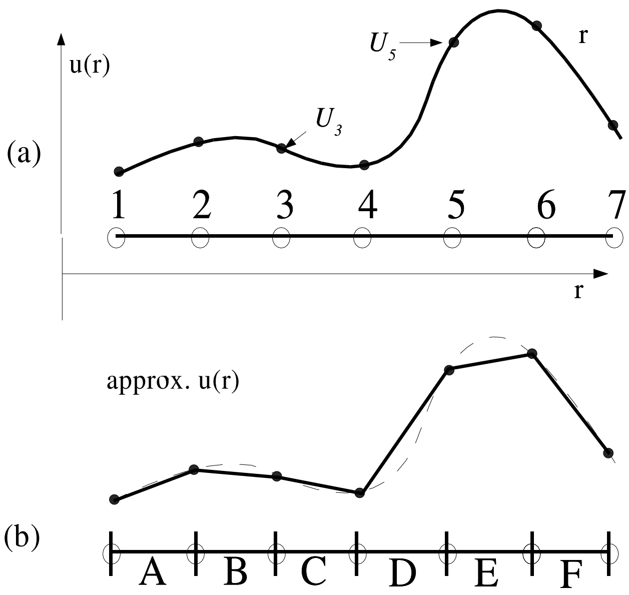

Figure 6: 1D example of interpolation with linear elements. In (a), the exact function is shown, whereas (b) shows the interpolated function given the exact values U_i at each node.

are only continuous within each element. This means that quantities like stress and strain will not be continuous from one element to the next. This is illustrated in 1D in fig. 6. In (a), a displacement field $u(r)$ is shown, with the nodal values U_i as indicated. Using linear elements leads to the interpolated field shown in (b). It is clear that the slopes of the piecewise linear segments are not continuous across the element boundaries.

The example of fig. 6 can sometimes be confusing. In our problem, we do not know the exact field $u(r)$. Instead we are devising a method to solve for the nodal values U_i, after which we obtain an approximation for $u(r)$ throughout the domain. There is no guarantee that the nodal values obtained from our solution will satify $U_i = u(r_i)$ where $u(r_i)$ is the exact (and unknown) displacement at the position of node i, as fig. 6 suggests.

4.2 Elemental Quantities

Without any loss of generality, the integrals of eqn. (27) can be broken down into sums of integrals over the individual elements. We can therefore write the global force vector and stiffness matrix as

$$\boldsymbol{f} = \sum_{elements} \int_{\partial V_s^e} \boldsymbol{N}^T \boldsymbol{t}_0 dA \tag{31}$$

$$\boldsymbol{K} = \sum_{elements} \int_{V^e} \boldsymbol{B}^T \boldsymbol{D} \boldsymbol{B} dV, \tag{32}$$

where V^e is the volume of element e. Before choosing the appropriate shape functions, such a change from a single integral over the entire body to a sum of integrals over smaller regions provides no computational benefits, but the suitable choice of shape functions with compact support makes it possible to achieve a real gain in computational efficiency by an element-by-element summation.

The stresses, strains and tractions in a specific element, by virtue of the compact support of the shape functions, depend only on the displacements of nodes directly connected to that element. Therefore, we can define an *elemental* stiffness matrix and force vector, compute these quantities, and then add them together in the correct way to assemble the global quantities.

Thus, we define

$$\boldsymbol{f}^e = \int_{\partial V_s^e} (\boldsymbol{N}^e)^T \boldsymbol{t}^e dA \tag{33}$$

$$\boldsymbol{K}^e = \int_{V^e} (\boldsymbol{B}^e)^T \boldsymbol{D}^e \boldsymbol{B}^e dV, \tag{34}$$

where the superscript e refers to the properties and quantities of a specific element and the total stiffness matrix and force vector become

$$\boldsymbol{K} = \sum_{elements} \boldsymbol{K}^e \tag{35}$$

$$\boldsymbol{f} = \sum_{elements} \boldsymbol{f}^e. \tag{36}$$

Starting from a displacement vector, $\boldsymbol{U}^e$ that contains only the displacements of the nodes directly associated with the element, we find the displacement, strain and stress within each element from equations analogous to the global equations presented earlier:

$$\boldsymbol{u}(\boldsymbol{x}) = \boldsymbol{N}^e(\boldsymbol{x})\boldsymbol{U}^e \tag{37}$$

$$\boldsymbol{\epsilon}(\boldsymbol{x}) = \boldsymbol{S}\boldsymbol{u} = \boldsymbol{B}^e\boldsymbol{U}^e \tag{38}$$

$$\boldsymbol{\sigma}(\boldsymbol{x}) = \boldsymbol{D}^e\boldsymbol{\epsilon}. \tag{39}$$

where $\boldsymbol{x}$ is now taken to refer only to positions inside the element under consideration. For example, consider the 1D mesh of linear elements in fig. 4(b). The global displacment vector $\boldsymbol{U}$ contains seven components, one for each node. There will be a total of six elemental displacement vectors, each with two components. For example $\boldsymbol{u}^e$ for the element labelled B would contain the displacements of the nodes numbered 2 and 3 in the global system. Advantages in the use of the elemental quantities come mainly in efficiencies that can be gained when coding the FEM. Also, we shall see in the discussion of 2D and 3D meshes later that the elemental quantities provide a more efficient means of numerical integration.

Note the natural capacity of the FEM to include non-uniform material properties in a body. Each element can have a different relationship between stress and strain simply by assigning different elemental material stiffness matrices, $\boldsymbol{D}^e$.

Clearly, there is a straightforward correspondence between the elemental vectors $\boldsymbol{U}^e$ and $\boldsymbol{X}^e$ and the global vectors $\boldsymbol{U}$ and $\boldsymbol{X}$ through the global numbering of the nodes. We shall see how the element-by-element approach also allows for efficient assembly of global quantities in section 4.5.

4.3 A Simple Example: Axi-Symmetric, Plane Strain Elasticity

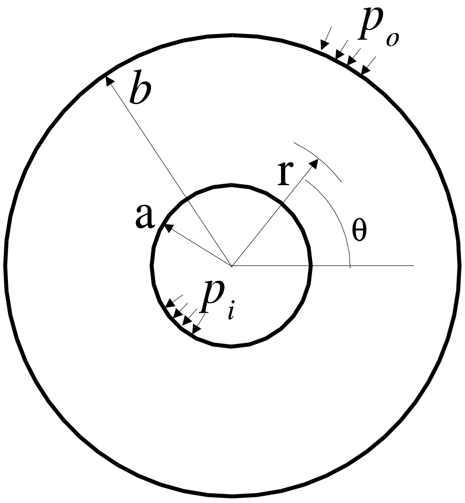

Figure 7: Pressurized cylinder.

The problem of a long cylinder, subjected to both internal and external pressure (see fig. 7), is one that can be described by one variable: the radial component of displacement. The symmetry of the cylinder is such that material points only move along radial lines. If the cylinder is long along its axis, and the ends constrained, there is also no strain in the axial direction. Further, the solution to this problem can be found analytically in many elementary mechanics texts (for example [5, 6]). Here, we develop the matrices $\boldsymbol{S}$, $\boldsymbol{B}^e$ and $\boldsymbol{D}^e$ for the simple case of the plane strain, axi-symmetric elastic cylinder using linear elements. A 1D FEM program is provided with this tutorial that will perform elastic (and plastic) calculations on this domain. Details of $\boldsymbol{S}$ and $\boldsymbol{D}^e$ for other important systems can be found in references such as [4, 7, 8].

For axi-symmetric plane strain, there is only one non-zero displacement, the radial displacement u_r. Within each linear element, there are 2 nodes (see fig. 4), and thus

$$u_r(r) = \left[\begin{array}{cc} N_1(r) & N_2(r) \end{array} \right] \left[\begin{array}{c} U_1 \\ U_2 \end{array} \right] = U_1 N_1 + U_2 N_2. \tag{40}$$

As well, only 2 components of the strain are non-zero

$$\epsilon_{rr} = \frac{\partial u_r}{\partial r} \tag{41}$$

$$\epsilon_{\theta\theta} = \frac{u_r}{r} \tag{42}$$

and so we obtain $\boldsymbol{S}$ from

$$\left[\begin{array}{c} \epsilon_{rr} \\ \epsilon_{\theta\theta} \end{array} \right] = \left[\begin{array}{c} \frac{\partial}{\partial r} \\ \frac{1}{r} \end{array} \right] u_r = \boldsymbol{S} u_r. \tag{43}$$

The matrix $\boldsymbol{B}^e$ is then

$$\boldsymbol{B}^e = \boldsymbol{S}\boldsymbol{N}^e = \left[\begin{array}{cc} \frac{\partial N_1}{\partial r} & \frac{\partial N_2}{\partial r} \\ \frac{N_1}{r} & \frac{N_2}{r} \end{array} \right] = \left[\begin{array}{cc} \frac{-1}{r_2 - r_1} & \frac{+1}{r_2 - r_1} \\ \frac{r_2 - r}{r(r_2 - r_1)} & \frac{r - r_1}{r(r_2 - r_1)} \end{array} \right], \tag{44}$$

where we have used the results of fig. 5. Finally, the material stiffness matrix comes from the linear elastic relations between stress and strain. In this case, with only two non-zero strains we have

$$\sigma_{rr} = \frac{\tilde{E}}{1 - \tilde{\nu}^2} (\epsilon_{rr} + \tilde{\nu} \epsilon_{\theta\theta}) \tag{45}$$

$$\sigma_{\theta\theta} = \frac{\tilde{E}}{1 - \tilde{\nu}^2} (\epsilon_{\theta\theta} + \tilde{\nu} \epsilon_{rr}), \tag{46}$$

where

$$\tilde{E} = E/(1 - \nu^2), \quad \tilde{\nu} = \nu/(1 - \nu), \tag{47}$$

E is Young's modulus and ν is Poisson's ratio. From these equations we can write

$$\boldsymbol{\sigma} = \left[\begin{array}{c} \sigma_{rr} \\ \sigma_{\theta\theta} \end{array} \right] = \boldsymbol{D}^e \boldsymbol{\epsilon} = \frac{\tilde{E}}{1 - \tilde{\nu}^2} \left[\begin{array}{cc} 1 & \tilde{\nu} \\ \tilde{\nu} & 1 \end{array} \right] \left[\begin{array}{c} \epsilon_{rr} \\ \epsilon_{\theta\theta} \end{array} \right]. \tag{48}$$

Note that the axial stress, σ_{zz} is also non-zero, but it does not enter into the calculations directly[3] and can be left out of the formulation. It can be obtained, if necessary, as a post-processing step from the final solution using

$$\sigma_{zz} = \nu \left(\sigma_{rr} + \sigma_{\theta\theta} \right). \tag{49}$$

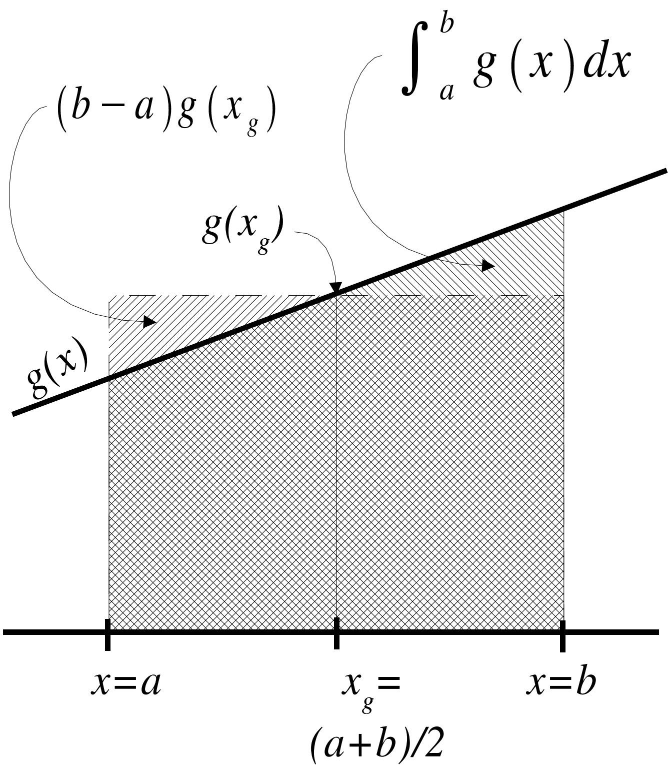

Figure 8: Gaussian quadrature of a linear function.

4.4 Numerical Integration

The solution procedure outlined in the previous section requires the rapid and accurate evaluation of an integral over each element for every evaluation of the quantities ψ, $\boldsymbol{f}^e$ and $\boldsymbol{K}^e$. An extremely efficient numerical scheme that retains good accuracy and is well-suited to the FEM is *Gaussian quadrature.*

Gaussian quadrature allows for the integration of a function by the evaluation of that function at only a small number of points. Formally, the volume integral of a function $g(\boldsymbol{x})$ becomes

$$\int_{V^e} g(\boldsymbol{x})dV = \sum_{g=1}^{n_{gauss}} w_g g(\boldsymbol{x}_g), \tag{50}$$

where $\boldsymbol{x}_g$ are a set of discrete Gauss points and w_g are the corresponding Gauss weights. Polynomial functions of a known degree can be integrated exactly with a well-defined number of Gauss points. Consider, for example, evaluation of the integral of a linear function over an interval from a to b as shown in fig. 8. This integral can be computed exactly by multiplying the length of the interval (w_g) by the function at the midpoint ($g(x_g)$) as illustrated in fig. 8. Thus, we require only one Gauss point in this case, but note that the location of the Gauss

[3] This is because $\epsilon_{zz} = 0$ and therefore the product $\epsilon_{zz}\sigma_{zz}$ in eqn. (16) does not contribute to the integral.

point must be the center of the interval for the integral to be accurate.

The integrands of eqns. (16), (33) and (34) are not of known polynomial order because of the generality of the function W. However, for the special case of a linear elastic material, the polynomial order of the functions to be integrated is known exactly and depends only on the choice of elements and shape functions. Hence, the required number and location of Gauss points to achieve the desired accuracy is also known. Details of the locations and weights for higher order polynomials in one, two and three dimensions can be found in references [4, 7, 8].

The important computational efficiency that is gained by using Gaussian quadrature is that quantities like $(\boldsymbol{B}^e)^T\boldsymbol{D}^e\boldsymbol{B}^e$ need only be evaluated and stored at the Gauss points, and thus they can be stored as numerical values rather than functions of position. Further, notice that $\boldsymbol{B}^e$ and $\boldsymbol{D}^e$ do not depend on the solution variable (the displacements). Therefore, they need only be evaluated once at the time that the initial mesh is defined, and can be subsequently re-used throughout the solution of multiple loading configurations. For non-linear material behaviour, this remains true of the matrix $\boldsymbol{B}^e$, although the material stiffness $\boldsymbol{D}^e$ will change with changes in displacements.

In our axi-symmetric example with linear elements, a single Gauss point at the center of each element will suffice to provide sufficient integration accuracy. In performing the integration, however, we must remember that we are really integrating around a cylinder, and thus an approximation to the area swept out by the element in one full revolution must be used. Thus we choose a single Gauss point with position and weight

$$r_g = \frac{r_1 + r_2}{2} \tag{51}$$

$$w_g = r_2 - r_1, \tag{52}$$

and the integral becomes

$$\int_{V^e} (\boldsymbol{B}^e)^T \boldsymbol{D}^e \boldsymbol{B}^e dV = 2\pi r_g w_g \left\{ (\boldsymbol{B}^e(r_g))^T \boldsymbol{D}^e \boldsymbol{B}^e(r_g) \right\}. \tag{53}$$

The integrations involved in computing the elemental force vector $\boldsymbol{f}^e$ are handled in a similar fashion, and will not be elaborated here. In essence, integrals over the surface (in 3D) or along the boundary (in 2D) are treated using shape functions compatible with the reduced dimensionality compared to the integral over the body. It is worth commenting, however, on the physical interpretation of what the integration of $\boldsymbol{f}^e$ effectively accomplishes. Recall that $\boldsymbol{t}^e$ is a general traction applied over a surface of an element. Schematically, it may look something like fig. 9(a), applied to either a linear (3-node) or a quadratic (6-node) triangular element. The integration of this traction effectively computes equivalent forces to be applied directly to the nodes in the element, which are then assembled into the global force vector.

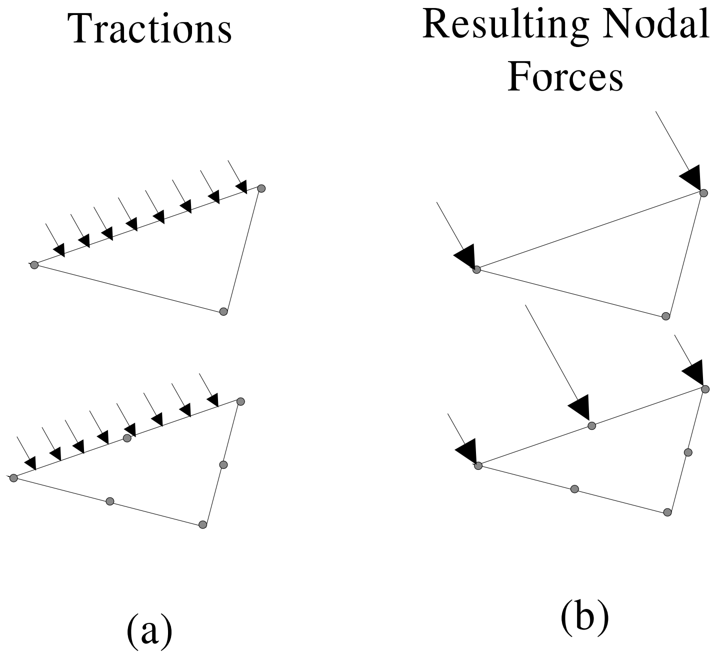

Figure 9: Schematic of the result of integrating the applied tractions. (a) Applied tractions. (b) Equivalent forces lumped onto nodes.

This is illustrated schematically in fig. 9(b). Note that the distribution of the force between the element's nodes will depend on the elemental shape functions being used.

In our simple example, there is either an internal pressure p_i or an external pressure p_o, or both. Thus the global force vector will be all zeroes but for the entries corresponding to the nodes at $r = a$ and $r = b$. These will be

$$f_a = 2\pi a p_i \tag{54}$$

$$f_b = -2\pi b p_o \tag{55}$$

where care has been taken to use the correct outward surface normal in each case and a positive pressure is conventionally taken to mean a compressive force acting on an area.

4.5 Global Stiffness Matrix Assembly

In the previous section, we computed the elemental stiffness matrix, $\boldsymbol{K}^e$. Note that this matrix will contain $Dn_e \times Dn_e$ entries, where n_e is the number of nodes per element and D is the number of dimensions of the problem. The final step before solving the matrix equation is to assemble these elemental matrices into the global stiffness matrix.

The elemental stiffness matrix was computed with reference to a local numbering scheme for the nodes, but the numbering of the nodes in the global displacement vector $\boldsymbol{U}$ must be followed in the final equation. However, a straightforward mapping can be used to insert the elemental stiffness entries into the global stiffness matrix.

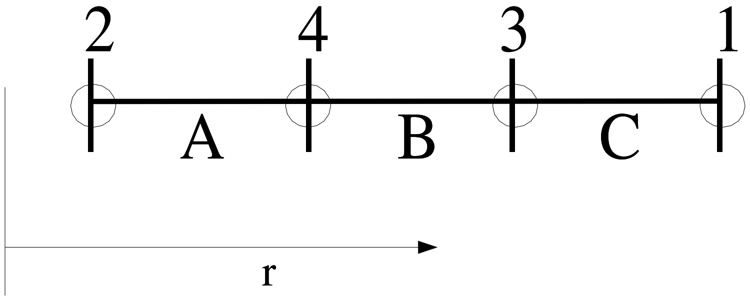

Figure 10: Simple one-dimensional mesh with three linear elements and four randomly numbered nodes.

Consider a specific mesh in our simple one-dimensional axi-symmetric formulation, containing four nodes and three elements as shown in fig. 10. The elements are labeled A, B and C, but for generality, the nodes have been numbered in a random order. Assume that we have computed elemental stiffness matrices $\boldsymbol{K}^A$, $\boldsymbol{K}^B$ and $\boldsymbol{K}^C$. For example, we have found $\boldsymbol{K}^A$ by considering nodes 2 and 4, and found values that we will denote by[4]

$$\boldsymbol{K}^A = \begin{bmatrix} K_{11}^A & K_{12}^A \\ K_{21}^A & K_{22}^A \end{bmatrix}. \tag{56}$$

Similar notation will be used for elements B and C. Note that the subscripts 1 and 2 in $\boldsymbol{K}^A$ refer to the local node numbering within the element, and globally these nodes are numbers 2 and 4. Globally, then, this matrix relates the forces and displacements of nodes 2 and 4, but contributes nothing to interactions between any other pair of nodes. We can then expand the elemental stiffness matrix to global size as follows

$$\boldsymbol{K}^{A,Global} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & K_{11}^A & 0 & K_{12}^A \\ 0 & 0 & 0 & 0 \\ 0 & K_{21}^A & 0 & K_{22}^A \end{bmatrix}, \tag{57}$$

and similarly expand $\boldsymbol{K}^B$ and $\boldsymbol{K}^C$

$$\boldsymbol{K}^{B,Global} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & K_{22}^B & K_{21}^B \\ 0 & 0 & K_{12}^B & K_{11}^B \end{bmatrix}, \quad \boldsymbol{K}^{C,Global} = \begin{bmatrix} K_{22}^C & 0 & K_{21}^C & 0 \\ 0 & 0 & 0 & 0 \\ K_{12}^C & 0 & K_{11}^C & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \tag{58}$$

[4] Often, and certainly for this linear elastic example, $\boldsymbol{K}^e$ is symmetric and therefore $K_{12}^e = K_{21}^e$

since element B connects nodes 3 and 4, while element C joints nodes 1 and 3. The global stiffness matrix, from eqn. (32) is then

$$\boldsymbol{K} = \sum_{elements} \boldsymbol{K}^{e,Global} = \boldsymbol{K}^{A,Global} + \boldsymbol{K}^{B,Global} + \boldsymbol{K}^{C,Global} \tag{59}$$

and therefore

$$\boldsymbol{K} = \begin{bmatrix} K_{22}^C & 0 & K_{21}^C & 0 \\ 0 & K_{11}^A & 0 & K_{12}^A \\ K_{12}^C & 0 & K_{22}^B + K_{11}^C & K_{21}^B \\ 0 & K_{21}^A & K_{12}^B & K_{22}^A + K_{11}^B \end{bmatrix}. \tag{60}$$

4.6 Boundary Conditions

Boundary conditions consist of two types, as illustrated in fig. 1. The first type, in which there is a prescribed traction on the boundary, has naturally led to the definition of the force vector $\boldsymbol{f}$. As we have discussed, this boundary condition will be satisfied up to the approximation involved in "lumping" the distributed traction to the nodes.

The second condition, where the displacement is prescribed on a segment of the boundary, requires the displacement of any node on this segment to be exactly the prescribed value. In this case, we do not wish to "solve" for the displacement of these nodes but rather to enforce their displacements as a constraint on the solution of eqn. (28). In the plane strain example discussed herein, this corresponds to prescribing the radial displacement at either the outer or inner surface of the cylinder, rather than specifying a pressure.

Displacement constraints can be treated directly by a simple rearrangement of the order of the scalar equations in eqn. (28). Practically speaking, this amounts to a renumbering of the nodes, although efficient FEM implementations can perform this operation through appropriate book-keeping without actual renumbering. Imagine we renumber the nodes so that all the nodes which have fixed displacement appear first in the vector $\boldsymbol{U}$. Then we can partition our matrix equation as

$$\begin{bmatrix} \boldsymbol{K}_{FF} & \boldsymbol{K}_{FN} \\ \boldsymbol{K}_{NF} & \boldsymbol{K}_{NN} \end{bmatrix} \begin{bmatrix} \boldsymbol{U}_F \\ \boldsymbol{U}_N \end{bmatrix} = \begin{bmatrix} \boldsymbol{F}_F \\ \boldsymbol{F}_N \end{bmatrix}, \tag{61}$$

where the subscript F refers to the "fixed" degrees of freedom, where the displacement is constrained and the subscript N means "not fixed". This can be written as two separate equations

$$\boldsymbol{K}_{FF}\boldsymbol{U}_F + \boldsymbol{K}_{FN}\boldsymbol{U}_N = \boldsymbol{F}_F \tag{62}$$

$$\boldsymbol{K}_{NF}\boldsymbol{U}_F + \boldsymbol{K}_{NN}\boldsymbol{U}_N = \boldsymbol{F}_N, \tag{63}$$

and the second of these can be inverted to find displacement of all nodes that are “not fixed”

$$\boldsymbol{U}_N = \boldsymbol{K}_{NN}^{-1} \left(\boldsymbol{F}_N - \boldsymbol{K}_{NF} \boldsymbol{U}_F \right). \tag{64}$$

Generally, forces will arise on the fixed nodes due to their constraint. These forces can now be computed directly from eqn. (62) if they are desired.

4.7 Matrix Inversion

The final step in the solution is the inversion of the matrix $\boldsymbol{K}$. There are a variety of matrix inversion algorithms available in the literature (see, for example, [9], [10]) and their details need not be discussed here. Quite often, the matrix $\boldsymbol{K}$ is symmetric and therefore symmetric solvers can be used. As well, significant efficiency in both storage and computational time can be gained by recognizing that the matrix is quite sparse, *i.e.* there are many zero entries as shown in eqn. (60). Algorithms to re-number the nodes such that the resulting stiffness matrix has an minimized bandwidth are also widely available. A minimized bandwidth implies that the non-zero entries are moved as close to the diagonal as possible by appropriately renumbering the nodes. This will leave the zero entries out in the top-right and lower-left corners of the matrix and allows for reduced storage requirements and faster computations. Due to time constraints, the details of such computational optimizations are not discussed in this tutorial, however we note that the QC code provided includes a bandwidth optimizer and uses an efficient symmetric matrix solver. The stiffness matrix itself is stored in such a way as to minimize the amount of wasted (zero) storage.

5 QC Finite Elements: Hyper elastic, finite deformation formulation

The foregoing discussion has focussed on the simple case of linear elasticity and small strains to clarify the essential concepts of the FEM. However, within QC the FEM is slightly more complicated due to the nonlinear material response and nonlinear (finite strain) deformation response that is required. In the remainder of the tutorial, these aspects are discussed.

5.1 Shape functions in 2D and 3D: The parent element and the isoparametric formulation.

In 2D or 3D, as illustrated in fig. 11, the shape and size of each element will differ depending on the distribution and density of the nodes. Thus, the shape functions within each element will be different in each element. For efficient implementation of the shape functions for an

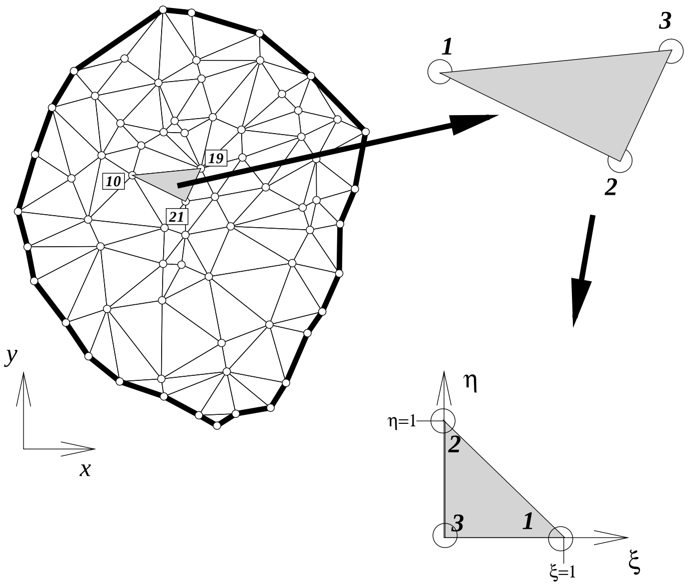

Figure 11: Elements of arbitrary size and shape are first refered to a local node numbering scheme, and then mapped to a parent element for efficient implementation.

arbitrary mesh, the concept of a *parent element* is used. In this discussion, we will focus on the simple case of 3-node triangular elements in 2D, but the procedure applies for any element type in any number of dimensions.

Consider the highlighted element in fig. 11, which is defined by nodes 10, 21 and 19. The corresponding shape functions N_{10}, N_{21} and N_{19} need to be defined within the element. In order to do so, the nodes are first referred to by a numbering scheme that is local to the element in question. The element is then mapped to the parent element by a transformation of coordinates from the physical space $\boldsymbol{x}$ to the parent space $\boldsymbol{\xi}$. This mapping is conveniently accomplished by interpolating the nodal coordinates $\boldsymbol{X}_i$ using shape functions defined in the parent element. Thus within a given element

$$\boldsymbol{x}(\boldsymbol{\xi}) = \boldsymbol{N}^e(\boldsymbol{\xi})\boldsymbol{X}^e, \tag{65}$$

where we have defined $\boldsymbol{N}^e$ and $\boldsymbol{X}^e$, the *elemental* shape function matrix and the *elemental* nodal vector. Specifically, for the example of the 3-node triangular elements in 2D these take the form

$$\boldsymbol{N}^e = \left[\begin{array}{cccccc} N_1 & 0 & N_2 & 0 & N_3 & 0 \\ 0 & N_1 & 0 & N_2 & 0 & N_3 \end{array}\right] \quad \boldsymbol{X}^e = \left[\begin{array}{c} x_1 \\ y_1 \\ x_2 \\ y_2 \\ x_3 \\ y_3 \end{array}\right], \tag{66}$$

where the numbering, $i = 1 \dots 3$, refers to the local node-numbering illustrated in fig. 11. These matrices are the 2D, isoparametric analog to the 1D example of eqn. (37). The linear shape functions for these elements are

$$\begin{aligned} N_1 &= \xi && (67) \\ N_2 &= \eta && (68) \\ N_3 &= 1 - \xi - \eta. && (69) \end{aligned}$$

Note that within the parent element, the nodal displacements can be interpolated using the same shape functions, *i.e.*,

$$\boldsymbol{u}(\boldsymbol{\xi}) = \boldsymbol{N}^e(\boldsymbol{\xi})\boldsymbol{U}^e, \tag{70}$$

and therefore this is often referred to as the *isoparametric* formulation. In principal, one could use different interpolation schemes for $\boldsymbol{x}$ and $\boldsymbol{u}$, making the scheme something other than isoparametric.

5.2 Solution Procedure: Energy Minimization

Returning to eqn. (8), prior to the introduction of the linear elastic strain energy density, we can write a more general energy for the discretized system as

$$\psi = \sum_{e=1}^{n_{elem}} \left(\int_{V^e} W[\boldsymbol{F}(\boldsymbol{x})] dV - \left[\int_{\partial V_s^e} \boldsymbol{t}_0^T \boldsymbol{N} dA \right] \boldsymbol{U} \right). \tag{71}$$

This is the hyper-elastic analogue of eqn. (23), where we have broken up the integral over the body into a sum of integrals over each element. We have introduced the approximate form of $\boldsymbol{u}$ from eqn. (18), and note that the nodal displacements $\boldsymbol{U}$ are not functions of position and therefore can come out of the last integral. There is an implicit, nonlinear, dependence of W on $\boldsymbol{U}$ through $\boldsymbol{F}$ as we shall see next. Making use of the isoparametric mapping to the parent element requires a change of variables from $\boldsymbol{x}$ to $\boldsymbol{\xi}$, so that

$$\psi = \sum_{e=1}^{n_{elem}} \left(\int_{\Omega^e} W[\boldsymbol{F}(\boldsymbol{\xi})] \mathcal{J} d\Omega - \left[\int_{\partial V_s^e} \boldsymbol{t}_0^T \boldsymbol{N} dA \right] \boldsymbol{U} \right), \tag{72}$$

where Ω^e is the volume of the parent element in the transformed space, and $\mathcal{J}$ is the Jacobian determinant of the mapping, *i.e.*,

$$\mathcal{J} = |\boldsymbol{J}|\,, \tag{73}$$

where

$$\boldsymbol{J} \equiv \frac{\partial \boldsymbol{x}^T}{\partial \boldsymbol{\xi}}. \tag{74}$$

Using the elemental form for the interpolation of $\boldsymbol{x}$ (eqn. (65)) and indicial notation for clarity[5], $\boldsymbol{J}$ is given by

$$J_{ij} = \frac{\partial x_i}{\partial \xi_j} = \frac{\partial N^e_{ik}}{\partial \xi_j} X^e_k \tag{75}$$

Note that we will, for brevity, leave the surface integral of eqn. (72) in terms of the physical coordinates, $\boldsymbol{x}$.

Our goal is to solve for the displacement vector $\boldsymbol{U}$ such that ψ is minimized. Thus, we require that the vector equation

$$\frac{\partial \psi}{\partial \boldsymbol{U}} = 0 = \sum_{e=1}^{n_{elem}} \left(\int_{\Omega^e} \frac{\partial}{\partial \boldsymbol{U}} \left(W[\boldsymbol{F}(\boldsymbol{x})] \right) \mathcal{J} d\Omega - \int_{\partial V^e_s} \boldsymbol{t}_0^T \boldsymbol{N} dA \right) \tag{76}$$

is satisfied. This provides $3n$ equations for the $3n$ unknown nodal displacements in $\boldsymbol{U}$. Note that the energy density W depends on $\boldsymbol{U}$ through the deformation gradient, $\boldsymbol{F}$. Thus we can write in indicial notation

$$\frac{\partial W}{\partial U_i} = \frac{\partial W}{\partial F_{jk}} \frac{\partial F_{jk}}{\partial U_i} = P_{jk} \frac{\partial N^e_{ji}}{\partial \xi_l} J^{-1}_{lk}\,, \tag{77}$$

where we have used eqns. (5) and (70) to obtain the result

$$F_{ij} = \delta_{ij} + \frac{\partial N_{ik}}{\partial \xi_l} J^{-1}_{lj} U_k. \tag{78}$$

Inserting these results back into eqn. (76) we have

$$0 = \sum_{e=1}^{n_{elem}} \left(\int_{\Omega^e} P_{jk} \frac{\partial N^e_{ji}}{\partial \xi_l} J^{-1}_{lk} \mathcal{J} d\Omega - \int_{\partial V^e_s} t^0_j N_{ji} dA \right) \tag{79}$$

for all $i = 1 \ldots 3n$

5.3 The Newton-Raphson Method

Because the energy functional W is generally nonlinear with respect to $\boldsymbol{U}$, it is necessary to use an iterative scheme to solve eqn. (79). One approach is the Newton-Raphson (NR) method, whereby we imagine starting from some initial guess to the displacement vector,

[5] The Appendix to this document contains a review of indicial notation.

$\boldsymbol{U}^0$. We seek to iteratively improve on this guess through incremental changes $\Delta\boldsymbol{U}^m$ that will lower the total energy of the system from $\psi(\boldsymbol{U}^m)$ to $\psi(\boldsymbol{U}^m + \Delta\boldsymbol{U}^m)$ where $\boldsymbol{U}^m$ is the mth iterant of the solution. We can find $\Delta\boldsymbol{U}^m$ by using a linear approximation to eqn. (79):

$$\left.\frac{\partial\psi}{\partial\boldsymbol{U}}\right|_{\boldsymbol{U}^{m+1}} \approx \left.\frac{\partial\psi}{\partial\boldsymbol{U}}\right|_{\boldsymbol{U}^{m}} + \left.\frac{\partial^2\psi}{\partial\boldsymbol{U}\partial\boldsymbol{U}}\right|_{\boldsymbol{U}^{m}}\left(\boldsymbol{U}^{m+1} - \boldsymbol{U}^m\right) = 0, \tag{80}$$

where the last equality comes from recognizing that we seek $\boldsymbol{U}^{m+1}$ that satisfies eqn. (79) given that $\boldsymbol{U}^m$ does not. Solving for $\boldsymbol{U}^{m+1}$

$$\boldsymbol{U}^{m+1} = \boldsymbol{U}^m + \Delta\boldsymbol{U}^m \tag{81}$$

where

$$\Delta\boldsymbol{U}^m = (\boldsymbol{K}^m)^{-1}\boldsymbol{r}^m \tag{82}$$

and $\boldsymbol{K}^m$ and $\boldsymbol{r}^m$, referred to as the *tangent stiffness matrix* and the *residual vector* respectively, are equal to

$$\boldsymbol{K}^m = \left.\frac{\partial^2\psi}{\partial\boldsymbol{U}\partial\boldsymbol{U}}\right|_{\boldsymbol{U}^{m}} \tag{83}$$

$$\boldsymbol{r}^m = \left.\frac{\partial\psi}{\partial\boldsymbol{U}}\right|_{\boldsymbol{U}^{m}}. \tag{84}$$

Note the similarity between eqn. (82) and eqn. (28). While the tangent matrix and residual must be computed with each iteration, their structure is the same as for the elastic case. Thus, the same storage and solution techniques can be used. This process can then be iterated until the norm of $\boldsymbol{r}^m$ is made suitably small.

In practice, the algorithm just described may have difficulty converging for functions that are significantly different than the linear approximation of eqn. (80). As a result, the actual approach is to use the $\Delta\boldsymbol{U}^m$ as a *direction* for the displacement increment, rather than the increment itself. Thus, each change to the displacements takes the form

$$\boldsymbol{U}^{m+1} = \boldsymbol{U}^m + \alpha\Delta\boldsymbol{U}^m \tag{85}$$

where α is chosen such that $\psi(\boldsymbol{U}^{m+1})$ is minimized along the direction of $\Delta\boldsymbol{U}^m$.

The quantities $\boldsymbol{K}$ and $\boldsymbol{r}$ required for the NR method can be computed by differentiating eqn. (72). The results of this differentiation are

$$\boldsymbol{r} = \sum_{e=1}^{n_{elem}} \boldsymbol{r}^e \tag{86}$$

$$\boldsymbol{K} = \sum_{e=1}^{n_{elem}} \boldsymbol{K}^e, \tag{87}$$

where we have defined the *elemental* residual vectors and *elemental* stiffness matrices as

$$r_i^e = \int_{\Omega^e} P_{jk} \frac{\partial N_{ji}^e}{\partial \xi_l} J_{lk}^{-1} \mathcal{J} d\Omega - \int_{\partial V_s^e} \boldsymbol{t}_j^0 N_{ji} dA \tag{88}$$

$$K_{ij}^e = \int_{\Omega^e} C_{mkrs} \frac{\partial N_{rj}^e}{\partial \xi_n} J_{ns}^{-1} \frac{\partial N_{pi}^e}{\partial \xi_l} J_{lk}^{-1} \mathcal{J} d\Omega. \tag{89}$$

These are analogous to the elemental quantities $\boldsymbol{f}^e$ and $\boldsymbol{K}^e$ given in eqns. (33) and (34) for the linear elastic system.

The above integrals are, as in the elastic case, handled using Gaussian quadrature, although in the QC formulation this is nearly a trivial statement. Because of the linear elements employed, the deformation gradient, and therefore the strain energy and its derivatives, are constant throughout a given element. Thus, the integration amounts to multiplying a constant value times the area of the element.

5.4 Generalized Plane Strain

The QC method makes use of 2D, 3-noded, triangular elements. Although the body is assumed to be 2D, the displacement field permits 3 components. This is sometimes refered to as a "generalized plane strain" formulation. In molecular dynamics simulations of a crystal, an equivalent situation would be one where periodic boundary conditions are employed in the out-of-plane direction, with the periodic length being as small as possible to fully reproduce the crystal structure.

The main effect of generalized plane strain on the details presented herein is that the interpolated phyisical coordinates, $\boldsymbol{x}$, and the interpolated displacements $\boldsymbol{u}$ have different dimensions. The interpolation of $\boldsymbol{x}$ is

$$\boldsymbol{x} = \begin{bmatrix} N_1 & 0 & N_2 & 0 & N_3 & 0 \\ 0 & N_1 & 0 & N_2 & 0 & N_3 \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ x_2 \\ y_2 \\ x_3 \\ y_3 \end{bmatrix}, \tag{90}$$

where the numbering, $i = 1 \ldots 3$, refers to the local node-numbering in the element. Because the displacements have 3 components, they are written

$$\boldsymbol{u}(\boldsymbol{x}) = \begin{bmatrix} N_1 & 0 & 0 & N_2 & 0 & 0 & N_3 & 0 & 0 \\ 0 & N_1 & 0 & 0 & N_2 & 0 & 0 & N_3 & 0 \\ 0 & 0 & N_1 & 0 & 0 & N_2 & 0 & 0 & N_3 \end{bmatrix} \begin{bmatrix} u_1 \\ v_1 \\ w_1 \\ u_2 \\ v_2 \\ w_2 \\ u_3 \\ v_3 \\ w_3 \end{bmatrix}. \tag{91}$$

The shape functions, N_i, are as given in eqns. (67)-(69).

This imposes constraints on the deformation gradient, which will assume the form

$$F_{ij} = \begin{bmatrix} 1 + \frac{\partial u}{\partial x} & \frac{\partial u}{\partial y} & 0 \\ \frac{\partial v}{\partial x} & 1 + \frac{\partial v}{\partial y} & 0 \\ \frac{\partial w}{\partial x} & \frac{\partial w}{\partial y} & 1 \end{bmatrix}. \tag{92}$$

6 Exercises

1. Consider the function $u(x) = 3x^2 + 2x + 1$ on the domain $0 \leq x \leq 3$. Using 3 nodes located at $x = \{0, 1, 3\}$ and $U_i = u(x_i)$:

   i. Find the approximate $u(x)$ using 2 linear elements.

   ii. Find the approximate $u(x)$ using 1 quadratic element.

   iii. Verify that the interpolated function is C^0 continuous across element boundaries.

2. Verify that eqn. (30) is satisfied for the shape functions shown in fig. 5.

3. Use Gaussian integration to integrate the quadratic function $g(x) = Ax^2 + Bx + C$ on the domain $x = -1 \ldots 1$.

   i. Verify that 2 Gauss points at $x_g = \pm 1/\sqrt{3}$ and $w_g = 1$ yields the exact integral.

   ii. Compute the error if $x_g = \pm 1/2$ and $w_g = 1$ are used instead.

4. Verify that eqn. (11) holds.

5. Derive eqns. (75) and (89).

A Appendix: Indicial Notation

Indicial notation is a short-hand notation which facilitates the writing and manipulation of expressions in continuum mechanics in general and elasticity theory in particular. The discussion in this section is based partly on Lai et al. [11].

A.1 Dummy indices and the summation convention

Consider the sum,

$$S = a_1x_1 + a_2x_2 + \ldots + a_nx_n.$$

We can write this using a summation symbol,

$$S = \sum_{i=1}^{n} a_ix_i = \sum_{j=1}^{n} a_jx_j = \sum_{m=1}^{n} a_mx_m.$$

Indices i, j and m are *dummy indices*, in the sense that the sum is independent of the letter used.

Einstein's Summation Convention: A simplified notation where the $\sum$ symbol is dropped and any index appearing twice in a term is taken to be a dummy index and summed over,

$$S = a_ix_i = a_jx_j = a_mx_m = a_1x_1 + a_2x_2 + \ldots + a_nx_n.$$

Examples for $n = 3$,

1. $a_ix_i = a_1x_1 + a_2x_2 + a_3x_3$
2. $a_ia_i = a_1^2 + a_2^2 + a_3^2$
3. $a_{ii} = a_{11} + a_{22} + a_{33}$
4. Vector $\mathbf{a}$ is written a_i

Scalar Product: If $\mathbf{a}$ and $\mathbf{b}$ are vectors with components a_i and b_i, we have $\mathbf{a} \cdot \mathbf{b} = a_ib_i$ and $|\mathbf{a}| = \sqrt{a_ia_i}$.

Notes:

(i) An index *cannot* appear more than twice in an expression.
The expression $a_ib_ix_i$ is meaningless. To sum over three or more terms it is necessary to write the sum explicitly $\sum_{i=1}^{n} a_ib_ix_i$.

(ii) Double sums and beyond follow as expected, e.g.

$$\begin{aligned} a_{ij}x_ix_j &= a_{11}x_1x_1 + a_{12}x_1x_2 + a_{13}x_1x_3 + \\ &\quad a_{21}x_2x_1 + a_{22}x_2x_2 + a_{23}x_2x_3 + \\ &\quad a_{31}x_3x_1 + a_{32}x_3x_2 + a_{33}x_3x_3. \end{aligned}$$

A.2 Free Indices

An index which appears only once in each term of an equation is referred to as a *free index.* A free index takes on the values $1, 2, \ldots, n$, one at a time, e.g.

$$a_{ij}x_j = b_i. \tag{93}$$

Here i is a free index and (93) corresponds to a system of n equations,

$$\begin{array}{ll} a_{11} & x_1 + a_{12}x_2 + \ldots a_{1n}x_n = b_1 \\ \vdots & \\ a_{n1} & x_1 + a_{n2}x_2 + \ldots a_{nn}x_n = b_n \end{array}$$

Notes:

(i) All terms in an expression must have the same free indices.
The expression $a_{ij}x_j = b_k$ is meaningless.

(ii) There can be as many free indices as necessary.
For example in the expression $a_{ijk}x_k$, i and j are free indices resulting in $n \times n$ expressions.

A.3 Kronecker Delta – δ_{ij}

The Kronecker Delta is defined as follows,

$$\delta_{ij} = \begin{cases} 1 & \text{if} \quad i = j \\ 0 & \text{if} \quad i \neq j \end{cases} .$$

An important property of δ_{ij} is *index substitution*,

$$a_i\delta_{ij} = a_j.$$

Proof (for $n = 3$):

$$a_i\delta_{ij} = a_1\delta_{1j} + a_2\delta_{2j} + a_3\delta_{3j} = \begin{cases} a_1 & \text{if} \quad j = 1 \\ a_2 & \text{if} \quad j = 2 \\ a_3 & \text{if} \quad j = 3 \end{cases} = a_j, \qquad \text{qed.}$$

Examples:

1. $a_{ij}\delta_{ij} = a_{ii} = a_{jj} = a_{11} + a_{22} + a_{33}$
2. $\mathbf{e}_i \cdot \mathbf{e}_j = \delta_{ij}$, if $\mathbf{e}_1$, $\mathbf{e}_2$ and $\mathbf{e}_3$ are unit vectors perpendicular to each other.
3. **Scalar product:** $\mathbf{a} \cdot \mathbf{b} = (a_i\mathbf{e}_i) \cdot (b_j\mathbf{e}_j) = a_i b_j(\mathbf{e}_i \cdot \mathbf{e}_j) = a_i b_j \delta_{ij} = a_i b_i$
4. **Vector magnitude:** $|\mathbf{a}| = \sqrt{\mathbf{a} \cdot \mathbf{a}} = \sqrt{a_i a_i}$

A.4 Permutation Symbol – e_{ijk}

The Permutation Symbol is defined as follows,

$$e_{ijk} = \begin{cases} 1 & \text{if } i,j,k \quad \text{form an even permutation of } 1,2,3 \\ -1 & \text{if } i,j,k \quad \text{form an odd permutation of } 1,2,3 \\ 0 & \text{if } i,j,k \quad \text{do not form a permutation of } 1,2,3 \end{cases} .$$

Thus,

$$\begin{aligned} e_{123} &= e_{231} = e_{312} = 1 \\ e_{321} &= e_{132} = e_{213} = -1 \\ e_{111} &= e_{112} = e_{113} = \ldots = 0. \end{aligned}$$

Examples:

1. Show $e_{ijk}\delta_{ij} = 0$.
   Proof: $e_{ijk}\delta_{ij} = e_{iik} = 0$, qed.
2. For a right-handed triad $\mathbf{e}_1$, $\mathbf{e}_2$ and $\mathbf{e}_3$, we have $\mathbf{e}_1 \times \mathbf{e}_2 = \mathbf{e}_3$,
   $\mathbf{e}_2 \times \mathbf{e}_3 = \mathbf{e}_1$, etc.
   ⇒ In indicial notation, $\mathbf{e}_i \times \mathbf{e}_j = e_{ijk}\mathbf{e}_k$
3. **Vector product:** $\mathbf{a} \times \mathbf{b} = (a_i\mathbf{e}_i) \times (b_j\mathbf{e}_j) = a_i b_j(\mathbf{e}_i \times \mathbf{e}_j) = e_{ijk} a_i b_j \mathbf{e}_k$
4. **e-$\boldsymbol{\delta}$ Identity:** $e_{ijm}e_{klm} = \delta_{ik}\delta_{jl} - \delta_{il}\delta_{jk}$

A.5 Differential Operators

Differentiation with respect to the cartesian coordinates x_i is indicated by a comma symbol in indicial notation. For example,

Gradient

$$\nabla S = \frac{\partial S}{\partial x_i} = S_{,i}$$

Divergence

$$\nabla \cdot \mathbf{V} = \frac{\partial V_1}{\partial x_1} + \frac{\partial V_2}{\partial x_2} + \frac{\partial V_3}{\partial x_3} = \frac{\partial V_i}{\partial x_i} = V_{i,i}$$

Laplacian

$$\nabla^2 S = \frac{\partial^2 S}{\partial x_1^2} + \frac{\partial^2 S}{\partial x_2^2} + \frac{\partial^2 S}{\partial x_3^2} = \frac{\partial^2 S}{\partial x_i \partial x_i} = S_{,ii}$$

References

[1] Modulef. A free, modular finite element library. Available at http://www-rocq.inria.fr/modulef/english.html.

[2] T. Belytschko, Y. Krongauz, D. Organ, M. Fleming, and P. Krysl. Meshless methods: An overview and recent developments. *Computer Methods in Applied Mechanics and Engineering*, 139:3–47, 1996.

[3] S.N. Atluri and Tulong Zhu. New concepts in meshless methods. *Int. J. Num. Meth. Engng.*, 47:537–556, 2000.

[4] *Abaqus Theory Manual Version 5.7.* HKS Inc., Pawtucket, Rhode Island, 1997.

[5] R.D. Cook and W.C. Young. *Advanced Mechanics of Materials.* Prentice Hall, Upper Saddle River, New Jersey, 2nd edition, 1999.

[6] Jacob Lubliner. *Plasticity Theory.* MacMillan, New York, 1990.

[7] J.N. Reddy. *An Introduction to the Finite Element Method.* McGraw-Hill, New York, 2nd edition, 1993.

[8] O.C. Zienkiewicz and R.L. Taylor. *The Finite Element Method*, volume I. McGraw-Hill, London, 4th edition, 1989.

[9] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in FORTRAN: The Art of Scientific Computing.* Cambridge University Press, 2nd edition, 1992.

[10] Linpack. A collection of linear algebra subroutines in FORTRAN. Available at http://www.netlib.org/linpack/.

[11] Lai WM, Rubin D, Krempl E. *Introduction to Continuum Mechanics.* Oxford: Pergamon, 1993.